J Netw Syst Manage (2014) 22:150–173
DOI 10.1007/s10922-013-9285-1

Network Load Predictions Based on Big Data and the Utilization of Self-Organizing Maps

Aimilia Bantouna · Giorgos Poulios · Kostas Tsagkaris · Panagiotis Demestichas

Received: 16 November 2012 / Revised: 30 August 2013 / Accepted: 10 September 2013 /
Published online: 19 September 2013

Abstract The pervasiveness of computers in everyday life has already increased and keeps increasing the available digital data both in volume and variety/disparity. This large and dynamic availability of digital data is referred to as Big Data and is very promising in bringing forward new insights and knowledge. For obtaining these insights, the proper combination and processing of the data is required. However, the dynamicity and the increasing size of data start making their handling impossible for analysts and raise many concerns on the manner in which data will be processed from now on. Towards this direction, this paper proposes a tool that processes and combines disparate data in order to create insights regarding a future network load. In particular, the tool (based on the unsupervised machine learning technique of Self-Organizing Maps) builds knowledge on the network load that is encountered with respect to the date of interest, the location, the weather, and the features of the day (e.g., weekend, bank holiday, etc.). The obtained results reveal that the tool is capable of learning the traffic pattern and thus predicting the network load that will be encountered in the near or distant future given information for the above presented parameters with small deviations (up to 0.000553 in terms of Mean Square Error). Moreover, the tool maintains only the most representative data instances and thus reduces the data storage requirements with no loss of information.

A. Bantouna (✉) · G. Poulios · K. Tsagkaris · P. Demestichas
Department of Digital Systems, University of Piraeus (UPRC), Piraeus, Greece
e-mail: abantoun@unipi.gr
URL: tns.ds.unipi.gr

G. Poulios
e-mail: gpoulios@unipi.gr

K. Tsagkaris
e-mail: ktsagk@unipi.gr

P. Demestichas
e-mail: pdemest@unipi.gr

Keywords Big data · Dynamic · Resource planning · Load prediction · Unsupervised machine learning · Self-Organizing Maps

1 Introduction

According to [1], unstructured data increase more than 50 % every year due to both numerous available applications and user devices. In particular, the pervasiveness of the network technologies in everyday life through social networking, public information, Rich Site Summary (RSS, also known as Real Simple Syndication) feeds, and other applications create a vast amount of diverse data. Moreover, the large number of disparateries, in software and hardware, user devices (such as laptops, notebooks, mobiles and others) further increase the volume, the variety, and the velocity of the available digital data making their management difficult with on-hand database management tools. These data have been referred to as "Big Data", and despite their difficulty being managed, they have been conceptualized to offer valuable insight on application provisions and network technology aspects when they get analyzed and properly processed. The heavy data analysis that is required for retrieving these insights suggests that new architectures and mechanisms are needed for (a) handling the volume of data that will be stored, (b) aggregating, (c) exploiting, and (d) building knowledge on them. In telecommunication networks, the term "knowledge", refers to information useful for the operators and the network, which cannot be directly monitored. Following the knowledge definition of the European Committee for Standardizations [2] and specifically, the "Official Guide to Good Practice in Knowledge Management", "knowledge is the combination of data/information with the opinions/skills/experience of experts, which can be humans or computational systems and results in a valuable asset that can be used to aid decision making".

On the other hand, despite the large recent research initiatives that target a more flexible and automated resource management, current resource management is rather manual and is quite static. Moreover, resources are planned based on the worst case scenario, i.e., the most demanding scenario. However, network technologies evolve and penetrate everyday life, while the users' mobility increases. The continuously changing environment often results in calling for reconfigurations at various time scales. Thus, the dynamicity that the network should handle increases as well. Additionally, dynamic resource planning, especially when combined with autonomicity can reduce both the Capital Expenditure (CAPEX) and the Operational Expenditure (OPEX). More specifically, dynamic resource planning means that the system is capable of changing its parameters, e.g., transmit power, so as to either expand its capacity and serve more users, when there is high load in an area or to operate using less power when less users need to be served. Power levels impact OPEX. Moreover, autonomicity in the reconfiguration of a network will further reduce OPEX since less human resources are required. Finally, dynamically configuring the network, based on real conditions minimizes the infrastructure investments, which currently are often guided by the worst case scenarios and thus, the CAPEX. Overall, dynamic resource planning of the network is a requirement for

coping with the continuously changing environment and facilitates cost-efficient solutions. Predictions of load that will be encountered in the network in the near or distant future are expected to facilitate future dynamic network planning and resource management, but they cannot be directly monitored. It needs to be extracted from processing and/or combining proper unstructured data. In other words, we will need to acquire knowledge about it.

Motivated by these two observations, this paper proposes a machine learning based tool that exploits Big Data for building knowledge on network load and predicting it. The motivation and the high level description of the problem are summarized in Fig. 1. The tool is anticipated to enhance network management procedures by guiding more dynamic network planning and compressing the available data that need to be stored with the least possible loss of information. In fact, the stored information should, and will, eventually be of higher interest to the Network Operators (NOs). As a final result, the task of heavy data analysis that provides insight into the NOs will be transferred from humans via manual processes to an automated mechanism that is part of a larger system or framework.

The rest of the paper is structured as follow. Section 2 familiarizes the reader with the problem and the directions needed to move closer to its solution. Section 3 analyzes how similar issues are currently treated as well as other initiatives in this area. Moreover, Sect. 4 analyzes the proposed tool, i.e., the mechanisms that comprise it, while Sect. 5 presents the data that were used and the obtained results. Finally, the main conclusions of this paper and the future plans of this research are summarized in the last section.

2 Problem Statement

This paper addresses 2 challenges:

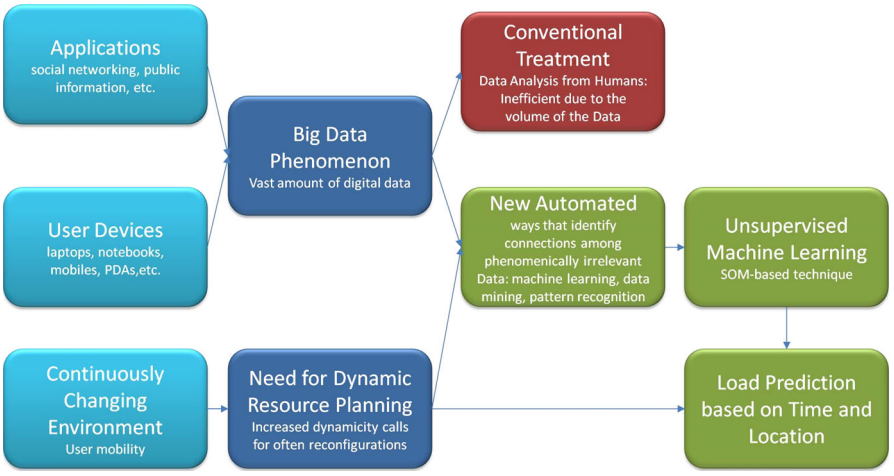

Fig. 1 Motivation and high level problem statement

1. the need for a mechanism that will analyze Big Data, build knowledge upon them and decrease the data that will eventually need to be stored;
2. the need for short- and long-term predictions of network load in order to enhance decision making mechanisms that dynamically plan the resources of the network and manage it.

Towards this direction, the paper uses the unsupervised machine learning technique of Self-Organizing Maps (SOMs) [3, 4] in order to build knowledge on network load and reduce the storage requirements, i.e., the volume of the data that will need to be stored. It also proposes a mechanism that exploits this knowledge to predict network load in the near and distant future. SOMs, as defined by Kohonen, receive multi-dimensional data, i.e., data that come from diverse sources and clusters them based on their similarity. This clustering eventually allows the mechanism to keep only some representative data and discard the rest without any loss of information. A mechanism that exploits this knowledge to predict the load that will be encountered in the near or distant future is proposed and validated.

