Noname manuscript No.
(will be inserted by the editor)

Performance Analysis of Network Traffic Predictors in the Cloud

Bruno L. Dalmazo · João P. Vilela · Marilia Curado

Received: date / Accepted: date

Abstract Predicting the inherent traffic behaviour of a network is an essential task, which can be used for various purposes, such as monitoring and managing the network's infrastructure. However, the recent surge of dynamic environments, such as Internet of Things and Cloud Computing have hampered this task. This means that the traffic on these networks is even more complex, displaying a nonlinear behaviour with specific aperiodic characteristics during daily operation. Traditional network traffic predictors are usually based on large historical data bases which are used to train algorithms. This may not be suitable for these highly volatile environments, where the strength of the force exerted in the interaction between past and current values may change quickly with time. In light of this, a taxonomy for network traffic prediction models, including the review of state of the art, is presented here. In addition, an analysis mechanism, focused on providing a standardized approach for evaluating the best candidate predictor models for these environments, is proposed. These contributions favour the analysis of the efficacy and efficiency of network traffic prediction among several prediction models in terms of accuracy, historical dependency, running time and computational overhead. An evaluation of several prediction mechanisms is performed by assessing the Normalized Mean Square Error and Mean Absolute Percent Error of the values predicted by using traces taken from two real case studies in cloud computing.

Bruno L. Dalmazo (corresponding author)
E-mail: dalmazo@dei.uc.pt

João P. Vilela
E-mail: jpvilela@dei.uc.pt

Marilia Curado
E-mail: marilia@dei.uc.pt

CISUC, Department of Informatics Engineering
Pólo II - Pinhal de Marrocos, Coimbra
University of Coimbra, Portugal

Keywords Cloud computing · prediction models taxonomy · computational complexity · network traffic prediction analysis

1 Introduction

Cloud computing is a basis for providing benefits well beyond Information Technology (IT) cost savings. It comprises a set of service models to scale on-demand, such as Infrastructure as a Service (IaaS), Software as a Service (SaaS) or Platform as a Service (PaaS). These services can be offered to a wide range of clients through an institution called cloud provider [1].

On a daily basis, the cloud provider has to deal with a huge number of devices and virtual machines so that it can handle all the assets of its network infrastructure. It is increasingly being found that neglecting this area of management can cause irreparable economic damage to businesses and their customers [2]. In light of this, the network administrators of these cloud-supporting networks must monitor and analyse these networks so that relevant information about network traffic can be collected and used to support decision-making. After all this information has been gathered, it is possible to identify and analyse suspicious network traffic patterns. These patterns can help in planning strategies to prevent similar problems from occurring in the future [3].

Analysing network traffic is a means of facilitating the monitoring and management of computer networks. In this context, a network traffic predictor is a tool that uses accumulated statistics over time to make inferences about the future behaviour of network traffic [4]. Thus, when an abnormality is forecast, the network administrator will have time to act even before the problem arises. For this reason, network traffic prediction has been receiving a great deal of attention from the scientific community. In addition, network traffic prediction is also important in other fields such as: traffic shaping for improved Quality of Service [5], prediction of bandwidth requirements [6], conceiving more accurate simulation models [7], admission control [8], and adaptive applications [9].

1.1 Requirements for Cloud Network Traffic Prediction

Characterizing and monitoring network traffic is becoming a more complex task, particularly with the surge in traffic arising from the huge number of individuals and machines that are permanently connected to the Internet. The challenge is even greater in cloud computing because its traffic is apt to undergo sudden changes [10] [11], and the elastic and scalable nature of cloud environments may be easily confused with traffic anomalies [12].

Network traffic prediction requires an accurate model, which can capture the statistical features of the real condition of the traffic. However, the degree of accuracy needed for predicting the network traffic deteriorates quickly as the historic interval increases [13], and thus calls for a model that is suited to the dynamics of cloud traffic. These observations suggest that this kind of network traffic is different from traditional IT network traffic, and requires special attention in some aspects. This implies that the cloud network traffic predictor must not only have

traditional features such as accuracy, but also some other essential requirements for the cloud environment. These include the following:

1. **Low historical dependency**: Models for predicting data traffic usually take into account large amounts of historical data. However, employing these models is not the most suitable approach for carrying out the traffic prediction of cloud computing systems, because the network baseline does not have the same periodic behaviour as traditional networks [14];
2. **Low complexity**: Several prediction models have been proposed in the literature that can be employed in various network traffic environments. However, most of them are not appropriate for dealing with a large amount of information in a short period of time and keeping a low computational complexity [15] [16];
3. **Online prediction**: Offline predictions are usually made to test the predictor efficiency under specific conditions and in the training phase. However, an effective monitoring of cloud computing networks must be constantly carried out to address detection issues as they occur or even beforehand. This statement applies to online traffic prediction mechanisms [6].

In short, from this set of requirements it is possible to select a group of candidate predictors that is suitable for cloud network traffic prediction. On the basis of this scenario, we consider that the choice of an ideal predictor model for cloud network traffic will involve a tradeoff between prediction error, historical data dependence, computational costs, and timely response.

1.2 Open Issues and Contributions

Most research studies on network traffic prediction have focused on classical methods that rely heavily on historical data such as time series, neural networks and machine learning. However, there is still no consensus among the research community about which model is best suited for cloud network traffic prediction.

There is usually only a fine line separating concerns about high accuracy from computational costs, and sometimes it is difficult to determine where the border line should be drawn. The challenge in cloud network traffic prediction is to minimize the computational cost as much as possible, while keeping acceptable levels of accuracy. This is not a trivial issue, since most of the current prediction models are not able to keep a low computational complexity while dealing with a high degree of workload information in a short period of time. The main problem with these models is the increasing computational overhead in accordance with the size of the input data. For instance, approaches based on large historical dependency could obtain a slightly better degree of accuracy than other models based on short historical dependency. However, when compared with local analysis approaches, they have a much higher computational complexity to compute the predictions [15].

To address these issues, we have provided a taxonomy for network traffic prediction models, followed by a detailed review of the state of the art. This lays the foundation for selecting a set of prediction models that is suitable for the cloud environment. Thus, we propose an analysis mechanism through which a standardized approach is adopted for evaluating the candidate predictor models using real traces. By employing this mechanism, it is possible to carry out a wide-ranging

study of the performance of several predictor models in a cloud computing environment. This article extends our previous works ([17] [18]) which proposed a methodology based on a sliding window that defines the amount of traffic that is considered for traffic prediction, and thus ensures that it is suitable for dynamic environments such as cloud computing.

The remainder of the paper is organized as follows. Section 2 presents a review of state of the art and a taxonomy with the most prominent related studies on network traffic prediction models. Section 3 details the prediction methods deemed applicable to the highly dynamic cloud computing environment requirements. Section 4 describes the methodology used to evaluate the prediction mechanisms, whilst Section 5 presents the evaluation and discusses the results. Section 6 concludes with some final remarks and potential directions for future research.

2 State of the Art

The classification of network traffic prediction models is a useful means of helping us understand the existing approaches, addressing new challenges and finding out features that need to be improved [20] [21]. In this section, we address the main concepts in this field and outline the most commonly used techniques. In addition, we propose a taxonomy for assessing and listing the main benefits and drawbacks of each prediction mechanism. Figure 1 depicts the taxonomy that we will now describe in detail.

The choice of the forecast model should take account of the purpose of the prediction as well as the characteristics that reflect the main properties of the data, such as trends, seasonality, patterns of variation and time dependence. After that, we are able to store, organize and analyse the data, and make inferences

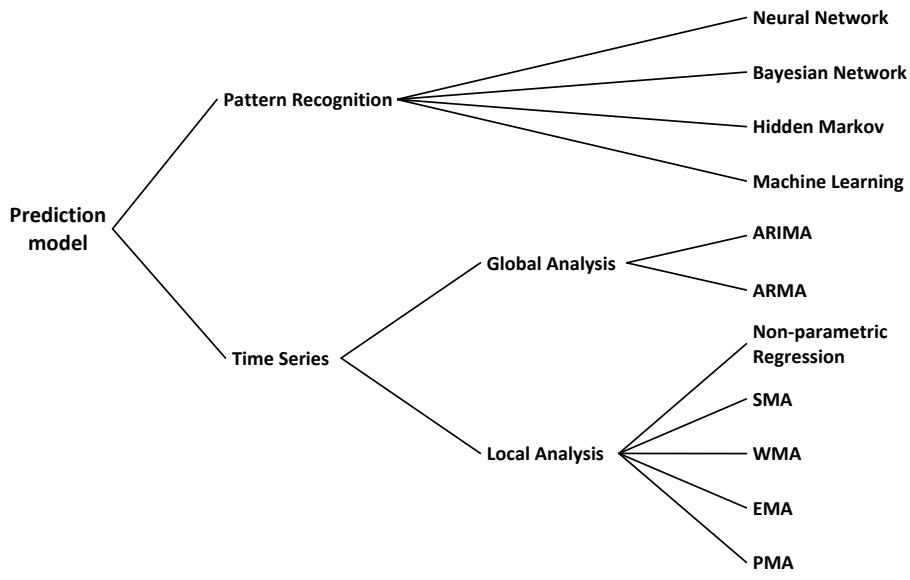


Fig. 1: Taxonomy of Prediction Models

about their future behaviour [22]. There are several predictor models for network traffic in the literature. On top of the taxonomy, we will divide the models into two distinct categories: Pattern Recognition and Time Series.

2.1 Pattern Recognition

As in the case of the Time Series, predictors based on pattern recognition use quantitative information. The term "quantitative" refers to a type of information based on quantifiable data (objective properties). A prediction system based on pattern recognition requires taking note of many issues such as feature extraction, selection and cluster analysis.

