IMPROVED EXTENSION NEURAL NETWORKS FOR LEAD-ACID BATTERY MODELING

by

Yusuf Sipahi

Submitted to the Graduate School of Engineering and Natural Sciences
in partial fulfillment of
the requirements for the degree of
Master of Science

Sabancı University

February 2011

Improved Extension Neural Networks

For Lead-Acid Battery Modeling

APPROVED BY:

Assist. Prof. Dr. Ahmet Onat (Thesis Advisor)	....
Assoc. Prof. Dr. Berrin Yanıkoğlu	....
Assoc. Prof. Dr. Serhat Yeşilyurt	....
Assoc. Prof. Dr. Albert Levi	....
Assist. Prof. Dr. Hüsnü Yenigün	....

DATE OF APPROVAL: ....

Acknowledgements

Foremost, I would like to express my sincere gratitude to my supervisor Assist. Prof. Dr. Ahmet Onat for the continuous support of my study and research, for his patience, motivation, enthusiasm, and immense knowledge. His guidance helped me in all the time of research and writing of this thesis. I feel fortunate to have had the opportunity to both learn from and work alongside with him.

Secondly, I would like to thank my projectmate Sena Ergullu for her support and contribution to this project. I am also grateful to all my labmates: Duruhan Ozcelik, Bulut Coskun, Umut Sen, Serhat Dikyar, Ozan Tokatli, Ahmetcan Erdogan, Alper Ergin, Can Palaz and Kadir Haspalamutgil. Most importantly, none of this would have been possible without the love and patience of my family. My immediate family to whom this dissertation is dedicated to, has been a constant source of love, concern, support and strength all these years. Last but not least, I wish to express my gratitude to my love, Pelin Arslan, not only for her constant encouragement but also her patience and understanding throughout.

This thesis was further made possible by funding from Santez project involving Sena Ergullu and Ahmet Onat.

Improved Extension Neural Networks For Lead-Acid Battery Modeling

Yusuf Sipahi

EECS, Master's Thesis, 2011

Thesis Supervisor: Ahmet Onat

Keywords: Extension Neural Networks, Fault Diagnosis, Lead-Acid Battery, Artificial,Modeling of Nonlinear Dynamic Systems, Artificial Intelligence Methods

Abstract

There is an increasing demand for man-made dynamical systems to be reliable and safe. If a fault can be detected quickly, appropriate actions should be taken to prevent critical accidents, high cost malfunctions or failures. The key point in fault diagnosis is the assumption of the availability of good mathematical model of the plant. Mathematical modeling of non-linear dynamical systems may be computationally hard and time consuming. Therefore, modeling the plant using machine learning methods such as Neural Networks (NN), fuzzy logic, extension neural networks (ENN) can be more advantageous.

Although a dynamical system is modeled via machine learning methods, there can be non-measurable states which are used in the system. Even though they are estimated with mathematical approaches, they can drift in time. Classification methods can be applied totally or to initialize the mathematical estimation. Although ENN is one of the promising classification methods, it sometimes gives poor results due to insensitivity to scatter of data-points. Its shifting and updating property takes more iterations than comparable methods to give an acceptable error rate.

In this thesis, we propose improved extension neural networks (IENN) which improve on ENN's linear clustering method by using quadratic clustering and generating clustering criteria which depend on statistical properties of the training set. Rechargable Lead-Acid Battery is modeled via feed-forward NN approach and its state of charge is classified via proposed IENN method. The proposed method produces more accurate classifying results than ENN.

Kurşun-Asit Batarya Modellemesi İçin Geliştirilmiş Genişletilmiş Yapay Sinir Ağları

Yusuf Sipahi
EECS, Master Tezi, 2011
Thesis Supervisor: Ahmet Onat

Keywords: Genişletilmiş Yapay Sinir Ağları, Hata Bulguları, Kurşun-Asit Batarya, Non-Lineer Dinamik Sistemlerin Modellenmesi, Yapay Zeka Metodları

Özet

İnsan yapımı dinamik sistemlerinin güvenilir ve gürbüz olmasına gittikçe artan ihtiyaç var. Eğer bir hata çabuk bir şekilde bulunur, gerekli önlemler alınırsa sistem kritik kazalardan, yüksek maliyetli hasarlardan ve de daha büyük hatalardan kurtarılabilir. Hata teşhisinde tesisin iyi bir matematiksel modelinin varlığı büyük önem taşır. Non-Lineer dinamik bir sistemin matematiksel modellemesinin bulunması zor ve zaman alıcıdır. Bu nedenle, yapay sinir ağı (YSA), bulanık mantık ve genişletilmiş yapay sinir ağları (GYSA) gibi yapay zeka ile öğrenme yöntemleri daha avantajlı olabilir.

Dinamik sistemlerin modellemesi yapay zeka ile öğrenme yöntemleri ile yapılsa bile, sistemde kullanılan, ölçülemeyen durumlar olabilir. Bunlar matematiksel olarak hesaplansalarda, zamanla hesaplanan değerler kayabilir. Sınıflandırma yöntemleri tamamen veya hesaplamayı doğru noktaya çekebilmek için kullanılabilir. GYSA umut veren bir sınıflandırma yöntemidir ama veri noktalarının dağılımına hassasiyeti olmadığı için bazen kötü sonuçlar verebilir. GYSA, kabul edilebilir bir hataya ulaşana kadar kullandığı kaydırma ve güncelleme özelliği, diğer karşılaştırılabilir metodlara göre daha uzun sürer.

Bu tezde, sunulan geliştirilmiş genişletilmiş yapay sinir ağları (GGYSA) ile GYSA' nın lineer kapsamasını kuadratik olarak değiştiriyor ve eğitim setinin istatiksel özelliklerine göre kapsama kriterini geliştiriyoruz. Şarj edilebilir kurşun asit bataryanın YSA ile modeli oluşturulup ve şarj durumu önerilen GGYSA ile bulunmuştur. Önerilen GGYSA metodunun GYSA yöntemine göre daha doğru sınıflandırma sonuçları verdiği görülmüştür.

TABLE OF CONTENTS

List of Figures **xii**

List of Tables **xv**

1 Introduction **1**

1.1 Fault Diagnosis and Their Usage 1

1.2 Contributions of Thesis 3

1.3 Implementation of Proposed Method on Lead-Acid Battery 4

2 Background **6**

2.1 Dynamic-System Modeling with Artificial Neural Network 6

2.1.1 Feed-Forward Neural Network Architecture 9

2.2 Classification Via Extension Neural Networks 12

2.2.1 Extension Theory 13

2.2.2 Extension Neural Networks 16

3 Improved Extension Neural Networks **23**

3.1 Using A Hybrid Approach For Classification 30

3.1.1 Validation 32

4 Rechargeable Lead Acid Battery Principles and Measurement Methods **34**

4.1 Lead Acid Battery Principles of Operation 34

4.2 Battery Characteristics 36

4.2.1 Capacity 36

4.2.2 State Of Charge 37

4.2.3 Effects of Temperature 39

4.3 Experimental Setup and Measurement Methods 39

4.3.1 Experimental Setup 39

4.3.2 Measurement Methods 42

5 Results **47**

5.1 Modeling Lead-Acid Battery via Feed-Forward Neural Network 47

5.2 Lead-Acid Battery State Of Charge Estimation Via Improved Extension Neural Networks 51

5.2.1 ENN 52

5.2.2 Linear IENN 52

5.2.3 Quadratic IENN 53

5.2.4 Hybrid IENN 54

5.2.5 Comparison of Performance with IENN 55

6 Conclusion **60**

List of Figures

1.1.1 Model Based FD Approach 4

1.3.1 An example NN illustration for a dynamic system 5

2.1.1 An example ANN illustration for a dynamic system 8

2.1.2 (a) Training Phase (b) Execution Phase 9

2.1.3 Feed-Forward Neural Network Architecture 10

2.2.1 Extended Correlation Function 16

2.2.2 Extension Distance 18

2.2.3 Extension Neural Network Architecture 19

2.2.4 Updating Separators:(a)Before Update; (b)After Update 22

3.0.1 Updating Separators:(a) Recent Class View;(b) After Few Iterations Class View 24

3.0.2 (a) Linear Separator (b) Non-Linear Separator 25

3.0.3 (a) ENN Clustering with one input (b) ENN Update Clustering with one input (c)IENN Non-Linear Clustering with one input (d)IENN Update with one input 28

3.0.4 (a) Linear and Non-Linear separator with One Input (b)Linear and Non-Linear separator with Two Inputs 29

3.0.5 Learning Rate vs. Iteration Number 30

3.1.1 Hybrid Classification where input data set for class A has large variance whereas B is narrow 31

3.1.2 Measurements collected more than one region 32

4.1.1 Vehicle Electrical Network 35

4.3.1 Battery data acquisition system experimental setup 40

4.3.2 Experimental Setup Overview 41

4.3.3 Circuit Model of Battery 42

4.3.4 Voltage vs. Time Graph for making steady state voltage measurements 44

4.3.5 Current vs. Time Graph for making steady state current measurements 44

4.3.6 Internal Resistance vs. SoC 45

4.3.7 Measured and Calculated OCV vs. SoC 46

4.3.8 SCC vs. SoC 46

5.1.1 Input Current 48

5.1.2 Output Voltage 49

5.1.3 SoC 49

5.1.4 Battery Modeling: (a) Training Stage (b) Testing Stage 50

5.1.5 NN Output vs. Battery Output Red - NN Output, Blue - Battery Output 50

5.2.1 Comparison of ENN, Low α_{thr} Hybrid IENN and NN 57

5.2.2 Comparison of High α_{thr} Hybrid IENN, Linear IENN and Quadratic IENN 58

List of Tables

5.1 Classes of SOC 51

5.2 ENN 10-fold Cross Validation Results 52

5.3 Linear IENN Classifying 10-fold Cross Validation Results 53

5.4 Quadratic IENN 10-fold Cross Validation Results 53

5.5 Low α_{thr} Hybrid IENN 10-fold Cross Validation Results 54

5.6 High α_{thr} Hybrid IENN 10-fold Cross Validation Results 55

5.7 Feed-Forward NN 10-fold Cross Validation Results 55

5.8 Classification Performance 56

1 INTRODUCTION

This chapter gives an overview about fault diagnosis (FD), model based methods, their usage and demanded machine learning methods.

1.1 Fault Diagnosis and Their Usage

Fault diagnosis is generally performed by comparing the real-time signals and parameters of a plant with those of its model. Any discrepancies are interpreted to identify a fault in the plant and its location. Designing a model for a nonlinear plant such as a commercial electromechanical device or a control system is difficult even if the plant does not have a complex structure, because its parameters are not disclosed. In such cases, a nonlinear model can be derived using machine learning methods such as neural network or fuzzy logic by examining samples of its input-output signals.

In this thesis, a less frequently used method, the extended neural network is taken up and an improved version, the "Improved Extended Neural Network“ (IENN) is proposed. The performance of the method is compared with other methods' and results are presented. Although this paper focuses on modeling of nonlinear systems for fault diagnosis applications, the proposed method is general and may have many application areas.

In daily life, there is an increasing demand for man-made dynamical systems to be more reliable and safer. If a fault can be detected quickly, appropriate actions can be taken and prevent from critical accidents, high cost malfunction or failure. In [12] it is stated that, the fault is a state that may lead to a malfunctions or failures of the system. This statement explains the distinction between a fault and a failure. After detecting the fault, next process of FD is the fault isolation, identification or classification.

Consequently, the idea of FD methods is to investigate a system under normal conditions and compare the found investigations with the actual system running in real-time. By using hardware redundancy technique, multiple physical devices are provided and their output signals are compared with the actual devices' output signals to detect the type of a fault and its location. This technique has high costs, therefore it is not preferred. Second technique is to mimic a system by using analytical redundancy. Analytical redundancy creates mathematical models which gives the same output value with the system for a given input. This technique is more preferable than hardware redundancy due to no cost.

Analytical approaches are divided into two groups: quantitative models and the models generated by machine learning methods. Observers [9], parameter estimation [11] and parity equations [10] are some of the used quantitative approaches in use with analytical redundancy. Machine learning techniques are used to mimic a system. Two of the most famous machine learning approaches used in FD are fuzzy modeling and neural network (NN) approach. In this thesis NN is used for modeling and the reasons can be found in Chapter 2.1.

The model of a model based FD approach is given in Fig. 1.1.1. In closed-loop systems, although plant changes from normal condition to faulty condition, controller tries to move the plant to normal condition, therefore by measuring only plant output y may not give positive information about a failure. According to this situation, mathematical model of the plant is created and as an input, actuator's output u_a is given to it. As a result output of the model $\hat{y}$ and y is compared to find residual r. This residual is then used in fault identification techniques.

The key point in FD approach is the assumption of availability of good mathematical model of the plant. In practice, this idea is not valid because unavoidable modeling uncertainties arise due to modeling errors, noise measurement and external disturbances which effects the performance of FD approach and giving false fault alarms as stated in [2]. This makes quantitative model-based analytical approaches very difficult to use in real systems. And also mathematical modeling of non-linear dynamical systems may be computationally hard and time. Therefore, using machine learning approaches is more advantageous than using mathematical approaches.

A non-linear system [2], with one output can be described as in (1.1) where $x \in \Re^n$ is state vector, $u \in \Re^m$ is input vector, $y \in \Re$ is the output of the system, and ξ, f: $\Re^n$ x $\Re^m \to \Re^n$ are the smooth vector fields, which represent the nominal system and change in the system due to a fault. Modeling uncertainty η: $\Re^n$ x $\Re^m$ x $\Re^+ \to \Re^n$, is also a smooth vector field, and h: $\Re^n \to \Re$ is a smooth function. Time profile of a fault is represented by the function β: $\Re \to \Re$. If a sudden (abrupt) fault happens in the system, function β becomes step function, whereas in slowly developing faults (incipent) it becomes a ramp function.

$$\begin{aligned} \dot{x}(t) &= \xi(x(t), u(t)) + \eta(x(t), u(t), t) + \beta(t - T) f(x(t), u(t)) \\ y &= h(x(t)) \end{aligned} \tag{1.1}$$

In machine learning approaches, these vector fields and functions of the non-linear systems are imitated by pattern recognition techniques, using state vector, input vector and output to generate a plant model as illustrated in Fig. 1.1.1.

1.2 Contributions of Thesis

Modeling of a system accurately is important in FD as discussed previously. In most of the modeling problems some non-measurable state variables are necessary to model the system accurately. When estimation of these states are difficult or integration errors accumulate in time, classification methods can be applied totally or to initialize the estimation. ENN is one of such promising classification methods. Although it

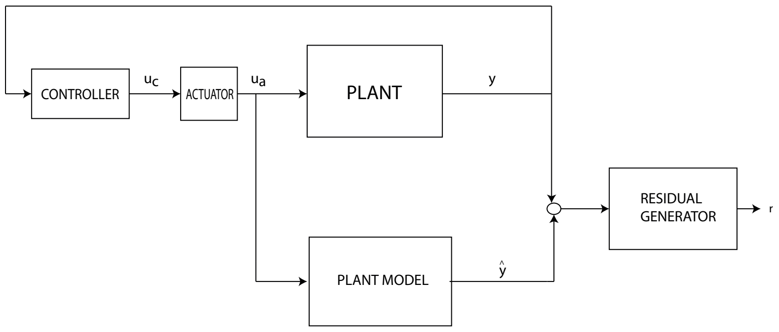


Figure 1.1.1: Model Based FD Approach

sometimes gives poor results due to insensitivity to scatter of data-points and due to its shifting and updating property, it requires more iterations than comparable methods to give an acceptable error rate. In this thesis we propose novel Improved ENN classification methods (IENN) which improves the performance of ENN by:

- making ENN's linear clustering method quadratic.
- generating clustering criteria, depending on statistical properties of the training set.
- hybridizing the cluster separations by using linear or quadratic separators based on the statistical properties of the data.

