Neural Networks for Short-Term Load Forecasting: A Review and Evaluation

Henrique Steinherz Hippert, Carlos Eduardo Pedreira, and Reinaldo Castro Souza

***Abstract*—Load forecasting has become in recent years one of the major areas of research in electrical engineering, and most traditional forecasting models and artificial intelligence techniques have been tried out in this task. Artificial neural networks (NNs) have lately received much attention, and a great number of papers have reported successful experiments and practical tests with them. Nevertheless, some authors remain skeptical, and believe that the advantages of using NNs in forecasting have not been systematically proved yet.**

In order to investigate the reasons for such skepticism, this review examines a collection of papers (published between 1991 and 1999) that report the application of NNs to short-term load forecasting. Our aim is to help to clarify the issue, by critically evaluating the ways in which the NNs proposed in these papers were designed and tested.

***Index Terms*—Load forecasting, multilayer perceptrons, neural network applications, neural networks, overfitting.**

I. Introduction

THE FORECASTING of electricity demand has become one of the major research fields in electrical engineering. The supply industry requires forecasts with lead times that range from the short term (a few minutes, hours, or days ahead) to the long term (up to 20 years ahead). Short-term forecasts, in particular, have become increasingly important since the rise of the competitive energy markets. Many countries have recently privatised and deregulated their power systems, and electricity has been turned into a commodity to be sold and bought at market prices. Since the load forecasts play a crucial role in the composition of these prices, they have become vital for the supply industry.

Load forecasting is however a difficult task. First, because the load series is complex and exhibits several levels of seasonality: the load at a given hour is dependent not only on the load at the previous hour, but also on the load at the same hour on the previous day, and on the load at the same hour on the day with the same denomination in the previous week. Secondly, because there are many important exogenous variables that must be considered, specially weather-related variables. It is relatively easy to get forecasts with about 10% mean absolute percent error (MAPE); however, the costs of the error are so high that research that could help reducing it in a few percent points would be amply justified. A often quoted estimate in [10] suggests that an increase of 1% in the forecasting error would imply (in 1984) in a £10 million increase in operating costs per year (recent studies on the economic aspects of load forecasting are [9], [26] ,[41], [76]).

Most forecasting models and methods have already been tried out on load forecasting, with varying degrees of success. They may be classified as *time series (univariate) models*, in which the load is modeled as a function of its past observed values, and *causal models*, in which the load is modeled as a function of some exogenous factors, specially weather and social variables. Some models of the first class suggested in recent papers are multiplicative autoregressive models [60], dynamic linear [27] or nonlinear [80] models, threshold autoregressive models [43], and methods based on Kalman filtering [46], [69], [81]. Some of the second class are Box and Jenkins transfer functions [34], [47], ARMAX models [91], [92], optimization techniques [94], nonparametric regression [11], structural models [36], and curve-fitting procedures [85]. Despite this large number of alternatives, however, the most popular causal models are still the linear regression ones [30], [35], [67], [74], [82], and the models that decompose the load, usually into *basic* and *weather-dependent* components [10], [31], [45], [69]. These models are attractive because some physical interpretation may be attached to their components, allowing engineers and system operators to understand their behavior. However, they are basically linear devices, and the load series they try to explain are known to be distinctly nonlinear functions of the exogenous variables.

In recent times, much research has been carried out on the application of artificial intelligence techniques to the load forecasting problem. Expert systems have been tried out [39], [73], and compared to traditional methods [62]. Fuzzy inference [64] and fuzzy-neural models [7], [65] have also been tried out. However, the models that have received the largest share of attention are undoubtedly the artificial neural networks (NNs). The first reports on their application to the load forecasting problem were published in the late 1980's and early 1990's [21]. Since then, the number of publications has been growing steadily. Judging from the number of papers, NN-based forecasting systems have not turned into a "passing fad," as it was feared they might [12]. It seems that they have been well accepted in practice, and that they are used by many utilities [50].

Nevertheless, the reports on the performance of NNs in forecasting have not entirely convinced the researchers in this area, and the skepticism may be partly justified. Recent reviews and textbooks on forecasting argue that there is little systematic evidence as yet that NNs might outperform standard forecasting

Manuscript received August 24, 1999. H. S. Hippert was supported by a Ph.D. scholarship granted by the Brazilian Foundation for the Co-ordination of Higher Education and Graduate Training (PICDT-CAPES).

H. S. Hippert is with the Department of Statistics, Universidade Federal de Juiz de Fora, Brazil.

C. E. Pedriera and R. C. Souza are with the Department of Electrical Engineering, Pontificia Universidade Catolica do Rio De Janeiro, Brazil.

Publisher Item Identifier S 0885-8950(01)02306-9.

methods [13], [58]. Reviews of NN-based forecasting systems have concluded that much work still needs to be done before they are accepted as established forecasting techniques [33], [38], [96], and that they are promising, but that "a significant portion of the NN research in forecasting and prediction lacks validity" [1].

How could this skeptical attitude adopted by some experts be reconciled with the apparent success enjoyed by the NNs in load forecasting? In order to investigate this matter we reviewed 40 papers that reported the application of NNs to short-term load forecasting. These papers were selected from those published in the leading journals in electrical engineering between 1991 and 1999 (conferences proceedings were not considered). We found that, on the whole, two major shortcomings detract from the credibility of the results. First, most of the papers proposed NN architectures that seemed to be too large for the data samples they intended to model, i.e., there seemed to be too many parameters to be estimated from comparatively too few data points. These NNs apparently overfitted their data and one should, in principle, expect them to yield poor out-of-sample forecasts. Secondly, in most papers the models were not systematically tested, and the results of the tests were not always presented in an entirely satisfactory manner.

This paper is organized as follows. In Section II we give a short introduction to NN modeling. In Section III we briefly compare the approaches taken by each paper to the load forecasting problem, and we outline the main features of the *multilayer perceptrons* they proposed. In Section IV we summarize the choices and procedures reported in each paper for data pre-processing, NN design, implementation and validation. In Section V we focus on the problems of *overfitting* and *model validation* of the proposed models. In Section VI we briefly review papers that suggest NN architectures other than the *multilayer perceptron*, or some ways to combine them to linear methods. Section VII is the conclusion.

II. A Short Introduction to Neural Networks

In this section we provide a short introduction to neural networks (a complete treatment of the subject may be found in [8], [37]). Artificial neural networks are mathematical tools originally inspired by the way the human brain processes information. Their basic unit is the artificial *neuron*, schematically represented in Fig. 1. The neuron receives (numerical) information through a number of input nodes (four, in this example), processes it internally, and puts out a response. The processing is usually done in two stages: first, the input values are linearly combined, then the result is used as the argument of a nonlinear *activation function*. The combination uses the *weights* w_i attributed to each connection, and a constant *bias* term θ, represented in the figure by the weight of a connection with a fixed input equal to 1. The activation function must be a nondecreasing and differentiable function; the most common choices are either the identity function $(y = x)$, or bounded sigmoid (s-shaped) functions, as the logistic one $(y = 1/(1 + e^{-x}))$.

The neurons are organized in a way that defines the network *architecture*. The one we shall be most concerned with in this paper is the *multilayer perceptron* (MLP) type, in which the

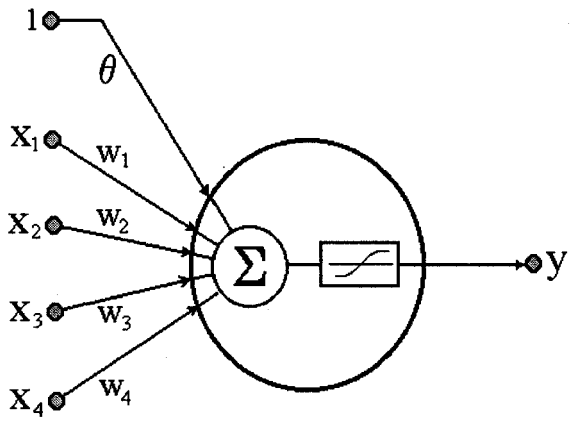

Fig. 1. An artificial neuron.

