HYBRID LEARNING ALGORITHM FOR INTELLIGENT SHORT-TERM LOAD FORECASTING

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
THE MIDDLE EAST TECHNICAL UNIVERSITY

BY

AYÇA KUMLUCA TOPALLI

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY
IN
THE DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING

JUNE 2003

Approval of the Graduate School of Natural and Applied Sciences

Prof. Dr. Tayfur Öztürk
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Doctor of Philosophy.

Prof. Dr. Mübeccel Demirekler
Head of Department

This is to certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Doctor of Philosophy.

Assoc. Prof. İsmet Erkmen
Supervisor

Examining Committee Members

Prof. Dr. Kemal Leblebicioğlu (Chairman) _____

Prof. Dr. Tuna Balkan _____

Prof. Dr. Sezai Dinçer _____

Assoc. Prof. Aydan M. Erkmen _____

Assoc. Prof. İsmet Erkmen _____

ABSTRACT

HYBRID LEARNING ALGORITHM FOR INTELLIGENT SHORT-TERM LOAD FORECASTING

Kumluca Topallı, Ayça

Ph.D., Department of Electrical and Electronics Engineering

Supervisor: Assoc. Prof. İsmet Erkmen

June 2003, 104 pages

Short-term load forecasting (STLF) is an important part of the power generation process. For years, it has been achieved by traditional approaches stochastic like time series; but, new methods based on artificial intelligence emerged recently in literature and started to replace the old ones in the industry. In order to follow the latest developments and to have a modern system, it is aimed to make a research on STLF in Turkey, by neural networks. For this purpose, a method is proposed to forecast Turkey's total electric load one day in advance. A hybrid learning scheme that combines off-line learning with real-time forecasting is developed to make use of the available past data for adapting the weights and to further adjust these connections according to the changing conditions. It is also suggested to tune the step size iteratively for better accuracy. Since a single neural network model cannot cover all load types, data are clustered due to the differences in their characteristics. Apart from this, special days are extracted from the normal training sets and handled separately. In this way, a solution is proposed for all load types, including working days, weekends and special holidays. For the

selection of input parameters, a technique based on principal component analysis is suggested. A traditional ARMA model is constructed for the same data as a benchmark and results are compared. Proposed method gives lower percent errors all the time, especially for holiday loads. The average error for year 2002 data is obtained as 1.60%.

Keywords: Artificial Intelligence, Hybrid Learning, Neural Networks, STLF.

ÖZ

AKILLI KISA DÖNEM YÜK ÖNGÖRÜMÜ İÇİN KARMA ÖĞRENİM ALGORİTMASI

Kumluca Topallı, Ayça

Doktora, Elektrik ve Elektronik Mühendisliği Bölümü

Tez Yöneticisi: Doç. Dr. İsmet Erkmen

Haziran 2003, 104 sayfa

Kısa dönem yük öngörümü (KDYÖ), güç üretim sürecinin önemli bir kısmıdır. Yıllardır, olasılıksal zaman dizisi gibi geleneksel yaklaşımlarla elde edilmiştir; fakat son zamanlarda yazında, yapay zekaya dayalı yeni yöntemler ortaya çıkmış ve endüstride eskilerinin yerini almaya başlamıştır. Son gelişmeleri izlemek ve çağdaş bir sisteme sahip olmak için, sinirsel ağlarla Türkiye'de KDYÖ üzerine bir araştırma yapmak hedeflenmiştir. Bu amaçla, Türkiye'nin toplam elektrik yükünü bir gün önceden öngören bir yöntem düşünülmüştür. Var olan geçmiş verileri ağırlıkları uyarlamada kullanmak ve bu bağlantıları değişen şartlara göre daha fazla ayarlamak için, çevrim-dışı öğrenim ile gerçek zamanlı öngörümü birleştiren karma bir öğrenim planı geliştirilmiştir. Daha doğru öngörüm için, adım aralığını döngülü ayarlamak da ayrıca önerilmiştir. Bütün yük tiplerini tek bir sinirsel ağ modeli kapsayamayacağından dolayı, veriler karakter farklılıklarına göre kümelendirilmişlerdir. Bundan başka, özel günler normal egitim kümelerinden çıkarılmış ve ayrı olarak ele alınmışlardır. Böylece, çalışma günlerini, hafta sonlarını ve özel tatil günlerini içeren tüm yük tipleri için bir çözüm önerilmiştir.

Giriş değiştirgenlerinin seçimi için, ana bileşen çözümlemesine dayalı bir teknik önerilmiştir. Denektaşı olarak geleneksel bir ARMA modeli oluşturulmuş ve aynı veriler için sonuçlar karşılaştırılmıştır. Önerilen yöntem her zaman, özellikle de tatil yükleri için daha düşük hatalar vermiştir. 2002 yılı verileri için ortalama yanılgı %1.60 olarak elde edilmiştir.

Anahtar Sözcükler: Yapay Zeka, Karma Öğrenim, Sinirsel Ağlar, KDYÖ.

For My Country…

ACKNOWLEDGMENTS

First of all, I would like to thank to my advisor Assoc. Prof. İsmet Erkmen, who has always guided me into right directions by his extensive knowledge and experience on the subject; and, encouraged, motivated and trusted me while doing this research. I wish to express my gratitude to the thesis progress committee members Assoc. Prof. Aydan M. Erkmen and Prof. Dr. Tuna Balkan for giving close attention to my work and for making valuable comments and suggestions since the beginning of the thesis. My thanks are also for the examining committee members for analyzing and evaluating my work.

I owe too much to my husband Mr. İhsan Topallı for his endless understanding, patience; and also for his great help in editing this thesis and software contributions.

I am thankful to Research Assistant Evren Gürkan for her continuous help and support that she has given to me always with a great politeness and smiling face. My appreciations are also for Research Assistant Selçuk Kılınç from Ninth September University, who has never avoided his helps in supplying and sharing the materials that I could not reach, especially for the literature survey part.

I wish to thank to Mr. Kaya Eriş for his helps in preparation of the hardcopies of this thesis.

I am grateful to Turkish Electricity Authority for opening up the load consumption database for this scientific research and to Mr. Mustafa Sert from Turkish General Directorate of Meteorology for providing me the temperature measurements.

The last but not the least, I would like to thank to my family who instilled me the free thinking and the joy of making researches.

TABLE OF CONTENTS

LIST OF TABLES

TABLE

LIST OF FIGURES

FIGURE

CHAPTER 1

INTRODUCTION

The forecasting of electric loads with lead times from a few minutes to seven days is usually referred to as short-term load forecasting (STLF). STLF plays a key role for the economic and secure operation of power systems. Basic functions such as unit commitment, hydro-thermal coordination, interchange evaluation and security assessment require a reliable short-term load forecast [4].

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 loads at several past hours, even on the loads at past days. Secondly because, there are many important variables that must be considered, such as weather-related variables [27]. 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 any research reducing it in a few percent points would be amply justified. As an illustration of the importance of forecast accuracy, it was estimated that an increase of 1% in forecast error caused an increase of 10 million pounds in operating costs per year for one electric utility in the United Kingdom [6].

The traditional methods to STLF can be broadly classified as time series and causal models. In the first one, the load is modeled as a function of its past observed values, whose examples are multiplicative autoregressive models, dynamic linear models, and methods based on Kalman filtering. General problems with the time series approach include the inaccuracy of prediction and numerical

instability. One of the reasons this method often gives inaccurate results is that it does not make use of weather variables such as temperature, humidity, wind speed and cloud cover. The time series approach mostly utilizes computationally cumbersome matrix-oriented algorithms which, in certain cases, may be unstable [54].

The second classical method models the load as a function of some exogenous factors, especially weather variables. Examples for this class are Box-Jenkins transfer functions, ARMAX models, non parametric regression and curve fitting procedures. These regression-based approaches use linear or piecewise-linear representations for the forecasting functions. The relationship between load and weather variables; however, is not stationary, but depends on spatio-temporal elements. Conventional regression approach does not have the versatility to address this temporal variation. It rather produces an averaged result [54]. Therefore, an adaptable technique is needed.

Recently, with the developments of artificial intelligence, alternative solutions to the STLF problem have been proposed. Expert systems, fuzzy inference and fuzzy-neural models have been tried out; however, the greatest deal of attention has been undoubtedly denoted to the use of neural networks over the last decade [27]. The main reason why neural networks became so popular lies in their ability to learn complex relationships through a training process with historical data that are difficult to model with conventional techniques. This capability enables neural networks to model the strong but nontrivial and nonlinear relationships that exist between future load and factors affecting it.

In the literature, several parameters are being used as the performance measures. MAPE is the most common one. Mean square error, root mean square error, mean square percentage error, histogram of the errors, etc. are also encountered. A comprehensive survey about the previous published works done on STLF by intelligent methods is given in the second chapter of this thesis. Best achievement among the most similar works to the research presented here can be given as [51]. Papadakis reported 1.67% error in this paper for year 1995 using fuzzy neural networks. Generally, it is expected to have forecast errors less than 2.00% to be

considered as being successful. This thesis prefers MAPE as the performance measure and has the best result as 1.60% for year 2002.

Nevertheless, the performance of neural networks in forecasting has not entirely convinced the skeptical researchers in this area. Recent reviews and text books on forecasting argue that there is little systematic evidence that neural networks might outperform standard forecasting methods, concluding that much work still needs to be done before they are accepted as established forecasting techniques [21, 26, 71].

It is hoped here that, this thesis would be a step in the way to the reconciliation of this skeptical attitude.

1.1. Objectives

The main objective of this thesis is to forecast Turkey's total electric load one day in advance. The resulting model should make real-time predictions, so the structure should support this property. It is aimed that the proposed system is robust, intelligent, and generalizable for changing conditions. It should give reasonable forecasts not only for regular data but also for days that have abnormalities due to holidays or large temperature variations. It should be successful enough to replace the traditional methods and it should bring a novel understanding to the solution of the problem. It should reach a performance better than 2.00% hourly forecast error as stated in the literature.

A model is to be proposed for satisfying all these requirements and hence for having a good place in literature.

1.2. Motivation

STLF is one of the most important problems in power industry. Many works have been reported all over the world on STLF application using neural networks. Beside this, classical STLF methods are being replaced with neural networks in electric generation facilities all over the world. It would be a success to show the possibility of doing a research on this subject in Turkey as good as the others.

Furthermore, as the neural network approach gives better results than the classical methods, it would decrease the redundant excessive generation and thus lower the cost. That would be advantageous to Turkey's economy.

Another motivation is to present a successful neural network model and application against the unacceptance of neural networks in some science communities.

Almost all neural network researches on STLF utilize feedforward type configurations, there are very few work reported about STLF with recurrent type neural networks. However, dynamics of the process could be better modeled with feedbacks.

Some of the works in the literature do not consider effects due to special days; they have excluded such conditions from their systems. On the other hand, in order to apply it to real life, it is important to propose a model that has a solution for all conditions.

And also, to present a model that makes use of historical data but at the same time that adapts itself to new coming data and thus has the ability to perform real-time forecasts would be a complete work.

1.3. Methodology

In order to propose a solution to the STLF problem, artificial intelligence approach is chosen, as an alternative to the traditional regression-based approaches.

Elman network, which is a subclass of recurrent neural networks is used as the structure. This kind of networks has not only feedforward but also feedback connections and this construction helps learning.

A hybrid learning algorithm is proposed which combines off-line training with real-time learning to take the advantage of experience gained by past data and to make instantaneous forecasts. Since the process is real-time, the model should be dynamic. Therefore, step size is made adaptive rather than to be fixed.

Knowing that one neural network will not be capable of handling all load types, several data clusters are formed. As a resemblance measure, correlation analysis is selected. Thinking that the past loads, temperature and time (hour, day, season, etc.) play the greatest roles in next day's load, they are used as the input variables to the proposed model.

1.4. Contributions

The main contribution of this thesis is forming a recurrent neural network that runs and learns in real-time to forecast electric load one day in advance. To do so, available data is examined and the best connection weights, for starting to the real-time forecasts, are found through an off-line training phase.

In order to determine about optimum size of the input vector, a preprocessing method, using the principal component analysis, is proposed. By this way, input selection is based on a systematic approach rather than 'trial and error' style.

Step size variation method is adapted to Elman's type neural networks. Thinking that the real-time process should be intelligent enough to give quick responses to changing conditions, step size, which is an important parameter in learning, is made adjustable at each iteration.

Proposing a solution for all load profiles, including weekdays, weekends and holidays is another contribution. Neural network forecasts are sufficiently good for weekdays and weekends; but, they have to be revised and modified for holidays. Therefore, a new approach that combines all similar forecasts for past years and gives a correction term is suggested for such cases.

Load characteristics follow local trends and hence national databases show differences and previous results do not help understanding the local behaviors. Therefore, data analysis and clustering should be done specifically. Here, daily correlations are taken as a resemblance measure and data is clustered accordingly.

A similar work that makes use of such model configuration and proposes such methods has not been encountered in the literature.

1.5. Flow of the Thesis

The chapters of this thesis are organized as follows: an introduction to the process, known researches in the literature, mathematical basics of the used methods, data analysis and grouping, proposed method, performed experiments and obtained results, comparison with the classical methods, conclusions and suggestions for future works.

"Chapter 1: Introduction" starts with the definition of the STLF and states its importance. It mentions about various techniques having been applied to the STLF process and discusses their advantages and drawbacks. Their successes are compared with the results obtained in this research. Objectives, motivation and methodology are also described in this chapter.

"Chapter 2: Literature Survey" summarizes recent works on STLF and achieved results with basically neural network approach, all of which have been published in the international refereed journals.

"Chapter 3: Theoretical Background" is devoted to the mathematical preliminaries of recurrent neural networks, training algorithm and step size adaptation. Necessary equations, formulas, etc. to build up the model are provided there.

"Chapter 4: Data Analysis and Preprocessing" is about examining the electric load data and understanding their behavior before constructing the model. The daily and monthly profiles and daily correlations are investigated and a clustering approach is applied based on the information gained through this analysis. Neural networks are designed according to the clusters formed as explained in this chapter.

"Chapter 5: Proposed Model and Obtained Results for Regular Days" describes the developed hybrid learning algorithm in detail and explains all the models created for each cluster. Performed experiments are presented and results are discussed. Then the overall results are given in a section to see the big picture.

"Chapter 6: Handling the Special Days" shows the differences between regular and special holiday loads and proposes a correction method for special days to

adjust the neural network forecasts. Results for each special days types are presented, as well as the best and the worst output graphs.

“Chapter 7: Other Models for Comparison” gives the alternative models slightly different than the proposed one and their result in order to make comparisons and see the necessity of applied methods. This chapter also aims to give a similar STLF work done by statistical methods for benchmarking purposes. Since these traditional methods have proved their applicability in this area, it is necessary to compare the results achieved with the proposed new approach and with the classical ones. This chapter covers such comparisons together with brief theoretical information about the statistical approaches.

“Chapter 8: Principal Component Analysis for Feature Extraction” describes a preprocessing method to find the optimum input vector size of the neural networks, based on the idea of disposal of the component slowing the convergence.

“Chapter 9: Conclusions” provides the consequents of the thesis work, discussions of the obtained results, and suggestions for future researches.

CHAPTER 2

LITERATURE SURVEY

Plenty of works can be found in literature on STLF. Early researches use classical methods such as Box-Jenkins, ARMA, ARIMAX, SARIMAX, etc. [8, 20, 22, 23, 45, 46, 47, 50, 51, 56, 66, 69]. The first attention to artificial intelligence approaches in this field was paid in late 1980's and since then many works have been reported both from academy and industry utilizing neural networks or other artificial intelligence methods to solve the STLF problem. Today, a great number of facilities in the world (mainly in the USA) have replaced their old systems with these intelligent ones. It seems that this interest will growingly continue in the future.

Some of these works are mentioned here briefly. Only papers that have been published recently in the refereed journals are considered and the ones whose main interests are STLF by neural networks are taken into account in order not to lose focus.

Abraham and Nath's work [1] evaluates the use of two popular soft computing techniques and conventional statistical approach based on Box-Jenkins autoregressive integrated moving average (ARIMA) model to predict electricity demand in the State of Victoria, Australia. The soft computing methods considered are an evolving fuzzy neural network and an artificial neural network trained using scaled conjugate gradient algorithm (SCGA) and backpropagation. The forecast accuracy is compared with the forecasts used by Victorian Power Exchange (VPX) and the actual energy demand. To evaluate, they consider load

demand patterns for 10 consecutive months taken every 30 minutes for training the different prediction models. They obtain root mean square error (RMSE) of 0.0092 for the neuro-fuzzy system, 0.0323 for the SCGA, 0.118 for backpropagation, and 0.0423 for ARIMA. Their results show that the neuro-fuzzy system performed better than neural networks, ARIMA model and the VPX forecasts.

AlFuhaid et al. [2] use a small neural network that pre-processes some of the data and produces estimates of peak load, valley load, and total load, which are fed, together with some other data, such as relative humidity or wind speed, saying that they have a strong effect on the human sensation of thermal discomfort, into a very large neural network that computes next day's profile. They forecast loads at each half-hour and so their model needs 48 output neurons.

The paper of Bakirtzis et al. [4] presents the development of a neural network based STLF model for the Energy Control Center of the Greek Public Power Corporation (PPC). The model forecasts daily load profiles with a lead time of one to seven days. Attention is paid for the accurate modeling of holidays. A fully connected three layer feedforward neural network is used in this development. The neural network has 63 input neurons; the first 48 represent historical hourly load data for two past days, 49-56 are maximum and minimum daily temperatures for the present day and the temperature forecasts for the forecast day at two weather stations, 57-63 represent the day of the week, bit encoded. The model has 24 hidden neurons and 24 output neurons representing next day's 24 hourly forecast loads. A single neural network is used for the load forecasting of all day types and the model parameters are updated on daily basis. For holiday forecasting, they make use of the method proposed by Papalexopoulos but they improve it especially for forecasting sequences of holidays. The average result for one-day ahead forecast of normal days is 2.24%, while it is 3.56% for the holidays.

Charytoniuk and Chen [7] propose an approach to very short term load forecasting by neural networks. The developed forecasting system predicts eight values of load for the time leads from 20-90 minutes in 10-minute increments by separate

neural networks and they are retrained once a day. Time of day and relative increments in load from the recent past are used as input variables. For the 20-60 minute forecasts, MAPE lies in a range of 0.4-1.1%. Average of these forecasts during a four-week summer period is 0.66%.

Chen et al. [9] suggest a nonfully connected network, in order to reduce the number of weights. They try to adapt the Box-Jenkins methodology for fitting ARIMA models, and select the lags by the analysis of the autocorrelation functions and the partial autocorrelation functions. 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.

Choueiki et al. [10] investigate the ability to solve the STLF problem with neural networks by conducting a fractional factorial experiment. They analyze the results of the experiment and identify the factors, such as number of neurons and layers, activation functions, stopping criteria, etc., that affect forecasting performance. They derive rules to build a "quasioptimal" neural network. They come up with a rather unusual architecture with a recurrent neural network and sinusoidal activation functions in the hidden layers.

The use of a weighted least squares procedure when training a neural network to solve the STLF problem is investigated in the work of Choueiki et al. [11]. Results indicate that during the on-peak periods (10:00-17:00), the neural network that implements the weighted least squares procedure (NNWLS) outperforms the neural network that implements the least squares procedure (NNLS) with MAPEs 1.72% and 2.26%, respectively. During the off-peak period (01:00-09:00, 18:00-00:00) however, NNLS model is superior with errors 2.63% and 3.22%, respectively. They suggest that the weighted least squares procedure be further studied by utilities which experience large variations in their marginal energy cost profiles and use neural networks in solving the STLF problem.

Since the load series are often nonstationary, Chow and Leung [12] suggest that neural networks could be used to model the first differences of the series, as nonlinear extensions to the ARIMA models. They deal only with the week day profiles, discarding the weekends and holidays.

Daneshdoost et al. [13] classify the data into 48 fuzzy subsets according to temperature and humidity; each subset was modeled by a separate neural network. However, the data are subdivided into too many classes, and there are not enough profiles left in each class to permit network training.

Dash et al. [15] use a functional-link network that has only one neuron. The inputs are a set of sinusoids, past forecasting errors, and temperatures. The neuron has 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.

The two works of Drezga and Rahman [16, 18] use 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. 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. Average forecast errors for 24-hour lead time are 2.05% for weekdays and 2.47% for weekends.

Erkmen and Topallı report four methods for STLF in their recent work [19]. These methods are generalized learning vector quantization for data clustering, genetic algorithms for optimum topology, neural networks and fuzzy logic for forecasting. The one giving the most successful forecasts is a hybrid neural network model which combines off-line and on-line learning and performs real-time forecasts 24-hour in advance. Loads from all day types are predicted with 1.73% average error for working days, 1.75% for Saturdays and 2.06% for Sundays.

Hippert et al. [27] present a review which examines a collection of papers, published between 1991 and 1999, that report the application of neural networks to STLF. Their aim is to evaluate the ways in which the neural networks proposed in these papers are designed and tested.

Ho et al. [28, 29] use a neural network to forecast next day's peak load, which is needed as an input to the expert system that forecasts next day's profile.