In this sense, Artificial Neural Networks (ANN) [23], Bayesian Networks [24], Hidden Markov [25], and Machine Learning [26] techniques have been receiving increasing attention in the field of prediction models. Pattern Recognition is an attempt to model the human brain, which means that pattern recognition basically involves learning from experience. In addition, the model must be able to maintain a good accuracy, which is not simple in short time, since the precision of these techniques is dependent on sufficient historical data being available.

Chen *et al.* [28] use genetic programming to build a Flexible Neural Tree (a flexible multi-layer feed-forward neural network) for online network traffic prediction. This approach was adopted to obtain a better understanding of the main features of traffic data. Moreover, the proposed method is able to forecast small time-scale traffic measurements and can reproduce the statistical features of real traffic measurements. However, to achieve reliable results, it requires an initial input that is dependent on the characteristics of the data under evaluation. Hongying [29] also presents a neural network (back-propagation neural network) trained by a modified quantum-behaved particle swarm optimization in order to predict the dynamic network traffic flow.

Auld *et al.* [30] propose a tool based on Bayesian Networks to support Internet traffic identification. This model is useful for recognizing the future behaviour of the network traffic by taking into account past experience. By means of this approach, it was possible to classify several application types without a source or destination address, by using samples of traffic that could enable the categorization to be made on the basis of commonly available information. However, the authors concluded that the accuracy of the classification declines quickly over time as the nature of the Internet traffic changes. This limitation hampers its applicability in cloud computing.

Dainotti *et al.* [25] resort to a statistically-based approach to classify network traffic by combining network traffic with different categories of network applications. Traffic prediction is performed by taking into account characteristics such as inter-packet times and payload size, as well as their temporal correlation. The proposed solution involves a classification of the packet-level traffic based on a Hidden Markov Model. The aim of this work is to use the obtained classification to offer different levels of Quality of Service (QoS) that depend on the class of traffic. It also assists in the enforcement of security policies for different applications and the identification of malicious traffic flows. All the evaluations are carried out by analysing offline traffic from different network topologies.

Methods from the machine learning theory [26] involve developing systems with the capability of learning from the recognition of old patterns. Nguyen and Armitage conducted a survey with many applications for machine learning [27]. Among them, machine learning also has the potential to support network traffic prediction. In this context, the application of machine learning involves several stages. First, the features must be selected to feed the algorithm. After that, these features are assigned to flows that are calculated over multiple packets. The machine learning classifier creates rules by linking the features with well-known traffic behaviour. Finally, the algorithm is able to predict the network traffic based on previous learned rules.

Bermolen and Rossi [31] propose a solution based on a support vector machine to solve the problem of link load forecasting based only on their past measurements. This machine learning approach showed a noticeable variation in the prediction performance carried out for a given training set size. When the training set is smaller than 50% of all the historical data, the authors consider the model to be under-trained and the prediction error is large, and thus precludes its application in real-time.

In general, pattern recognition approaches are performed in two stages: training and forecasting. The training and forecasting tasks have to be performed in different time-scales. Although the model training is an offline operation, it has to be done periodically so that a strict degree of accuracy is maintained, while the forecasting has to be made continuously and online [31]. If these two stages overlap, there will be an increase in the workload, thus making it unsuitable to online traffic prediction in dynamic environments such as cloud.

2.2 Time Series

In general terms, time series are an ordered sequence of the values of a variable at equally spaced time intervals. Time series techniques take account of the internal features in the data flow such as autocorrelation and seasonality. On the basis of these features, it is possible to estimate the future behaviour of the data flow from a set of past data. In this context, the scope of the analysis enables us to classify time series approaches into two types: global and local.

Global Analysis

A global analysis is based on assumptions about the shape of the data that can be numerically described by taking into account all the values of the population. More importantly, the data sample has to represent the distribution of the population in question, and the normal distribution is usually adopted for this kind of approach. In this context, a global analysis procedure allows more conclusions to be made about the data. However, there is a limitation in the approach that follows the normal distribution since the model does not work properly for small sample sizes ($n < 30$). Nevertheless, a local analysis procedure is a good means of overcoming this drawback [32].

If the series follows a repetitive pattern which is believed to be constant at each slot of time (for instance, from week to week or from day to day), a seasonal

adjustment may be required to anticipate the behavioural pattern. The advantage of a seasonal prediction model is that it fits the seasonal pattern, by enabling the periodic variations to be included in the prediction results. The disadvantage is that this type of prediction model requires a time window with a large number of values, and this adds an extra workload to compute the estimation. In this case, two models can be used to forecast the time series, Autoregressive Moving Average (ARMA) [33] or its Integrated variant (ARIMA) [34].

Sang and Li [6] use the ARMA model to evaluate the network traffic prediction regarding two points of view: (*i*) how far into the future the network traffic can be forecast; and (*ii*) about the minimum acceptable forecast error. In the prediction assessment, the results show a tradeoff between prediction error and control timescale. Furthermore, the paper shows that the prediction accuracy deteriorates as the slot time size increases.

Moayedi and Masnadi-Shirazi propose a network traffic prediction and anomaly detection model based on ARIMA [35]. The ARIMA model gives a description of a stationary stochastic process in terms of polynomials to fit the dataset. In their paper, they decompose the data flow to isolate the anomalies from the normal traffic variation. The authors then try to predict anomalies separately from the normal traffic, and the paper evaluation shows that the anomalies have been successfully detected. Their work was evaluated with synthetic data and depends on large historical data. Zhao [36] uses wavelets for analysing time domain signal of the time series for improving the prediction accuracy level. However, the computational complexity in predicting each wavelet coefficient is high.

Local Analysis

Local analysis procedures are considered to be more inaccurate than a global analysis because they use less information in making their calculations. The data correlation only relies on local data to make assumptions about the population. For instance, it fits in with the real data using just one subset to develop a function that describes the behaviour or the variation of the data. This is an advantage since the analysis does not require global information to make statistical inferences [32].

Sometimes, the time series has a linear dependence with regard to its series of values, and shows a constant growth factor. In this case, non-parametric regression method can be regarded as a prediction model type [37]. In light of this, regression models may be used for estimating a polynomial function that represents the time series trend. However, cloud computing provides a dynamic environment with a complex network traffic behaviour, and is far from having a linear trend pattern. This means that it requires high degree polynomials to fit the network traffic baseline. To achieve an online prediction with accurate results, the prediction model requires a constant adjustment of a polynomial function. However, this incessant process is expensive [38] and thus causes the regression models to conflict with the *Low complexity* requirement for cloud network traffic prediction.

When a local analysis is expected, a moving average model may compute a local average of data at the end of a time window, on the assumption that this is the best estimate to represent the current mean value around which the data is ranged. The size of the time window can be adjusted dynamically making these models more suitable if the time series change suddenly. A moving average model

is closely correlated to a local analysis procedure since it combines simplicity with an attempt to build up a function to describe a local trend.

Papadopouli *et al.* [7] evaluate a set of forecast algorithms to characterize the traffic load in an IEEE802.11 infrastructure. Their work describes the Simple Moving Average (SMA) as the unweighted mean of the previous data points in the time series. In addition, SMA is less demanding than more complex predictors, such as Autoregressive Integrated Moving Average (ARIMA), which requires more parameters to compute the prediction. They point out some of the advantages of SMA, such as its simplicity, low complexity and ease of application. Together with this, Lee *et al.* [39], show that SMA also provides a basic and efficient tendency index.

Li *et al.* [40] study anomaly detection methods for high-speed network traffic. The purpose of this work is to come up with a sensible mechanism for detecting significant changes in massive data streams with a large number of flows. Through a model based on a Weighted Moving Average (WMA), the algorithm estimates the value of the next interval and compares with the real traffic. After that, all traffic that does not match the reference model is considered to be an anomaly, thus it is able to detect Distributed Denial-of-Service (DDoS) and scan attacks.

Klinker [41] describes mathematical tools to identify and predict market trends. In particular, their study shows that the Exponential Moving Average (EMA) can be used for an effective forecasting of network traffic combined with a local analysis of the historical data.

In a previous work [17], we proposed a systematic approach for estimating network traffic by resorting to a statistical method based on a Poisson process (Poisson Moving Average - PMA). In addition, we used a dynamic sliding window size algorithm (DyWiSA) to weight past observations by taking advantage of well-known network traffic features such as short-range dependence [18].

2.3 Pattern Recognition vs. Time Series

A comparison between Pattern Recognition and Time Series has already been performed in [37] and [42]. Zhang and Qi compare a neural network with an autoregressive integrated moving average (ARIMA) [37]. In this case, even when the neural network is trained with all original data available, its performance is inferior to the ARIMA model. This shows that neural networks are not able to capture trend variations effectively. Approaches based on pattern recognition, such as neural networks, have a serious drawback: they learn the training patterns but lose the ability to make generalizations, which means that the model may give inaccurate results for unknown patterns [42].

Moreover, another limitation of pattern recognition in network traffic prediction for cloud is the fact that a considerable amount of offline computation is needed to train it properly. In order to make pattern recognition approaches effective, these models must be trained on a representative data set; otherwise, it may not be feasible to achieve a satisfactory degree of accuracy [43]. For instance, research studies have shown that using 50% of the data for training is not enough to provide accurate predictions [31]. Other studies that even using 100% of the historical data show that pattern recognition achieves worse results than the Autoregressive Integrated Moving Average [42]. Wei Li and Andrew W. Moore [44] show that the labour required in hand-classification process increases along with

Table 1: Summary of Related Work

Approach	Desirable features			
	Low historical dependency	**Low complexity**	**Online prediction**	**High accuracy**
SMA – Maria Papadopouli *et al.* [7]	√	√	×	×
SMA – Wan-I Lee *et al.* [39]	√	√	×	×
WMA – Aiping Li *et al.* [40]	√	√	×	√
EMA – Frank Klinker [41]	√	√	×	√
PMA – Bruno Dalmazo *et al.* [17]	√	√	×	√
DyWiSA – Bruno Dalmazo *et al.* [18]	√	√	√	√
ARMA – Jose L. Torres *et al.* [33]	×	×	√	√
ARIMA – Moayedi and Shirazi [35]	×	×	√	√

the size of the training set using Machine Learning approach. In addition, this category of predictors does not meet the *Low historical dependency* requirement, and is computationally expensive, going against the *Low complexity* requirement for cloud computing.