In order to also support long term predictions of load, the mechanism builds knowledge using not only network parameters but parameters coming from other sources as well. In particular, network load patterns are expected to be related to the area in which the network load is observed and shows the time, the day, the weather, and the features of the days. For example, in southern countries the load in an entertainment area will probably decrease on a rainy day as most people would prefer staying at home rather than going outside. Moreover, user habits and thus, load patterns may differ among countries, e.g., in northern countries, user preferences for going out (in an entertainment area) may not be directly influenced by the weather. Accordingly, load would be expected to decrease in business areas during a weekend or a (bank) holiday while load in domestic or entertainment areas would most likely increase in similar cases. The above statements/examples are only qualitative estimations and do not necessarily apply in all cases. Therefore, more quantitative study and validation is needed to also facilitate the use of the predictions in dynamic network management and resource planning.

In a nutshell, this paper proposes a tool that is capable of (a) using Big Data in terms of collecting and processing past observations/measurements of diverse sources; (b) building knowledge on the network load that is observed with respect to the past observations; (c) exploiting the knowledge obtained so as to predict the network load under predefined conditions; and thus, (d) offering insights into decision making mechanisms, which are responsible for dynamically managing network resources and guiding potential reconfigurations of the network.

3 Related Work

The availability of large amounts of unstructured data, which come from various sources and change quickly, are regularly referred to as Big Data and, although data often seem irrelevant to each other, researchers see much potential in exploiting and combining them so as to derive high level information and new insights for the

business world. Easier access to them through the Web facilitates the research towards this direction. Two website examples that host such free datasets for exploitation and/or exploration are Amazon and Data.gov, a Web site that was initiated in 2009 in Washington and makes all kind of government data accessible to the public. Examples that combine data for inferring useful information have already been recorded in areas involving computers and electronic products, as well as finance, insurance, and government [1] while initiatives have also shown up. A Big Data R&D Initiative was announced in March 2012 in the US [5, 6]. The initiative commits more than $200 million in new funding for enhancing the ability of building knowledge that provides insight from Big Data in sectors such as science, engineering, health, geology, and national security.

Big Data also raises concerns on the manner in which all these unstructured data can be managed and processed. Towards this direction, core technologies are needed to collect, store, preserve, manage, analyze, and share them. In other words, Big Data requires exceptional technology to efficiently process large quantities of data within tolerable timeframes.

Although, Big Data analytics can be done with the software tools commonly used as part of the advanced analytics' disciplines, such as predictive analytics and data mining, the unstructured data sources used for Big Data analytics may not fit in traditional data warehouses. Furthermore, traditional data warehouses may not be able to handle the processing demands posed by Big Data. As a result, a new class of Big Data technology has emerged and is being used involving technologies such as NoSQL databases, Hadoop, MapReduce, in-memory databases [7, 8] and HPCC Systems from LexisNexis [9]. The McKinsey report in 2011 [10] suggests that the involved technology should also include A/B testing, association rule learning, classification, cluster analysis, crowd sourcing, data fusion and integration, ensemble learning, genetic algorithms, machine learning, natural language processing, neural networks, pattern recognition, predictive modeling, regression, sentiment analysis, signal processing, supervised and unsupervised learning, simulation, time series analysis and visualization. In particular, advances in machine learning, data mining, and visualization are often mentioned when referring to Big Data issues and envisaged to enable new ways of extracting useful information and knowledge in a timely fashion from the available massive data sets and thus, complement and extend existing methods of hypothesis testing and statistical inference.

Towards this direction, a SOM seems to be a very good candidate, since it is an unsupervised machine learning technique that clusters and mines multi-dimensional data by mapping them on a 2D-map according to their similarity. The main contribution of this paper is the application, testing, and validation of a SOM to manage and analyze Big Data. Moreover, the study will focus on the problem of network load prediction, i.e., the data that will be selected will be related to this issue.

Research regarding load prediction can also be found in the literature. Indicative studies that have been conducted in this area are analyzed in [11–13], and [14]. The authors in [11] present a technique that is based on time-series theory and hierarchical clustering for predicting the network load. The input parameters that

define each flow are the source and the destination IP and the source and destination ports. Ref. [12] focuses on the network load that is observed in IEEE 802.11 based infrastructures. It proposes and evaluates several traffic forecasting algorithms based on various traffic models that employ periodicity, traffic history, and flow-related information. Moreover [13], proposes a mechanism for predicting load in highly dynamic distributed online games and compares it to other approaches. The input used for each subarea in this prediction mechanism can also be monitored by the network and is the entity count at equidistant past intervals. This parameter is exploited by the mechanism so as to identify the entity count at the next time step. In [14], the Seasonal AutoRegressive Integrated Moving Average (SARIMA) process is proposed for modeling the traffic of a network in terms of the number of bytes passing through the observed link during the time interval. Exploiting its capability of Kalman recursions, the mechanism can eventually predict the bytes that will be encountered in the future but it is based on the fact that the network load repeatedly follows the same pattern without being influenced by factors other than the time. All of these mechanisms, apart from [14], base their knowledge and their functions on network observable parameters. This means that their prediction is rather short-term, e.g., they cannot predict the load in the next month, and this leaves little time for the system to react, e.g., be reconfigured. In this context, the main contribution of this paper is that it is the first study that combines so many phenomenally unrelated, more human oriented Big Data such as the area, the date, the weather, and the features of the date in order to provide long-term predictions as well.

Studies that are considered to be closely related to load prediction are those which deal with congestion prediction and/or avoidance. For example, the mechanism proposed in [15] predicts how close to congestion a link will be in the next timestamp. In order to do so, the proposed mechanism is also based on SOMs and builds knowledge with respect to network monitored data, the current load, and its trend for the last interval. Accordingly, [16] proposes a mechanism that predicts congestion of Asynchronous Transfer Mode (ATM) networks. The mechanism is based on Rough-Fuzzy Neural Networks (RFNN) and is capable of learning and estimating the arriving flow feature, i.e., the total bytes that arrive to the node in an interval. Finally, it predicts if congestion is to be encountered by combining this information with (a) the node buffer, (b) the interval, and (c) the queue size. The Kalman Congestion Avoidance (KACA) scheme for traffic and congestion management that is applied in ATM networks is presented in [17]. In the same study, the KACA scheme is also qualitatively compared to two more schemes, namely the Ohio State University (OSU) scheme and the Dynamic-Congestion Avoidance using Proportional Control (D-CAPC) scheme. However, not even these studies are time and/or spatially oriented.

Load prediction schemes and tools based on time and space can be found in the literature from the field of power electrics. Authors in [18] based on the weather, the day, the time, and the electric load of the past days make short-term predictions of the electric load. Two techniques are exploited for developing two different mechanisms for predicting the electric load. These are the Multilayer Perceptron Neural Network (MLP) and the SOM. The comparison of the two mechanisms

reveals a similar performance. Ref. [19] also targets the forecast of the daily energy load. In this case, the parameter of the date being a holiday (or not) is also taken into account. As a result, the year, month, day of the month, day of the week, if it is a holiday or not, and the load in Megawatts (MW) consumed are what compose the historical data that are exploited as input for the predictions of the mechanism.

This last study, presented in [19], is the closest in terms of used parameters to the study that is presented hereafter. The most related study, in terms of input parameters from the area of telecommunication networks, can be found in [20]. This study focuses on the self-optimization of networks. The proposed mechanism uses clustering techniques for learning and recognizing characteristic patterns, e.g., traffic patterns, performance parameter patterns, and/or configuration parameter patterns, for one or more sites and/or cells of the network given the time of the day, the day of the week, and the geographical area. Accordingly, the network element that is involved is automatically auto-configured for the duration and/or geographical areas defined by the cluster from which the pattern originates.