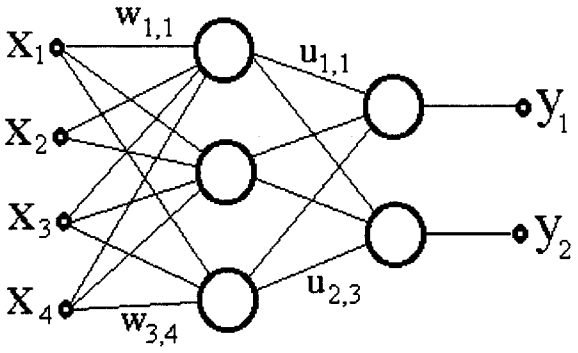

Fig. 2. A two-layer feed-forward neural network.

neurons are organized in layers. The neurons in each layer may share the same inputs, but are not connected to each other. If the architecture is *feed-forward*, the outputs of one layer are used as the inputs to the following layer. The layers between the input nodes and the output layer are called the *hidden layers*. Fig. 2 shows an example of a network with four input nodes, two layers (one of which is hidden), and two output neurons. The parameters of this network are the weight matrix $W_{3\times4}$ (containing the weights $w_{i,j}$ that connect the neuron i to the input j), the weight matrix $U_{2\times3}$, and the bias vector $\Theta_{5\times1}$ (the bias connections have not been represented in the figure). If logistic functions are used for the activation of the hidden layer, and linear functions used for the output layer, this network is equivalent to the model

$$y_k = \sum_{j=1}^{3} \left(u_{kj} \cdot \frac{1}{1 + \exp\left(-\sum_{i=1}^{4} w_{ji} x_i + \theta_j \right)} \right) + \theta_k$$

which shows how complex and flexible even a small network can be.

The estimation of the parameters is called the "training" of the network, and is done by the minimization of a loss function (usually a quadratic function of the output error). Many

optimization methods have been adapted for this task. The first training algorithm to be devised was the *back-propagation* one, which uses a steepest-descent technique based on the computation of the gradient of the loss function with respect to the network parameters (that is the reason why the activation functions must be differentiable). Many other training algorithms, though, are now available.

In load forecasting applications, this basic form of *multilayer feed-forward* architecture shown above is still the most popular. Nevertheless, there is a large number of other designs, which might be suitable for other applications.

Artificial NNs have been developed and extensively applied since the mid-1980's. There are many reports of successful applications [25], particularly in pattern recognition and classification [42], and in nonlinear control problems, where they may have some advantages over the traditional techniques. Since quantitative forecasting is based on extracting patterns from observed past events and extrapolating them into the future, one should expect NNs to be good candidates for this task. In fact, NNs are very well suited for it, for at least two reasons. First, it has been formally demonstrated that NNs are able to approximate numerically any continuous function to the desired accuracy (see [37], [96] for references). In this sense, NNs may be seen as multivariate, nonlinear and nonparametric methods, and they should be expected to model complex nonlinear relationships much better than the traditional linear models that still form the core of the forecaster's methodology. Secondly, NNs are data-driven methods, in the sense that it is not necessary for the researcher to postulate tentative models and then estimate their parameters. Given a sample of input and output vectors, the NNs are able to automatically map the relationship between them; they "learn" this relationship, and store this learning into their parameters. As these two characteristics suggest, NNs should prove to be particularly useful when one has a large amount of data, but little *a priori* knowledge about the laws that govern the system that generated the data.

In terms of theoretical research in forecasting, NNs have progressed from computing point estimates to computing both confidence intervals [89] and conditional probability densities [44], [90]. In terms of practical applications in forecasting, the success of NNs seems to depend on the kind of problem under consideration. An overview of the application of NNs to forecasting may be found in [88].

III. An Overview of the Proposed NN-Based Forecasting Systems

Most of the papers under review proposed multilayer perceptrons that might be classified into two groups, according to the number of output nodes. In the first group are the ones that have only one output node, used to forecast next hour's load, next day's peak load or next day's total load. In the second group are the ones that have several output nodes to forecast a sequence of hourly loads. Typically, they have 24 nodes, to forecast next day's 24 hourly loads (this series of hourly loads is called the "load profile").

We start with the first group. Reference [68] used three small-sized NNs to forecast hourly loads, total loads and peak loads (only one of these NNs is included in the Tables I–III). Reference [40] used a NN to forecast next day's peak load, which is needed as an input to the expert system in [39] that forecasts next day's profile. [14] suggested a nonfully connected network, in order to reduce the number of weights. Reference [70] proposed two NNs, one of which included a linear neuron among the sigmoidal ones in the hidden layer. Reference [23] experimented with feed-forward and recurrent NNs to forecast hourly loads, and was the only paper to report that linear models actually performed *better* than those NNs.

These NNs with only one output neuron were also used to forecast profiles, in either of two ways. The first way was by repeatedly forecasting one hourly load at a time, as in [28], [29]. The second way was by using a system with 24 NNs in parallel, one for each hour of the day: [61] compared the results of such a system to those of a set of 24 regression equations; [18] considered the load as an stationary process with level changes and outliers, filtered these out by a Kalman filter, and modeled the remaining load by a NN system; [86] considered the load as the output of a dynamic system, and modeled it by a set of 24 recurrent NNs.

Most of the profile forecasting, however, was done with the NNs of the second group (the ones with several input nodes). Reference [55] divided a day into 3 periods, and had a large

TABLE I
Input Classification

Paper (1)	Input variables to the forecasting system (2)	Variable the classification is based on (3)	No. of classes (4)
[14]	L, T	C	7
[40]	L, T	C	11
[53]	L, T	…	1
[68]	L, T	C	2
[70]	L, T	C	5
[2]	L,T, H	C	5
[6]	L,T	C	7
[15] [16]	L, T	C, C	8, 12
[17]	L, T, H, W	…	1
[28]	L, T	C	2
[49]	L, T, H	C	7
[50]	L, T, H, W	C	7
[51]	L, T, H	C, C	7, 3
[52]	L, T, f(T)	C	7
[54]	L	L	15
[55]	L	C	6
[57]	L, T	C, C	7, 12
[63]	L, T	C, T	105 (?)
[66]	L, T, f(T)	C	7
[72]	L, T	T	2
[83]	L, T, H, W	C	3
[56]	L, LP	…	1

(2) and (3)-C: the day's position in the calendar (weekday/weekend/holiday, month, or season), **L:** load, **T:** temperature, **H:** humidity, **W:** weather variables (other than T and H), **f(T):** nonlinear functions of T, **LP:** load parameters. **(4)** There are as many sets of classes as classification criteria. Cells marked with "...": values were not reported in the paper.

TABLE II
NN ARCHITECTURES AND IMPLEMENTATION

Paper	What to forecast	No. of neurons (in "main" or "typical" MLP)	Activation functions	Stopping rule
(1)	(2)	(3)	(4)	(5)
[14]	hourly loads	30/10/1	...	...
[40]	peak load	46/60/1	S/S	tol
[53]	hourly loads	14/5/3/1	...	tol
[68]	hourly loads	6/10/1	S/S	tol
[70]	total load	5/8/1	S/L	tol
[2]	profile	107/35/35/48	S/S/L	tol
[6]	profile	63/24/24	...	...
[15] [16]	profile	86/28/24	Sin/S	cv
[17]	profile	81/81/24	S/S	# iter
[28]	profile	15/10/1	S/S	...
[49]	profile	19:25/20:40/1	S/S	cv
[50]	profile	79/30:60/24	S/S	cv
[51]	profile	72/ ... /24	S/S	...
[52]	profile	64/48/24	...	tol
[54]	profile	51/31/24	...	tol
[55]	profile	48/70/24/4:10	S/S	tol
[57]	profile	38/10/24	S/S	# iter
[63]	profile	5:15/3/1	...	# iter
[66]	profile	77/24/24	S/S	...
[72]	profile	10/6/6/24	S/S/L	cv
[83]	profile	26/36/24	S/S	tol
[56]	next minute's load	38/30/25/16	S/S/L	tol

(3) input/hidden/output layers. Some papers reported ranges for the number of neurjons. They are indicated by colnons. **(4) L:** linear, **S:** signoidal, **Sin:** sinusoidal. **(5) cv:** cross–validation, **tol:** training was carried on until a specified tolerance (in-sample error) was reached, **# iter:** training was carried on for a fixed number of iterations. Cells marked with"...": values were not reported in the paper.

NN to forecast for each period. Reference [57] experimented with three NNs to model data from two utilities, and concluded that NNs are "system dependent," i.e., must be tailored for each specific utility. Reference [66] included nonlinear functions of temperature to the inputs, and also suggested a procedure to improve the forecasting on holidays. Reference [6] improved on it, specially for forecasting sequences of holidays. Reference [52] trained a NN with data from a small utility, and found it necessary to smooth their sample data by a manual pre-processing procedure. Reference [15] used a *fractional factorial experiment* to find out the "quasioptimal" network design parameters (number of neurons and layers, activation functions, stopping criteria, etc.) and came up with a rather unusual architecture: a recurrent NN, with sinusoidal activation functions in the hidden layers. In [16], the same NN was trained by a weighted least squares procedure in which the weights were the marginal energy costs. Since the load series are often nonstationary, [17] suggested that NNs could be used to model the first differences of the series, as nonlinear extensions to the ARIMA models. Other authors dealt with the problem of nonstationarity by detrending the series [63], [83], [84] or by filtering it with a Kalman filter [18].

