EVOLUTIONARY OPTIMIZED HYBRID COMPUTATIONAL INTELLIGENCE MODEL FOR THE PREDICTION OF GAS COMPONENTS

BY

MUHAMMAD IMTIAZ HOSSAIN

A Thesis Presented to the
DEANSHIP OF GRADUATE STUDIES

KING FAHD UNIVERSITY OF PETROLEUM & MINERALS

DHAHRAN, SAUDI ARABIA

In Partial Fulfillment of the
Requirements for the Degree of

MASTER OF SCIENCE

In

COMPUTER SCIENCE

June, 2011

KING FAHD UNIVERSITY OF PETROLEUM & MINERALS
DHAHRAN 31261, SAUDI ARABIA

DEANSHIP OF GRADUATE STUDIES

This thesis, written by **Muhammad Imtiaz Hossain** under the direction of his thesis advisor and approved by his thesis committee, has been presented to and accepted by Dean of Graduate Studies, in partial fulfillment of the requirements for the degree of **MASTER OF SCIENCE IN COMPUTER SCIENCE.**

Thesis Committee

Tarek Helmy El-B[illegible]

Dr. Tarek Helmy El-Basuny (Advisor)

[illegible]

Dr. Abdulazeez Abdulraheem (Co-Advisor)

[illegible]

Prof. Dr. Moustafa Elshafei (Member)

Kanaan Faisal

Dr. Kanaan Abed Faisal (Member)

[illegible]

Dr. Lahouari Ghouti (Member)

[illegible]

Dr. Adel Ahmed
(Department Chairman)

[illegible]

Dr. Salam A. Zummo
(Dean of Graduate Studies)

عمادة الدراسات العليا
DEANSHIP OF GRADUATE STUDIES

Date 26/6/11

DEDICATION

DEDICATED TO MY FAMILY

ACKNOWLEDGMENT

First and foremost, I thank Allah (subhanu wa taala) for bestowing me with health, patience, and knowledge to complete this work. Acknowledgement is due to the King Fahd University of Petroleum & Minerals (KFUPM) for the support given to this research through its excellent facilities and for granting me the opportunity to pursue my graduate studies with financial support.

The author would like to acknowledge the affirm provided by King Abdulaziz City for Science and Technology (KACST) through the Science & Technology Unit at KFUPM for funding this work under the project no GSP-18-101 as part of the MS Thesis.

I acknowledge, with deep gratitude and appreciation, the inspiration, encouragement, valuable time and guidance given to me by Dr. Tarek Helmy, who served as my major advisor. Thereafter, I am deeply indebted and grateful to Dr. Abdulazeez Abdulraheem, my co-advisor, and Prof. Dr. Moustafa Elshafei, my committee member, for their extensive guidance, continuous support, and personal involvement in all phases of this research. I am also grateful to my other committee members, Dr. Lahouari Ghouti and Dr. Kanaan Abed Faisal for their constructive guidance, valuable advices and cooperation.

I also acknowledge the sincere and untiring efforts of Dr. Sayed M. Rahman of the Research Institute (RI) at KFUPM. The author would like to thank Dr. Md. Rafiul Hassan of ICS department and Dr. Amar Khoukhi of System Engineering department for their careful suggestions and help. A warm regards goes to Dr. Saifur Rahamn, Dr. Wasef A. Nofal, Mr. Sujhat and Mr. Fatai of the RI for their cooperation in dataset preparation and suggestions. A Special thanks to Mr. Abdulrahim Muhammadain of the Petroleum Engineering department at KFUPM for his expert judgment to help correlating the problem domain with Computational Intelligence. A special thanks to former PhD candidate Dr. Tanvir Parvez of ICS department. I am also indebted to the ICS department chairman Dr. Adel Ahmed and other faculty members for their support.

Thanks to all my elders and friends for their helpful comments, resulting in a significant improvement in the quality of this thesis. Finally, I would like to express my deepest gratitude to my mother, father, wife, sister, brother and all other relatives, for their emotional and moral support throughout my academic career and also for their love, patience, encouragement and prayers.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

THESIS ABSTRACT

Name	: Muhammad Imtiaz Hossain
Title	: Evolutionary Optimized Hybrid Computational Intelligence Model for the Prediction of Gas Components
Major Field	: Computer Science
Date of Degree	: June 2011

In this thesis, gas components in a multi-stage separator are predicted using Hybrid Computational Intelligence (HCI) and Ensemble of HCI (EHCI) models. We have used Root Mean Square Error (RMSE), Correlation Coefficient (CC), training time and number of negatively predicted values as performance measures of the HCI, EHCI models and compared with Equation of State and Empirical Correlation based Chevron Phase Calculation Program (CPCP) as a benchmark. First, we have used the evolutionary algorithm based Genetic Algorithm to optimize the parameters of the Computational Intelligence models such as Artificial Neural Network, Support Vector Regression and Adaptive Neuro-Fuzzy Inference System in order to form HCI models. We observed that for 2 out of 6 gas components, the performance of HCI models is better than CPCP but for the rest 4 gas components the performance is very close to the CPCP in terms of RMSE and CC but truly could not outperform it. Due to this reason and because the generalization ability of an ensemble is usually much stronger than that of base learners, we have developed heterogeneous and homogenous types of EHCI models. The experimental results of the EHCI models show that for 5 out of 6 gas components, the EHCI models outperformed both the individual HCI models and CPCP in terms of RMSE and CC with no negative predicted values at all.

ملخص الرسالة

الاســــــــم : محمد امتياز حسين

عنوان الرسالة : نموذج حسابي مُهَجن ومُحَسَّن لتنبؤ بمكوّناتِ الغازِ

التخصـــــص : علوم الحاسب

تاريخ التخــرج : يونيو 2011

يعتبر التنبؤ بمركبات الغازات في عمليات فصل الغازات عن الزيوت متعددة المراحل تحديا وذلك لأن معدل تغير المركبات تتنوع كثيرا مع تغير الضغط والحرارة. ولا توجد علاقة مباشرة بين هذه المتغيرات حيث إنها تعتمد على المكمن - بل وأحيانا على العينة - وعلى والمركبات الابتدائية للزيت. وعمليات فصل الغازات عن الزيوت هذه لها أهمية كبيرة في إنتاج كل من الغازات والزيوت. تقدير كمية المركب الغازي قبل إنتاجه قد يساعد في تخفيض تكلفة الإنتاج، وزيادة كفاءة الإنتاج، وتحديد جودة الزيت. وعادة ما تستخدم طريقتا معادلة الحال والارتباط التجريبي لتحليل الحال تعمل على ما يرام في بعض الحالات لكنها غير قادرة على التنبؤ لبعض الخصائص فى مكونات الموائع. معادلة كثير من الحالات. ففي حالات الهيدروكربونات المعقدة تكون كلتا الطريقتين ذات نتائج غير مرضية. لذا، فقد تزايد الطلب على تقنيات الذكاء الحاسوبي في مجال هندسة النفط وغيره. ليس هذا فحسب، بل إن أنظمة مهجنة من تقنيات الذكاء الحاسوبي المختلفة باتت تطرح من أجل كفاءة أعلى. في هذه الرسالة، نقوم بالتنبؤ بمركبات الغازات في عمليات فصل الغازات عن الزيوت متعددة المراحل باستخدام أنظمة الذكاء الحاسوبي الهجينة ونماذج موحدة لها. وقد قمنا باستخدام معايير متعددة لقياس كفاءة العمل. ابتدأنا بإيجاد أمثل القيم لمتغيرات أنظمتنا عبر خوارزميات جينية، مما أسفر عن تحسن التنبؤ لمكونين اثنين من أصل ستة مكونات للغاز. ثم قمنا بعمل نماذج متجانسة وغير متجانسة من تقنيات الذكاء الحاسوبي المختلفة مما أسفر عن تحسين التنبؤ لخمسة من تلك المكونات مقارنة بأداء الطرق التقليدية.

CHAPTER 1

INTRODUCTION

Gas components prediction in multi-stage oil and gas separation process is a challenging task as the rate of changes of the components varies in bulk by the change of pressure and temperature. There is no straight forward relation between these changes rather it depends on reservoir characteristics and initial oil components which might vary by reservoir to reservoir and even sample to sample. Gas separation is an important process which is essential for oil and gas production. Quantifying the gas composition prior to the production may help in cutting down the production cost, maximizing the production efficiency and determining the quality of oil. Equation of State (EOS) and Empirical Correlation (EC) are generally used for fluid components analysis. EOS works properly under some stable conditions but unable to estimate the properties of all substances under all conditions accurately. EOS is basically a poor predictive tool for complex hydrocarbon system and on the other hand EC has limited accuracy though it doesn't involve complex calculation. Computational Intelligence (CI) techniques such as Artificial Neural Network (ANN), Support Vector Regression (SVR) and Adaptive Neuro-Fuzzy Inference System (ANFIS) have gained immense popularity in many areas of research including petroleum engineering and outperform the conventional EOS and

EC based techniques in many cases. Moreover, combinations of CI models with evolutionary optimized models will have an obvious advantage in their performance when applied in complex domains of application. In this thesis, gas components in a multi-stage separator are predicted using Hybrid Computational Intelligence (HCI) and Ensemble of HCI (EHCI) models. We have used Root Mean Square Error (RMSE), Correlation Coefficient (CC), training time and number of negatively predicted values as performance measures of the HCI, EHCI models and compared with conventional EOS and EC based Chevron Phase Calculation Program (CPCP) as a benchmark. First, we have used the evolutionary algorithm based Genetic Algorithm (GA) to optimize the parameters of the CI models mentioned above in order to form HCI models. We observed that for 2 out of 6 gas components, the performance of HCI models is better than CPCP but for the rest 4 gas components the performance is very close to the CPCP in terms of RMSE and CC but truly could not outperform it. Due to this reason and because the generalization ability of an ensemble is usually much stronger than that of base learners, we have developed heterogeneous and homogenous types of EHCI models. The experimental results of the EHCI models show that for 5 out of 6 gas components, the EHCI models outperformed both the individual HCI models and CPCP in terms of RMSE and CC with no negative predicted values at all.

1.1. Hybrid and Ensemble Computational Intelligence Models

Computational Intelligence (CI) is a branch of computer science that deals with problems for which do not have any effective solutions. Researches in CI have produced a huge collection of algorithms, grouped into the main CI paradigms. These CI techniques are used repeatedly to create hybrid intelligent systems, where different algorithms from different CI paradigms are combined to form a hybrid model. This process required a re-implementation of existing CI algorithms. In addition to the CI components of a hybrid system, a communication protocol among the CI techniques are needed to be defined and implemented.

A Hybrid Computational Intelligent (HCI) system combines at least two CI techniques. For example, combining a ANN with a Fuzzy Inference System (FIS) results in a hybrid neuro-fuzzy system. HCI models are defined as any effective combination of CI techniques in sequential or parallel manner that performs superior to simple CI techniques [**1**,**2**]. HCI was adopted in several scientific papers during the last decade, as an extension to the standard experimentation along with other well-known CI techniques, in various application domains [**3**,**4**,**5**]. In EUNITE 2001 [**6**], it was stated that "intelligent hybrid systems" are meant to be any combinations of intelligent technologies (e.g. neuro-fuzzy approaches, evolutionary optimized networks, etc.) but particularly those, which have an noticeable advantage in their performance when it is applied in complex domains of application (either by means of accuracy obtained, or by means of comprehensibility of the acquired results). The main challenge of HCI model is the collaboration efficiency of each component. Another important factor of a hybrid system is the speed of process and the time needed to produce a generalized high-performance decision model. The evidence drawn from recent literature [**1**,**2**] on the effectiveness of a specific kind of hybrid methodologies in a variety of real-world applications could render this hybrid scheme as method of choice for the decision makers. In this sense, to solve different parts of the overall problem can be manipulated effectively by different intelligent techniques (e.g. cluster formation, feature selection for reduction of complexity and so forth), a fact that often leads to the establishment of a hybrid intelligent model for better handling of the problem. Choosing an appropriate HCI model is vital as one should start from referring to the particular advantages and disadvantages of each of the standard CI techniques. The evolutionary computation, genetic programming, etc., models are time consuming in the training phase, so the complexity of this method has to be accounted for prior to the design stage. On the other hand, they perform very well in generalization and robust model building from complex data. Figure 1 shows optimization of SVR model using GA.

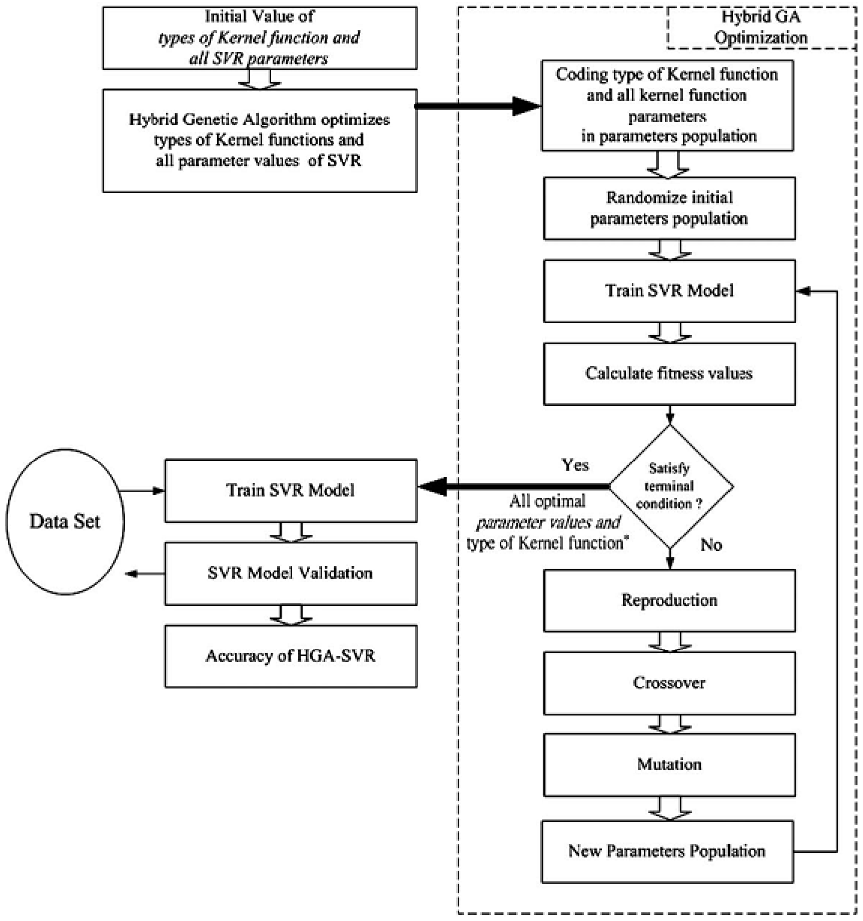

Initial Value of types of Kernel function and all SVR parameters
Hybrid Genetic Algorithm optimizes types of Kernel functions and all parameter values of SVR
Hybrid GA Optimization
Coding type of Kernel function and all kernel function parameters in parameters population
Randomize initial parameters population
Train SVR Model
Calculate fitness values
Satisfy terminal condition ?
Yes
No
All optimal parameter values and type of Kernel function*
Data Set
Train SVR Model
SVR Model Validation
Accuracy of HGA-SVR
Reproduction
Crossover
Mutation
New Parameters Population

Figure 1: Optimization Process of Genetic SVR [7]

If the problem involves uncertainty, incorporating the Fuzzy Logic (FL) in hybrid model would be a good choice. Fuzzy rule based approaches are by far advantageous in handling of approximate or vague concepts existing within a dataset. ANN is typical a black box architecture (i.e. of very low comprehensibility of the produced decision model) that prove superior in handling numerical data and highly non-linear domains of application. Thus the Neuro-fuzzy systems are usually superior to simple ANN, due to the fact that a

ANN "suffers" from noise, whereas the neuro-fuzzy system has the ability to "absorb" the noise with the use of the embedded membership functions. Fuzzy-genetic systems are preferable than simple FIS as the fuzzy-genetic approaches do not have to define oneself the rule-base. For similar reasons, a neuro-fuzzy system is superior to a simple FIS, as the neuro-fuzzy systems do not have to tune the rule-base. Generally the reliability and the availability of the data under processing are also a crucial factor for the success or the failure of a specific hybrid intelligent methodology.

Ensemble learning is a way of combining different CI or HCI models (at least two) in parallel or sequential manner. Ensemble model contains a number of learners which are usually called *base learners*. The generalization ability of an ensemble is usually much stronger than that of base learners [**8**,**9**]. Actually, ensemble learning is appealing because that it is able to boost *weak learners* which are slightly better than random guess to *strong learners* which can make very accurate predictions. So, "base learners" are also referred as "weak learners". It is noteworthy, however most theoretical analyses work on weak learners, base learners used in practice are not necessarily weak since using the not-so-weak base learners often shows better performance. Base learners are usually generated from training data by a *base learning algorithm* which can be ANN, SVR, ANFIS or other kinds of machine learning algorithms.

Figure 2: Ensemble of CI Models

Most ensemble methods use a single base learning algorithm to produce *homogeneous* base learners, but there are also some methods which use multiple learning algorithms to produce *heterogeneous* learners. In the latter case there is no single base learning algorithm and so it is called *individual learners* or *component learners* to “base learners”, while the names “individual learners” and “component learners” can also be used for homogeneous base learners. It is difficult to trace out the starting point of the history of ensemble methods since the basic idea of deploying multiple models has been in use for a long time, yet research on ensemble learning became popular since the 1990s. The first is an applied research conducted by Hansen and Salamon [**10**] at the end of 1980s, where they found that predictions made by the combination of a set of classifiers are often more accurate than predictions made by the best single classifier. The second is a theoretical research conducted in 1989, where Schapire [**11**] proved that *weak learners* can be boosted to *strong learners*, and the proof resulted in Boosting, one of the most influential ensemble methods.