4 The Tool

The proposed tool is divided into two phases consisting of one mechanism for each (Fig. 2) and is tested on a network of Wi-Fi hotspots. During the first phase, the mechanism collects observations/data with respect to the observed network load, the Wi-Fi hotspot, the timestamp, the weather, and the selected feature of the day. The following parameters are used as input: (i) the area/access point (AP0, AP7, AP8, AP37, AP64 or AP66) expressed in a 6D variable that consists of 0 and 1 s depending on the access point from which the observation was received, e.g., 100000 for AP0 or 001000 for AP8; (ii) the time expressed in minutes (0–1,440 min); (iii) the day (Sunday to Saturday) expressed in a 7D variable that consists of 0 s and 1 s (similarly to the access points, e.g., 1000000 for Sunday or 0001000 for Wednesday); (iv) the week of the year expressed as an integer (1–52); (v) the mean temperature of that day in Celsius; (vi) the precipitation of that day in millimeters; (vii) (bank) holidays expressed in a boolean way (0 or 1); and (viii) the observed load from the access point to the end users under these circumstances in Mbps. These data are then processed/combined so as to transform to information of interest for the NO, i.e.; to knowledge, with respect to the pattern that the load exhibits at a given area in the period of time and to the environmental conditions and the (bank) holidays.

Given the self-similarity nature of network traffic, i.e., the fact that the pattern (when and how the load increases and/or decreases) slightly changes, learning the pattern is expected to offer insights into the load to be encountered in the future (near or distant) at this area. In particular, this knowledge becomes available and is provided to the NO when the latter triggers the second mechanism, i.e., the proposed load prediction mechanism, through a request that specifies (i) the area/access point, (ii) the time, (iii) the day, (iv) the week of the year, (v) the expected mean temperature, (vi) the expected precipitation, and (vii) if the day of interest is going to be a holiday or not. Alternatively, the trigger could include only the area/access

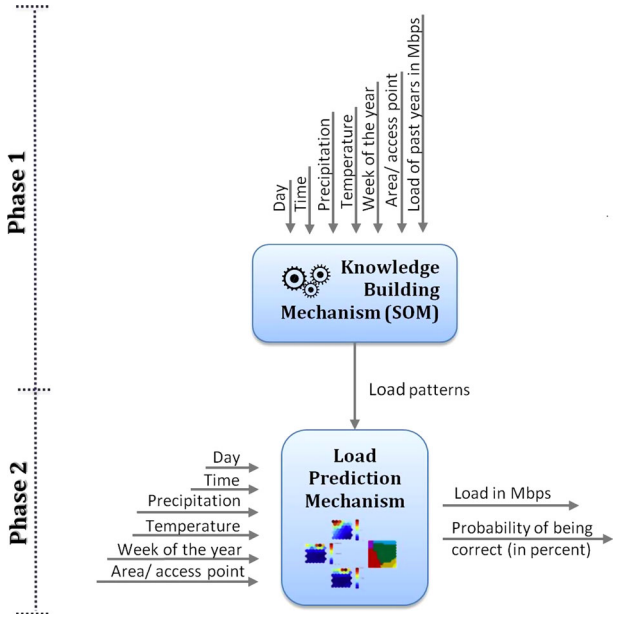

Fig. 2 Phase 1 (*upper part* of the figure): building knowledge on the pattern of the load; Phase 2 (*lower part* of the figure): use of the built knowledge for predicting the load

point, the time and the date, leaving the rest of the information to be completed by a preprocessing module that would receive them from e.g., a calendar RSS feed.

Consequently, the load prediction mechanism that uses the knowledge on the traffic pattern of all the areas with respect to the time and the considered parameters is able to predict the network load. These predictions are returned to the NO expressed in Mbps and followed by a percentage that designates the certainty of the mechanism with respect to this prediction, i.e., the probability of being correct. Sections 4.1 and 4.2 present the two mechanisms of the tool, i.e.,

(a) the knowledge building and data minimization mechanism, which is capable of handling large amounts of data coming from long time-windows, i.e., multiple months and diverse sources and which derives useful and meaningful, higher level information regarding the load of the network; and
(b) the learning-based mechanism for a load prediction mechanism that will predict the load in the future and provide the NOs with such an insight.

4.1 Knowledge Building and Data Minimization

In order to build knowledge from the past experience of the network and not only from its current state, this mechanism needs to include a machine learning technique. The selected technique in this study is the unsupervised neural network called the Self-Organizing Map (SOM).

As a technique based on neural networks, SOM mimics how human perception functions and thus, it is expected to be an adequate substitute for human processes. On the other hand, the term 'unsupervised' characterizes those learning techniques that do not require the desired output of the algorithm or a reward when coming to the right conclusion/correct action during their training; i.e., during the process of identifying the pattern of the data. Thus, using this unsupervised technique provides us with the advantage of not needing the network load that will be encountered when trained. However, it will need the network load that it encountered in the past under a similar context, e.g., at the same access point, time, day, week, temperature, precipitation, and occasion of having a (bank) holiday or not. The training of such techniques and the SOM in particular are thus more flexible and necessary so that the training data can be collected much easier.

The technique was first proposed by Kohonen [3] back in 1997 [4] and is capable of mapping multi-dimensional data, known in SOM theory as data samples on 2D maps. One of the main advantages of a SOM is the fact that this representation is made in such a way that the distance, in terms of difference, between the different data samples can actually be maintained after their representation on the 2D map. In particular, the data are mapped through a process where the more similar the data are, the closer the elements of the map (known as cells of the map) to which they are assigned. The way that the data are mapped is actually the training process of the technique.

In the context of this paper, the dimensions of the data are 19 and comprise 6 dimensions for the area/access point, 1 for the time, 7 for the day, 1 for the week, 1 for the mean temperature, 1 for the precipitation, 1 for holidays and 1 for the load. The analysis of these dimensions is summarized in Table 1.

The steps of the training of the map are depicted in Fig. 3 and are as follows: (1) the map is initialized, i.e., each of its elements/cells is represented by a vector that is comprised of as many components as the dimensions of the data; (2) each data sample is also expressed as a vector with weights that are equal to the values of the dimensions of the data sample and is mapped on the cell whose vector is closest to it when using Euclidean distance; (3) the most important part is that the vectors of the

Table 1 Summary of the input variables, their dimensions, and the parameters

Variable	Dimensions	Parameters
Access point	6	AP0, AP7, AP8, AP37, AP64, AP66
Time	1	T
Day	7	Sun, Mon, Tues, Wed, Thur, Fr, Sat
Week of the year	1	WoY
Mean temperature	1	MT
Precipitation	1	P
Holiday	1	H
Load	1	L

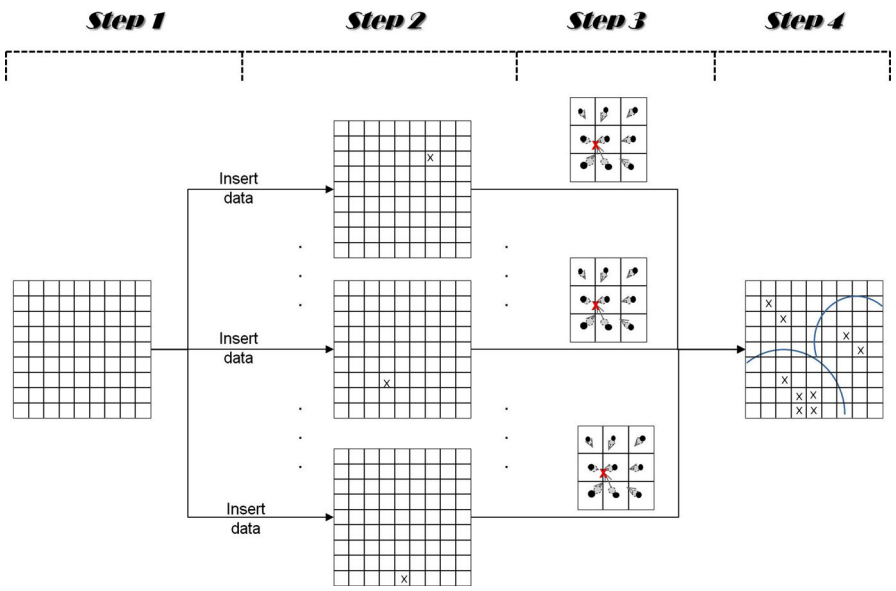

Fig. 3 Overview of the SOM technique

data that are inserted into the training process of the map, also influence the weights of the map (cells) vectors as to adjust them closer to their weights; (4) the end of the training process finds the map complemented with the multi-dimensional data and split into clusters since the distance between the data samples is now represented by the distance between the cells of the map. This is also the reason why this technique is very popular not only as a representation/visualization one but as a clustering technique as well. It should be noted here that, although the total number of parameters that were used for each data sample is 19, only 18 of them were used for the creation of the clusters and the knowledge building, i.e., the components of the vectors were only 18. The measurements of the load (19th parameter/dimension) have been incorporated in the data sample only to be used later during the load prediction.