Some papers suggested systems in which a number of NNs worked together to compute the forecasts. Reference [2] used a small NN that pre-processed some of the data and produced estimates of peak load, valley load, and total load, which were fed, together with some other data, into a very large NN that computed next day's profile. Reference [54] suggested a system of 12 NNs, one for each month of the year. In order to improve the forecast for "anomalous" days (holidays), the daily load profiles were classified by a Kohonen self-organized map. Reference [51] proposed a system in which the results of hourly, daily and weekly modules (38 NNs in total) were linearly combined. This system was replaced in [49] by a smaller one, composed of 24 NNs, one for each hour of the day. Later, some of these authors proposed a system with only two NNs [50]. One of them was trained to produce a first forecast of tomorrow's profile. The other one was trained to estimate tomorrow's load changes with respect to today's loads; these changes, added to today's loads, made up a second forecast of tomorrow's profile. The forecasts produced by both methods were linearly combined. It is argued that the second NN allowed the system to adapt more quickly to abrupt changes in temperature. In [63] the hourly loads were classified according to the season (seven classes), to the day of the week (three classes) and to the period of the day (five classes), and each of these classes was modeled by one of the independent NNs that made up a very large system. Reference [72] used a neural-gas network (a kind of NN used for vector

TABLE III
NUMBER OF PARAMETERS AND SAMPLE SIZES

Paper	No. of parameters (in"main" or "typical" MLP)	Total no. of MLPs in forecasting system	Total no. of parameters (weights + thresholds)	Size of training sample	Size of test sample (in days)
(1)	(2)	(3)	(4)	(5)	(6)
[14]	163	1	163	336	14
[40]	2,881	1	2,881	30	4
[53]	97	1	97	...	120
[68]	81	1	81	840	30
[70]	57	1	57	7	365
[2]	6,768	2	6,931	365	365
[6]	2,136	1	2,136	270	365
[15] [16]	3,132	1	3,132	1,460	365
[17]	8,610	1	8,610	136	40
[28]	171	2	342	144	56
[49]	421:1,081	24	10,104 : 25,944	1,095	182
[50]	3,144 : 6,264	2	6,288 : 12,528	1,095	365
[51]	...	38	...	1,095	365
[52]	4,296	1	4,296	1,460	365
[54]	2,380	12	28,560	31	21
[55]	5,384	6	32,004	...	182
[57]	654	12 (?)	7,848	365	84
[63]	22 : 52	105 (?)	2,310 : 5,460	1,095	365
[66]	2,472	1	2,472	1,825	365
[72]	276	2	552	365	365
[83]	1,860	3	5,580	30	10
[56]	2,361	1	2,361	1,440	...

(2) Some papers reported ranges for the number of neurons in their MLPs; these ranges are indicated by colons; **(3)** The numbers marked with (?) represent our best guesses, since the actual numbers were not clearly reported in the papers; **(4)** The ranges for the number of weights correspond to the ranges for the number of neurons. Cells marked with"...": values were not reported in the paper.

quantization [59]) to classify the data into two groups, *summer* and *winter*, which were modeled by separate feed-forward NNs. The forecasts were combined by a fuzzy module.

Fuzzy logic, another artificial intelligence technique, has also been tried in combination with NNs in some of the most recent papers. Reference [83] included a "front-end fuzzy processor" that received quantitative and qualitative data, and put out four fuzzy numbers that measured the expected load change in each of the four periods into which the target day had been divided; these numbers, together with some temperature data, were fed to the NN that computed the forecasted profile. The fuzzy pre-processing reduced the number of NN inputs and allowed the system to work on qualitative data. Reference [53] placed the fuzzy engine *after* the NN. The NN provided a "provisional forecast" based on past loads, which was afterwards modified by the fuzzy engine with basis on the temperature and type of day (regular or holiday). Particular attention was given to the modeling of holidays. Reference [22] classified the data into 48 fuzzy subsets according to temperature and humidity; each subset was modeled by a separate NN. Reference [72], already mentioned above, used a fuzzy module to combine results from two separate NNs. Reference [48] forecasted the demand of a residential area by decomposing it into a *normal load* and a *weather sensitive load*. The normal load was modeled by three NNs (for week days, Saturdays and Sundays); the weather sensitive load was modeled by a fuzzy engine, with basis on weather data.

The only paper that dealt with very short-term forecast was [56]. The authors compared auto-regressive, fuzzy and NN models in the minute-by-minute forecasting of the load in the next half-hour.

We discuss these papers more fully in the Sections IV and Section V, focusing on the way the MLPs models were designed and tested. We are not concerned with the fuzzy modules; fuzzy neural networks [7], [65] are also outside the scope of this paper. A few papers suggested NN architectures other than the MLP, sometimes combined with traditional linear methods; we review these briefly in Section VI.

IV. ISSUES IN DESIGNING A NN-BASED FORECASTING SYSTEM

Neural networks are such flexible models that the task of designing a NN-based forecasting system for a particular application is far from easy. There is a large number of choices that have to be made, but very few guidelines to help the designer through them. Some recent theoretical contributions can be found in [3],

[78], [79], [87], but they have not been much tested in practice as yet.

The design tasks can be roughly divided into four headings:

A. Data pre-processing;
B. NN designing;
C. NN implementation;
D. Validation.

This subdivision is somewhat artificial, as these stages in practice tend to overlap, but it is useful in the organization of what is to follow. In most of the papers we reviewed, the authors made their choices guided by empirical tests and simulations. In the next four sub-sections we summarize these choices; in Section V, we discuss their consequences and implications.

A. Data Pre-Processing

Before data are ready to be used as input to a NN, they may be subjected to some form of *pre-processing*, which usually intends to make the forecasting problem more manageable. Pre-processing may be needed to reduce the dimension of the input vector, so as to avoid the "curse of dimensionality" (the exponential growth in the complexity of the problem that results from an increase in the number of dimensions). Pre-processing may also be needed to "clean" the data, by removing outliers, missing values or any irregularities, since NNs are sensitive to such defective data. References [52], [72] devised heuristics to regularize their data; [18] filtered out the irregularities with a Kalman filter.

Pre-processing also frequently means partitioning the input space so that a "local" model may be designed for each subspace. These models will be simpler and will require less data than a "global" model. In load forecasting, this is usually done by *classifying* the input data (past load profiles or weather data), and then using separate NNs to model data from each class.

The most important factor to determine the shape of the load profile is the calendar date (see Table I); the week day profiles are typically very different from the weekend profiles. Thus, the basic classification is into two groups: week days and weekend days. References [17], [68] dealt only with the week day profiles, discarding the weekends and holidays. Reference [57] ignored such distinction, but got poor results as a consequence on weekends and holidays. Sometimes the profiles of the days just before Saturdays or after Sundays may be disturbed by the weekend, so that special classes may be needed for Mondays, Fridays, or even Thursdays [2], [63], [70]. Weekend days may be further classified according to the social customs and working patterns in the country. If this process is continued, the number of classes may rise to eleven [40], though the usual number is seven (one class for each day of the week). The typical within-week profiles may change from season to season, and they should then be further classified according to month or season [63], [72].

Holidays pose a special problem. Some authors group them with the weekend days; others reserve them special classes, or devise heuristics to deal with them [6], [50], [51], [66].

The second most important factor to affect the load profile is the weather. Because of this, the days may be classified according to the weather conditions, by statistical measures of similarity [28], [29], [70], by fuzzy engines [22], and by neural-gas networks [72].

In [54] the profiles themselves (not the temperatures or the calendar dates) were classified by a 8×8 Kohonen self-organized map. The classes were then interpreted by the system operator, so that the class to which the target day would belong could be predicted.

These classification procedures give a class label to each of the profiles in the training sample. These labels will make up a new variable that must be included in the model. This may be done in either of two ways. The first one is by coding these class labels and using the codes as input variables. That may mean adding to the NN as many input nodes as the number of classes, and having each class represented by a dummy variable [6], [14], [15], [49], [50], [52], [57], [66]; or, alternatively, numbering the classes (normally on a binary basis), and feeding these numbers to the NN through a few input nodes, so that a large number of classes may be represented by a comparatively small number of input nodes [2], [22], [54].