1.2. PROBLEM BACKGROUND

Petroleum deposits are naturally occurring mixtures of organic compounds consisting mainly of hydrogen and carbon and are termed as hydrocarbons which are mainly Methane, Ethane, Propane, Butane, and other organic compounds. The deposits found in the gaseous form are called 'natural gas' and that in the liquid form is called 'crude oil'. Apart from hydrocarbon gases, *non-hydrocarbon* gases also exist in the reservoirs in varying amounts. The non-hydrocarbon gases are treated as contaminants which are nitrogen (N_2), hydrogen (H_2), carbon dioxide (CO_2), hydrogen sulfide (H_2S), and rare gases such as helium. Crude oil and gases are found underground at elevated pressure and temperature conditions. Gas extracted with crude oil from oil wells (called "associated" gas) must be separated at the wellhead. Producing, separating, transporting, and storing petroleum fluids are the primary responsibilities of a petroleum and natural gas engineer. At every stage of the petroleum exploration and production business, a hydrocarbon fluid engineer is needed. Hydrocarbon fluid engineers might find themselves dealing with activities such as reserve evaluations, drilling operations, reservoir analyses, production operations, and gas processing. Most of the fluid handling protocols require the engineer to derive a priori about how the fluids will behave under a wide range of pressure and temperature conditions. Optimal design and efficient operation of hydrocarbon production handling and processing systems strongly depends on accurate knowledge of fluid phase behavior. Usually it is much economical to use three to four stages of separation for the hydrocarbon mixture. Maximization of condensate yield is virtually impossible without the tools for accurate prediction of the amount of liquid existing under a given condition of pressure, temperature and composition. Therefore, having advanced predictive tools for the characterization of hydrocarbon phase behavior with highest accuracy proves to be a key solution so as to overtake the economics of hydrocarbon systems.

The function of oil production is focused on separating the oil well stream into three components or "phases" (oil, gas, and water) into marketable products or disposes them in an environmental friendly manner. "Separators" is a mechanical device where gas is flashed from the liquids and "water" is separated from the oil (Figures 3, 4). These steps also remove light hydrocarbons from oil to produce a stable crude oil with volatility (i.e., vapor pressure) that meet the required criteria. Separators are classified as "two-phase" if they separate gas from the total liquid stream and "three-phase" if they also separate the liquid stream into its crude oil and water components. The separated gas is compressed before commercializing. Modeling such mechanism is very crucial for controller design, fault detection and isolation, process optimization and dynamic simulation [**12**]. In this thesis, we focused on predicting the three-stage separators' gas compositions as they form the main processes in the upstream petroleum industry and have a significant economic impact on produced oil quality.

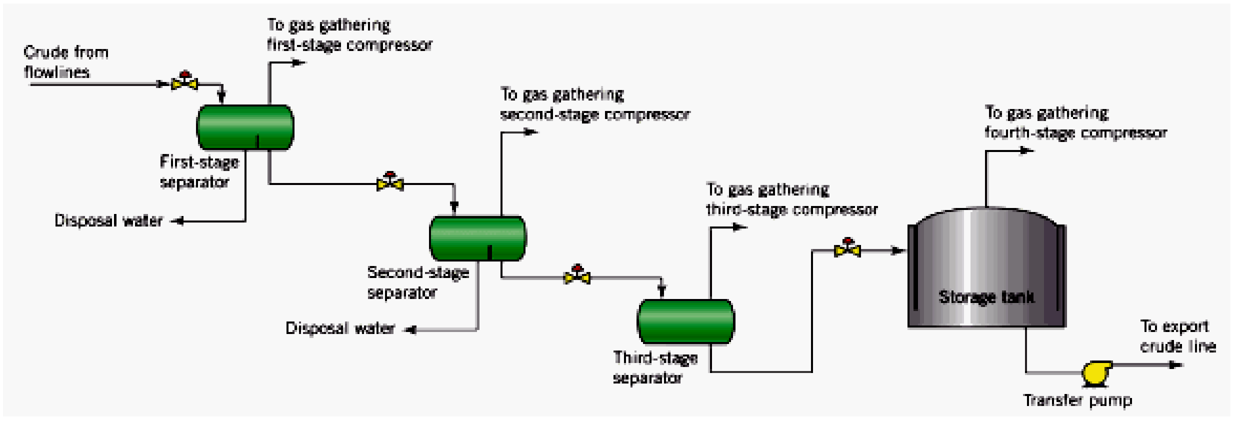

Crude from flowlines
To gas gathering first-stage compressor
First-stage separator
Disposal water
To gas gathering second-stage compressor
Second-stage separator
Disposal water
To gas gathering third-stage compressor
Third-stage separator
To gas gathering fourth-stage compressor
Storage tank
To export crude line
Transfer pump
Figure 3: Multistage Surface Separation Facility [13]

Figure 4: Separators in Oil Field

Capacity and efficiency of gas/liquid separation is a major issue in natural gas production. One of the problems encountered in the field of petroleum is the fact that the behavior of the multiphase flow under the prevailing circumstances is complex and quite difficult to predict. A complication that occurs when attempting to quantify the behavior of these multiphase flows is that under high pressure the properties of the mixture may differ considerably from those of the same mixture under atmospheric conditions. This effect requires expensive experimental equipment to conduct experiments under actual circumstances and equally expensive computing equipment and software to carry out numerical flow simulations. Produced gas contains liquid and solid constituents. The removal of these constituents forms the most important process before delivery. The liquid almost invariably consist of water and hydrocarbons that are gaseous under reservoir conditions but condenses during the production due to drop off pressure and temperature.

Figure 5: Conceptual View of Reservoir and Three Stage Separator

Oil resides in the reservoir at huge temperature and pressure such as 250°F and 5000 psi respectively. After the oil extracted from reservoir it is collected in sequential tanks under low temperature and pressure such as 150°F and 175 psi respectively (Figure 5). Due to this huge loss of temperature and pressure gas releases and gets separated. Most of the time 3 separators are used and in some cases up to 4. The pressure and the temperature in these separators decrease as oil moves further and so gas releases. At the stage 3 the pressure and temperature becomes normal such as 14.7 psi and 60°F. According to Figure 6, Stage1 consist of oil and gas (Oil1+Gas1). Gas1 is then extracted and Oil1 moves to Stage2 separator. As pressure and temperature decrease, gases come out from Oil1. Therefore Oil1 = Oil2 + Gas2. In the same way, Oil2 moves to Stage3 separator which is at atmospheric pressure and temperature (14.7 psi and 60°F). At this stage too, gases are released. Therefore, Oil2 = Oil3 + Gas3.

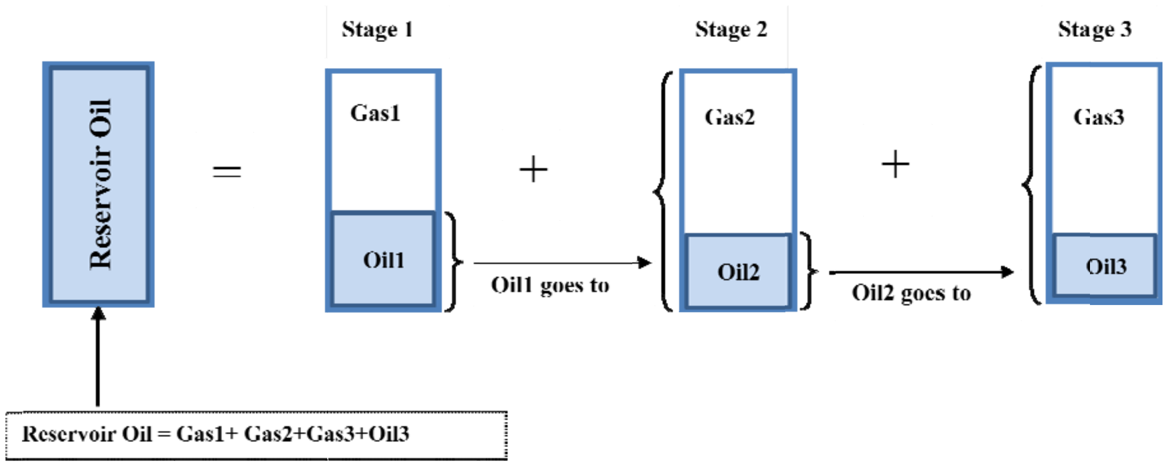


Figure 6: Oil and Gas Flow in Three-stage Separator System

Gas components prediction is carried out in few research areas [**14**,**15**,**16**,**17**,**18**]. But to the best of our knowledge no noticeable work has been done in the field of gas components prediction in multistage separator using CI techniques. In the industry, EOS and EC are usually used for fluid compositions analysis and determining other oil/gas properties. For example, Chevron's Phase Calculation Program (CPCP) is a program based on EOS and EC being used in industries for various purposes. CPCP is designed to help the engineer to calculate the phase compositions, densities, viscosities, thermal properties, and the interfacial tensions between phases for liquids and vapor in equilibrium. One of the applications of CPCP is that it considers reservoir crude oil compositions, C7+ Molecular weight and density, separator stage temp and pressure as input to predict the gas compositions on particular stage using EOS and EC. EOS is useful for description of fluid properties like Pressure Volume Temperature (PVT). But there is no single EOS that accurately estimates the properties of fluids under all conditions. The EOS has adjustment issues against the phase behavior data of reservoir fluid of known composition while the EC has limited accuracy [**19**]. In the recent years, CI techniques such as ANN, SVR and ANFIS have gained immense popularity in solving various petroleum related problem like PVT properties, Porosity, Permeability, Viscosity prediction, etc [**20**,**21**,**22**,**23**,**24**,**25**,**24**,**26**,**27**]. Each of the CI techniques have some limitations and is already proved in the literature that an ensemble or hybrid of these models have better generalization ability than a single CI model [**26**,**27**,**28**,**29**]. In this thesis, heterogeneous and homogeneous EHCI models are developed to learn the complex relationship between the input and the output parameters to predict the gas compositions in multi-stage separator. The accuracy of ensemble model depends on the diversity and accuracy of each member of ensemble model [**9**,**30**,**31**] . We enforced diversity by using heterogeneous and homogeneous ensemble models as well as random sampling from the idea of Bagging and Boosting. HCI is used to enforce accuracy of each member of the ensemble models. The hybrids are designed in order to be benefitted from the strengths of the individual techniques and to complement the weaknesses of each of them and thus enforce accuracy on the unseen data. Experimental results show that the generalization ability of EHCI outperforms the GA optimized single CI models. EHCI also outperforms the conventional EOS and EC based CPCP for most of the hydrocarbons and non-hydrocarbons in gases.

1.3. Problem Statement

Enormous volume of gas is released from oil/gas production process. The function of oil production is focused on separating the oil well stream into three components or “phases” (oil, gas, and water) into marketable products or disposes them in an environmental friendly manner. “Separators” is a mechanical device where gas is flashed from the liquids and “water” is separated from the oil. Knowing the gas compositions produced in separator helps to determine the quality of oil and optimize the production process. We want to predict the gas compositions in multistage separator that releases from oil production process. In the industry, EOS and EC are usually used for fluid compositions analysis and determining other oil/gas properties. The EOS has adjustment issues against the phase behavior data of reservoir fluid of known composition while the EC has limited accuracy. CI techniques such as ANN, SVR and ANFIS have gained immense popularity in solving various petroleum related problem like PVT, Porosity, Permeability, Viscosity prediction, etc and outperform EOC and EC based techniques. To achieve better prediction accuracy we have used EHCI models to predict the gas components in the multi-stage separator.

1.4. OBJECTIVE

The goal of this thesis is to develop EHCI models to solve a regression problem of gas components prediction in multi-stage separator. We used ANN, SVR and widely used hybrid model ANFIS as members of EHCI models. To have better generalization, ensemble members should be diverse and accurate as well. We improved the accuracy of the CI models by using Evolutionary Algorithms (EA) based Genetic Algorithm (GA). To enforce diversity we have used different model structure of the same CI model results in creating homogeneous EHCI models. Different CI models are used to create heterogeneous EHCI model and thus having diversity among the members of the ensemble. Moreover we have used the concept of "boosting" sampling techniques that applied in classification problem to boost the performance of the EHCI members in sequential manner which results in more diverse members.

1.5. MOTIVATION

Gas separation is an essential step for gas/oil production process. Gas composition predictions beforehand may help in cutting down the production cost, maximizing the production efficiency and determining the quality of stock tank oil. EOS and EC are generally used for fluid compositions analysis. But there is no single EOS that accurately estimates the properties of all substances under all conditions. EOS is poor predictive tools for complex hydrocarbon system and EC has limited accuracy. To the best of our knowledge CI techniques are not yet applied to sort out this problem. Moreover, CI techniques successfully outperform EOS and EC techniques in many applications of petroleum industry and there is no existing robust solution gas compositions prediction in multistage separators.

1.6. DATASETS DESCRIPTION

In this thesis, we have collected data of 60 different crude oil samples from Asian oil reservoirs. We have also collected around 17 samples from European oil reservoirs [**32**]. To increase the number of training samples we have synthesized 50 samples from the available data by using the material balance method [**33**]. We have used 80% of the relevant samples for training and validation and the remaining 20% used for testing.

We used the reservoir crude oil sample compositions as well as other available information as an input and the separator gas compositions as output. We are predicting the gas compositions in the further stage at certain temperature and pressure. Input parameters consist of mole percent of the non-hydrocarbon and hydrocarbon contents of reservoir crude oil sample. Non-hydrocarbons i.e. N_2, H_2S and CO_2 and hydrocarbons i.e. Methane (CH_4 as C1), Ethane (C_2H_6 C2), Propane (C_3H_8 as C3), Butane (C_4H_{10} C4), Pentane (C_5H_{12} as C5), Hexane (C_6H_{14} as C6), Heptanes & heavier (C7+) are present in the crude oil. Isomers of C4 and C5 are also presents. Hydrocarbons are distinguished by the number of carbon atoms in the molecule. Methane (CH_4) is symbolized as C1 which represents one carbon atom in the molecule (Figure 7).

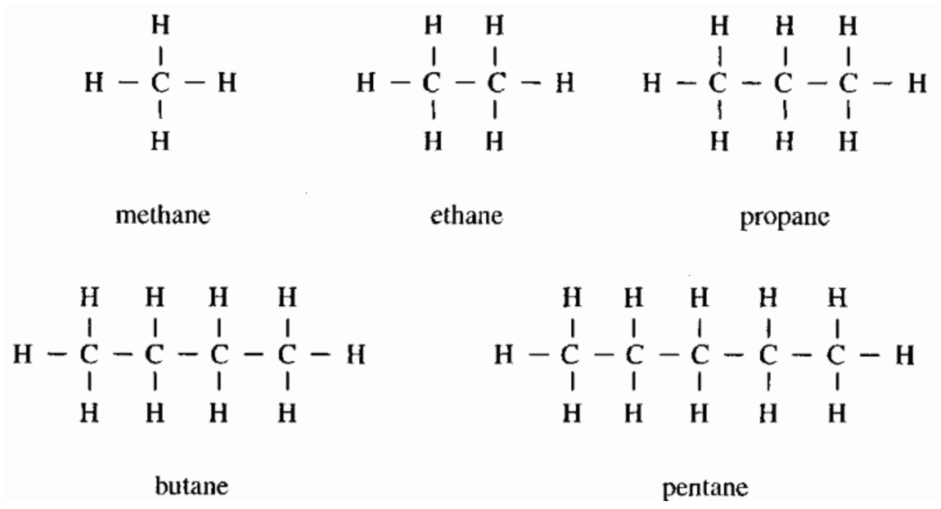


Figure 7: Chemical Bonding of Hydrocarbons

At normal temperature and pressure C1 to C4 are gases, C5 to C16 are liquids and those with more than 16 atoms of carbon are in solid state. All the components of gas and hydrocarbons occur in liquid state in the reservoir due to the presence of high pressure and temperature. The other available information is also used as an input such as stock tank API gravity (American Petroleum Institute gravity is a good indicator of its quality and is the major basis for its pricing which is a measurement of the density of crude oil), Bubble point pressure (BPP), reservoir temperature, separator stage pressure and temperature, C7+ molecular weights, C7+ density. Output parameters consist of mole fraction of different non-hydrocarbon and hydrocarbon gases such as N_2, CO_2, H_2S, C1, C2, C3, iC4, nC4, iC5, nC5, C6, and C7+ at each stage.

1.7. Datasets Analysis

The statistical analysis of the inputs and outputs data is provided in Table 1 and Table 2. Table 1 and Table 2 provide the statistical descriptions of the predictor variables and predicting variables respectively. The mean and standard deviation provide insights about the dispersion, and the maximum and minimum values indicate the range of the data. Skewness is a measure of the asymmetry of the data around the sample mean and its negative value indicates that the data are spread out more to the left of the mean than to the right. On the other hand, the positive value of skewness means that the data are spread out more to the right. If the value of skewness is zero, it can be concluded that the distribution is normal distribution or any perfectly symmetric distribution. The skewness values of the data used in this study revealed that majority of the predictor and predicting variables are spread out more to the right of the mean and there is no clear indication that the data are generated from any perfectly symmetric distribution process. Kurtosis reveals the outlier-prone characteristics of a distribution and the kurtosis of the normal distribution is 3. The kurtosis values of the input data indicate that all the variables are less outlier-prone than the normal distribution except for four cases. On the other hand the kurtosis values of the output data indicate that half of the variables are less outlier-prone than the normal distribution except for five cases.