The exact learning technique that was used was not originally introduced by Kohonen in [3, 4]. The growing [21], the parameterless [22], the hierarchical SOMs [23] and their hybrids [24, 25] are more automated versions of a SOM in terms of not requiring that the user defines the value of so many algorithmic parameters. In particular, a growing SOM enables the map to adjust its size according to its needs for better organizing the data. A parameterless SOM provides flexibility to the map by not demanding that the user specify the number of map cells that will be affected by new training data, i.e., data used for the training, but deciding its own best option while a hierarchical SOM grows in interacting layers allowing better maintenance of the information incorporated in the multiplicity of the data. The mechanism proposed in this paper is based on a hybrid of these versions, namely the Parameterless Growing SOM (PLGSOM) as defined in [25], i.e., a version that is capable of adjusting both its size and its learning parameters. This occurs in order to

enhance its performance and to allow the use of the technique without requiring a priori knowledge of the data set that will be used and/or human intervention to test and specify the optimal value of the algorithmic parameters. The less the parameters that need to be tuned, the more self-adaptable the technique. In this context, a parameterless growing SOM is more appropriate for online learning mechanisms in self-adaptive/autonomic systems, like the ones envisaged for future networks. Here, it is used for building the knowledge that is represented by a map similar to the one in the final phase of Fig. 3.

In addition to this, and in order to verify the improved performance of the PLGSOM compared to the parameterless SOM, their performances were tested using the same data sets and comparing one to the other. The performance metric that was used is the Mean Square Error (MSE). In an indicative test case, the MSE when using the PLGSOM was equal to 0.000553 while in the case of the parameterless, the MSE was equal to 0.000563. This shows that the PLGSOM performs even better than parameterless SOM.

4.2 Learning-Based Mechanism for Load Prediction

As explained in Sect. 4.1, the SOM technique is used to build the knowledge on the load that is observed with respect to the geographical area, the time, the day, the week, the mean temperature, the precipitation, and the (bank) holidays. The load prediction is made based on this knowledge. In particular, the prediction is based on the fact that when mapping new data samples on the designed map, the new data sample is expected to be mapped among similar ones. Thus, knowing the load that was observed with those data samples, one can safely predict the load that is related to the new data sample. More precisely, the predicted value of the load is considered as equal to the mean value of the loads that have been mapped on the specific cell where the new data sample was mapped. In order for this to be accomplished, the load prediction requests should be in the form of a message consisting of similar input parameters to those used during the training of the map.

Moreover, the load prediction is followed by a percentage C that designates the certainty of the mechanism with respect to this prediction, i.e., the probability of being correct. This percentage is calculated by (1), where σ and ΔL are the standard deviation and the range of the loads that have been mapped on the cell on which the data sample in question was mapped, respectively.

$$C = \left(1 - \frac{\sigma}{\Delta L/2}\right) \tag{1}$$

This percentage can also be used as a trust metric, i.e., as a metric of moving forward with or without this prediction. For example, if a network operator received an answer of the form (10, 99 %) that would inform him that the load for this context will be 10 Mbps with a certainty of 99 %, then he would most probably move forward with this prediction. On the other hand, if he received an answer like (100000, 40 %), which informs him that the mechanism predicts a 100 Gbps load but is only 40 % sure of this prediction, then the network operator probably would decide to avoid any reconfiguration as he is not that sure of this prediction.

Additionally, the mechanism is capable of providing insight with respect to the future load of an area either in a time period equal to the observation interval time or for longer time periods by calculating the average of the network load for them. In other words, the request includes the following parameters: (i) the area/access point in question AP; (ii) the time period in question (T_0—first minute of the period and T_f—last minute of the period); (iii) the day in question—DayID; (iv) the week of the year in question—WoY, e.g., 28th week of the year; (v) the expected temperature of that day—T; (vi) the expected precipitation—P; and (vii) if it is going to be a holiday or not—H. Thus, the request will be formed as LInitReq(AP, T_0, T_f, DayID, WoY, T, P, H).

This request (from now on called initial request) is then broken into more requests, equal to the number of times that the time interval of the observations fits in the requested period. For example, if the period is 1 h and the time interval is 15 min, then the initial request is split in 4 new requests. These requests, consisting of (i) the area/access point in question—AP, (ii) the time in question—tk, (iii) the day in question—DayID, (iv) the week of the year in question—WoY, (v) the expected temperature of that day—T, (vi) the expected precipitation—P, and (vii) if it is going to be a holiday or not—H are formed as LReq(AP, tk, DayID, WoY, T, P, H), and are mapped on the created SOM. The load of each request is equal to the mean value of the loads that had been mapped on the specific cell during the knowledge building phase.

Thus, the SOM will return the load prediction L_n in Mbps and the probability of being correct is the mechanism of this prediction in percentage C_n through equal (in the number) responses of the form LRes(AP, tk, DayID, WoY, T, P, H, Ln, Pn). Finally, the mechanism responds to the request by sending back an aggregated response that consists of (i) the area/access point—AP; (ii) the day—DayID; (iii) the week of the year—WoY; (iv) the expected temperature of that day—T; (v) the expected precipitation—P; (vi) if it is going to be a holiday or not—H; (v) the first minute $T_{1,0}$ and the last minute $T_{1,f}$ of the 1st sub-period of the time period in question which is comprised of minutes with an equal load; (vi) the load prediction of this sub-period L_1; (vii) how certain the mechanism is of this prediction C_1; (viii) the first minute $T_{2,0}$ and the last minute $T_{2,f}$ of the 2nd sub-period of the time period in question; (ix) the load prediction of this sub-period L_2; (x) how certain the mechanism is of this prediction C_2, etc., i.e., the final response is in the form of LFinRes(AP, DayID, WoY, T, P, H, $T_{1,0}$, $T_{1,f}$,L_1, C_1, $T_{2,0}$, $T_{2,f}$, L_2, C_2, …, $T_{n,0}$, $T_{n,f}$, L_n, C_n).