Due to the proposed method, more accurate classifying results than ENN's are acquired.

1.3 Implementation of Proposed Method on Lead-Acid Battery

Lead-acid battery has non-linear characteristics. Rather than using mathematical approaches, Neural Network (NN) modeling approach is used in this thesis. Predicting the recent reduction amount of capacity which is called state of charge (SoC) is

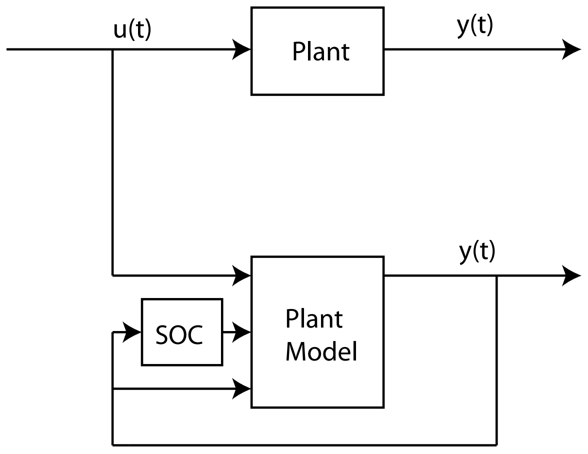


Figure 1.3.1: An example NN illustration for a dynamic system

a difficult task while the battery is under operation. Current integration method is widely used in literature to find SoC but it is a weak method due to accumulation of integration errors in time. In the case of this weakness, a model which predicts the SoC is highly needed and can be used as illustrated in Fig. 1.3.1 where $u(t)$ is the current flowing through the battery and $y(t)$ is the terminal voltage of the battery. Our purpose in this thesis is to improve ENN to IENN and classify the instant measured and calculated values of the battery to predict the ten-spot regions of SoC accurately.

2 BACKGROUND

In the previous chapter, the advantage of applying machine learning approaches compared to mathematical approaches is discussed. In this direction, artificial neural networks (ANN) model is generated to mimic the battery terminal voltage and the performance of the model is investigated in this thesis. Additionally, proposed IENN classification method is used to classify SoC accurately. Therefore, this chapter gives a background information about ANN and ENN.

2.1 Dynamic-System Modeling with Artificial Neural Network

ANN was designed to reproduce a human brain which makes generalization with a previously learned event. It is a supervised learning model, in learning phase it uses system's input-output data pairs. In [3], the main strengths of ANN is summarized as follows:

- Easily deals with complex problems.
- From the learned situations, generalization of known circumstances to unknown circumstances.
- Because of the high degree of parallel structure, gives low operational response times "after training phase" due to fast calculations.

Due to adapting to complex problems easily, in the last two decades ANN has become the most famous modeling technique in FD. Model-based fault diagnosis methods heavily depend on the accuracy of the model. FD uses ANN's strong nonlinear mapping and robustness to noise. Rather than using a mathematical model, ANN is more beneficial because it gives fast response times and it can be placed in to the on-line fault diagnosis systems. In this thesis, feed-forward neural network is used because of common usage in FD literature.

ANN mimics the plant by using system's input output pairs, and generates a function which represents the plant. Due to the ability of robust mapping of input vector $u(t)$ and state vector $x(t)$ to output vector $y(t)$, even under with the presence of noise ANN is a useful tool in fault diagnosis.

A dynamic-system can be defined as shown in (2.1) where f is a non-linear function, $y(t)$ is output, $u(t)$ is input and $x(t)$ is state vector. This plant can be modeled using a NN by feeding back current and delayed values of its outputs and known states. An example of using ANN for a dynamic model is illustrated in Fig. 2.1.1.

$$y(t) = f(u(t), x(t)) \tag{2.1}$$

Nonlinear model represented with a nonlinear function f, can be mimicked by ANN to obtain a dynamic ANN, making it suitable for dynamic models for which mathematical models are too difficult or too expensive to obtain.

In the training phase, ANN is trained with the input and output pairs of the system under examination which is in absence of a fault. Clearly, the number of input nodes is fixed on the basis of the number of input/output signal samples necessary to describe the system structure. A setup similar to Fig. 2.1.2 (a) can be used to obtain training data for an ANN with the plant. Fig. 2.1.2 (b) is the execution phase, setting the trained ANN in parallel with the plant under control. This makes it possible to detect faults.

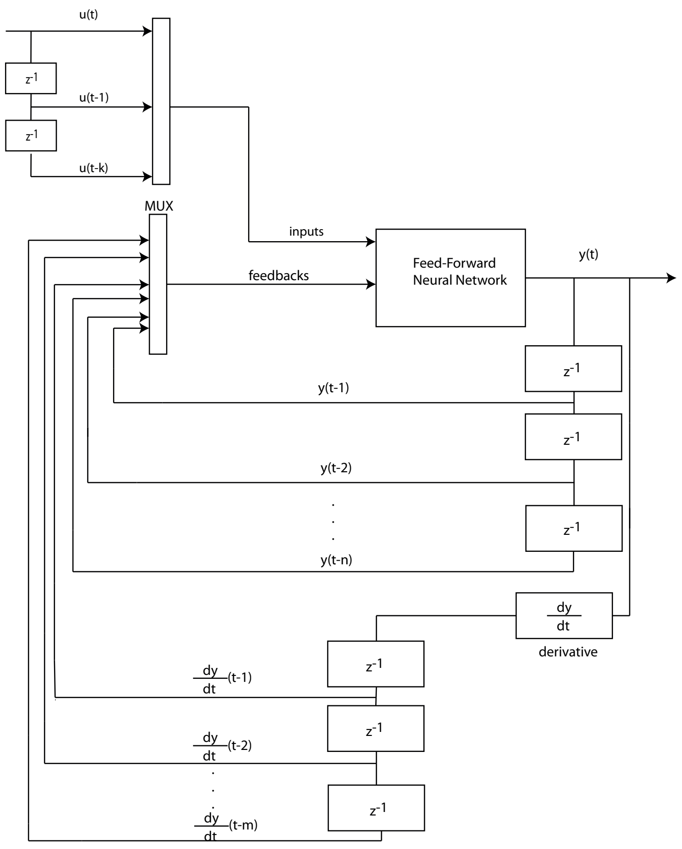


Figure 2.1.1: An example ANN illustration for a dynamic system

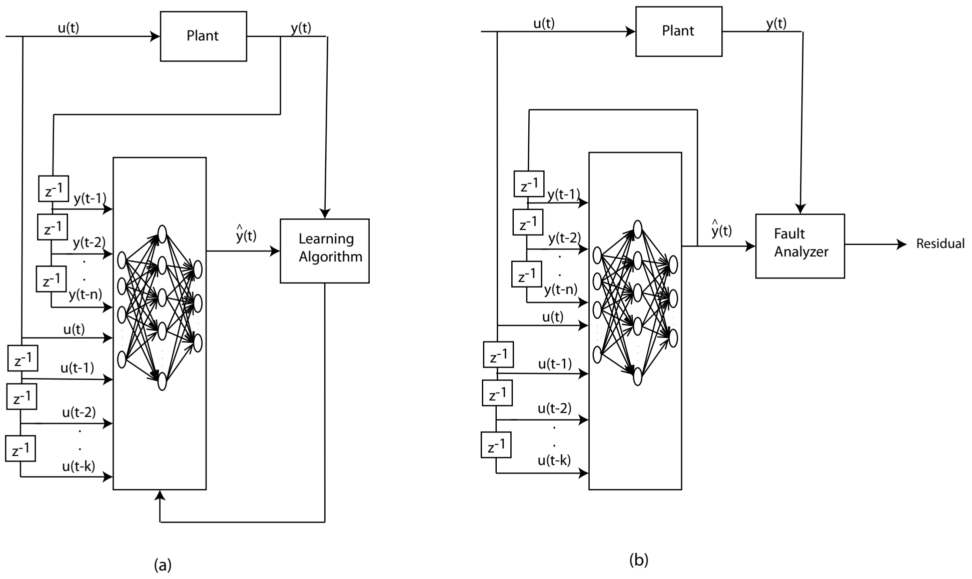


Figure 2.1.2: (a) Training Phase (b) Execution Phase

2.1.1 Feed-Forward Neural Network Architecture

A feed-forward neural network (ffNN) architecture is shown in Fig. 2.1.3 where two layers are used which are called hidden and output layer, respectively. Depending on the complexity of the system, more layers can be used, whereas in literature it is not recommended because generalization is reduced.

There are two operations in training an ffNN using backpropagation method as stated in [1]. First operation involves calculation of output o_k^p by feeding p^{th} instance data to the input layer x_i and passing through hidden and output layers weights. This operation is called feed-forward operation. The output of n_j hidden neuron is y_j and it is calculated by (2.2), where f_h is the hidden node activation function. Then o_k^p can be formulated by (2.2), where f_o is the output node activation function.

$$
\begin{aligned}
net_k &= \sum_{j=1}^{M} y_j w_{jk} \\
o_k^p &= f_o(net_k)
\end{aligned}
\tag{2.2}
$$

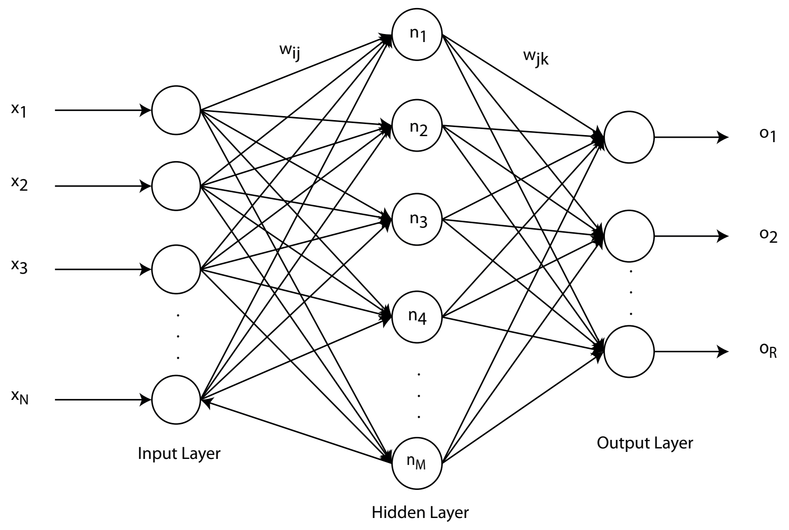


Figure 2.1.3: Feed-Forward Neural Network Architecture

$$net_j = \sum_{j=1}^{N} x_i w_{ij}$$
$$y_j = f_h(net_j) \tag{2.3}$$

Some of the commonly used activation functions are:

$$\text{Sigmoid Function} : f(n) = \frac{1}{1+e^{-n}}$$
$$\text{Tangent Sigmoid Function} : f(n) = \frac{e^{2n}-1}{e^{2n}+1} \tag{2.4}$$
$$\text{Linear Function} : f(n) = n$$

Second operation is called backpropagation. Error on pattern p which is the input to the network, is defined by E^p. It is calculated via summing the squares of difference between the desired outputs t_k^p, which are called target outputs, and the calculated o_k^p as shown in (2.5) where R is the number of outputs. Weights are adjusted until desired error rate is found.

Weight update for output layer is as follows:

$$E^p = \frac{1}{2}\sum_{k=1}^{R}(t_k^p - o_k^p)^2 \tag{2.5}$$

For reducing error, weight update is needed. Therefore weight derivative of E^p should be checked as in (2.6).

$$\begin{aligned} \frac{\partial E^p}{\partial w_{jk}} &= \frac{\partial E^p}{\partial net_k}\frac{\partial net_k}{\partial w_{jk}} \\ \frac{\partial net_k}{\partial w_{jk}} &= o_j \end{aligned} \tag{2.6}$$

Implementing gradient descent, change in the output weight Δw_{jk} is shown in (2.7) where η is the learning rate.

$$\Delta w_{jk} = -\eta \frac{\partial E^p}{\partial net_k} o_j \tag{2.7}$$

By using chain rule, $\frac{\partial E^p}{\partial net_k}$ is obtained as shown in (2.8). By combining (2.2) and (2.8), $\frac{\partial o_k^p}{\partial net_k}$ can be determined as in (2.9).

$$\frac{\partial E^p}{\partial net_k} = \frac{\partial E^p}{\partial o_k^p}\frac{\partial o_k^p}{\partial net_k} \tag{2.8}$$

$$\frac{\partial o_k^p}{\partial net_k} = f'(net_k) \tag{2.9}$$

Calculation of $\frac{\partial E^p}{\partial o_k^p} = -(t_k^p - o_k^p)$, therefore weight update for $w_j k$ is (2.10).

$$\Delta w_{jk} = \eta f'(net_k)(t_k^p - o_p^k)o_j \tag{2.10}$$

Weight update for hidden layer is as follows: In this step, the n_j hidden node weight w_{ij} is adjusted. Derivative of the E^p with respect to $w_i j$ is calculated via (2.11).

$$\frac{\partial E^p}{\partial w_{ij}} = \frac{\partial E^p}{\partial net_j}\frac{\partial net_j}{\partial w_{ij}} \tag{2.11}$$

By using chain rule, (2.11) expands to (2.12), where y_j is the output of hidden neuron n_j

$$\begin{aligned}
\frac{\partial E^p}{\partial net_j} &= \frac{\partial E^p}{\partial y_j}\frac{\partial y_j}{\partial net_j} \\
\frac{\partial E^p}{\partial w_{ij}} &= \frac{\partial E^p}{\partial y_j}\frac{\partial y_j}{\partial net_j}\frac{\partial net_j}{\partial w_{ij}} \\
\frac{\partial E^p}{\partial net_j} &= \frac{\partial E^p}{\partial y_j} f'(net_j)x_i
\end{aligned} \tag{2.12}$$

Target of n_j is not known. Hence, $\frac{\partial E^p}{\partial y_j}$ can only be calculated through n_j's contribution to the derivative of E^p with respect to net_k at the output nodes as shown in (2.13).

$$\frac{\partial E^p}{\partial y_j} = \sum_{k=1}^{R} w_{jk}\frac{\partial E^p}{\partial net_k} \tag{2.13}$$

In (2.13), $\frac{\partial E^p}{\partial net_k}$ which is calculated in (2.8), leads weight update process to be bounded with output weight update.

$$\Delta w_{ij} = -\eta \left(f'(net_j)x_i \sum_{k=1}^{R} w_{jk}\frac{\partial E^p}{\partial net_k} \right) \tag{2.14}$$

2.2 Classification Via Extension Neural Networks

Extension Neural Network (ENN) is a new pattern recognition classification method based on concepts from ANN and extension theory (ET). ENN uses extension distance (ED) to measure the similarity between instances and classes. In FD, ENN's classification property can be implemented to find non-measurable states of plant to use it in dynamic modeling. However shifting same type of cluster and not investigating the scatter of inputs may cause ENN to classify patterns poorly. Further discussions about the proposed improvements to ENN are in Chapter 2.2.

2.2.1 Extension Theory

Extension Theory (ET) was proposed by Cai [4] to solve contradictory problems in 1983. Contradictory problems can not be solved by given conditions until a proper transformation of the conditions is implemented. In engineering applications, Laplace transformation for example, is used to make a problem solvable by transforming it into another domain. ET deals with these incompatible or contradictory problems and re-formalizes the concepts to give a solution. There are similarities between Fuzzy Set Theory (FST) and ET. In [22], FST is explained as a generalization of well known standard sets to extend applications field. In standard set applications, transfer function shows if an element belongs to a class or not. FST extends this set to [0,1], showing the degree an element belongs to the class. In [6], it is explained that ET extends FST from [0,1] to $[-\infty,\infty]$ and therefore, this situation leads up with an element, belonging to each extension set to a different degree. However, although ET works on the degree of an element belonging to a class like FST, it also considers the degree of not belonging to a class.