1.7.1. Input Data Analysis

Table 1: Statistical Properties - Input Data

Input	N2	CO2	H2S	C1	C2	C3	i_C4	n_C4	i_C5	n_C5	C6	C7+	C7+ Density	C7+ MW	Stage Temp	Stage Pres	BPP	ST API	Res Temp
Mean	0.30	1.61	0.74	18.16	6.90	6.81	1.43	4.53	2.01	2.95	4.45	50.15	0.87	260.36	116.29	126.29	2183.18	36.89	193.22
Std Dev	0.66	2.18	1.88	13.90	3.06	1.68	0.61	1.05	0.70	0.78	1.68	15.16	0.04	39.70	33.75	110.81	750.19	10.32	27.96
Max	4.70	7.38	12.37	47.70	14.05	12.03	3.34	7.76	4.22	4.79	9.79	75.11	0.93	350.00	315.00	519.00	3986.00	124.10	280.00
Min	0.00	0.00	0.00	0.52	1.29	2.83	0.79	2.52	1.19	1.38	1.88	24.97	0.72	193.00	50.00	14.70	390.00	24.20	130.00
Skewness	4.92	1.26	4.31	0.44	-0.04	0.35	1.46	0.97	1.04	0.24	1.00	0.15	-1.20	0.32	2.01	1.07	0.10	5.58	0.60
Kurtosis	26.82	0.10	22.00	-0.99	-1.10	0.84	1.01	0.81	0.54	-0.82	0.56	-1.48	3.72	-0.63	9.72	1.23	0.01	45.76	0.39

1.7.2. Output Data Analysis

Table 2: Statistical Properties - Output Data

Input	N2	CO2	H2S	C1	C2	C3	i_C4	n_C4	i_C5	n_C5	C6	C7+
Mean	0.65	3.61	1.14	47.74	19.78	15.62	2.35	5.34	1.24	1.38	0.83	0.31
Std Dev	1.41	4.28	2.91	22.36	6.15	10.32	2.34	4.64	1.21	1.22	0.75	0.30
Max	9.66	16.44	18.86	82.40	36.17	42.36	9.94	20.67	6.68	6.52	3.95	1.80
Min	0.00	0.00	0.00	2.98	8.37	3.81	0.41	1.04	0.16	0.19	0.08	0.00
Skewness	4.86	1.26	4.21	-0.47	0.41	0.97	1.68	1.32	1.76	1.64	1.88	2.23
Kurtosis	25.72	0.26	20.76	-0.79	-0.18	-0.42	2.18	0.62	3.27	2.63	3.58	6.59

1.8. Contribution of this Thesis

1. A systemic way of building Ensemble of HCI (EHCI) Models.
2. An Evolutionary Algorithm based Genetic Algorithm is effectively used to optimize the parameters of the members' of EHCI models.
3. The Output of each member of EHCI models is aggregated in linear and non-linear manners and analyzed the consequences in EHCI models.
4. The EHCI models are applied in gas components prediction in multistage separator, where there is no robust CI based solution available for this problem.

1.9. Thesis Organization

The thesis is structured as follows:

Chapter 2: Presents literature review of the CI, HCI, Ensemble models and the techniques used for gas composition prediction in various research areas.

Chapter 3: Gives an in depth description of ANN, SVR, ANFIS and GA.

Chapter 4: Provides the methodology of building HCI models.

Chapter 5: Describes in details of EHCI models building steps.

Chapter 6: Illustrates the experimental results, analysis of the results obtained by both HCI and EHCI models followed by a comparison with the CPCP benchmark.

Chapter 7: Provides conclusions and future work.

CHAPTER 2

LITERATURE REVIEW

2.1. COMPUTATIONAL INTELLIGENCE IN PETROLEUM ENGINEERING

Farhan et al. [**20**] have developed a reliable predictive tool using ANN for the forecasting of optimum operating conditions of a surface facility for the recovery of condensates from natural gases. They used ANN consists of 2 hidden layers with 30 and 15 neurons, 13 input neurons (C1 – C7+, N_2, H_2S, CO_2, Pressure, MW C7+, SG C7+) and 3 output neurons (optimum CGR, API, and pressure at stage2). The "log–sigmoidal" and "purelin" function were utilized as the transfer function within middle layers. The network is able to predict optimum operating conditions for maximum surface condensate recovery with minimal error. They also implemented TEKA criteria [**34**] to determine the input relevancy. They found that surface condensate recovery from natural gases is highly dominated by the amounts of lights (C1), intermediates (C4), heavies (C7+) and pressure at the operating stage. Beyond these, the non-hydrocarbons have more influence than the other hydrocarbons.

Elsharkawy et al. [**19**] presents two general regression neural network (GRNN) models to predict the changes in retrograde gas condensate composition and to estimate the pressure depletion behavior of gas condensate reservoirs. The first model, GRNNM1, is developed to predict dew point pressure and gas compressibility at dew point using initial composition of numerous samples while the second model, GRNNM2, is developed to predict the changes in well stream effluent composition at any stages of pressure depletion. GRNNM2 can also be used to determine the initial reservoir fluid composition using dew point pressure, gas compressibility at dew point, and reservoir temperature. The study showed that the GRNN models general are accurate, valid, and reliable.

Moghadassi et al. [**35**] described details about the need of ANN for prediction oil/gas properties. EOS are useful for description of fluid properties such as pressure-volume-temperature (PVT). At present, there is no single equation of state that accurately estimates the properties of all substances under all conditions. In that work they proposed a new method based on ANN for estimation of PVT properties of compounds. ANN is a model based on some experimental results that is proposed to predict the required data because of avoiding more experiments. They found minimum Mean Square Error (MSE) of 0.000606 by using ANN with sixty neurons in the hidden layer and conclude that ANN's capability to estimate the PVT properties is one of the best estimating methods with high performance.

Aminzadeh et al. [**23**] highlights the applications of soft computing and artificial intelligence in the oil industry, using geological and geophysical data. The strength and weakness of human intelligence versus machine intelligence and the need for combining human and machine intelligence is pointed out. It is argued that the role soft computing methods (ANN, Fuzzy Logic (FL) and evolutionary computing) that can play a good role in establishing “hybrid” intelligence for addressing E&P problems.

Xie et al. [**36**] have developed a methodology that provides permeability estimates for all rock-types or lithologies, for a wide range of permeability. This is a hybrid Genetic Programming and Fuzzy/Neural Net inference system and which utilizes lithologic and permeability facies as indicators. The results from conducting cross-validation suggest this methodology is robust in estimating permeability in complex heterogeneous reservoirs. Hybrid GP-Fuzzy/Neural system has been shown to be robust in estimating permeability from elastic parameter input. This system yields the estimated permeability that matches core permeability more consistently.

Chang and Chang [**28**] used the ANFIS to build a prediction model for reservoir management. To illustrate the applicability and capability of the ANFIS, the Shihmen reservoir, Taiwan, was used as a case study. They used a large number (132) of typhoon and heavy rainfall events with 8640 hourly data sets collected in past 31 years. To investigate whether this neuro-fuzzy model can be cleverer (accurate) if human knowledge, i.e. current reservoir operation outflow, is provided, they developed two ANFIS models: one with human decision as input, another without. They demonstrated that the ANFIS can be applied successfully and it can provide high accuracy and reliability for reservoir water level forecasting in the next three hours. They consequently found that the model with human decision as input variable has consistently superior performance with regard to all used indexes than the model without this input.

Wafaa and Alaa [**37**] suggested an intelligent technique using FL and ANN to determine reservoir properties from well logs. Fuzzy curve analyses based on fuzzy logic were used for selecting the best related well logs with core porosity and permeability data. ANN is used as a nonlinear regression method to develop transformation between the selected well logs and core measurements. The technique was demonstrated with an application to the well data in West July oil field, Gulf of Suez, Egypt for the Miocene Upper Rudeis reservoirs (Asal and Hawara formations). The results shows that the technique can make more accurate and reliable reservoir properties estimation compared with conventional computing methods. This intelligent technique can be utilized as a powerful tool for reservoir properties estimation from well logs in oil and natural gas development projects.

2.2. Computational Intelligence in Gas Related Study

Sheng-wei Fei et al. [**22**] proposed Support Vector Machine (SVM) with Genetic Algorithm (SVMG) to forecast the ratios of key-gas in power transformer oil, among which GA is used to determine free parameters of support vector machine. The experimental results indicate that the SVMG method can achieve greater accuracy than ANN under the circumstance of small training data. SVMG implements the principle of structural risk minimization in place of experiential risk minimization, which makes it have excellent generalization ability in the situation of small sample. And it can change a non-linear learning problem into a linear learning problem in order to reduce the algorithm complexity by using the kernel function idea. In addition, GA can be used to select suitable parameters to forecast the ratios data of key-gas, which avoids over fitting or under-fitting of the SVM model occurring be-cause of the improper determining of these parameters.

Mohaghegh and Balan [**21**] used their efforts toward the development of a new and novel methodology for optimal design of hydraulic fracture treatments in a gas storage field. A hybrid system that is consisted of two neural networks and a genetic algorithm routine was developed for design and optimization of hydraulic fracturing procedures in a gas storage field in Ohio. The major difference between these systems with conventional two or three dimensional frac simulators was that the developed hybrid system provide a solution for frac treatment design and optimization in the absence of conventional reservoir data that were an absolute necessity when using conventional (2D or 3D) simulators. They used available data, without access to reservoir data such as permeability, porosity, thickness and stress profiles. The hybrid system developed in this study is able to forecast gas storage well deliverability with higher than 95% accuracy. This system is also capable of helping the practicing engineers to design optimum hydraulic fractures. The developed system is currently being used to select candidate wells and to design frac jobs in the aforementioned field.

Ozmen and Tekce [**17**] presented a system which is made of an array of eight phthalocyanine-coated QCM sensors and an ANN to find the corresponding composition of a gas mixture. The digital data collected from the sensor responses were pre-processed by a sliding window algorithm, and then used to train a three layer ANN to determine the gas compositions. The system is tested with the following gas mixtures: (1) ethanol–acetone, (2) ethanol–trichloroethylene, (3) acetone–trichloroethylene. They demonstrated that finding the compositions of gas mixtures using an array of QCM sensors and ANN is possible. The success rate in identifying the constituent component amounts of the approach 84.5% for gas 1, 94.3% for gas 2. Similarly, average prediction errors are 15.5% for gas 1, 5.7% for gas 2 and 10.6% overall. The sensor array and the method developed to process the sensor data in this work is promising for future experiments. Although the system developed in this work is applicable only when a gas mixture belongs to the certain specified categories.

Shokir et al. [**24**] presented a new pure hydrocarbon gas and gas mixture viscosity model over a wide range of temperatures and pressures as a function of gas density, pseudo-reduced temperature, pseudo-reduced pressure, and the molecular weight of pure and hydrocarbon gas mixtures. The new model designed seems to be simpler and eliminated the numerous computations involved in any EOS calculation. The developed new model yields a more accurate prediction of the pure gas and gas mixture viscosity with the lowest average absolute relative error (5.6%) among all tested gas viscosity correlations. They resolved that the GP-modeling approach is capable of estimating the viscosity of pure and hydrocarbon gas mixtures with high accuracy compared to the experimental values. This work could be extended to develop a universal viscosity correlation considering gas condensate and sour natural gas mixtures.

Ilkhchi et al. [**38**] proposed an optimal and improved model to make a quantitative and qualitative correlation between Normalize Oil Content (NOC) and well log responses by integration of neural network training algorithms and the committee machine concept. This committee machine with training algorithms (CMTA) combines Levenberg–Marquardt (LM), Bayesian regularization (BR), gradient descent (GD), one step secant (OSS), and resilient back-propagation (RP) algorithms. Each of these algorithms has a weight factor showing its contribution in overall prediction. The optimal combination of the weights is derived by a genetic algorithm. They performed a case study where 231 data composed of well log data and measured NOC from three wells of South Pars Gas Field were clustered into 194 modelling dataset and 37 testing samples for evaluating reliability of the models. The result shows that the CMTA provides more reliable and acceptable results than each of the individual neural networks differing in training algorithms. Also CMTA can accurately identify production pay zones (PPZs) from well logs.

2.3. RESEARCH IN ENSEMBLE

Polikar [**8**] discussed about bootstrap-inspired techniques in CI, specifically in ensemble of classifiers based algorithms. The crux in this article is to generate an ensemble of diverse classifiers, where each classifier is trained on a strategically selected bootstrap sample of the original data. Pragmatically he discussed the ability of bootstrap-based approaches so as to signify the outcomes of implementations of such approaches on a variety of real-world problems. He used several examples of these algorithms that create strong classifiers from an ensemble of weaker ones. Such algorithms make good use of small datasets by training multiple classifiers on bootstrap samples of the available data. As a result he concluded that new ensemble is generated using each new dataset, where individual classifiers are trained with bootstrapped samples of the training data, whose distribution is adjusted to ensure that the novel information is efficiently learned.

Zhao et al. [**39**] performed a constructive survey on the ANN ensembles, including effective analysis and general implement steps of ensembles. Compared with a single ANN, the ensemble is able to efficiently improve the generalization ability of the classifier work. They concluded that ensemble of network can improve the generalization performance of a classification system greatly. Furthermore the availability of local minima in the individuals in neural network ensemble are expected to have different local minima of error surface thus increased the diversity of ensemble. They mentioned that the challenge of Ensemble researchers is how to effectively design the individual works that is not only highly correct, but also different as much as possible.

He and Shen [**40**] used a bootstrap methods for time-series prediction are used to construct multiple learning models, and then use a combination function to combine the output of each model for the final predicted output. ANN model as the base learning algorithm and applied this approach to the foreign currency exchange rate predictions. Both daily prediction and weekly prediction indicate that the proposed method can significantly improve the forecasting performance compared to the traditional single neural network based approach. After training, testing points are sent to every ANN model and a combination function is used to combine the outputs from individual neural networks

Lai et al. [**41**] proposed a new nonlinear ANN ensemble model for financial time series forecasting. It starts with the generation of many different neural networks then they used the principal component analysis technique is used to select the appropriate ensemble members. They did ANN ensemble using SVR method. Testing was done using two real financial time series. A novel triple-phase nonlinear ensemble predictor for financial time series forecasting is proposed. The effectiveness of the proposed nonlinear ensemble approach, implying that the proposed nonlinear ensemble model can be used as a feasible approach to financial time series forecasting is demonstrated experimentally.

Dong and Han [**42**] used ensemble methods for weak classifiers and whether they are effective for strong classifiers is not clear. SVM had been the state-of-the- art performance for the Text Classification (TC) tasks. Due to the complexity of the TC problems, it becomes a challenge to systematically develop classifiers with better performance. They deployed five types of data partitioning ensemble of SVMs were experimentally compared on two well-accepted benchmark collections, and they found that disjoint partitioning ensembles of SVMs with stacking performed best and consistently outperformed the single SVM. They also found that bagging and cluster partitioning ensembles are not effective to combine strong classifiers like SVM, and boosting always achieves worse results on all of the collections.

Melville et al. [**43**] compared the sensitivity of bagging, boosting, and decorate to three types of imperfect data: missing features, classification noise, and feature noise. In comparing bagging, boosting and decorate, bagging is quite sensitive while boosting is fairly robust but that decorate is constructs diverse committees using artificial data. It has been shown to generally outperform both boosting and bagging when training data is limited. For missing data, they found that Decorate is the most robust. For classification noise, bagging and Decorate are both robust, with bagging being slightly better than Decorate, while boosting is quite sensitive. For feature noise, all of the ensemble methods increase the resilience of the base classifier. They concluded that Bagging performs the best at combating high amounts of classification noise. In the presence of noise in the features, all ensemble methods produce consistent improvements over the base learner.

Chen [**31**] focused mainly on the diversity among ensemble members and the regularization. He proved that diversity highly correlates with the generalization error only when diversity is low, and the correlation decreases when the diversity exceeds a threshold. He investigated error diversity in x using negative correlation learning (NCL) in detail. This provides a Bayesian formulation of RNCL and implements RNCL by two techniques: gradient descent with Bayesian Inference and evolutionary multi-objective algorithm. According to him the numerical results demonstrate the superiority of RNCL. Left-truncated Gaussian is used by him prior for this probabilistic model to obtain a set of sparse and non-negative combination weights. He summarized various selection-based and weight-based algorithms for ensemble pruning, which aims to reduce the size of ensemble and simultaneously improve the generalization performance by balancing diversity, regularization and accuracy in the ensemble.

Pasquariello et al. [**44**] presented a comparison of classification strategy based on the combination of the outputs of a ANN ensemble and the application of SVM classifiers in the analysis of remotely sensed data. On analysis they proved that the non linear, Multi-Layer Perceptron (MLP) based, combination provides the best results among the different combination schemes. This method gave a combination error lower than that of the best classifier in the ensemble. A performance enhancer can be obtained by using a non linear combiner, such as the MLP neural network: the value of the combination error was the lowest. The application of a further MLP module to combine the outputs of the ensemble helps to overcome some of the main limitations of the generalization capability of each single module in the ensemble. When a more transparent formalism is required in understanding why a combination scheme is better than another and in what circumstances, the Bayesian and the error correlation matrix are the preferable techniques for selecting the coefficient of the linear combination. Since they are more robust with respect to the generalization issue and give the similar results like MLP.