The above described process with an indicative example is depicted in Fig. 4. According to the example, let's assume that the initial request was "what will the load be at AP0 (100000) during the period that starts from the 480th minute of the day and ends at the 600th min of the day, on Monday (0100000), the 28th week of the year given the fact that the temperature will be 30 °C, it won't rain, and will not be a holiday?". Thus, the request would be LInitReq(100000, 480, 600, 0100000, 28, 30, 0, 0). The mechanism will split this initial request in 8 new requests, one for each time that the interval time (15 min) fits in the time period of 120 min, i.e., will split the initial request to 8 new ones of the form LReq(100000, $480 < tk < 600$, 0100000, 28, 30, 0, 0) and map them. For each such request, the SOM will send a

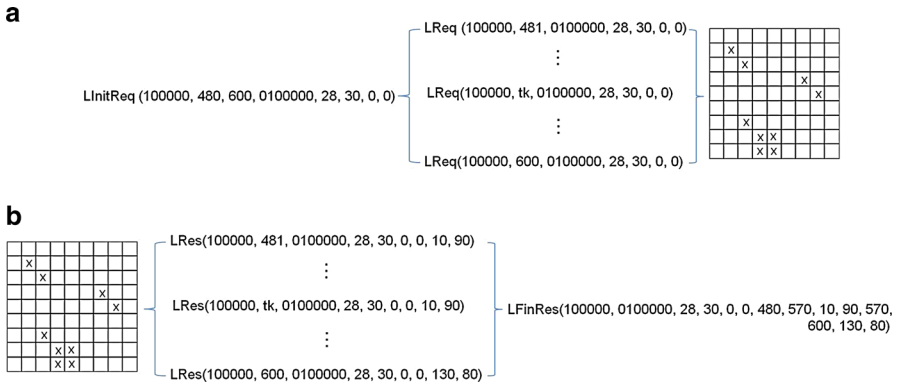

Fig. 4 Load prediction mechanism: an example: **a** a long request is treated as many "15-min" requests, which are then, **b** aggregated in one "Final" response

response similar to LRes(100000, 480 < tk < 600, 0100000, 28, 30, 0, 0, 10, 90). Finally, assuming that the load level prediction will be 10 Mbps for the period 480–570 with a certainty of 90 % and 130 Mbps for the period 570–600 with a certainty of 80 %, the mechanism will respond to the request with the following aggregated response: LFinRes(100000, 0100000, 28, 30, 0, 0, 480, 570, 10, 90, 570, 600, 130, 80).

5 Validation of the Tool

This section presents the way the tool was validated. Towards this direction, the used data and the required pre-processing process is explained in Sect. 5.1 while the results from the tests are analyzed in Sect. 5.2.

5.1 Data and Data Pre-processing

The data used for the research were retrieved from the CRAWDAD database [26]. In particular, the retrieved dataset refers to user session traces that were collected from a large number of Wi-Fi hotspots of "Île sans fil" [27] in Montréal, Québec, Canada for 3 years. The used trace [28] contains 587782 user sessions for 69689 (distinct) users, which were collected from 206 hotspots for the time period from August 28th, 2004 up to August 28th, 2007.

From this trace, the timestamp, the incoming bytes, and the access point ID of the entries that were related to the 6 access points with the highest load were exploited. Those access points were the AP0, AP7, AP8, AP37, AP64 and AP66. These data were then pre-processed so that (a) the timestamp to be translated into the minute of the day, the day (expressed as a 7D variable) and the week of the year; (b) the interval between the entries to be equal to 15 min; and (c) the access point ID to become a 6D variable. Let us note here that both the day and the access

point IDs have to be expressed as 7D and 6D variables, respectively, so as to be processed correctly by the SOM. The SOM technique is very much based on the Euclidean distance between the data. Thus, if the days were only numerically identified, e.g., 1 for Sunday, 2 for Monday, 3 for Tuesday, etc., the SOM would misinterpret their distance for a sign of more or less similar load patterns among the days. Thus, it would suppose that e.g., Monday's observed load pattern is more similar to Tuesday's than it is to Friday's (since Monday–Tuesday Euclidean distance would be equal to 1 and Monday–Friday Euclidean distance would be equal to 4), which is something that cannot be assumed, nor can it be assumed that there is a linear relationship between days of the week (1 through 7) and the observed network load. Using the 7D variable for the day, the Euclidean distance among them (no matter what the days are) is equal to $\sqrt{2}$. The same applies for the IDs of the access points.

Moreover, the data had to be complemented with weather and holidays. The information related to the mean temperature and the precipitation of each day from August 24th, 2004 up to August 24th, 2007 for the area of Montréal, Québec, Canada, i.e., the area where the access points are located, were retrieved from [29] using the Montréal-Pierre Elliott Trudeau International Airport as a reference point and a custom Java tool.

As soon as this was completed, for each load measurement there were data designating (i) the access point from which the load measurement had been monitored, (ii) the time, (iii) the day, (iv) the week of the year, (v) the temperature, (vi) the precipitation during the day that the load was monitored and (vii) if that day was a holiday or not.

Eventually, the data had to become normalized and to be in a format recognizable by the SOM (Table 2). The latter means that the data had to be organized in data samples each of which contained one value for each of the 8 variables (access point ID, time, day, week of the year, temperature, precipitation, holidays and load measurement).

To summarize, before performing the tests, a custom Java tool was used to pre-process the data, i.e., to:

- Read the trace file from [28], extract the information of the timestamp, the AP and the load, convert it in 1–7D variables and make the time interval of the observations equal to 15 min;
- Retrieve the temperature and the precipitation of the area from [29] so as to complement the observations;
- Retrieve from online calendars the (bank) holidays of the area;
- Normalize the values of the data; and
- Gather all the information in the form of data samples, i.e., in one file in which each row (observation) had information for the 8 variables, which had 1–7 dimensions each (see also Table 2).

This file was the one that was inserted in the "Knowledge Building and Data minimization mechanism" of Sect. 4.1 in order to train the PLGSOM and build knowledge on the network load.

Table 2 Sample of data files that were inserted in the "Knowledge Building and Data minimization mechanism" in order to train the PLGSOM and build knowledge on the network load

Observation/data sample	AP0	AP7	AP8	AP37	AP64	AP66	Time	Sunday	Monday
1	1	0	0	0	0	0	48	0	0
2	0	0	0	0	1	0	300	0	0

Observation/data sample	Tuesday	Wednesday	Thursday	Friday	Saturday	Week of the year	Mean temperature	Precipitation	(Bank) holiday	Load
1	1	0	0	0	0	5	20	0	0	54
2	0	0	1	0	0	9	15	1	0	10

5.2 Results

This section summarizes the results that were obtained during the evaluation of the performance of the mechanism. Three types of results are provided (a) qualitative results, (b) quantitative results, and (c) results with respect to the efficiency of the tool to address some of the issues related to the Big Data management. The section concludes with the comparison of the results obtained from this scenario and the results obtained from other scenarios.

For both the qualitative and the quantitative results, knowledge on the load had been built using the "Knowledge Building and Data Minimization" mechanisms of Sect. 4.1 and the load prediction requests were sent to the "Learning-based Mechanism for Load Prediction" of Sect. 4.2. The latter returned the load predictions that were then compared to the real observed values from the dataset. The qualitative results were obtained in the form of comparative diagrams for each prediction while in the case of the quantitative results, 3 different error metrics were used for increasing the probability of finding similar error metrics in other studies and thus making the study comparable to them. Those are (a) the Mean Square Error (MSE), (b) the Root Mean Square Error (RMSE), and (c) the Mean Absolute Error (MAE). Moreover, for both the qualitative and the quantitative results two types of tests were performed 1. those that used the same data for both building the knowledge and for the validation of the proposed tool, and 2. those that used different data for the two functions/processes. The main difference between them is lying in the fact that in the first case, the tool has "seen" the exact same data and thus has the specific experience while in the second case, the tool is familiar with similar, but not the same data; i.e., has similar experiences from which it will try to reach the safest conclusions it can and predict the future load.

5.2.1 Qualitative Results

The qualitative results refer to comparative diagrams between the predicted and the real load values. Figure 5 depicts examples for each access point for (a) seen and (b) unseen evaluation data, respectively.

As it can be observed from the comparative diagrams of the figure in both types of tests, although the mechanism has learn the pattern of the load, i.e., when there is a load, it fails to predict some of its peaks. This may be caused by unpredictable events that suddenly attract (with no prior similar experience) more users, e.g., the place where the access point is hosted became more popular for some reason, e.g., openings of a new coffee shop next to it, or the access point next to it stopped working and thus all users are now connected on it.