The membership function of ET can be defined by $K(x)$ where x is an element, $K(x)$ shows the degree an element belongs to a class. In the case of $K(x)$ <0, it describes the degree of x not belonging to a class. The region $0 < K(x) < 1$ corresponds to fuzzy set theory, which implies the degree of x belonging to a class. When $K(x) < -1$, x does not have any possibility to belong to a class. When $-1 < K(x) < 0$, x has a still possibility to belong to a class if that class is adjusted. These regions are shown in 2.2.1.

ET is composed of Matter-Element Theory and Extension Set Theory. To understand the aspects of Extension Theory, these two pillars should be analyzed individually.

Matter-Element Theory

Classical mathematics are familiar with quantity and forms of objects. Whereas, Matter-Element Theory(MET) considers both quality and quantity of an object. In real world, things are represented by their quantity and quality. Therefore, MET deals with contradictory problems' quality and quantity. ET considers transforming these contradictory problems to matter-element models and analyze them with their quality and quantitative change.

$$R = (N, C, V) \tag{2.15}$$

where in matter R, N is the name or type, C is its characteristic and V is the corresponding value for the characteristic. An element can have many characteristics. Therefore, that many characteristics and corresponding values are identified. An example in (2.16) is given about multiple characteristics. Equation (2.16) shows that Yusuf's height is 178 cm and his weight is 98 kg. These characteristics form a set. Matter element is used in extension sets via correlation functions to determine membership degree of a pattern which is randomly taken from whole space with these sets. Correlation functions and extension sets are described next in 2.2.1.

$$R = \begin{bmatrix} Yusuf, & Height, & 178cm \\ & Weight, & 98kg \end{bmatrix} \tag{2.16}$$

Extension Set Theory

Let U be a space of objects and x be an element of this space as shown in (2.17).

$$A = \{(x, y) | x \in U, y = K(x)\} \tag{2.17}$$

where A is the extension space. $K(x)$ maps patterns x taken from U space to a membership grade between $[-\infty,\infty]$. Extension set can be shown in three regions

(2.18).

$$\begin{aligned} A^{+} &= \{(x,y) \| x \in U, y = K(x) \geq 0\}, \\ A^{0} &= \{(x,y) \| x \in U, y = K(x) = 0\}, \\ A^{-} &= \{(x,y) \| x \in U, y = K(x) \leq 0\} \end{aligned} \tag{2.18}$$

where A^{+} is the positive region which represents the degree of x belonging to a class. Whereas A^{-} shows the negative region which represents the degree of x not belonging to a class. A^{0} is the zero boundary region, in this region $x \in A^{+}$ and $x \in A^{-}$.

Let X_{in} and X_{out} be real number intervals between (a,b) and (c,d), where $X_{in} \subset X_{out}$. X_{in} and X_{out} are called concerned and neighborhood domains, respectively. The correlation function can be summarized as (2.19). The correlation function is used for calculating the membership degree between x and X_{in}, X_{out}.

$$\begin{aligned} \rho(x, X_{in}) &= \left| x - \frac{a+b}{2} \right| - \frac{b-a}{2} \\ \rho(x, X_{out}) &= \left| x - \frac{c+d}{2} \right| - \frac{d-c}{2} \end{aligned} \tag{2.19}$$

The extended correlation function's (2.20) shape is shown in Fig. 2.2.1. For further details about these regions please refer to the last part of 2.2.1.

$$K(x) = \left\{ \begin{array}{cc} -\rho(x, X_{in}) & x \in X_{in} \\ \dfrac{\rho(x, X_{in})}{\rho(x, X_{out}) - \rho(x, X_{in})} & x \notin X_{in} \end{array} \right\} \tag{2.20}$$

In [22], extension theory is used in misfire FD of gasoline engines and faults in the system are successfully found.

In 2.2.2, ET and Neural Networks are combined to maintain a hybrid method. The aim of creating a hybrid method is to enhance classification efficiency and accuracy. Extension Neural Network(ENN) is briefly explained in 2.2.2 and implemented in state of charge estimation of a lead acid battery which is explained in the following chapter.

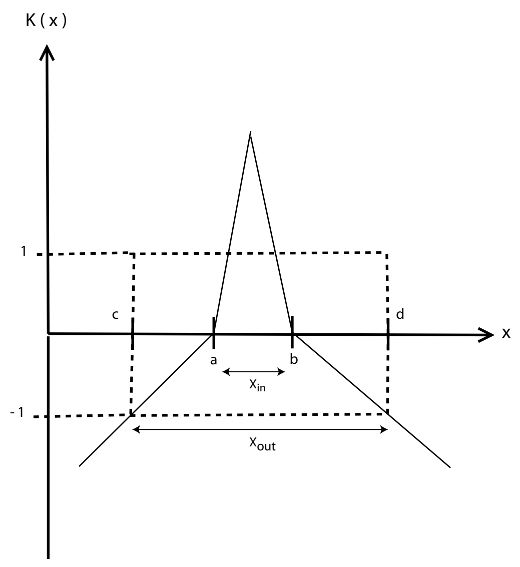


Figure 2.2.1: Extended Correlation Function

2.2.2 Extension Neural Networks

Extension Neural Network is a hybrid method for classification of patterns with the help of NN and ET concepts. While ET makes distance measurement for classification, NN is used for its fast and adaptive learning capability. ENN was first proposed in 2003 by Wang [20]. It implements an appropriate classification method for features which are defined in a range. In [20], it is shown that ENN gives better or equal accuracy and less memory consumption in classification than Multilayer Perceptron NN, Probabilistic NN, Learning Vector Quantization and Counter Propagation Neural Networks.

ENN is used in many areas for classification. Monitoring condition of machinery, which follows the parameters of the machinery, classifies them and makes failure detection [23]. In [19], ENN approach is implemented for classification of brain MRI data, specifically tissue classification. Another approach is implemented in [8], which deals with fault recognition in automotive engine. Ignition and oxygen sensor malfunction faults are classified with high accuracy. Also [7], is concerned with the state of charge estimation in Lead-Acid Batteries. The purpose of this thesis and [7] are similar. In this thesis, Improved ENN is proposed and used rather than ENN and this is the main difference with [7].

Figure 2.2.3 shows an illustration of ENN. The nodes in the output layer are a representation of the outputs of the nodes in the input layer through a set of weights. The total number of inputs and outputs are expressed by n and n_c, respectively. The total number of instances is N_p. Data-points are denoted by x_{ij}^p, meaning i^{th} instance ($i = 1, ..., N_p$), j^{th} characteristic value ($j = 1, ..., n$) corresponding to material p. Here x_{ij}^p becomes the input and o_{ik} the extension neural network output node k for instance i. Between input x_{ij}^p and output o_{ik}, there are two sets of weights denoted by w_{jk}^U and w_{jk}^L. These two weights are determined by searching the lower and upper boundaries of j^{th} input of the training data. The upper boundary w_{jk}^U, is found by searching the maximum value for j^{th} input node out of all j^{th} input instances. And the lower boundary w_{jk}^L is determined vice versa. These two weights are adjusted in each iteration to make classification more accurate and efficient. The nodes o_{ik} in output layer are the indicators of which class an input vector belongs to. If i^{th} instance's inputs correspond to the class k, then in the output layer o_{ik} should be smaller than the other output nodes. This situation denotes that the i^{th} instance's inputs' distance to k^{th} class, is smaller than the other classes. The transfer function of Fig. 2.2.3 is shown in (2.22), where k^* is the index of the estimated class. Figure 2.2.2 represents (2.21). Weights w_{kj}^U and w_{kj}^L are the points where $ED_{ik}(x) = 1$. Further details about extension distance (ED), shown in (2.21) and adjustment of weights are discussed in the following section.

$$ED_{ik} = \sum_{j=0}^{n} \left(\frac{|x_{ij}^p - z_{kj}| - \frac{w_{kj}^U - w_{kj}^L}{2}}{|\frac{w_{kj}^U - w_{kj}^L}{2}|} + 1 \right) \tag{2.21}$$

$$k = 1, 2, ..., n_c$$

$$o_{ik} \equiv ED_{ik} \tag{2.22}$$

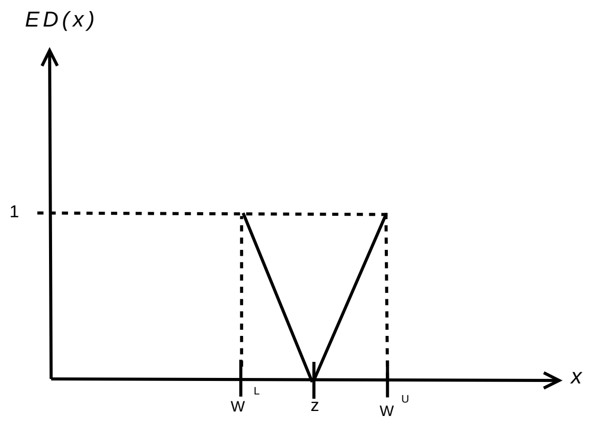


Figure 2.2.2: Extension Distance

$$k^* = arg \min_k (o_{ik}) \tag{2.23}$$

Extension Neural Networks Learning Algorithm

The architecture of ENN in Fig. 2.2.3 is expressed by matter-element model as shown in (2.24). ENN is a supervised learning method which provides inferring a function from supervised training data. Training data is the composition of input and desired output pairs.

$$R_k = \begin{bmatrix} class_k, & c_1, V_{k1} \\ & c_2, V_{k2} \\ & \ldots\ldots \\ & \ldots\ldots \\ & c_n, V_{kn} \end{bmatrix} k = 1, 2, ..., n_c \tag{2.24}$$

In (2.24), $class_k$ is the name of the k^{th} class. The symbols c_1 to c_n represent the characteristic. V_{kj}, denotes the range for the characteristic c_j of $class_k$. The range value V_{kj} is determined by w^U_{kj} and w^L_{kj}. Next we continue with how to find weights w^U and w^L.

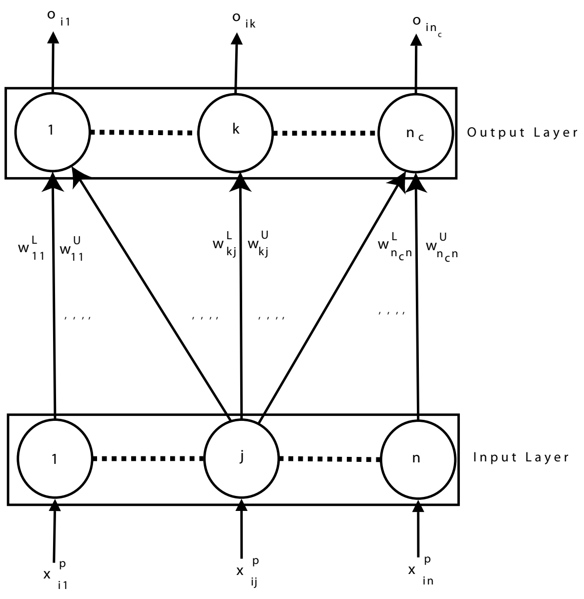


Figure 2.2.3: Extension Neural Network Architecture

Learning proceeds as follows: At the initial step, weights are determined by using (2.25), searching the maximum and minimum j^{th} input for k^{th} class among all instances to find w^U_{kj} and w^L_{kj}, respectively.

$$\begin{aligned} & w^U_{kj} = \max_i \{x^k_{ij}\} \\ & w^L_{kj} = \min_i \{x^k_{ij}\} \\ & i = 1, ..., N_p \\ & k = 1, ..., n_c \\ & j = 1, 2, ..., n \end{aligned} \tag{2.25}$$

$V_{kj} = [w^L_{kj}, w^U_{kj}]$ is determined initially by (2.25), therefore, it depends on training data.

After maintaining matter-element model, center of clusters are determined by V_{kj} as shown in (2.26). Note that clusters are the representers of classes. Each class has the same number of clusters as the number of inputs.

$$\begin{aligned} &Z_k = \{z_{k1}, z_{k2}, ..., z_{kn}\} \\ &z_{kj} = \frac{w_{kj}^U + w_{kj}^L}{2} \\ &k = 1, 2, ..., n_c \\ &j = 1, 2, ..., n \end{aligned} \tag{2.26}$$

After initial steps are done, if these initial values are not sufficient for classification, weights and center of clusters should be updated to classify more accurately. For calculating the accuracy of classification, learning performance rate equation is used which is shown in (2.27).

$$E_\tau = \frac{N_m}{N_p} \tag{2.27}$$

N_m is the total errorously classified instances and N_p is the total number of instances. Update of weights and center of clusters are proceeded until the learning performance rate is low enough. While learning, all the instances should be used. In every iteration, an instance should be chosen randomly among the training data. In (2.28), for training, i^{th} pattern, whose desired outcome should be p is chosen randomly out of the training set.

$$\begin{aligned} &X_i^p = \{x_{i1}^p, x_{i2}^p, ..., x_{in}^p\} \\ &1 \leq p \leq n_c \end{aligned} \tag{2.28}$$

In the next step, ED method is used to determine the class. X_i^p is the input vector and the vector elements are the characteristics' values. The distance between a training instance X_i^p's data-points x_{ij}^p and every cluster is calculated. In (2.21), the distance between randomly taken instance's input and k^{th} class is calculated. After each input's distance is calculated for a certain class, the distances are summed up to find the total distance. This procedure should be done for every class. The class which gives the minimum distance is the class that the ENN classifies the instance to. However instance's desired outcome is p (2.21). If the minimum ED shows that $k^* = p$, then no update is needed. If $k^* \neq p$, then update is needed to make classification more accurate.

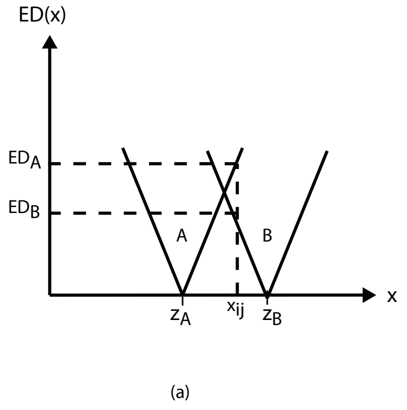
In training phase, if $k^* \neq p$, the separator is shifted according to the closeness of inputs to the cluster centers. Amount of shift is directly proportional to the distance. The mis-classified class's separator k^* is shifted away from the instance's inputs while the desired class's separator p is shifted near to them as formulated in (2.29) and (2.30). Center of clusters and weights are both modified.

$$\begin{aligned} z_{pj}^{new} &= z_{pj}^{old} + \eta(x_{ij}^{p} - z_{pj}^{old}) \\ z_{k^*j}^{new} &= z_{k^*j}^{old} - \eta(x_{ij}^{p} - z_{k^*j}^{old}) \end{aligned} \tag{2.29}$$

$$\begin{aligned} w_{pj}^{L(new)} &= w_{pj}^{L(old)} + \eta(x_{ij}^{p} - z_{pj}^{old}) \\ w_{pj}^{U(new)} &= w_{pj}^{U(old)} + \eta(x_{ij}^{p} - z_{pj}^{old}) \\ w_{k^*j}^{L(new)} &= w_{k^*j}^{L(old)} - \eta(x_{ij}^{p} - z_{k^*j}^{old}) \\ w_{k^*j}^{U(new)} &= w_{k^*j}^{U(old)} - \eta(x_{ij}^{p} - z_{k^*j}^{old}) \end{aligned} \tag{2.30}$$

where η, is the learning rate.

An update example is given in Fig. 2.2.4 which has a total number of two clusters. Although the instance X_i behaves as if it belongs to class A, (2.21) shows that X_i belongs to the class B. Therefore, cluster A and B are updated with the formulas (2.29) and (2.30) as shown in Fig. 2.2.4(b) so that (2.21) gives $ED_A < ED_B$. Note that training continues until (2.27) converges to an acceptable value.