Zhani M. F. *et al.* [45] show that enlarging training data set does not really improve traffic predictability using a time series-based approach (ARMA and ARIMA models). In addition, several weighted sampling schemes can be employed such as: Simple Moving Average (SMA), Weighted Moving Average (WMA), Exponential Moving Average (EMA) or Poisson Moving Average (PMA). The WMA is more sensitive to recent values than a SMA. However, an EMA is usually preferred to a WMA, because its exponentially weighted average does a more sensible work of discounting the older data and its smoothing parameter is continuous since it is readily optimized to each new iteration [7]. PMA, which is based on a Poisson process, usually fits the network traffic behaviour better than the other short-term predictors [18].

Summing up, as was shown previously, approaches based on pattern recognition and non-parametric regression method are not considered for cloud traffic prediction due to their high complexity and historical dependency. Table 1 summarizes several network traffic predictor models regarding desirable requirements for cloud computing environment.

Approaches based on local analysis generally display low levels of historical dependency. This leads to a low complexity solution for traffic prediction by reducing the amount of data required for processing, when compared with the strong historical dependency models. Both of them (local and global analysis), allow online traffic prediction (using DyWiSA for local analysis approaches), but with solutions of different levels of complexity. On the one hand, models with global analysis usually achieve accurate results in prediction. On the other hand, models based on local analysis provide a solution with lower computational complexity (see Subsection 3.7).

This work compares several local and global analysis approaches by taking into account the requirements of the cloud computing environment (highlighted in Table 1). In particular, it provides a systematic methodology for evaluation of predictors, that makes it easier to compare different models in terms of accuracy, historical dependency, time and computational overhead.

3 Prediction Approaches

The model of behavioural prediction can usually be characterized by using a time series of historical values (*e.g.* network traffic packets). With the aid of historical traffic data, it is possible to predict future cloud network traffic. Hence, future values can be forecast based on a correlation between the variation of the values at the time and in its current state.

As noted earlier, there is a large and growing body of literature that regards network traffic prediction as a means of facilitating the monitoring and management of computer networks [46]. Although most research studies employ classical methods that are largely based on historical data such as time series and neural networks, some recent works [10] [11] show that long-term historical dependence is not suited to cloud computing due to the high volatility of this environment. In this section we consider previous prediction models which carry out forecasting on the basis of local and global data analysis.

3.1 Simple Moving Average - SMA

The Simple Moving Average (SMA) is the most popular of the moving averages used for predicting based on local analysis. It is calculated as the unweighted mean of the previous n data values as shown in Figure 2(a).

The term moving is used because for each new slice of time, the oldest data value is dropped from the sliding window as soon as the new value becomes available. An example of a simple equally weighted moving average for a sample of n values v is the mean of the previous n values of the time series, as we can see in Equation 1.

$$SMA = \frac{\sum_{i=1}^{n}(v_i)}{n} \tag{1}$$

3.2 Weighted Moving Average - WMA

When using a moving average technique as described in the previous subsection, each of the coefficients used to compute the predicted value is weighted equally. However, it might sometimes be useful to put more weight on recent observations that are closer to the time period being predicted. In this case, we use a weighted moving average technique. As a general rule, a weighted average is any average that uses several coefficients to provide various weights for data at different positions in the sliding window. Figure 2(b) shows an example of this function inside a sliding window.

In this work, WMA specifically refers to weights that increase in arithmetical progression. In a sliding window with n samples, in the WMA, the newest value has weight w_n, the second newest w_{n1} and so on until the oldest value goes down to zero. The sum of the weights in a WMA have to be 1. In a normal case, for each weight w and value v in the sliding window, the denominator will always be the sum of the individual weights, as can be seen in Equation 2.

$$WMA = \frac{\sum_{i=1}^{n}(w_i * v_i)}{\sum_{i=1}^{n}(w_i)} \tag{2}$$

3.3 Exponential Moving Average - EMA

The Exponential Moving Average (EMA) is an exponentially weighted moving average that applies weighting coefficients which decrease exponentially in the course of time inside a sliding window. The weighting for each older value decreases exponentially, but never reaches zero. Figure 2(c) shows an example of the weight decrease.

In a similar way to other moving average techniques, EMA must only be used for a set of data without seasonal behaviour [47]. This moving average technique reacts faster to recent value changes than with a simple moving average that attributes more weight to the latest changes and less to the changes that lie further away. The formula for calculating EMA is given by Equation 3:

$$EMA = \frac{\sum_{i=1}^{n}(exp^i * v_i)}{\sum_{i=1}^{n}(exp^i)} \tag{3}$$

3.4 Poisson Moving Average - PMA

The Poisson distribution is a natural choice for describing the probability of the number of occurrences in a field or continuous interval (usually time or space), such as number of defects per square meter, number of accidents per day or number of network packets per minute [17]. We note that in our study the unit of measure (time) is continuous, but the random variable (number of packets) is discrete. In other words, a Poisson process is used to determine the probable minimum and maximum number of transactions that can occur within a given time period, from a series of discrete values [48].

In order to predict the network traffic behaviour, the Poisson distribution is partitioned into a sliding window containing network traffic packages from a cloud based system, from max value to the left of the distribution. The sliding window aims to restrict the analysed information to a local domain, mapping the input and output information with a similar structure to a queue. In this approach, the Poisson parameter lambda (λ) is also utilized to define the size of the sliding window.

From a time interval of size t = λ, let a truncated Poisson distribution of size n be represented $p1$, $p2$, ...,pn. The latest values of network traffic (y_t) in a cloud data flow may be associated inside a sliding window of size = λ as follows:

$$\hat{y}_t = p_1 y_{t-1} + p_2 y_{t-2} + ... + p_\lambda y_{t-\lambda} \tag{4}$$

where $\hat{y}_t$ represents the result of the prediction process, namely, the expected value of the network traffic. Equation 5 summarizes this process, whilst Figure 2(d) presents the behaviour's function inside the sliding window.

$$PMA = \sum_{i=1}^{\lambda} p_i y_{t-i} \tag{5}$$

3.5 Autoregressive Moving Average - ARMA

With regard to a time series analysis, the ARMA model provides a description of a stationary stochastic process in terms of two polynomials – first an autoregressive (AR) and, secondly, a moving average (MA). The ARMA is usually referred as the ARMA(p,q) model where p is the order of the autoregressive part and q is the order of the moving average part [6].

Given a time series data, the ARMA model is able to characterize and then forecast future values within the time series and can be defined as shown below:

$$X_t = c + \sum_{i=1}^{p} \varphi_i X_{t-i} + \varepsilon_t + \sum_{i=1}^{q} \theta_i \varepsilon_{t-i} \tag{6}$$

where c is a constant, φ_1, ..., φ_p are the autoregressive parameters of the model, θ_1, ..., θ_q are the moving average parameters of the model and the random variable ε_t, ε_{t-i} are white noise error, usually assumed to be a Gaussian distribution with zero mean [49].

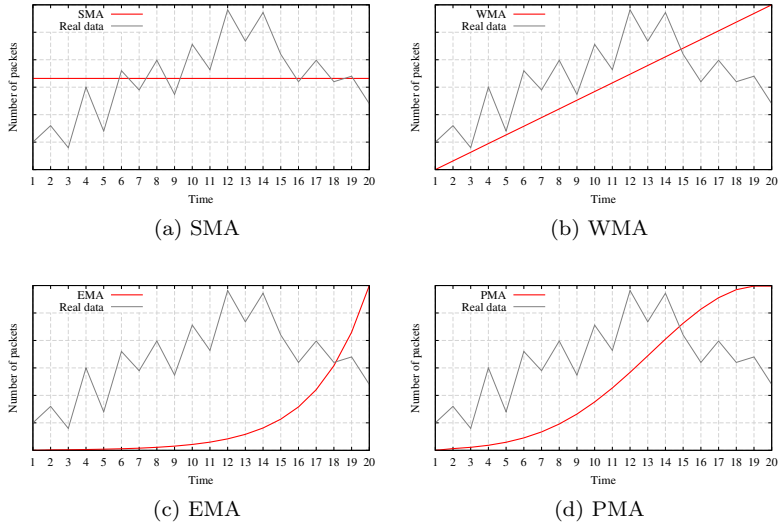
3.6 Autoregressive Integrated Moving Average - ARIMA

The Autoregressive Integrated Moving Average - ARIMA is a model powered with a time series and used either to obtain a better understanding of the data behaviour or to forecast future points in the series [52]. ARIMA is a predictor model which is a generalization of an ARMA model [53].

The ARIMA model is referred to in the literature as ARIMA(p,d,q) where the parameters p and q have the same mean as in the ARMA model. The difference is the d parameter which refers to the order of the integrated part of the model. All the parameters must be non-negative integer numbers. Given a time series of data X_t where t is an integer index and the X_t is any real number, an ARIMA(p,d,q) model is given by:

$$\left(1 - \sum_{i=1}^{p} \phi_i B^i\right) (1 - B)^d X_t = \delta + \left(1 + \sum_{i=1}^{q} \theta_i B^i\right) \varepsilon_t \tag{7}$$

where B is the backshift operator, *i.e.*, the previous element of a time series (X_{t-1}), ϕ_1, ..., ϕ_p are the autoregressive parameters of the model, θ_1, ..., θ_q are the moving average parameters of the model and the random variable ε_t is the white noise error, as described in the ARMA model.



