New Models for Long-Term Internet Traffic Forecasting Using Artificial Neural Networks and Flow Based Information

Márcio L. F. Miguel
Departamento de Redes e Serviços IP
COPEL Telecomunicações S.A.
Curitiba - PR, Brazil
marcio.miguel@copel.com

Manoel C. Penna, Julio C. Nievola, Marcelo E. Pellenz
PPGIa, PUCPR
Curitiba - PR, Brazil
{penna,nievola,marcelo}@ppgia.pucpr.br

***Abstract*—This paper investigates the use of ensembles of artificial neural networks in predicting long-term Internet traffic. It discusses a method for collecting traffic information based on flows, obtained with the NetFlow protocol, to build the time series. It also proposes four traffic forecasting models based on ensembles of TLFNs (Time-Lagged FeedFoward Networks), each one differing from the others by the way it reads the training data and by the number of artificial neural networks used in the forecasts. The proposed prediction models are confronted with the classic method of Holt-Winters, by comparing the mean absolute percentage error (MAPE) of the forecasts. It is concluded that the proposed models perform well, and can be considered a good option for planning network links that transport Internet traffic.**

***Index Terms*—Time series forecasting, Artificial neural network, Internet traffic forecasting, Internet data flows.**

I. Introduction

Traffic engineering in telecommunications networks is a traditional and well explored research field. However, the study of traffic planning for the Internet is still limited, because of the inherent difficulties on obtaining the traffic matrix for the global network. Forecasting the traffic volume for specific links is usually done based on customers' future needs and on the rate of bandwidth sharing among ISPs. These empirical approaches are not adequate for planning large networks, and alternative methods shall be investigated to support the planning process and to reduce the forecast errors.

Time series forecasting (TSF) is a subject largely discussed in literature, which deals with the prediction of the future behavior of chronologically ordered variables [1]. Before TSF can be applied the historical data must be collected, and a forecasting model must be selected. The presence or absence of specific components in the time series, such as trend, seasonality, and non-linearity is important in selecting the model. Several approaches have been considered for TSF models, such as the method of Holt-Winters (HW) [1], the autoregressive integrated moving average (ARIMA) [2] and artificial neural networks (ANN) [3].

978-1-4673-0269-2/12/$31.00

In the past, several studies have demonstrated the predictability of network traffic by different TSF models, but most of them addressed short term forecast horizons. The study in [4] introduced a methodology for long term forecast of Internet traffic based on wavelet multi-resolution analysis and linear time series model. The authors presented a methodology to predict future needs of network resources, which can handle long term trends and strong non-linearity in multiple time scales. It was concluded that the resulting model can provide accurate forecasts for at least six months ahead. The study in [5] compared TFS models for network traffic prediction based on autoregressive (AR), moving average (MA), autoregressive moving average (ARMA), and ARIMA models. The goal was to investigate suitability of these models in expressing the nature of future traffic. The authors evaluated if the models satisfy the stationary assumption by using Auto Correlation Function (ACF) and Partial Auto Correlation Function (PACF). They concluded the AR model can produce the best predictions when network traffic is classified on a daily basis.

TSF models based on ANN have been proposed to explore the non-linearity characteristics of network traffic. In [6] the authors proposed an approach to detect trends of available bandwidth in communication links. They used classical traffic traces organized by protocol type (TCP, UDP and ICMP) for training the ANNs. Individual neural networks were constructed to predict each type of network traffic, and the results indicated that the ANN can accurately capture the network traffic efficiently, with superior performance to NWS. The prediction of TCP/IP traffic has been evaluated in [7]. Several experiments were performed over real-world data, considering different time scales and forecasting horizons. The results showed that the ANN approach is competitive when compared with HW and ARIMA models. In [8], the same research team has evaluated the ANN approach by using univariate and multivariate strategies. Historical data was collected from the links of the UK academic network backbone (UKERNA). Forecasting is achieved by a Time Lagged Feedforward Network (TLFN) in an arrangement that takes as input a sliding window over the time series. A sliding window is the set of time lags used to build a forecast and is used to select the training data. In the univariate approach a single sliding window is used whereas multiple sliding windows are used in a multivariate setting.

More recently the study in [9] analyzed time series of Internet requests made to a workstation. The authors used wavelet filters based on multi-resolution analysis along with the Seasonal Autoregressive Moving Average (SARIMA) to forecast the volume of network traffic. The use of filters is based on the principle that most of the network traffic signals have noise, thus filtered signals should produce better forecasting. The authors compared the results with simple SARIMA, obtaining more accurate forecasts for filtered data. The study in [10] presents a comprehensive comparison of TSF methods. Experiments evolving ARIMA models and models based on neural networks ensembles (NNE) models were conducted over Internet data collected from the UK academic network backbone. It was concluded that both ARIMA and NNE produce the best results for data scales of five minutes and one hour, and that HW produce the best results at the daily scales.