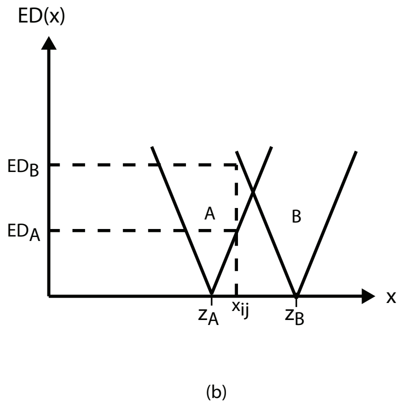


Figure 2.2.4: Updating Separators:(a)Before Update; (b)After Update

3 IMPROVED EXTENSION NEURAL NETWORKS

We propose Improved Extension Neural Network (IENN) in this thesis to improve the performance of ENN in classifying various patterns. Some patterns may need to be separated by a separator which has sharper boundaries whereas others may need to be separated by a wide boundary. This issue depends on the scatter of pattern's data-points to the space. In this situation, because ENN represents every pattern using the same type of separator, it can not answer the previously explained circumstance. For example, training data with few extreme outliers but low variance may be incorrectly represented by a wide separator.

Figure 3.0.1 shows how ENN classifies given patterns. After updating classes as illustrated in Fig. 3.0.1(b), while class B includes two patterns from class A, it leaves out the patterns which belong to class B, to class C because of just shifting the separator. This happens due to the insensitivity to scatter of patterns. This type of mis-classification issues, decrease the classification performance. Due to such problems IENN method is proposed in this thesis.

IENN is similar to ENN. In IENN, the center of separator is not shifted; center of cluster is selected at the mean of the instances and kept the same whereas the arms of the separators are moved according to the scatter of data-points. If training instances' characteristic values have low variance, arms of the separators get narrower and vice versa. According to the performance of the IENN, separators are implemented as a linear function (3.1) or non-linear(3.2) function as shown in Fig. 3.0.2. Separator

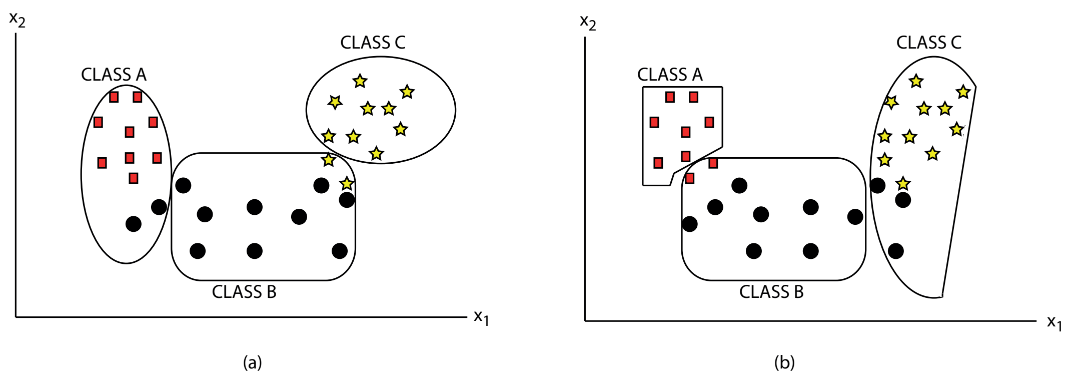


Figure 3.0.1: Updating Separators:(a) Recent Class View;(b) After Few Iterations Class View

as defined by Fig. 3.0.2 represents a cluster of data belonging to a class. Note that iterations continue until (2.27) converges to an acceptable value as implemented in ENN.

$$\begin{aligned}
&SL_{Ujk}(x) = a_{Ujk}x + b_{Ujk}\\
&SL_{Ljk}(x) = a_{Ljk}x + b_{Ljk}\\
&k = 1, 2, ..., n_c\\
&j = 1, 2, ..., n
\end{aligned} \tag{3.1}$$

$$\begin{aligned}
&SQ_{Ujk}(x) = a_{Ujk}x^2 + b_{Ujk}x + c_{Ujk}\\
&SQ_{Ljk}(x) = a_{Ljk}x^2 + b_{Ljk}x + c_{Ljk}\\
&k = 1, 2, ..., n_c\\
&j = 1, 2, ..., n
\end{aligned} \tag{3.2}$$

where, the upper and lower sides of the linear and non-linear separators are defined with different parameters: a_{Ujk}, b_{Ujk}, c_{Ujk} and a_{Ljk}, b_{Ljk}, c_{Ljk} respectively. For linear separator, upper part of linear separator SL_U is defined by the points (w^U_{kj},1), (z_{kj},0) and lower part of SL_L, by (w^L_{kj},1), (z_kj,0). In non-linear separator case, the upper and lower parts SQ_U, SQ_L are defined similar to linear separator, and an additional point (3.3) is given which interprets that SQ_U and SQ_L derivative at $x = z_{kj}$ is zero.

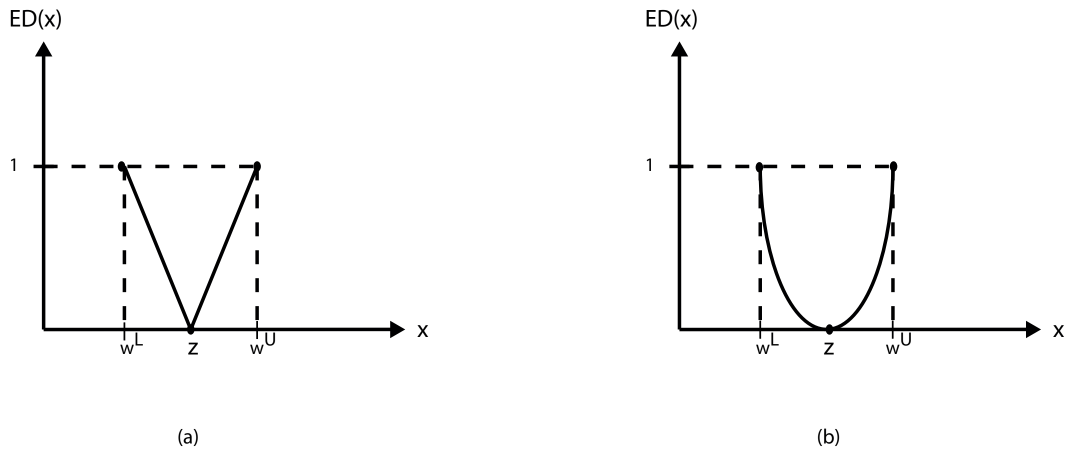


Figure 3.0.2: (a) Linear Separator (b) Non-Linear Separator

Therefore, via using these points coefficients of the separators can be determined.

$$\frac{dy(z_{kj})}{dx} = 2a_{Ujk}z_{kj} + b_{Ujk} = 0 \tag{3.3}$$

Initial weight estimation is kept the same as in (2.25). Center of cluster calculation is changed from (2.26) to (3.4), where z_{kj} is the mean of the training data for j^{th} input for k^{th} class. n_k is the number of training instances for class k. During the update stage, center of cluster is not updated because the training data does not change, therefore mean of classes do not change.

$$\begin{aligned}
&Z_k = \{z_{k1}, z_{k2}, ..., z_{kn}\} \\
&z_{kj} = \frac{1}{n_k}\sum_{i=1}^{n_k} x_{ij}^k \\
&k = 1, 2, ..., n_c \\
&j = 1, 2, ..., n
\end{aligned} \tag{3.4}$$

Update is only done to weights as summarized in (2.30). If the given instance input is closer to lower weight, lower weight is modified, else upper weight is modified. By doing weight update, linear or non-linear separator gets narrower to diverge from a mis-classified data-point or gets wider to classify the data-point within the desired cluster. An example is illustrated about non-linear separator update and an update with ENN classifier in Fig. 3.0.3.

Calculation of extension distance is different than (2.21). Total distance calculated to a class k for instance i via linear separators is indicated with improved extension distance linear value $IEDL_{ik}$, and quadratic is $IEDQ_{ik}$. Total distance is calculated by using (3.6) or (3.8), depending on the used separator type. The placement of x_{ij} based on the center of cluster z_{kj} indicates if the data-point is located to the right or left side of the separator in (3.5) or (3.7), depending on the used separator type. Class k^*, which has least distance calculated is selected as X_i's class (2.23).

$$IED_L(x_{ij}, k) = \left\{ \begin{matrix} SL_{Ujk}(x_{ij}) & x_{ij} > z_{kj} \\ SL_{Ljk}(x_{ij}) & x_{ij} < z_{kj} \end{matrix} \right\} \tag{3.5}$$

$$IEDL_{ik} = \sum_{j=1}^{n} IED_L(x_{ij}, k) \tag{3.6}$$

$$IED_Q(x_{ij}, k) = \left\{ \begin{matrix} SQ_{Ujk}(x_{ij}) & x_{ij} > z_{kj} \\ SQ_{Ljk}(x_{ij}) & x_{ij} < z_{kj} \end{matrix} \right\} \tag{3.7}$$

$$IEDQ_{ik} = \sum_{j=1}^{n} IED_Q(x_{ij}, k) \tag{3.8}$$

Comparing the performance of linear separator with quadratic separator is not a trivial task. Linear separator is more suitable for data with large variance while quadratic separator is not. Total calculated distance is the sum of j^{th} input to the k^{th} separator so that if system has one input, linear and quadratic separator would give the same results in classifying although, total calculated distance is different as illustrated in Fig. 3.0.4 (a). Both intersection points of linear separators and non-linear separators give the same result. Therefore, distance of x_{i1} in any case gives $IED_Q(x_{i1}, A) > IEDQ_Q(x_{i1}, B)$ or $IED_L(x_{i1}, A) > IED_L(x_{i1}, B)$. Data-point x_{i1}'s distance to the center of a cluster z_{k1} is projected to $IED_L(x_{ij}, k)$ as a linear distance in linear separator, whereas it is projected to $IED_Q(x_{ij}, k)$ as a quadratic distance in non-linear separator. If a data-point x_{i2} is far out from cluster B as illustrated in Fig. 3.0.4 (b), $IEDQ(x_{i2}, B) > IEDL(x_{i2}, B)$. In Fig. 3.0.4 (b) assume that second input has high variance through cluster B and a data point x_{i2} exists where i^{th} instance belongs to cluster B. In non-linear case, because of the quadratic increasing of the separator, $IED_Q(x_{i1}, A) + IED_Q(x_{i2}, A) < IED_Q(x_{i1}, B) + IED_Q(x_{i2}, B)$, therefore instance x_i behaves as if it belongs to the class A. Whereas in linear separator case, $IED_L(x_{i1}, A) + IED_L(x_{i2}, A) > IED_L(x_{i1}, B) + IED_L(x_{i2}, B)$ which shows x_i belongs to the class B. The reason is; by defining cluster B as a linear separator, it is defined as second input has high variance among cluster B. Therefore, a data-point located outside of the weight point of cluster B gives less IED_L values than quadratic separator gives IED_Q. Eventually, linear separators must be used for high variance input data sets and non-linear separators for vice versa.

In the example illustrated in Fig. 3.0.3 (a) and (c), ENN and IENN clustering updates are shown respecitvely. Note that system has only one input. In Fig. 3.0.3 (a), although x_{51} is classified correctly, x_{11} is misclassified in to cluster B. After the update is done as illustrated in (b), although x_{11} is classified correctly, this time x_{51} is misclassified. To classify x_{51} to cluster C, doing more updates might be helpful but there is a possibility that, x_{11} can be misclassified again. Whereas non-linear IENN update, illustrated in (c) classified both data-points correctly.

Learning rate η, used in ENN update (2.30), changes with respect to iteration number (3.9), where x represents the current iteration number, E is the maximum number of iterations. Learning rate η decreases with the increasing number of iterations to a minimum set point. At first iterations, class boundaries oscillate much and get influenced by every data-point. As learning of relationship proceedes, η is reduced. This idea provides learning to be faster and reduces influence of noisy input data. An example of the process of η is shown in Fig. 3.0.5 where η is defined as 0.1 at $x = 1$ and 0.001 at $x = 2000$.

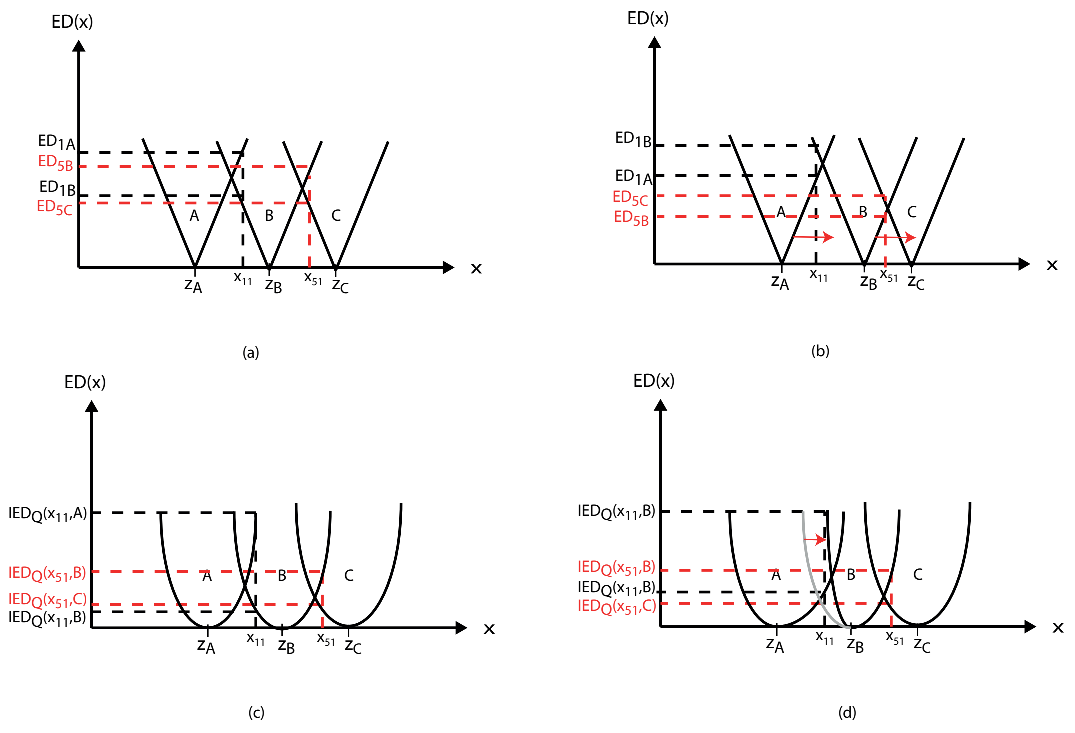


Figure 3.0.3: (a) ENN Clustering with one input (b) ENN Update Clustering with one input
(c)IENN Non-Linear Clustering with one input (d)IENN Update with one input