The paper of Khan et al. [32] presents a comparative study of six soft computing models namely multilayer perceptron networks, Elman recurrent neural network, radial basis function network, Hopfield model, fuzzy inference system and hybrid fuzzy neural network for the hourly electricity demand forecast of Czech Republic. The soft computing models were trained and tested using the actual hourly load data obtained from the Czech Electric Power Utility for seven years (January 1994 – December 2000). A comparison of the proposed techniques is presented for predicting 48 hourly demands for electricity. Simulation results indicate that hybrid fuzzy neural network and radial basis function networks are the best candidates for the analysis and forecasting of electricity demand for the experimented data, with the following MAPEs: For weekday forecast, 1.00% by radial basis function networks, 0.94% by fuzzy neural network; and for weekend forecast, 1.32% by radial basis function networks, 2.00% by fuzzy neural network

Khotanzad et al. [33] presents lately an approach to short-term load forecasting in a deregulated and price-sensitive environment. A real-time pricing type scenario is envisioned where energy prices could change on an hourly basis with the consumer having the ability to react to the price signal through shifting his electricity usage from expensive hours to other times when possible. In this work, a price-sensitive load forecaster is developed. This forecaster consists of two stages; an artificial neural network based price-insensitive load forecaster followed by a fuzzy logic system that transforms the price-insensitive load forecasts of the first stage into price-sensitive forecasts. This forecaster is tested on three price-sensitive databases and it is shown that it produces superior results to the price-insensitive neural network forecasters. By this approach, they improve their forecasts according to the databases by 18.5% (error is reduced from 2.87% to 2.34%), 12.5% (from 4.37% to 3.83%) and 14.6% (from 4.79% to 4.09%).

In [36], Khotanzad et al. propose a system in which the results of hourly, daily and weekly modules (38 neural networks in total) are linearly combined. This system is replaced in [35] by a smaller one, composed of 24 neural networks, one

for each hour of the day. Some of these authors propose a system with only two neural networks [34]. One of them is trained to produce a first forecast of tomorrow's profile. The other one is trained to estimate tomorrow's load changes with respect to today's loads; these changes, added to today's loads, make up a second forecast. The forecasts produced by both methods are linearly combined. It is argued that the second neural network allows the system to adapt more quickly to abrupt changes in temperature. They report that variations in the number of hidden neurons do not significantly affect forecasting accuracy.

Kiartzis et al. [37] train a neural network with data from a small utility, and found it necessary to smooth their sample data by a manual pre-processing procedure. They devise heuristics to regularize their data. They use piecewise linear-quadratic functions of the temperature as input variables.

In the paper of Kim et al. [38], a model for STLF that integrates neural networks and fuzzy expert system is presented. The load forecast is obtained by passing through two steps. In the first procedure, the neural networks are trained with the load patterns corresponding to the forecasting hour, and the provisional load is forecast by the trained neural network. In the second phase, the fuzzy expert systems modify it considering the possibility of load variations due to the changes in temperature and the day type, regular or holiday. Proposed model is tested with the data from Korea Electric Power Corporation. Results show that specific rules are required to deal with the consecutive holidays that have inconsistent periods with the previous year. Moreover, several rules are added for special days that have elections, rainy season, typhoon or special television programs. The proposed model predicts the load of holidays with a similar forecasting accuracy of non-holidays where the conventional methods or neural networks provide poor forecasts. Non-holiday results are given for four months: 1.25% for January, 1.16% for February, 1.30% for March, and 1.07% for April. The average error for holidays is 2.19%.

The purpose of the paper [39] by Kim et al. is to propose a new STLF method for special days in anomalous load conditions. These days include public holidays, consecutive holidays, and days preceding and following holidays. The proposed

method uses a hybrid approach of artificial neural network based technique and fuzzy inference method to forecast the hourly loads of special days which are classified into five different day types. Five neural network models for each day type are used to forecast the scaled load curves of special days, and two fuzzy inference models are used to forecast maximum and the minimum loads of special days. Finally, the results of these two models are combined to forecast the 24 hourly loads of special days. The proposed method is tested with actual load data of special days in Korea for the years of 1996 and 1997. The average percent error is 1.78% with a maximum value of 9.31%.

Lamedica et al. [40] suggest a system of 12 neural networks, one for each month of the year. In order to improve the forecast for anomalous days, the daily load profiles themselves are classified by a 8×8 Kohonen self-organized map. The classes are then interpreted by the system operator, so that the class to which the target day belongs can be predicted.

An approach based on combined fuzzy theory and artificial neural network is developed for STLF in Liang and Cheng's work [42]. In the proposed approach, the Pearson analysis method is first applied to choose two load patterns of historical load records that are similar to the load pattern to be forecast. Then, these two load patterns and required weather parameters are fuzzified and input to a neural network for training or testing purposes. The proposed approach is demonstrated by the practical data from Taiwan Power Company.

Lu et al. [43] experiment with three neural networks to model data from two utilities, and concluded that neural networks are system dependent, i.e., must be tailored for each specific utility. They ignore the weekend/weekdays distinction, but get poor results as a consequence on weekends and holidays.

A fuzzy modeling method is developed in the paper of Mastorocostas et al. for STLF [44]. Identification of the premise part and consequent part is separately accomplished via the Orthogonal Least Squares (OLS) technique. Input selection is automatically performed, given an input candidate set of arbitrary size, formulated by an expert. The performance of the model is evaluated using the

load data from the Greek interconnected power system. The attained yearly average forecast error is 1.76% for year 1995.

In the work of Mohammed et al. [48] the hourly loads are 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 neural networks that made up a very large system. They report their results in mean absolute errors stating that the utilities would rather evaluate forecasting systems by that kind of errors produced. They include a histogram of the errors.

A modeling technique based on the fuzzy curve notion is proposed in Papadakis' paper [51] to generate fuzzy models for STLF. Different forecast models are developed for each day type in every season. The model is considered as a fuzzy neural network described in terms of a parameter vector and is trained using a genetic algorithm with enhanced learning and accuracy attributes. The performances of the developed fuzzy models are tested using load data of the Greek interconnected power system. They achieve a MAPE of 1.67% with the data from year 1995.

The paper of Papalexopoulos [53] presents the development and implementation of an artificial neural network based STLF model for the Energy Control Center of the Pacific Gas & Electric Company (PG&E), California, USA. Three types of variables are used as inputs to the neural network: historical loads, seasonal related and weather related information to forecast the peak and hourly loads. They cluster the data according to the day types, yielding seven separate neural networks. They use data from 1986 up to 1990 to make forecasts for 1991 and they compare the results with the regression based model existing at that time in PG&E's Energy Control Center. They suggest a procedure to improve the forecasting on holidays by modifying the basic forecast considering the holiday data of several past years. For hourly forecast, they report the error as 1.96%.

Park et al. [54] use three small sized neural networks to forecast hourly loads, total loads and peak loads, dealing only with the weekday profiles, discarding the weekends and holidays. Hourly temperature and load data for Seattle/Tacoma area

in the interval of 1 November 1988 – 30 January 1989 are collected by the Puget Sound Power and Light Company. MAPEs for the hourly, total and peak load forecasts are 2.04%, 1.68%, and 1.40%, respectively. The neural network typically shows higher error in the days when there are specific start-up activities such as Mondays with for example 4.19% error when the weekly average is 1.73%..

Peng et al. [55] propose two neural networks, one of which includes a linear neuron among the sigmoidal ones in the hidden layer. They design special classes for Mondays, Fridays and even for Thursdays thinking that weekend profile disturbs those days. They train their MLPs on small subsets that have data from only a few past days selected through statistical measures of similarity. That results in samples that are very homogeneous, but also very small.

In Saini and Soni's work [58] the daily electrical peak load forecasting is done using the feedforward neural network based upon the conjugate gradient backpropagation methods, by incorporating the effect of 11 weather parameters, the previous day peak load information, and the type of day. To avoid the trapping of the network into a state of local minima, the learning rate and error goal optimizations are performed. For redundancy removal in the input variables, reduction of the number of input variables is done by the principal component analysis method. The resultant data set is used for the training of a three-layered neural network. To increase the learning speed, the weights and biases are initialized according to the Nguyen and Widrow method. To avoid over fitting, an early stopping of training is done at the minimum validation error. The daily weather and electrical peak load data of four years of Haryana Vidyut Prasaran Nigam Ltd., India is taken for this study. Data from 1997 to 1999 are used for neural network training. Data of the year 2000 are used to test the trained neural network. Among the experimented conjugate gradient algorithms, the Powell-Beale method has given the best performance with 2.31% MAPE.

Senjyu et al. [59] suggest one-hour ahead load forecasting against rapid changes in temperature and load consumption in the forecast day. Their model is a three-layer feedforward neural network, with 20 units in the hidden layer and past

electric and temperature differences are given as input variables. However, they do not forecast the load itself but their model outputs a correction term. By adding this term to the past similar day data, they obtain a forecast, which is especially successful for special days. They use Euclidian norm to evaluate the similarity between a forecast day and a previous day. Their approach is illustrated through an application to the actual load data of Okinawa Electric Power Company in Japan, having a MAPE of 1.18%.

In the paper of Silva and Moulin [60], three techniques for the computation of confidence intervals for the neural network based short-term load forecasting are presented: error output, resampling and multilinear regression adapted to neural networks. A comparison of the three techniques is performed through simulations of on-line forecasting. It is shown that the performances of the confidence interval estimation methods strongly rely on the similarity between the past data and the current data.

Self-organizing neural networks are used in the work of Srinivasan et al. [61]. They train Kohonen networks to find typical profiles for each day of the week, and then use a fuzzy engine to compute corrections to those profiles, based on the weather variables and on the day types. Training of the FNN load forecasting model is carried out based on two years of scaled load data and weather forecast data, from 1 January 1993 to 31 December 1994, obtained from Singapore Power and the Meteorological Station of Singapore. Each Kohonen's network is trained to make forecast for one month. The average forecasting errors are 0.83% for weekdays including Saturdays, 0.75% for Sundays, and 0.84% for public holidays.

Taylor and Buizza [62] investigates the use of weather ensemble predictions, rather than traditionally used single weather point forecasts, in the application of neural networks to load forecasting for lead times from one to ten days ahead. The results show that using weather ensemble predictions leeds to improvements in accuracy for all ten lead times. These improvements brought the errors noticeably closer to those of the method using actual observed weather, substituted for the

weather variables in the neural network load model, which is an unattainable benchmark.

Topallı and Erkmen suggest hybrid learning for STLF in [64], whose details are to be described in detail in the forthcoming chapters. Clustering in this work is done on hourly basis, i.e., there are 24 data sets and hence 24 neural networks. Average error for year 2000 with this model is 2.45%, with the best daily error of 0.70%.

The aim of Tzafestas' paper [65] is to provide a collective unified survey study on the application of computational intelligence model-free techniques to STLF of electric power plants. Four classes of methodologies, namely neural networks, fuzzy logic, genetic algorithms and chaos are addressed. The paper covers eight representative case studies, which show the relative merits and performance that can be achieved by the various forecasting methods under a large repertory of geographic, weather and other peculiar conditions.

In the paper of Vermaak and Botha [68], it is postulated that the load can be modeled as the output of some dynamic system, influenced by a number of weather, time and other environmental variables. Recurrent neural networks are used to construct empirical models for this dynamic system. Because of the nonlinear dynamic nature of these models, the behavior of the load prediction system is captured in a compact and robust representation. This is illustrated by the performance of recurrent models on the STLF of the nation-wide load for the South African utility, ESKOM. A comparison with feedforward neural networks is also given. They obtain 2.57% error with feedworward neural networks, while it is reduced to 2.02% with recurrent ones.

Yoo and Pimmel [70] develop a self-supervised adaptive neural network to perform STLF for a large power system covering a wide service area with several heavy load centers. They propose a self-organizing neural network model, in which the neurons were split into two clusters; one of them received past load data, the other received temperature data. In using data from year 1993 as a test case, they find 1.92% error for day-ahead forecasting.

CHAPTER 3

THEORETICAL BACKGROUND

3.1. Recurrent Neural Networks

Recurrent neural networks are neural networks with one or more feedback loops. Given a multi layer perceptron (MLP) as the basic building block, the application of feedback can take a variety of forms. Feedback may be from the output neurons to the input layer. Yet another possible form is from the hidden neurons to the input layer. When the MLP has two or more hidden layers, the possible forms of feedback expand even further.

In this thesis, Elman's recurrent neural network is chosen as the model structure which has been shown to perform well in comparison to other recurrent architectures [3]. Elman's network contains recurrent connections from the hidden neurons to a layer of context units consisting of unit delays. These context units store the outputs of the hidden neurons for one time step, and then feed them back to the input layer, as shown in Figure 3.1.

The hidden neurons in Figure 3.1 have some record of their prior activation, which enables the network to perform learning tasks that extend over time. The hidden neurons also feed the output neurons that report the response of the network to the externally applied stimulus. Due to the nature of feedback around the hidden neurons, these neurons may continue to recycle information through the network over multiple time steps, and thereby discover abstract representations of time.

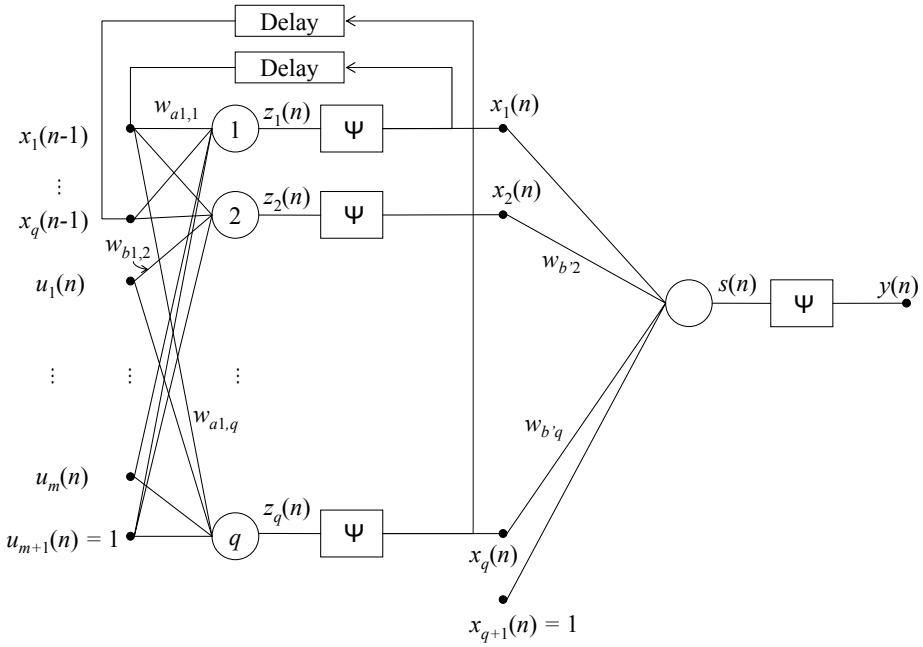

Figure 3.1. Elman's Recurrent Neural Network.

3.1.1. Why Recurrent Neural Networks are Chosen?

Neural networks are powerful tools that can capture the structure in data by learning. Often the batch learning paradigm is assumed, where the learner is given all training examples simultaneously and allowed to use them as often as desired. In large practical applications batch learning is experienced to be rather infeasible and instead on-line learning is employed.

In the on-line learning scenario, only one example is given at a time. So it is less memory consuming and at the same time it fits well into more natural learning. Apart from easier feasibility and data handling, the most important advantage of on-line learning is its ability to adapt to changing environments. If the learning machine does not detect and follow the change, it is impossible to learn the data properly and large generalization errors will result. With batch learning, changes

go undetected, whereas on-line learning will track the changes and yield good approximation results [49].

On-line learning is a stochastic process. The intrinsic noise due to the random pattern presentation enables transitions between different minima. This inherent noise of on-line learning makes it possible to escape from undesired local minima of the error potential on which the learning rule performs stochastic gradient descent [25].

Due to these reasons and thinking that forecasting the electric demand is a real-time process, on-line learning is taken into account in this work. Since 'feedback' is a valuable information in on-line learning to adapt the weights instantly, recurrent type neural networks which have not only feedforward connections, but also feedback associations are used.

The feedback enables recurrent neural networks to acquire state representations, which make them suitable devices for diverse applications such as nonlinear prediction and modeling. The use of feedback has the potential of reducing the memory requirement significantly [67].

3.1.2. Learning Algorithm for Elman's MLP Structure

Standard backpropagation is simply an efficient and exact method for calculating all the derivatives of a single target quantity, such as classification error with respect to a large set of input quantities. Recurrent learning extends backpropagation so that it applies to dynamic systems. This allows one to calculate the derivatives needed when optimizing an iterative analysis procedure, a neural network with memory, or a control system which maximizes performance over time [57].

The variables in Figure 3.1 can be expressed mathematically as:

$$z_j(n) = \overline{w}_{aj}^T(n)\bar{x}(n-1) + \overline{w}_{bj}^T(n)\overline{u}(n)$$

$$x_j(n) = \Psi\left(z_j(n)\right) \quad , \quad j = 1,...,q$$

$$s(n) = \overline{w}_{b'}^{T}(n)\overline{x}(n)$$

$$y(n) = \Psi\big(s(n)\big)$$

Since there is no feedback at the output layer of Elman's network, the weight update for this layer is done by standard error backpropagation.

Error is defined as:

$$e(n) = d(n) - y(n)$$

where $d(n)$ is the desired output for the n^{th} training sample.

The instantaneous sum of squared errors at time n is defined in terms of $e(n)$ by

$$\mathrm{E}(n) = \frac{1}{2}e^2(n)$$

The objective of the learning process is to minimize E(n). The adjustment applied to synaptic weight vector $\overline{w}_{b'}(n)$ is therefore determined by

$$\begin{aligned}
\Delta\overline{w}_{b'}(n) &= -\eta\frac{\partial \mathrm{E}(n)}{\partial\overline{w}_{b'}(n)} = -\eta\frac{\partial E(n)}{\partial e(n)}\frac{\partial e(n)}{\partial\overline{w}_{b'}(n)} = -\eta e(n)\frac{\partial e(n)}{\partial\overline{w}_{b'}(n)} \\
&= -\eta e(n)\frac{\partial e(n)}{\partial y(n)}\frac{\partial y(n)}{\partial\overline{w}_{b'}(n)} = \eta e(n)\frac{\partial y(n)}{\partial\overline{w}_{b'}(n)} = \eta e(n)\frac{\partial\Psi(s(n))}{\partial s(n)}\frac{\partial s(n)}{\partial\overline{w}_{b'}(n)} \\
&= \eta e(n)\dot{\Psi}(s(n))\frac{\partial s(n)}{\partial\overline{w}_{b'}(n)}
\end{aligned}$$

It can be shown that

$$\dot{\Psi}(t) = t(1-t)$$

So,

$$\Delta \overline{w}_{b'}(n) = \eta e(n)s(n)[1-s(n)]\frac{\partial}{\partial \overline{w}_{b'}(n)}\sum_{k=1}^{q+1} w_{b'k}(n)x_k(n) = \eta e(n)s(n)[1-s(n)]\overline{x}(n)$$

where η is the step size parameter.

For the hidden layer, weights between $x_i(n-1)$ and $z_j(n)$ are labeled as $w_{ai,j}(n)$ while the weights between $u_i(n)$ and $z_j(n)$ are called as $w_{bi,j}(n)$.

$$\Delta w_{ai,j}(n) = -\eta\frac{\partial E(n)}{\partial w_{ai,j}(n)} = \eta e(n)s(n)[1-s(n)]\frac{\partial}{\partial w_{ai,j}(n)}\sum_{k=1}^{q+1} w_{b'k}(n)x_k(n)$$

$$= \eta e(n)s(n)[1-s(n)]\sum_{k=1}^{q} w_{b'k}(n)\frac{\partial x_k(n)}{\partial w_{ai,j}(n)}$$

Let's define $\Lambda^k_{ai,j}(n)$ as the partial derivative of the state variable $x_k(n)$ with respect to the weight $w_{ai,j}(n)$. Hence,

$$\Delta w_{ai,j}(n) = \eta e(n)s(n)[1-s(n)]\sum_{k=1}^{q} w_{b'k}(n)\Lambda^k_{ai,j}(n)$$

This partial derivative can be extended as follows:

$$\Lambda_{ai,j}^{k}(n) \stackrel{\Delta}{=} \frac{\partial x_k(n)}{\partial w_{ai,j}(n)} = \frac{\partial x_k(n)}{\partial z_k(n)} \frac{\partial z_k(n)}{\partial w_{ai,j}(n)} = \dot{\Psi}\left(z_k(n)\right) \frac{\partial z_k(n)}{\partial w_{ai,j}(n)}$$

$$= z_k(n)[1 - z_k(n)] \frac{\partial}{\partial w_{ai,j}(n)} \left[\sum_{l=1}^{q} w_{al,k}(n) x_l\left(n-1\right) + \sum_{l=1}^{m+1} w_{bl,k}(n) u_l(n) \right]$$

Since the input signal does not depend on the weights,

$$\frac{\partial u_i(n)}{\partial w_{ai,j}(n)} = 0, \forall i$$

$$\Lambda_{ai,j}^{k}(n) = z_k(n)[1 - z_k(n)] \left[\sum_{l=1}^{q} w_{al,k}(n) \frac{\partial x_l\left(n-1\right)}{\partial w_{ai,j}(n)} + \delta_{kj} x_i\left(n-1\right) \right]$$

$$= z_k(n)[1 - z_k(n)] \left[\sum_{l=1}^{q} w_{al,k}(n) \Lambda_{ai,j}^{l}(n-1) + \delta_{kj} x_i\left(n-1\right) \right]$$

where δ_{kj} is the Kronecker delta,

$$\delta_{kj} = \begin{cases} 1 & k = j \\ 0 & elsewhere \end{cases}$$

Similarly,

$$\Delta w_{bi,j}(n) = \eta e(n) s(n)[1 - s(n)] \sum_{k=1}^{q} w_{b'k}(n) \Lambda_{bi,j}^{k}(n)$$

and

$$\Lambda_{bi,j}^{k}(n) = z_k(n)[1 - z_k(n)]\left[\sum_{l=1}^{q} w_{bl,k}(n)\Lambda_{bi,j}^{l}(n-1) + \delta_{kj}u_i(n)\right]$$

These recursive equations describe the nonlinear state dynamics of the learning process. Initial conditions are specified as

$$\Lambda_{ai,j}^{k}(0) = \Lambda_{bi,j}^{k}(0) = 0, \forall i, j, k$$

which implies that initially the recurrent network resides in a constant state.