These are events that are not captured by any of the selected variables and thus the mechanism cannot learn them or predict them. A potential solution towards this direction that will be considered in the future is to offer the operator or the user of the mechanism the opportunity of informing the mechanism of such a change. Alternatively, a more autonomic solution would be to add a feedback loop

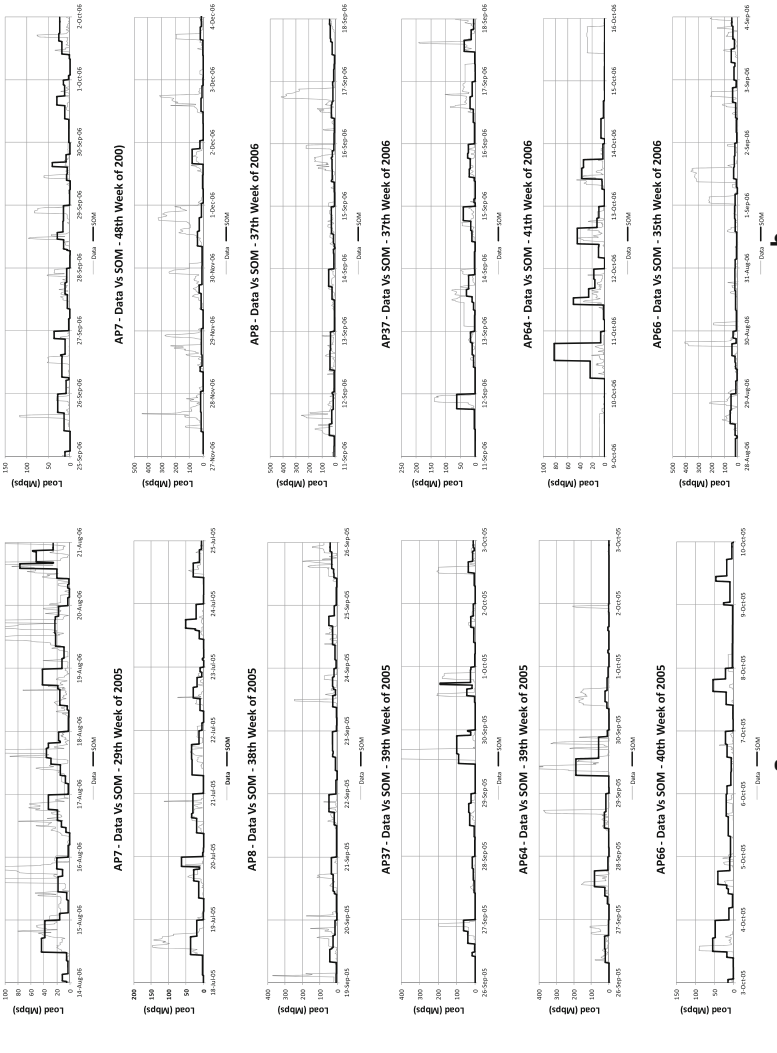

Fig. 5 Indicative examples of comparative diagrams of real and predicted load values for each access point (AP0, AP7, AP8, AP37, AP64 and AP66) when using **a** seen and **b** unseen evaluation data

to the mechanism that would inform it about the difference between the prediction and the actual load, as part of online knowledge building. This would then insert a correction factor that would re-adjust the learned pattern. This last solution would probably benefit the results of the used dataset a lot where the network load seems to increase a lot each year. Figure 6 captures the changes of the network load of one of the access points, namely the AP0, over the years.

5.2.2 *Quantitative Results*

Table 3 summarizes some indicative tests and their respective quantitative results. In the table, the data that were used for knowledge building are referred to as training data while those used for the evaluation of the mechanism are named evaluation data. Moreover, the table also points out the size of the map, measured in cells, which in the case of growing the SOM also represents how adjusted the map is to the training data.

Additionally, for the purposes of the evaluation of the mechanism, the initial dataset was split in three sub-datasets: (a) Y1 which refers to the dates from August 28th, 2004 up to August 28th, 2005, (b) Y2 involving data from August 28th, 2005 up to August 28th, 2006, and (c) Y3 which expands from August 28th, 2006 up to August 28th, 2007. For each of the created maps (1–3) two tests have been performed, one using the training data for the evaluation as well (1b, 2b, and 3b) and one having different training and evaluation data (1a, 2a, and 3a).

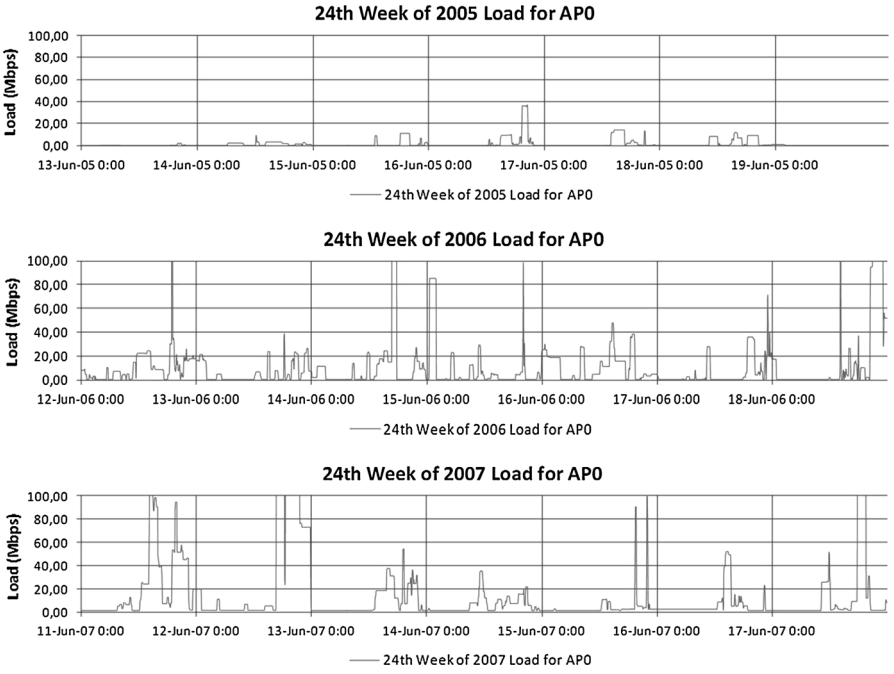

Fig. 6 Diagram of the network load of AP0 for the 24th week of the years 2005, 2006 and 2007

Table 3 Quantitative results

	Training data	Map size	Evaluation data	MSE	RMSE	MAE
1a	Y1 & Y2	3,465	Y3	0.000557	0.023607	0.007925
1b	Y1 & Y2	3,465	Y1–Y2	0.000181	0.013469	0.005442
2a	Y2	2,695	Y3	0.000578	0.024048	0.009601
2b	Y2	2,695	Y2	0.000277	0.016655	0.007579
3a	Y1 & Y2	928	Y3	0.000553	0.023517	0.007925
3b	Y1 & Y2	928	Y1–Y2	0.000191	0.013817	0.005768

Comparing the results presented in the table, it must be noticed that the mechanism performs significantly better when the trigger refers to a case that has already been "seen" by the SOM. This observation was expected and that is also one more reason why online training has been considered.

Another useful observation is the fact that when only Y2 was used for building the knowledge (cases 2a and 2b); the performance of the mechanism deteriorates. This test was selected mainly for identifying if observations older than 1 year deteriorate the performance of the mechanism. According to the obtained results, it seems that older observations not only don't deteriorate the performance of the mechanism but they even enhance it. This is probably due to the fact that when using data only from Y2, less data have contributed to the knowledge building and thus, the SOM is familiar with fewer combinations of the involved variables.

The last observation, which comes from the comparison of 3a and 3b with 1a and 1b, respectively is that smaller maps result in better performance of the mechanism when the evaluation data differ from the training data and worse performance when the same data are used for both knowledge building and the evaluation. This is related to the fact that the bigger the map is, the more adjusted to the training data it becomes. In machine learning, this is also known as overtraining, which eventually results in the learning technique being unable to generalize well. For avoiding such cases, the performance of the technique between all the test cases (i.e., with seen or unseen data) should be kept as close as it gets. Thus, the best performance of the mechanism among the 3 presented in the table is considered to be the one using the 3rd map. Figure 7a depicts this map while Fig. 7b–g depict the component maps of the access points and the days, the holidays, the time, the week of the year, the temperature and the precipitation, respectively.