Most of the previous study has obtained historical data for Internet traffic forecasting by using the Simple Network Management Protocol (SNMP) to measure the number of bytes traversing the interfaces of network equipments. Depending on the Management Information Base (MIB), the measurements provide only information about the number of bytes, what can be a limitation when there is a need for information in other layers of the protocol stack. To circumvent this limitation, we adopted another strategy for collecting traffic data, based on flow monitoring, which allows the separation of data traffic according to its origin and destination. In our study, the traffic volume by flow is organized in time series used to build ensembles of TLFNs for long-term forecast. The contribution of this paper is twofold: (i) the definition of a method for collecting TSF data from Internet data flows, and (ii) the proposition of new long-term forecasting models based on multiple settings of TLFNs. The data collection method is applied to real Internet traffic of an ISP, and the TFLN arrangements are validated by comparison to the HW method.

The remainder of this paper is structured as follows. Section II presents the method for colleting and formatting the time series based on Internet data flows. The forecasting models are given in Section III. The experiments and the results are presented and discussed in Section IV. In section V, the closing conclusions are drawn.

II. Time Series Based on Internet Data Flows

The Internet is constructed from the interconnection of Autonomous Systems (AS), which are so called because their administrators define the internal routing policies. To characterize the traffic between ASs we use the NetFlow protocol, the de facto standard used in the Internet for flow management [11].

A flow is defined as a sequence of unidirectional data with common characteristics of origin and destination within a certain time-frame. For IP traffic, flow characteristics are defined by the IETF RFC 2123 [12]. A new flow is created when a packet not belonging to an existing flow is received. A flow expires when it doesn't transmit any packet during 15 seconds, when its duration exceeds 30 minutes, or when the corresponding TCP connection is closed. Only one information record is generated independently of the number of packages that formed a flow.

The measurement of flows is performed by a NetFlow server in cooperation with the network elements as shown in Figure 1. The flows are generated by running the Netflow protocol at edge routers and are subsequently directed to the NetFlow server, which receives and stores the data in local files. Among the open source NetFlow collectors, the most widely used are the Flow-tools [13] and Nfdump [14]. We select the NetFlow version 5 protocol for two reasons: first, because it provides flow information organized by AS of origin and destination, and second, because it allows the use of Nfsen, a package of programs for collecting and analyzing NetFlow data. Two issues should be resolved before building the time

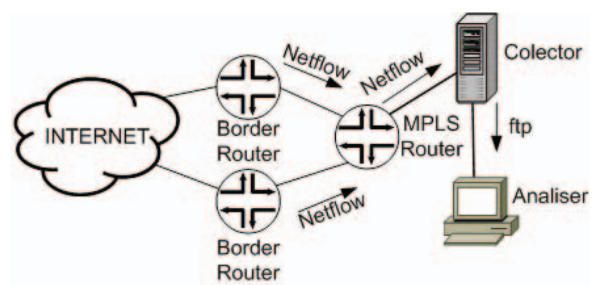

Figure 1. Diagram of flow collection.

series: the choice of time scale and the selection of a set of representative ASs. The daily scale is preferred because daily traffic represents the more pronounced seasonality in Internet traffic [4]. The Internet is composed of thousands of ASs but there is a concentration of traffic from a few of them. Thus, only a treatable quantity of ASs should be selected, and a preliminary trial shall be run to select them.

The procedure for time series construction is as follows: (i) run a trial to obtain the volume of traffic per AS, with respect to the ISP. (ii) Select the representative ASs by applying the 80-20 rule with respect to the observed volume of traffic. This rule, also known as the Pareto's principle, states that, for many events, roughly 80% of the effects come from 20% of the causes. (ii) Accumulate the volume of traffic per AS in intervals of five minutes, producing 288 samples per day. (iii) Choose the 99-percentile of the 288 samples to represent the traffic volume of the corresponding day.

The limitation of daily traffic to the 99th percentile of traffic accumulated in 5 minute intervals is a strategy for filtering traffic noise. This can be considered a conservative approach because it takes the peak traffic as representative for the traffic of one day. The filtering measure avoids possible outliers caused by traffic spikes during less than 15 minutes.