Redondo et al. [**45**] proposed two new ensemble combiners based on the Mixture of Neural Networks model. Two different network architectures on the methods based on the Mixture of Neural Networks: the Basic Network (BN) and the Multilayer Feed forward Network (MF) is incorporated experimentally. A comparison of the mixture combiners was proposed by them with three different mixture models and other traditional combiners are presented. The results show that the mixture combiners proposed are the best way to build multi-net systems among the methods studied in the paper in general. The two new combiners are applied to ensembles of Multilayer Feed forward networks previously trained with Simple ensemble. In first which is Mix-SE-BN, Basic Network as gating network is applied to weight and combine the outputs provided by the networks of the ensemble previously trained with Simple Ensemble. In the second one, Mix-SE-MF, Multilayer Feed forward network is applied as gating network to combine the ensemble previously trained with Simple ensemble. In experiments the first mixture model, Mix-BN-BN, the Basic Network is used as expert and gating networks. In the second, Mix-MF-BN, the Multilayer Feed forward network is used as expert networks whereas the Basic Network is used as gating network. In the last one, Mix-MF-MF, the Multilayer Feed forward network is used as expert and gating networks. To compare the combiners proposed with the seven traditional combiners, they have used ensembles of 3, 9, 20 and 40 networks previously trained with Simple Ensemble. After comparing the two sets mean Increase of Performance and the mean Percentage of Error Reduction are calculated with respect to a single MF network to compare all the methods. It was found by them, the mixture combiners on Simple Ensemble are the best way to build Multi-Net systems among the models and combiners studied, also the combiners proposed are more robust than the traditional ones. Similar results are obtained in other cases too. It is proved that the accuracy of an ensemble of Multilayer feed forward networks can be improved by applying the gating network of the Mixture of ANNs as ensemble combiner.

Zhou et al. [**46**] analyzed the relationship between the generalization ability of the neural network ensemble and the correlation of the individual neural networks, which reveals that ensembling a selective subset of individual networks is superior to ensembling all the individual networks in some cases. An algorithm called GASEN is proposed by them, which trains several individual neural networks and then employs genetic algorithm to select an optimum subset of individual networks to constitute an ensemble. Comparing with a popular ensemble approach, i.e. averaging all, and a theoretically optimum selective ensemble approach, i.e. enumerating, GASEN has preferable performance in generating ensembles with strong generalization ability in relatively small computational cost is proved experimentally.

Wen et al. [**47**] investigated whether a hybrid approach combining different stock prediction approaches together can dramatically outperform the single approach and compare the performance of different hybrid approaches. The hybrid model includes three well-researched prediction algorithms: back propagation neural network (BPNN), ANFIS and SVM They were utilized independently to single-step forecast the stock price, and then they were integrated into a final result by a combining strategy. Two different combining methods are investigated by them. The first method is a linear combination of the three forecasts. The second method combines them by a neural network. Combining the single algorithm considerately, a better performance can be received is verified experimentally. A number of soft computing approaches have successfully applied in the prediction of stock price and showed good performance.

Aljahdali et al. [**48**] explored GA as an alternative approach to derive different software reliability models. GA is a powerful machine learning and optimization techniques to estimate the parameters of well known reliably growth models. The reason of choosing GA for this task is its capability of estimating optimal parameters through learning from historical data. Experiments were conducted to confirm these hypotheses by evaluating the predictive capability of the developed ensemble of models and the results were compared with traditional models. Predictability of software reliability using ensemble of models trained using GA arte measured. The study is applied on three study sets; Military, Real Time Control and Operating System. In comparison to the predictability of the single AR model and ensemble of AR models trained by GA algorithm over the trained and test data is concerned, the ensemble of models performed better the single model. Also, they found that the weighted average combining method for ensemble has a better performance in a comparison with average method. This due to the GA learned weights which decide the contribution of each model in the final results.

Zainal et al. [**49**] used an ensemble of one-class classifiers where each uses different learning paradigms. Three techniques are incorporated which are: Linear Genetic Programming (LGP), ANFIS and Random Forest (RF). The strengths from the individual models were evaluated by them and ensemble rule was formulated. Empirical results show an improvement in detection accuracy for all classes of network traffic; Normal, Probe, DoS, U2R and R2L. RF was also able to address imbalance dataset problem that many of machine learning techniques fail to sufficiently address it. Ensemble of different learning paradigms can improve the detection accuracy is demonstrated in this paper. This was achieved by assigning proper weight to the individual classifiers in the ensemble model. Using experimentation they found out that, LGP has performed well in all the classes except the U2R attacks. In contrary, RF shows a better true positive rate for U2R class. Thus, by including the RF in the assemble model, the overall performance particularly the result for U2R class has improved.

Chandra and Yao [**50**] tried to come up with a co-evolutionary framework with a view to synthesize evolutionary ensemble learning algorithms. Ensembles of learning machines have been outperforming single predictors in many cases. This happens usually when they constitute members which form a diverse and accurate set. Keeping in mind they developed a multi-objective evolutionary optimization as a formidable ensemble construction technique. In addition to presenting detailed empirical results and comparisons with a wide range of algorithms in the machine learning literature in this paper they tried to explicate on the intricacies of the proposed framework. All ensemble learning methods are essentially based on a very simple idea which is their goal of having diverse and accurate members within them which help them to outperform single learners. An algorithm called DIVACE is also proposed by them with an idea to enforce diversity and accuracy within the ensemble explicitly within a multi-objective evolutionary setup. This becomes an evolutionary framework that uses a myriad of diversity enforcement ideas rolled into one evolutionary ensemble learning algorithms in terms of the average test error rates for the Australian credit card assessment dataset.

CHAPTER 3

OVERVIEW OF COMPUTATIONAL INTELLIGENCE TECHNIQUES

3.1. ARTIFICIAL NEURAL NETWORK

Artificial Neural Network (ANN) is a machine learning approach inspired by the biological nervous systems that mimic human brain performing a particular learning task. ANN is parallel computing systems consisting of large number of simple processing unit called "*neuron*", similar to neuron of brain with many connections. Each node is characterized by a node function with fixed or adjustable parameters. The node function is called "*Activation function*" is used for scaling the output of each neuron. The architecture of ANN depends on the pattern of connections between the neurons. A *learning algorithm* is required to determine the weights on the connections. One of the main properties of ANNs is their ability to learn from data. In general, the available data can be divided into two parts: one part for training, and the other for testing. The training phase of a ANN is a process to determine optimum parameters values to sufficiently fit the training data. The basic learning rule is the well-known back-propagation method which seeks to minimize some measure of error, usually a sum of squared differences between a network outputs and desired outputs. When the test error is much larger than the training error, then an over-fitting problem has occurred. “Over-training” diminishes the forecasting capability of the network due to its structure which is excessively adapted to the training data. In order to be trained a network architecture is needed to be prepared. By using the training algorithm the complex relationship is learned by the network from the input and output data of the given problem. The model performance is always checked by means of distinct test data, and a relatively good fitting is expected, especially in the testing phase. Methods of learning can be of two categories: *Supervised Learning* and *Unsupervised Learning* depend on the availability of information. In the supervised learning, the weights are usually obtained by minimizing some error function which measures the difference between the desired output values and those computed by the neural network. In unsupervised learning, the data is presented to the network without any external information and the network must discover by itself the patterns.

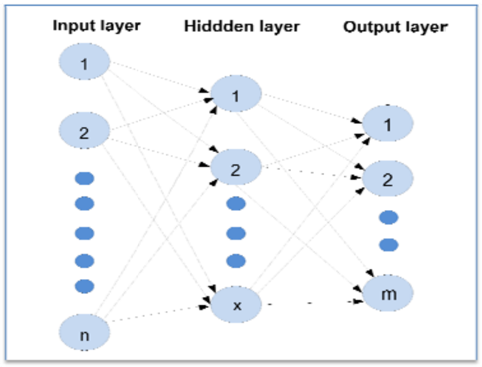


Figure 8: Multi-Layer Perceptron

Multi-Layer Perceptron (MLP) is one of the widely used ANN that has gained vast popularity in many research areas [**14**,**20**,**35**]. MLP has one input layer, one output layer, and one or more hidden layers of processing units and no feed-back connections (Figure 8). The hidden layers sit in between the input and output layers, and are thus hidden from the outside world. It can be learn a particular function from the input and output relation by adjusting the values of the connections (weights) between neurons. Typically, MLP is adjusted, or trained, so that a particular input leads to a specific target output. The weights are adjusted, based on a comparison of the output and the target, until the network output matches the target. Typically, many such input/target pairs are needed to train a network. In general ANN has been trained to perform complex functions in various fields including petroleum, pattern recognition, pattern, identification, classification, speech recognition, computer vision, control systems, etc. Figure 9 shows a neuron with sigmoidal activation function where,

$$a = \sum_{j=1}^{m} x_j(n) w_j(n)$$

$$y = \sigma(a) = \frac{1}{\left(1 + e^{-a}\right)}$$

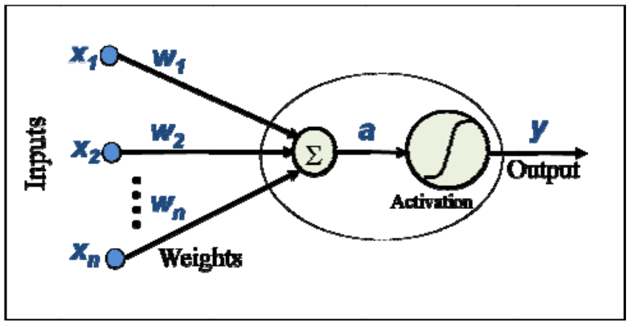


Figure 9: Neuron with Sigmoid-Function

Figure 10 shows two kind of activation function used in this dissertation.

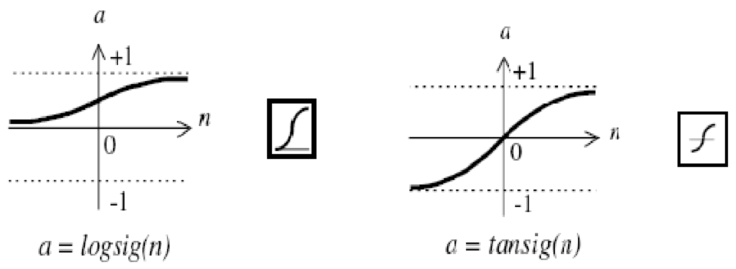


Figure 10: log-sigmoidal and tan-sigmoidal Activation Function

3.2. SUPPORT VECTOR REGRESSION

Support Vector Machines (SVMs) were first introduced by Boser et al. at the COLT 1992 conference [**51**]. In 1995 the soft margin classifier was introduced by Cortes and Vapnik [**52**]. In the same year the algorithm was extended to the case of regression by Vapnik in *The Nature of Statistical Learning Theory* [**53**]. Support Vector Regression (SVR) as a regression version of SVMs. The main idea is always the same for both SVR and SVM which is to minimize error and individualizing the hyper-plane which follows the maximum margin algorithm: a non-linear function is leaned by linear learning machine mapping into high dimensional kernel induced feature space. The capacity of the system is controlled by parameters that do not depend on the dimensionality of feature space. Margin is a distance between optimal hyper-plane and a vector which lies closest to it. The decision boundary (optimal hyper-plane) should be as far away from the data of both classes as possible (Figure 11).

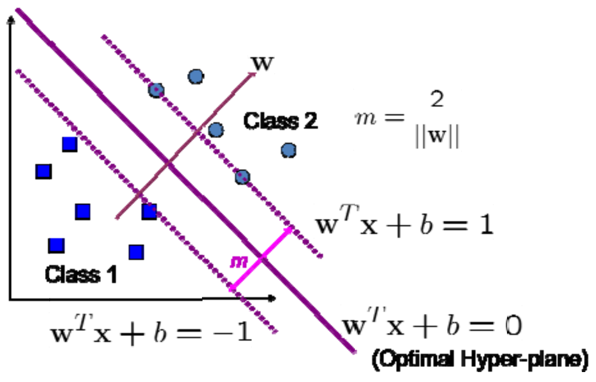

w
Class 2
$m = \frac{2}{||\mathbf{w}||}$
$\mathbf{w}^T\mathbf{x} + b = 1$
Class 1
m
$\mathbf{w}^T\mathbf{x} + b = -1$
$\mathbf{w}^T\mathbf{x} + b = 0$
(Optimal Hyper-plane)

Figure 11: Hyper-plane & Margin of SVM

One of the advantages of SVM/SVR as the part of it is that, it can be used to avoid difficulties of using linear functions in the high dimensional feature space and optimization problem is transformed into dual convex quadratic programs. In regression case the loss function is used to penalize errors that are greater than threshold - ε. Such loss functions usually lead to the sparse representation of the decision rule, giving significant algorithmic and representational advantages. SVM/SVR maps input vectors to a higher dimensional feature space via non-linear mapping (Φ) where a maximal separating hyper plane is constructed by performing linear regression in this space [**51**,**53**]. This is shown in Figure 12. The Figure 12 (a) shows the two dimensional data having the circular decision boundary which is linearly non-separable. The Figure 12 (b) shows the mapping of the data into three dimensional spaces where the circular decision boundary becomes a linear hyper-plane. The Figure 12 (c) shows the two dimensional projection of the Figure 12 (b).

Figure 12 (a, b, c): Support Vector Machine [54]

3.2.1. Insensitivity Zone

In the case of regression, a margin of tolerance ε is set in approximation to the SVR which would have already being inferred from the problem. The generalization ability of SVR is ensured by special properties of the optimal hyper-plane that maximizes the distance to training examples in a high dimensional feature space. They relied on defining the loss function that ignores errors, which are situated within the certain distance of the true value. This type of function is often called – epsilon intensive – loss function. In Figure 13 only the points outside the shaded region contribute to the cost insofar, as the deviations are penalized in a linear fashion. Figure 13 depicts the variation of model performance with sizes of the tube.

Parameter ε controls the width of the ε-insensitive zone, used to fit the training data. The value of ε can affect the number of support vectors used to construct the regression function. The bigger ε, the fewer support vectors are selected (Figure 14). Support vectors are points that lie on the boundary or outside the tube. On the other hand, bigger ε-values results in more 'flat' estimates. It is obvious that the thinner the "tube", the more complex the model.

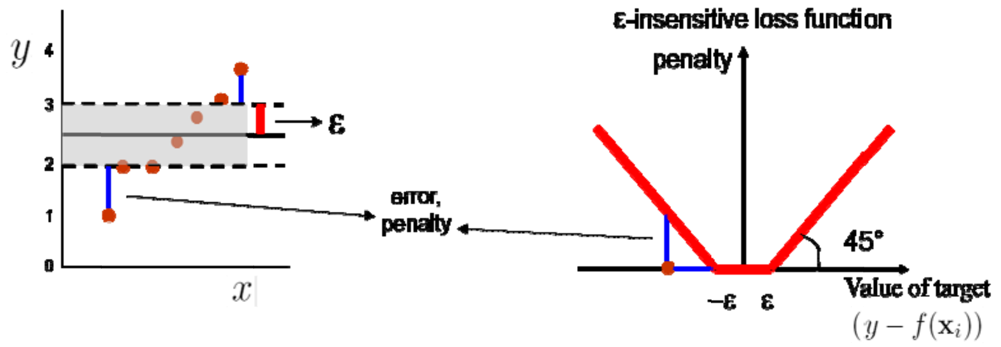
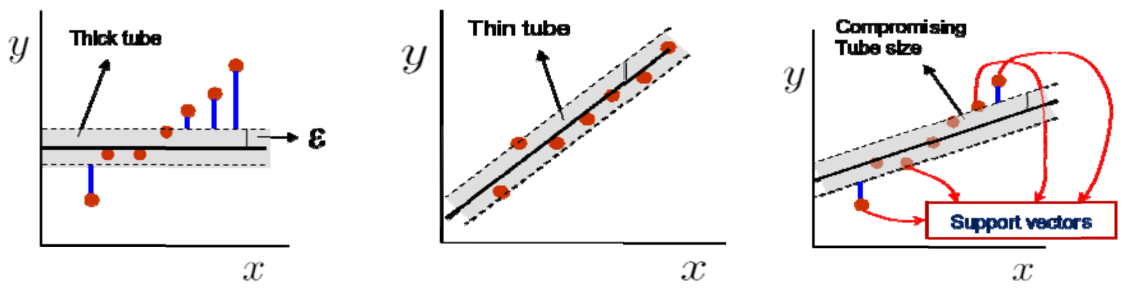


Figure 13: ε – insensitive loss function

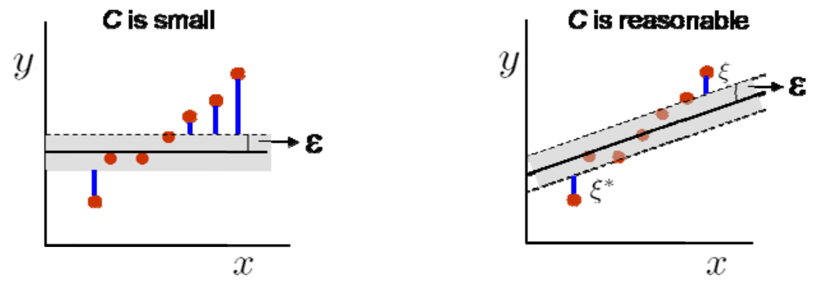


(a): Under-fitting (b): Over-fitting (c): Good Generalization

Figure 14: Effects of ε tube in SVR

3.2.2. Regularization Parameter

The regularization parameter C determines the tradeoff between the model complexity (flatness) and the degree to which deviations larger than ε are tolerated in optimization formulation for example, if C is too large (infinity), then the objective is to minimize the empirical risk only, without regard to model complexity part in the optimization formulation. Hence, both C and ε values affect model complexity (but in a different way). Figure 15 shows the effect of C in SVR.