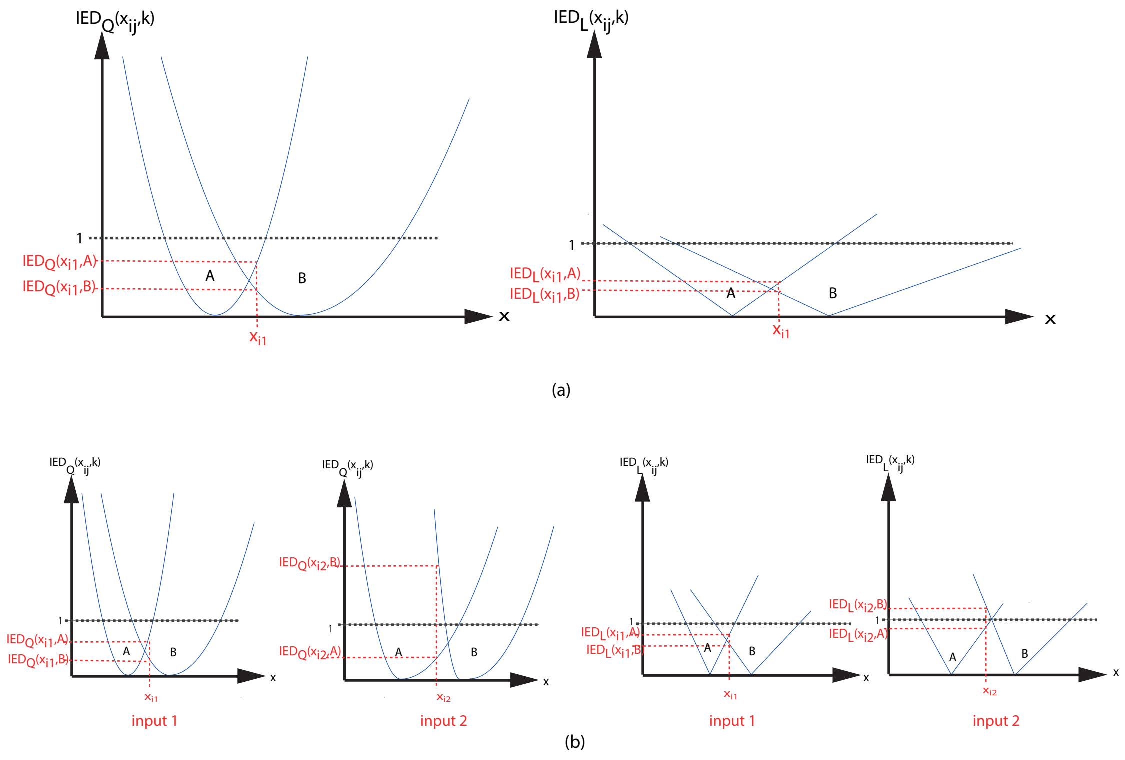


Figure 3.0.4: (a) Linear and Non-Linear separator with One Input (b)Linear and Non-Linear separator with Two Inputs

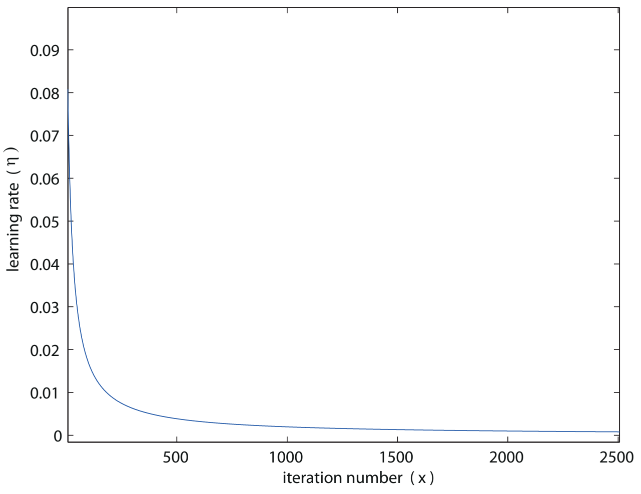


Figure 3.0.5: Learning Rate vs. Iteration Number

$$\begin{aligned} &\eta(x) = \frac{1}{ax+b} \\ &1 < x < E \end{aligned} \tag{3.9}$$

3.1 Using A Hybrid Approach For Classification

Due to the variation difference of data-points on clusters as discussed earlier, a hybrid approach can be used to classify input sets with large variance by linear separator and vice versa with non-linear separator. For cluster k, each input j's variance is

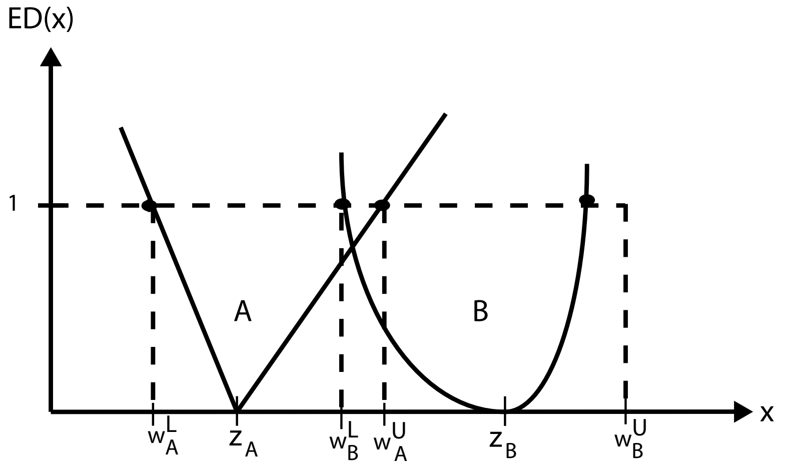


Figure 3.1.1: Hybrid Classification where input data set for class A has large variance whereas B is narrow

calculated with the formula (3.10).

$$\begin{aligned} &\mu_{kj} = z_{kj} \\ &\sigma_{kj}^2 = \frac{1}{n_k}\sum_{i=1}^{n_k}(x_{ij}^k - \mu_{kj})^2 \\ &k = 1, 2, ..., n_c \\ &j = 1, 2, ..., n \end{aligned} \tag{3.10}$$

To decide j^{th} input separator type for cluster k, a threshold parameter α_{thr} should be chosen from the interval $(0, 1)$. This threshold value is mapped to some point $\sigma_{thr_{kj}}^2$ between the maximum and the minimum variance of σ_{kj}^2 as shown in (3.11). If σ_{kj}^2 is smaller than the $\sigma_{thr_{kj}}^2$, non-linear separator is used for k^{th} cluster for j^{th} input and vice versa is classified via linear separator. Figure 3.1.1 illustrates A and B clusters where j^{th} input with large and small variance, respectively.

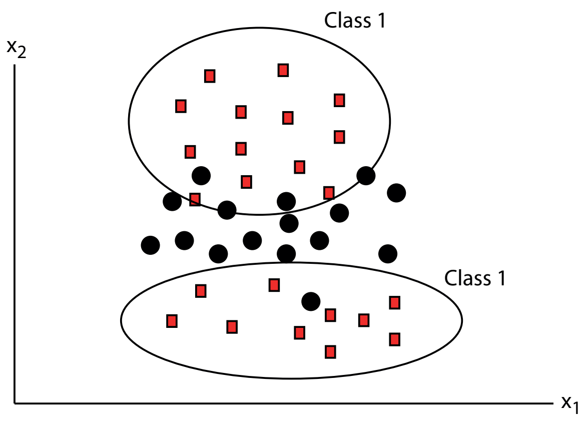


Figure 3.1.2: Measurements collected more than one region

$$
\begin{aligned}
&\sigma^2_{max_j} = \max_k\{\sigma^2_{kj}\} \\
&\sigma^2_{min_j} = \min_k\{\sigma^2_{kj}\} \\
&\sigma^2_{thr_j} = (\sigma^2_{max_j} - \sigma^2_{min_j})\alpha_{thr} + \sigma^2_{min_j} \\
&k = 1, 2, ..., n_c \\
&j = 1, 2, ..., n \\
&\alpha_{thr} \in (0, 1)
\end{aligned}
\tag{3.11}
$$

Note that both ENN and IENN methods give unreliable results if input data sets are divided into more than one region for the same class as illustrated in Fig. 3.1.2. However, most dynamic systems have state and output spaces which progress continuously, therefore such divisions do not occur in normal cases.

3.1.1 Validation

Validation is a technique for testing the generalization of a statistical analysis on an independent data set and cross validation is the commonly used validation method. Cross validation involves partitioning the data set into training set and a validation set. A model with one or more unknown parameters is trained with training data set and generalization performance is tested on validation data set. This procedure is done to use suitable parameters.

10-Fold Cross Validation

In testing the generalization performance of classification techniques, 10-fold cross validation is used. Total data set is divided into ten partitions. One partition is assigned to validation set and the remaining partitions are assigned to training set. Each classification algorithm with different parameters is trained with the training set partitions and tested on validation set partition. This procedure is proceeded iteratively until every partition is assigned to validation set. In each iteration, error rate for the validation set is assigned to E_i as shown in (3.12) where i is the iteration number, V_e is the number of mis-classified patterns and V_n is the total number of patterns in validation set.

$$E_i = \frac{V_e}{V_n} \tag{3.12}$$

After ten iterations, total generalization performance E is found by calculating the mean error as shown in (3.13).

$$E = \frac{1}{10}\sum_{i=1}^{10} E_i \tag{3.13}$$

4 RECHARGEABLE LEAD ACID BATTERY PRINCIPLES AND MEASUREMENT METHODS

4.1 Lead Acid Battery Principles of Operation

A battery is a device which converts chemical energy to electrical energy. The oldest widespread rechargeable battery technology is the Lead-acid battery. There are two types of lead-acid batteries: Valve Regulated Lead Acid (VRLA) battery and Lead Acid Wet Cell (LAWC) battery. VRLA batteries use immobilized sulfuric acid electrolyte in gel form. This opportunity provides reducing the chance of leakage of electrolyte. In this research, (LAWC) battery is used because of their widely usage on cars and buses.

LAWC battery is an integral component of a vehicle's electrical network. It is also called starter battery because it is used to start combustion engine of a vehicle. By using this property in early 1900's, the need for hand-cranking of engines was eliminated. After that year automobile battery is used for engine starting, ignition and vehicle lighting and as a buffer to store instantaneous power generated by the alternator. Therefore, it may be said that main purpose of the battery in vehicles is to start engine. After starting engine, the secondary role of providing current to consumers is processed. A vehicle electrical network is shown in Fig. 4.1.1. The battery is placed between the alternator and consumers of electricity such as starter motor, lights, radio, air condition etc. Whenever the alternator is producing more

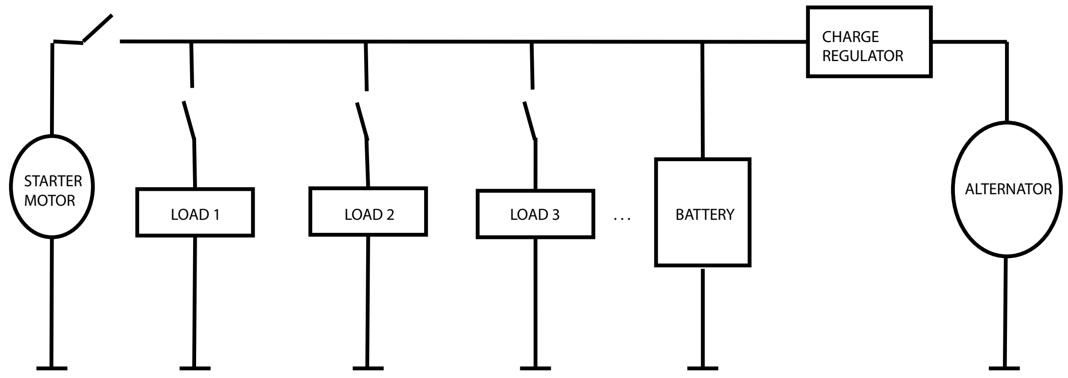


Figure 4.1.1: Vehicle Electrical Network

power than needed, remaining current is used to charge the battery. Lead-Acid cells contain a positive electrode (anode) made of lead oxide (PbO_2), negative electrode (cathode) made of pure lead (Pb) and the cathode and anode is immersed in sulfuric acid (H_2SO_4) electrolyte.

Electrochemical energy is stored in active materials which are bonded to anode and cathode grids. When a circuit is connected to the grids through the terminals of battery, according to the flow of the charge direction, i.e. charging or discharging electrons are transfered through one active material to other active material as their chemical composition change. The electrolyte is responsible of transfer of ions between these active materials. The chemical reactions during discharge can be described as follows:

Positive electrode:

$$PbO_2 + HSO_4^- + 3H^+ + e^- \rightarrow PbSO_4 + 2H_2O \tag{4.1}$$

Negative electrode:

$$Pb + HSO_4^- \rightarrow PbSO_4 + H^+ + 2e^- \tag{4.2}$$

Charging causes (4.1) and (4.2) occurs in the reverse direction.

As seen above, positive electrode is accepting electrons from negative electrode. In discharge operation, lead sulfate is produced and if the battery is over-charged or left standing in the discharged state for a long time, lead sulfate coats the electrodes until battery is recharged. Under these circumstances, the sulfuric acid mixture can separate into two distinct layers with the water rising to the top and acid sinking to the bottom. In the low part of the mixture, acid concentration rises and corrosion of the bottom half lead plates of battery occurs. Battery efficiency and life cycle will be reduced. In discharge chemical reaction density of sulfuric acid is reduced, because of H_2O production. With the effect of reduced density of sulfuric acid, battery efficiency becomes more bound to temperature. It is also explained in [21] that reduction of electrolyte density causes residual capacity to decrease and accordingly this effects electromotive force to decrease. In the charging case chemical reaction (1) and (2) is reversed. If battery is overcharged, it causes H_2O to decompose into hydrogen and oxygen gas. Acid fumes vaporize through vent caps and (irreversible)material loss occurs.

4.2 Battery Characteristics

4.2.1 Capacity

The capacity of a Lead Acid Battery can be explained as an amount usable electrical charge of the battery. Ampere-hour (Ah) is an expression for capacity. It is calculated as:

$$C_{max} = I^n \text{x} T \tag{4.3}$$

Equation (4.3) is called peukert's effect. I is the current measured in amperes (A), T is time in hours, n is the Peukert's number and C_{max} is the maximum capacity of the battery. The Peukert's number is available from manufacturers. This equation, gives the true capacity if the battery is discharged at 1 Amp. Capacity estimation is done by using single battery in [5]. Capacity measurement is done under different discharge currents and temperature conditions. It is seen that as the amount of current drawn from the battery increases, estimated capacity is decreased.

In [15] equivalent lead-acid battery model is generated by imposing the manufacturers' data. Output of the peukert's effect algebraic function is one of the used input to the generated battery model to predict battery terminal voltage output.

4.2.2 State Of Charge

The State of Charge (SoC) is the energy stored in the battery at a given time compared to its capacity, in percent. For different SoCs, battery acts differently. When SoC is maximum, open circuit voltage is expected to be between 12.7V - 13.0V. In an empty state, it is expected to be 10.5V. Also density of electrolyte can be used to accurately estimate SoC. In full SoC, density is 1.25 g/mL - 1.27 g/mL and in empty SoC, it is measured as 1.12 g/mL - 1.14 g/mL.

It is difficult to estimate SoC electrically, because of nonlinear dynamics of the battery. Faulty calculated SoC can lead to faulty estimates of battery model output. Much effort is being put into the research to estimate the SoC accurately and precisely. The articles [21], [14], [17], [18] mention SoC estimation, based on current integration, internal resistance and open circuit voltage. For current integration technique; if total integral of current is known since full or empty, SoC can be estimated. Due to integration errors, this technique has difficulty in practice. The formula is defined as:

$$SoC(t) = 100 \left(\frac{Q_c - \int_0^t i_d(\tau) d\tau}{Q_c} \right) \tag{4.4}$$

Where, $i_d(\tau)$ is the current drawn from battery and Q_c is the actual capacity estimated via integrating the discharge current of exactly 1A from a full charged state until terminal voltage of 10.5V is reached.

Internal resistance is known to increase as the active material in the battery decreases. Therefore as the battery discharges, internal resistance increases. When the SoC is about maximum, the gradient of internal resistance is not large. This situation, causes difficulty in estimation of SoC by only using internal resistance data. Estimation of internal resistance is briefly discussed in the next chapter.

Short circuit current is the current that is expected to flow through the battery under a load of zero ohms. The value of the short circuit current depends on the internal resistance and open circuit voltage. Short circuit current and internal resistance are inversely proportional, therefore just using short circuit current is not sufficient in estimating the SoC of the battery.

Open circuit voltage V_{oc} is also related to SoC as the SoC decreases, V_{oc} decreases accordingly. However, just using V_{oc} data is not sufficient to estimate SoC. Estimation and measuring technique is discussed in the next chapter.