5.2.3 Results with Respect to the Big Data Issues

According to the SOM theory (see also Sect. 4.1 and [3, 4]), the SOM maps multi-dimensional data onto 2D maps. This means that the dimensions of the data decrease from many to 2. Moreover, the training of the SOM results in the formation of clusters. Given the fact that the data (samples) that comprise the clusters are similar to each other, each cluster can be represented by only one data sample without any loss of information. Thus, storing only one data sample per cluster is enough to maintain the information; the rest can be discarded.

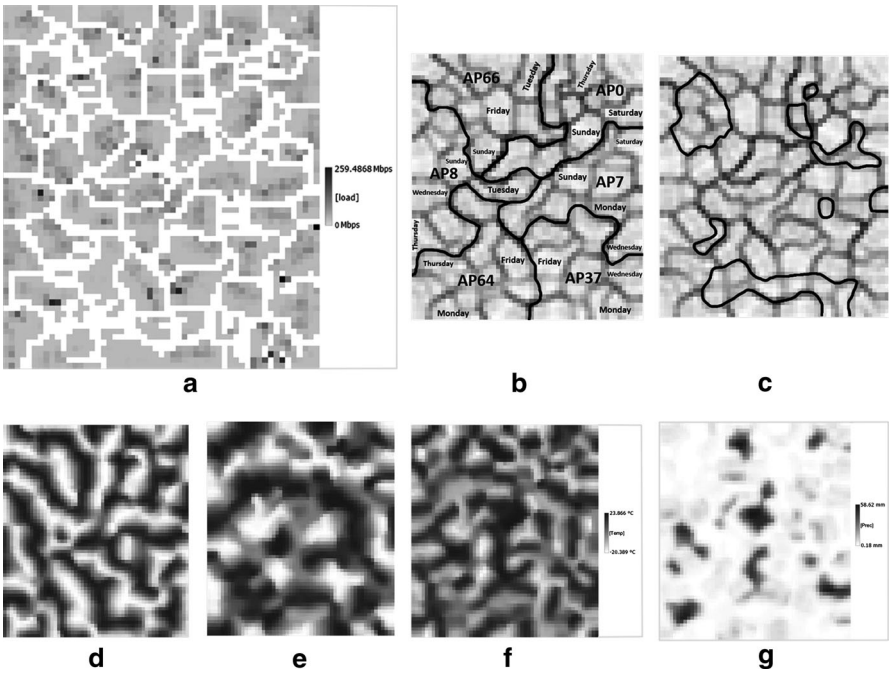

Fig. 7 The map from which the best results were obtained from the **a** load, **b** access points and days, **c** holidays, **d** time, **e** week of the year, **f** temperature and **g** precipitation points of view. The marked areas in **c** refer to the holiday clusters while in **d–f**, and **g** the higher the value of the respective parameter, the darker the area on which the data sample is mapped

In other words, using a SOM is expected to enhance Big Data management in terms of (a) handling data from different sources, composing and treating them as data samples; (b) decreasing the amount of data during the transition of multiple dimensions to two when representing the multi-dimensional data on 2D maps; and (c) decreasing the storing requirements since a representative sample of the data (only the most informative instances, i.e., instances that are very different to the currently observed ones) along with the clusters of the map are enough to maintain all the information.

The data that were exploited for this research were initially 3,994,560 unstructured numbers per year of the dataset. These data, during the pre-processing phase of the mechanism were organized in data samples, i.e., were transformed in 3,994,560/ $19 = 210{,}240$ structured data of 19 dimensions each per year of observations. When the information carried by these data samples was incorporated in the SOM of 928 cells, i.e., when the data were mapped on the SOM, the only data that needed to be stored for maintaining all the information were the 928 vectors of the cells of the map.

Thus, the mechanism is capable of pre-processing and exploiting Big Data (in terms of disparity). Moreover, the dimensions of the data can be reduced from 19 to 2 during their visualization on the map. Last but not least, given the fact that the best performance was received by the map that was created using the data of 2 years and its size was equal

Table 4 Summary of the results obtained from the three scenarios

Study	Denormalized MAE (Mbps)
Main parameters + weather conditions	1.198
Main parameters + (bank) holidays	25.652
All parameters (current paper)	26.030

to 928 cells, the mechanism offers the capability of reducing the size of the data that need to be stored from $2 \times 210{,}240 = 420{,}480$ to 928, i.e., approximately 453 times.

5.2.4 Comparison

Our first study related to the exploitation of Big Data and a SOM for predicting network load tested the behavior of the Wi-Fi hotspots on a university campus with respect to the time, the day, the area, the weather, and the load that had been observed in the past. Using past observations of the network with respect to these parameters, the mechanism learned the network behavior and was capable of predicting the load that would be encountered given the time, the day, and the weather conditions in which one is interested. Another scenario, where the data of [28] are combined with information related only to (bank) holidays has also been examined in the past. Therefore, in that scenario, the mechanism learned and predicted the load under the context of the access point, the time, the day, the week, and the holidays. Eventually, the scenario examined in this paper combines all the parameters considered in the other 2 scenarios aiming to take into account more variables that may influence the network load. This section compares these 3 scenarios in order to reach valuable conclusions with respect to the most efficient combination of variables that can predict the network load.

To begin with, Table 4 summarizes the best obtained results from the 3 scenarios in the case of unseen data in terms of denormalized MAE.

According to the table, the best results were obtained when apart from the main parameters, i.e., the area, the time, the day, and the load, only weather was taken into account. However, as the datasets used in the first scenario (main parameters + weather conditions) and the last 2 scenarios were different, the outcome needs to be re-validated with the same dataset for all three scenarios. Moreover, user preferences may not be equally influenced in all geographical areas by all these factors, e.g., in northern cities/countries, the users' intention to use the network may be less influenced by an upcoming (bank) holiday than in southern cities/countries.

6 Conclusions

One challenge in the area of Big Data is the design of mechanisms that can derive meaningful, as well as useful information for the stakeholders or other intermediate systems and mechanisms in an efficient and automated fashion. Often, another challenge is the storage of such data which, depending on their volume, might be

prohibitive. In the latter case, the effective removal of redundancy (i.e., compression) either lossy or lossless depending on the problem is a solution.

Regarding telecommunication networks, huge amounts of data can be collected when, for instance, traffic-related parameters are measured on a high granularity, for long time periods and for multiple nodes. The NO is then forced to maintain measurements in a certain time-window and discard the ones that fall out of it.

As a result, network operators may collect large amounts of historical data related to the load of the network that need to get organized in less data while maintaining or even augmenting/enhancing the quality of the information. In other words, they need to be transformed in higher level information and thus, become more easily usable by the network operators.

This paper proposed a mechanism that clusters such data with respect to the area, the time, the day, the week of the year, the mean temperature, the precipitation, and information regarding the (bank) holidays in order to minimize their volume with no loss of their information. The mechanism utilizes the data and builds knowledge upon them with respect to their relation to each other, e.g., how does the day of the week influence the network load? The built knowledge is then exploited by a second mechanism in order to predict/infer the load that will be required from the users in the future at a specific area.

The evaluation of the tool that combines qualitative and quantitative results and results with respect to the management of Big Data revealed that the proposed tool is capable of learning the traffic pattern with deviations up to 0.000553 in terms of MSE while it reduces data up to 453 times.

Acknowledgments The research leading to these results has been performed within the UniverSelf project (www.univerself-project.eu) and received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under Grant Agreement No. 257513.