(a): Under-fitting (b): Good Generalization

Figure 15: Effect of C in SVR

3.2.3. Linear Support Vector Regression

Linear SVR perform linear regression in the feature space. Unlike in least square regression, the error function is ε-insensitive loss function. Instinctively, mistake less than ε is ignored and thus leads to sparseness similar to SVM. The Figure 16 shows an example of one-dimensional linear regression function with – epsilon intensive – band. The variables measure the cost of the errors on the training points. These are zero for all points that are inside the band.

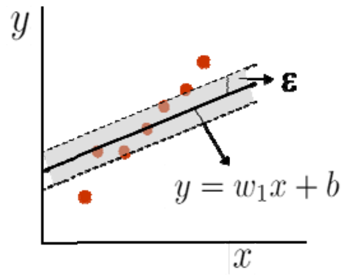


Figure 16: Linear SVR

3.2.4. Non-Linear Support Vector Regression

Non-Linear SVR performs the same way of linear SVR. The only difference is that the data input space *x* transform into a *higher*-dimensional feature space Φ(*x*) where a hyper-plane can be constructed. Linear operation in the feature space is equivalent to non-linear operation in input space. But mapping involves high computational burden and hard to get a good estimate.

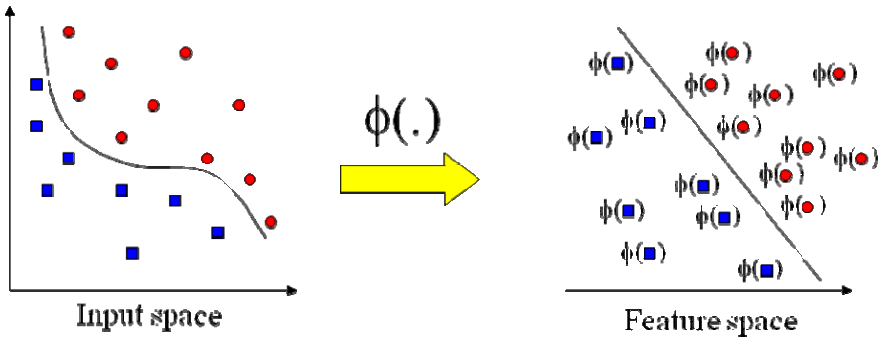
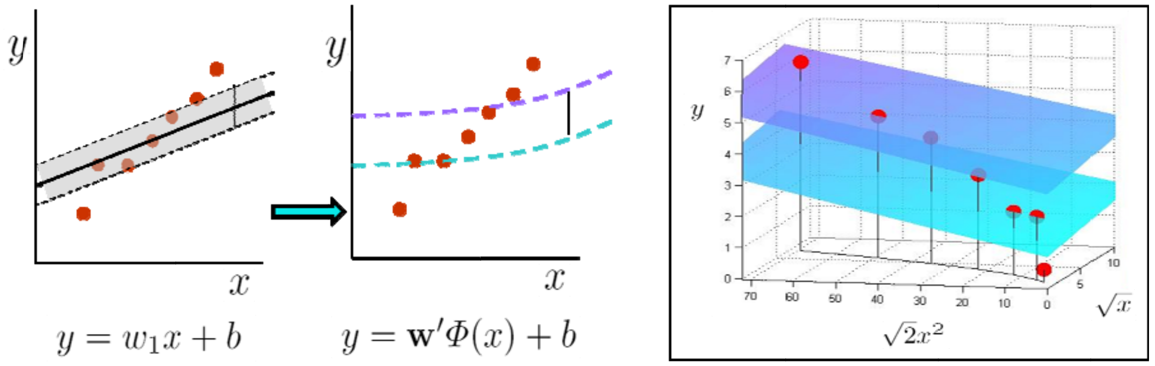


Figure 17: Input Space Transformation

Figure 17 shows the transformation of input space into higher dimension where data can be separated linearly. Figure 18 (a) shows non-linear SVR of the corresponding linear

SVR. Figure 18 (b) shows the two hyper-planes and ε-insensitive zone is between this two planes. The optimal hyper-plane resides exactly the same distance of this two planes.



(a): Non-linear SVR (b): 3D View of Hyper-planes

Figure 18: Non-linear SVR

3.2.5. The standard formulation of the SVR model

Suppose a given training dataset $\{x_i, y_i\}_{i=1}^{n}$ where $x \in \mathbf{R}^d$ and $y \in \mathbf{R}$, and a nonlinear mapping to a higher dimensional space $: \mathbf{R}^d \to \mathbf{R}^h$ where $d \le h$. SVR finds the optimal value of w and b such that $f(x) = w_i \phi_i(x) + b$, Where $\phi_i(x)$ is the feature of inputs x and both w and b are coefficients. In ε-SVR [**53**] goal is to find a function $f(x)$ that has at most ε deviation from the actually obtained targets y_i for all the training data, and at the same time is as flat as possible. The errors are not taken into account as long as they are less than ε, but will not accept any deviation larger than this. A real life example can be not to lose more than ε money when dealing with exchange rates. The SVR function is

$$f(x) = w_i \emptyset(x_i) + b \qquad \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots(1)$$

In linear regression, we actually minimize the error function

$$R = \frac{1}{2}\sum_{i=1}^{N}\{f(x_i) - y_i\}^2 + \frac{\lambda}{2} \parallel w \parallel^2$$

By replacing the quadratic error function by ε-insensitive error function we get

$$R_{reg}(C) = C\sum_{i=1}^{N} E_\varepsilon(f(x_i) - y_i) + \frac{1}{2} \parallel w \parallel^2$$

Where the error function is

$$E_\varepsilon(f(x_i) - y_i) = \begin{cases} 0 & \text{for } |f(x_i) - y| < \varepsilon \\ |f(x_i) - y| - \varepsilon & \text{otherwise} \end{cases}$$

For a target point to lie inside the ε tube

$$f(x_i) - \varepsilon \le y_i \le f(x_i) + \varepsilon \qquad \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots(2)$$

Our target is to find a flat model. The flatness in (1) means that a small *w* is needed. One way to ensure is to minimize the norm of *w* i.e. $\|w\|^2$. This problem can be written as

Minimize $\frac{1}{2} \parallel w \parallel^2$

Subject to

$$\begin{cases} y_i - w_i \emptyset(x_i) - b \leq \varepsilon \\ w_i \emptyset(x_i) + b - y_i \geq \varepsilon \end{cases} \qquad \text{......................................(3)}$$

However this may not be the case always. We may need to allow some errors. Simply we need a soft margin that allows points lie outside the tube [Figure 14]. For this we need to introduce slack variables in Eq. (2)

$$y_i \leq f(x_i) + \varepsilon + \xi_i$$

$$y_i \geq f(x_i) - \varepsilon - \xi_i^* \qquad \text{....................................(4)}$$

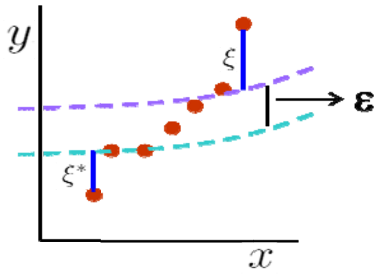


Figure 19: Soft Margin of non-linear SVR

Hence the formulation of Eq. (3) can be written as in Vapnik [**53**]

Minimize

$$\frac{1}{2} \parallel w \parallel^2 + C \sum_{i=1}^{n} (\xi_i + \xi_i^*)$$

Subject to

$$\begin{aligned} & y_i - w_i \emptyset(x_i) - b \leq \varepsilon + \xi_i \\ & w_i \emptyset(x_i) + b - y_i \leq \varepsilon + \xi_i^* \\ & \xi_i, \xi_i^* \geq 0 \end{aligned} \qquad \cdots\cdots\cdots\cdots(5)$$

The constant $C > 0$ determines the trade-off between the flatness of f and the upper bound of the deviations larger than ε will be tolerated. In most cases the optimization problem of Eq. (5) can be solved more easily in its dual formulation. The key idea is to construct a Lagrange function from the primal objective function and the corresponding constraints, by introducing a dual set of variables. This constrained optimization problem is solved using the following primal Lagrangian form:

$$\begin{aligned} L = \frac{1}{2} \parallel w \parallel^2 + C \sum_{i=1}^{n} (\xi_i + \xi_i^*) - \sum_{i=1}^{n} (\eta_i \xi_i + \eta_i^* \xi_i^*) \\ - \sum_{i=1}^{N} \alpha_i^* (\varepsilon + \xi_i^* - y_i + w_i \emptyset(x_i) + b) \\ - \sum_{i=1}^{N} \alpha_i (\varepsilon + \xi_i - y_i + w_i \emptyset(x_i) + b) \end{aligned} \qquad \cdots\cdots\cdots\cdots(6)$$

The Saddle point of L has to be found by minimization with respect to w, b, ξ_i, ξ_i^* and maximization with respect to langrage multipliers $\alpha_i, \alpha_i^*, \eta_i, \eta_i^*$. These dual variables have to satisfy positivity constraints, i.e. $\alpha_i^{(*)}, \eta_i^{(*)} \geq 0$. It follows from the saddle point condition that the partial derivatives of L with respect to the primal variables w, b, ξ_i, ξ_i^* have to vanish for optimality.

$$\frac{\partial L}{\partial w} = 0 \Rightarrow w - \sum_{i=1}^{n} (\alpha_i - \alpha_i^*) K(x_i, x_j) = 0 \qquad \cdots\cdots(7)$$

$$\frac{\partial L}{\partial b} = 0 \Rightarrow \sum_{i=1}^{n} (\alpha_i - \alpha_i^*) = 0 \qquad \cdots\cdots(8)$$

$$\frac{\partial L}{\partial \xi_i} = 0 \Rightarrow \alpha_i + \eta_i = C \qquad \cdots\cdots(9)$$

$$\frac{\partial L}{\partial \xi_n^*} = 0 \Rightarrow \alpha_i^* + \eta_i^* = C \qquad \cdots\cdots(10)$$

Now Eq. (6) can be rewritten as dual optimization problem from Eq. (7), (8), (9) and (10) as follows

Maximize

$$-\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}(\alpha_i-\alpha_i^*)\left(\alpha_j-\alpha_j^*\right)K\left(x_i,x_j\right)$$

$$-\varepsilon\sum_{i=1}^{n}(\alpha_i+\alpha_i^*)+\sum_{i=1}^{n}(\alpha_i-\alpha_i^*)y_i$$

Subject to

$$\sum_{i=1}^{n}(\alpha_i-\alpha_i^*)=0$$

$$0\le a_n\le C$$

$$0\le a_n^-\le C \qquad \cdots\cdots(11)$$

In deriving Eq. (11) the dual variables η_i , η_i^* is eliminated by the condition of Eq. (9) and (10) which can be reformulated as $\eta_i = C-\alpha_i$ and $\eta_i^* = C-\alpha_i^*$. After calculating α_i *and* α_i^* In Eq. (11) the optimal desired weights vector of the regression hyper-plane is represented as

$$w^*=\sum_{i=1}^{n}(\alpha_i-\alpha_i^*)K\left(x_i,x_j\right) \qquad \cdots\cdots(12)$$

Therefore the regression equation would be

$$f(x,\alpha,\alpha^*)=\sum_{i=1}^{n}(\alpha_i-\alpha_i^*)K\left(x_i,x_j\right)+b \qquad \cdots\cdots(13)$$

Here, $K(x_i, x_j)$ is called the kernel function. The value of the kernel is equivalent to the inner product of two vectors x_i and x_j in the feature space $\emptyset(x_i)$ and $\emptyset(x_j)$. Therefore, $K(x_i, x_j) = \emptyset(x_i) \times \emptyset(x_j)$. The inner product can be computed by K without going through the map $\emptyset(.)$ which is also known as *kernel trick*. In practice, we specify K, thereby specifying $\emptyset(.)$ indirectly. Intuitively, $K(x_i, x_j)$ represents our desired notion of similarity between data x_i and x_j and this is from our prior knowledge. Any function that satisfies Mercer's condition can be used as the Kernel function by [**53**]. The Polynomial and Gaussian are the most widely used kernel function. Few standard kernels are provided below

Linear : $$K(x, x_i) = \langle x, x_i \rangle$$

Polynomial: $$K(x, x_i) = \langle x, x_i \rangle^d$$

Gaussian: $$K(x, x_i) = exp\left[-\frac{\| x - x_i \|^2}{2\sigma^2}\right]$$

3.3. Adaptive Neuro-Fuzzy Inference System

In recent years a new branch of CI named "soft computing" aims to integrate the power of Artificial Neural Network (ANN) and Fuzzy Inference Systems (FIS). ANN possesses exciting capabilities such as learning, adaptation, fault-tolerance, parallelism and generalization whereas FIS performs an inference mechanism under cognitive uncertainty

[**4**]. To enable a system to deal with cognitive uncertainties in a manner more like humans, the concept of ANN can be incorporated into fuzzy logic. The resulting hybrid system is called a neuro-fuzzy network [**55**]. Adaptive Neuro-Fuzzy Inference System (ANFIS) is claimed as a universal approximator to represent highly non-linear functions more powerfully than conventional statistical methods [**56**]. Intelligence methodologies such as Neuro- Fuzzy inference is a method that interprets the relationship between input and output by means of some set of fuzzy ‘‘IF-THEN’’ rules e.g.

IF X is A THEN Y is B

Where A and B are labels of fuzzy sets, e.g. “hot”, “cold”. Each fuzzy set is characterized by appropriate membership functions that map each element to a membership value between 0 and 1. The “IF” part is called antecedent and the “THEN” part is called consequent of a rule can have multiple parts linked by Boolean operators (AND, OR) which have equivalent fuzzy operators (MIN, MAX). A fuzzy inference system are composed of “rule base” signifying fuzzy rules, a “database” defining the membership functions of the fuzzy sets, and a “reasoning mechanism” which performs the inference procedure (Figure 20).

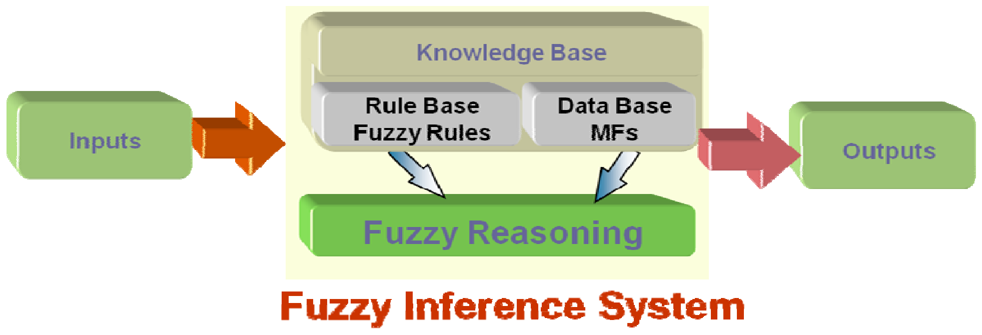


Figure 20: Fuzzy Inference System

Among various fuzzy inference systems, Tagaki–Sugeno system is more suitable for sample-data based fuzzy modeling [**57**] in which the output of each rule is predetermined function of input variables. To give an example, in a first-order Sugeno model with two inputs (x,y), the i_{th} rule is described as

Rule 1: IF x is A_1 AND y is B_1 ***THEN $f_1=p_1x+q_1y+r_1$***

Rule 2: IF x is A_2 AND y is B_2 ***THEN $f_2=p_2x+q_2y+r_2$***

Where variable x and y stands for the fuzzy sets corresponding to the domain of each linguistic label, and p_i is a set of adjustable parameters. The final output, f is the weighted average of each rules $f = \sum_i \overline{w_i} f_i$ where w_i is the weight of the i_{th} rule. The Figure 21 shows the first order Sugeno FIS and the corresponding ANFIS model.

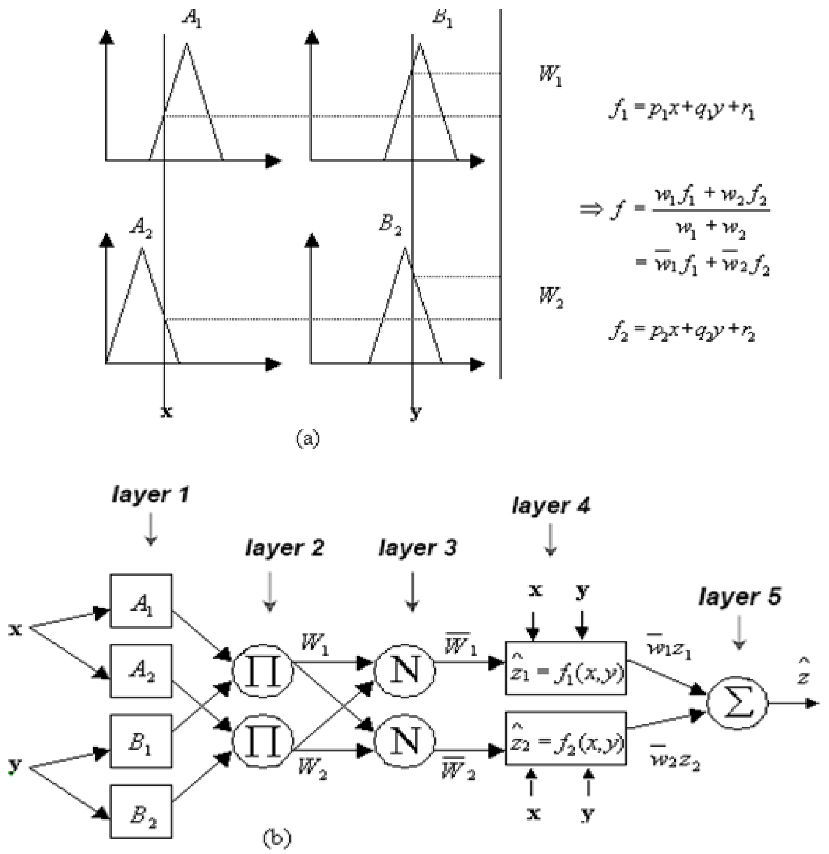


Figure 21: (a) First Order Sugeno Fuzzy Model, (b) Corresponding ANFIS Network Architecture.