The second way to use this information is by building separate NN models for each class (or a common NN, with a separate set of weights for each class) [22], [28], [51], [54], [55], [57], [63], [72]. However, if the data are subdivided into too many classes, there will not be enough profiles left in each class to permit network training. Extreme instances of this subdivision are found in [22], [63].

B. Designing the Neural Network

Selecting an appropriate *architecture* is in general the first step to take when designing a NN-based forecasting system. Many types of architecture have already been used in forecasting applications. In all the papers discussed in this chapter, however, the authors used the NN work-horse, the multilayer perceptron, usually a *feed-forward* one (the exceptions were the recurrent networks in [15], [16], [86]). Most of them were *fully-connected* (the exception was in [14]).

Having chosen this type of architecture, one must then decide on the number of hidden layers, of input nodes, of neurons per layer, and on the type of activation functions (see Table II). The number of hidden layers is not difficult to determine. It has been shown that one hidden layer is enough to approximate any continuous functions, although two layers may be useful in some specific circumstances (see [37] for references). The papers we reviewed used either one or two hidden layers. Defining the activation function for the hidden neurons is also not difficult. This functions must be differentiable and nondecreasing; most papers used either the *logistic* or the *hyperbolic tangent* functions, and it is not clear whether the choice has any effect on the forecasting accuracy [96].

In the following sub-sections we report how the number of output neurons, input nodes, and hidden neurons were defined in the papers under review.

1) Selecting the Number of Output Neurons: Six of the papers listed in Table II used one-output NNs to produce one-step-ahead forecasts: forecasts for next day's peak load or total load, or forecasts for hourly loads (that is, given the load series up to hour h, forecasts for the load at hour $h+1$). Most of the reviewed

papers however were concerned with forecasting profiles. They did this in one of three ways: a) iterative forecasting; b) multi-model forecasting; c) single-model multivariate forecasting.

a) Iterative Forecasting: This is done by forecasting one hourly load at a time and then aggregating this load to the series, so that the forecasts for the later hours will be based on the forecasts for the earlier ones. If the model is an ARIMA, it may be shown that the forecasts will eventually converge to the series average. However, it is not clear what happens if the model is a MLP. Reference [38] studied the accuracy of multi-steps forecasts obtained iteratively by a MLP on the M-competition time series, and found that this MLP outperformed the statistical models; however, [19], [20] reported that the MLP outputs may behave chaotically. Among the papers we reviewed, [28], [29] used only this iterative method, whereas [55], [57] experimented with both methods a) and c), and found that their results were roughly equivalent.

b) Multi-Model Forecasting: This is a common method for load forecasting with regression models: using 24 different models, one for each hour of the day. Among the papers we reviewed, [18], [61], [86] used systems of 24 NNs in parallel. The advantage of this method is that the individual networks are relatively small, and so they are not likely to be overfitted.

c) Single-Model Multivariate Forecasting: This is done by using a multivariate method to forecast all the loads at once, so that each profile is represented by a 24-dimensional vector. This method was used by most of the researchers, who designed MLPs with 24 neurons in the output layer. (The exceptions were [2], that forecast loads at each half-hour and so needed 48 output neurons; and [55], that divided the profile into three parts, forecasted by separated MLPs.) This method, however, has two serious drawbacks. The first one is that the MLPs must be very large in order to accommodate 24 output neurons ([56], doing minute-by-minute forecasting for the next half-hour using this method, tried to reduce the number of output nodes by compressing the 30-dimension output vector into 16 nodes, using a transformation technique). If the loads of one or two previous days are used as inputs, 24 or 48 input nodes will be required, and the number of MLP parameters will very likely run into the thousands.

The second drawback is that treating each day as a vector means that one year of data will yield only 365 data points, which seems to be too few for the large MLPs required. Trying to increase the sample size by aggregating data from years way back in the past may not be feasible, because in most places the load series show a very clear upward trend.

2) Selecting the Number of Input Nodes: After selecting the number of layers and the number of output neurons required, one must choose the inputs. There are very few theoretical considerations to help in this decision; usually, one must have some *a priori* knowledge about the behavior of the system under study, and of the factors that condition the output of that system.

The first variable to be used is almost certainly the load itself, as the load series is strongly autocorrelated. If the MLP forecasts the profiles, treating them as 24-dimension vectors, the researcher does not have much choice: either he uses data from one past day [2], [17], [51] or from two [6], [15], [16]. No experiments with three or more past days were reported, because they would imply having 72 or more input nodes for the loads only. If however the MLP is forecasting hourly loads, the problem becomes more complex, since the researcher then needs to select what lagged load values should be used as inputs. Some authors tried to adapt the Box and Jenkins methodology for fitting ARIMA models, and selected the lags by the analysis of the autocorrelation functions (ACF) and the partial autocorrelation functions (PACF) [14], [57]. In doing so, however, they run the risk of discarding lagged variables that showed no significant linear correlation to the load, but which were strongly nonlinearly correlated to it. Reference [28] used phase-space embedding, a technique that represents a system by one variable and its lagged versions, to help determining which lagged values of the load series should be used as inputs.

As the short-term load forecasting problem has been intensively studied for decades, there are some empirical guidelines that may help in selecting among the candidate exogenous variables. The main variable to be included is the air temperature, since it has been known since de 1930's that the demand rises on cold days because of the use of electric space- and water-heating devices, and on hot days, because of air conditioning. The function that relates the temperature to the load is clearly nonlinear; that is, of course, one of the main motivations to use NN in this context, since NNs can easily deal with nonlinear relationships. However, since this function seems to be U-shaped in many countries (see for example the graphs relating peak load to maximum temperature in [34], [35], [95]), some authors used piecewise linear-quadratic functions of the temperature as input variables [52], [66]. The idea was borrowed from earlier papers that modeled the components of linear regression models by such piecewise functions [67] or by polynomial functions [34]. Some authors experimented with other variables, such as relative humidity or wind speed, since they have a strong effect on the human sensation of thermal discomfort and may help explaining the use of heating and cooling devices [2], [17], [22], [50], [51]. Others concluded that the only significant weather variable was the temperature [6], [53], [66]. In most cases, however, the authors did not have much choice, as data on weather variables other than temperature were simply unavailable.

As the MLPs were used in these papers as nonlinear regression models, load forecasts required weather forecasts. Most authors run their simulations using observed weather values instead of forecasted ones, which is standard practice in load forecasting; however, one should keep in mind that the forecasting errors in practice will be larger than those obtained in simulations, because of the added weather forecast uncertainty (for some studies on the effect of this uncertainty see [27], [75]).

3) Selecting the Number of Hidden Neurons: Determining the number of neurons in the hidden layer may be more difficult than determining the size of the input or the output layers. There is again little theoretical basis for the decision, and very few successful heuristics have been reported [96]. The issue may be roughly compared to that of choosing the number of harmonics to be included in a Fourier model to approximate a function; if they are too few, the model will not be flexible enough to model the data well; if they are too many, the model will overfit the data. In most papers, authors chose this number by trial and

error, selecting a few alternative numbers and then running simulations to find out the one that gave the best fitting (or predictive) performance. Some of the papers reported that variations in the number of hidden neurons did not significantly affect forecasting accuracy [6], [49].

C. Neural Network Implementation

After an MLP has been designed, it must be trained (that is, its parameters must be estimated). One must select a "training algorithm" for this task. The most common in use is the *backpropagation* algorithm, based on a steepest-descent method that performs stochastic gradient descent on the error surface, though many alternatives to it have been proposed in recent years. Since these algorithms are iterative, some criteria must be defined to stop the iterations. In most of the papers reviewed, training was stopped after a fixed number of iterations, or after the error decreased below some specified tolerance (see Table II). These criteria are not adequate, as they insure that the model fits closely to the training data, but do not guarantee good out-of-sample performance; they may lead to *overfitting* of the model (this point is further discussed in Section V).

Lastly, the training samples must be appropriately selected. Since NNs are "data-driven" methods, they typically require large samples in training. References [28], [29], [70] trained their MLPs on small subsets that included data from only a few past days selected through statistical measures of similarity. That resulted in samples that were very homogeneous, but also very small.

D. Neural Network Validation

The final stage is the validation of the proposed forecasting system. It is well known that goodness-of-fit statistics are not enough to predict the actual performance of a method, so most researchers test their models by examining their errors in samples other that the one used for parameter estimation (*out-of-sample* errors, as opposed to *insample* errors). Some of the papers reviewed did not clearly specify whether the results they reported had been obtained insample or out-of-sample.