3.2. Adapting the Step Size with Stochastic Optimization

It is known that the learning of neural networks is effective if the step size is appropriately chosen. A too small η makes learning impractically slow and is therefore not useful and a too large η spoils the convergence of learning. So the best thing to do could be using a variable learning rate. An adaptation method, proposed in [3] is used here.

Most probably, the simplest minimization technique is gradient descent:

$$\overline{w}(n+1) = \overline{w}(n) - \eta\overline{\nabla}E_w(n)$$

Such techniques are devised only for deterministic optimization. In many situations, however, one has to use stochastic optimization, the gradient of *E* being corrupted with noise:

$$\overline{w}(n+1) = \overline{w}(n) - \eta\left[\overline{\nabla} E_w(n) + \overline{r}(n)\right]$$

where $\overline{r}(n)$ is a zero-mean random vector. For example, in many real-life situations it is desired to follow an optimum that is changing slowly with time, and a stream of values of $\overline{\nabla} E_w(n)$ with corrupting noise is available. In the neural networks field, real-time, in other words, stochastic training might be wanted because of trying to model a slowly time varying system [3].

Step size adaptation methods for deterministic gradient optimization were proposed in the works of Kesten [31] and Jacobs [30]. The central idea behind these methods is that if successive updates of w_i are made in the same direction, then the movement along that component should be made faster. On the other hand, if successive updates are made in opposite directions, then the movement along that component should be made slower.

The method uses an independent, adaptive learning rate parameter η_i for each component w_i. The components are thus updated according to

$$w_i(n+1) = w_i(n) - \eta_i(n)\frac{\partial E(n)}{\partial w_i}$$

and the parameters are updated by

$$\eta_i(n+1) = \begin{cases} \eta_i(n) \cdot a & \text{if} \quad \dfrac{\partial E(n)}{\partial w_i} \cdot \dfrac{\partial E(n-1)}{\partial w_i} > 0 \\ \eta_i(n) \cdot b & \text{if} \quad \dfrac{\partial E(n)}{\partial w_i} \cdot \dfrac{\partial E(n-1)}{\partial w_i} < 0 \end{cases}$$

where typical values for a and b are 1.1 and 0.9.

This step size adaptation method has shown to be very effective in increasing the learning speed of MLPs trained by backpropagation.

This method, however, is not directly applicable to stochastic gradient optimization, since the partial derivatives are not available in such a case.

3.2.1. The Stochastic Adaptation Method

We consider iterative minimization algorithms of the form

$$\overline{w}(n+1) = \bar{f}\left(\overline{w}(n), \overline{\eta}(n), \bar{r}(n)\right)$$

We shall assume that we can obtain a noisy estimate of the gradient of the function to be minimized,

$$\overline{g}(n) = \overline{\nabla} E_w(n) + \bar{s}(n)$$

where $\bar{s}(n)$ is a random vector with zero mean.

The expected value of E after the next iteration will be

$$\left\langle E\left(\overline{w}(n+1)\right)\right\rangle = \left\langle E\left[\bar{f}\left(\overline{w}(n), \overline{\eta}(n), \bar{r}(n)\right)\right]\right\rangle$$

and a reasonable criterion for the choice of $\overline{\eta}(n)$ would be to choose the one that would minimize this expected value.

Calculation yields,

$$\frac{\partial\left\langle E\left(\overline{w}(n+1)\right)\right\rangle}{\partial \eta_i(n)} = \left\langle \left(\overline{g}(n+1) - \bar{s}(n+1)\right) \bullet \frac{\partial \bar{f}}{\partial \eta_i(n)} \right\rangle$$

where $\bullet$ represents the inner product. In several situations it is reasonable to assume that the random vectors $\bar{r}(n)$ and $\bar{s}(n+1)$ are independent from one other, since they are obtained in different iterations of the minimization procedure. For example, in stochastic backpropagation the random term obtained at each iteration depends on the pattern presented to the network at that iteration. If the patterns are drawn independently, from the training set, at every iteration, the assumption seems reasonable. Then,

$$\frac{\partial\left\langle E\left(\bar{w}(n+1)\right)\right\rangle}{\partial\eta_i(n)} = \left\langle \bar{g}(n+1) \bullet \frac{\partial\bar{f}}{\partial\eta_i(n)} \right\rangle$$

Let us then see how to indirectly make the derivatives approach zero. It is often the case that the optimal parameters $\bar{\eta}(n)$ change slowly with n. For example, in a stochastic gradient procedure, when approaching a quadratic minimum, the optimal parameters change asymptotically with $1/n$, and thus change very slowly in the asymptotic regime. Let us consider using a stochastic gradient adaptation of the parameter vector, aimed at driving the right hand side of the above equation to zero:

$$\eta_i(n) = \eta_i(n-1) + K_i(n) \cdot \frac{\partial\bar{f}}{\partial\eta_i} \bullet \bar{g}(n+1)$$

This expression uses $\bar{g}(n+1)$, which is not available at the n-th iteration. But given the assumption that $\bar{\eta}(n)$ changes slowly with n, one can use the values from the previous iteration,

$$\eta_i(n) = \eta_i(n-1) + K_i(n) \cdot \frac{\partial \bar{f}}{\partial \eta_i} \bullet \bar{g}(n)$$

which are all available at the *n*-th iteration.

Plain gradient descent corresponds to the update function

$$\bar{f} = \bar{w}(n) - \eta(n)\left[\bar{\nabla} E_w(n) + \bar{r}(n)\right]$$

where $\eta(n)$ is a scalar step size parameter. The expression in square brackets is a noisy estimate of the gradient. We can use the same estimate that was assumed available above, resulting in

$$\bar{f} = \bar{w}(n) - \eta(n) \cdot \bar{g}(n)$$

Applying the adaptation rule, it is obtained

$$\eta(n) = \eta(n-1) + K(n) \cdot \sum_i \bar{g}_i(n-1) \cdot \bar{g}_i(n)$$

which is the step size adaptation method for the basic gradient algorithm.

This basic algorithm is usually not efficient. It is preferable, in almost every case, to use the multiple step sizes variant, instead of the basic one.

Therefore a more general gradient based optimization algorithm is used. Assuming the update equation of the form

$$\bar{f} = \bar{w}(n) - \mathrm{N}(n) \cdot \bar{g}(n)$$

where the step size parameter has been replaced by a matrix **N**. The use of a matrix will allow us to use variants of the gradient procedure which move along a direction that does not necessarily coincide with that of the gradient, to accelerate the optimization.

The special case where **N** is restricted to be diagonal is taken, which corresponds to using an independent step size parameter for each of the components of $\overline{w}$. If η_i designates the *i*-th diagonal element of **N**, the parameter adaptation equation becomes

$$\eta_i(n) = \eta_i(n-1) + K_i(n) \cdot g_i(n-1) \cdot g_i(n)$$

It is convenient to adapt step sizes in a geometric way. This makes the adaptation method insensitive to the order of magnitude of the optimal step size parameters, and allows the adaptation to reach very large, as well as very small parameter values quickly. The geometric adaptation can be achieved by choosing

$$K_i(n) = k\eta_i(n-1)$$

yielding the update equation

$$\eta_i(n) = \eta_i(n-1)\left[1 + k \cdot g_i(n-1) \cdot g_i(n)\right]$$

The adaptation procedure should be insensitive to the specific function that is being minimized, so that the same value of the parameter *k* can be used for almost any *E*. However, here the adaptation speed depends heavily on the values of the partial derivatives of *E*. If *E* is multiplied by a constant *p*, *k* should be multiplied

by $1/p^2$, in order not to change the adaptation speed. To eliminate that dependency, the choice of $K_i(n)$ is modified to

$$K_i(n) = \frac{k\eta_i(n-1)}{\nu_i(n)}$$

where $\nu_i(n)$ is an exponential average of the square of $g_i(n)$, obtained through

$$\nu_i(n) = \gamma\nu_i(n-1) + (1-\gamma)[g_i(n)]^2$$

The parameter update equation then becomes

$$\eta_i(n) = \eta_i(n-1)\left[1 + k \cdot \frac{g_i(n-1) \cdot g_i(n)}{\nu_i(n)}\right]$$

It is found that the value $k = 0.01$ to be appropriate for most situations. This value yields a parameter adaptation speed of about 1% per iteration. $\gamma = 0.9$ seems appropriate for most situations.

3.2.2. Recurrent MLP Case

The proposed variable step size method should be adapted for the recurrent MLP structure. Recalling the weight adaptation formula for the output layer,

$$\Delta w_{b'i}(n) = \eta_i(n)e(n)s(n)[1-s(n)]x_i(n)$$

yields

$$g_{b'i}(n) = -e(n)s(n)[1-s(n)]x_i(n)$$

Then the update sequence looks like:

- $g_{b'i}(n) = -e(n)s(n)[1-s(n)]x_i(n)$
- $\nu_{b'i}(n) = \gamma\nu_{b'i}(n-1) + (1-\gamma)\left[g_{b'i}(n)\right]^2$
- $\eta_{b'i}(n) = \eta_{b'i}(n-1)\left[1 + k \cdot \frac{g_{b'i}(n-1) \cdot g_{b'i}(n)}{\nu_{b'i}(n)}\right]$
- $w_{b'i}(n+1) = w_{b'i}(n) + \eta_{b'i}(n)e(n)s(n)[1-s(n)]x_i(n)$

Similarly, the update sequence can be written for the hidden layer as:

- $$\Lambda_{i,j}^{k}(n) = \begin{cases} \dot{\Psi}(z_k(n))\left[\sum_{l=1}^{q} w_{al,k}(n)\Lambda_{i,j}^{l}(n-1)\right] & k \neq j \\ \dot{\Psi}(z_k(n))\left[\sum_{l=1}^{q} w_{al,k}(n)\Lambda_{i,j}^{l}(n-1) + \xi_i(n-1)\right] & k = j \end{cases}$$
- $g_{i,j}(n) = -e(n)\dot{\Psi}(s(n))\sum_{k=1}^{q} w_{b'k}(n)\Lambda_{i,j}^{k}(n)$
- $\nu_{i,j}(n) = \gamma\nu_{i,j}(n-1) + (1-\gamma)\left[g_{i,j}(n)\right]^2$
- $\eta_{i,j}(n) = \eta_{i,j}(n-1)\left[1 + k \cdot \frac{g_{i,j}(n-1) \cdot g_{i,j}(n)}{\nu_{i,j}(n)}\right]$
- $w_{i,j}(n+1) = w_{i,j}(n) + \eta_{i,j}(n)e(n)\dot{\Psi}(s(n))\sum_{k=1}^{q} w_{b'k}(n)\Lambda_{i,j}^{k}(n)$

with initializations: $\Lambda^k(0) = \mathbf{0}$, $g(0) = \mathbf{0}$, $\nu(0) = \mathbf{0}$, $\mathbf{N}(0) = \mathbf{0.5}$.

CHAPTER 4

DATA ANALYSIS AND PREPROCESSING

The available data for this research are Turkey's total hourly actual loads for the years 1999, 2000 (except January), 2001 (except 13 – 31 December) and 2002 (except 24 – 31 December), obtained through Turkish Electricity Authority; and, the hourly temperature measurements taken at Istanbul for the same years, obtained through Turkish General Directorate of Meteorology. In order to use these data in a meaningful and logical manner, first of all they should be closely analyzed and their dynamics should be clearly understood. Then they can be clustered into smaller sets according to some common characteristics and separate models can be built for each cluster. This is necessary because it has always been emphasized in the literature that it is impossible to reflect every different type of load behavior with a single model.

Statistical methods are widely used to analyze data. Here, correlation is considered as the major tool for getting an idea about the data. Detailed information is given in the following sections of this chapter.

4.1. Turkey's Electric Load Profile

The load profile is dynamic in nature. A broad spectrum of factors such as temporal, seasonal and annual variations affects the load level. In addition, total system load is subjected to random disturbances caused by sudden increase of large loads or outages. Figure 4.1 shows hourly load averages for each day of the week from years 1999 to 2002.

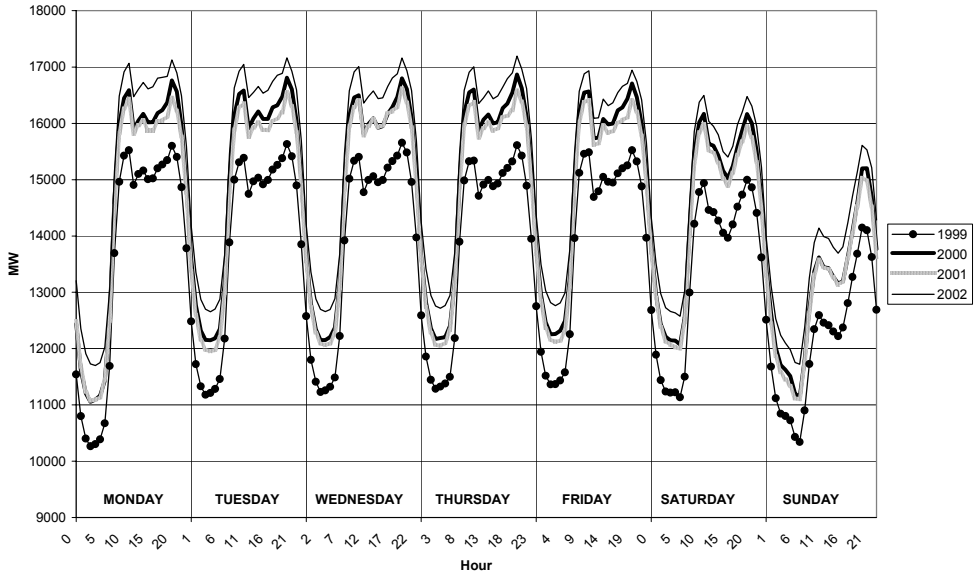

Figure 4.1. Hourly load averages for each day of the week (1999 – 2002).

The first observation is that, in years, related with the developing technology, industrialization, increasing number of electric household equipment etc., demand is getting higher.

As shown in Figure 4.1, apart from the absolute values, hourly averaged daily load shapes are almost identical for each year. Besides this graph gives an idea about how the electric load varies from hour to hour and day to day. It is seen that four working days (Tuesday to Friday) have very similar patterns. Monday demand is lower from the beginning of the day till the morning; but it catches the working-day trend for the rest of the day. Saturdays and Sundays are different than the other days.

Figure 4.2 represents the monthly averages of the load for the same years, 1999 – 2002.

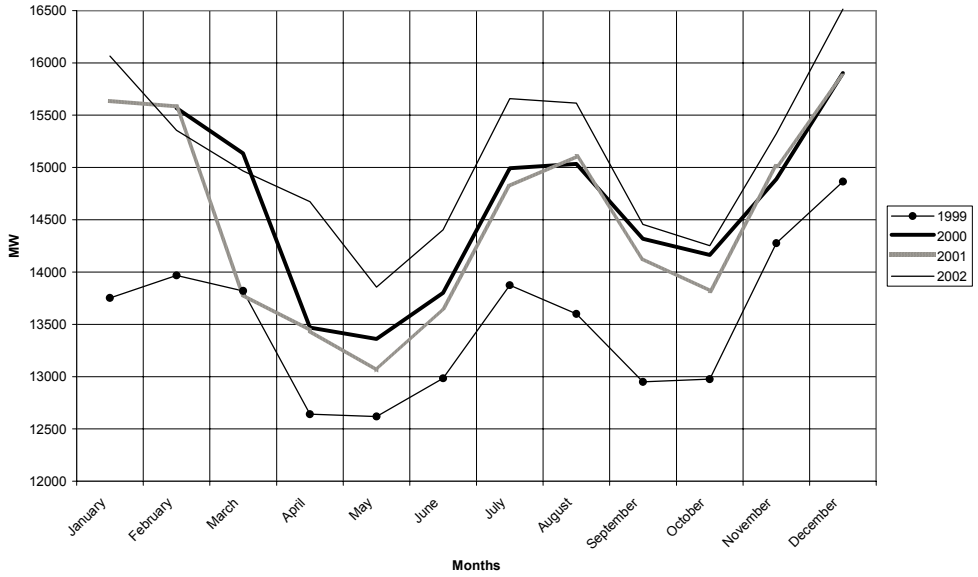

Figure 4.2. Monthly averages of the load (1999 – 2002).

Seasonal variations can be seen easily in Figure 4.2. Winter demand is the greatest. Not as high as winter months, summer load is still large. Spring, especially May has the lowest demand. Autumn time is on average, neither too big, nor too small. Climate effects or moving religious holidays might change these loads a little bit but the general trend remains almost the same.

4.2. Correlation Analysis

If the training set of a neural network contains patterns that have characteristics close to each other and if the output carries the same kind of information as the inputs then this model gives successful results. In order to perform a preliminary work for this hypothesis, a measure of the resemblance between the daily load sequences is thought to be established. In this respect, the correlation function is taken into consideration.

Cross correlation coefficients are computed for each data pair as follows:

$$C_{xy} = \frac{S_{xy}}{\sqrt{S_{xx} S_{yy}}}$$

with

$$S_{xy} = \sum_{i=1}^{n} |x_i - \bar{x}||y_i - \bar{y}| \ , \ S_{xx} = \sum_{i=1}^{n} (x_i - \bar{x})^2 \ , \ S_{yy} = \sum_{i=1}^{n} (y_i - \bar{y})^2$$

where x and y represent the data pairs, $\bar{x}$ and $\bar{y}$ are the mean values calculated over the samples and n is the number of samples.

Table 4.1 summarizes the daily correlations for electric load consumptions in year 2002.

Table 4.1. Daily load correlations in year 2002.

	Mon	Tue	Wed	Thu	Fri	Sat	Sun
Mon	1.0000	0.9879	0.9770	0.9753	0.9691	0.9409	0.7667
Tue		1.0000	0.9886	0.9851	0.9793	0.9561	0.8016
Wed			1.0000	0.9906	0.9850	0.9657	0.8177
Thu				1.0000	0.9913	0.9716	0.8251
Fri					1.0000	0.9793	0.8288
Sat						1.0000	0.8776
Sun							1.0000

As seen from Table 4.1, weekdays are highly correlated with each other; but, Saturday and Sunday have lower correlations with each other and with weekdays. Since correlation depends on covariance which is an unbiased variable, lower

demand effect for Monday morning cannot be seen here; but apart from this, Monday is the day which has the lowest correlations with the other weekdays, these numbers support the comments made on daily load profile graph given in Section 4.1.

Table 4.2 presents the correlations of the daily data with the days from the previous week. Note that, Table 4.1 is the correlations of the daily data with the other days in the same week.

Table 4.2. Correlations of weekly separated data in year 2002.

	Mon(w)	Tue(w)	Wed(w)	Thu(w)	Fri(w)	Sat(w)	Sun(w)
Mon(w-1)	0.9714						
Tue(w-1)	0.9659	0.9638					
Wed(w-1)	0.9729	0.9701	0.9615				
Thu(w-1)	0.9759	0.9773	0.9653	0.9651			
Fri(w-1)	0.9739	0.9775	0.9679	0.9653	0.9643		
Sat(w-1)	0.9592	0.9679	0.9610	0.9597	0.9550	0.9789	
Sun(w-1)	0.7923	0.8158	0.8155	0.8127	0.8028	0.8280	0.8819

For weekdays, it can be said that the same week data are more meaningful and valuable since correlations are reduced when the data are separated in time. For weekends, their data show different characteristics than the weekdays and it is seen that Saturday – Saturday or Sunday – Sunday data are more correlated than Saturday – weekday or Sunday – weekday data.

4.3. Data Clustering

Under the light of Turkey's electric load profile given above and correlation analysis performed on the available data, an efficient clustering can be done. First of all, special days should be excluded from the regular day data and handled separately since their characteristics are completely different. Then, four

weekdays (Tuesday-Friday) can be examined in the same cluster. It does not seem necessary to create a distinct cluster for each of these weekdays as they are highly correlated. Moreover, a cluster should be formed for the first hours of Monday, because they come just after the weekend and do not resemble the other weekdays. The remaining hours of Monday can be evaluated in the working days cluster. For weekends, two clusters should be formed as Saturdays and Sundays since they have unique characteristics. One exception can be done here, the single day national holidays that come across to Sundays are not too much different than the regular Sundays, so they can be put together in the same cluster. In summary, by looking at the trends and the statistics, one can cluster the data as follows:

1. Early Monday, hours between 00:00 – 08:00
2. Weekdays, from Monday 09:00 – Friday 23:00
3. Saturday
4. Sunday
5. Special Day

CHAPTER 5

PROPOSED MODEL AND OBTAINED RESULTS FOR REGULAR DAYS

A model is proposed here to forecast Turkey's total electric load one day in advance by the artificial intelligence approach. This model uses Elman type recurrent neural networks and provides a hybrid learning, which combines both off-line and real-time trainings. The aim is to prepare the model for real-time forecasts by training it with the available past data. Therefore, the hourly load data of a year (1999, 2000 or 2001) are used in off-line learning to adjust randomly initialized synaptic weights, and then the model undergoes real-time learning with the data of the next year (2000, 2001 or 2002, respectively). The next year's data are used as if they were real-time data by feeding them to the network in time order and only once. Errors are calculated as the actual data become available and weights are further updated in this phase.

Joining these two types of learning has an advantage of starting real-time application with the weights that are already brought near to optimal values.

Step size is adjusted in each real-time iteration by looking at the error gradient for improving the adaptation of the network.