References

1. Lohr, S.: The Age of Big Data. New York Times. http://www.nytimes.com/2012/02/12/sunday-review/big-datas-impact-in-the-world.html?_r=1 (2012). Accessed 20 February 2012
2. European Committee for Standardization Website. http://www.cen.eu/cen/News/Newsletter/Pages/default.aspx. Accessed 17 October 2012
3. Kohonen, T.: Self-Organizing Maps. Series in Information Sciences, 2nd ed., vol. 30. Springer, Heidelberg (1997)
4. Kohonen, T.: The self-organizing map. Neurocomputing **21**, 1–6 (1998)
5. Gianchandani, E.: Obama Administration Unveils $200 M Big Data R&D Initiative. The Computing Community Consortium Blog. http://www.cccblog.org/2012/03/29/obama-administration-unveils-200m-big-data-rd-initiative/ (2012). Accessed 26 June 26 2012
6. Weiss, R., Zgorski, L.-J.: Obama Administration Unveils "Big Data" Initiative: Announces $200 Million in New R&D Investments. Office of Science and Technology Policy Executive Office of the President. http://www.whitehouse.gov/sites/default/files/microsites/ostp/big_data_press_release.pdf (2012). Accessed 4 July 2012
7. Rouse, M.: Definition: In-memory Database. WhatIs.com. http://whatis.techtarget.com/definition/in-memory-database (2012). Accessed 24 June 2013
8. Vizard, M.: The Rise of In-Memory Databases. shashdot.org. http://slashdot.org/topic/datacenter/the-rise-of-in-memory-databases/ (2012). Accessed 30 June 2013
9. Gianchandani, E.: NIST's BIG DATA Workshop: Too Much Data, Not Enough Solutions. The Computing Community Consortium Blog. http://www.cccblog.org/2012/06/21/nists-big-data-workshoptoo-much-data-not-enough-solutions/ (2012). Accessed 26 June 2012

10. Manyika, J., Chui, M., Bughin, J., Brown, B., Dobbs, R., Roxburgh, C., Hung Byers, A.: Big Data: The Next Frontier for Innovation, Competition, and Productivity. McKinsey Global Institute, McKinsey&Company, Insights & Publications. http://www.mckinsey.com/Insights/MGI/Research/Technology_and_Innovation/Big_data_The_next_frontier_for_innovation (2011). Accessed 2 July 2012
11. Sharma, A., Pejovic, V.: Towards, Real-Time Prediction of Network Traffic Load. University of Computer Science, Department of Computer Science. http://www.cs.ucsb.edu/~veljko/docs/cs290d_paper.pdf (2009). Accessed 9 July 2012
12. Papadopouli, M., Raftopoulos, E., Shen, H.: Evaluation of short-term traffic forecasting algorithms in wireless networks. In: Proceedings of the 2nd Conference on Next Generation Internet Design and Engineering, 3–5 April 2006, Valencia, Spain, pp. 102–109. http://www.ics.forth.gr/mobile/publications/ngi06.pdf (2006). Accessed 5 July 2012
13. Nae, V., Prodan, R., Fahringer, T.: Neural network-based load prediction for highly dynamic, distributed online games. In: Proceedings of the 14th International European-Parallel Conference on Parallel Processing (Euro-Par '08). Lecture Notes in Computer Science (LNCS), vol. 5168, pp. 202–211 (2008)
14. Fillatre, L., Marakov, D., Vaton, S.: Forecasting seasonal traffic flows. In: Proceedings of the Workshop on QoS and Traffic Control, Paris, France (2005)
15. Bantouna, A., Tsagkaris, K., Stavroulaki, V., Poulios, G., Demestichas, P.: Learning techniques for context diagnosis and prediction in cognitive communications. In: Zhang H., Grace D. (eds.) Cognitive Communications: Distributed Artificial Intelligence (DAI), Regulatory Policy & Economics, Implementation, pp. 231–256. Wiley, Hoboken (2012)
16. Qian-Mu, L., Man-Wu, X., Feng-Yu, L.: Network congestion prediction based on RFNN. In: Proceedings of the IEEE International Conference on Systems, Man and Cybernetics 2005 (2005). doi:10.1109/ICSMC.2005.1571477
17. Lim, A. O., Ab-Hamid, K.: Kalman prediction method for congestion avoidance in ATM networks. In: Proceedings of the Technology, Education and Networking Conference (TENCon) 2000, vol. 1, pp. 346–351 (2000)
18. Valero, S., Aparicio, J., Senabre, C., Ortiz, M., Sancho, J.: Comparative analysis of Self Organizing Maps vs. multilayer perceptron neural networks for short-term load forecasting. In: Proceedings of the International Symposium Modern Electric Power Systems (MEPS), pp. 1–5 (2010)
19. Aquino, I., Perez, C., Chavez, J. K., Oporto, S.: Daily load forecasting using quick propagation neural network with a special holiday encoding. In: International Joint Conference on Neural Networks (IJCNN) 2007, pp. 1935–1940 (2007)
20. White paper: Pattern Recognition and Learning Based Self Optimizing Networks. Reverb intelligent SON solutions. http://www.reverbnetworks.com/white-papers (2012). Accessed 9 August 2012
21. Zhou, J., Fu, Y.: Clustering High-Dimensional Data Using Growing SOM. Advances in Neural Networks—ISNN 2005. Lecture Notes in Computer Science (LNCS), vol. 3497, pp. 63–68 (2005)
22. Berglund, E., Sitte, J.: The parameterless self-organizing map algorithm. IEEE Trans. Neural Netw. **17**(2), 305–316 (2006)
23. Sarasamma, S.T., Zhu, Q.A., Huff, J.: Hierarchical kohonen net for anomaly detection in network security. IEEE Trans. Syst. Man Cybern. Part B Cybern. **35**(2), 302–312 (2005)
24. Dittenbach, M., Merkl, D., Rauber, A.: Growing hierarchical self-organizing map. In: Proceedings of the International Joint Conference on Neural Networks (IJCNN '00), vol. 6, pp. 15–19 (2000)
25. Kuremoto, T., Komoto, T., Kobayashi, K., Obayashi, M.: Parameterless-growing-SOM and its application to a voice instruction learning system. J. Robot. (2010). doi:10.1155/2010/307293
26. CRAWDAD database: http://crawdad.cs.dartmouth.edu/index.php. Accessed 23 July 2012
27. The Île Sans Fil website: http://www.ilesansfil.org/. Accessed 10 August 2012
28. CRAWDAD database: Trace ilesansfil/wifidog/session/04_07 (v. 2007-08-27). http://crawdad.cs.dartmouth.edu/meta.php?name=ilesansfil/wifidog#N1006B. Accessed 10 August 2012
29. Wunderground.com Website: http://www.wunderground.com/history/. Accessed 15 August 2012

Author Biographies

Aimilia Bantouna is a research engineer in Telecommunication Networks and Integrated Services Laboratory (TNS Lab) and a PhD candidate in Machine-Learning Techniques of Future Networks at the University of Piraeus Research Center (UPRC). Her research interests are machine learning techniques,

and most particularly unsupervised ones, that can be applied for enhancing decision making processes and trust in autonomic systems.

Giorgos Poulios is an MSc student in Techno-economic Management and Security of Digital Systems in the Digital Systems Department of University of Piraeus. Moreover, since 2011, he is working as a Researcher in Telecommunication Networks and Integrated Services Laboratory (TNS Lab) focusing on machine learning, prediction of upcoming events in networks and software development over different.

Kostas Tsagkaris is a senior research engineer and adjunct lecturer at the department of Digital Systems at the University of Piraeus. His research work and interests are in the design, management and optimization of fixed/wireless networks and services, self-organizing/autonomic networks, machine-learning and software engineering. He has more than 130 publications in the areas above.

Panagiotis Demestichas is a professor at the department of Digital Systems at the University of Piraeus, Greece since 2002. He has 20 years of experience in national and international R&D projects. His research interests include the design and performance evaluation of wireless and fixed broadband networks, software engineering, service/network management, algorithms, complexity theory, and queuing theory.