III. Forecasting Models

Time can be incorporated into the design of artificial neural networks implicitly or explicitly. The implicit representation of time consists in adding a structure of short-term memory in the input layer of a static ANN. The resulting configuration is a TLFN, whose training model can be obtained by a delay window that works as a short-term memory. The delay window consists of p units of delay inserted in $(p+1)$ inputs to the network, as illustrated in Figure 2. For each sample from the time series, there is a shift of position on the values present in the entry. This mechanism ensures that the reading order of the series is preserved, retaining the time dimension and the dependence between samples [15]. The transfer of time series data to the inputs of the network is performed by using a sliding window of size $p+1$, covering $x(k)$, $x(k-1)$, ..., $x(k-p)$, which drops the last transferred value and includes the next input, maintaining the same number of entries. The sliding window can select elements from the training data in multiple different ways. For example, if the sliding window is to read sequentially the training data, it can be described by a list of subsequent numbers, e.g. $< 1, 2, 3, 4, 5, 6, 7 >$ with $p+1 = 7$. If the reading is not sequential, the sliding windows can specify the specific coverage by the relative positions within a list, e.g. $< 1, 8, 15, 22, 29, 36, 43 >$, with $p+1 = 7$ and $d = 7$, where d represents the distance between samples. The models proposed in our work are based in the structure of the training data and in the number of neural networks used in forecasts.

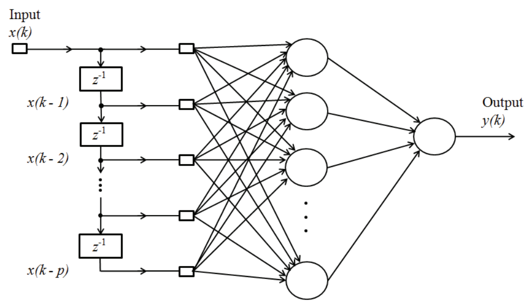

Figure 2. Time-lagged Feedforward Network. [15]

The following notation is used for presenting the structure of the training data: Vectors are denoted by bold lower case. Matrices are denoted by upper case and are specified by their row vectors. Index of vector elements are defined within brackets, by expressions separated by colon or semicolon. When colon appears once, the first expression defines the initial value of the vector index and the second its final value (with unitary increment). When colon appears twice, the first expression defines the initial value, the second the increment value, and the third the final value. The semicolon is used to index specific elements.

Let $\mathbf{h}$ be the array that stores the time series of an autonomous system, with $w = |\mathbf{h}|$ being the number of elements in $\mathbf{h}$. Two new vectors are formed from $\mathbf{h}$, the vector $\mathbf{t} = [t_1, \ldots, t_{z_1}] = \mathbf{h}[1 : z_1]$, with $z_1 = 2w/3$; and the vector $\mathbf{v} = [v_1, \ldots, v_{z_2}] = \mathbf{h}[z_1 + 1 : z_2]$. They correspond to the training time series, $\mathbf{t}$, and the validation time series, $\mathbf{v}$.

Let $x(k)$ be an element of a time series and $y(k)$ the next element to be forecasted. $y(k)$ is used in the training phase as the output of a training sample, as showed in Figure 2. Several arrays of the form $[x(k-p), x(k-p+1), \ldots, x(k), y(k)]$ are provided to the training algorithm, each one corresponding to a training sample. The training data is organized in the $\mathbf{D}$ matrix where each row is a vector corresponding to a training sample. For example, in equation (1), $\mathbf{d}_1$ corresponds to the first training sample.

$$\mathbf{D} = \begin{bmatrix} \mathbf{d}_1 \\ \vdots \\ \mathbf{d}_m \end{bmatrix} = \begin{bmatrix} d_{11} & \ldots & d_{1n} \\ \vdots & \ddots & \vdots \\ d_{m1} & \ldots & d_{mn} \end{bmatrix} \tag{1}$$

The elements forming a row in $\mathbf{D}$ are selected differently from $\mathbf{t}$, according to the model. The building of $\mathbf{D}$ can be seen as the selection of elements in $\mathbf{t}$ according to a sliding window of size $n = p+2$. The elements in each line are those selected from $\mathbf{t}$, subject to the selection criterion specified by the window. The window moves towards the evolution of time at each training step, dropping from the training input the oldest element and adding the next output, as showed in Figure 3. In the example $n = 7$, and the sliding window covers seven consecutive elements $< 1, 2, 3, 4, 5, 6, 7 >$, selecting $\mathbf{d}_9 = [t_9, \ldots, t_{15}]$ as the ninth training sample, $\mathbf{d}_{10} = [t_{10}, \ldots, t_{16}]$ as the tenth, and so on.