In [14] and [21] an equation to model the battery is proposed; composed of internal resistance, electromotive force and temperature. It uses least square estimation technique to find unknown parameters of the equation. Proposed technique estimates SoC with an error rate around 10% - 15%. Article [17], gives an overview of SoC estimation techniques. Contrary to [14] and [21], it discusses Kalman Filter (KF), Impedance Spectroscopy (IS) and Artificial Neural Network (NN) in estimation of SoC. It concludes that IS is hard to be implemented to online SoC estimation while KF gives perspectives for high dynamic usage. And also mentions that if training data is enough, NN can be implemented to SoC estimation. [16], uses NN to estimate SoC of faulty and proper Lead-Acid Batteries. As a training data it uses capacitance parameters, internal resistance, electrical power, temperature, V_{oc} and average of voltage and current for a period of time.

In [13] dynamic behavior of dynamic batteries is analyzed by using electrochemical impedance spectroscopy. Frequency ranges are separated to show the effects of mass transport, charge transfer and capacitance between electrodes, and electromagnetic field. It is shown that SoC mainly influences the low frequency characteristics like mass transport characteristic. Therefore, by inspecting the low frequency characteristics SoC can be determined.

4.2.3 Effects of Temperature

The characteristic of the battery is highly effected by the temperature. With higher temperatures, the battery has higher capacity and V_{oc} while it has lower capacity and V_{oc} in lower temperatures. Therefore, it can be said that temperature effects on capacity have similarities with the relationship of amount of current drawn and capacity. In [5], temperature effects on capacity is also studied. Same fully charged battery is tested under different temperature environment. Under $-5°C$ temperature environment, discharging with 13 A of current shows 70 Ah of capacity. On the other hand, under $35°C$ temperature environment, discharging with 13 A of current shows 110 Ah of capacity which is significant. In [16], environment temperature is used in NN to estimate SoC accurately.

In this thesis, temperature effect is neglected. It is assumed that temperature is not below $20°$ or above $30°$. All tests are done and collected under these conditions.

4.3 Experimental Setup and Measurement Methods

In this section, brief information about the experimental setup is given. Single measurement of internal resistance R_{in}, short circuit current I_{sc} and V_{oc} are not sufficient, in estimation of SoC as explained in Section 4.2.2 and can not be simply made during operation. These three data sources are used together to create a meaningful classification results. The techniques used for the measurement and those for the calculation of R_{in}, I_{sc} and V_{oc} are also expressed in this chapter.

4.3.1 Experimental Setup

IENN presented in Section 2.2, were implemented on Matlab via using data acquired by a data acquisition system dSpace which provides Matlab and Simulink tools for

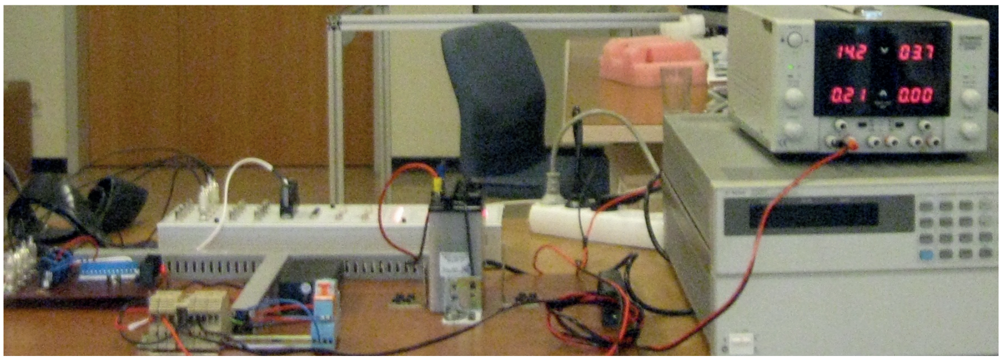


Figure 4.3.1: Battery data acquisition system experimental setup

measuring anolog signals. For data acquisition and automated supervision of the experimental setup, ControlDesk application of dSpace is used. In Fig. 4.3.1, battery charge/discharge equipment is illustrated.

Equipment Illustrated in Fig. 4.3.1 consists of:

- 5 Ah 12 V wet cell lead acid battery.
- Three HP 1146A AC/DC Current Probes which are used to measure charge, discharge and battery input current.
- Topward laboratory DC power supply, used to provide constant charge current to the battery.
- Agillent N3300A programmable electronic load, used to draw current from the battery.
- dSpace data acquisition system, used for measuring anolog signals and providing constant discharge or charge current via controlling the electronic load on RS232 channel.

Figure 4.3.1 is represented in Fig. 4.3.2 as a block diagram, where I_S and I_L are power supply and electronic load current, respectively. The direction of battery current I_B depends on the difference between I_L and I_S. Current Sensor 1, measures

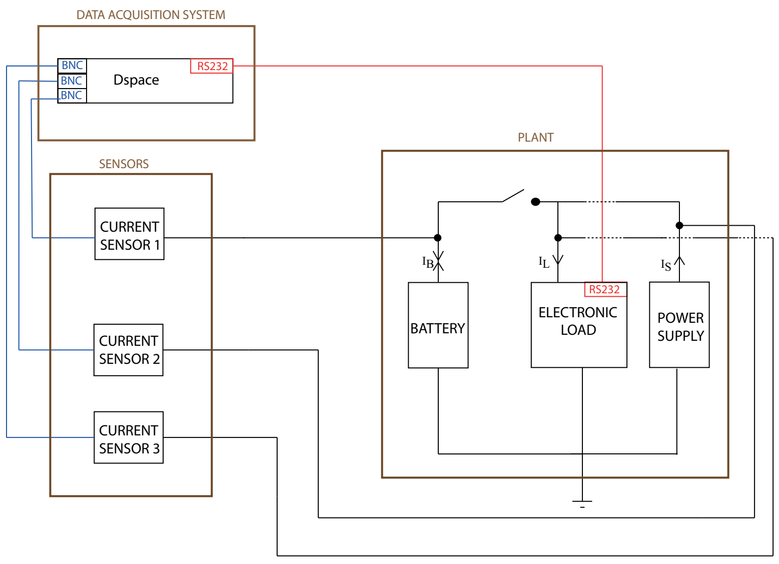


Figure 4.3.2: Experimental Setup Overview

the total current flowing through the battery. Current Sensor 2, measures the charge current flowing to the battery. Current Sensor 3, measures the current drawn by electronic load. Redundant sensors are used for cross checking for erroneous readings. Dspace controls the electronic load by sending reference load current values via RS232 communication. Since the power supply generates constant currents I_S, by adjusting the load current I_L the battery current I_B can be set desired.

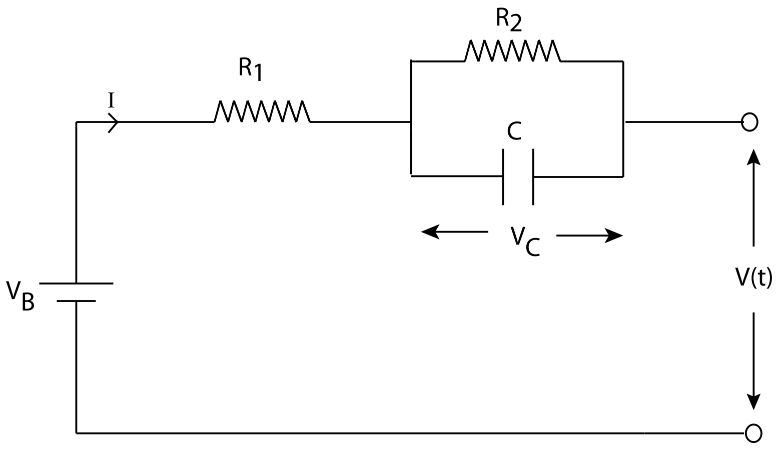


Figure 4.3.3: Circuit Model of Battery

4.3.2 Measurement Methods

Estimation of Internal Resistance

Estimation of R_{in} is done by implementing an equivalent circuit battery model. To find R_{in}, equivalent circuit model of a battery is used to get voltages and current values in [18], [21] and [14]. For R_{in} calculation, the circuit model in Fig. 4.3.3 is used. R_1 is the electrolyte resistance, including electrode resistance. R_2 is the charge transfer resistance between the electrode and electrolyte solution. C represents the static capacitance formed between electrolyte and electrode. V_B and $V(t)$ are the battery and terminal voltage, respectively. Equation (4.5) gives $V(t)$ by using equivalent circuit shown in Fig. 4.3.3. Effects of ambient temperature and aging effects are neglected.

$$V(t) = V_B - I(R_1 + R_2) + R_2 I e^{-\frac{t}{CR_1}} - V_C e^{-\frac{t}{CR_2}} \tag{4.5}$$

In (4.5), I is the current flow through the battery. When $V(t)$ is in steady state condition, (4.5) can be changed into (4.6). If two values V_1, V_2, I_2, I_2 under different loads are captured, from (4.6), (4.7) is obtained. It is assumed that the time interval is sufficiently small that R_{in} and SoC do not change significantly. Note that if $I = 0$ than $V_{oc} = V_B$.

$$V(t) = V_B - I(R_1 + R_2) \tag{4.6}$$

$$R_1 + R_2 = \frac{V_2 - V_1}{I_1 - I_2} \tag{4.7}$$

In Fig. 4.3.4, experimentally measured V_1 and V_2 values used in (4.7) are shown. As seen in the graph these two values are taken as battery reaches steady state. Similarly, I_1 and I_2 of Fig. 4.3.5 are used in (4.7). The terminal voltage, within a time window of 2 sec, is checked. If it is stabilized within that period, it is accepted as steady state voltage, like $V_1 = V(1.045)$. After taking one read at steady state, I_B is changed to find another steady state terminal voltage, $V_2 = V(1.075)$. During operation, naturally occurring load changes (current changes) can be used to the same effect.

To produce training data, a fully charged battery is discharged with 0.4 A and 0.2 A alternately, taking terminal measurements as described until battery terminal voltage of 10.5 V, which is accepted as empty state, is reached. Since SoC changes almost linearly by time under these circumstances, R_{in} can be calculated for all SoC regions by implementing (4.7). The result is illustrated in Fig. 4.3.6.

Estimation of Open Circuit Voltage

In previous sections, V_{oc} is stated as the terminal voltage of the battery when there is no load connected to it. To measure V_{oc}, battery should be left at rest for some period under no load, until terminal voltage acts in steady state. In real life conditions, while battery is under operation, resting a battery is unrealistic. Rather than measuring V_{oc}, by using (4.8), V_{oc} is calculated. This calculation is done after R_{in} is calculated.

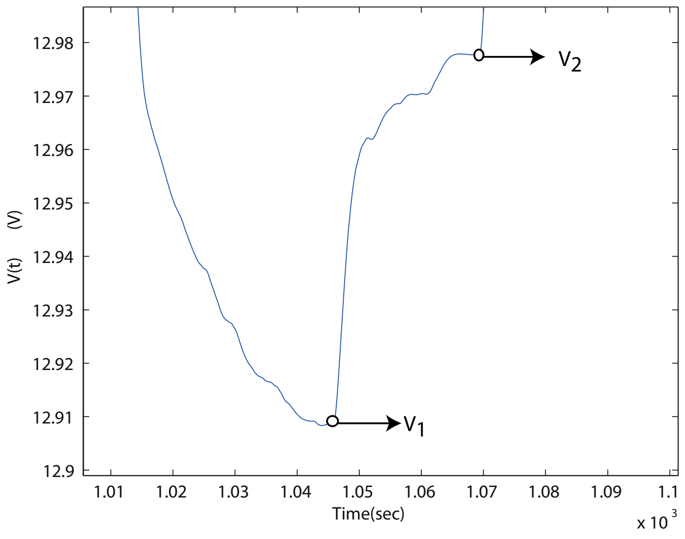


Figure 4.3.4: Voltage vs. Time Graph for making steady state voltage measurements

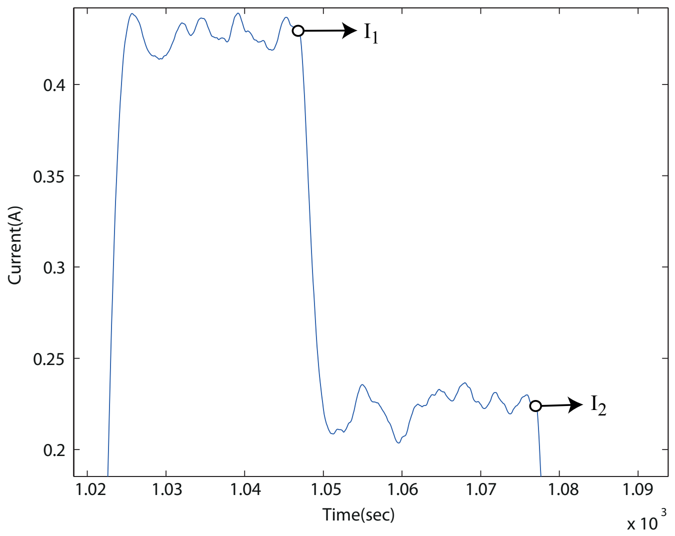


Figure 4.3.5: Current vs. Time Graph for making steady state current measurements

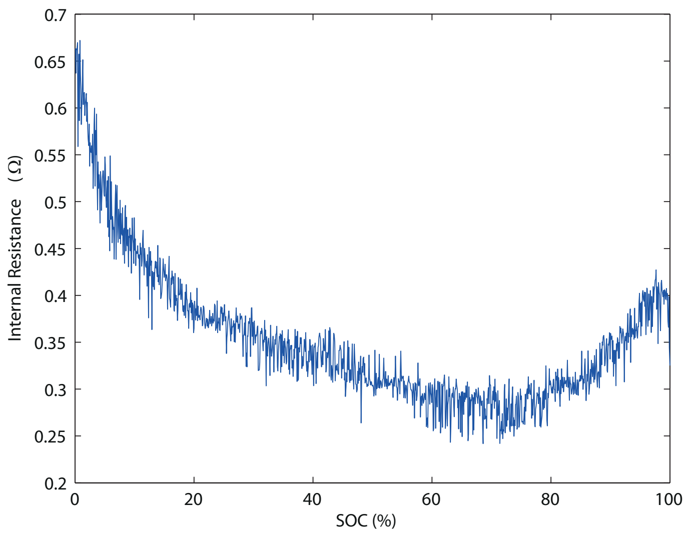


Figure 4.3.6: Internal Resistance vs. SoC

Note that V_2 and I_2 is shown in Fig. 4.3.4 and Fig. 4.3.5, respectively.

$$V_{OC} = V_2 + I_2 R_{in} \tag{4.8}$$

Notice that, V_2 is the steady state voltage which occurs by the constant current, I_1. In Fig. 4.3.7, both calculated and measured V_{oc} is compared.

Estimation of Short Circuit Current

I_{sc} is defined as the current flowing through the battery when the battery is short circuited. In (4.9), it is stated that, V_{oc} and R_{in} should be known to calculate I_{sc}.

$$I_{SC} = \frac{V_{OC}}{R_{in}} \tag{4.9}$$

In Fig. 4.3.8, change of I_{sc} with respect to SoC is illustrated.

These measurements have desired shapes as seen in literature. They act similar variations with other lead-acid batteries but because the total capacity is different according to the type of the lead-acid battery, measured values show differences.