While modeling FIS, it is difficult to decide the shape of the membership functions simply by observing the data. These parameters can be chosen to adapt the membership functions by the variation of the input/output data, rather than choosing the parameters arbitrarily associated with a given membership function. This is where the so-called neuro-adaptive learning technique incorporated into ANFIS can help. Let us assume a FIS with two inputs x, y and one output z with the first order of Sugeno Fuzzy Model is shown in Figure 21 (a), the reasoning mechanism can be implemented into a feed-forward neural network with supervised learning capability, which is known as ANFIS architecture (Figure 21 (b)). Jang et al. [**56**] developed a hybrid-learning rule for ANFIS which is faster than the classical back-propagation method by combining the gradient method and the least squares estimate to identify antecedent and consequent parameters. The square nodes in Figure 21 (b) indicate adaptive nodes with parameters and circle modes indicate fixed nodes without parameters. ANFIS basically implements a first order Sugeno-style fuzzy system. Although it is quite easy to express linguistically the relation between input and output, it is difficult to fit the fuzzy model to the target data using trial and error. A better approach is to approximate the target function with a piece-wise linear function and interpolate, in some way, between the linear regions. In the *Takagi-Sugeno* model the idea is that each rule in a rule base defines a region for a model, which can be linear. This is achieved by clustering the input space [**58**]. We have used subtractive clustering to create initial FIS and then trained that FIS using ANFIS hybrid learning algorithm. The functionality of nodes in ANFIS, as a five layered feed-forward neural structure layers can be summarized as follows:

Layer 1: The first layer consists of square nodes that perform fuzzification with chosen membership function. The parameters in this layer are called premises (antecedent) parameters. Nodes in this layer are adaptive. Membership functions of input variables are used as node functions.

$$O_{1,i} = \mu_{A_i}(x) \qquad for\, i = 1,2$$

$$O_{1,i} = \mu_{B_{i-2}}(y) \qquad for\; i = 3,4$$

Where

$$\mu_A(x) = \frac{1}{1 + \left| \frac{x - c_i}{a_i} \right|^{2b_i}}$$

$$\mu_B(y) = \frac{1}{1 + \left| \frac{y - c_i}{a_i} \right|^{2b_i}}$$

Layer 2: In the second layer, the T-norm operation is performed to produce the firing strength of each rule. Nodes in this layer are fixed with outputs and the T-norm operator perform fuzzy AND operation.

$$O_{2,i} = w_i = \mu_{A_i}(x)\mu_{B_i}(y), \quad i = 1,2$$

$$O = \prod_{j=1}^{n} \mu(x_j) = \mu(x_1).\mu(x_2).....\mu(x_j)....\mu(x_n)$$

Layer 3: In the third layer the nodes are fixed with outputs generating the normalized firing strengths by calculating the ratio of the i_{th} rule firing strength to the sum of all rules' firing strength is calculated in the third layer.

$$O_{3,i} = \overline{w_i} = \frac{w_i}{w_1 + w_2}$$

Layer 4: The fourth layer consists of square nodes that perform multiplication of normalized firing strengths with the corresponding rule. The parameters in this layer are called consequent parameters. Nodes are adaptive with node function given by Layer 1 for a first-order model, and with parameters referred to as defuzzifier or consequent parameters.

$$O_{4,i} = \overline{w_i} f_i = \overline{w_i}(p_i x + q_i y + r_i)$$

Layer 5: The fifth layer the single node is fixed with output which calculated by the sum of all incoming signals in the fifth layer.

$$O_{5,i} = F = \sum_i \overline{w_i} f_i = \frac{\sum_i w_i f_i}{\sum_i w_i}$$

Table 3: Two passes in the hybrid learning procedure in ANFIS

ANFIS with Hybrid Learning	**Forward Pass**	**Backward Pass**
Premise Parameters (nonlinear)	Fixed	Gradient descent
Consequent parameters (linear)	Least-square estimator	Fixed
Signals	Node outputs	Error signals

3.4. EVOLUTIONARY ALGORITHM

Evolutionary Algorithms (EA) is a population-based optimization technique. It operates on a population of potential solutions to produce a better solution. The basic idea is to represent every individual of the potential solution as an array of sequences of strings, *chromosomes*. Each string in the chromosome is called a *gene* and the position of a gene is called its *locus*. The values that genes might take are called *alleles.* The initial population of the potential solutions is created randomly and it evolves according to processes that are based on natural evolution, such as selection, recombination or crossover, and mutations. During these operations, which are called *evolutionary operations*, every chromosome in the population is evaluated and receives a fitness value representing an objective or a fitness function. According to their fitness values, the most successful chromosomes are selected for the crossover process to produce new offspring that might have better fitness values. The mutation process is applied to add diversity to the potential solutions. An evolutionary algorithm is characterized by the following five components:

(1) *Encoding:* a mechanism to represent the population of potential solutions.
(2) *Initialization:* a mechanism to create the initial population of the potential solution.
(3) *Fitness function:* an objective function or evaluation function that is used to assign the fitness values to the chromosomes.

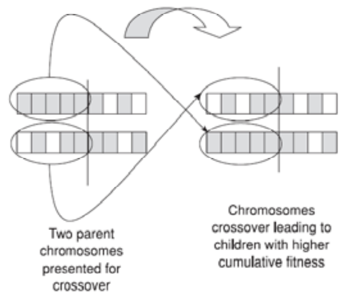
(4) *Evolutionary operators:* Crossover and Mutation.

(5) *Working parameters:* a set of values of the different parameters such as population size and chromosome length.

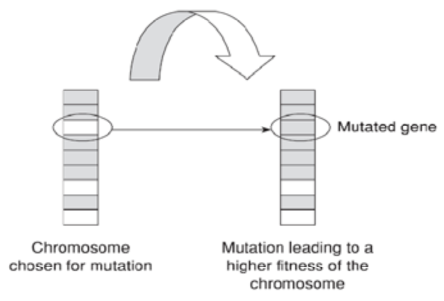
Genetic Algorithm (GA) is one of the important classes of EA. GA is a non-comprehensive search techniques based on natural selection, the process that derives biological evolution. GA is used to determine the global optima or the sub optima of a given function (or a process) that may or may not be subject to constraints. Unlike other search-based optimization procedures such as Hill Climbing or Random Search, GA has consistently achieved good performance in terms of balancing between the two conflicting objectives of any search procedure, which are the *exploitation* of the best solution and the *exploration* of the search space. GA has a number of other interesting features that differentiate them from other derivative based classical optimization techniques in two main ways:

- Classical Algorithm generates a single point at each iteration. The sequence of points approaches an optimal solution. Selects the next point in the sequence by a deterministic computation.
- GA generates a population of solutions at each iteration. The best point in the population approaches an optimal solution. Selects the next population by computation which uses random number generators.

The basic idea of GAs is to choose first a random population in the range of optimization, with a fixed size *n* (*n* usually depends on the search range, the accuracy required and the nature of the function itself). Using the so-called *binary encoding* procedure, each variable is represented as a string of *q* binary digits. This leads to a population of elements represented by a matrix of *n* rows and *q* columns. A set of "genetic" operators is then applied to this matrix to create a new population at which the function *f* attains increasingly larger values. The most common operators that have been used to achieve this task are: Selection, Crossover and Mutation.


Two parent chromosomes presented for crossover
Chromosomes crossover leading to children with higher cumulative fitness

(a): Crossover


Mutated gene
Chromosome chosen for mutation
Mutation leading to a higher fitness of the chromosome

(b): Mutation

Figure 22: Crossover & Mutation [59**]**

Figure 23: Schematic presentation of Genetic Algorithm

CHAPTER 4

HYBRID COMPUTATIONAL INTELLIGENCE MODELS

4.1. HYBRID OF MULTI-LAYER PERCEPTRON WITH GENETIC ALGORITHM

We have used Genetic Algorithm (GA) to find the optimum structure of the Multi-Layer Perceptron (MLP), the hidden neurons' transfer function and the type of the training algorithm that would fit that structure. Initially we run MLP without incorporating GA for each output components. As we have limited number of training samples, we decided to keep the network structure small and so used only one hidden layer. We normalized the data from -1 to 1 so in the output layer we have used tan-sigmoidal activation function in order that it can map the output as normalized format. As we predicted one output at a time, so we have used one node in the output layer. Figure 24 depicts the basic structure of ANN model used in this thesis. We have used are learning rate: 0.001, epochs: 300, error goal: 0.00001. The other parameter we decided to be optimized by GA. We tried to achieve the number of nodes in the hidden layer. We kept the range of hidden nodes between 1~63 to keep the network simpler. The second parameters that we achieved by GA is the hidden neurons' activation function. The possible options for activation functions are either tan-sigmoidal or log-sigmoidal. We have also found the training algorithm either LM (Levenberg-Marquardt) or Rprop (Resilient Backpropagation).

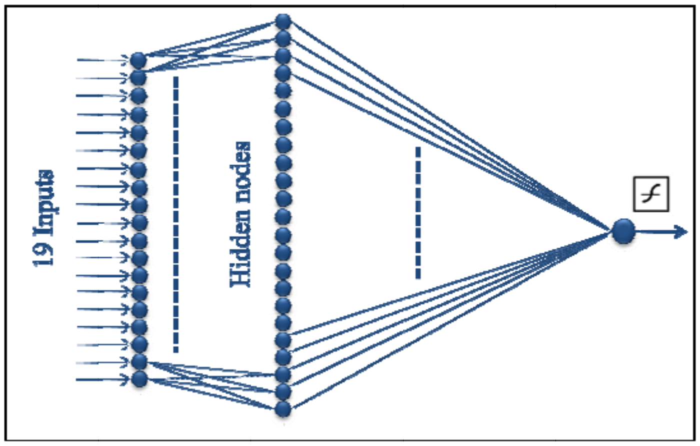


Figure 24: MLP structure with Fixed Parameters

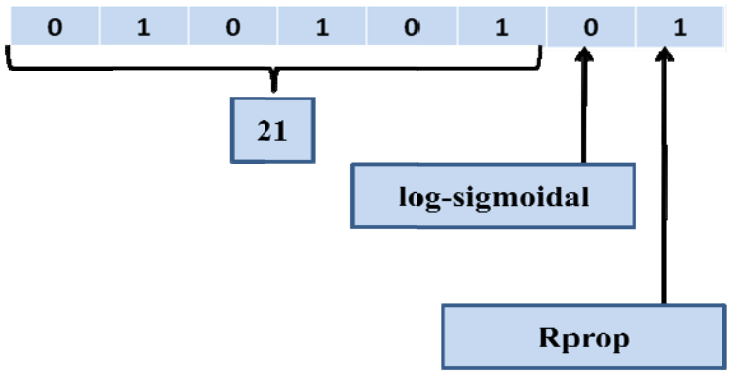


Figure 25: Binary Encoded Chromosome for MLP (8 bits)

We have used binary encoding to represent chromosome for MLP parameters optimization (Figure 25). We have used population type bit string. And the right most bit is to decide the training algorithm. The second bit is to choose activation function in the hidden layers. And the left most 6 bits is to choose the number of nodes in the hidden layer.

4.2. Hybrid of Support Vector Regression with Genetic Algorithm

The optimal parameter search on SVR plays a crucial role in building a prediction model with high prediction accuracy and stability. From the initial run of the problems we have decided to use "polynomial" type kernel. The degrees of polynomial we have used are 0.5, 2 and 3. For different member of ensemble we have used different degree of polynomial to make the model diverse. Genetic Algorithm can automatically optimize the SVR parameters and thus increase the predictive accuracy and capability of generalization [**7**,**25**,**60**]. The chromosome for SVR are encoded into Real Valued Encoding in the following ranges of C (0.0001~100), ε (0.0001~ 0.6), λ (0.000000001~ 0.001). An example of SVR chromosome representation is given in Figure 26. The ranges are decided from the initial prediction performance of each gas component.

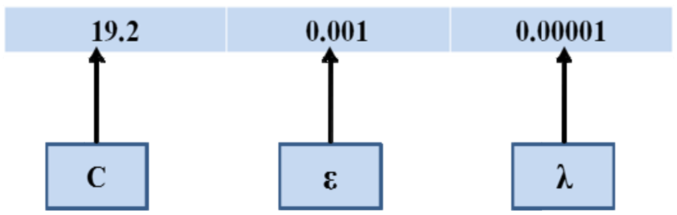


Figure 26: Chromosome for SVR

4.3. Hybrid of Adaptive Neuro-Fuzzy Inference System with Genetic Algorithm

The performance of ANFIS depends on the initial FIS. The more the initial FIS represented better, the better the performance of ANFIS would be. The FIS we have created by using *Subtractive Clustering (Subclust)* [**61**]. Subclust is one of the clustering algorithms based on a measure of the density of data points in the feature space. It generates the rules that approximate a function. The rule extraction method first uses SC to determine number of rules and input membership functions equation. Each fuzzy cluster is mapped into a generalized bell shaped (Figure 27) membership function which is defined as

$$bell\,(x; a, b, c) = \frac{1}{1 + \left|\frac{x-c}{a}\right|^{2b}}$$

where, c is centre of cluster, a is cluster radius, b is slope (a linear function)

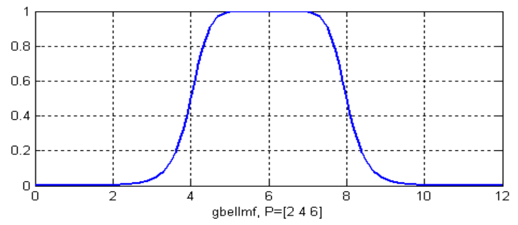


Figure 27: Generalized bell-shaped membership function

We decided to optimize the radius ‘a’ of Subclust by GA. The range of the radius we choose to be (0.2~0.9). Figure 28 shows a chromosome representation of ANFIS which is only one real encoded value. Figure 29 shows the steps of creating ANFIS from the initial FIS created using Subclust. Figure 30 shows the flow chart of radius optimization using GA.

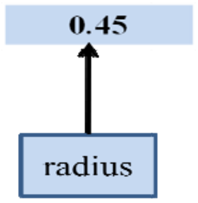


Figure 28: Chromosome of ANFIS

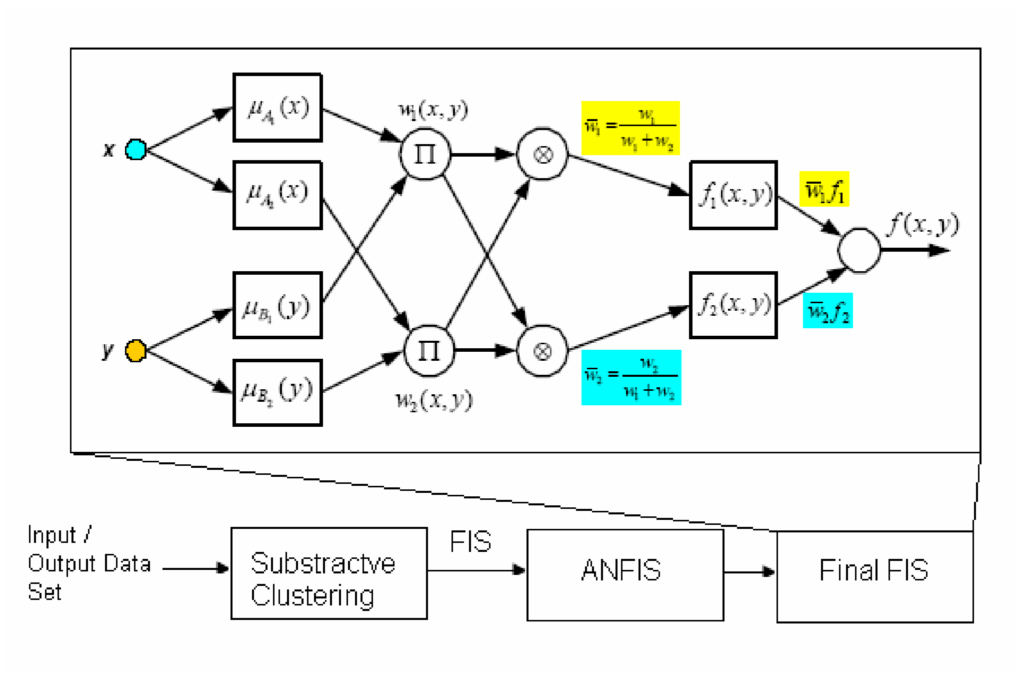


Figure 29: FIS Generation by ANFIS using Subtractive Clustering [58]

Figure 30: Radius (Subclust) Optimizing Steps using GA

4.4. Genetic Algorithm Parameters

The parameters of GA we have used for different gas components are given below:

- Population : 20~100
- Generation : 20~50
- Crossover fraction : 0.4~0.9
- Elite individual: 2
- Mutated Individual: (population – elite) – ((crossover fraction) x population)

4.5. Objective Function

The performance of the HCI model mostly depends on the design of the objective function. We have used RMSE or CC as a criterion of measuring fitness. Lower RMSE and higher CC values represents better models. So we have designed two objective functions where in one we setup the criterion is minimizing RMSE whereas in other the criterion is to maximize CC. some gas components perform well with minimizing RMSE objective while some other perform well with the objective maximizing CC.