V. Discussion

In this section we discuss the implications and consequences of the choices made in the papers under review on the issues of *design*, *implementation* and *validation* of NN models. We shall use some of the guidelines proposed in [1] for the evaluation of these choices.

A. Evaluating the Neural Network Design—The Problems of Overfitting and Overparameterization

The problem of *overfitting* is frequently mentioned in the NN literature. It seems, however, that different authors give this word different meanings. For instance, [1] remark that the usual MLPs trained by backpropagation "are known to be seriously prone to overfitting" and that this could be prevented by avoiding excessive training. On the other hand, [33], [96] remark that MLPs are prone to overfit sample data because of the large number of parameters that must be estimated. These authors mean two different things, and we should better start by defining our terms.

"Overfitting" usually means estimating a model that fits the data so well that it ends by including some of the error randomness in its structure, and then produces poor forecasts. In MLPs, as the remarks above imply, this may come about for two reasons: because the model was overtrained, or because it was too complex.

One way to avoid overtraining is by using *cross-validation*. The sample set is split into a *training* set and a *validation* set. The NN parameters are estimated on the training set, and the performance of the model is tested, every few iterations, on the validation set. When this performance starts to deteriorate (which means the NN is overfitting the training data), the iterations are stopped, and the last set of parameters to be computed is used to produce the forecasts.

Another way is by using *regularization* techniques. This involves modifying the cost function to be minimized, by adding to it a term that penalizes for the complexity of the model. This term might, for example, penalize for the excessive curvature in the model by considering the second derivatives of the output with respect to the inputs. Relatively simple and smooth models usually forecast better than complex ones. Overfitted NNs may assume very complex forms, with pronounced curvature, since they attempt to "track down" every single data point in the training sets; their second derivatives are therefore very large and the regularization term grows with respect to the error term. Keeping the total error low, therefore, means keeping the model simple. None of the papers we reviewed, however, used regularization techniques.

Overfitting however may also be a consequence of *overparameterization*, that is, of the excessive complexity of the model. The problem is very common in MLP-based models; since they are often (and improperly) used as "black-box" devices, the users are sometimes tempted to add to them a large number of variables and neurons, without taking into account the number of parameters to be estimated. Many methods have been suggested to "prune" the NN, i.e., to reduce the number of its weights, either by shedding some of the hidden neurons, or by eliminating some of the connections [77]. However, the adequate rate between the number of sample points required for training and the number of weights in the network has not yet been clearly defined; it is difficult to establish, theoretically, how many parameters are too many, for a given sample size.

Returning now to the load forecasting problem, Table III compares the sizes of the parameter sets to the sizes of the training sets in the papers under review. (The number of parameters was not explicitly reported in any of the papers. We computed it by piecing together the information about the number of MLPs and the number of neurons per MLP in the forecasting systems.) It may be seen, by comparing columns (4) and (5), that most of the proposed MLPs, specially the ones that forecasted profiles, had more parameters than training points. This is a consequence of the way they were designed; using 24 output neurons implies that the MLPs will be large, and the samples, small. One should, in principle, expect these MLPs to fit their training data very well (in fact, to *overfit* them), but one should not expect them to produce good forecasts.

B. Evaluating the Neural Network Implementation

The next stage in modeling is the implementation of the NN, that is, the estimation of its parameters. The guidelines proposed in [1] for evaluating "Effectiveness of implementation" were based on the question: was the NN properly trained and tested, so that its performance was the best it could possibly achieve? According to these authors, the NN can be said to have been properly implemented if

i) it was well fitted to the data (the errors in the training sample must be reported);
ii) its performances in the training sample and in the test samples were comparable;
iii) its performances across different test samples were coherent.

Few of the papers under review reported any insample results [52], [61], so effectiveness of implementation may not be examined further.

C. Evaluating the Neural Network Validation

Guidelines for evaluating the "Effectiveness of validation" were also proposed in [1]. The evaluation is based on the question: was the performance of the proposed method fairly compared to that of some well-accepted method? A method is considered to have been properly validated if: i) its performance was compared to that of well-accepted methods; ii) the comparison was based on the performance on test samples; iii) the size of the test samples were adequate, so that some inference might be drawn.

Item i) may be interpreted in two ways. First, the proposed method may be compared to some "naïve" method, which provides a (admittedly low) benchmark. The proposed method must be noticeably better than the naïve method, otherwise there would be no point in adopting it. The naïve forecast is also useful, besides, to show the reader how difficult a forecasting problem is; as the size and load profiles of the utilities concerned vary greatly across the reviewed papers, it would have been interesting to see how difficult the problems were in each case (no paper reported them, however). Second, the performance of the proposed method may be compared to that of a good standard method. The proposed method may be not much more accurate than the standard one, but it must have some other advantage (it may be easier to use, for instance).

It is difficult to find a good standard for comparison in a problem like load forecasting. ARMAX or regression models would probably be a good choice, but it must be admitted that fitting them would require as much hard work as fitting the MLPs they must test. That is perhaps the reason why most papers did not make any kind of comparison (only [14], [15], [53], [55], [56], [61], [63], [66], [68], [83] reported comparisons to standard linear models). However, if there are no comparisons, the reports on the performance of a proposed method are difficult to interpret. We do not believe that comparisons to other NNs or to fuzzy engines are valid, as these models are not yet considered "standard" or "well accepted" methods.

We should like to add yet another item to the guidelines suggested above: that iv) the results of these comparisons should be thoroughly examined by means of the standard techniques used in forecasting, and reported as fully as possible. In most papers, the forecasting errors were not examined in detail.

Most reported only the Mean Absolute Percent Errors (MAPE); few also reported the standard deviation of the errors [2], [17], [55], [70], [86]. Although MAPE has become somewhat of a standard in the electricity supply industry, it is clearly not enough in this context. The choice of error measures to help comparing forecasting methods has been much discussed, as a consequence of the many competitions that were started in the 1980's [4], [5], [32]. Most authors agree that the loss function associated with the forecasting errors, if known, should be used in the evaluation of a method. MAPE would be an adequate error measure if the loss function were linear (and linear in percentage, not in absolute error); however, recent studies [41], [76] and the experience of system operators indicates that the loss function in the load forecasting problem is clearly nonlinear, and that large errors may have disastrous consequences for a utility. Because of this, measures based on squared error are sometimes suggested, as they penalize large errors (Root Mean Square Error was suggested in [4], Mean Square Percentage Error in [5]). Also, it is generally recognized that error measures should be easy to understand and closely related to the needs of the decision-makers. Some papers reported that the utilities would rather evaluate forecasting systems by the absolute errors produced [63], [66], and this suggests that Mean Absolute Errors could be useful (they were reported in [6], [63], [66]).

In any case, error measures are only intended as summaries for the error distribution. This distribution is usually expected to be normal white noise in a forecasting problem, but it will probably not be so in a complex problem like load forecasting (specially if this distribution is seen as multivariate, conditioned on lead time). No single error measure could possibly be enough to summarize it. The shape of the distribution should be suggested. A few papers included graphs of the cumulative distribution of the errors [22], [63], [70], [86], others suggested this distribution by reporting the percentage of errors above some critical values [52], [63], [66], percentiles [70], [86], or the maximum errors [2], [28], [52]. A histogram of the errors was included in [63]. The possibility of serial correlation should be investigated by graphical means (scatterplots and correlograms) and portmanteau tests [58], [61]. Most of the papers reviewed, however, largely bypassed these standard forecasting practices.

VI. Some Proposed Alternatives

A few papers have proposed architectures other than the MLP, or new ways to combine MLPs to standard statistical methods. In [95], the variables were grouped into a few more or less homogeneous groups, and sorted according to an index that measured how much each variable was (nonlinearly) correlated to the load. The ones which were most correlated were fed to a network that implemented a projection pursuit regression model. Twenty-four such NNs were used to forecast a profile; each one of them had a single hidden layer, with 5 neurons that were grouped into "sub-nets."

Reference [71] dealt with the load series in the frequency domain, using signal-processing techniques. The series was

decomposed into three components of different frequencies, which were forecasted by separate Adaline neurons (a kind of linear neurons, see [37]). The series of forecasting errors was also decomposed into three components: an autoregressive and a weather-dependent one, both forecast by Adalines, and random noise.

Reference [24] used a *functional-link* network that had only one neuron. The inputs were a set of sinusoids, past forecasting errors, and temperatures. The neuron had a linear activation function, and so this network may be interpreted as a linear model that decomposed the load into a weather-independent component (modeled by a Fourier series), a weather-dependent component (modeled by polynomial functions and by "functional links"), and random noise.