More than one TLFN shall be trained, depending on the model. After the training stage, the TLFN ensemble is used to build another time series, stored in the vector $\mathbf{f}$, which contains the results of long-term forecasts. The forecast vector, $\mathbf{f}$, is compared with the validation vector, $\mathbf{v}$, in order to assess the model accuracy. The study in [7] uses the same concepts

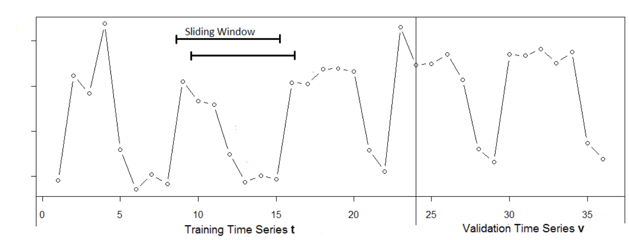

Figure 3. Sliding window.

to derive an ANN based forecast model, where the TLFN is trained for a forecast horizon of size one. The training data is selected by a sequential sliding window and the i-th row of $\mathbf{D}$ is computed according to equation (2). The network is stimulated by a set of $n-1$ consecutive components of a time series to make a prediction. The number of rows, m, in $\mathbf{D}$ is given by (3). It is important because it corresponds to the number of possible training samples.

$$\mathbf{d}_i = \mathbf{t}\left[i : (n-1+i)\right] \tag{2}$$

$$m = |\mathbf{t}| - n + 1 \tag{3}$$

In the following we propose four long-term forecasting models based on ensembles of TLNFs. Each model differs from the others by the way it reads the training data and by the number of ANNs included in its ensemble. Two of the proposed models use the forecast results to provide long-term forecasts. Also, two of the proposed models consider the seasonality in time series.

A. R-PR Model

The first model is referred as forecast horizon of size R with prevision reuse (R-PR). It extends the previously described model to a forecast horizon of R steps by training one TLNF with the same procedure (i.e., the size of the ensemble is 1). To perform a forecast in a horizon of size R, the trained network is used multiple times in a sequence of R prevision steps. At each new step, the previous TLFN output is used as the new input, with the oldest input being discarded.

The $\mathbf{f}$ vector is built as follows: The last $n-1$ elements of $\mathbf{t}$ are used as the input to predict the first element in $\mathbf{f}$. Then the first value in $\mathbf{f}$ is used together with the last $n-2$ elements in $\mathbf{t}$ to predict the second element, which is in turn used along with $n-3$ last elements to predict the third element, and so on.

B. R-NPR Model

In the second model, referred as R-NPR, several TLNFs are trained to perform long term forecasts, each one performing forecasts in a different time horizon. Thus, for a forecast horizon of size R, we need to train an ensemble of R different networks $\{N_1, \ldots, N_R\}$. The training procedure is similar to that previously described, except that the element used as the output, $y(k)$, is r positions ahead of $x(k)$ in $\mathbf{t}$. The corresponding sliding window is $< 1, 2, 3, \ldots, n-1, n-1+r >$. Let $\mathbf{D}_r$ be the training matrix for the network that perform forecasts in r steps ahead. The i-th row in $\mathbf{D}_r$ is given by equation (4) and the number of rows is given by (5). An example of $\mathbf{D}_4$, with $n = 6$ and $|\mathbf{t}| = 15$, is given by equation (6). The considered sliding window is $< 1, 2, 3, 4, 5, 6 >$. Table I illustrates the use of $\mathbf{D}_4$ example as the training data for R-NPR model.

$$\mathbf{d}_i^r = \mathbf{t}\left[i : n-1; n-1+r\right] \tag{4}$$

$$m = |\mathbf{t}| - r - n + 2 \tag{5}$$

$$\mathbf{D}_4 = \begin{bmatrix} \mathbf{d}_1^4 \\ \mathbf{d}_2^4 \\ \mathbf{d}_3^4 \\ \mathbf{d}_4^4 \\ \mathbf{d}_5^4 \\ \mathbf{d}_6^4 \\ \mathbf{d}_7^4 \end{bmatrix} = \begin{bmatrix} t_1 & t_2 & t_3 & t_4 & t_5 & t_9 \\ t_2 & t_3 & t_4 & t_5 & t_6 & t_{10} \\ t_3 & t_4 & t_5 & t_6 & t_7 & t_{11} \\ t_4 & t_5 & t_6 & t_7 & t_8 & t_{12} \\ t_5 & t_6 & t_7 & t_8 & t_9 & t_{13} \\ t_6 & t_7 & t_8 & t_9 & t_{10} & t_{14} \\ t_7 & t_8 & t_9 & t_{10} & t_{11} & t_{15} \end{bmatrix} \tag{6}$$

The network N_r is used to forecast r steps ahead. We need an ensemble of size $|\mathbf{v}|$ to build the vector $\mathbf{f}$. Each network is stimulated by the $n-1$ last elements in $\mathbf{t}$. The forecast produced by N_1 is stored in f_1, the forecast produced by N_2 in f_2, and so on.