CHAPTER 5

ENSEMBLE OF HYBRID COMPUTATION INTELLIGENCE MODEL

5.1. ENSEMBLE LEARNING

Ensemble Learning employs a committee of multiple learning machines and combines their outputs performing as a single decision maker. Figure 31 shows an ensemble of N number of CI models. The principle is that the combined decision of ensemble members should have better overall accuracy, on average, than any individual member. Numerous empirical and theoretical studies showed that ensemble accuracy significantly exceed the single model [**9**,**30**,**39**]. The underlying principle of ensemble learning is that every model has limitations and makes errors. Moreover, different learning algorithm suit with different problems. The goal of ensemble learning is to manage each learning algorithms' strengths and weaknesses automatically, leading to the best possible decision being taken overall.

Figure 31: Ensemble of Hybrid CI Model

5.1.1. Why Ensembles is Better?

To understand that why the generalization ability of an ensemble is usually much stronger than that of a single learner, Dietterich [**62**] gave three reasons by viewing the nature of machine learning as searching a hypothesis space for the most accurate hypothesis. The first reason is that, the training data might not provide sufficient information for choosing a single best learner. For example, there may be many learners perform equally well on the training data set. Thus, combining these learners may be a better choice. The second reason is that, the search processes of the learning algorithms might be imperfect.

For example, even if there exist a unique best hypothesis, it might be difficult to achieve since running the algorithms result in sub-optimal hypotheses. Thus, ensembles can compensate for such imperfect search processes. The third reason is that, the hypothesis space being searched might not contain the true target function, while ensembles can give some good approximation. For example, it is well-known that the classification boundaries of decision trees are linear segments parallel to coordinate axes. If the target classification boundary is a smooth diagonal line, using a single decision tree cannot lead to a good result yet a good approximation can be achieved by combining a set of decision trees. Note that those are intuitive instead of rigorous theoretical explanations.

Krough and Vedelsby [**63**] proved the formulation of ensemble error in case of regression using a linearly weighted ensemble. Let us assume the task is to learn a function *f(x)* and the training samples are drawn randomly from the distribution *p(x)*. Suppose the ensemble consist of N base learners, in our cases base learners are HCI models (Figure 31) and the output of the i_{th} HCI model is $O_i(x)$. The output of the ensemble is defined as

$O_{en}(x) = \sum_i W_i O_i(x)$ …………………………………………………….….. (i)

The diversity on input x of an individual HCI is defined as

$d_i = \big(O_i(x) - O_{en}(x)\big)^2$……………………………………………….………... (ii)

Then ensemble diversity on input x is

$d_{en} = \sum_i W_i d_i = \sum_i W_i \big(O_i(x) - O_{en}(x)\big)^2$………………………….…………..… (iii)

The quadratic errors of the i_{th} HCI model and of the ensemble are respectively as follows:

$e_i(x) = \left(f(x) - O_i(x)\right)^2$ …………………………………………………... (iv)

$e_{en}(x) = \left(f(x) - O_{en}(x)\right)^2$ ……………………………………………..…. (v)

From Eq. (3) and Eq. (5) we can write

$e_{en}(x) = \sum_i W_i e_i(x) - d_{en}(x)$ ……………………………………..……… (vi)

The average of over the input distribution p(x) can be written as follows:

Average error of individual model, $E_i(x) = \int dxp(x)\, e_i(x)$ …………………..….. (vii)

Average error of ensemble model, $E_{en}(x) = \int dx\, p(x)\, e_{en}(x)$ ……………..…. (viii)

Average diversity of individual model, $D_i(x) = \int dxp(x)\, d_i(x)$ ……………….. (ix)

From the Eq. (vii, viii, ix), the ensemble generalization error of Eq. (vi) can be formulated as

$E_{en} = \bar{E} - \bar{D}$ ……………………………………………………………… (x)

Where

$\bar{E} = \sum_i W_i E_i(x)$ is the weighted average of the generalization errors of the individual HCI model and

$\bar{D} = \sum_i W_i D_i(x)$ is the weighted average of the diversity among these HCI models which is a non-negative value.

Eq. (x) show that an ideal ensemble consists of highly correct HCI models that disagree as much as possible and the generalization error of the ensemble is always smaller than the average of the individual errors, that is $E_{en} < \bar{E}$.

5.1.2. Diversity & Accuracy

Theoretically the ensemble error can be described as two distinct components: first of all, the accuracies of the individual models and secondly, a term for their interactions, i.e. their diversity. In a regression problem squared is commonly used for error measure. Using a linear combiner the accuracy-diversity breakdown for regression problem is called the "Ambiguity Decomposition" by Krogh and Vedelsby [**63**]. They showed that the squared error of the linearly combined ensemble, *f(x)*, can be broken into a sum of two components:

$$\boldsymbol{(f(x) - d)^2 = \frac{1}{T}\sum_{t=1}^{T}(f_t(x) - d)^2 - \frac{1}{T}\sum_{t=1}^{T}\left(f_t(x) - f(x)\right)^2} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\text{(a)}$$

The first term on the right hand side is the average squared error of the individual models, while the second term computes the variation between the predictions. This second term is called "ambiguity" which is always positive. This assures that, for any arbitrary data point, the ensemble squared error is always less than or equal to the average of the individual squared errors.

The optimal "diversity" can be thinking of as a credit assignment problem. If a committee of doctors as a whole concluded an erroneous diagnosis of a disease, how much of this error should be attributed to each doctor? In particular, how much of the committee decision is due to the accuracies of the individual doctor and how much is due to their interactions when they were combined. The intuition here can also be understood by a fairground game example explained by Brown [**64**]. Let us imagine groups of five players, playing “guess the weight of the cake": if a player's guess is close enough to the true weight, that group will win the cake. The fairground manager states that each player can only submit one guess. The dilemma seems to be in whose guess the group should submit. However, the Ambiguity decomposition in Eq. (a) shows that taking the average of their guesses will always on average be closer than choosing a player at random and submitting their guess.

Note that this is qualified with “on average" it may well be that one of the predictions will in fact be closer than the average prediction, but there is no way of identifying which predictor to choose, other than random. It can be seen that greater diversity in the predictions (i.e. a larger ambiguity term) will result in a larger gain over the average individual performance. However it is also clear that there is a trade off to be had: too much diversity would cause the average error to be extremely large.

In summary, the definition of diversity depends on the problem. In a regression problem, the optimal diversity is the trade-off between the bias, variance and covariance components of the squared error. In a classification problem, with a linear combiner, there exists partial theory to relate the classifier correlations to the ensemble error rate. In a classification problem with a voting combiner, there is no single theoretical framework or definition of diversity.

5.1.3. Bias & Variance

The bias component tells us how accurate the model is, on average across different possible training sets. The variance component tells us how sensitive the learning algorithm is to small changes in the training set. Let us assume we have target variable Y, vector of inputs X, Prediction model $f(x)$. Therefore,

$Y = f(X) + \varepsilon,$ Where ε~N (0, 1), $E(\varepsilon) = 0, Var(\varepsilon) = \sigma_\varepsilon^2$

Then for an input point $X = x$ (uniform random variable in [0,1]), the following figures in Figure 32 depicts some scenario of over-fitting and under-fitting by the effect of bias and variances.

(a): y=h(x)+ε where ε~N(0,1)	(b): Low variance, high bias method (under-fitting)
(c): Low bias, high variance method (over-fitting)	(d): No noise doesn't imply no variance (but less variance)

Figure 32: Effect of Bias and Variances

5.1.4. Bias –Varinace Relation with Model's Complexity

Figure 33: Bias and Variance of a Model [65]

As complexity of the model is increased, bias decreases (a better fit to data) and variance increases (fit varies more with data) (Figure 33). Uaually, the bias is a decreasing function of the complexity, while variance is an increasing function of the complexity (Figure 34).

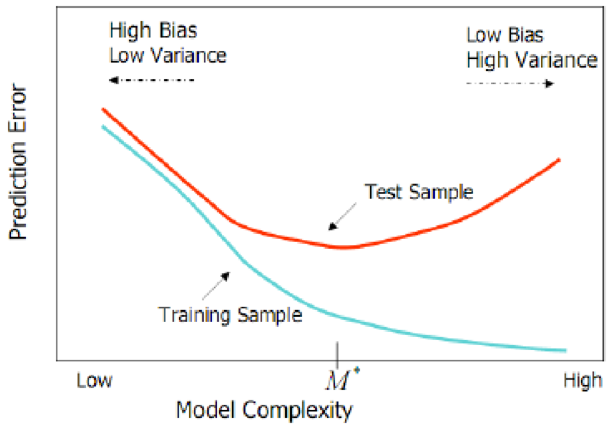


Figure 34: The Variation of Bias & Variance Model Complexity

Generalization performance of a learning method measure of prediction capability on independent test data thus guides model selection. Training data usually affect monotonically increasing performance with model complexity. Training error is computed by average loss over training samples. Increasing the model complexity would cause in decreasing training error consistently and drops to zero with high enough complexity i.e. over fitting situation would occur (Figure 35).

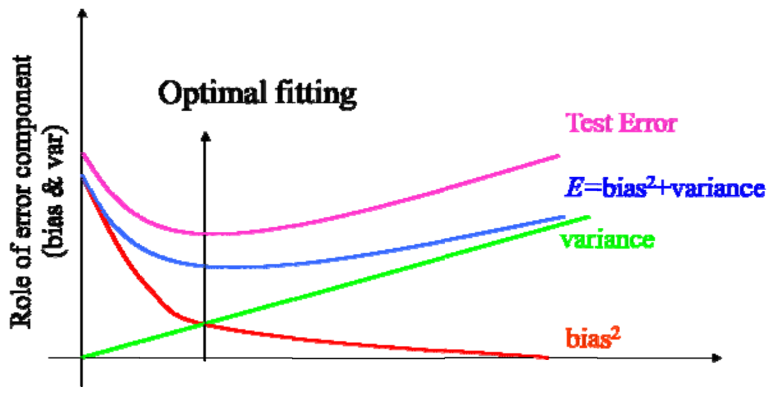


Figure 35: Minimum Test Error Occur at Minimum of "E"

5.1.4.1. <u>Some Examples of Bias-Variance Influences with CI Models' Complexity</u>

Example 1 – Bias-Variance with SVR's degree of polynomial

- Low degree polynomial has high bias (fits poorly) but has low variance with different data sets
- High degree polynomial has low bias (fits well) but has high variance with different data sets

Example 2 – Error, bias and variance w.r.t the number of neuron in the hidden layer of MLP (Figure 36).

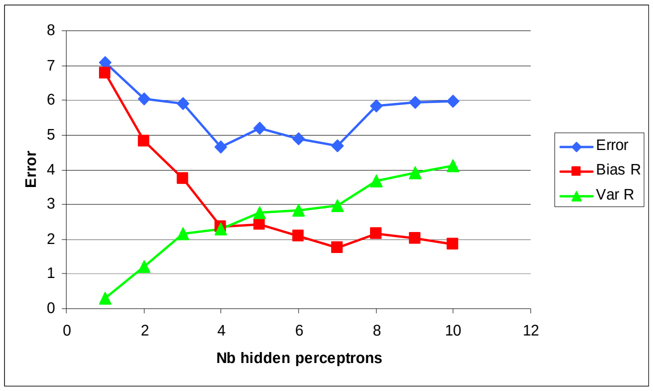


Figure 36: Bias-Variance Effect with Increasing Hidden Nodes

Example 3 **–** At fixed model complexity, bias remains constant and variance decreases with the learning sample size (Figure 37).

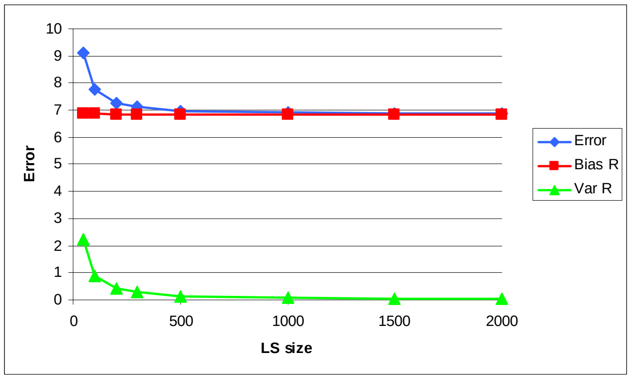


Figure 37: Variance Affect with Training Sample Size and Fixed Complexity

Example 4 **–** When the complexity of the model is dependent on the learning sample size, both bias and variance decrease with the learning sample size (Figure 38).

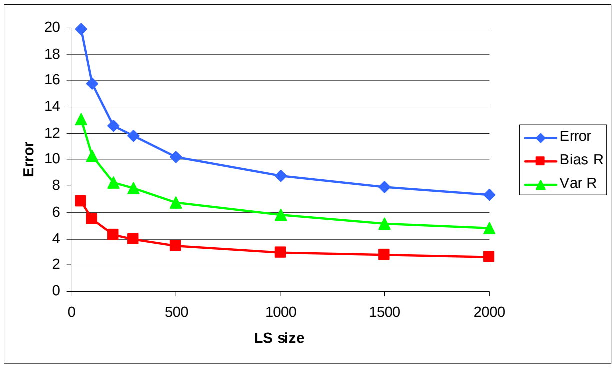


Figure 38: Bias-Variance Affect with Training Sample Size and Complexity

5.1.5. Bias-Variance Decomposition

The bias-variance decomposition is a useful theoretical tool to understand the performance characteristics of a learning algorithm. Brown [**64**] explained the bias-variance analysis by dartboard example quoting from Moore and McCabe [**66**] (Figure 39).

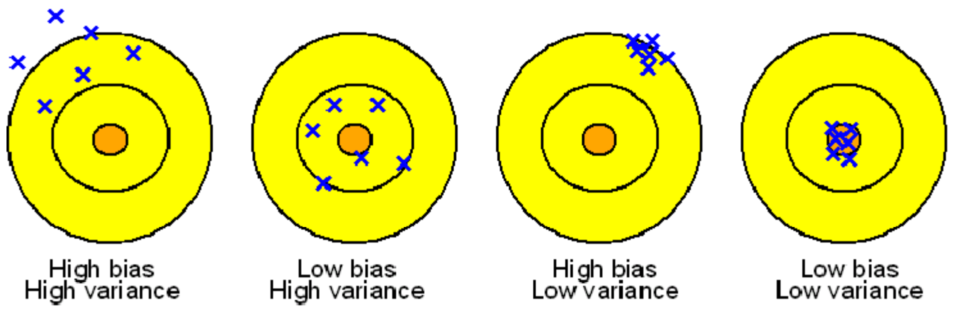


Figure 39: Effect of Bias and Variance – Dartboard Analogy [64]

Each dart is thrown after training the "dart-throwing" model in a slightly different manner. If the darts vary wildly, the learner is high variance. If they are far from the bull's eye, the learner is high bias. The efforts to reduce variance often cause increases in bias, and vice-versa. A large bias and low variance is an indicator that a learning algorithm is prone to over-fitting the model.

The ideal is clearly to have both low bias and low variance; however this is often difficult, giving an alternative terminology as the bias-variance 'dilemma'. The idea of a trade-off between diversity-accuracy is suggested by Geman et al. [**67**] as "Bias-Variance decomposition". In fact, there is a deep connection between these results. Mathematically, this can be quantified as a decomposition of the mean squared error function. For a testing example f(x) with target d, the decomposition is:

$$\boldsymbol{\varepsilon_D}\{(\boldsymbol{f}(\boldsymbol{x})-\boldsymbol{d})^{\mathbf{2}}\} = (\boldsymbol{\varepsilon_D}\{\boldsymbol{f}(\boldsymbol{x})-\boldsymbol{d}\})^{\mathbf{2}} + \boldsymbol{\varepsilon_D}\{(\mathrm{f}(\mathrm{x}) - \varepsilon_{\mathrm{D}}\{\mathrm{f}(\mathrm{x})\})^2\} \text{……………………..(b)}$$

5.1.6. Bias-Variance-Covariance Decomposition

The Bias-Variance-Covariance decomposition is a theoretical result underlying Ensemble Learning algorithms. It is an extension of the Bias-Variance decomposition, for linear combinations of models. Taking the expected value of Eq (a) above over all possible training sets gives us the ensemble analogy to the bias-variance decomposition described by Ueda and Nakano [**30**], called the "Bias-Variance-Covariance decomposition". This shows that the expected squared error of an ensemble f(x) from a target d is:

$$\boldsymbol{\varepsilon_D}\left\{\left(\bar{\boldsymbol{f}}(\boldsymbol{x})-\boldsymbol{d}\right)^{\mathbf{2}}\right\} = \overline{\boldsymbol{bias^2}} + \frac{\mathbf{1}}{\boldsymbol{T}}\overline{\boldsymbol{var}} + \left(\mathbf{1}-\frac{\mathbf{1}}{\boldsymbol{T}}\right)\overline{\boldsymbol{covar}} \qquad \text{…………………(c)}$$

The expectation is with respect to all possible training datasets D. While the bias and variance terms are constrained to be positive, the covariance between models can become negative. Therefore, the definition of diversity comes forward as an extra degree of freedom in the bias-variance dilemma. This extra degree of freedom allows an ensemble to approximate functions that are difficult to find with a single model [**9**]. Shortly the error is composed of the average bias of the models, plus a term involving their average variance, and a final term involving their average pair wise co-variance. This shows that while a single model has a two-way bias-variance trade off, an ensemble is controlled by a three-way trade off. This ensemble trade off is often referred to as the accuracy-diversity dilemma for an ensemble.