Self-organizing NNs were also used. Reference [84] trained Kohonen networks to find "typical" profiles for each day of the week, and then used a fuzzy engine to compute corrections to those profiles, based on the weather variables and on the day types. Reference [93] proposed an unusual self-organizing NN model, in which the neurons were split into two clusters; one of them received past load data, the other received temperature data. Reference [20] used a Kohonen NN to classify the normalized sample profiles and to identify the "typical" profiles of each class. When forecasting for a Tuesday in October (for example), one checked in which classes Tuesdays fell in the past Octobers, and averaged the typical profiles of those classes. This average profile was then "de-normalized" in order to produce the final forecast (i.e., it was multiplied by the standard deviation and added to the mean; both parameters must have been previously forecasted by some linear method).

VII. Conclusion

The application of artificial NNs to forecasting problems has been much studied in recent times, and a great number of papers that report successful experiments and practical tests have been published. However, not all the authors and researchers in forecasting have been convinced by those reports, and some believe that the advantages of using NNs have not been systematically proved yet. The aim of this review is to contribute to clarifying the reasons for this skepticism, by examining a collection of recent papers on NN-based load forecasting, and by critically evaluating the ways the systems they proposed had been designed and tested.

This examination led us to highlight two facts that may be among the reasons why some authors are still skeptical:

a) Most of the proposed models, specially the ones designed to forecast profiles, seemed to have been overparameterized. Many were based on *single-model multivariate* forecasting, i.e., they regarded the profiles as vectors with 24 components that should be forecasted simultaneously by a single NN with 24 output neurons. This approach led to the use of very large NNs, which might have hundreds of parameters to be estimated from very small data sets. One would expect these NNs to have overfitted their training data, and one would not, in principle, expect them to produce good out-of-sample forecasts.

b) The results of the tests performed on these NNs, though apparently good, were not always very convincing. All those systems were tested on real data; nevertheless, in most cases the tests were not systematically carried out: the systems were not properly compared to standard benchmarks, and the analysis of the errors did not make use of the available graphical and statistical tools (e.g., scatterplots and correlograms).

In short, most of those papers presented seemingly misspecified models that had been incompleted tested. Taken by themselves, they are not very convincing; however, the sheer number of similar papers published in reputable journals, and the fact that some of the models they propose have been reportedly very successful in everyday use, seem to suggest that those large NN-based forecasters might work, after all, and that we still do not properly understand how overparameterization and overfitting affect them.

We believe, in conclusion, that more research on the behavior of these large neural networks is needed before definite conclusions are drawn; also, that more rigorous standards should be adopted in the reporting of the experiments and in the analysis of the results, so that the scientific community could have more solid results on which to base the discussion about the role played by NNs in load forecasting.

Acknowledgment

The authors would like to thank Prof. R. R. Bastos (Univ. Federal de Juiz de Fora, Brazil) for his careful reading of the first version of the manuscript, and also Prof. D. Bunn (London Business School, UK) for many useful suggestions on the second version.

References

[1] M. Adya and F. Collopy, "How effective are neural networks at forecasting and prediction? A review and evaluation," *J. Forecast.*, vol. 17, pp. 481–495, 1998.
[2] A. S. AlFuhaid, M. A. El-Sayed, and M. S. Mahmoud, "Cascaded artificial neural networks for short-term load forecasting," *IEEE Trans. Power Systems*, vol. 12, no. 4, pp. 1524–1529, 1997.
[3] U. Anders and O. Korn, "Model selection in neural networks," *Neural Networks*, vol. 12, pp. 309–323, 2000.
[4] J. S. Armstrong and F. Collopy, "Error measures for generalizing about forecasting methods: Empirical comparisons," *Int. J. Forecast.*, vol. 8, pp. 69–80, 1992.
[5] J. S. Armstrong and R. Fildes, "Correspondence on the selection of error measures for comparisons among forecasting methods," *J. Forecast.*, vol. 14, pp. 67–71, 1995.
[6] A. G. Bakirtzis, V. Petridis, S. J. Kiartzis, M. C. Alexiadis, and A. H. Maissis, "A neural network short term load forecasting model for the Greek power system," *IEEE Trans. Power Systems*, vol. 11, no. 2, pp. 858–863, 1996.
[7] A. G. Bakirtzis, J. B. Theocharis, S. J. Kiartzis, and K. J. Satsios, "Short-term load forecasting using fuzzy neural networks," *IEEE Trans. Power Systems*, vol. 10, no. 3, pp. 1518–1524, 1995.
[8] C. M. Bishop, *Neural Networks for Pattern Recognition*. Oxford: Claredon Press, 1997.
[9] D. W. Bunn, "Forecasting loads and prices in competitive power markets," *Proc. IEEE*, vol. 88, no. 2, pp. 163–169, 2000.
[10] D. W. Bunn and E. D. Farmer, Eds., *Comparative Models for Electrical Load Forecasting*: John Wiley & Sons, 1985.
[11] W. Charytoniuk, M. S. Chen, and P. Van Olinda, "Nonparametric regression based short-term load forecasting," *IEEE Trans. Power Systems*, vol. 13, no. 3, pp. 725–730, 1998.
[12] C. Chatfield, "Neural networks: Forecasting breakthrough or passing fad?," *Int. J. Forecast*, vol. 9, pp. 1–3, 1993.