In order to prevent model from over-fitting and memorizing the data in the off-line learning, the data are divided into training and validation sets. After randomizing the weights, input/output pairs from the training set are randomly presented for a predetermined number of cycles, which is taken as 1,000,000 here. Error is backpropagated and weights are adjusted in each cycle. At the end of each

100,000 cycle, weights are stored and the model is tested with the validation set, formed by 10% of the off-line data, chosen randomly and never given to the neural network during the off-line training. In this way, there are ten validation errors and corresponding ten weight sets when the off-line learning is finished. Weights giving the minimum validation error are considered as the final off-line weights.

Proposed hybrid learning algorithm can be summarized in the following steps:

A. Off-line phase:

1. Randomize weights
2. For 1,000,000 iterations
    i. Choose a random input/output pair from the training set, give to the network.
    ii. Find the neural network output by feedforward calculations.
    iii. Find the error, backpropagate it and update the weights.
    iv. At the end of each 100,000 cycle, give the validation data, find the average error. Save this error and the weights.
3. Take the weights which cause the lowest validation error.

B. Real-time forecast phase:

1. Present the next year's data in time order. Start with the weights found in the off-line phase. For each input,
    i. Calculate the output.
    ii. Find the percent error.
    iii. Update the weights and the step size.
2. Take the average of the percent errors.

As the detailed analysis is given in Chapter 4, data are clustered into four pieces: Early Monday (hours between 00:00 and 08:00), Weekdays (Monday 09:00 – 23:00, Tuesday, Wednesday, Thursday, and Friday), Saturday, and Sunday.

Separate neural networks are formed for each cluster and different input variables are used. But, the neural network configuration is always the same. An Elman network with one hidden layer having ten neurons, and sigmoid nonlinearity is the fixed model structure.

Loads that are used as inputs to the neural networks are normalized according to the yearly minimum and maximum values. There is no problem for off-line data since they are available for the whole year. However, real-time data for a complete year will not be at hand at the time of forecast, and thus the lowest and the greatest loads cannot be determined. So, real-time data should be normalized using the off-line data range. Knowing that the electric consumption is increasing every year, minimum value is taken as the minimum of the off-line data and maximum value is taken as 10% more of the off-line maximum. Consequently, both off-line and real-time data sets are normalized with these new minimum and maximum values in order to synchronize them.

There are two input parameters that are common to all neural networks: hour and season. To present the cyclic continuity, hour is given as a half sinusoid

$$h_C = \sin(\pi h / 24)$$

where h_C is the cyclical hour and h is the actual hour.

Season input is determined by looking at the monthly averaged loads, which are described in Chapter 4, in detail. To reflect these variations, season input is given as in Table 5.1.

Apart from these four clusters, special days are gathered together and considered in a separate chapter. A “special” forecast approach is proposed for them. The details of the proposed model for all regular data types, performed experiments, obtained results and discussions can be found below.

Table 5.1. Neural network input representing the season.

Months	Season Input
April, May	0.1
June, September, October	0.3
March, July, August, November	0.6
January, February, December	0.9

5.1. Early Monday Model

Electric load data analysis explained in Chapter 4 shows that the first hours of Monday have different characteristics than the other weekdays. Yet, they do not follow the weekend trend. They are like a transition phase between weekend and weekday behaviors. Therefore, they are taken into account in a separate cluster.

A data set is formed containing only Monday loads between hours zero and eight. This is used as the desired set for calculation of the error at the neural network output. As inputs, together with the time and weather information, loads from the past three weekdays are taken. Input and output variables can be shown as:

Output: $L_{Mon}(w,h)$

Inputs: h_C, s,

$L_{Fri}(w-1,h)$, $L_{Thu}(w-1,h)$, $L_{Wed}(w-1,h)$,

$T_{Fri}(w-1,h)$, $T_{Thu}(w-1,h)$, $T_{Wed}(w-1,h)$

where L represents the load; subscript of L is the day; w is the week of the year; h is the hour of the day, between zero and eight; h_C is the cyclical hour input, s is the season input, varying according to the month under consideration; and T is the temperature. The neural network is trained as described in the previous section.

Mondays which are national or religious holidays are not considered as desired outputs in the training set, instead they are handled separately. Furthermore, special days cannot be used as inputs since they have lower loads than a regular

Monday and mislead the weight updates. Therefore some of the Mondays cannot be used as desired outputs although they are regular days because of their special day type inputs. Such cases are listed in Table 5.2.

An experiment is performed with regular Mondays that can be both inputs and desired outputs. MAPEs for real-time forecasts are given in Table 5.3.

Table 5.2. Mondays that cannot be the desired outputs due to special day inputs.

1999	2000	2001	2002
25 Jan	22 May	03 Sep	02 Sep
05 Apr	04 Sep		
26 Apr			
24 May			
01 Nov			

Table 5.3. Real-time forecast results for Early Monday model.

	1999	2000	2001	2002
MAPE	19.38	2.32	1.89	1.95

One may notice that year 1999 error is quite high as compared to the other years. This is because there is no 1998 data to train the network off-line as the hybrid method proposes. Hence, real-time forecasts start with random weights and give unsuccessful results. Therefore, this is a good example to show that the hybrid learning is worthwhile. For years 2000, 2001 and 2002, results are acceptable. In 2002, 30 September is the best Monday with the lowest error (0.41%), and 15 April is the worst one as having the greatest error (4.67%). Real and neural network forecast values (NN) in this cluster of these two Mondays and corresponding percent errors are shown in the following figures (5.1 – 5.4).

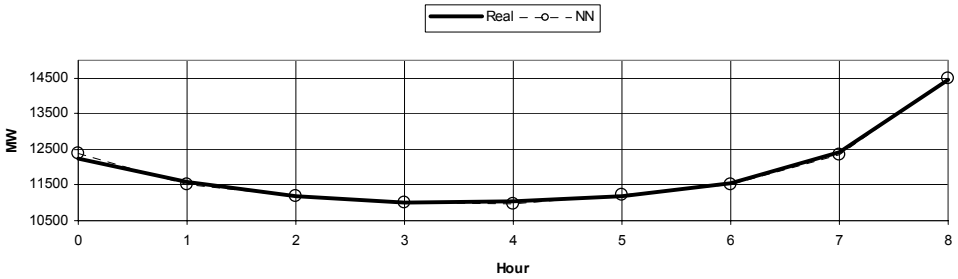

Figure 5.1. Actual and forecast values for the best Monday, 30 Sep 2002.

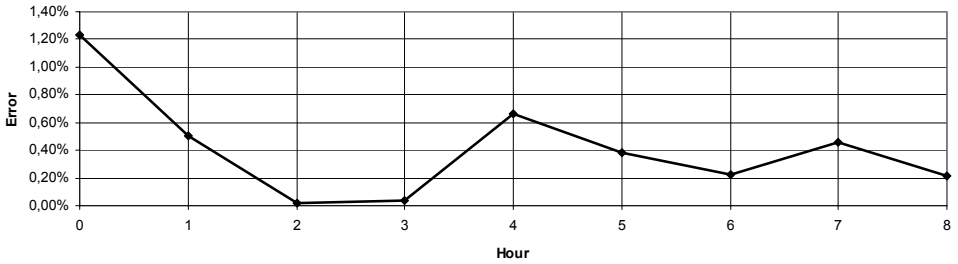

Figure 5.2. Percent errors for the best Monday.

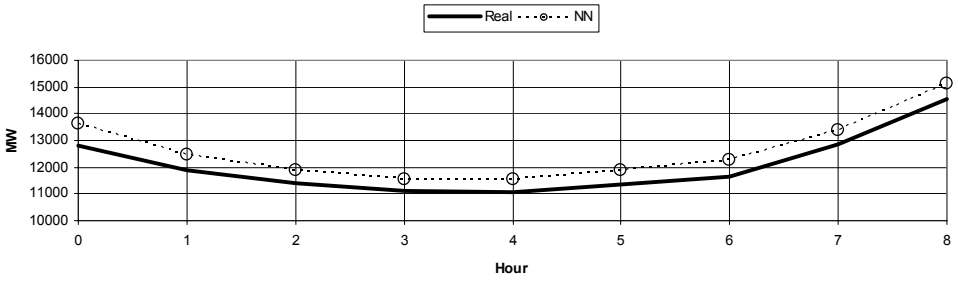

Figure 5.3. Actual and forecast values for the worst Monday, 15 Apr 2002.

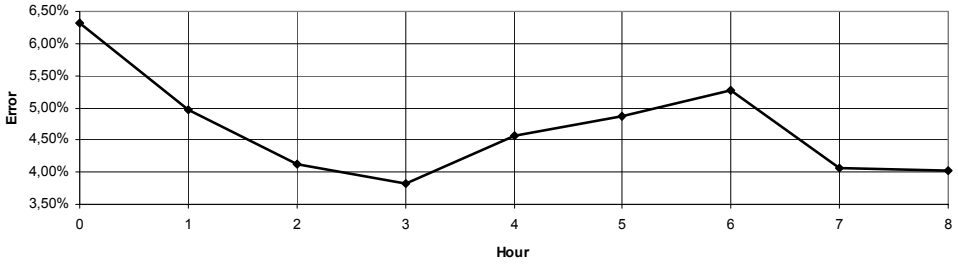

Figure 5.4. Percent errors for the worst Monday.

A solution is also needed for the Mondays that cannot be the desired outputs, as given in Table 5.2. Since they should not use special days as input, first, special days can be forecast as if they were regular days, then these regular data can be considered as inputs to be used in forecasting the days in Table 5.2. A second method might be making use of the load of just one previous day in the same cluster. In this case, data can be directly copied without involving the forecasting process. However, the cluster consists of Mondays, hence data are separated in one week time, or even more if there is a special day in between. Taking the previous data may not be enough since this interval is rather long. For example, temperature might change considerably in a week and this affects the load consumption. Therefore, together with the past Monday data, two previous Sundays can be used to express weekly variations. Copying the past Monday load after multiplying it by the ratio of Sunday loads would be a good approach. Results for these three approaches are represented in Table 5.4 for comparison.

It is understood from Table 5.4 that the third method is better than the others. The first one uses hypothetical values as inputs and this expands the error further. Introducing the correction term in the last one improves the results considerably. Following the weekly trend through the loads of two past Sundays and using the amount of change in copying the previous Monday load yield good results for the data that cannot be used in the neural network model.

Table 5.4. Average percent errors of two approaches for Mondays in Table 5.2.

Day	MAPE (Replacing special day inputs with their regular forecasts)	MAPE (Copying the previous Monday)	MAPE (Using the ratio of past two Sundays)
22 May 2000	11.47	1.10	2.37
04 Sep 2000	8.07	0.84	0.98
03 Sep 2001	5.40	2.58	1.76
02 Sep 2002	3.93	4.77	2.99
AVERAGE	**7.21**	**2.32**	**2.03**

5.2. Weekday Model

All working days from Monday to Friday, except the first nine hours of Monday, are gathered in a set and used for training the corresponding neural network. Data analysis given in Chapter 4 shows that, weekdays are highly correlated to each other; so for a weekday output, again weekday inputs should be used. Therefore, inputs and the output are organized as follows:

Output: $L(d,h)$

Inputs: h_C, d, s,

$L(d-1,h)$, $L(d-2,h)$, $L(d-3,h)$,

$T(d-1,h)$, $T(d-2,h)$, $T(d-3,h)$

where h, s, L and T are as defined before; and d represents the day of the week, a number between one and five to discriminate between weekdays. One can notice that, for the outputs from Monday, Tuesday and Wednesday, some inputs are coming from the working days of the previous week. Yearly averaged percent errors for real-time forecasts are presented in Table 5.5.

These results given in Table 5.5 are quite successful, except for the year 1999 as explained before, and have a meaning that to put the weekday data in the same set and to forecast the load with the correlated data are the right things to do. A question may arise what would happen if separate neural networks were used for

each weekday instead of using a single neural network for all of them. It is for sure that this will increase the learning time (unless parallel processing is possible) and make it hard to save and maintain the parameters but it might be worth trying it if it reduces the error.

Table 5.5. Percent errors for Weekday model.

	1999	2000	2001	2002
MAPE	10.66	1.52	1.37	1.30

Hence, five new neural networks are constructed each for a different weekday (for the last 15 hours of Monday and for the entire hours of the others). The resulting errors can be seen in Table 5.6.

Table 5.6. MAPEs for separate daily neural networks.

Day	2000	2001	2002
Mon (9-23)	2.13%	1.82%	1.81%
Tue	1.44%	1.43%	1.25%
Wed	1.44%	1.26%	1.26%
Thu	1.48%	1.29%	1.21%
Fri	2.14%	1.52%	1.22%
AVERAGE	**1.72%**	**1.46%**	**1.35%**

The results for the single model which are given in Table 5.5 are overall results. However, comparing them with the daily neural network results in Table 5.6 does not mean much. They can be compared with the averaged errors in Table 5.6 and

it is seen that single neural network results are better; or, it is possible to find the daily distributions of these overall errors and make daily comparisons afterwards. Instead of averaging all the data, errors are averaged accordingly to the belonging days and they are shown in Table 5.7.

Table 5.7. Daily distributions of the errors in Table 5.5.

Day	2000	2001	2002
Mon (9-23)	2.07%	1.80%	1.95%
Tue	1.49%	1.34%	1.22%
Wed	1.41%	1.23%	1.22%
Thu	1.37%	1.21%	1.26%
Fri	1.47%	1.41%	1.09%
AVERAGE	**1.52%**	**1.37%**	**1.30%**

Now, the daily results presented in Table 5.6 and 5.7 can be compared. Separate neural network approach show better performances only in three cases: Tuesday 2000, Monday 2002 and Thursday 2002. But even these improvements are not so big. Correlation analysis has already showed that weekdays are carrying very much similarities. Therefore, there is no need to keep separate daily models and it is appropriate to use the single neural network.

To show the best and the worst daily performances in year 2002, Figures 5.5 to 5.8 are given. 22 August 2002 has the lowest forecast error with 0.48%, whereas 6 November 2002 has the greatest error with 3.62%.

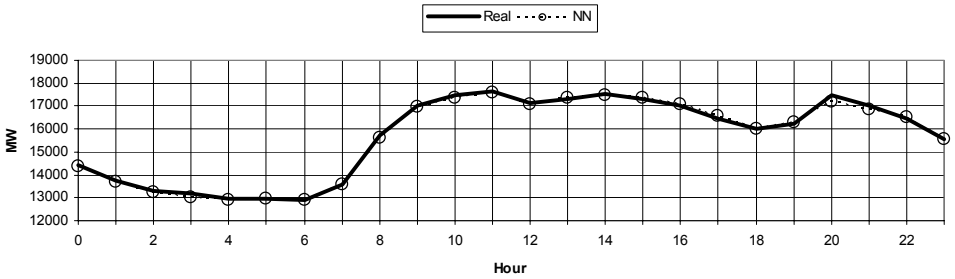

Figure 5.5. Actual and forecast values for the best weekday, 22 Aug 2002, Thu.

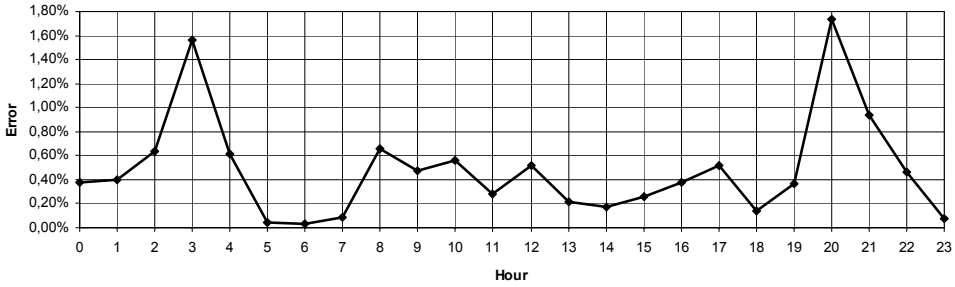

Figure 5.6. Percent errors for the best weekday.

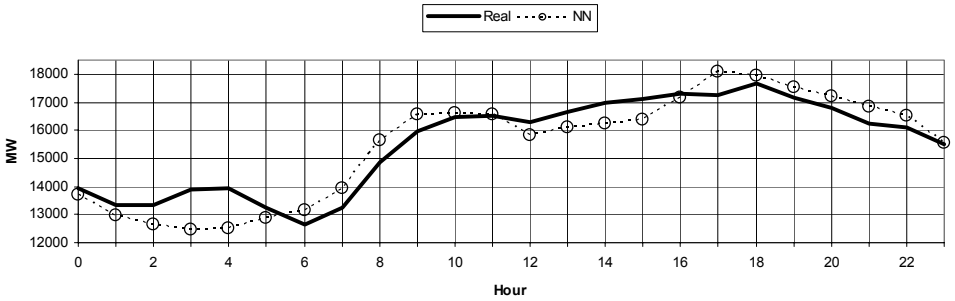

Figure 5.7. Actual and forecast values for the worst weekday, 6 Nov 2002, Wed.

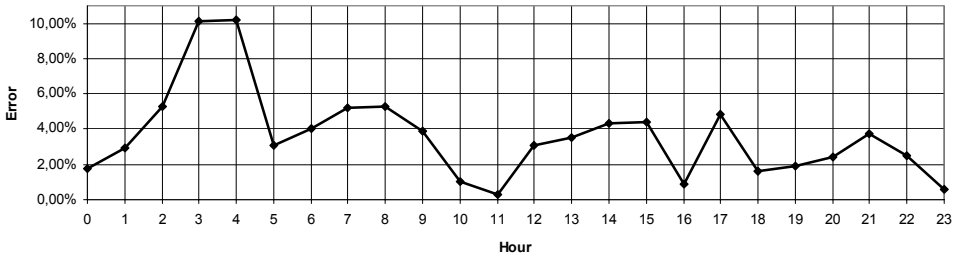

Figure 5.8. Percent errors for the worst weekday.

As in the Early Monday model, several days cannot be the desired outputs whose inputs coincide a special day. They are handled with two methods mentioned in the previous section and results are summarized in Table 5.8.

The third column in Table 5.8 shows that copying the previous data is slightly better than the first method. Since the resemblance between weekdays is high, there is no surprise to obtain such results. The data in this case are close to each other; therefore, no ratio term is needed.

Table 5.8. Weekdays whose inputs have special day(s) and their forecast errors.

Day	MAPE (Replacing Special day inputs with their regular forecasts)	MAPE (Copying the previous Weekday)
21 Mar 00 Tue	2.84	1.83
22 Mar 00 Wed	2.92	0.83
23 Mar 00 Thu	2.17	1.01
22 May 00 Mon	1.28	1.13
23 May 00 Tue	1.84	1.85
24 May 00 Wed	0.86	0.94
31 Aug 00 Thu	3.18	1.91
01 Sep 00 Fri	1.23	2.57
04 Sep 00 Mon	4.58	2.76
13 Mar 01 Tue	2.46	4.32
14 Mar 01 Wed	1.34	1.10
15 Mar 01 Thu	1.79	0.90
24 Apr 01 Tue	6.36	4.34
25 Apr 01 Wed	1.63	2.97
26 Apr 01 Thu	1.06	1.00
31 Aug 01 Fri	5.91	5.36
03 Sep 01 Mon	3.21	2.58
04 Sep 01 Tue	1.03	3.42
30 Oct 01 Tue	4.84	4.12
31 Oct 01 Wed	1.68	3.48
01 Nov 01 Thu	2.66	2.33
26 Feb 02 Tue	6.78	5.78
27 Feb 02 Wed	2.63	2.62
28 Feb 02 Thu	1.37	0.57
24 Apr 02 Wed	3.39	3.11
25 Apr 02 Thu	2.51	3.58
26 Apr 02 Fri	2.45	2.05
02 Sep 02 Mon	1.27	2.43
03 Sep 02 Tue	1.49	2.67
04 Sep 02 Wed	1.00	0.84
30 Oct 02 Wed	4.16	3.72
31 Oct 02 Thu	2.97	2.44
01 Nov 02 Fri	1.74	1.26
10 Dec 02 Tue	3.47	4.92
11 Dec 02 Wed	3.67	2.62
12 Dec 02 Thu	3.57	1.72
AVERAGE	**2.70**	**2.53**

5.3. Saturday Model

Looking at the correlation results given previously, it is thought that Saturday load is represented best by the past Saturday data. Hence, input/output configuration becomes as follows:

Output: $L_{Sat}(w,h)$

Inputs: h_C, s,

$L_{Sat}(w-1,h)$, $L_{Sat}(w-2,h)$, $L_{Fri}(w,h)-L_{Fri}(w-1,h)$,

$T_{Sat}(w-1,h)$, $T_{Sat}(w-2,h)$, $T_{Fri}(w,h)-T_{Fri}(w-1,h)$

Here apart from the past Saturday data, Friday differences are taken to indicate the load tendency of the current week with respect to the last week. Table 5.9 lists the results.

Table 5.9. Yearly averaged percent errors for Saturday model.

	1999	2000	2001	2002
MAPE	17.35	2.02	1.70	1.45

The proposed algorithm suggests quite successful forecasts since they match to the actual values with very low errors. For the year 1999, the hybrid learning is not applicable and hence the resulting error is so high. In order to see the performance limits, the best and the worst cases are given for year 2002. 12 October has the most successful result with 0.85% error; and 26 January fails with 2.18% error. Figures 5.9 – 5.12 show the hourly loads and errors for these two Saturdays.

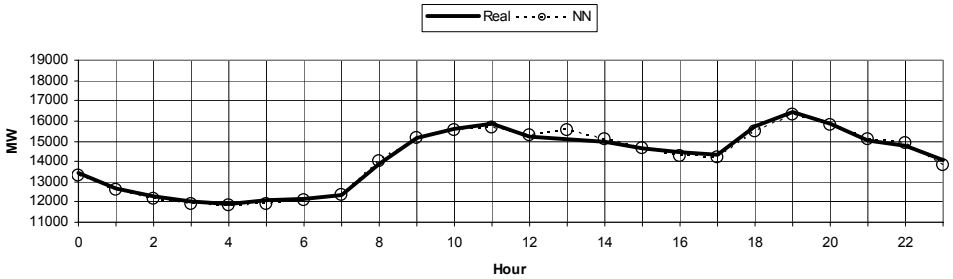

Figure 5.9. Actual and forecast values for the best Saturday, 12 Oct 2002.