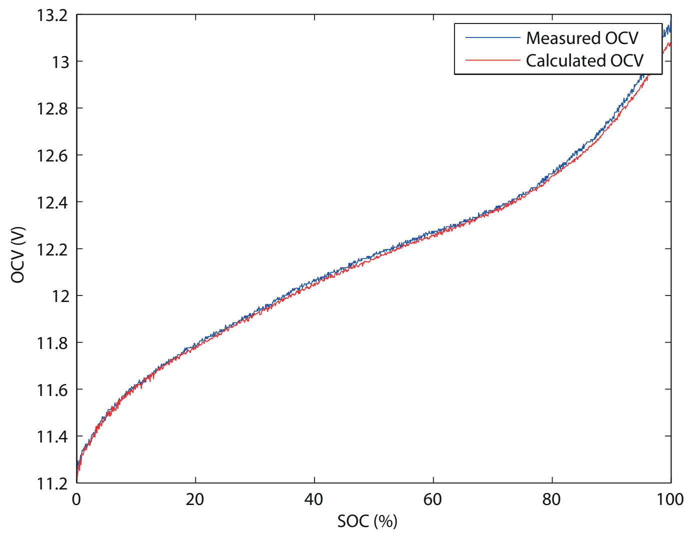


Figure 4.3.7: Measured and Calculated OCV vs. SoC

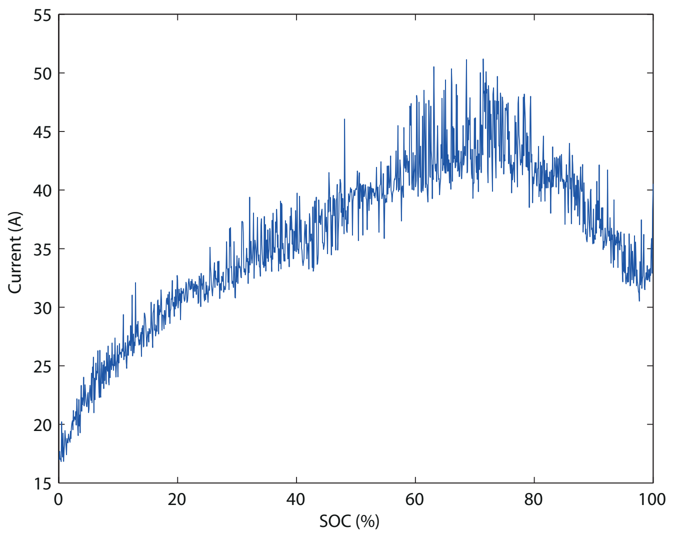


Figure 4.3.8: SCC vs. SoC

5 RESULTS

In this chapter, implementation of feed-forward NN to lead acid battery modeling and using IENN to classify SoC is discussed. The results are given involving IENN comparison with other classification methods such as ENN, and feed-forward NN.

5.1 Modeling Lead-Acid Battery via Feed-Forward Neural Network

Modeling a dynamic system via NN is discussed in Chapter 2.1. In the special case of lead-acid battery modeling the aim is to predict the terminal voltage based on the current state of charge and load in Amperes. Therefore, the input to the NN is defined as $V(t-1)$, $V(t-2)$, $I(t)$, $I(t-1)$ and SOC as illustrated in Fig. 5.1.4 (a). Since SoC cannot be directly measured, it must be estimated. To obtain data describing the performance of the lead-acid battery under wide operating conditions, it was exercised using the experimental setup with different charging and discharging currents for different SoC values. The battery started from an initial condition of $SOC = 100$, and the current integration method was used to calculate the actual SoC of the battery. The battery was first charged with 200mA for 250 sec, then discharged with 200mA, then this was repeated with 400, 600, ..., 1400 and 1500mA currents. Since charge and discharge was interchanged, SoC did not change significantly at the end. Then the battery was discharged with constant current of 1000mA to $90\%, 80\%, \ldots, 0\%$ SoC (calculated using current integration method), and the process repeated for each SoC value. However in long term operation integration errors will accumulate

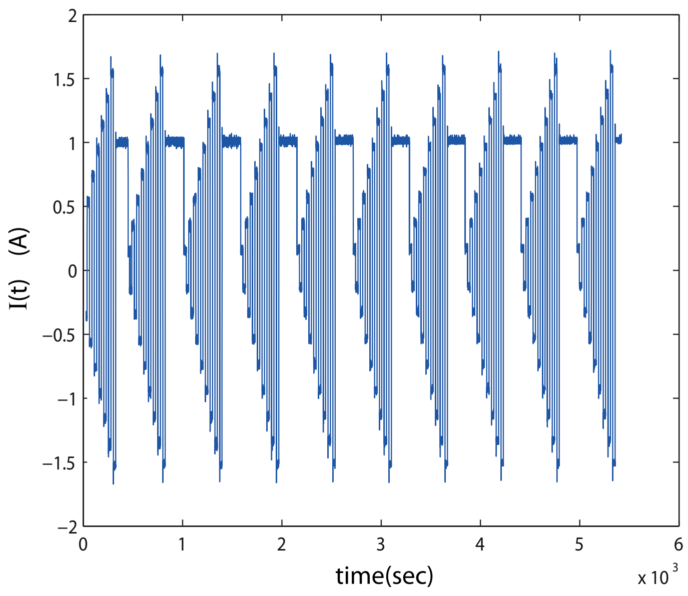


Figure 5.1.1: Input Current

and deviate from the actual SoC. Therefore in the second part of this chapter IENN is substituted to estimate SoC. The experimentally measured current, voltage and SoC values for this data acquisition process is shown in Fig. 5.1.1, Fig. 5.1.2 and Fig. 5.1.3 respectively.

In the testing stage, illustrated in Fig.5.1.4 (b), the NN was exercised with an actual battery with previously unseen data and its output compared with the actual battery, as illustrated in Fig. 5.1.5. It can be seen that it closely follows the battery terminal voltage. The maximum difference between the actual output voltage $V(t)$ and NN's output $\hat{V}(t)$ is 0.05 V.

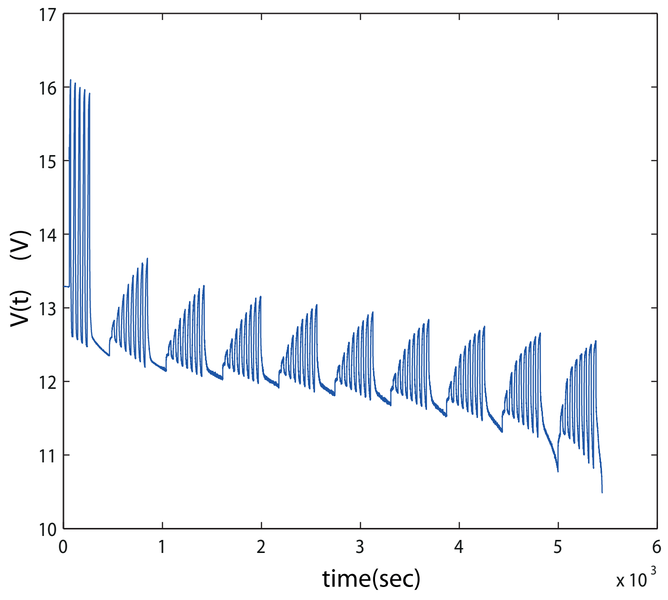


Figure 5.1.2: Output Voltage

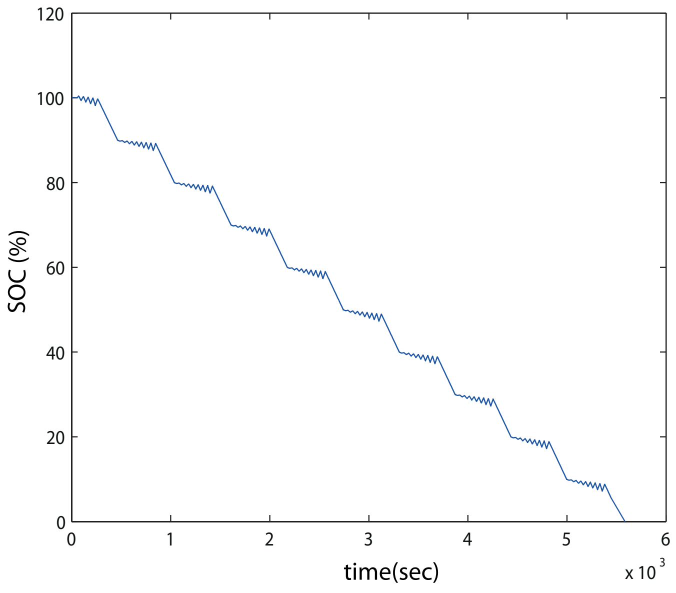


Figure 5.1.3: SoC

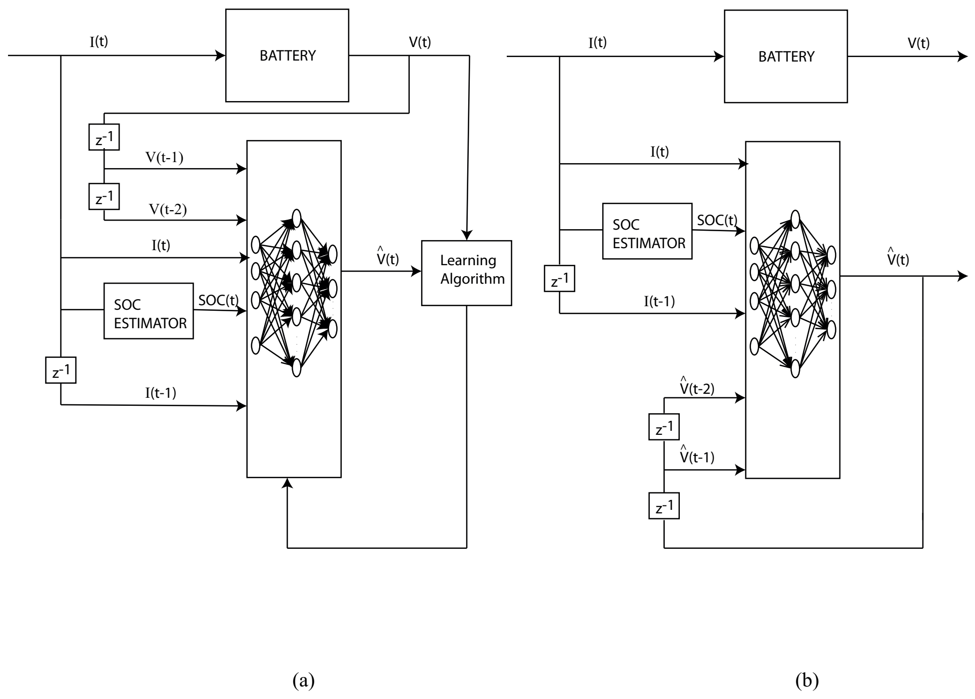


Figure 5.1.4: Battery Modeling: (a) Training Stage (b) Testing Stage

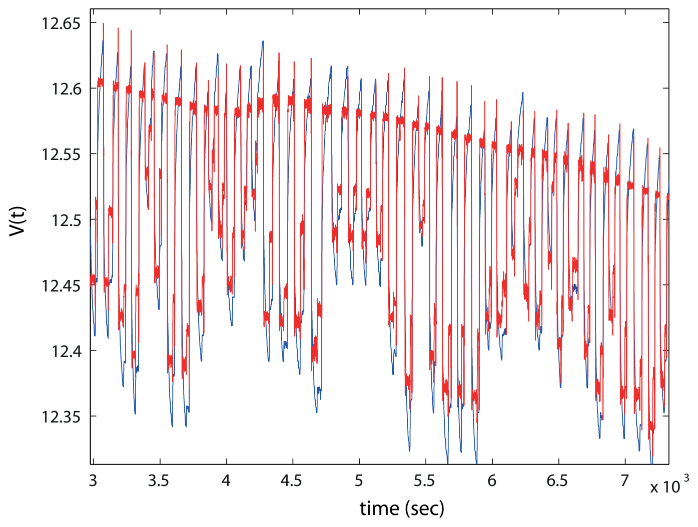


Figure 5.1.5: NN Output vs. Battery Output

Red - NN Output, Blue - Battery Output

5.2 Lead-Acid Battery State Of Charge Estimation Via Improved Extension Neural Networks

The proposed methods can be used to improve the SoC estimation that suffers from integration errors, described in the previous chapter. In this section, SoC classification will be used to compare the performances of the proposed methods and the original ENN method.

The purpose is to classify the SOC of lead-acid battery into ten classes according to the possible range of V_{oc}, I_{sc} and R_{in}. These ten classes and their definitions are listed in Table 5.1. Measurement methods of V_{oc}, I_{sc} and R_{in} were discussed in the previous chapter. To obtain data describing battery's SoC, battery is discharged from $SoC = 100$ until terminal voltage reaches 10.5V. In this interval the battery is discharged with 400mA and 200mA until $V(t)$ is accepted to be in steady state condition respectively. The process described in 4.3.2 and the experimental data obtained is used for the following tests. By using (4.7) and (4.9), R_{in} and I_{sc} are calculated. Total interval is divided into ten spaces and each space represents one class from Table 5.1.

Table 5.1: Classes of SOC

Class	Definition	Class	Definition
C1	SOC is around 90	C6	SOC is around 40
C2	SOC is around 80	C7	SOC is around 30
C3	SOC is around 70	C8	SOC is around 20
C4	SOC is around 60	C9	SOC is around 10
C5	SOC is around 50	C10	SOC is around 0

The lead acid battery data that was experimentally collected to predict SoC was used to train ENN, Linear IENN, Quadratic IENN, Hybrid IENN with low α_{thr}, Hybrid IENN with high α_{thr} and feed-forward NN. The performance of these approaches was tested via using 10-fold cross-validation method. Note that each algorithm is trained with 1000 iterations. Their performance results are compared next.

5.2.1 ENN

An ENN was trained using the method explained in Section 2.2.2. Performance of the method (3.13) is shown in Table 5.2 with respect to learning rate. The best performance was the same, about %14.1 error in the testing phase, obtained with $\eta = 0.01$ and $\eta = 0.001$, i.e, fast reduction of learning rate was not beneficial. Because of the shifting property of ENN, classifying is poor as expected. Increasing the number of iterations might help for accurate classification but in any case shifting the separators effects correctly classified instances to become mis-classified. Training with $\eta = 0.1$ causes weight updates to be high, therefore shifting amount increases and mis-classification rate increases. Consequently, to get good results from ENN classification iteration number should be kept high while η should be kept small.

Table 5.2: ENN 10-fold Cross Validation Results

Parameters	Training Error	Test Error
η=0.1	15.22%	0.1512%
η=0.01	14.13%	14.13%
η=0.001	14.13%	14.13%

5.2.2 Linear IENN

In this thesis IENN was one of the methods proposed to improve the performance of ENN. In this section, linear IENN performance will be presented. The same data set as in Sec. 5.2.1 was used to train IENN with only linear separators. The results can be seen in Table 5.3. The meaning of the first column is $\eta = 0.1$ in first iteration and decreases to 0.0001 at 2000th iteration for the first row, decreasing function is defined in 3.9.

Table 5.3: Linear IENN Classifying 10-fold Cross Validation Results

Parameters	Training Error	Test Error
(η=0.0001,2000)	3.65%	4.96%
(η=0.0001,10000)	3.00%	4.79%
(η=0.0001,30000)	3.84%	5.21%

The middle row, linear separator with $\eta = 0.0001$ at 10000th iteration gives the best fitting and testing performance. Whereas increasing η decreases the speed of learning, decreasing η causes linear separator to be effected by noisy input data and causes it to perform poorer.

Overall, a significant improvement in performance is already observed because the error rate in the test phase is now reduced to 4.8% from 14.1% of ENN.

5.2.3 Quadratic IENN

Table 5.4: Quadratic IENN 10-fold Cross Validation Results

Parameters	Training Error	Test Error
(η=0.0001,2000)	3.09%	4.63%
(η=0.0001,10000)	2.77%	4.38%
(η=0.0001,30000)	2.68%	3.88%

The same test was applied to the proposed Quadratic IENN method. The result is shown in Table 5.4. The performance is better than linear IENN, with the testing phase error value dropping as low as 3.8%. This suggests that the input data set has low variance.