[13] ——, "Forecasting in the 1990's," in *Proc. V Annual Conf. of Portuguese Soc. of Statistics*, Curia, 1997, pp. 57–63.
[14] S. T. Chen, D. C. Yu, and A. R. Moghaddamjo, "Weather sensitive short-term load forecasting using nonfully connected artificial neural network," *IEEE Trans. Power Systems*, vol. 7, no. 3, pp. 1098–1105, 1992.
[15] M. H. Choueiki, C. A. Mount-Campbell, and S. C. Ahalt, "Building a 'Quasi Optimal' neural network to solve the short-term load forecasting problem," *IEEE Trans. Power Systems*, vol. 12, no. 4, pp. 1432–1439, 1997.
[16] M. H. Choueiki, C. A. Mount-Campbell, and S. C. Ahalt, "Implementing a weighted least squares procedure in training a neural network to solve the short-term load forecasting problem," *IEEE Trans. Power Systems*, vol. 12, no. 4, pp. 1689–1694, 1997.
[17] T. W. S. Chow and C. T. Leung, "Neural network based short-term load forecasting using weather compensation," *IEEE Trans. Power Systems*, vol. 11, no. 4, pp. 1736–1742, 1996.
[18] J. T. Connor, "A robust neural network filter for electricity demand prediction," *J. Forecast.*, vol. 15, no. 6, pp. 437–458, 1996.
[19] M. Cottrell, B. Girard, Y. Girard, M. Mangeas, and C. Muller, "Neural modeling for time series: A statistical stepwise method for weight elimination," *IEEE T. Neural Nets.*, vol. 6, no. 6, pp. 1355–1364, 1995.
[20] M. Cottrell, B. Girard, and P. Rousset, "Forecasting of curves using a Kohonen classification," *J. Forecast.*, vol. 17, pp. 429–439, 1998.
[21] T. Czernichow, A. Piras, K. Imhof, P. Caire, Y. Jaccard, B. Dorizzi, and A. Germond, "Short term electrical load forecasting with artificial neural networks," *Engineering Intelligent Syst.*, vol. 2, pp. 85–99, 1996.
[22] M. Daneshdoost, M. Lotfalian, G. Bumroonggit, and J. P. Ngoy, "Neural network with fuzzy set-based classification for short-term load forecasting," *IEEE Trans. Power Systems*, vol. 13, no. 4, pp. 1386–1391, 1998.
[23] G. A. Darbellay and M. Slama, "Forecasting the short-term demand for electricity—Do neural networks stand a better chance?," *Int. J. Forecast.*, vol. 16, pp. 71–83, 2000.
[24] P. K. Dash, H. P. Satpathy, A. C. Liew, and S. Rahman, "A real-time short-term load forecasting system using functional link network," *IEEE Trans. Power Systems*, vol. 12, no. 2, pp. 675–680, 1997.
[25] T. Dillon, P. Arabshahi, and R. J. Marks II, "Everyday applications of neural networks," *IEEE T. Neural Nets.*, vol. 8, no. 4, pp. 825–826, 1997.
[26] A. P. Douglas, A. M. Breipohl, F. N. Lee, and R. Adapa, "Risk due to load forecast uncertainty in short term power system planning," *IEEE Trans. Power Systems*, vol. 13, no. 4, pp. 1493–1499, 1998.
[27] ——, "The impact of temperature forecast uncertainty on bayesian load forecasting," *IEEE Trans. Power Systems*, vol. 13, no. 4, pp. 1507–1513, 1998.
[28] I. Drezga and S. Rahman, "Input variable selection for ANN-based short-term load forecasting," *IEEE Trans. Power Systems*, vol. 13, no. 4, pp. 1238–1244, 1998.
[29] ——, "Short-term load forecasting with local ANN predictors," *IEEE Trans. Power Systems*, vol. 14, no. 3, pp. 844–850, 1999.
[30] R. F. Engle, C. Mustafa, and J. Rice, "Modeling peak electricity demand," *J. Forecast.*, vol. 11, pp. 241–251, 1992.
[31] J. Y. Fan and J. D. McDonald, "A real-time implementation of short-term load forecasting for distribution power syst.," *IEEE Trans. Power Systems*, vol. 9, no. 2, pp. 988–994, 1994.
[32] R. Fildes, "The evaluation of extrapolative forecasting methods," *Int. J. Forecast.*, vol. 8, pp. 81–98, 1992.
[33] W. L. Gorr, "Research prospective on neural network forecasting," *Int. J. Forecast.*, vol. 10, pp. 1–4, 1994.
[34] M. T. Hagan and S. M. Behr, "The time series approach to short term load forecasting," *IEEE Trans. Power Systems*, vol. PWRS-2, no. 3, pp. 785–791, 1987.
[35] T. Haida and S. Muto, "Regression based peak load forecasting using a transformation technique," *IEEE Trans. Power Systems*, vol. 9, no. 4, pp. 1788–1794, 1994.
[36] A. Harvey and S. J. Koopman, "Forecasting hourly electricity demand using time-varying splines," *J. American Stat. Assoc.*, vol. 88, no. 424, pp. 1228–1236, 1993.
[37] S. Haykin, *Neural Networks—A Comprehensive Foundation*, 2nd. ed. Upper Saddle River, NJ: Prentice Hall, 1999.
[38] T. Hill, L. Marquez, M. O'Connor, and W. Remus, "Artificial neural network models for forecasting and decision making," *Int. J. Forecast.*, vol. 10, pp. 5–15, 1994.
[39] K. L. Ho, Y. Y. Hsu, C. F. Chen, T. E. Lee, C. C. Liang, T. S. Lai, and K. K. Chen, "Short term load forecasting of Taiwan power system using a knowledge-based expert system," *IEEE Trans. Power Systems*, vol. 5, no. 4, pp. 1214–1221, 1990.
[40] K. L. Ho, Y. Y. Hsu, and C. C. Yang, "Short term load forecasting using a multilayer neural network with an adaptive learning algorithm," *IEEE Trans. Power Systems*, vol. 7, no. 1, pp. 141–149, 1992.
[41] B. F. Hobbs, S. Jitprapaikulsarn, S. Konda, V. Chankong, K. A. Loparo, and D. J. Maratukulam, "Analysis of the value for unit commitment of improved load forecasting," *IEEE Trans. Power Systems*, vol. 14, no. 4, pp. 1342–1348, 1999.
[42] L. Holmstrom, P. Koistinen, J. Laaksonen, and E. Oja, "Neural and statistical classifiers—Taxonomy and two case studies," *IEEE T. Neural Nets.*, vol. 8, no. 1, pp. 5–17, 1997.
[43] S. R. Huang, "Short-term load forecasting using threshold autoregressive models," *IEE Proc.—Gener. Transm. Distrib.*, vol. 144, no. 5, pp. 477–481, 1997.
[44] D. Husmeier and J. G. Taylor, Eds., *Neural Networks for Conditional Probability Estimation: Forecasting Beyond Point Predictions (Perspectives in Neural Computing)*: Springer-Verlag, 1999.
[45] O. Hyde and P. F. Hodnett, "An adaptable automated procedure for short-term electricity load forecasting," *IEEE Trans. Power Systems*, vol. 12, no. 1, pp. 84–93, 1997.
[46] D. G. Infield and D. C. Hill, "Optimal smoothing for trend removal in short term electricity demand forecasting," *IEEE Trans. Power Systems*, vol. 13, no. 3, pp. 1115–1120, 1998.
[47] G. M. Jenkins, "Practical experiences with modeling and forecast," *Time Series.*, 1979.
[48] H. R. Kassaei, A. Keyhani, T. Woung, and M. Rahman, "A hybrid fuzzy, neural network bus load modeling and predication," *IEEE Trans. Power Systems*, vol. 14, no. 2, pp. 718–724, 1999.
[49] A. Khotanzad, R. Afkhami-Rohani, T. L. Lu, A. Abaye, M. Davis, and D. J. Maratukulam, "ANNSTLF—A neural-network-based electric load forecasting system," *IEEE T. Neural Nets.*, vol. 8, no. 4, pp. 835–846, 1997.
[50] A. Khotanzad, R. Afkhami-Rohani, and D. Maratukulam, "ANNSTLF—Artificial neural network short-term load forecaster—Generation three," *IEEE Trans. Power Systems*, vol. 13, no. 4, pp. 1413–1422, 1998.
[51] A. Khotanzad, R. C. Hwang, A. Abaye, and D. Maratukulam, "An adaptive modular artificial neural network hourly load forecaster and its implementation at electric utilities," *IEEE Trans. Power Systems*, vol. 10, no. 3, pp. 1716–1722, 1995.
[52] S. J. Kiartzis, C. E. Zoumas, J. B. Theocharis, A. G. Bakirtzis, and V. Petridis, "Short-term load forecasting in an autonomous power system using artificial neural networks," *IEEE Trans. Power Systems*, vol. 12, no. 4, pp. 1591–1596, 1997.
[53] K. H. Kim, J. K. Park, K. J. Hwang, and S. H. Kim, "Implementation of hybrid short-term load forecasting system using artificial neural networks and fuzzy expert systems," *IEEE Trans. Power Systems*, vol. 10, no. 3, pp. 1534–1539, 1995.
[54] R. Lamedica, A. Prudenzi, M. Sforna, M. Caciotta, and V. O. Cencelli, "A neural network based technique for short-term forecasting of anomalous load periods," *IEEE Trans. Power Systems*, vol. 11, no. 4, pp. 1749–1756, 1996.
[55] K. Y. Lee, Y. T. Cha, and J. H. Park, "Short-term load forecasting using an artificial neural network," *IEEE Trans. Power Systems*, vol. 7, no. 1, pp. 124–132, 1992.
[56] K. Liu, S. Subbarayan, R. R. Shoults, M. T. Manry, C. Kwan, F. L. Lewis, and J. Naccari, "Comparison of very short-term load forecasting techniques," *IEEE Trans. Power Systems*, vol. 11, no. 2, pp. 877–882, 1996.
[57] C. N. Lu, H. T. Wu, and S. Vemuri, "Neural network based short term load forecasting," *IEEE Trans. Power Systems*, vol. 8, no. 1, pp. 336–342, 1993.
[58] S. Makridakis, S. C. Wheelwright, and R. J. Hyndman, *Forecasting—Methods and Applications*, 3rd ed, NY: John Wiley & Sons, 1998.
[59] T. M. Martinetz, S. G. Berkovich, and K. L. Schulten, ""Neural-gas" network for vector quantization and its application to time-series prediction," *IEEE T. Neural Nets.*, vol. 4, no. 4, pp. 558–568, 1993.
[60] G. A. N. Mbamalu and M. E. El-Hawary, "Load forecasting via suboptimal seasonal autoregressive models and iteratively reweighted least squares estimation," *IEEE Trans. Power Systems*, vol. 8, no. 1, pp. 343–348, 1993.
[61] J. S. McMenamin and F. A. Monforte, "Short-term energy forecasting with neural networks," *Energy J.*, vol. 19, no. 4, pp. 43–61, 1998.
[62] I. Moghram and S. Rahman, "Analysis and evaluation of five short-term load forecasting techniques," *IEEE Trans. Power Systems*, vol. 4, no. 4, pp. 1484–1491, 1989.