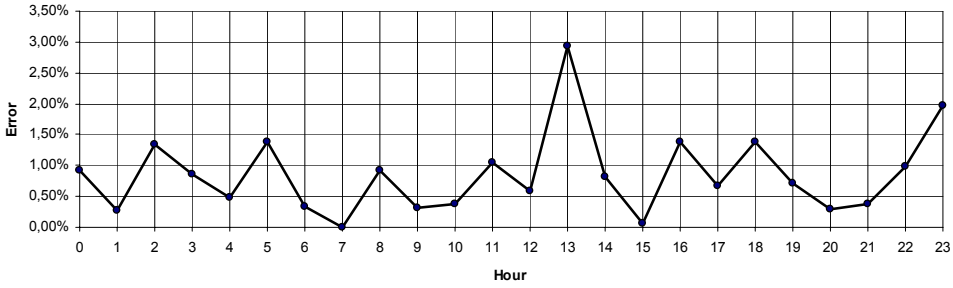

Figure 5.10. Percent errors for the best Saturday.

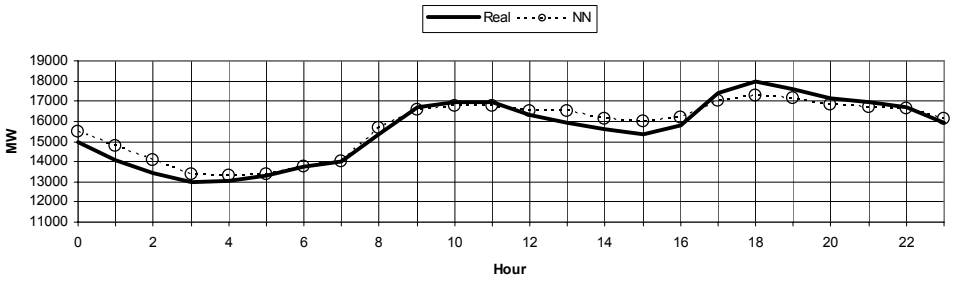

Figure 5.11. Actual and forecast values for the worst Saturday, 26 Jan 2002.

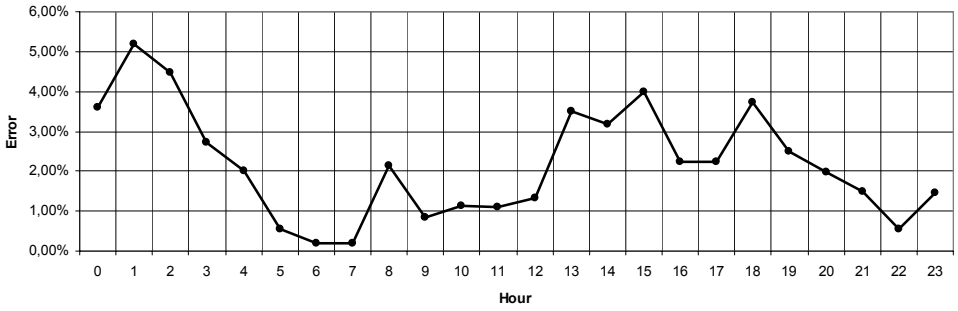

Figure 5.12. Percent errors for the worst Saturday.

As explained before, due to the structure of the model, some past data are used as inputs and these inputs might be from special days. In this case, learning would be deviated from the regular trend. So, the days having such inputs should not be put into the training set, instead they should be considered separately. Replacing special days with their regular forecasts and then using them as inputs is one way to solve this problem; but it does not give satisfactory results for the previous models, so there is no need to repeat it here. Another way is to apply the previous Saturday data from the same cluster, corrected by the ratio of two past Friday loads, as in the Early Monday model which has satisfactory results. Table 5.10 shows the results of this last alternative to such Saturdays.

Results in Table 5.10 are quite successful. By giving this solution to such Saturdays, the model for Saturday cluster becomes complete, covering all the relevant data.

Table 5.10. Results for Saturdays whose inputs coincide special days.

Day	MAPE (Using the ratio of past two Fridays)
25 Mar 2000	2.69
01 Apr 2000	0.92
17 Mar 2001	1.27
24 Mar 2001	1.94
26 May 2001	1.48
02 Jun 2001	1.38
02 Mar 2002	1.27
09 Mar 2002	1.32
14 Dec 2002	1.85
21 Dec 2002	2.10
AVERAGE	**1.62**

5.4. Sunday Model

As in the case of Saturday, Sunday data is mostly correlated with itself; therefore the model is constructed accordingly: a neural network to forecast only Sunday load, using past Sunday loads. Again, in order to emphasize weekly difference, past two Saturday loads and temperatures are also given as inputs. Expressions for input and output variables can be written as:

Output: $L_{sun}(w,h)$

Inputs: h_C, s,

$L_{Sun}(w-1,h)$, $L_{Sun}(w-2,h)$, $L_{Sat}(w,h)-L_{Sat}(w-1,h)$,

$T_{Sun}(w-1,h)$, $T_{Sun}(w-2,h)$, $T_{Sat}(w,h)-T_{Sat}(w-1,h)$

where all the variables are as defined earlier. Yearly averaged errors for the experiments performed for this cluster are given in Table 5.11.

Table 5.11. Yearly averaged percent errors for Sunday model.

	1999	2000	2001	2002
MAPE	19.93	1.85	1.77	1.94

As the error figures in Table 5.11 state, the neural network forecasts by the hybrid learning are very close to the real load values. This fact is visualized in the figures below, from 5.13 to 5.16. The former two figures belong to 12 May 2002, which has the lowest error (0.82%) and the latter two are of 31 March 2002, the worst Sunday in 2002 with 7.66% error.

For the Sundays having inputs from special days, the solution proposed for the Saturday model is tested and daily averaged results are presented in Table 5.12.

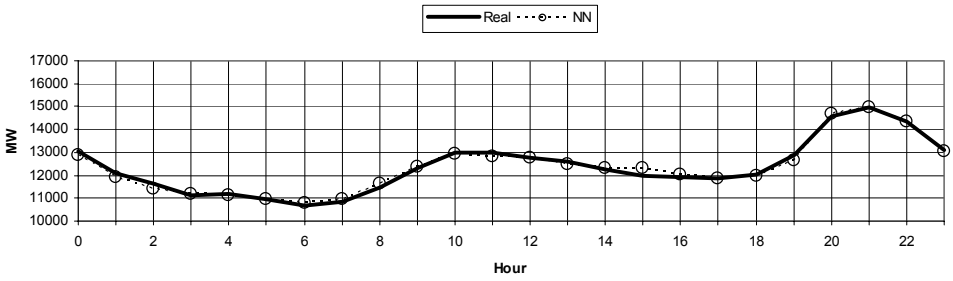

Figure 5.13. Actual and forecast values for the best Sunday, 12 May 2002.

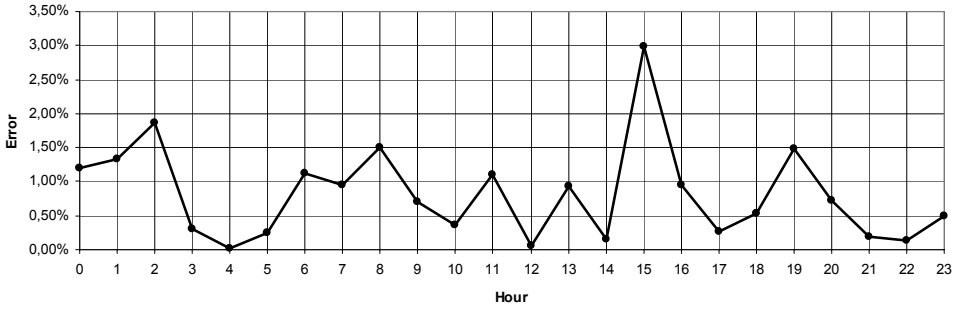

Figure 5.14. Percent errors for the best Sunday.

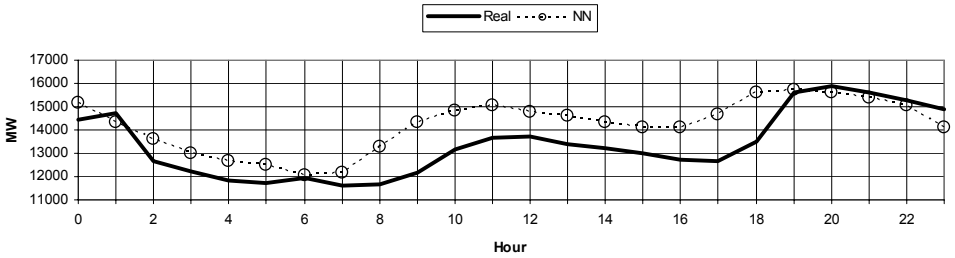

Figure 5.15. Actual and forecast values for the worst Sunday, 31 March 2002.

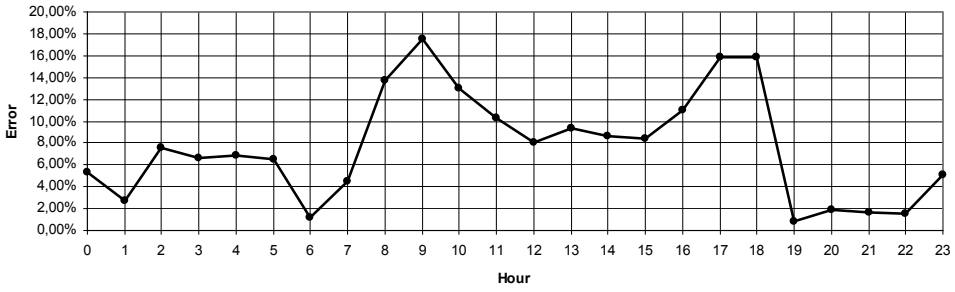

Figure 5.16. Percent errors for the worst Sunday.

Table 5.12. Results for Sundays whose inputs coincide special days.

Day	MAPE (Using the ratio of past two Saturdays)
26 Mar 2000	4.08
02 Apr 2000	3.38
05 Nov 2000	3.31
12 Nov 2000	2.77
18 Mar 2001	1.49
25 Mar 2001	4.03
20 May 2001	2.33
27 May 2001	2.87
03 Mar 2002	4.17
10 Mar 2002	2.98
15 Dec 2002	1.36
22 Dec 2002	1.47
AVERAGE	**2.85**

These error figures in Table 5.12 are not as good as Saturday model, but still not very high.

5.5. Overall Results

The STLF method proposed in this thesis clusters the data into four pieces and also handles special days separately. Because of these special days, some regular data in the clusters cannot be the outputs, but used only as inputs. If they were forecast by neural networks, they would need inputs from special days, which would misdirect the learning. Therefore, their forecasts are achieved by another method, as to be described.

In the previous sections, neural network results for regular data and the forecasts for such days were given separately; but here, they are presented together in order to give an idea about the overall yearly results, except the forecasts of special days, of course. Table 5.13 summarizes the percent errors according to the clusters.

Table 5.13. Summary of the forecast errors by the proposed model.

Cluster	Years		
	2000	2001	2002
Early Monday	2.29	1.89	1.97
Weekday	1.52	1.45	1.39
Saturday	2.01	1.68	1.47
Sunday	2.02	1.85	1.99
WEIGHTED AVERAGE	**1.69**	**1.56**	**1.51**

Table 5.13, especially the last row, emphasizes the general performance of the proposed method. It gives reasonable forecasts, with Weekday cluster being the most successful one. It is generally difficult to have a good estimate for Sunday data. Indeed, the error figures here are higher than the other clusters, but still in the acceptable ranges. Similarly, Monday morning shows unique characteristics, hard to capture. But the model performs well also for it. Saturday cluster has lower errors like Weekday set. These experiments are repeated for three yearly data pairs as seen in Table 5.13 and all results are consistently close to each other. This proves the generality of the proposed model and applicability to the data of new coming years.

CHAPTER 6

HANDLING THE SPECIAL DAYS

6.1. Holidays in Turkey

It is a known fact that electric consumption decreases on holidays and shows different trend than the regular days. Therefore, they should be analyzed separately. In Turkey, there are two kinds of holidays, national and religious. National holidays are fixed in time, but religious holidays are moving each year. Table 6.1 shows these holidays in years 1999 to 2002.

Some notes should be added to Table 6.1. For the religious holidays, there is a half-day holiday for preparation which is not shown in Table 6.1. For the Republic Day, 28 October is half working day for public offices. Furthermore, if a holiday covers three weekdays, the rest of the week could be announced as holiday by the government; or if a single day holiday comes across a Tuesday or Thursday, adjacent Monday or Friday could be declared as holiday, as well. Moreover, after the long holidays, Mondays, as the first working day, show different characteristics; hence they should be in the special day cluster. For these reasons, Table 6.1 is extended and the final form is given in Appendix. Here, Table 6.2 is taken to show the distribution of the existing data.

Table 6.1. National and religious holidays in years 1999 to 2002.

	Name	Date	1999	2000	2001	2002
National	National Sovereignty and World Children's Day	23 Apr	Fri	Sun	Mon	Tue
	Ataturk Remembrance - Youth and Sports Day	19 May	Wed	Fri	Sat	Sun
	Victory Day	30 Aug	Mon	Wed	Thu	Fri
	Republic Day	29 Oct	Fri	Sun	Mon	Tue
Religious	Ramadan Feast	Moves	19-21 Jan Tue-Thu	8-10 Jan Sat-Mon 27-29 Dec Wed-Fri	16-18 Dec Sun-Tue	5-7 Dec Thu-Sat
	Sacrifice Feast	Moves	28-31 Mar Sun-Wed	16-19 Mar Thu-Sun	5-8 Mar Mon-Thu	22-25 Feb Fri-Mon

Table 6.2. Number of regular vs. special days in the available data.

	1999		2000		2001		2002	
	N	%	n	%	n	%	n	%
Regular Weekday	247	67.67	224	66.87	237	68.50	244	68.35
Special Weekday	14	3.84	15	4.48	11	3.18	11	3.08
Regular Weekend	98	26.85	84	25.07	91	26.30	97	27.17
Special Weekend	6	1.64	12	3.58	7	2.02	5	1.40
TOTAL	365	100	335	100	346	100	357	100

To see the difference between holidays and regular days, the following figures (6.3 – 6.13) are drawn.

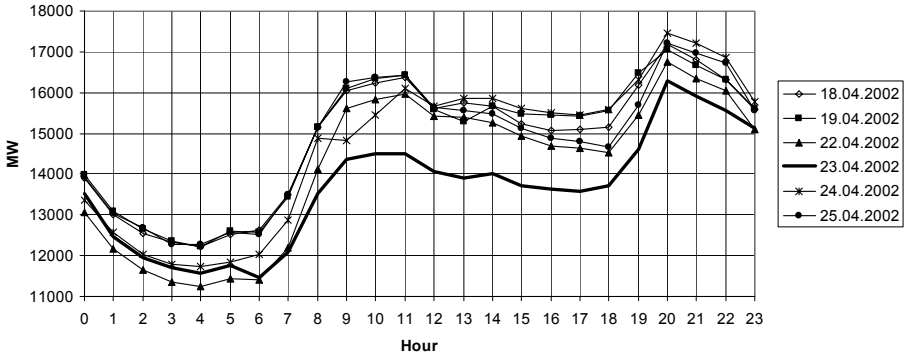

Figure 6.3. Load differences between 23 Apr 2002 Tue and neighboring days.

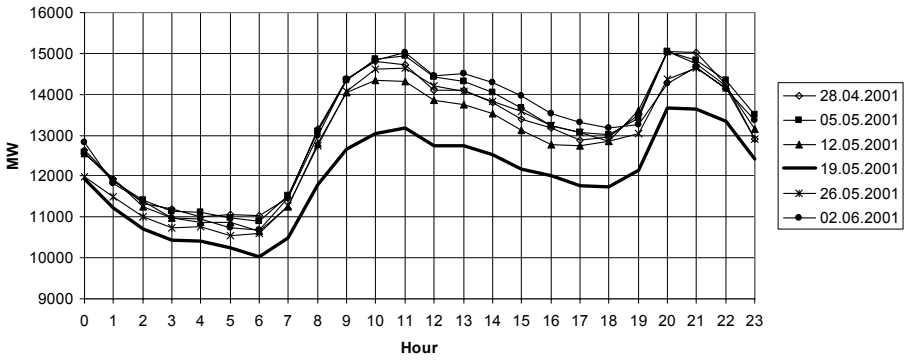

Figure 6.4. Load differences between 19 May 2001 Sat and neighboring Saturdays.

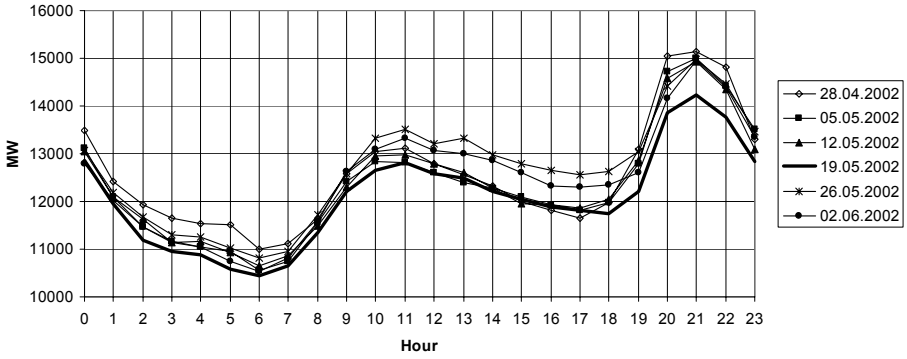

Figure 6.5. Load differences between 19 May 2002 Sun and neighboring Sundays.

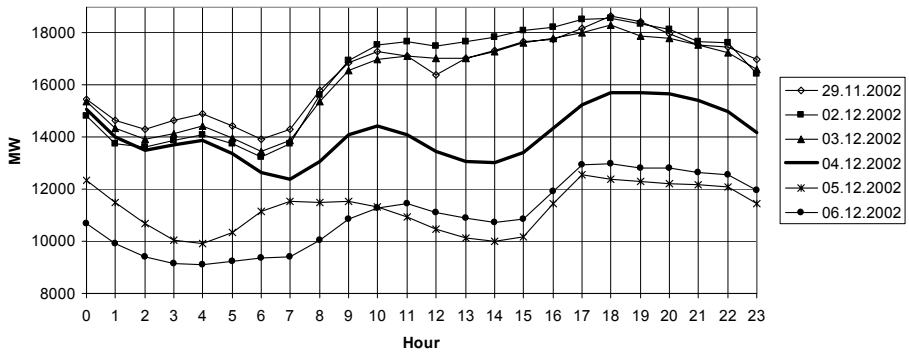

Figure 6.6. Load differences between 4 Dec 2002 Wed and neighboring days.

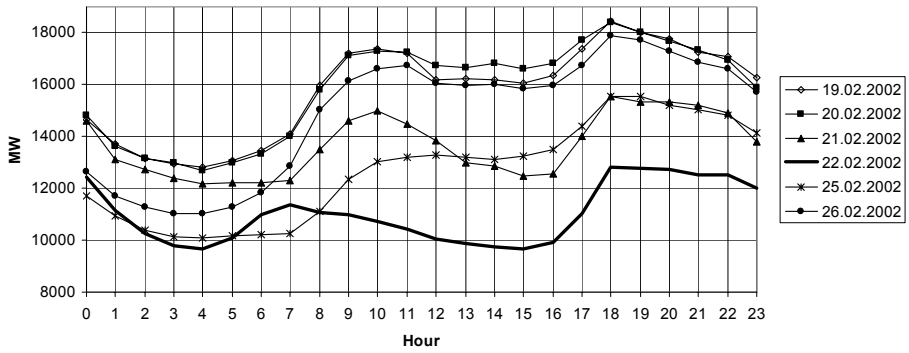

Figure 6.7. Load differences between 22 Feb 2002 Fri and neighboring days.

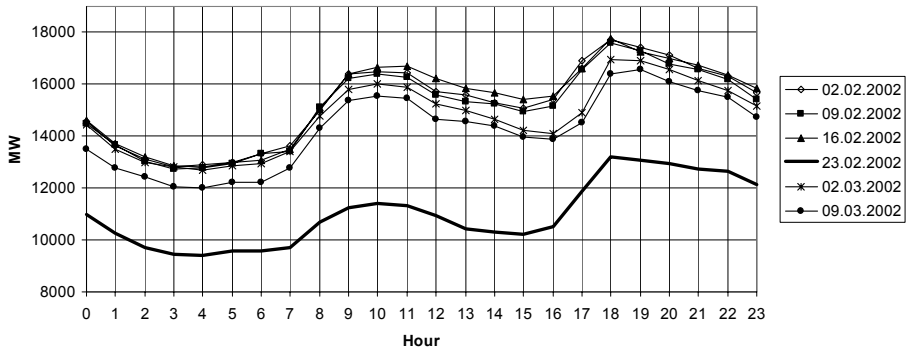

Figure 6.8. Load differences between 23 Feb 2002 Sat and neighboring Saturdays.