Table I
USE OF MATRIX $\mathbf{D}_4$ FOR R-NPR MODEL

Sample	$x(k-4)$	$x(k-3)$	$x(k-2)$	$x(k-1)$	$x(k)$	$y(k)$
1	t_1	t_2	t_3	t_4	t_5	t_9
2	t_2	t_3	t_4	t_5	t_6	t_{10}
3	t_3	t_4	t_5	t_6	t_7	t_{11}
4	t_4	t_5	t_6	t_7	t_8	t_{12}
5	t_5	t_6	t_7	t_8	t_9	t_{13}
6	t_6	t_7	t_8	t_9	t_{10}	t_{14}
7	t_7	t_8	t_9	t_{10}	t_{11}	t_{15}

C. SR-PR Model

The third model, referred as SR-PR, considers the size S of the seasonality period. The vector $\mathbf{h}$ is split in S subseries, one for each relative position in the seasonality period. One TLNF is trained for each subseries, producing an ensemble with S networks $\{N_1, \ldots, N_S\}$. The sliding window selects elements having the same relative position in the seasonality period. Let $\mathbf{D}_s$ be the training matrix for s-th position in the seasonality period. The i-th row of $\mathbf{D}_s$ is given by equation (7). The index g of the last element from $\mathbf{t}$ that can be present in $\mathbf{D}_s$ is given by (8). The number of rows (m) in $\mathbf{D}_s$ is given by (9). This model introduces an important limitation in the number of training samples, because only points at the same relative position in the seasonality period can be used. An example of $\mathbf{D}_3$, with $n = 6$, $S = 7$ and $|\mathbf{t}| = 90$, is given by equation (10). The considered sliding window is $< 1, 8, 15, 22, 29, 36 >$.

$$\mathbf{d}_i^s = \mathbf{t}\left[(i-1)S + s : S : (n-1)S + (i-1)S + s\right] \tag{7}$$

$$g = |\mathbf{t}| - \mathrm{mod}\left(\frac{|\mathbf{t}| - s}{S}\right) \tag{8}$$

$$m = \mathrm{ceil}\left(\frac{g - S(n-1)}{S}\right) \tag{9}$$

$$\mathbf{D}_3 = \begin{bmatrix} \mathbf{d}_1^3 \\ \mathbf{d}_2^3 \\ \mathbf{d}_3^3 \\ \mathbf{d}_4^3 \\ \mathbf{d}_5^3 \\ \mathbf{d}_6^3 \\ \mathbf{d}_7^3 \end{bmatrix} = \begin{bmatrix} t_3 & t_{10} & t_{17} & t_{24} & t_{31} & t_{38} \\ t_{10} & t_{17} & t_{24} & t_{31} & t_{38} & t_{45} \\ t_{17} & t_{24} & t_{31} & t_{38} & t_{45} & t_{52} \\ t_{24} & t_{31} & t_{38} & t_{45} & t_{52} & t_{59} \\ t_{31} & t_{38} & t_{45} & t_{52} & t_{59} & t_{66} \\ t_{38} & t_{45} & t_{52} & t_{59} & t_{66} & t_{73} \\ t_{45} & t_{52} & t_{59} & t_{66} & t_{73} & t_{80} \end{bmatrix} \tag{10}$$

Each trained network is used to forecast one point ahead for the corresponding position within the seasonality period. For example, the network N_3 is used to forecast one point ahead for the third position of the seasonality period. To perform a forecast in a horizon of size R, the trained network is used multiple times in R prevision steps, as in R-PR model. The vector $\mathbf{f}$ is built as follows: The last row in $\mathbf{D}_s$ is the input for the first forecast with network $\mathbf{N}_s$, that is, the first forecast for the s-th relative position in the seasonality period. Subsequent forecasts are performed reusing the last previous forecast, as in R-PR model.