5.2.4 Hybrid IENN

Table 5.5: Low α_{thr} Hybrid IENN 10-fold Cross Validation Results

Parameters	Training Error	Test Error
(η=0.0001,2000)($\alpha_{thr} = 0.2$)	3.43%	4.71%
(η=0.0001,12000)($\alpha_{thr} = 0.2$)	3.06%	4.13%
(η=0.0001,25000)($\alpha_{thr} = 0.2$)	2.97%	4.21%
(η=0.0001,2000)($\alpha_{thr} = 0.4$)	3.44%	4.63%
(η=0.0001,12000)($\alpha_{thr} = 0.4$)	3.04%	4.21%
(η=0.0001,25000)($\alpha_{thr} = 0.4$)	2.91%	4.71%
(η=0.0001,2000)($\alpha_{thr} = 0.6$)	3.37%	4.71%
(η=0.0001,12000)($\alpha_{thr} = 0.6$)	2.88%	4.46%
(η=0.0001,25000)($\alpha_{thr} = 0.6$)	2.79%	4.30%

Next, cross-validation method is implemented on the proposed hybrid IENN method to determine its performance which is expected to be better compared to the previous methods. An extra variable that must be selected here is α_{thr}. Small α_{thr} causes the system to act like a linear IENN whereas a large α_{thr} causes the system to act like a quadratic IENN. Results in Table 5.3 and 5.4 suggest that keeping α_{thr} high should lead the classification accuracy to improve. The results are separated into two parts, with Table 5.5 summarizing results for lower α_{thr} and Table 5.6 for higher values. The best error rate can be seen for $\alpha_{thr} = 0.9$, with an error rate of around 3.9%. This is similar in performance to quadratic IENN. Changing α_{thr} causes the hybrid IENN to perform differenyly. It can also be seen that the learning rate should be selected appropriately for best performance.

For having an idea about classification accuracy, ENN and proposed IENN method is compared with mostly used feed-forward NN methods as seen in Table 5.7. For NN classification, feed-forward NN with 3 neurons has the best classification accuracy as expected. In literature, for NN classification it is stated that n hidden neurons

Table 5.6: High α_{thr} Hybrid IENN 10-fold Cross Validation Results

Parameters	Training Error	Test Error
(η=0.0001,2000)($\alpha_{thr} = 0.8$)	3.38%	4.79%
(η=0.0001,12000)($\alpha_{thr} = 0.8$)	2.86%	4.21%
(η=0.0001,25000)($\alpha_{thr} = 0.8$)	2.79%	4.13%
(η=0.0001,2000)($\alpha_{thr} = 0.9$)	3.30%	4.79%
(η=0.0001,12000)($\alpha_{thr} = 0.9$)	2.86%	3.88%
(η=0.0001,25000)($\alpha_{thr} = 0.9$)	2.75%	4.38%

for 2^n classes gives best classification accuracy. After 3 hidden neurons, increasing neuron number causes loss of generalization and leads to overfiting of data. In cross validation feed-forward NN gives worse accuracy than the proposed IENN method.

Table 5.7: Feed-Forward NN 10-fold Cross Validation Results

Parameters	Training Error	Test Error
2 hidden neurons	5.86%	6.45%
3 hidden neurons	5.48%	5.21%
5 hidden neurons	5.37%	5.87%
7 hidden neurons	5.08%	5.45%

5.2.5 Comparison of Performance with IENN

After finding the generalization performance of classification algorithms via 10-fold cross validation method, classification methods which give the best generalization value between each other are selected to be tested with unseen data samples with known SoC. Distribution of desired and estimated classes are shown in Fig. 5.2.1 and Fig. 5.2.2, results are summarized in Table 5.8.

In Fig. 5.2.1 true classification values and estimated classification values using

ENN, NN and low α_{thr} Hybrid IENN are presented. Low α_{thr} shows that in classification mostly linear separators are used. Dashed blue line represents the true class line and the diamond, square and circle symbols appeared on the dash line means correct classifying for low α_{thr} Hybrid IENN, NN and ENN respectively. ENN has misclassifications at classes 2, 4, 5, 6 and 7. It did not just misclassified edge samples but it classified some samples to class 3 and 4 although they belong to class 6. NN and low α_{thr} Hybrid IENN has misclassifications in classes 2, 5, 6, 7 and 2, 3, 5, 7 respectively. According to the graph, NN and low α_{thr} Hybrid IENN misclassified samples belong to the edges of the estimated class and desired class.

In Fig. 5.2.2 true classification values and estimated classification values using linear IENN, quadratic IENN and high α_{thr} Hybrid IENN are presented. High α_{thr} shows that in classification mostly quadratic separators are used. Plus, square and x symbols represents high α_{thr} Hybrid IENN, linear IENN and quadratic IENN classifications. High α_{thr} Hybrid IENN, linear IENN and quadratic IENN has misclassifications at classes 2, 3, 5, 7 and 2, 3, 5, 6, 7 and 2, 3, 4, 5, 7 respectively. The misclassification occurs at edge samples.

Table 5.8: Classification Performance

Classification Method	Test Error
ENN ($\eta = 0.01$)	17%
Feed-Forward NN (3 hidden neurons)	7%
Linear IENN (η=0.0001,10000)	7%
Hybrid IENN (η=0.0001,12000)($\alpha_{thr} = 0.2$)	5%
Quadratic IENN (η=0.0001,30000)	5%
Hybrid IENN (η=0.0001,12000)($\alpha_{thr} = 0.9$)	4%

Table 5.8 explains the performance of all the methods and shows correct classification rates. It confirms the effectiveness of proposed IENN. As expected, the proposed IENN classification methods give significantly more accurate results than

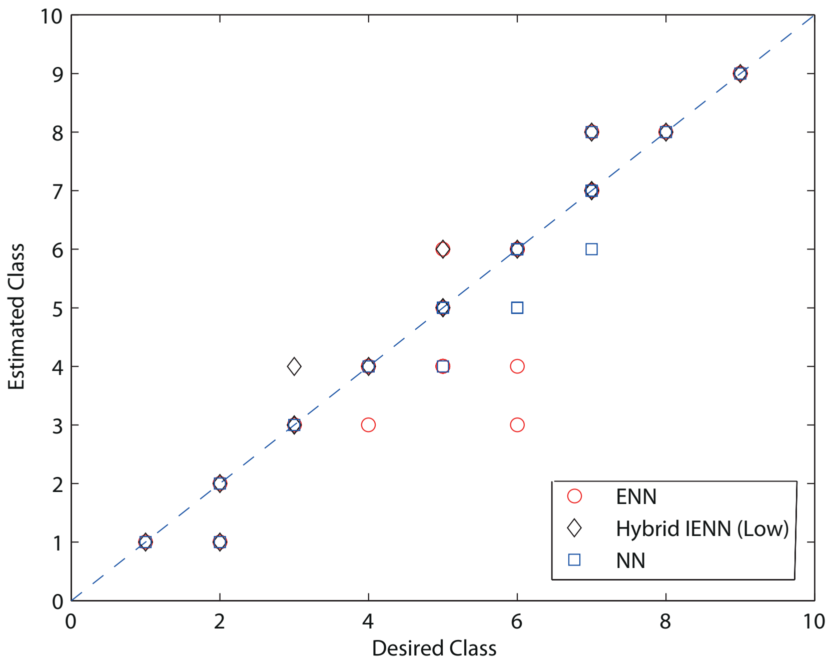


Figure 5.2.1: Comparison of ENN, Low α_{thr} Hybrid IENN and NN

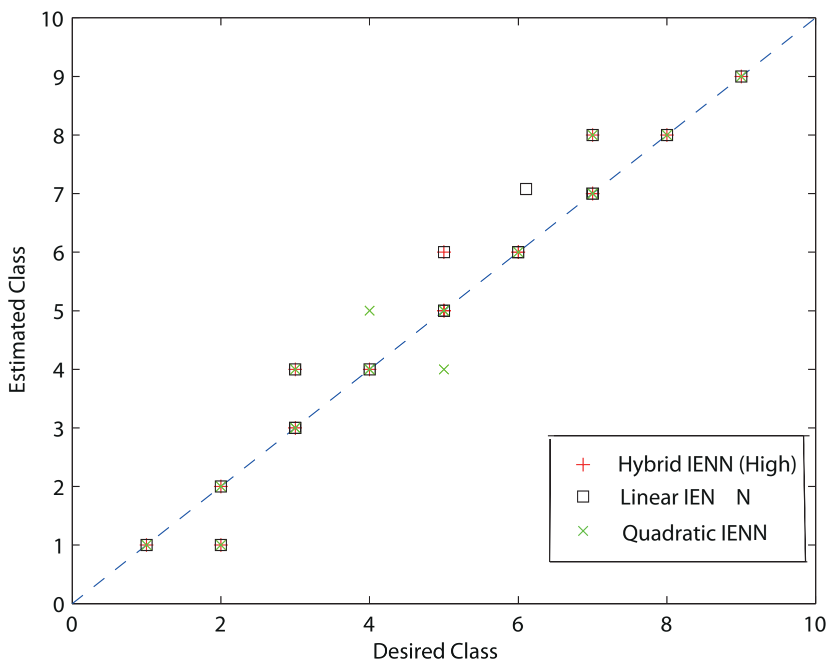


Figure 5.2.2: Comparison of High α_{thr} Hybrid IENN, Linear IENN and Quadratic IENN

ENN and feed-forward NN. Hybrid IENN with high α_{thr} has the best performance as expected. Consequently, we can say that the experiment reaches its purpose.

6 CONCLUSION

In this thesis, a novel classification method, the “improved extended neural network”(IENN) which builds upon extended neural network method (ENN) was proposed. Separators that classify each dimension of the input into classes are designed to have fixed centers at the mean of each class data set, and their width is changed by training. Linear and quadratic separators with suitable training methods were also proposed.

IENN was applied to state of charge (SoC) classification of lead-acid batteries. Since SoC cannot be measured directly but is a critical value in determining the performance of such batteries, their accurate estimation is important in areas such as fault diagnosis. Although SoC can be theoretically found by integrating the current, after a period of time integration errors cause the estimated battery terminal voltage to diverge. As a remedy, the proposed method is applied to classification of SoC of lead acid batteries to be used in their dynamic modeling. The proposed method can also be used to predict non-measurable state vectors of a nonlinear system where non-measurable state vectors estimation is complex, the input data samples of the proposed method represent continuous time signals and the nonlinear system must be modeled with only input-output relationships.

In classification of SoC, according to results based on data taken from a battery charge-discharge experimental setup, the proposed hybrid IENN performs with 4% classification error rate which is significantly better than both existing ENN and ffNN methods tested with suitable parameters. Therefore, it can be said that the experiments done in this thesis have reached their purpose. A parameter α_{thr} was also proposed which changes the behavior of IENN as applied to data sets with different variance. In classification of SoC, it is investigated that the variance of the data sets, used to predict the SoC range are low. Therefore, hybrid IENN with high α_{thr} values provides more accurate classification results than the other IENN approaches.

Further work is planned to apply the proposed method in real-time battery SoC estimation to be used as a part of a dynamic battery model in a fault diagnosis scenario. Rather than using current integration technique in SoC estimation, the proposed IENN will be used as a estimator model of SoC for dynamic battery model. The SoC input of the ffNN model of the battery will be fed with the output of the classification result of the proposed IENN model. SoC estimation will not drift in time, so that ffNN will not diverge from the actual terminal voltage after a period of time. Consequently, the model of the battery is expected to conserve its robustness in time. The method is also expected to perform well in classification of residuals in fault diagnosis systems. We plan to investigate properties of IENN to determine its performance and applicability to a broad range of problems.

Bibliography

[1] F. Amini, H. M. Chen, G. Z. Qi, and J. C. S. Yang. Generalized neural network based model for structural dynamic identification, analytical and experimental studies. *Intelligent Information Systems*, pages 138–142, 1997.

[2] Marios M. Polycarpou Arun T. Vemuri. Robust nonlinear fault diagnosis in input-output systems. *International Journal of Control*, 68:343 – 360, 1997.

[3] A. Bernieri, M. D'Apuzzo, L. Sansone, and Savastano M. A neural network approach for identification and fault diagnosis on dynamic systems. *IEEE Transactions on Instrumentation and Measurement Technology*, 43(6):867–873, 1994.

[4] W. Cai. The extension set and incompatibility problem. *Science Exploration*, 1983.

[5] Margaret A. Casacca and Ziyad M. Salameh. Determination of lead-acid battery capacity via mathematical modeling techniques. *IEEE Transactions on Energy Conversion*, 7(3):93–98, 1992.

[6] Kuei-Hsiang Chao, Meng-Hui Wang, Hung-Cheng Chen, and Yi-Feng Tseng. A novel clustering algorithm based on the extension theory and genetic algorithm. *Expert Systems with Applications*, 36(4):82698276, 2009.

[7] Kuei-Hsiang Chao, Meng-Hui Wang, and Chia-Chang Hsu. A novel residual capacity estimation method based on extension neural network for lead-acid batteries. *International Symposium on Neural Networks*, pages 1145–1154, 2007.

[8] Kuei-Hsiang Chao, Meng-Hui Wang, Wen-Tsai Sung, and Guan-Jie Huang. Using enn-1 for fault recognition of automotive engine. *Expert Systems with Applications*, 37(4):29432947, 2010.

[9] Jie Chen and Ron J. Patton. *Robust Model Based Fault Diagnosis For Dynamic Systems*, pages 35–38. Kluwer Academic, 1999.

[10] Janos Gertler. *Fault detection and diagnosis in engineering systems*, pages 183–246. CRC Press, 1998.

[11] R. Iserman. Process fault detection based on modeling and estimation methods. *Automatica*, 20(4):387–404, 1984.

[12] R. Iserman. Supervision, fault-detection and fault-diagnosis methods - an introduction. *Control Engineering Practice*, 5:639–652, 1997.

[13] Andreas Jossen. Fundamentals of battery dynamics. *Journal Of Power Sources*, 154(2):530–538, 2005.

[14] A. Kawamura and T. Yanagihara. State of charge estimation of sealed lead-acid batteries used for electric vehicles. *Power Electronics Specialists Conference*, 1:583–587, 2002.

[15] Nosh K. Medora and Alexander Kusko. An enhanced dynamic battery model of lead-acid batteries using manufacturers' data. *Telecommunications Energy Conference*, pages 1–8, 2006.

[16] Yoshifumi Morita, Sou Yamamoto, Sun Hee Lee, and Naoki Mizuno. On-line detection of state-of-charge in lead acid battery using both neural network and on-line identification. *IEEE Industrial Electronics*, pages 3379–3384, 2006.

[17] Sabine Piller, Marion Perrin, and Andreas Jossen. Methods for state-of-charge determination and their applications. *Journal Of Power Sources*, pages 113–120, 2001.

[18] Shinya Sato and Atsuo Kawamura. A new estimation method of state of charge using terminal voltage and internal resistance for lead acid battery. *Power Conversion Conference*, 2:565–570, 2002.

[19] Chuin-Mu Wang, Ming-Ju Wu, Jian-Hong Chen, and Cheng-Yi Yu. Extension neural network approach to classification of brain mri. *Intelligent Information Hiding and Multimedia Signal Processing*, pages 515–517, 2009.

[20] M.H. Wang and C.P Hung. Extension neural network and its applications. *International Joint Conference On Neural Networks*, 5-6:779–784, 2003.

[21] Takahiro Yanagihara and Atsuo Kawamura. Residual capacity estimation of sealed lead-acid batteries for electric vehicles. *Power Conversion Conference*, 2:943–946, 1997.

[22] Jun Ye. Application of extension theory in misfire fault diagnosis of gasoline engines. *Expert Systems with Applications*, 36(2):1145–1154, 2009.

[23] Juncai Zhang, Xu Qian, Yu Zhou, and Ai Deng. Condition monitoring method of the equipment based on extension neural network. *Chinese Control and Decision Conference*, pages 1735–1740, 2010.