[63] O. Mohammed, D. Park, R. Merchant, T. Dinh, C. Tong, A. Azeem, J. Farah, and C. Drake, "Practical experiences with an adaptive neural network short-term load forecasting system," *IEEE Trans. Power Systems*, vol. 10, no. 1, pp. 254–265, 1995.
[64] H. Mori and H. Kobayashi, "Optimal fuzzy inference for short-term load forecasting," *IEEE Trans. Power Systems*, vol. 11, no. 1, pp. 390–396, 1996.
[65] S. E. Papadakis, J. B. Theocharis, S. J. Kiartzis, and A. G. Bakirtzis, "A novel approach to short-term load forecasting using fuzzy neural networks," *IEEE Trans. Power Systems*, vol. 13, no. 2, pp. 480–492, 1998.
[66] A. D. Papalexopoulos, S. Hao, and T. M. Peng, "An implementation of a neural network based load forecasting model for the EMS," *IEEE Trans. Power Systems*, vol. 9, no. 4, pp. 1956–1962, 1994.
[67] A. D. Papalexopoulos and T. C. Hesterberg, "A regression-based approach to short-term system load forecasting," *IEEE Trans. Power Systems*, vol. 5, no. 4, pp. 1535–1547, 1990.
[68] D. C. Park, M. A. El-Sharkawi, R. J. Marks II, L. E. Atlas, and M. J. Damborg, "Electric load forecasting using an artificial neural network," *IEEE Trans. Power Systems*, vol. 6, no. 2, pp. 442–449, 1991.
[69] J. H. Park, Y. M. Park, and K. Y. Lee, "Composite modeling for adaptive short-term load forecasting," *IEEE Trans. Power Systems*, vol. 6, no. 2, pp. 450–457, 1991.
[70] T. M. Peng, N. F. Hubele, and G. G. Karady, "Advancement in the application of neural networks for short-term load forecasting," *IEEE Trans. Power Systems*, vol. 7, no. 1, pp. 250–257, 1992.
[71] ——, "An adaptive neural network approach to one-week ahead load forecasting," *IEEE Trans. Power Systems*, vol. 8, no. 3, pp. 1195–1203, 1993.
[72] A. Piras, A. Germond, B. Buchenel, K. Imhof, and Y. Jaccard, "Heterogeneous artificial neural network for short term electrical load forecasting," *IEEE Trans. Power Systems*, vol. 11, no. 1, pp. 397–402, 1996.
[73] S. Rahman and O. Hazim, "A generalized knowledge-based short-term load-forecasting technique," *IEEE T. Power Syst*, vol. 8, no. 2, pp. 508–514, 1993.
[74] R. Ramanathan, R. Engle, C. W. J. Granger, F. Vahid-Araghi, and C. Brace, "Short-run forecasts of electricity loads and peaks," *Int. J. Forec.*, vol. 13, pp. 161–174, 1997.
[75] D. K. Ranaweera, G. G. Karady, and R. G. Farmer, "Effect of probabilistic inputs in neural network-based electric load forecasting," *IEEE T. Neural Nets.*, vol. 7, no. 6, pp. 1528–1532, 1996.
[76] ——, "Economic impact analysis of load forecasting," *IEEE Trans. Power Systems*, vol. 12, no. 3, pp. 1388–1392, 1997.
[77] R. Reed, "Pruning algorithms—A survey," *IEEE T. Neural Nets.*, vol. 4, no. 5, pp. 740–747, 1993.
[78] A. P. N. Refenes and A. D. Zapranis, "Neural model identification, variable selection and model accuracy," *J. Forecast.*, vol. 18, pp. 299–332, 1999.
[79] ——, *Principles of Neural Model Identification, Selection and Adequacy—with Applications to Financial Econometrics*: Springer-Verlag, 1999.
[80] R. Sadownik and E. P. Barbosa, "Short-term forecasting of industrial electricity consumption in Brazil," *J. Forecast.*, vol. 18, pp. 215–224, 1999.
[81] S. Sargunaraj, D. P. Sen Gupta, and S. Devi, "Short-term load forecasting for demand side management," *IEE Proc.—Gener. Transm. Distrib.*, vol. 144, no. 1, pp. 68–74, 1997.
[82] S. A. Soliman, S. Persaud, K. El-Nagar, and M. E. El-Hawary, "Application of least absolute value parameter estimation based on linear programming to short-term load forecasting," *Elect. Power & Energy Syst.*, vol. 19, no. 3, pp. 209–216, 1997.
[83] D. Srinivasan, A. C. Liew, and C. S. Chang, "Forecasting daily load curves using a hybrid fuzzy-neural approach," *IEE Proc.—Gener. Transm. Distrib.*, vol. 141, no. 6, pp. 561–567, 1994.
[84] D. Srinivasan, S. S. Tan, C. S. Chang, and E. K. Chan, "Parallel neural network-fuzzy expert system strategy for short-term load forecasting: System implementation and performance evaluation," *IEEE Trans. Power Systems*, vol. 14, no. 3, pp. 1100–1006, 1999.
[85] J. W. Taylor and S. Majithia, "Using combined forecasts with changing weights for electricity demand profiling," *J. Oper. Res. Soc.*, vol. 51, no. 1, pp. 72–82, 2000.
[86] J. Vermaak and E. C. Botha, "Recurrent neural networks for short-term load forecasting," *IEEE Trans. Power Systems*, vol. 13, no. 1, pp. 126–132, 1998.
[87] J. P. Vila, V. Wagner, and P. Neveu, "Bayesian nonlinear model selection and neural networks: A conjugate prior approach," *IEEE T. Neural Nets.*, vol. 11, no. 2, pp. 265–278, 2000.
[88] A. S. Weigend and N. A. Gershenfeld, Eds., *Time Series Prediction: Forecasting the Future and Understanding the Past*. Reading, MA: Addison-Wesley, 1994.
[89] A. S. Weigend and D. A. Nix, "Prediction with confidence intervals (local error bars)," in *Proc. Int. Conf. Neural Info. Processing (ICONIP'94)*, Seoul, Korea, 1994, pp. 847–852.
[90] A. S. Weigend and A. N. Srivastava, "Predicting conditional probability distributions: A connectionist approach," *Int. J. Neural Syst.*, vol. 6, pp. 109–118, 1995.
[91] H. T. Yang and C. M. Huang, "A new short-term load forecasting approach using self-organizing fuzzy ARMAX models," *IEEE Trans. Power Systems*, vol. 13, no. 1, pp. 217–225, 1998.
[92] H. T. Yang, C. M. Huang, and C. L. Huang, "Identification of ARMAX model for short term load forecasting: An evolutionary programming approach," *IEEE Trans. Power Systems*, vol. 11, no. 1, pp. 403–408, 1996.
[93] H. Yoo and R. L. Pimmel, "Short term load forecasting using a self-supervised adaptive neural network," *IEEE Trans. Power Systems*, vol. 14, no. 2, pp. 779–784, 1999.
[94] Z. Yu, "A temperature match based optimization method for daily load prediction considering DLC effect," *IEEE Trans. Power Systems*, vol. 11, no. 2, pp. 728–733, 1996.
[95] J. L. Yuan and T. L. Fine, "Neural-network design for small training sets of high dimension," *IEEE T. Neural Nets.*, vol. 9, no. 2, pp. 266–280, 1998.
[96] G. Zhang, B. E. Patuwo, and M. Y. Hu, "Forecasting with artificial neural networks: The state of the art," *Int. J. Forecast.*, vol. 14, pp. 35–62, 1998.

Henrique S. Hippert is an Assistant Professor at the Department of Statistics (Univ. Federal de Juiz de Fora, Brazil). Received the D.Sc. degree from Pontificia Universidade Catolica do Rio de Janeiro (Brazil). Main research interests: forecasting and neural networks.

Carlos E. Pedreira received the Ph.D. degree in 1987 from the Imperial College, University of London (EE Department). He has been an Associate Professor at the Pontificia Universidade Catolica do Rio de Janeiro since 1993 (Assistant Prof. 1987–1993). Dr. Pedreira is the Founding President of the Brazilian Neural Networks Council (1992–1994). Papers published in IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS, *Intern. J. Neural Systems, Mathematical Programming, J. Computation Intelligence in Finance*. Patents for medical devices registered in Brazil. Member of the editorial board of the *J. Comp. Intell. in Finance* since 1997, and of the *Computational Finance* Prog. Committee since 1994. Hobbies: art photography, gourmet cooking and wine tasting.

Reinaldo Castro Souza received the Ph.D. degree in 1979 from Warwick University, Coventry, UK (Statistics Department) and afterwards spent a period (1986/1987) as a Visiting Fellow at the Statistics Department of the London School of Economics. His major research interests are in the field of Time Series Analysis and Forecasting. He has been an Associate Professor at Pontificia Universidade Catolica do Rio de Janeiro since 1990. Member of the IIF (International Institute of Forecasters) and President of the Brazilian Operations Research Society since 1994. He has published papers in international journals such as: *J. Forecast.*, *Intern. J. Forecast.*, *J. Applied Meteorology*, *Latin-American O. R. J.*, *Stadistica*, among others. Hobbies: sports (tennis, volley-ball, soccer), arts and French literature.