D. SR-NPR Model

The fourth model (SR-NPR) is constructed by training an ensemble of $S \cdot R$ networks. The matrix $\mathbf{D}_{s,r}$ is used to train the network responsible for previsions with a forecast horizon of size r ($r = 1, \ldots, R$) for the s-th position in the seasonality period ($s = 1, \ldots, S$). The i-th row in $\mathbf{D}_{s,r}$ is given by equation (11). The index g of the last element from $\mathbf{t}$ that can be present in $\mathbf{D}_{s,r}$ is given by (12). The number of rows in $\mathbf{D}_{s,r}$ is given by (13).

$$\begin{aligned}\mathbf{d}_i^{s,r} = \mathbf{t}[&(i-1)S+s : S : (n-i-3)S+s \\ &; (n+i-2)S+s+r]\end{aligned} \tag{11}$$

$$g = |\mathbf{t}| - \mathrm{mod}\left(\frac{|\mathbf{t}| - s - r}{S}\right) \tag{12}$$

$$m = \mathrm{ceil}\left(\frac{g - S(n-1)}{S}\right) \tag{13}$$

An example of $\mathbf{D}_{7,4}$, with $n = 6$, $S = 7$, and $|\mathbf{t}| = 90$, is given by equation (14). The considered sliding window is $< 1, 7, 14, 21, 28, 35 >$. The network $N_{s,r}$ is used to make a prevision with a forecast horizon of size r for the s-th position of the seasonality period. The number of ensembles necessary to build $\mathbf{f}$ is $(|\mathbf{v}|/s \cdot r)$. The input for prevision $f_{s \cdot r}$ in $\mathbf{f}$ is the last row in $\mathbf{D}_{s,r}$.

$$\mathbf{D}_{7,4} = \begin{bmatrix} \mathbf{d}_1^{7,4} \\ \mathbf{d}_2^{7,4} \\ \mathbf{d}_3^{7,4} \\ \mathbf{d}_4^{7,4} \\ \mathbf{d}_5^{7,4} \\ \mathbf{d}_6^{7,4} \\ \mathbf{d}_7^{7,4} \end{bmatrix} = \begin{bmatrix} t_7 & t_{14} & t_{21} & t_{28} & t_{35} & t_{46} \\ t_{14} & t_{21} & t_{28} & t_{35} & t_{42} & t_{53} \\ t_{21} & t_{28} & t_{35} & t_{42} & t_{49} & t_{60} \\ t_{28} & t_{35} & t_{42} & t_{49} & t_{56} & t_{67} \\ t_{35} & t_{42} & t_{49} & t_{56} & t_{63} & t_{74} \\ t_{42} & t_{49} & t_{56} & t_{63} & t_{70} & t_{81} \\ t_{49} & t_{56} & t_{63} & t_{70} & t_{77} & t_{88} \end{bmatrix} \tag{14}$$

IV. Experiments and Results

In the evaluation scenario, two border routers collect traffic flow information through the NetFlow protocol version 5. These routers are connected out of the ISP via 1 Gbps links, connecting it to two relay traffic providers, to the point of exchange of regional traffic, and to the point of exchange of domestic traffic. Flows collected at edge routers are forwarded to a server running the nfcapd daemon. Before being transmitted, the flows are sampled at a rate of $1 : 1000$. On the server, binary data is converted to text mode and the traffic volume is accumulated in five minutes intervals.

Preliminary tests conducted in February 2009 showed that about 80% of all traffic received by the ISP is concentrated in 50 out of $26,120$ ASs (the 80-20 rule). An experiment that ran from May to August 2010 selected the set of representatives ASs. It was observed that the 50 most loaded out of $27,089$ ASs were responsible for $78,45\%$ of the traffic ASs. Fifty traces were then constructed in a daily scale, with entries containing the 99-percentile of the values accumulated in five minutes intervals. Each trace is formed by a set of 435 samples collected during the period between September 23th, 2010 and December 1st, 2010. Therefore, we have $|\mathbf{h}| = 435$, $|\mathbf{t}| = 290$, and $|\mathbf{v}| = 145$.

Table II
Selected Parameter Values

Model	Number of Input Nodes $(n-1)$	Number of Hidden Nodes	Number of Training Samples
R-PR	7	4	283
R-NPR	10	2	136
SR-NPR	1	6	41
SR-PR	10	2	41

The next step is to select the ANN parameters. It was shown in [16] that two of the most important TLNF parameters for time series forecasting are the number of input nodes and the number of hidden nodes. Few input nodes may provide insufficient information to accurate predictions, while an excessive number increases the possibility of irrelevant inputs. The number of hidden nodes is important because a network with few nodes have limited learning ability. A large number of hidden nodes can lead to over-fitting, making the network specialized in the training data, becoming unable to generalize [16].