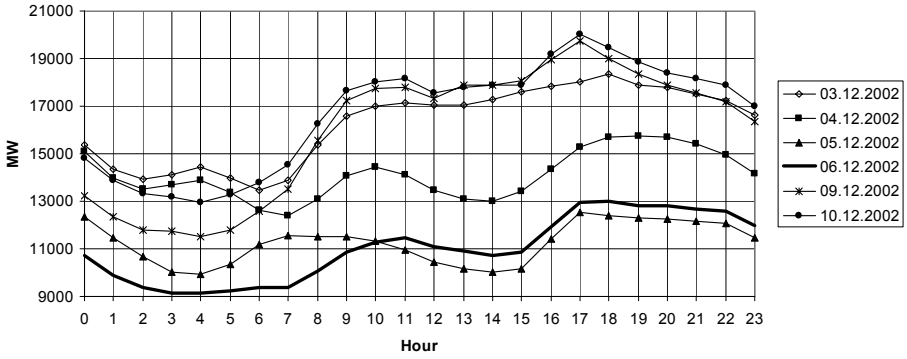

Figure 6.9. Load differences between 6 Dec 2002 Fri and neighboring days.

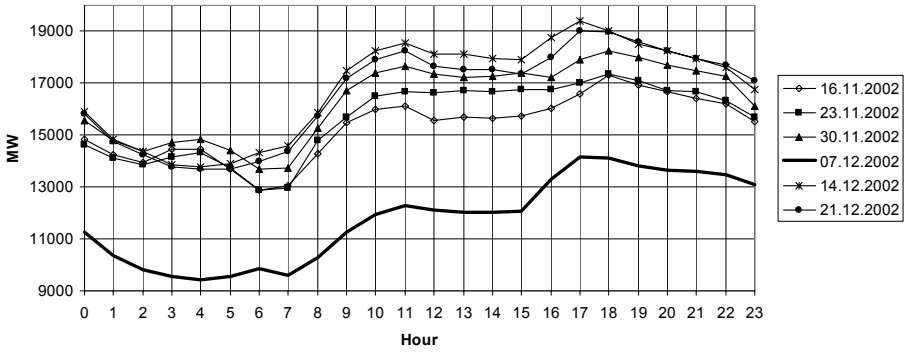

Figure 6.10. Load differences between 7 Dec 2002 Sat and neighboring Saturdays.

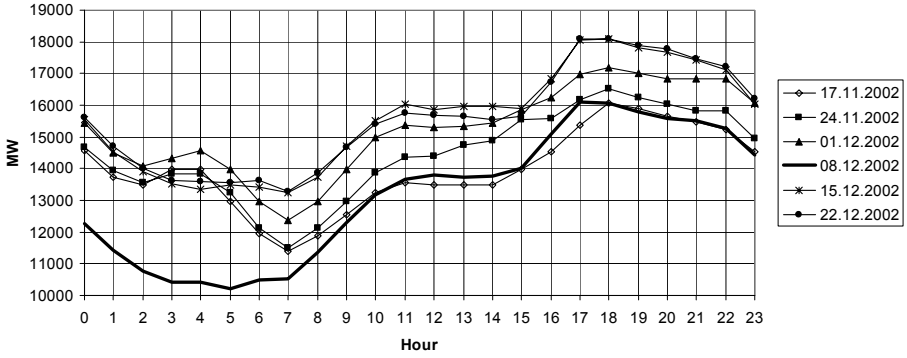

Figure 6.11. Load differences between 8 Dec 2002 Sun and neighboring Sundays.

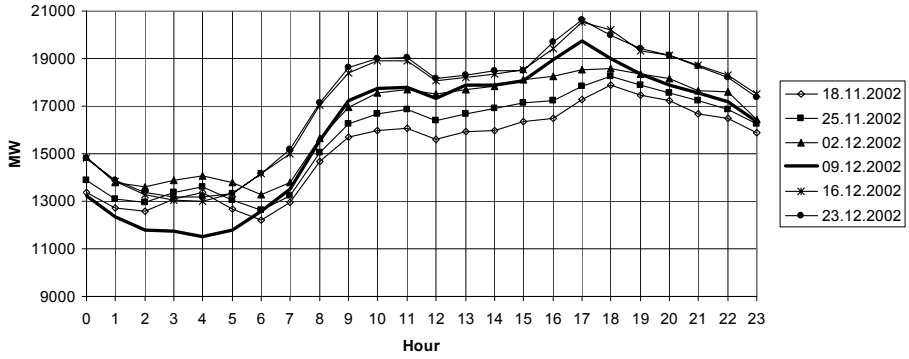

Figure 6.12. Load differences between 9 Dec 2002 Mon and neighboring Mondays.

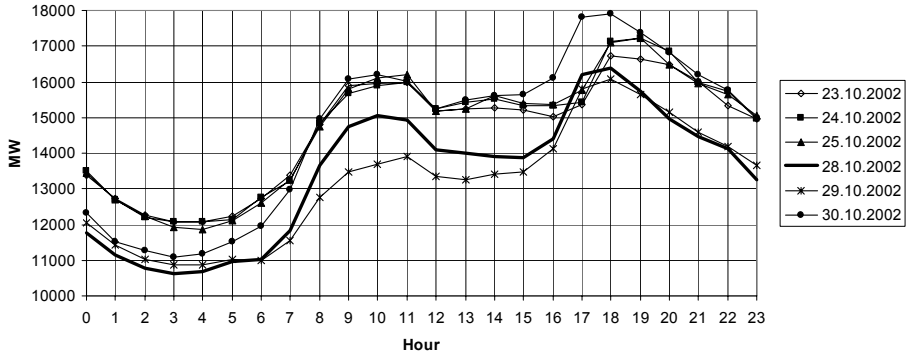

Figure 6.13. Load differences between 28 Oct 2002 Mon and neighboring days.

It is understood from the above figures that the holidays show several different behaviors:

- If it is a single-day holiday and if it is a weekday (Figure 6.3), then the first six-eight hours are as if regular hours; there is an evident decrease in working hours; evening load is still below the regular load but it seems trying to catch it.
- If it is a single-day holiday and if it is a Saturday (Figure 6.4), then the load is under the average for all hours.

- If it is a single-day holiday and if it is a Sunday (Figure 6.5), then there is no difference than an ordinary Sunday.
- If it is the preparation day for a religious holiday (Figure 6.6), it has regular loads till the morning but then, it decreases. The following feast days have lesser loads.
- If it is the first day of a religious holiday (Figure 6.7), then the consumption is less than the previous preparation day; and, quite below the other days in the same week.
- If it is midday of a religious holiday, regardless of the day type (Figures 6.8-6.10), the load is below the average for all hours.
- If it is the last day of a religious holiday (Figure 6.11), then the first hours have small loads, the rest is like a normal day.
- If it is a Monday as the first working day after a long holiday (Figure 6.12), then the first quarter of the day has lower electric loads.
- If it is a 28 October that coincides a weekday (Figure 6.13), then it shows different characteristics than the other weekdays.

These points should be taken into consideration when forming the neural network model and handling the special day data.

6.2. Proposed Model for Special Days

In the previous section, a comprehensive analysis on special days was given. Since determining the special days and extracting the differences among themselves are of important parts of the forecasting process, they are examined carefully.

The neural networks designed for regular load forecasting cannot be directly used for special day load forecasting; because, holiday loads are lower than the regular loads. Therefore, large errors are observed [4].

Actually, the regular neural networks can be employed if their outputs are adjusted to remove the gap between holiday and regular data. To remove this gap,

holiday data from previous years are observed and a correction term is calculated. This correction term is then used to subtract an amount from the neural network output, found as if it were a normal day load. It can be shown mathematically as:

$$L_{Special}(d,h) = y_{NN}(d,h) - C(d,h)y_{NN}(d,h)$$
$$= \big(1 - C(d,h)\big)y_{NN}(d,h)$$

where $L_{Special}(d,h)$ is the special day load to be forecast for day d and hour h; $y_{NN}(d,h)$ is the neural network output which is the regular day forecast for the same day and hour; and $C(d,h)$ is the correction term in percentage, changing according to day and hour, introduced for holiday adjustment.

The regular forecast component $y_{NN}(d,h)$ is obtained via the neural network which is trained for the same day type with the special day under consideration, but without including the special days in training. Therefore, it is expertized in forecasting the normal loads and output becomes larger than the load of the holiday. That is why a correction term is needed.

The correction term in the equation above is the average of the percent deviations of the regular neural network forecasts from the actual loads of the special days in previous years. This can be expressed as:

$$C(d,h) = \frac{1}{n}\sum_{i=1}^{n}\frac{y_{NN}(i,h) - L(i,h)}{y_{NN}(i,h)}$$

where $y_{NN}(i,h)$ is the regular neural network output for the i^{th} special day from the previous years; $L(i,h)$ is the actual load, and n is the number of the special days. A similar approach was tried in the work of Bakirtzis et al. [4], but they have chosen to use an absolute value in MWs as the correction term, not the

percent of the forecast value. They have obtained that correction term from the absolute differences of the previous years' predictions. This approach is not followed here but modified as taking the percent variations; thinking that percentage is more informative than absolute values since yearly load consumptions do not remain the same.

It should also be noticed that, the electric load differs in holiday type. Therefore, special days should be clustered among themselves and the same type of data should be used in calculating the correction term. Taking into consideration the analysis given above, the following clustering can be done for the holidays in Turkey:

a. Single-day special day, coincides to weekday,
b. Single-day special day, coincides to weekend,
c. Preparation day to a religious holiday,
d. The first day of a religious holiday,
e. The midday of a religious holiday,
f. The last day of a religious holiday,
g. Mondays after religious holidays,
h. 28 October.

These days have their unique load characteristics and should be considered separately. One exception can be done to single-day holidays that coincide to Sundays. They are not so much different than the regular Sunday data; therefore, there is no need to form a cluster for this kind of data; instead, they can be put into the Sunday training set.

Another point is that, this method depends on the loads of previous years; hence, it should use as many years as possible in order to give reliable and stable results. This means that, only the special days of year 2002 can be forecast with the available data.

6.3. Experiments on Special Days

Special day load forecasting is the most difficult part of this work. Since the behavior of such days does not follow the regular trend of normal days, they should be handled discretely. A method is proposed for this purpose, as described in the previous section.

According to the proposed method, in order to forecast a special day load, it is necessary to use past years' data. Therefore, special days from year 2002 undergo to the experiment. First, a special day from year 2002 is taken and the cluster containing that special day is found. As the holidays have different load consumptions in a year, they are clustered according to their load profiles, as given before. After that, each special day in the cluster is forecast by the corresponding neural network as if it were a regular day. Since the actual loads for the past special days are available, correction term is calculated and the regular forecast for the special day under consideration is adjusted.

Table 6.3 lists the special days of year 2002, states their clusters, gives the actual, base forecast and corrected forecast loads and shows two corresponding errors.

As seen from Table 6.3 that all the resultant errors are below 7% and half of them are below 5%. These numbers cannot generally be reached for special days. Actually, if they were predicted by a neural network only and no corrective action were taken then the forecasts would not be so successful, as shown in the Regular Error column in the table. On the average, correction term makes the percent error reduce from 19.86% to 4.10%, which proves its validity and necessity.

In Appendix, one can find the complete list of special days and see that three of them are not placed in Table 6.3: 21 February 2002 Thursday (preparation day to the religious holiday), 24 February 2002 Sunday (midday of the religious holiday) and 25 February 2002 Monday (last day of the religious holiday). They cannot be forecast by this method; because, there are no data available from the previous years in the same cluster with these days. For example, 7 January 2000 Friday is in the same cluster with 21 February 2002 Thursday but it is missing in the

database at hand. Therefore, there can be no correction term for 21 February. However, in case of having a complete database, such problems do not exist.

Table 6.3. Year 2002 special days and their corrected forecasts.

Special Day	Cluster	Actual Load (MW)	Regular Forecast (MW)	Corrected Forecast (MW)	Regular Error (%)	Corrected Error (%)
22 Feb 02 Fri	1st day of rel. hol.	11028	15868	10471	44.76	5.73
23 Feb 02 Sat	Midday of rel. hol.	12691	14950	11977	19.35	5.47
23 Apr 02 Tue	Single-day, Weekday	13646	14622	13596	7.17	1.32
19 May 02 Sun	Single-day, Weekend	12092	12277	12277	1.53	1.53
30 Aug 02 Fri	Single-day, Weekday	14605	15769	14670	7.86	1.13
28 Oct 02 Mon	28 October	13445	14176	12760	5.96	4.81
29 Oct 02 Tue	Single-day, Weekday	13184	14158	13495	10.34	4.08
04 Dec 02 Wed	Preparation Day	14106	16256	14051	15.38	2.05
05 Dec 02 Thu	1st day of rel. hol.	11262	16195	11633	44.32	3.80
06 Dec 02 Fri	Midday of rel. hol.	11007	16153	11634	47.65	5.85
07 Dec 02 Sat	Midday of rel. hol.	11776	15539	12191	33.47	5.68
08 Dec 02 Sun	Last day of rel. hol.	13179	14630	12674	12.60	5.15
09 Dec 02 Mon	Mon after rel. hol.	15959	17205	17022	7.81	6.66
AVERAGE		**12922**	**15215**	**12958**	**19.86**	**4.10**

In the Figures 6.14 – 6.17 given below, the best and the worst results for the above special days are presented. The effect of correction term is clearly seen in these figures.

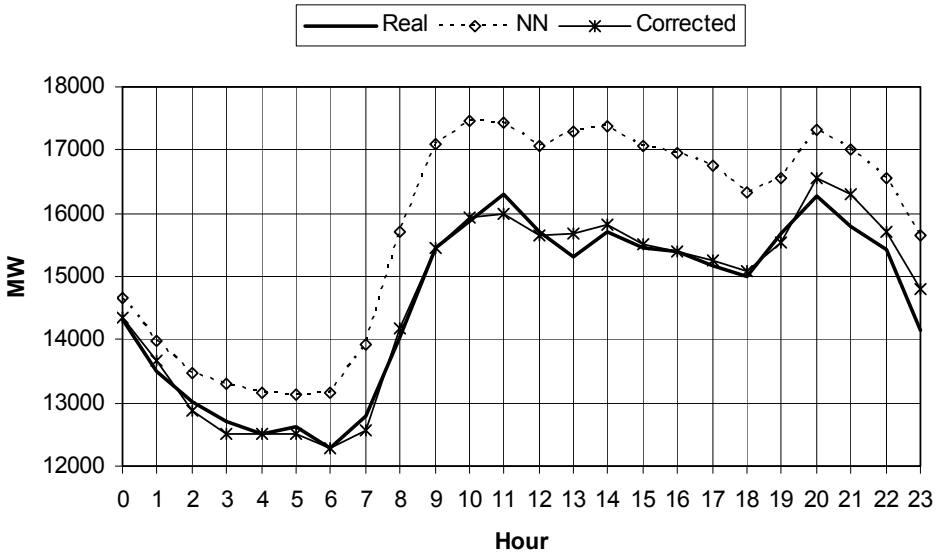

Figure 6.14. Actual, forecast and corrected values for the best special day, 30 Aug 2002, Fri.

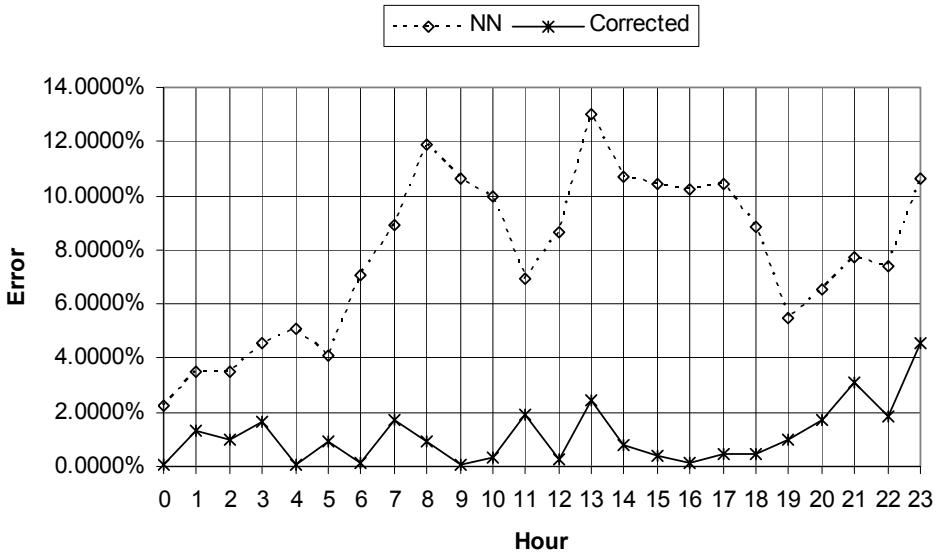

Figure 6.15. Percent forecast and corrected errors for the best special day.

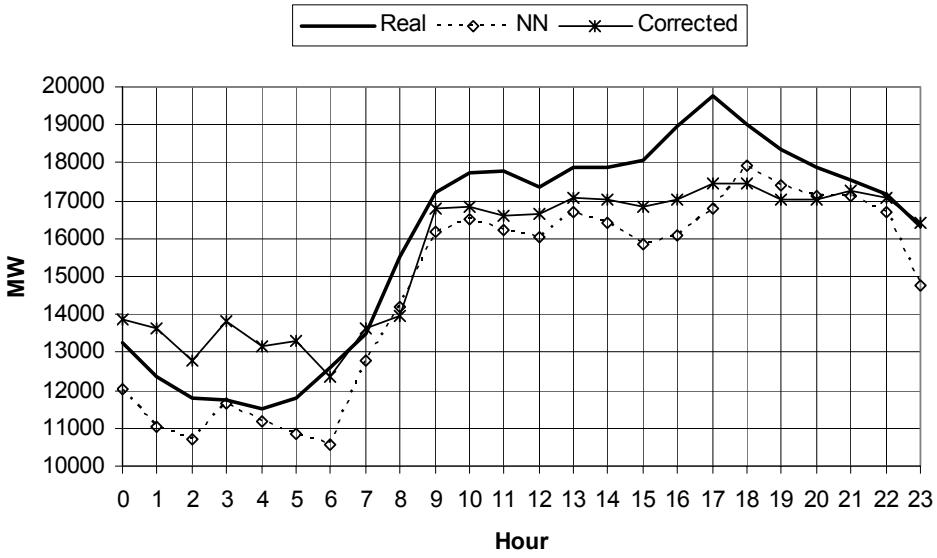

Figure 6.16. Actual, forecast and corrected values for the worst special day, 9 Dec 2002, Mon.

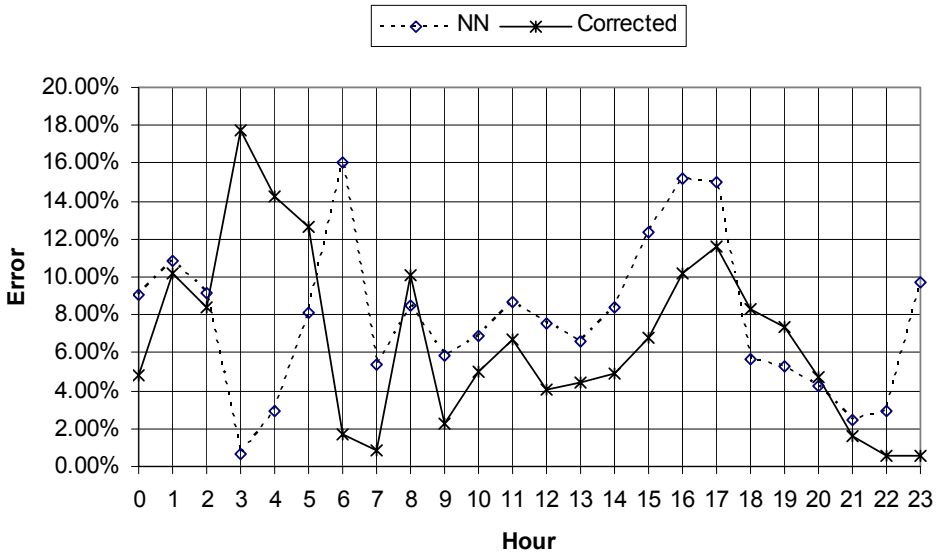

Figure 6.17. Percent forecast and corrected errors for the worst special day.

In literature, it is suggested to include the special days into the Sunday cluster, thinking that both have lower loads. This experiment is performed here in order to compare results with the proposed special day model.

A neural network is constructed and trained off-line with the data from year 2001, covering Sundays and special days. An additional input is introduced to discriminate Sundays from special days as having two different values: 0.9 for Sundays and 0.1 for special days. Then, year 2002 data are presented in on-line learning again with Sundays and special days. One can recall that, the average percent error for Sunday cluster in 2002 was 1.94%, without the special days. Now in this case, error is increased to 3.65% whose distribution is 2.85% for Sunday part and 6.89% for special day part. Therefore, it can be said that this combination causes Sunday forecasts be worsened and also it does not give satisfactory results for special days.

This means that, the proposed special day load forecasting model is better than putting the special days in Sunday cluster. The averaged percent error for them by the proposed model is 4.10%, which can be considered as a satisfactory result in the special day class.

As an overall error figure, it can be given that the average of real-time forecast errors in year 2002, including working days, weekends and special holidays is 1.60%, which should be considered as a successful result.

CHAPTER 7

OTHER MODELS FOR COMPARISON

As stated before, the proposed intelligent method consists of both off-line and real-time learning. The idea is to start real-time forecasting with a prepared model and to tune it further for changing conditions. In addition, step size is made adaptive, temperature measurements are used and data is clustered in a daily basis manner.

In order to comment about the results and see the performance of this proposed method, it should be compared with some other models, designed for the other alternatives. For this purpose, the following models are constructed.

7.1. No Off-line Learning

Weights are started from random initial conditions for real-time forecasts, so it does not make use of off-line learning. By picking input/output samples in time order, weights are adjusted. For example, the real-time forecasts for year 1999 are done in this way since there are no year 1998 data available. Weekday model for this year gives 10.66% error, which is an unacceptable value and shows the impossibility of starting a real-time application with a non-trained neural network hence proves the necessity of hybrid learning.