(a) SMA (b) WMA

(c) EMA (d) PMA

Fig. 2: Behaviour of the models inside the sliding window

3.7 Computational Complexity Comparison

There are various algorithms that can be used for solving the problem of predicting network traffic. In general, a solution to this problem may be seen as a set of instructions that guarantee the correct answer can be found. In other words, for a problem p, an algorithm is a finite process for computing $p(n)$ for any given input n [54].

The performance analysis is conducted for the average volume of network traffic, which defines the next sliding window size. Once the sliding window size is set, it is possible to estimate the number of operations that will be performed by the predictors.

For the purpose of our analysis, we consider that the operation time to read the elements inside the sliding windows yields negligible complexity (or the ever constant $O(1)$). The SMA-based predictor has a function that assigns equal weight to all the elements of the sliding window, thus, the number of operations increases linearly with the size n. Other predictors evaluated in this work have special values to weight the elements inside the sliding window. For example, WMA uses a linear function that increases from zero to its maximum size, as illustrated in Figure 2(b). The EMA approach uses an exponential function, but the number of calculations and the window size increase at the same rate. Therefore, EMA has $O(n)$ complexity. PMA uses a weighting function based on the Poisson process and like EMA, PMA also has $O(n)$ complexity. Finally, the computational complexity for the ARMA predictor, as well as the ARIMA model, is $O(n^2)$ [55] [56] [57]. The behaviour of all these local analysis approaches is illustrated in Figure 2, which shows a window for n = 20.

It is worth noting that this performance evaluation considers the recursive use of the algorithms. In other words, from 1 to i, each element e_i uses all the results

from the previous e_{i-1} values. In summary, the analysis of algorithms is quite important since it is a means of obtaining performance evaluation criteria that are independent of the technology adopted or programming language used [58].

4 Methodology for Evaluating Traffic Predictors

The purpose of this analysis mechanism is to provide a standardized approach to evaluate the predictor models from real historical data of cloud-based network traffic. Figure 3 depicts the basis of our mechanism, by highlighting its main conceptual components, the personnel involved, and their interactions.

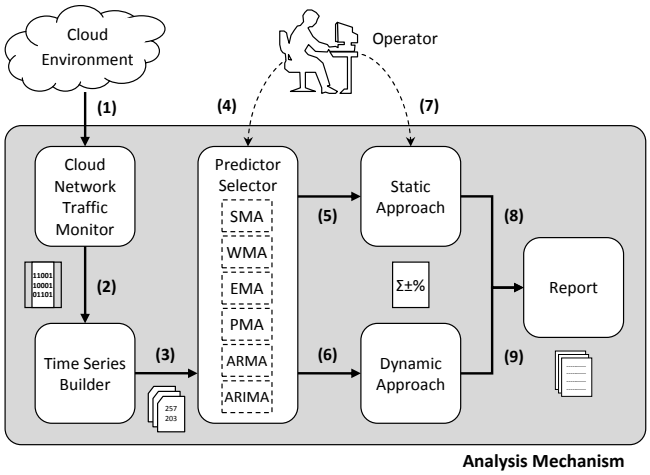


Fig. 3: Elements of the proposed mechanism and interactions

Real-time cloud traffic data (Flow 1) is constantly being gathered from the cloud environment by the *Cloud Network Traffic Monitor*. This data is subsequently processed by *Time Series Builder* to retrieve and organize relevant information (*e.g.* timestamp, number of network packets, and the protocols involved) for the following prediction (Flow 2). Once the *Time Series Builder* process is complete, the time series is ready to be read (Flow 3). In addition, an *Operator* may interact with the *Predictor Selector* to determine which predictor model will perform the prediction (Flow 4).

Following this, the time series (combined with the predictor model) is sent to the next two components (Flows 5 and 6). At this point, the prediction will be performed fully- and semi-automated (Dynamic and Static approaches). The latter approach is subject to an *a priori* data flow analysis to enable it to produce the statistics about the data or intervention by an *Operator* (Flow 7) and thus aid the process (manually set up with the aid of the statistical parameters). As the output of these processes, the *Report* component will generate a description regarding the accuracy of both prediction approaches (Flows 8 and 9).

Having presented a general outline of the analysis mechanism, in the following subsections there will be a more detailed description of: (i) the monitoring process

of cloud-based network traffic, (ii) creating a time series, and (iii) the Static and Dynamic approaches to predicting network traffic in a cloud computing environment.

4.1 The Cloud Network Traffic Monitor

Cloud computing provides a scalable and elastic environment, but geographically far away from the user. However, as previously mentioned, the prediction of the cloud network traffic requires having access to detailed information about the operation of the network (*e.g.* timestamp, protocols, etc.).

To address this issue and facilitate the prediction process, the *Cloud Network Traffic Monitor* must constantly monitor the cloud infrastructure (or a specific application). Thus, it will be able to capture the usage patterns and network traffic trends during a given time period. Basically, this component is responsible for counting all the traffic it receives from the network rather than counting only the frames that the controller is supposed to receive.

The cloud environment offers different levels of services to the clients. The lower level resources are usually restricted and hidden from the users (at the PaaS and SaaS level, for instance). Hence, the user does not have permission to monitor and control the whole network infrastructure. However, the cloud providers are able to monitor the network at the resource and virtualization layer since they are responsible for the low-level monitoring [59].

It is not within the scope of this study to propose a particular approach to monitor the cloud infrastructure. However, several tools have the potential to monitor the cloud environment with the aid of distributed agents in virtual machines, such as Nagios, OpenNebula, and Nimbus [60].

4.2 The Time Series Builder

As illustrated in Figure 3, once the data has been collected, it is sent to the next component. The *Time Series Builder* handles the data by measuring the number of packets in the network traffic at regularly spaced intervals (*i.e.* the time between the observations must be constant), and thus forms a discrete time series ordered by the time.

At the same time, the *Time Series Builder* has to filter all the data in a search for similar protocols and the excess data that does not match the requirements of the filtering will be dropped. When building a time series, the input value is computed for each single variable. As a result of this process, the time series will be ready for analysis by any of the prediction models.

4.3 The Predictor Selector

Traffic predictors usually operate over all of the previous data or resort to windows of a finite but fixed size [28]. However, the network traffic in the cloud computing environment may undergo sudden changes due to the large number of requests and dynamic demands which are made without any prior notification [10].

This led us to consider adopting the sliding window approach as a "forgetting" process, which makes it possible to restrict the amount of data to be processed. If the sliding window is small (which is normal for local analysis approaches), the model will be more sensitive to changes. In addition, it will generate low workload due to the reduced number of data packets that need processing. This occurs when the data flow has a stable behaviour. If the sliding window is large (such as the ARMA and ARIMA models), the predictor will hide any traffic anomalies. This situation arises when the time series is rapidly increasing (or reducing) the data flow.

Different approaches are required for local and global analysis so that these changes in traffic behaviour can be taken into account. When predicting network traffic data based on a global analysis, the ARMA and ARIMA models are used. In the case of ARMA and ARIMA modelling, we use the statistical environment *R* [66]. When estimating the network traffic by means of the ARMA model, the *Analysis Mechanism* selects the "FitARMA" package [64]. When the estimate was performed using the ARIMA model, the *Predictor Selector* used the "forecast" package [65] to fit the time series. For this, both packages use a function for returning automatically the best set of parameters (GetFitARMA and Auto.Arima, respectively) according to the algorithms presented in [64] and [65]. These functions conduct a search over possible model within the order constraints provided.

The variance between the previous and current sliding window was taken into account for the local analysis. The example illustrated in Figure 4 shows a sliding window with size four. Each value of the original data flow is weighted with a portion of the statistical distribution of the corresponding predictor model (SMA, WMA, EMA, and PMA). Thus, at time t, the sliding window has a set of four values $\{12, 16, 26, 18\}$. In the next turn, at time $t + 1$, the next value to enter inside the window will be 19, and when this occurs, the oldest value (12) leaves the sliding window. This process will be repeated as long as there is a data flow from the network.

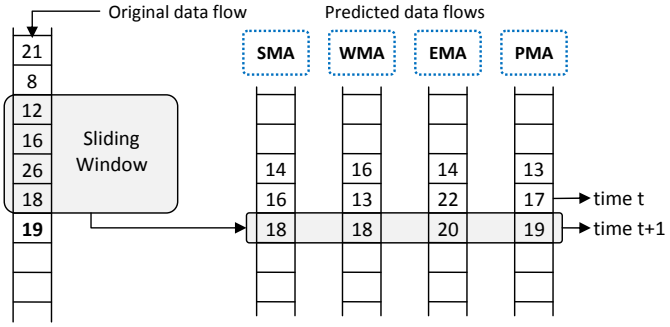


Fig. 4: Sample of prediction inside the sliding window

At this point, the *Operator* might interact with the *Predictor Selector* component so that it can choose the predictor model and define some parameters for the static approach (*e.g.* arithmetic mean, variance of the data, and time period). Thereafter,

the prediction will perform two types of approaches for each time period inside a sliding window for all the local analysis models, namely, static and dynamic.

4.3.1 Static Approach

Algorithm 1 shows the procedure for predicting traffic. The input (*i.e.* the data flow in Figure 4) corresponds to the time series traces (*tmTrace*). It should be remembered that the input data can be any set of cloud data. Furthermore, the *Operator* also has to set up the statistical parameters such as arithmetic mean, variance, standard deviation, number of slices, vector model, as illustrated at lines 3 and 4 of Algorithm 1 (for the static approach). When the algorithm works dynamically, these parameters are computed automatically inside the last sliding window. The number of slices is equivalent to the number of samples of data used for the calculation of the prediction coefficients (from the moving average models). Once the vector with slices has been properly shaped, the algorithm estimates the next value for the network traffic for each new value from the time series (line 12, *nextValue*). As output, the vector containing the network traffic prediction, is represented by *vPrediction*.