The parameters were selected in experiments where each ANN is trained 30 times for each parameter value. For each play, the mean absolute percentage error (MAPE) is computed by comparing $\mathbf{f}$ with $\mathbf{v}$. The average MAPE of the 30 runs is taken as the representative value of prevision errors (RVPE) for a particular AS. Thus, fifty RVPEs were computed (one for each AS), for each parameter value choice. The parameter value was selected as the one with the least mean RVPE value, compared with respect to the 95% confidence interval. The number of inputs nodes was selected from the range 1-11 while the number of hidden nodes was selected by a choice in $\{0, 2, 4, 6\}$. Table II presents the selected values for each model. Given the number of input nodes $(n-1)$, we can compute the worst case number of training samples for each model, according to equations (3), (5), (9) and (13). These values are shown also in Table II. The accuracy of the forecasting models was evaluated through another experiment. The predictions were stored in $\mathbf{f}$, for all AS, as described in the previous section. Comparing $\mathbf{f}$ with $\mathbf{v}$ produces a value for MAPE. The average MAPE of 30 plays is considered the RVPE for the corresponding AS.

Four criteria are used to assess forecasting accuracy: the percentage of times a model outperforms the HW method, the percentage of times a model is equivalent to HW, and the percentage of times a model underperforms HW. RVPE of an ANN model and the MAPE of HW are compared. The comparison is made with respect to the 95% confidence interval of RVPE, which is in its turn computed with the MAPE obtained in each of the 30 runs. The fourth assessment criterion is the percentage of times the forecasts present the RVPE above 200. It was observed that a set of predictions presenting the RVPE above this threshold correspond to unacceptable errors. We therefore consider it important that a prediction model shows

Table III
SUMMARY OF MODEL PERFORMANCE

Model	RVPE of models vs. MAPE of Holt-Winters (%)			% of times RVPE is
	Superior	Equivalent	Inferior	≥ 200
R-PR	18	62	20	2.0
R-NPR	26	30	44	10.0
SR-NPR	18	28	54	10.0
SR-PR	28	54	18	2.0

this behavior in the fewest possible situations.

Figure 4 illustrates the performance assessment of the proposed R-PR method. The figure shows, for each AS, the normalized difference between the RVPE obtained for one AS with its 30 plays and the MAPE obtained with the method of HW. The horizontal axis corresponds to the index of an AS and the vertical axis to the normalized difference. Negative normalized difference corresponds to a better performance of the proposed model compared with the HW method. The R-PR model performed better than the HW method in 18% of the cases, worse in 20%, and equivalently in 62%. A similar analysis was carried out for R-NPR, SR-NPR and SR-PR methods. The R-NPR presented better results for 26% of the cases, worse for 44% and equivalent for 30%. The model SR-NPR presented better performance than the HW method in 18% of the series, worse in 54%, and equivalent in 28%. The model SR-PR showed the best performance among the models tested, with better results than the HW in 28% of the cases, worse in 18%, and equivalent in 54%. Table III

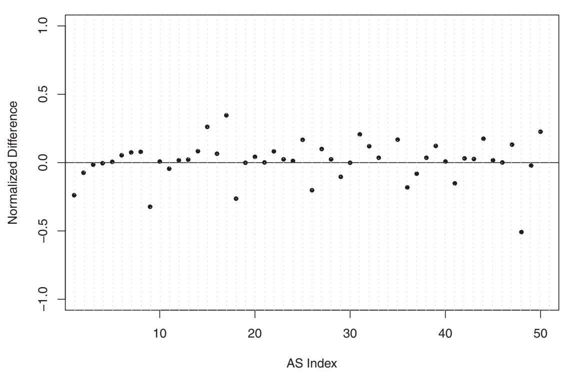

Figure 4. Results obtained with model R-PR.

summarizes the performance of all models with respect to HW method. R-PR can be considered equivalent, R-NPR and SR-NPR underperform, and SR-PR outperforms. The percentage of times the RVPE is greater than 200 is acceptable for R-PR and SR-PR. The percentage of times the MAPE obtained with HW is greater than 200 is the same as R-NPR and SR-NPR, that is 2.0% (not shown in the table). Figure 5 shows a set of graphs of the type boxplot with another comparison between the models. To obtain a boxplot, the RVPE of 30 plays with the same forecasting model is calculated, for every AS. The boxplot of each model is constructed with 50 RVPEs, one for each AS. The purpose of this comparison is to assess the variability of the models' errors according to AS time series. Note that the method of HW, which forecasts the same value for each of the 30 plays, also presents variability with respect to the RVPEs, because of the specific features of the different time series. It is observed in Figure 5 the equivalence

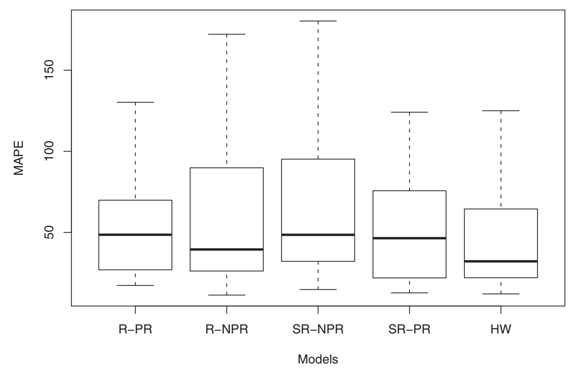

Figure 5. Boxplot for comparison of models.

between the models, which have equivalent mean RVPE. We can also observe the lowest variability for model SR-PR, which is desirable because it shows less dependence of the model with respect to specific features of the time series.