7.2. No Real-time Learning

In this model, off-line learning is performed but training is stopped at this point. In other words, model does not contain any real-time learning and the application

is run with constant weights, i.e., weights are no more updated after off-line learning has been finished. So, this model is designed to see the importance of weight adaptation in real-time.

Weekday model, which gave 1.30% error with year 2002 data by the proposed hybrid learning, presents 1.61% error when there is no real-time learning. It is obvious from this increase in the error that to continue learning in real-time improves the performance of the model. Real-time learning behaves like a fine tuning and if it is not applied, the neural network cannot adapt itself to changing conditions and it becomes difficult for it to give reasonable forecasts.

7.2. No Step Size Adaptation

Here, step size is taken as 0.5 and not updated during the learning. The MAPE for the weekday model with year 2002 data is increased from 1.30% to 1.47%. It is seen that making step size adaptive improves the model. Moreover, it is not a computationally complex algorithm and it has negligible effect on output calculations. Therefore, it is convenient to update the step size rather than fix it.

7.3. No Temperature Input

Temperature is one of the most important parameters in STLF. It is known that temperature variations directly affect the load consumption. In order to see this numerically, the weekday model is retrained without the temperature inputs. The resultant error, which rises to 2.05% (from 1.30%), shows this fact clearly.

7.4. No Data Clustering

In order to see what would happen if there were no clustering, a dummy model is configured. All data types from year 2001, except the special days, are given to a single neural network and it is trained off-line. Then all data from year 2002 (again no special days) is put through the model and it gives 2.93% error. From this result, one can understand that to cover all data types with a single neural network is not possible. Therefore, it can be said that a powerful clustering is necessary for this kind of forecasting problems.

7.5. Hourly Based Clustering

In this case, a separate neural network for each hour of the day is taken. Therefore, training sets are formed as having only one hour-type of the electric load. Two experiments are performed with this hourly based clustering: Weekends are separately handled; and considered with weekdays.

7.5.1. Weekends – Separately Handled

For working days there are 24 neural networks, representing the hours; and for weekends there are two neural networks, representing Saturdays and Sundays. Respective MAPEs for year 2002 are 1.73%, 1.75%, and 2.06%. Although these error figures are not high, the main drawback here is having 26 different models, all requiring separate training times. The proposed daily based clustering, which has four models, is more practical in this respect.

7.5.2. Weekends – Considered with Weekdays

In this model, there is no distinct neural network for weekends, 24 neural networks cover weekdays and weekends together. MAPEs when the special days of year 2002 are included and discarded are 2.45% and 2.10%, respectively. It is understood that considering weekends separately and discarding the special days are better ways to take.

7.6. Traditional Stochastic Time Series Approach

Traditional STLF models, such as regression or stochastic time series are widely used in electric generation units as they have proven their validity especially for weekday forecasts.

Any new method having a different approach than these conventional ones should give better results in order to be accepted. Therefore, the model proposed in this thesis should also be compared with a classical model that does the same task.

An introduction on the classical approaches, experiments performed and results obtained are given in this section.

Stochastic time series method appears to be the most popular approach that has been used and is still being applied to STLF in the electric power industry [47]. There are many names encountered in the literature for this approach, for example ARMA (autoregressive – moving average) models, ARIMA (integrated autoregressive – moving average) models, Box-Jenkins method, linear time series models, etc.

As a brief review, the load series, $y(t)$, is modeled as the output from a linear filter that has a random series input, $a(t)$, usually called a white noise. This random input has a zero mean and unknown fixed variance, $\sigma_a^2(t)$. Depending on the characteristics of the linear filter, different models can be classified as given in the following sections.

7.6.1. The Autoregressive (AR) Process

In the autoregressive process, the current value of the time series $y(t)$ is expressed linearly in terms of its previous values and a random noise $a(t)$. For an autoregressive process of order p, i.e., AR(p), this model can be written as:

$$y(t) = \phi_1 y(t-1) + \phi_2 y(t-2) + \ldots + \phi_p y(t-p) + a(t)$$

By introducing the backshift operator B that defines $y(t-1) = By(t)$, and consequently $y(t-m) = B^m y(t)$, the above equation can be written in the form:

$$\phi(B) y(t) = a(t)$$

where

$$\phi(B) = 1 - \phi_1 B - \phi_2 B^2 - \ldots - \phi_p B^p .$$

7.6.2. The Moving-Average (MA) Process

In the moving-average process, the current value of the time series $y(t)$ is expressed linearly in terms of current and previous values of a white noise series. For a moving average of order q, i.e., MA(q), this model can be written as

$$y(t) = a(t) - \theta_1 a(t-1) - \theta_2 a(t-2) - \ldots - \theta_q a(t-q)$$

A similar application of the backshift operator on the white noise series would allow the above equation to be written as:

$$y(t) = \theta(B) a(t)$$

where

$$\theta(B) = 1 - \theta_1 B - \theta_2 B^2 - \ldots - \theta_q B^q$$

7.6.3. The Autoregressive Moving-Average (ARMA) Process

In the autoregressive moving-average process, the current value of the time series $y(t)$ is expressed linearly in terms of its values at previous periods and in terms of current and previous values of a white noise. For an autoregressive moving-average process of order p and q, i.e., ARMA(p,q), the model is written as

$$y(t) = \phi_1 y(t-1) + \ldots + \phi_p y(t-p) + a(t) - \theta_1 a(t-1) - \ldots - \theta_q a(t-q)$$

By using the backshift operator defined earlier, the above equation can be written in the following form:

$$\phi(B)y(t) = \theta(B)a(t)$$

where $\phi(B)$ and $\theta(B)$ have been defined as above.

7.6.4. Experiments and Results

A stochastic time series model is constructed for STLF in order to be compared with the intelligent model, based on recurrent neural networks whose experimental results were presented previously.

Here, data of years 2001 and 2002 undergo the tests and ARMA is taken as the model. Parameters, p and q that cause the lowest error in year 2001 are found and corresponding ϕ and θ values are applied to year 2002 for estimating the 24-hour ahead load.

Regular data are again clustered into four sets as before and tests are repeated for each of them. Special days are grouped among themselves. Results are shown in Table 7.1, together with the errors obtained by the proposed recurrent neural network model.

Table 7.1. ARMA and recurrent neural network results for STLF in year 2002.

Cluster	ARMA Error (%)	RNN Error (%)
Early Monday	3.50	1.97
Remaining Weekdays	1.53	1.39
Saturday	2.72	1.47
Sunday	3.45	1.99
Special Days	10.34	4.10
WEIGHTED AVERAGE	**2.33**	**1.60**

It is obvious and easy to comment about Table 7.1. Classical method gives best forecasts for weekdays but not as successful as the recurrent neural network model. For weekends and Monday morning, it is quite worse and for special days, it is almost useless. These results show that, this traditional method – depending only on a time series and not making use of other parameters, such as temperature, hour of the day or day of the week, etc. – is weaker than the proposed adaptive, intelligent neural network based method.

CHAPTER 8

PRINCIPAL COMPONENT ANALYSIS FOR FEATURE EXTRACTION

In this work, the hourly data of Turkey's power consumption for several past years are available and it is desired to forecast a near future demand using neural networks. The factors affecting usage of the power are neither precisely known nor can be mathematically expressed. Furthermore, feeding neural networks with many parameters would lead unreasonable training times and redundancy. In this circumstances, using the principal component analysis (PCA) to eliminate the redundant parameters and thus speed up the learning would be worth considering. Because, it is known that, many algorithms are exponential in the dimensionality of the input, thus even reduction by a single dimension may provide valuable computational savings [16].

PCA is a linear procedure to find the direction in input space where most of the energy of the input lies. The projections of these components correspond to the eigenvalues of the input autocorrelation matrix. In other words, the principal components are obtained by projecting the data onto the orthogonal space spanned by a subset of the eigenvectors of the estimated autocorrelation matrix associated with the largest eigenvalues [62].

In the literature it is encountered with several applications at which neural networks and the PCA are considered together to extract the principal components of a data using a neural network.[5, 13, 40]. Using PCA to prepare the appropriate

inputs for the neural networks has not attracted much attention so far and this work can be considered as an original approach in this respect.

8.1. Wiener's Optimal Solution

Assuming a neural network configuration as shown in Figure 8.1 below

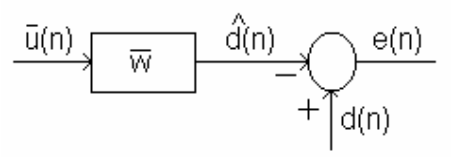

Figure 8.1. A linear neural network configuration.

where $\bar{u}(n)$ is the input vector, $d(n)$ is the desired output, $\hat{d}(n)$ is the neural network output and $\bar{w}$ is the weight vector, one can write

$$\hat{d}(n) = \bar{w}^{H} \cdot \bar{u}(n)$$

With the estimation error

$$e(n) = d(n) - \hat{d}(n)$$

the cost function to be minimized is

$$J(\bar{w}, n) = E\left\{|e(n)|^2\right\}$$

$$J(\bar{w},n) = E\{e(n)\cdot e^*(n)\} = E\{(d(n) - \bar{w}^H \cdot \bar{u}(n))\cdot (d^*(n) - \bar{u}^H(n)\cdot \bar{w})\}$$

$$J(\bar{w},n) = E\{|d(n)|^2\} - E\{d(n)\cdot \bar{u}^H(n)\}\cdot \bar{w} - \bar{w}^H \cdot E\{d^*(n)\cdot \bar{u}(n)\} + \bar{w}^H \cdot R_u \cdot \bar{w}$$

where $R_u = E\{\bar{u}(n)\cdot \bar{u}^H(n)\}$ is the autocorrelation matrix.

Since $J(\bar{w},n)$ is quadratic, it has a global minimum which can be found by equating the derivative of J with respect to $\bar{w}$ to zero.

If

$$\bar{p}(n) \stackrel{\Delta}{=} E\{\bar{u}(n)\cdot d^*(n)\}$$

is defined as the cross-correlation vector between the input and the desired response then,

$$J(\bar{w},n) = \sigma_d^2 - \bar{p}^H(n)\cdot \bar{w} - \bar{w}^H \cdot p(n) + \bar{w}^H \cdot R_u \cdot \bar{w}$$

where σ_d^2 is the variation of d(n).

It can be assumed that the global minimum of J is achieved when $\bar{w} = \bar{w}_o$. Then,

$$\frac{d}{d\bar{w}} J(\bar{w},n) = -2\bar{p}(n) + 2R_u \cdot \bar{w}\Big|_{\bar{w}=\bar{w}_o} = 0$$

The solution

$$R_u \cdot \overline{w}_o = \overline{p}(n)$$

is called the Wiener-Hopf equations.

So the optimal weights can be found by

$$\overline{w}_o = R_u^{-1} \cdot \overline{p}(n)$$

8.2. Convergence of the Steepest Descent Algorithm

Assuming a neural network configuration whose weights are adjusted and error is backpropagated using steepest descent method, weight update formula can be written as

$$\overline{w}(n+1) = \overline{w}(n) - \frac{1}{2}\eta\left(\overline{\nabla} J(n)\right)$$

where η is called as the learning rate.

Remembering from the previous section that

$$\overline{\nabla} J(n) = -2\overline{p}(n) + 2R_u \cdot \overline{w}(n)$$

then

$$\overline{w}(n+1) = \overline{w}(n) + \eta\left(\overline{p}(n) - R_u \cdot \overline{w}(n)\right)$$

Since

$$\overline{p}(n) = R_u \cdot \overline{w}_o$$

$$\overline{w}(n+1) = \overline{w}(n) + \eta \cdot R_u \cdot \left(\overline{w}_o - \overline{w}(n)\right) = \left(1 - \eta \cdot R_u\right) \cdot \overline{w}(n) + \eta \cdot R_u \cdot \overline{w}_o$$

Defining the deviation vector $\overline{c}(n)$ as

$$\overline{c}(n) \stackrel{\Delta}{=} \overline{w}(n) - \overline{w}_o$$

Then,

$$\overline{c}(n+1) = \overline{w}(n+1) - \overline{w}_o = \left(1 - \eta \cdot R_u\right) \cdot \overline{w}(n) - \left(1 - \eta \cdot R_u\right) \cdot \overline{w}_o$$

$$\overline{c}(n+1) = \left(1 - \eta \cdot R_u\right) \cdot \overline{c}(n)$$

Using the diagonalizability property of R_u, one can write

$$R_u = Q \Lambda_u Q^H$$

where Λ_u is the diagonal matrix consisting of eigenvalues, λ_k, of R_u and Q is the unitary modal matrix.

So,

$$\overline{c}(n+1) = \left(1 - \eta \cdot Q \Lambda_u Q^H\right) \cdot \overline{c}(n)$$

Multiplying both sides from left by Q^H yields

$$Q^H \cdot \bar{c}(n+1) = \left(Q^H - \eta \cdot \underbrace{Q^H Q}_{\mathrm{I}} \Lambda_u Q^H \right) \cdot \bar{c}(n) = \left(\mathrm{I} - \eta \cdot \Lambda_u\right) \cdot Q^H \bar{c}(n)$$

Defining $\bar{v}(n) \overset{\Delta}{=} Q^H \bar{c}(n)$ converts above equation to

$$\bar{v}(n+1) = \left(1 - \eta \cdot \Lambda_u\right) \cdot \bar{v}(n)$$

or

$$v_k(n+1) = \left(1 - \eta \cdot \lambda_k\right) \cdot v_k(n)$$

In terms of initial conditions

$$v_k(n) = \left(1 - \eta \cdot \lambda_k\right)^n \cdot v_k(0)$$

If η is chosen such that $\left|1 - \eta \cdot \lambda_k\right| < 1$ is achieved then v_k approaches to zero as n increases.

Hence v_k can be re-written exponentially as

$$v_k(n) = e^{-n/\tau_k} \cdot v_k(0)$$

Therefore,

$$e^{-1/\tau_k} = 1 - \eta \cdot \lambda_k$$

$$\tau_k = \frac{-1}{\ln|1 - \eta \cdot \lambda_k|}$$

Recalling that

$$\bar{v}(n) = Q^H \bar{c}(n) = Q^H \left(\bar{w}(n) - \bar{w}_o \right)$$

If v_k approaches to zero then $\bar{w}(n)$ approaches to $\bar{w}_o$ and convergence is accomplished.

In terms of neural network weights,

$$\bar{w}(n) = \bar{w}_o + Q \cdot \bar{v}(n) = \bar{w}_o + \sum_k \bar{q}_k v_k(n)$$

where $\bar{q}_k$'s are the orthonormal eigenvectors.

$$w_i(n) = w_{o_i} + \sum_{k=1}^{M} q_{k_i} \cdot e^{-n/\tau_k} \cdot v_k(0)$$

It is seen that weights contain *M* different time constants and the slowest time constant, i.e., the largest in magnitude determines the convergence time. Noticing that

$$\tau_k = \frac{-1}{\ln\left|1 - \eta \cdot \lambda_k\right|}$$

for a fixed learning rate, the slowest time constant is obtained because of the smallest eigenvalue.

8.3. Proposed PCA Method

The steepest descent algorithm states that, the slowest time constant of the weights determines the convergence time. This slowest time constant corresponds to the smallest eigenvalue of the autocorrelation matrix found for the input data.

If this smallest eigenvalue, and therefore the input data causing it, is discarded, time constant decreases and training process speeds up. This can be achieved utilizing principal component analysis. The PCA projects the input data from their original *n*-dimensional space onto the *m*-dimensional output space ($m < n$) performing a dimensionality reduction which retains most of the intrinsic information in the input data vectors; and the transformation matrix consists of eigenvectors which correspond to the largest eigenvalues.

Having an input vector of size *N*, the autocorrelation matrix is formed and eigenvalues are found. So, there are *N* eigenvalues and corresponding *N* eigenvectors which are orthonormalized.

If the transformation matrix Q is constructed by the eigenvectors, related to the largest *N*-1 eigenvalues, then the input space dimension is reduced by one, the smallest eigenvalue is discarded and convergence time will be decreased. Similarly, the largest *N*-2 eigenvalues can be used for transformation and the speed of the learning process is further improved. It should be noticed at this point that, the number of input layer nodes in the neural network model, which were *N* initially, reduces with these transformations and the network architecture is changed, affecting the weight dimensions.

But the question is how many eigenvalues to eliminate? Because, in each space reduction, the smallest eigenvalue is discarded; and at the same time, the

information content to be transferred is decreased, as well. So, after a time, although speed is improved, the learning ability of the network will be getting worse and it will be difficult to have outputs closer to the desired values.

Therefore, this dimension reduction algorithm should stop at an optimum point. That is, at the optimum point, the neural network should learn the system dynamics fast enough and should give successful results, with an acceptable, predefined error.

In order to determine the optimal neural network configuration, the following experiments have been performed.

8.4. Experiments with the PCA Method

Weekday model is taken into consideration to try and evaluate the PCA method, i.e., data from the weekday cluster are used to train the model. The output to be forecast and the initial input variables are as follows:

Output: $L(d,h)$

Inputs: h_C, d, s,

$$L(d-1,h),\ldots,L(d-1,h-5),T(d-1,h),\ldots,T(d-1,h-5),$$

$$L(d-2,h),\ldots,L(d-2,h-5),T(d-2,h),\ldots,T(d-2,h-5),$$

$$L(d-3,h),\ldots,L(d-3,h-5),T(d-3,h),\ldots,T(d-3,h-5)$$

The inputs related with time (hour, day and season) are not put through the transformation since they are for sure very effective on load consumption. But 18 past load values and 18 past temperatures may not be all crucial to forecast the output. At the same time, any of them cannot be directly discarded from the input vector since the least significant one is not known. Therefore, they are transformed into lesser dimensions by the proposed PCA method.

Reducing the dimension one by one down to having one load and one temperature variables yields 18 transformed input vectors and hence 18 neural networks, each

having different input layers. So the weight dimensions are changed, as well as the convergence cycles.

All these neural networks are trained upto 300,000 cycles and the percent erros in Figure 8.2 are obtained.

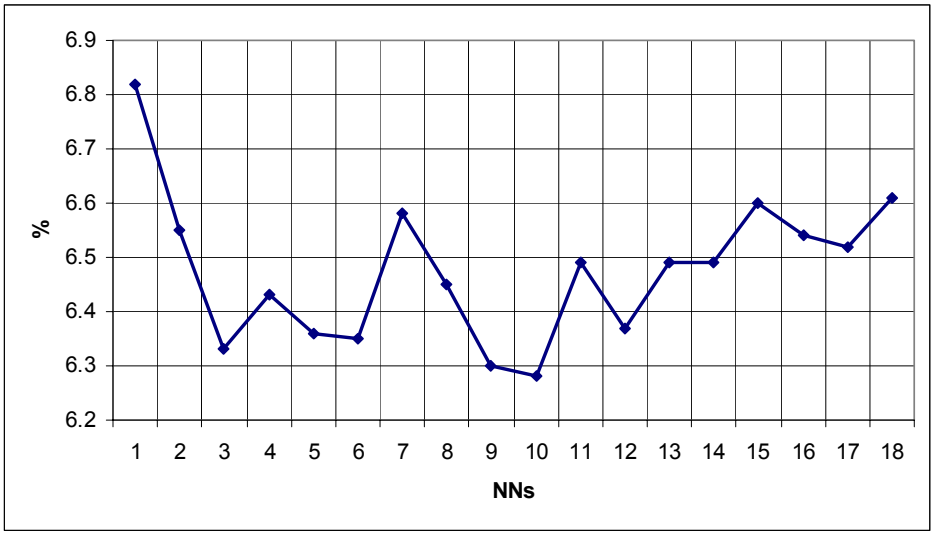

Figure 8.2. MAPEs of 18 different neural networks at 300,000 cycle.

The horizontal axis in Figure 8.2 represents the reduced size of the transformed load (or temperature) input vector. As a general trend, error is decreasing with the increasing amount of information (from one input case to ten input case) or with the increasing convergence speed (from 18 input case to ten input case). Therefore, in success and speed point of view, the one with ten inputs becomes the optimum neural network. This means that one can go on with this configuration knowing that the best input features have been selected.

To test this idea and finalize the model, a comparative experiment can be performed. Three models are taken into account: the neural network with the

optimum input topology, i.e., the one with ten transformed load and temperature inputs, the neural network with five transformed load and temperature inputs and the neural network with the untransformed inputs. These three models are trained until the output error reduces below 3.00% in order to get the convergence cycles. The results in Table 8.1 are obtained.

Table 8.1. Convergence comparison of three models.

Inputs	Convergence Cycles
10 (Optimum)	720,000
5	890,000
Untransformed	1,110,000

As stated before, the model with ten inputs is the optimum one. Although the neural network with five inputs has fewer variables, which speeds up the weight calculations, it is providing less information to the network; therefore, its learning capacity is worse than the optimum model, as seen in Table 8.1. The neural network with untransformed inputs gives full information, which is a valuable parameter in learning, but the large number of inputs affects the convergence negatively. Hence, it is slower than the optimum model.

Consequently, it can be said that this kind of preprocessing and feature extraction effort, carried on before the designing phase, prevents the neural network model from time losses in real-time operation.

CHAPTER 9

CONCLUSIONS

9.1. Discussions of the Results

As the highlights of this research, a hybrid learning for recurrent neural networks, which combines off-line and real-time trainings; data clustering considering the Turkey's load consumption profile; proposing solutions for all day types, including special days; extracting the most informative input features by PCA; adjusting step size iteratively while making real-time forecasts can be given.