Algorithm 1 Pseudocode for predicting network traffic

Input: Time series trace
Output: Prediction of network traffic

```
 1: Start
 2:     read tmTrace
 3:     read parameters
 4:     vector vPModel
 5:     vector vPrediction
 6:     for (i = 0; i < tmTrace.size(); i++) do
 7:         var nextValue ← 0
 8:         for (j = 0; j <= window.size(); j++) do
 9:             if (i − j >= 0) then
10:                 var tmp ← (vPModel[j] ∗ tmTrace[i − j])
11:             end if
12:             nextValue ← (nextValue + tmp)
13:         end for
14:         vPrediction.add(nextValue)
15:     end for
16: End
```

4.3.2 Dynamic Approach

The *Dynamic Approach* component is responsible for the definition of the window size that serves as input for the next sliding window. To reduce the complexity of predicting network traffic, time-bounded past information is considered by means of a sliding window of a size defined by the Dynamic Window Size Algorithm (Algorithm 2), which thus makes it suitable for online prediction in a cloud computing context [18].

Algorithm 2 describes the operation of the *Dynamic Window Size Algorithm*. It resorts to a sliding window of variable size, which changes according to the variance (σ^2) found in the last sliding window and the current sliding window.

A small variance indicates that the predicted data is close to the mean, while a high variance indicates that the predicted data is spread out from the mean. The theoretical maximum variance (σ^2_{max}) of a given set of data can be estimated by the product of the difference of its extreme values, y_a (lowest value), y_b (highest value), and the average, as follows [19]:

$$\sigma^2_{max} = (m - y_a)(y_b - m) \tag{8}$$

As input, the algorithm receives the average of the current sliding window, the average from the previous sliding window and the current sliding window. It compares the average of the old sliding window with the average of the current sliding window. To avoid unnecessary overhead of the algorithm, we select an α equal to 0.05 which represents a 95% confidence interval. In this context, the confidence interval corresponds to a boundary value for the population parameter where the difference between the current value and the mean of the last window is not statistically significant at a 5% level.

Algorithm 2 Dynamic Window Size

Input: Average of the current sliding window, *newAvg*
Average of the previous sliding window, *oldAvg*
Current sliding window, *sWindow*
Output: Next window size, *wSize*

```
1:  Start
2:      procedure DyWiSA(newAvg, oldAvg, sWindow)
3:          var wSize ← sWindow.size()
4:          var direct ← newAvg/oldAvg
5:          var inverse ← oldAvg/newAvg
6:          var ratio ← | direct − inverse |
7:          if (ratio > (1 + α)) then
8:              var volume = σ²max / σ²
9:              if (newAvg > oldAvg) then
10:                 wSize ← wSize + volume
11:             else
12:                 wSize ← wSize − volume
13:             end if
14:         end if
15:         return wSize
16:     end procedure
17: End
```

Let *direct* be a value which measures average changes between the current window and last window, and *inverse* represents its inverse. If the difference between *direct* and *inverse* is higher than the threshold $(1 + \alpha)$, *i.e.* statistically significant, the window size is increased (or decreased) by *volume*. Before the maximum variance of a sliding window can be quantified and, thus the variation of the window size known, a measurement is needed to express the largest variance possible inside a subset of the entire population. This is obtained through the ratio between the σ^2_{max} and the σ^2 inside a sliding window. This whole process is represented by the variable *volume* at line 8 of Algorithm 2.

Finally, the algorithm returns the window size that will be used by the *Sliding Window* component. This Dynamic Window Size Algorithm is at the core of the

dynamic approach since it dynamically adapts the window size by only resorting to local data from current and previous sliding windows, rather than global traffic data.

4.4 Report

For evaluating network traffic prediction techniques many different metrics are used to measure the quality of the forecasting [50]. The analysis mechanism provides as result a detailed report about the prediction models. For the static and the dynamic approaches, several statistical descriptors are calculated and arranged in a table such as arithmetic mean, mean square error, standard deviation, standard error, variance, etc.

The effectiveness of the prediction is measured through the Normalized Mean Square Error (NMSE) [51] and Mean Absolute Percent Error (MAPE) [52]. NMSE is defined as:

$$NMSE = \frac{1}{\sigma^2}\frac{1}{N}\sum_{t=1}^{N}\left(X_t - \hat{X}_t\right)^2 \tag{9}$$

where σ^2 is the variance of the time series over the prediction duration, X_t is the observed value of the time series at time t, $\hat{X}_t$ is the predicted value expected from X_t, and N is the total number of predicted values. This metric is widely utilized to assess prediction accuracy. Its results are compared with a trivial predictor, which statistically predicts the mean of the actual time series, in which case NMSE = 1. If NMSE = 0, this means that it is a perfect predictor, whereas NMSE > 1 means that the predictor performance is worse than that of a trivial predictor [51].

MAPE measures expressed errors as a percentage of the actual data over the prediction data. It is calculated as the average of the unsigned percentage error, and is defined by the formula:

$$MAPE = \begin{cases} \left(\frac{1}{N}\sum_{t=1}^{N}\frac{|X_t-\hat{X}_t|}{|X_t|}\right) * 100 \; if(X_t > 0) \\ \left(\frac{1}{N}\sum_{t=1}^{N}\frac{|X_t-\hat{X}_t|}{|\overline{X}|}\right) * 100 \; otherwise \end{cases} \tag{10}$$

where, X_t is the observed value, $\hat{X}_t$ is the predicted value and N represents the total number of values in the time series as well as referenced in NMSE. If the denominator is zero then the actual value X_t is replaced by the average of time series, $\overline{X}$. When having a perfect fit, MAPE is zero.

5 Evaluation and Discussion

Throughout this section, the time series setup that is used to assess this work, is provided. Furthermore, we evaluate the performance of the static and dynamic sliding window mechanisms employed for the local analysis of traffic prediction. In addition, the results are compared with all the predictors outlined in Section 3. We consider two case studies for evaluation: Dropbox datasets and Data Centre dataset.

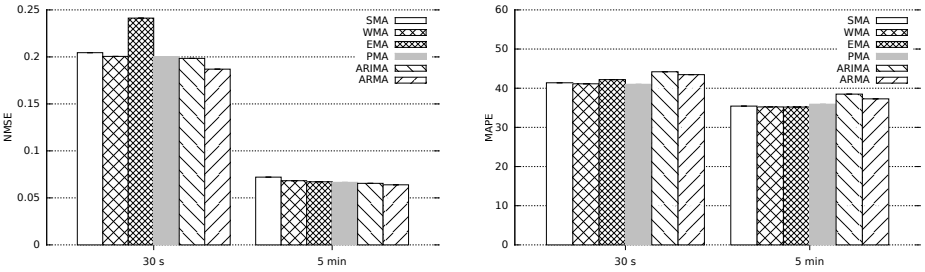
5.1 Dropbox Case Study

Two datasets from Dropbox monitoring were used for this case study, they are: Home 1 and Campus 2, as described in the [61]. Home 1 dataset consists of ADSL and Fiber to the Home customers of a nation-wide Internet Service Provider, but they might be able to use WiFi routers at home to share the connection. Campus 2 was collected in academic environments instead, such as wired workstations in research and administrative offices as well as campus-wide wireless access points.

5.1.1 Setup

The evaluated time series data is concerned with the employment of Dropbox, which is currently the most widely-used cloud storage system [61]. All the measurements and data provided in this subsection were collected from March 24, 2012 to May 5, 2012. The original Dropbox dataset encompasses more than 100 metrics about network traffic. However, for the purposes of this study, the *Time Series Builder* (Subsection 4.2) considers the total number of packets observed from the client (server) to the server (client) and SSL/TLS protocol.

Figure 5 illustrates the evaluation for two different windows size. For this, the time series was divided into intervals of thirty seconds and five minutes. In this scenario, we use the dynamic approach for evaluating the Campus 2 dataset, and the analysis mechanism was performed by applying a sliding window weighted with the four statistical models, ARMA and ARIMA models described in Section 3. The results show that the best level of accuracy can be achieved with 5 minutes interval because larger intervals present lower variance among the data. Other datasets evaluated in this work have presented similar behaviour when performed with thirty seconds and five minutes. For the remainder of the paper, we use 5 minutes interval for evaluating the predictions.



(a) NMSE (b) MAPE

Fig. 5: Evaluation for different sliding window sizes (Campus2)

5.1.2 Results and Discussion

First, Table 2 shows the results of the static approach in which the sliding window size remains constant during the prediction data. After that, we show the perfor-

mance of a dynamic approach which calculates the sliding window size by means of Algorithm 2 in Table 3. Finally, we compare the results with the forecast data that were obtained from the ARMA and ARIMA predictors.

Table 2: Static sliding window size arithmetic mean

Dataset		Mean			Std. Dev.	Variance	NMSE	MAPE
		Arithmetic	Square Error	Std. Error				
1	Home1	29.810	0	0.161	17.96	322.42	0	0
Approach								
2	SMA	29.777	130.39	0.136	15.24	232.19	0.4045	41.58
3	WMA	29.786	110.31	0.137	15.37	236.25	0.3422	36.50
4	EMA	29.806	83.84	0.153	17.08	291.66	0.2807	29.36
5	PMA	29.799	87.68	0.144	16.09	259.11	0.2720	28.90

Static Approach

Although past studies have evaluated different sliding window sizes [17], sliding windows with a size based on an arithmetic mean consistently provide the best results. In this context, Table 2 provides results for sliding window with an arithmetic mean in the static approach.