V. CONCLUSIONS

The use of statistical techniques or artificial intelligence in the planning of resources for ISP networks brings a new perspective for predicting the need for investment in equipment and expansion of communication links. This paper presented a methodology for the characterization of traffic between ASs by time-series containing the traffic volume per flow. The study also proposed four models for long-term prediction based on artificial neural networks.

The study assessed the prediction methods proposed by comparing it against the HW method. ANN-based models performed well and are flexible enough to be applied in other areas of time series forecasting, where nonlinearity is shown. The proposed models can be extended by new ways of reading the series during the training stage. Other techniques for parameter selection and optimization of the weights of the connections of the RNA can be studied in future work.

The forecast of the volume of traffic in flows per AS is considered very useful by the ISP planning team. They believe that the knowledge of accurate previsions for traffic volume per AS will help them to better plan the future routing policies, including the negotiation of traffic exchange with neighbors ASs and the determination of more precise estimates of needs in the expansion network links.

REFERENCES

[1] Makridakis S., Weelwright S. and Hyndman R., "Forecasting: Methods and Applications," *John Wiley & Sons*, 1998.

[2] Box G. and Jenkins G., "Time Series Analysis: Forecasting and Control," *Holden Day*, San Francisco, 1976.

[3] Vemuri V. and Rogers R., "Artificial Neural Networks: Forecasting Time Series," *The IEEE Computer Society Press*, Los Alamitos, 1994.

[4] Papagiannaki K., Taft N., Zhang Z. and Diot C., "Long-Term Forecasting of Internet Backbone Traffic," *IEEE Transactions on Neural Networks*, vol. 16, no. 5, pp. 1110-1124, 2005.

[5] Jung S., Kim C. and Chung Y., "A Prediction Method of Network Traffic Using Time Series Models," *Computational Science and Its Applications (ICCSA 2006)*, Lecture Notes in Computer Science, vol. 3982, pp. 234-243, 2006.

[6] Eswaradass A., Su X. and Wu M., "A Neural Network Based Predictive Mechanism for Available Bandwidth," *Proceedings of 19th IEEE International Parallel and Distributed Processing Symposium*, p. 10, Denver, 2005.

[7] Cortez P., Rio M., Rocha M. and Sousa P., "Internet Traffic Forecasting using Neural Networks," *IEEE 2006 International Joint Conference on Neural Networks*, pp. 4942-4949, Vancouver, 2006.

[8] Cortez P., Rio M., Rocha M. and Sousa P., "Topology Aware Internet Traffic Forecasting using Neural Networks," *Proceedings of the 17th International Conference On Artificial Neural Networks*, pp. , Porto, 2007.

[9] Syed A. R., Aqil Burney S. M. and Sami B., "Forecasting Network Traffic Load Using Wavelet Filters and Seasonal Autoregressive Moving Average Model," *International Journal of Computer and Electrical Engineering*, vol. 2, no. 6, pp. 979-983, December 2010.

[10] Cortez P., Rio M., Rocha M. and Sousa P., "Multi-scale Internet Traffic Forecasting using Neural Networks and Time Series Methods," *Expert Systems*, 2010.

[11] Cisco, "Introduction to Cisco IOS NetFlow - A Technical Overview," Available at: *http://www.cisco.com/en/US/prod/collateral/iosswrel/ps6537/ps6555 /ps6601/prod_white_paper0900aecd80406232.html*.

[12] Brownlee N., "Traffic Flow Measurement: Experiences with NeTraMe," IETF RFC 2123 Available at: *http://www.rfc-editor.org/rfc/rfc2123.txt*, 1997.

[13] Fullmer M., "Flow-Tools," Available at: *http://www.splintered.net/sw/flow-tools*.

[14] Haag P., "NFSEN," Available at: *http://sourceforge.net/projects/nfsen/*.

[15] Haykin S., "Nonlinear Dynamical Systems: Feedforward Neural Network Perspectives," *John Wiley & Sons*, 2001.

[16] Zhang G. P., Patuwo B. E. and Hu M. Y., "A Simulation Study of Artificial Neural Networks for Nonlinear Time-Series Forecasting," *Computers & Operations Research 28*, pp. 381-396, Atlanta, 2001.