Proposed hybrid learning prepares the model for real-time load forecasting by training it first with the available off-line data and getting the weights ready. During the real-time application, weights are undergone to a fine tuning operation in order to track the changing conditions. By merging these two phases, the neural network model gains experience from the past data; therefore, results become better than the standard learning methods.

Clustering is performed after a detailed data analysis, based on correlation measures, daily and seasonal variations, holiday behaviors, etc. Then separate neural network models are constructed for each cluster. Forecast errors with the proposed models come out to be smaller than a single neural network case where the training data is not clustered.

Special day forecast, which is the most difficult part of the STLF, is achieved by again the neural network method; but, the output is adjusted by a correction term, found through the difference between past years' forecasts and actual special day

loads. With this correction approach, errors reduce considerably and results turn out to be quite successful.

In order to suggest a solution for the neural network input selection problem, the principal component analysis is carried out to get the information content of the provided inputs as much as possible together with a fast learning. Results show that, input vector could be reduced to ten electric load and temperature elements from 18 initial components, which speeds up the process considerably.

The neural network that uses constant step size in real-time learning gives worse results than the model which proposes iterative step size adjustment.

Temperature is an important parameter on STLF. Models, that do not utilize temperature measurements in training, produce quite larger errors than the ones exploiting them as input parameters.

Overall results, as in the form of percent errors averaged through a year, verify that all the building blocks of this thesis mentioned above contribute positively to the solution of the STLF problem. For instance, 1.60% forecast error for all clusters of year 2002 including the special days, is not a bad result, as compared to the other similar works, which for instance obtained 1.67% error in [51].

Proposed neural network approach is compared with a traditional ARMA time series method and outperforms it in all day type results, especially for holidays. Therefore, the artificial neural network technology can be anticipated as a substitute of classical approaches for STLF. The advantages of neural networks include robustness to probability distribution assumptions, the ability to classify in the presence of nonlinear separation and their capability to perform reasonably well with incomplete data. This work proves the capability of artificial intelligence for STLF; therefore, it can be said that artificial intelligence methods are potential alternatives to other classical STLF methods.

9.2. Suggestions for Future Works

Both neural networks and STLF are wide areas to work on. Although a complete model is tried to be proposed in this thesis, some points are unavoidably left

inexperienced. Hoping to give an idea and perhaps a starting point for next researchers, the following suggestions on data clustering, input selection, training methods and modeling are given.

9.2.1. Suggestions on Data Clustering

Season information is given as an input variable in this work. Instead of this, separate neural networks can be formed having only the seasonal or monthly training data.

A data clustering can be done considering both season and day information, like Summer-Monday, Winter-Wednesday, etc.

Daily load profile has several characteristic regions, such as working hours, startup hours, evening, night, etc. data clustering can be performed according to this discrimination and separate neural networks can be formed.

In this work, special days are chosen from known holidays. However, load can have abnormality because of large temperature changes. Therefore, temperature forecasts can be taken into consideration for determining the special days.

In order to model special days, rough set approach can be utilized. Rough sets are efficient tools for classifying the data and differentiating the boundary classes. Special days can be thought as boundary classes and the main cluster to which they are close can be found by rough set analysis.

9.2.2. Suggestions on Input Selection

In this work, temperature and past load inputs are encoded continuously, hour is given as a sinusoidal component and day as a scalar. Since the importance of hour and day is not determined by their numeric value, they can be encoded binary.

There can be several alternative approaches that use different temperature variables as inputs. Among them, the greatest and the lowest temperature values of past days, the greatest and the lowest temperature forecasts of forecast day, the average temperature forecast of the forecast day and the temperature forecast of

the forecast hour can be mentioned. The difficulty here is in obtaining such forecasts.

9.2.3. Suggestions on Training Methods

The proposed hybrid training uses off-line weights for starting to real-time forecast and during the real-time forecasting phase, weight update continues. After a year is passed, for the next year, off-line training should be repeated with the last year's data. Instead of randomizing the initial weights in this off-line learning, the last weight values of the previous real-time training can be used.

The hybrid Levenberg-Marquart / simulated annealing algorithm can be used for training. The Levenberg-Marquart method is a deterministic optimization method used to find a local minimum of the error function. After finding such a minimum, the simulated annealing algorithm performs a search for possible other minima around the found one.

To avoid overtraining, regularization techniques can be used. This involves modifying the cost function to be minimized by adding 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 neural networks 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.

9.2.4. Suggestions on Modeling

Electric consumption profile is being changed due to different pricing intervals and due to new counters that can store such usage. Therefore, available past data will be no longer useful soon. A method could be proposed to correct deviations from the traditional load profile due to this fact.

A preliminary load forecast can be obtained by trained neural networks; then, a fuzzy expert system can modify this preliminary forecast considering the load variations due to changes in temperature or holiday profile.

Radial basis functions can be tried as model structure.

REFERENCES

1. Abraham A, Nath B, “A neuro-fuzzy approach for forecasting electricity demand in Victoria”, Applied Soft Computing Journal, Elsevier Science, Volume 1/2, pp. 127-138, 2001.

2. AlFuhaid, S., El-Sayed, M.A., Mahmoud, M.S., “Cascaded artificial neural networks for short-term load forecasting”, IEEE Transactions on Power Systems, Vol. 12, No. 4, pp. 1524-1529, November 1997.

3. Almedia L.B. et al., “Parameter Adaptation in Stochastic Optimization”, in: On-line Learning in Neural Networks (Ed. D. Saad), Cambridge University Press, 1998.

4. Bakirtzis, A.G. et al., “A neural network short term load forecasting model for the Greek power system”, IEEE Transactions on Power Systems, Vol. 11, No. 2, pp. 858-863, May 1996.

5. Baldi, P., Hornik, K., “Neural networks and principal component analysis: learning from examples without local minima”, Neural Networks, No. 2, pp. 53-58.

6. Bunn, D.W., and Farmer, E.D., Comparative Models for Electrical Load Forecasting, John Wiley & Sons, 1985.

7. Charytoniuk, W., Chen, M., "Very short-term load forecasting using artificial neural networks", IEEE Transactions on Power Systems, Vol. 15, No. 1, February 2000.

8. Chen, H., Liu, J., “A weighted multi-model short-term load forecasting system”, Proc. IEEE International Conference on Power System Technology, New York, NY, Vol. 1, pp. 557-561, 1998.

9. Chen, S.T., Yu, D.C., Moghaddamjo, A.R., "Weather sensitive short-term load forecasting using nonfully connected artificial neural network", IEEE Transactions on Power Systems, Vol. 7, No. 3, pp. 1098-1102, August 1992.

10. Choueiki, M.H., Mount-Campbell, C.A., Ahalt, S.C., “Building a ‘Quasi Optimal’ neural network to solve the short-term load forecasting problem”, IEEE Transactions on Power Systems, Vol. 12, No. 4, pp. 1432-1439, November 1997.

11. Choueiki, M.H., Mount-Campbell, C.A., Ahalt, S.C., "Implementing a weighted least squares procedure in training a neural network to solve the short-term load forecasting problem", IEEE Transactions on Power Systems, Vol. 12, No. 4, pp. 1689-1694, November 1997.

12. Chow, T.W.S., Leung, C.T., "Neural network based short-term load forecasting using weather compensation", IEEE Transactions on Power Systems, Vol. 11, No. 4, pp. 1736-1742, November 1996.

13. Cottrell, G., Munro, P., "Principal components analysis of images via backpropagation", Proceedings SPIE, Cambridge, 1988.

14. Daneshdoost, M. et al., "Neural network with fuzzy set-based classification for short-term load forecasting" IEEE Transactions on Power Systems., Vol. 13, No. 4, November 1998.

15. Dash, P.K. et al., "A real time short term load forecasting system using functional link network", IEEE Transactions on Power Systems, Vol. 12, No. 2, pp. 675-686, May 1997.

16. DeMers, D., "Dimensionality reduction for non-linear time series", Neural and Stochastic Methods in Image and Signal Processing (SPIE 1766), 1992.

17. Drezga, S., Rahman, S., "Input variable selection for ANN-based short-term load forecasting", IEEE Transactions on Power Systems, Vol. 13, No. 4, pp. 1238-1244, November 1998.

18. Drezga, S., Rahman, S., "Short-term load forecasting with local ANN predictors ", IEEE Transactions on Power Systems, Vol. 14, No. 3, pp. 844-850, August 1999.

19. Erkmen, I., Topallı, A., "Four methods for short term load forecasting using the benefits of artificial intelligence", Electrical Engineering, Published Online, DOI 10.1007/s00202-003-0163-9, April 2003.

20. Fan, J., McDonald, J.D., "A real-time implementation of short-term load forecasting for distribution power systems", IEEE Transactions on Power Systems, Vol. 9, No. 2, pp. 988-994, May 1994.

21. Gorr, W.L., "Research prospective on neural network forecasting", International Journal of Forecasting, Vol. 10, pp. 1-4, 1994.

22. Gross, G., Galiana, F.D., "Short term load forecasting", Proceedings of IEEE, Vol. 75, No. 12, pp. 1558-1573, 1987.

23. Hagan, M.T., Behr, S.M., "The time series approach to short term load forecasting", IEEE Transactions on Power Systems, Vol. 2, No. 3, pp. 785-791, August 1987.

24. Haykin S., Neural Networks A Comprehensive Foundation, Prentice Hall International Inc., New Jersey, 1999.

25. Heskes T.M., Kappen B., "On-line Learning Processes in Artificial Neural Networks", in: Mathematical Approaches to Neural Networks (Ed. J.G. Taylor), Elsevier Science Publishers, pp. 199-233, 1993.

26. Hill, T. et al., "Artificial neural network models for forecasting and decision making", International Journal of Forecasting, Vol. 10, pp. 5-15, 1994.

27. Hippert, H. S., Pedreira, C.E., Souza, R.C., "Neural networks for short-term load forecasting: a review and evaluation", IEEE Transactions On Power Systems, Vol. 16, No. 1, pp. 44-55, February 2001.

28. Ho, K.L. et al., "Short-term load forecasting of Taiwan power-system using a knowledge-based expert system." IEEE Transactions on Power Systems, Vol. 5, No. 4, pp. 1214-1221, 1990.

29. Ho, K.L., Hsu, Y.Y., Yang, C.C., "Short-term load forecasting using a multilayer neural network with an adaptive learning algorithm", IEEE Transactions on Power Systems, Vol. 7, No. 1, pp. 141-148, February 1992.

30. Jacobs, R, "Increased rates of convergence through learning rate adaptation", Neural Networks, 1:4, 1988.

31. Kesten, H., "Accelerated stochastic approximation", Annals of Mathematical Statistics, 29, pp. 41-59, 1958.

32. Khan, M.R., Abraham, A., Ondrusek, C., "Soft computing for developing short term load forecasting models in Czech Republic", Hybrid Information Systems, pp. 207-222, 2002.

33. Khotanzad, A., Zhou, E., Elragal, H., "A neuro-fuzzy approach to short-term load forecasting in a price-sensitive environment", IEEE Transactions on Power Systems, Vol. 17, No. 4, pp. 1273-1282, November 2002.

34. Khotanzad, A., Afkhami-Rohani, R., Maratukulam, D., "ANNSTLF – Artificial neural network short-term load forecaster – generation three", IEEE Transactions on Power Systems, Vol. 13, No. 4, pp. 1413-1422, November 1998.

35. Khotanzad, A. et al,. "ANNSTLF – A neural-network-based electric load forecasting system", IEEE Transactions on Neural Networks, Vol. 8, No. 4, pp. 835-846, July 1997.

36. Khotanzad, A. et al., "An adaptive modular artificial neural network hourly load forecaster and its implementation at electric utilities", IEEE Transactions on Power Systems, Vol. 10, No. 3, pp. 1716-1722, August 1995.

37. Kiartzis, S. J. et al., “Short-term load forecasting in an autonomous power system using artificial neural networks”, IEEE Transactions on Power Systems, Vol. 12, No. 4, pp. 1591-1596, November 1997.

38. Kim, K.H. et al., “Implementation of hybrid short-term load forecasting system using artificial neural networks and fuzzy expert systems”, IEEE Transactions on Power Systems, Vol. 10, No. 3, pp. 1534-1539, August 1995.

39. Kim, K.H., Youn, H.S., Kang, Y.C., “Short-term load forecasting for special days in anomalous load conditions using neural networks and fuzzy inference method”, IEEE Transactions on Power Systems, Vol. 15, No. 2, pp. 559-565, May 2000.

40. Kramer, M., “Nonlinear principal component analysis using autoassociative neural networks”, AIChE Journal, No. 37, pp. 233-243, 1991.

41. Lamedica, R. et al., “A neural network based technique for short-term forecasting of anomalous load periods”, IEEE Transactions on Power Systems, Vol. 11, No. 4, pp. 1749-1756, November 1996.

42. Liang, R., Cheng, C., “Combined fuzzy-neural network to short-term load forecasting”, Journal of Science and Technology, Vol. 8, No. 3, pp. 193-203, 1999.

43. Lu, C.N., Wu, H.T., Vemuri, B., “Neural network based short-term load forecasting”, IEEE Transactions on Power Systems, Vol.8, No. 1, pp. 336-342, February 1993.

44. Mastorocostas, P.A., Theocharis, J.B., Bakirtzis, A.G., “Fuzzy modeling for short term load forecasting using the orthogonal least squares method”, IEEE Transactions on Power Systems, Vol. 14, No. 1, pp. 29-36, February 1999.

45. Matthews, A., et al., “Weather normalized intermediate term capacity forecasting: a procedural innovation”, IEEE Transactions on Power Systems, Vol. 3, No. 3, pp. 1291-1297, August 1988.

46. Mbamalu, G.N., El-Hawary, M.E., “Load forecasting via subotimal seasonal autoregressive models and iteratively reweighted least squares estimation”, IEEE Transactions on Power Systems, Vol. 8, No. 1, pp.343-348, February 1993.

47. Moghram, S., Rahman, S., “Analysis and evaluation of five short-term load forecasting techniques”, IEEE Transactions on Power Systems, Vol. 4, No. 4, pp. 1484-1491, October 1989.

48. Mohammed, O. et al., “Practical experiences with an adaptive neural network short-term load forecasting system”, IEEE Transactions on Power Systems, Vol. 10, No. 1, pp. 254-265, February 1995.

49. Muller K.L. et al., “On-line Learning in Switching and Drifting Environments with Application to Blind Source Separation”, in: On-line Learning in Neural Networks (Ed. D. Saad), Cambridge University Press, pp. 93-110, 1998.

50. Nogales, F.J. et al., “Forecasting next-day electricity prices by time series models”, IEEE Transactions on Power Systems, Vol. 17, No. 2, pp. 342-348, May 2002.

51. Papadakis, S. E., “A novel approach to short-term load forecasting using fuzzy neural network”, IEEE Transactions on Power Systems, Vol. 13, No. 2, pp. 480–492, May 1998.

52. Papalexopoulos A.D., Hesterberg, T.C., “A regression-based aproach to short term system load forecasting”, IEEE Transactions on Power Systems, Vol. 5, No. 4, pp. 1535-1547, November 1990.

53. Papalexopoulos, A.D., Hao, S., Peng, T.M., “An implementation of a neural network based load forecasting model for the EMS”, IEEE Transactions on Power Systems, Vol. 9, No. 4, November 1994.

54. Park, C. et al., “Electric load forecasting using an artificial neural network”, IEEE Transactions on Power Systems, Vol. 6, No. 2, pp. 442-449, May 1991.

55. Peng, T.M., Huble, N.F., Karady, G.G., “Advancement in the application of neural networks for short-term load forecasting”, IEEE Transactions on Power Systems, Vol. 7, No. 1, February 1992.

56. Rajurkar, K.P., Nissen, J.L., “Data-dependent systems approach to short-term load forecasting”, IEEE Transactions on Systems, Man and Cybernetics, Vol. 15, No. 4, pp.532-536, 1985.

57. Lawrance, S., Giles, C.L., Fong, S., “Natural language grammatical inference with recurrent neural networks”, IEEE Transactions on Knowledge and Data Engineering, Vol. 12, No. 1, pp. 126-140, 2000.

58. Saini, L.M., Soni, M.K., “Artificial neural network-based peak load forecasting using conjugate gradient methods”, IEEE Transactions on Power Systems, Vol. 17, No. 3, pp. 907-912, August 2002.

59. Senjyu, T. et al., “One-hour-ahead load forecasting using neural network”, IEEE Transactions on Power Systems, Vol. 17, No. 1, February, 2002.

60. Silva A.P., Moulin L.S., “Confidence intervals for neural network based short-term load forecasting”, IEEE Transactions on Power Systems, Vol. 15, No. 4, pp. 1191-1196, November 2000.

61. Srinivasan, S.S. et al., “Parallel neural network-fuzzy expert system strategy for short-term load forecasting: system implementation and performance

evaluation", IEEE Transactions on Power Systems, Vol. 14, No. 3, pp. 1100-1106, August 1999.

62. Sudjianto, A., Wasserman, G.S., "A nonlinear extension of principal component analysis for clustering and spatial differentiation", IIE Transactions, No. 28, pp. 1023-1028, 1996.

63. Taylor, J.W., Buizza, R., "Neural network load forecasting with weather ensemble predictions", IEEE Transactions on Power Systems, Vol.17, No. 3, pp. 626-632, August 2002.

64. Topallı, A., Erkmen, I., "A hybrid learning for neural networks applied to short term load forecasting", Neurocomputing, Vol. 51, pp. 495-500, April 2003.

65. Tzafestas, S., Tzafestas, E., "Computational intelligence techniques for short-term electric load forecasting", Journal of Intelligent and Robotic Systems, Vol. 31, No. 1-3, pp. 7-68, 2001.

66. Vemuri, S., Huang, W.L., Nelson, D.J., "On-Line algorithms for forecasting hourly loads of an electric utility", IEEE Transactions on Power Applications & Systems, Vol. 100, No. 8, pp. 3775-3784, 1981.

67. Verbos, P., "Backpropagation through time", Proc. of the IEEE, Vol. 78, No. 10, pp. 1550-1560, October 1990.

68. Vermaak, J., Botha, E.C, "Recurrent neural networks for short-term load forecasting", IEEE Transactions on Power Systems, Vol. 13, No. 1, pp. 126-132, February 1998.

69. Yang H., et al., "Identification of ARMAX model for short-term load forecasting: an evolutionary programming approach", IEEE Transactions on Power Systems, Vol. 11, No. 1, pp.403-408, February 1996.

70. Yoo, H., Pimmel, R.L., "Short term load forecasting using a self-supervised adaptive neural network", IEEE Transactions on Power Systems, Vol. 14, No. 2, pp 779-784, May 1999.

71. Zhang, G., Patuwo, B.E., Hu, M.Y., "Forecasting with artificial neural networks: the state of the art", International Journal of Forecasting., Vol. 14, pp. 35–62, 1998.

APPENDIX

COMPLETE LIST OF SPECIAL DAYS

1999	2000	2001	2002
16 Jan Saturday	7 Jan Friday	3 Mar Saturday	21 Feb Thursday
17 Jan Sunday	8 Jan Saturday	4 Mar Sunday	22 Feb Friday
18 Jan Monday	9 Jan Sunday	5 Mar Monday	23 Feb Saturday
19 Jan Tuesday	10 Jan Monday	6 Mar Tuesday	24 Feb Sunday
20 Jan Wednesday	11 Mar Saturday	7 Mar Wednesday	25 Feb Monday
21 Jan Thursday	12 Mar Sunday	8 Mar Thursday	23 Apr Tuesday
22 Jan Friday	13 Mar Monday	9 Mar Friday	19 May Sunday
23 Jan Saturday	14 Mar Tuesday	10 Mar Saturday	30 Aug Friday
24 Jan Sunday	15 Mar Wednesday	11 Mar Sunday	28 Oct Monday
25 Jan Monday	16 Mar Thursday	12 Mar Monday	29 Oct Tuesday
27 Mar Saturday	17 Mar Friday	23 Apr Monday	4 Dec Wednesday
28 Mar Sunday	18 Mar Saturday	19 May Saturday	5 Dec Thursday
29 Mar Monday	19 Mar Sunday	30 Aug Thursday	6 Dec Friday
30 Mar Tuesday	20 Mar Monday	29 Oct Monday	7 Dec Saturday
31 Mar Wednesday	23 Apr Sunday	15 Dec Saturday	8 Dec Sunday
23 Apr Friday	19 May Friday	16 Dec Sunday	9 Dec Monday
19 May Wednesday	30 Aug Wednesday	17 Dec Monday	
30 Aug Monday	29 Oct Sunday	18 Dec Tuesday	
28 Oct Thursday	23 Dec Saturday		
29 Oct Friday	24 Dec Sunday		
	25 Dec Monday		
	26 Dec Tuesday		
	27 Dec Wednesday		
	28 Dec Thursday		
	29 Dec Friday		
	30 Dec Saturday		
	31 Dec Sunday		

VITA

Ayça Kumluca Topallı was born in Aydın, Turkey on October 26, 1972. She received her B.S. and M.S. degrees in Electrical and Electronics Engineering from Middle East Technical University in June 1994 as an honor student and in June 1997 as a high honor student, respectively. She worked in the Electrical and Electronics Engineering Departments of Middle East Technical University and Ninth September University as a research assistant from 1994 to 2000; five years in the former and one year in the latter. In 1998, she had been invited as a guest scientist to IROE, Florence and ICTP, Trieste for five months to make researches on her Ph.D. thesis. Since 2000, she has worked as an R&D project engineer in private companies, including TEBA Inc., VESTEL Electronics Inc., and TEON Software and Hardware Design House. Currently, she is working for BEKO Electronics Inc. in İzmir.

Her main areas of interest include neural networks, approximate reasoning, intelligent control, time series forecasting, optimization, and fuzzy logic. She has 16 papers published so far in international proceedings and journals.