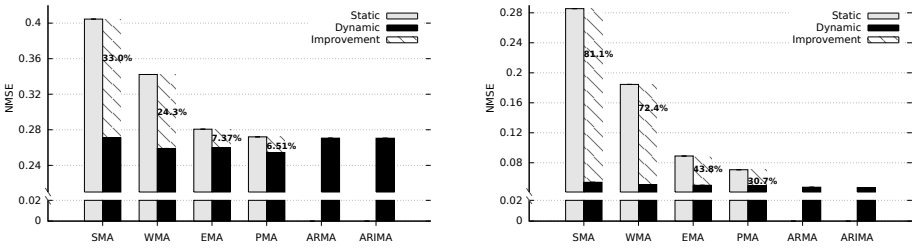
The first line shows statistics from the Dropbox dataset that was used as input for the predictors. The following lines show results for Simple Moving Average, Weighted Moving Average, Exponential Moving Average, and the Poisson-based prediction model, as described in Section 3.

While for most metrics the results of the remaining predictors are not far from those obtained by the Poisson approach, we highlight the results achieved in terms of NMSE and MAPE, where PMA excels when compared with the others. This means that the difference between the estimated values and the real values is the lowest in the overall result of the evaluation.

It should be noted that the other datasets evaluated in this work also generate these statistics tables by adopting a static approach. However, in order to avoid information redundancy, assessments of other datasets will be summarized in a graph.

Dynamic Approach

For the assessment of the dynamic approach, the Algorithm 2 was used to calculate the sliding window size. Furthermore, in the dynamic approach, the predictor models were evaluated from the two Dropbox traffic traces (Home 1 and Campus 2). Figure 6 illustrates the NMSE accuracy of the predictor models. All the local analysis predictor models were tested in their original version with a static window size, as well as by employing the dynamic window size methodology. Although our focus is on making a comparison between predictor models operating with a static window size and a dynamic window size, it was clear that the SMA consistently provides the worst results, irrespective of what window size methodology was used. At the other extreme, there is PMA, which provides the best overall results.



(a) Home1 (b) Campus2

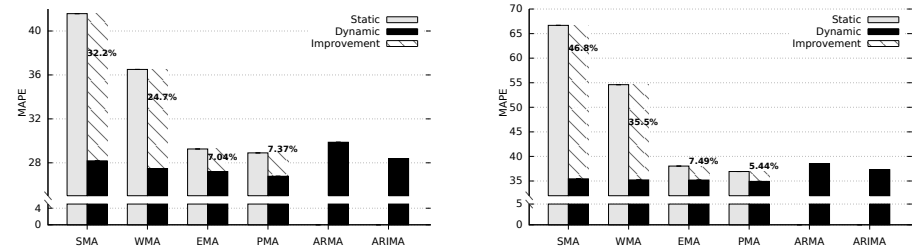
Fig. 6: The NMSE results from the Dropbox datasets

With regard to the comparison between the static and dynamic approaches, our results show that all the predictor models achieve better results when employing the dynamic window size methodology. There is further evidence of this in the NMSE results of Figure 6, which show that all the predictors are improved from as little as 6.51% for the best predictor model (PMA in Home1) to as much as 81.1% for the worst predictor identified (SMA in Campus2). ARMA and ARIMA models provide results that are slightly better for the Campus2 dataset.

Table 3: Dynamic sliding window size

Dataset		Mean			Std. Dev.	Variance	NMSE	MAPE
		Arithmetic	Square Error	Std. Error				
1	Home1	29.81	0	0.161	17.96	322.42	0	0
Approach								
2	SMA	29.802	93.58	0.146	16.31	265.94	0.2709	28.18
3	WMA	29.829	87.13	0.146	16.35	287.31	0.2590	27.47
4	EMA	29.818	84.62	0.153	17.13	273.38	0.2600	27.20
5	PMA	29.827	82.68	0.148	16.58	274.87	0.2543	26.77

It is worth noticing that in Figure 6 (a), WMA achieves a better result than EMA and this is not confirmed in Figure 7 (a). When the predictor model is



(a) Home1 (b) Campus2

Fig. 7: The MAPE results from the Dropbox datasets

assessed by NMSE, the data normalization process tends to improve the results of the predictor with the highest variance (see Equation 9). In this case, the WMA obtains better results than EMA because its predicted data shows the higher σ^2 (see Table 3). In order to avoid the problem of a wide variance of data, the *Dynamic Window Size Algorithm* was also evaluated by MAPE.

Figure 7 shows the performance of the predictor models in terms of error percentage. This illustrates the fact that in both cases (Home 1 and Campus 2) the error rate decreases when the dynamic window size methodology is employed. The overall MAPE results can be seen in Figure 7, which shows that the dynamic methodology improves the prediction results for all the local analysis predictors, from 5.44% (PMA) to 46.8% (SMA). While the ARMA and ARIMA models have similar results for Home1 and Campus2 in terms of MAPE, the best being the PMA.

Running Time

In this subsection, we present the time cost of running the prediction models presented in Section 3. In addition, we compare the performance between the first version of the static window size approach and the Dynamic Window Size Algorithm. Table 4 presents details about the processing time of the prediction models with different size of input data. It is possible obverse that the SMA, WMA, EMA and PMA models spend a similar time to compute the prediction. Analogously, the ARMA and ARIMA models present a similar time for performing predictions. These results are in accordance with the computational complexity discussed in the Subsection 3.7.

It is worth noticing that other datasets evaluated in this work have presented a similar processing time to perform the prediction. An entire view of the results is displayed in Figure 8 which shows the *Static Approach* as the average time of the SMA, WMA, EMA and PMA models with static window size. The *Dynamic Approach* represents the average time of the SMA, WMA, EMA and PMA models using the Dynamic Window Size Algorithm. It is clear that the *Dynamic Approach* presents the best time performance due the lower number of operations [18]. Moreover, the prediction time has been increased exponentially for ARMA and ARIMA models.

Naturally, the minimum real-time measurement that can be achieved depends on the hardware configuration (*i.e.*, processor clock rate, memory available, etc.). For comparative purposes, all the tests performed in this work were based on a standard personal computer with a DualCore Intel Core 2 Duo CPU 6300 1.86GHz and 3Gb DDR2-SDRAM.

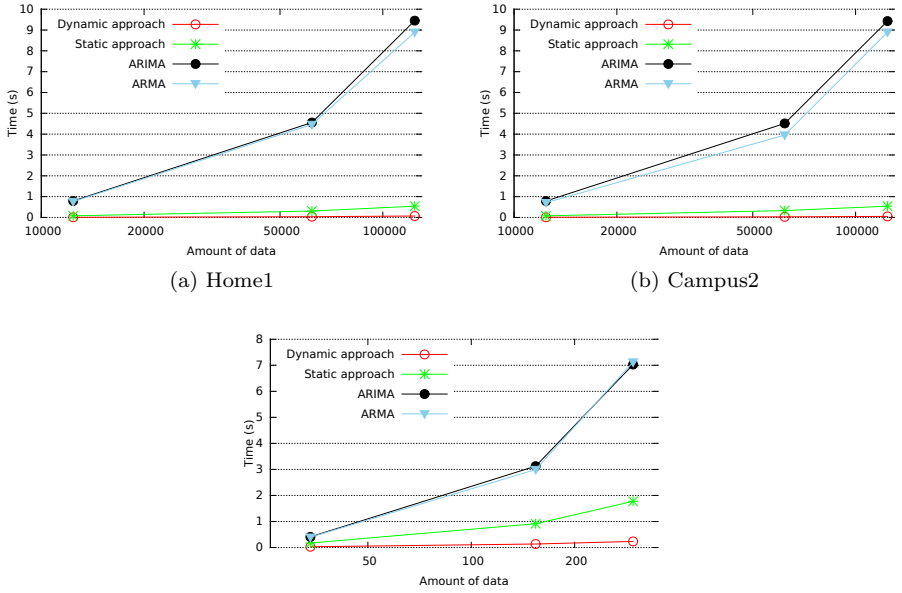
5.2 Data Centre Case Study

At the same time another dataset was used, for a better characterization of the cloud computing environment, and which provides data from monitoring a variety of services that are common in cloud computing [62]. In that work, the authors describe several services that can be found in the dataset such as webmail servers, web portals, instant messaging, web services and multicast video streams.

Table 4: Time consumption for Home1

Model	*123800 elements*		*61900 elements*		*12380 elements*	
	Static	Dynamic	Static	Dynamic	Static	Dynamic
SMA	0.069	0.238	0.034	0.130	0.006	0.026
WMA	0.076	0.614	0.038	0.345	0.006	0.085
EMA	0.076	0.645	0.038	0.363	0.006	0.092
PMA	0.076	0.670	0.038	0.391	0.006	0.112
ARIMA	-	9.456	-	4.558	-	0.791
ARMA	-	8.897	-	4.464	-	0.766

Results in seconds.



(a) Home1

(b) Campus2

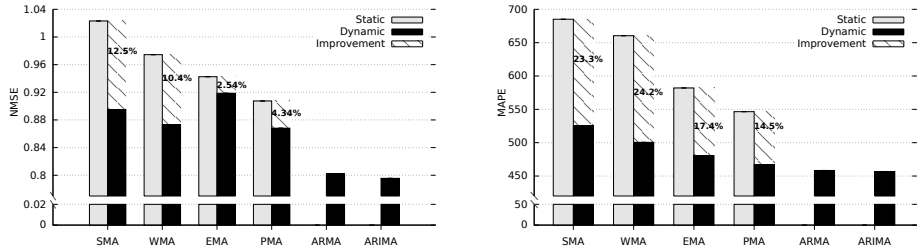
(c) Data Centre

Fig. 8: Time consumption comparison for all data sets

The authors use SNMP link statistics to examine the network-level impact in terms of link utilization, congestion, packet drops, and the dependence of these properties on the location of the links in the network topology and on a daily basis. In addition, the dataset has data from a two-layer topology that introduce server virtualization techniques in order to reduce heating and electric power consumption.

5.2.1 Setup

The data was collected in an academic environment and the dataset consists of more than five years of monitoring. However, the authors only provide a fraction



(a) NMSE (b) MAPE

Fig. 9: NMSE and MAPE results for Data Centre

of the total amount of data used in the original paper [62], for around 10 days. The granularity of the data generated is sixty seconds for each time slot and the resulting time series is called by *Data Centre* throughout this study. The *Data Centre* stores data from around 1000 servers located in the West and Mid-West of the U.S. The *Analysis Mechanism* was performed for the data in a similar way to what is described in the Dropbox case study.

5.2.2 Results and Discussion

We divided the evaluation report into two parts. First, all the local analysis approaches were assessed using the *Data Centre* dataset as input. In this case, the *Analysis Mechanism* is employed for the data by adopting both a static and dynamic approaches. In the static approach, the window size is invariable during the prediction data. In the case of the dynamic approach, Algorithm 2 calculates the sliding window size. After that, we carried out the prediction of the data with ARMA and ARIMA predictors.

Static Approach

In Figure 9, the prediction results are given by drawing on data from the *Data Centre*. In Figure 9 (a), we can see the bars in gray that represent the NMSE achieved for the local analysis predictors (SMA, WMA, EMA and PMA). By analogy with the Dropbox case study, PMA yields the best result which is close to 0.91 for NMSE. Figure 9 (b) shows the performance for MAPE. PMA, like in the NMSE evaluation, achieves a better result than the other local analysis predictors assessed in this work.

Dynamic Approach

The *Data Centre* dataset was also evaluated by Algorithm 2 so that it could generate results from a dynamic window size perspective. Figure 9 illustrates the accuracy of the prediction models.

It is clear that all the predictors significantly improve their results after adopting a dynamic window size approach to forecast the time series. This was confirmed in the NMSE and MAPE results of Figure 9. The EMA predictor showed the smallest improvement with 2.54% and the SMA showed the best improvement with 12.5% regarding NMSE. In the case of MAPE, Figure 9 (b) illustrates an improvement from 14.5% (PMA) to 24.2% for WMA. Figure 9 shows that the ARIMA model provides better results than all the others predictors to forecast the data from the *Data Centre* with NMSE = 0.796 and MAPE = 456.8%. Finally, regarding NMSE, the ARMA model achieves a high degree of accuracy when predicting the network traffic, but is slightly outperformed by the ARIMA model, which achieves the best prediction result. In the case of the MAPE evaluation, the results are not different and the ARIMA model provides the most accurate prediction in the context of the cloud *Data Centre*.

The results are based on a comparison between several predictors. In summary, the moving average approach represents a solution that computes a local average of data at the end of the time window, based on the assumption that this is the best estimate to represent the current mean value around which the data are ranging. These approaches are suitable if the time series is subject to sudden changes, as cloud computing traffic is. In this case, an anomaly may be easily absorbed within the time window without compromising the prediction as a whole [63].

Approaches using global analysis (ARMA and ARIMA) achieve more accurate results when predicting network traffic from the cloud *Data Centre*. However, local analysis approaches outperform the results for cloud applications monitoring, as was seen in the Dropbox case study. Moreover, with a smaller sliding window, oldest values also have less influence on the predicted network traffic. This indicates that a predictor that gives priority to recent history achieves better results for dynamic cloud computing environments.

On the one hand the global analysis achieves accurate results, while on the other, this kind of prediction is more expensive in computational terms than predictors based on local data analysis. In addition, a local data analysis is able to provide accurate predictions with relatively low levels of historical data dependency and computational complexity.

6 Conclusion

Network traffic prediction is relevant for many management applications such as: resource allocation, admission control and congestion control. In this work, a taxonomy for network traffic prediction models is proposed, as well as an analysis mechanism that provides a standardized approach for evaluating network traffic predictors based on global and local data analysis. The outcomes of our mechanism enable the performance comparison of several predictors in the cloud, particularly in terms of accuracy, historical dependency, time and computational overhead.

From the observation of the results of the Dropbox case study, it can be seen that all the predictions based on local analysis present a considerable improvement after using the Dynamic Window Size Algorithm (DyWiSA). Apart from this, the DyWiSA facilitates online traffic prediction due to its short dependency on historical data. Compared to other predictors, Simple Moving Average performed

significantly worse. Furthermore, besides the good results, the Poisson Moving Average has maintained the same computing complexity of the predictor models based on local analysis assessed in this work. Considering the *Data Centre* dataset with traffic from a diverse set of common cloud services, the ARIMA model shows a slight advantage over the other predictors in terms of accuracy. However, this is achieved at the cost of high computational complexity and time consumption. Poisson Moving Average, which is more attractive due to its lower computational complexity, has shown itself to be more suitable for dynamic cloud environments than the other predictor models assessed.

Future lines of research include: (*i*) splitting and analysing the cloud *Data Centre* network traffic by different applications to confirm if local analysis performs better in comparison to global analysis; and (*ii*) using Poisson Moving Average methodology to perform anomaly detection of network traffic in virtualized environments.

Acknowledgements This work was funded by CAPES and CNPq (Brazil) through the Ciência sem Fronteiras Program/2016.

References

1. P. Mell, T. Grance, The NIST definition of cloud computing, National Institute of Standards and Technology (2011).
2. P. Owezarski, J. Lobo, D. Medhi, Network and Service Management for Cloud Computing and Data Centers: A Report on CNSM 2012, Journal of Network and Systems Management, Springer, volume 21, pp. 707–712, (2013).
3. A. Dainotti, A. Pescape, K. Claffy, Issues and Future Directions in Traffic Classification, Network, IEEE volume 26, pp. 35 –40, (2012).
4. Prangchumpol, D, A Network Traffic Prediction Algorithm Based On Data Mining Technique, Proceedings of World Academy of Science, Engineering and Technology, number 79, pp. 1196, 2013.
5. M. Rahmani, K. Tappayuthpijarn, B. Krebs, E. Steinbach, R. Bogenberger, Traffic shaping for resource-efficient in-vehicle communication, IEEE Transactions on Industrial Informatics, volume 5, pp. 414–428, (Nov. 2009).
6. A. Sang, S.-q. Li, A predictability analysis of network traffic, Computer Networks volume 39, pp. 329–345, (2002).
7. M. Papadopouli, E. Raftopoulos, H. Shen, Evaluation of short-term traffic forecasting algorithms in wireless networks, in: 2nd Conference on Next Generation Internet Design and Engineering, 2006. NGI'06., IEEE, pp. 8–14, (2006).
8. B. Zhou, D. He, Z. Sun, Traffic modeling and prediction using ARIMA/GARCH model, Modeling and Simulation Tools for Emerging Telecommunication Networks. Springer, pp. 101–121, 2006.
9. K. Salah, K. Elbadawi, R. Boutaba, An Analytical Model for Estimating Cloud Resources of Elastic Services, in: Journal of Network and Systems Management. Springer., pp. 1–24, 2015.
10. H. Ballani, P. Costa, T. Karagiannis, A. I. Rowstron, Towards predictable datacenter networks., in: SIGCOMM, volume 11, pp. 242–253, 2011.
11. K. Vieira, A. Schulter, C. Westphall, C. Westphall, Intrusion Detection for Grid and Cloud Computing, It Professional volume 12, pp. 38–43, (2010).
12. D. Plonka, P. Barford, Network anomaly confirmation, diagnosis and remediation, in: 47th Annual Allerton Conference on Communication, Control, and Computing, 2009. Allerton 2009., pp. 128–135. doi:10.1109/ALLERTON.2009.5394858.
13. Y. Xie, Y. Zhang, Z. Ye, Short-term traffic volume forecasting using kalman filter with discrete wavelet decomposition, Computer-Aided Civil and Infrastructure Engineering volume 22, pp. 326–334, (2007).

14. W. Xiong, H. Hu, N. Xiong, L. T. Yang, W.-C. Peng, X. Wang, Y. Qu, Anomaly secure detection methods by analyzing dynamic characteristics of the network traffic in cloud communications, Information Sciences volume 258, pp. 403 – 415, (2013).
15. R. Buyya, J. Broberg, A. M. Goscinski, Cloud computing: Principles and paradigms, volume 87, John Wiley & Sons, 2010.
16. T.-S. Lim, W.-Y. Loh, Y.-S. Shih, A comparison of prediction accuracy, complexity, and training time of thirty-three old and new classification algorithms, Machine Learning volume 40, pp. 203–228, (2000).
17. B. L. Dalmazo, J. P. Vilela, M. Curado, Predicting traffic in the cloud: A statistical approach, in: Third International Conference on Cloud and Green Computing (CGC'13), pp. 121–126, 2013. doi:10.1109/CGC.2013.26.
18. B. L. Dalmazo, J. P. Vilela, M. Curado, Online traffic prediction in the cloud: A dynamic window approach, in: The 2nd International Conference on Future Internet of Things and Cloud (FiCloud'2014), pp. 9–14, 2014. doi:10.1109/FiCloud.2014.12.
19. B. L. Dalmazo, J. P. Vilela, M. Curado, Online traffic prediction in the cloud, in: International Journal of Network Management, volume 26, pp. 269–285. doi:10.1002/nem.1934, (2016).
20. N. Hoque, M. H. Bhuyan, R. Baishya, D. Bhattacharyya, J. Kalita, Network attacks: Taxonomy, tools and systems, Journal of Network and Computer Applications volume 40, pp. 307 – 324, (2014).
21. M. Whaiduzzaman, M. Sookhak, A. Gani, R. Buyya, A survey on vehicular cloud computing, Journal of Network and Computer Applications volume 40, pp. 325 – 344, (2014).
22. X. Lu, Z. Yu, B. Guo, X. Zhou, Predicting the content dissemination trends by repost behavior modeling in mobile social networks, Journal of Network and Computer Applications volume 42, pp. 197 – 207, (2014).
23. P. Cortez, M. Rio, M. Rocha, P. Sousa, Internet traffic forecasting using neural networks, in: International Joint Conference on Neural Networks, 2006. IJCNN '06., pp. 2635–2642, 2006. doi:10.1109/IJCNN.2006.247142.
24. B. Dalmazo, W. Cordeiro, L. Rabelo, J. Wickboldt, R. Lunardi, R. dos Santos, L. Gaspary, L. Granville, C. Bartolini, M. Hickey, Leveraging it project lifecycle data to predict support costs, in: IFIP/IEEE International Symposium on Integrated Network Management (IM'2011), pp. 249–256, 2011. doi:10.1109/INM.2011.5990698.
25. A. Dainotti, W. De Donato, A. Pescape, P. Salvo Rossi, Classification of network traffic via packet-level hidden markov models, in: Global Telecommunications Conference, 2008. IEEE GLOBECOM 2008. New Orleans, Louisiana, pp. 1–5, 2008.
26. J. Erman, A. Mahanti, M. Arlitt, Qrp05-4: Internet traffic identification using machine learning, in: Global Telecommunications Conference, 2006. GLOBECOM '06. IEEE, pp. 1–6, 2006. doi:10.1109/GLOCOM.2006.443.
27. T. Nguyen, G. Armitage, A survey of techniques for internet traffic classification using machine learning, Communications Surveys Tutorials, IEEE volume 10, pp. 56–76, (2008).
28. Y. Chen, B. Yang, Q. Meng, Small-time scale network traffic prediction based on flexible neural tree, Applied Soft Computing volume 12, pp. 274 – 279, (2012).
29. Jin, Hongying, Li, Linhao, Dynamic network traffic flow prediction model based on modified quantum-behaved particle swarm optimization, Journal of Networks, volume 8, number 10, pp. 2332–2339, 2013.
30. T. Auld, A. Moore, S. Gull, Bayesian neural networks for internet traffic classification, IEEE Transactions on Neural Networks volume 18, pp. 223–239, (2007).
31. P. Bermolen, D. Rossi, Support vector regression for link load prediction, Computer Networks, QoS Aspects in Next-Generation Networks volume 53, pp. 191 – 201, (2009).
32. D. A. Freedman, Statistical models: theory and practice, Cambridge University Press, 2009.
33. J. L. Torres, A. Garca, M. De Blas, A. De Francisco, Forecast of hourly average wind speed with {ARMA} models in navarre (Spain), Solar Energy volume 79, pp. 65–77, (2005).
34. H. Song, G. Li, Tourism demand modelling and forecastinga review of recent research, Tourism Management volume 29, pp. 203–220, (2008).
35. H. Zare Moayedi, M. Masnadi-Shirazi, Arima model for network traffic prediction and anomaly detection, in: International Symposium on Information Technology, 2008. ITSim 2008. volume 4, pp. 1–6, 2008. doi:10.1109/ITSIM.2008.4631947.
36. Hong Zhao, Multiscale analysis and prediction of network traffic, IEEE 28th International on Performance Computing and Communications Conference (IPCCC)., pp. 388–393, 2009.

37. G. Zhang, M. Qi, Neural network forecasting for seasonal and trend time series, European Journal of Operational Research, Decision Support Systems in the Internet Age volume 160, pp. 501 – 514, (2005).
38. Y. Jin, A comprehensive survey of fitness approximation in evolutionary computation, Soft computing volume 9, pp. 3–12, (2005).
39. W. Lee, C. Chen, K. Chen, T. Chen, C. Liu, A comparative study on the forecast of fresh food sales using logistic regression, moving average and bpnn methods, Journal of Marine Science and Technology volume 20, pp. 142–152 (2012).
40. A. Li, Y. Han, B. Zhou, W. Han, Y. Jia, Detecting Hidden Anomalies Using Sketch for High-speed Network Data Stream Monitoring, Applied Mathematics volume 6, pp. 759–765, (2012).
41. F. Klinker, Exponential moving average versus moving exponential average, Mathematische Semesterberichte volume 58, pp. 97–107, (2011).
42. B. Wilamowski, Neural network architectures and learning algorithms, Industrial Electronics Magazine, IEEE volume 3, pp. 56–63, (2009).
43. B. M. Akesson, H. T. Toivonen, A neural network model predictive controller, Journal of Process Control volume 16, pp. 937 – 946, (2006).
44. Li, Wei, Moore, Andrew W, A machine learning approach for efficient traffic classification, in: 15th International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems, 2007. MASCOTS'07. IEEE, pp. 310–317, 2007.
45. M. F. Zhani, H. Elbiaze, F. Kamoun, Analysis and prediction of real network traffic, in: Journal of Networks, volume 4, number 9, pp. 855–865, 2009.
46. L. Chunlin, L. Layuan, Multi-Layer Resource Management in Cloud Computing, Journal of Network and Systems Management, Springer, volume 22, pp. 100–120, (2014).
47. C. Chatfield, M. Yar, Holt-winters forecasting: some practical issues, The Statistician, pp. 129–140, (1988).
48. C. W. Gardiner, et al., Handbook of stochastic methods, volume 3, Springer Berlin, 1985.
49. W. W.-S. Wei, Time series analysis, Addison-Wesley publ, 1994.
50. Manish Joshi, Theyazn Hassn Hadi, A Review of Network Traffic Analysis and Prediction Techniques, arXiv preprint arXiv:1507.05722, 2015.
51. A. S. Weigend, N. A. Gershenfeld (Eds.), Time series prediction: Forecasting the future and understanding the past, Westview Press, 1994.
52. S. Makridakis, S. C. Wheelwright, R. J. Hyndman, Forecasting methods and applications, John Wiley & Sons, 2008.
53. H. Yin, L. Chuang, S. Berton, L. Bo, M. Geyong, Network traffic prediction based on a new time series model, Wiley Online Library. International Journal of Communication Systems, volume 18, number 8, pp. 711–729, 2005.
54. T. H. Cormen, C. E. Leiserson, R. L. Rivest, Introduction to Algorithms, volume 2, MIT Press Cambridge, 2001.
55. H. Feng, Y. Shu, Study on network traffic prediction techniques, in: International Conference on Wireless Communications, Networking and Mobile Computing, 2005. Proceedings., volume 2, IEEE, pp. 1041–1044, 2005.
56. J. F. Monahan, Some algorithms for the conditional mean vector and covariance matrix, Journal of Statistical Software volume 16, (2005).
57. M. Krunz, A. Makowski, Modeling video traffic using m/g/ infin; input processes: a compromise between markovian and lrd models, IEEE Journal on Selected Areas in Communications volume 16, pp. 733–748, (1998).
58. M. R. Garey, D. S. Johnson, Computers and intractability, volume 174, Freeman San Francisco, 1979.
59. R. Weingrtner, G. B. Brscher, C. B. Westphall, Cloud resource management: A survey on forecasting and profiling models, Journal of Network and Computer Applications volume 47, pp. 99 – 106, (2015).
60. G. Aceto, A. Botta, W. de Donato, A. Pescap, Cloud monitoring: A survey, Computer Networks volume 57, pp. 2093 – 2115, (2013).
61. I. Drago, M. Mellia, M. M. Munafò, A. Sperotto, R. Sadre, A. Pras, Inside Dropbox: Understanding Personal Cloud Storage Services, in: Proceedings of the 12th ACM SIGCOMM Conference on Internet Measurement. Berlin, Germany., IMC'12, pp. 481–494, 2012.
62. T. Benson, A. Akella, D. A. Maltz, Network traffic characteristics of data centers in the wild, in: Proceedings of the 10th ACM SIGCOMM Conference on Internet Measurement, IMC '10, ACM, New York, NY, USA, 2010, pp. 267–280. URL: http://doi.acm.org/10.1145/1879141.1879175. doi:10.1145/1879141.1879175.

63. C. Modi, D. Patel, B. Borisaniya, H. Patel, A. Patel, M. Rajarajan, A survey of intrusion detection techniques in cloud, Journal of Network and Computer Applications volume 36, pp. 42 – 57, (2013).
64. A. McLeod, Y. Zhang, Faster {ARMA} maximum likelihood estimation, Computational Statistics & Data Analysis volume 52, pp. 2166–2176, (2008).
65. R. J. Hyndman, Y. Khandakar, Automatic time series forecasting: the forecast package for R, Journal of Statistical Software volume 26, pp. 1–22, (2008).
66. R. C. Team, R: A language and environment for statistical computing. R foundation for statistical computing, Vienna, Austria, (2012).

Bruno L. Dalmazo has been a Ph.D. student at the University of Coimbra since 2011. He completed his Masters degree in Computer Science in 2011 at the Federal University of Rio Grande do Sul, Brazil. Bruno also received a Bachelors degree in Computer Science in 2008 from the Federal University of Santa Maria, Brazil. His main research interests involve network traffic prediction, as well as security and privacy in cloud computing environments.

João Vilela is an assistant professor at the Department of Informatics Engineering of the University of Coimbra, Portugal. He received his Ph.D. in Computer Science in 2011 from the University of Porto, Portugal. His main research interests are in security and privacy of computer and communication systems, with focus on wireless networks, cloud computing and mobile applications. Other interests include anticipatory networks and intelligent transportation systems.

Marilia Curado is an Assistant Professor at the Department of Informatics Engineering, University of Coimbra, Portugal, where she received a Ph.D. in Informatics Engineering on the subject of Quality of Service Routing. Her research interests are Quality of Service, Mobility, Routing, and Resilience.