5.1.7. Bias-Variance-Covariance Decomposition: An Illustration

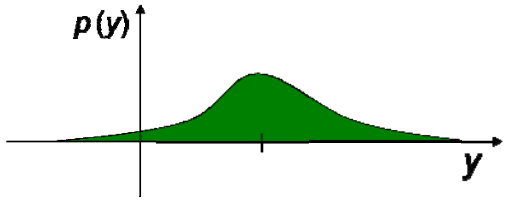


Figure 40: Target Variable *y*, Need to Find *p(y)*

In case of prediction we want find estimation let say $\hat{y}$ such that the expectation $\boldsymbol{E_y\{(y-\hat{y})^2\}}$ over the whole population is minimized. The estimation that minimizes the error can be computed by taking

$$\frac{\partial E_y}{\partial \hat{y}}\{(y-\hat{y})^2\}=0$$

$E_y\{-2.(y-\hat{y})\}=0,$

$E_y\{y\}-E_y\{\hat{y}\}=0,$

$\hat{y}=E_y\{y\}$

So, the estimation which minimizes the error is $E_y\{y\}$. In AL, it is called the Bayes model. But in reality, wet cannot compute the exact value of $E_y\{y\}$ because it involves the whole population which is not faceable to compute.

As $p(y)$ is unknown, find an estimation y from a sample of individuals, $LS = \{y_1, y_2, \dots , y_N\}$, drawn from the whole population. As LS are randomly drawn, the prediction $\hat{y}$ will also be a random variable.

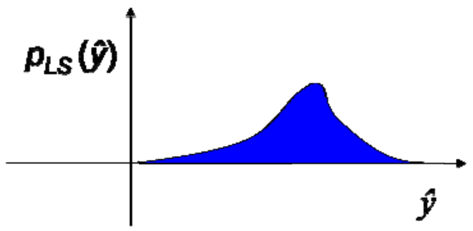


Figure 41: Estimation of ŷ Required

A good learning algorithm should not be good only on one learning sample but in average over all learning samples (of size N), we want to minimize:

$$E = E_{LS}\left\{E_y\{(y-\hat{y})^2\}\right\}$$

Let us analyze this error in more details –

$$E_{LS}\left\{E_y\{(y-\hat{y})^2\}\right\}$$

$$= E_{LS}\left\{E_y\left\{\left(y - E_y\{y\} + E_y\{y\} - \hat{y}\right)^2\right\}\right\}$$

$$= E_{LS}\left\{E_y\left\{\left(y - E_y\{y\}\right)^2\right\}\right\} + E_{LS}\left\{E_y\left\{\left(E_y\{y\} - \hat{y}\right)^2\right\}\right\} + E_{LS}\left\{E_y\{2\left(y - E_y\{y\}\right)\left(E_y\{y\} - \hat{y}\right)\}\right\}$$

$$= E_y\left\{\left(y - E_y\{y\}\right)^2\right\} + E_{LS}\left\{\left(E_y\{y\} - \hat{y}\right)^2\right\} + E_{LS}\{2\left(E_y\{y\} - E_y\{y\}\right)\left(E_y\{y\} - \hat{y}\right)\}$$

$$= E_y\left\{\left(y - E_y\{y\}\right)^2\right\} + E_{LS}\left\{\left(E_y\{y\} - \hat{y}\right)^2\right\}$$

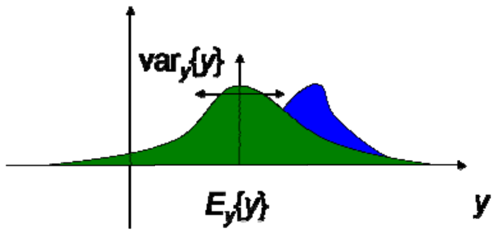


Figure 42: Variance of y

$$E = E_y\left\{\left(y - E_y\{y\}\right)^2\right\} + E_{LS}\left\{\left(E_y\{y\} - \hat{y}\right)^2\right\}$$

The first term is called residual error i.e. minimum attainable error. It is basically $\text{var}_y\{y\}$.

$$E_{LS}\left\{\left(E_y\{y\} - \hat{y}\right)^2\right\}$$

$$= E_{LS}\left\{\left(E_y\{y\} - E_{LS}\{\hat{y}\} + E_{LS}\{\hat{y}\} - \hat{y}\right)^2\right\}$$

$$= E_{LS}\left\{\left(E_y\{y\} - E_{LS}\{\hat{y}\}\right)^2\right\} + E_{LS}\{(E_{LS}\{\hat{y}\} - \hat{y})^2\} + E_{LS}\{2\left(E_y - E_{LS}\{\hat{y}\}\right)(E_{LS}\{\hat{y}\} - \hat{y})\}$$

$$= \left(E_y\{y\} - E_{LS}\{\hat{y}\}\right)^2 + E_{LS}\{(\hat{y} - E_{LS}\{\hat{y}\})^2\} + 2\left(E_y - E_{LS}\{\hat{y}\}\right)E_{LS}\{(E_{LS}\{\hat{y}\} - \hat{y})\}$$

$$= \left(E_y\{y\} - E_{LS}\{\hat{y}\}\right)^2 + E_{LS}\{(\hat{y} - E_{LS}\{\hat{y}\})^2\}$$

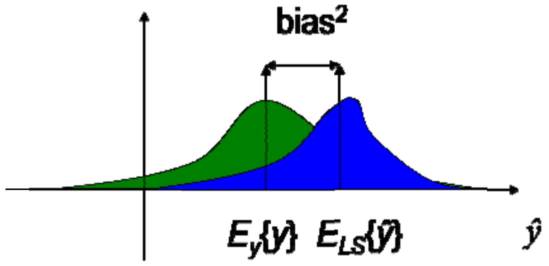


Figure 43: Bias between ŷ and y

$E = \text{var}_y\{y\} + \boldsymbol{(E_y\{y\}\text{-}E_{LS}\{\hat{y}\})^2} + \ldots$

Where, $\boldsymbol{E_{LS}\{\hat{y}\}}$ = average model (over all LS)

bias^2 = error between the Bayes and the average model

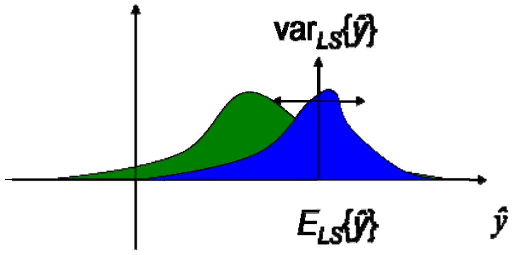


Figure 44: Variance of ŷ

$E = \text{var}_y\{y\} + \text{bias}^2 + \boldsymbol{E_{LS}\{(\hat{y}\text{-}E_{LS}\{\hat{y}\})^2\}}$

$\text{var}_{LS}\{\hat{y}\}$ = estimation variance = consequence of over-fitting

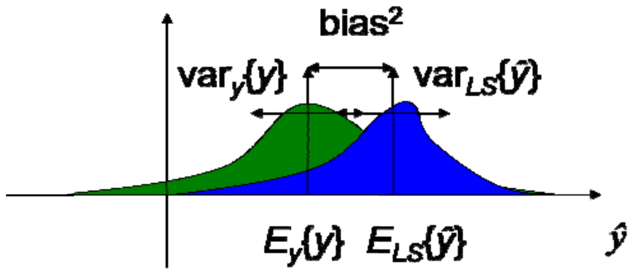


Figure 45: Variances & Bias between ŷ and y

$E = \mathrm{var}_y\{y\} + \mathrm{bias}^2 + \mathrm{var}_{LS}\{\hat{y}\}$

Now consider an input point $\underline{x} \subseteq X$. In regression problem we want to find a function ŷ(x) of several inputs. Using unit-square loss and regression fit the error is -

$$MSE(\underline{x}) = E_{\underline{x},y}[(y - \hat{y}(\underline{x}))^2]$$

$$= \sigma_\varepsilon{}^2 + [E\hat{y}(\underline{x}) - y]^2 + E[\hat{y}(\underline{x}) - E\hat{y}(\underline{x})]^2$$

$$= \sigma_\varepsilon{}^2 + Bias[\hat{y}(\underline{x})]^2 + Var[\hat{y}(\underline{x})]$$

Over all the learning sets:

$$E = E_{LS}\left\{E_{\underline{x},y}\left\{\left(y - \hat{y}(\underline{x})\right)^2\right\}\right\}$$

$$= E_{\underline{x}}\left\{E_{LS}\left\{E_{y|\underline{x}}\left\{\left(y - \hat{y}(\underline{x})\right)^2\right\}\right\}\right\}$$

$$= E_{\underline{x}}\left\{var_{y|\underline{x}}\{y\}\right\} + E_{\underline{x}}\left\{bias^2(\underline{x})\right\} + E_{\underline{x}}\left\{var_{LS}\{\hat{y}(\underline{x})\}\right\}$$

Therefore, $E_{LS}\{E_{y|\underline{x}}\{(y-\hat{y}(\underline{x}))^2\}\}$ = Noise($\underline{x}$) + Bias^2($\underline{x}$) + Variance($\underline{x}$)

- Noise(x) = $E_{y|\underline{x}}\{(y-h_B(\underline{x}))^2\}$: Noise quantifies how much y varies from $h_B(\underline{x}) = E_{y|\underline{x}}\{y\}$, the Bayes model. This is also called Irreducible Error: Variance of the target around the true mean

- Bias^2(x) = $(h_B(\underline{x})-E_{LS}\{\hat{y}(\underline{x})\})^2$: Bias measures the error between the Bayes model and the average model i.e. Amount by which average estimate differs from the true mean

- Variance(x) = $E_{LS}\{(\hat{y}(\underline{x})-E_{LS}\{\hat{y}(\underline{x}))^2\}$: Variance quantify how much $\hat{y}(\underline{x})$ varies from one learning sample to another i.e. Expected deviation of f^ around its mean

5.1.8. Challenges of Ensemble

An ensemble is a very successful technique where the outputs of a set of separately trained base learners are combined to form one unified prediction. First it can improve the generalization performance of a classification system greatly. Second, it can be viewed as an effective approach for CI as a result of its variety of potential applications and validity. Third, local minima are available for ensembles. Individuals in ensemble are expected to different local minima of error surface, increasing the diversity of ensemble. Although ensembles have been used widely, the key problem for researchers is how to effectively design the individual works that are not only highly correct, but also different as much as possible.

5.2. Ensemble Learning Algorithms

An ensemble basically consists of a set of models and a method to combine them. If we had a committee of people taking decisions, it is self-evident that we would not want them all to make the same bad judgments at the same time. With a committee of learning models, the same intuition applies: we will have no gain from combining a set of identical models. We wish the models to exhibit a certain element of "diversity" in their group behavior, though still retaining good performance individually.

It is known that Bagging can significantly reduce the variance, and therefore it is better to be applied to learners suffered from large variance, e.g., unstable learners such as decision trees or neural networks. Boosting can significantly reduce the bias in addition to reducing the variance, and therefore, on weak learners such as decision stumps, Boosting is usually more effective.

5.2.1. Bagging

Bagging is an Ensemble Learning technique. Breiman [**68**] developed the *bagging* ensemble based algorithm in which different training data subsets are randomly selected with replacement from the entire training data to train different individual models and combined by a uniform average or vote. Each member of the ensemble is constructed from a different training dataset. Each dataset is a bootstrap sample from the original. The name "Bagging" comes from "Bootstrap AGGregatING". Since a bootstrap samples N items uniformly at random with replacement, the probability of any individual data item not being selected is $p = (1 - \frac{1}{N})^N$. Therefore with large N, a single bootstrap is expected to contain approximately 63.2% of the original set, while 36.8% of the originals are not selected. Figure 46 shows flowchart of Bagging algorithm.

Figure 46: Flow Chart of Bagging

Bagging works best with unstable learners which differing generalization patterns with small changes to the training data. These are also known as high variance models, examples of which are Decision Trees and Neural Networks. Bagging therefore tends not to work well with very simple models. In effect, Bagging samples randomly from the space of possible models to make up the ensemble with very simple models the sampling produces almost identical (low diversity) predictions. Despite its apparent capability for variance reduction, situations have been demonstrated where Bagging can converge without affecting variance [**9**].

Figure 47: Idea of Bagging Ensemble

The idea behind Bagging is shown in Figure 47 where the average model $E_{LS}\{\hat{y}(\underline{x})\}$ has the same bias as the original method but zero variance. Usually, bagging reduces very much the variance without increasing too much the bias. It decreases the variance (because of averaging) but (slightly) increases the bias (because of the perturbation).

5.2.2. Boosting

Boosting is a family of ensemble learning methods which is a generally an effective method for improving the accuracy of any given learning algorithm. In the algorithm the successive networks are trained with a training set selected at random from the original training set, but the probability of selecting a pattern changes depending on the correct classification of the pattern and on the performance of the last trained network. Figure 48 shows the flowchart of Boosting algorithm that linearly combines the output of each CI models.

Figure 48: Flow Chart of Boosting

The Boosting framework is an answer to a question posed on whether two complexity classes of learning problems are equivalent: strongly learnable and weakly learnable. The Boosting framework is a proof by construction that the answer is positive, they are equivalent. The framework allows a “weak" model, only slightly better than random guessing, to be boosted into an arbitrarily accurate strong model. The motivation of boosting is to combine the output of many “weak” models to produce a powerful ensemble of models. Schapire and Freund [**11**,**69**] proved the equivalence between the strong learning model and weak learning model, and gave a boosting approach that convert the weak learning model into strong learning model directly. In Boosting ensemble, the distribution of a particular training set in the series is over-represented by the patterns that the earlier classifiers in the series recognize incorrectly. The individual classifiers are trained hierarchically to learn harder and harder parts if a classification problem. A weak model has a high bias (strictly, in classification, a model slightly better than random guessing). The main difference of Boosting with previous ensemble methods is building the models sequentially on modified versions of the data. The Predictions of the models are combined through a weighted sum or majority voting. Adaboost is the most well known and successful of the Boosting family, though there exist many variants specialized for particular tasks, such as cost-sensitive and noise-tolerant versions [**64**]. It was demonstrated by Dietterich [**62**] that when the number of outliers is very large, the emphasis placed on the hard samples can become detrimental to the performance of the AdaBoost. Friedman [**70**] put forward a variant of AdaBoost, called "Gentle AdaBoost" that puts less emphasis on outliers.

5.2.3. Adaboost

Adaboost is the most well known of the Boosting family of algorithms [**11**]. The algorithm trains models sequentially, with a new model trained at each round. At the end of each round, misclassified examples are identified and have their emphasis increased in a new training set which is then fed back into the start of the next round, and a new model is trained. The idea is that subsequent models should be able to compensate for errors made by earlier models. Some similarities with Bagging are evident; a key difference is that at each round n, Bagging has a uniform distribution D_n, while Adaboost adapts a non-uniform distribution. The ensemble is constructed by iteratively adding models. Each time a model is learnt, it is checked to ensure it has at least $\epsilon_n < 0.5$, that is, it has performance better than random guessing on the data it was supplied with. If it does not, either an alternative model is constructed, or the loop is terminated. After each round, the distribution D_n is updated to emphasize incorrectly classified examples. Mease and Wyner [**71**] presented a discussion of several questions on why and how Adaboost succeeds. The conclusion is, while no single theory can fully explain Boosting, each provides a different part of the still unfolding story.

5.3. Ensemble of Hybrid Computational Intelligence Models Building Approach

Typically, an ensemble is constructed in two steps. First, a number of base learners are produced, which can be generated in a *parallel* style (Bagging) or in a *sequential* style (Boosting) where the generation of a base learner has influence on the generation of subsequent learners. Then, the base learners are combined to use, where among the most popular combination schemes are *majority voting* for classification and *weighted averaging* for regression.

In this thesis we resolved a regression problem of gas compositions prediction in multistage separator using Ensemble of Hybrid CI (EHCI) models. We developed three homogeneous and one heterogeneous EHCI models using parallel scheme. Homogeneous models consist of same types of CI models as base learners and heterogeneous model consist of different types of CI models as base learners. We used the most popular CI models MLP, SVR and ANFIS as base learners of ensemble models which are successfully used in many problems of petroleum industry [**20**,**21**,**22**,**23**,**24**,**25**,**24**,**26**,**27**]. We combined GA with each base learner to have hybrid models. GA optimizes the most crucial parameters of each CI model which are mainly responsible for accuracy. To compare the performance of the EHCI models, results from CPCP is used as a benchmark. The EHCI models are found having improved generalization ability comparing CPCP and single HCI models.

Many approaches for designing individuals in ensemble have been developed in the literature. Here we focused to emphasize the accuracy in each CI model and at the same time tried to enforce diversity among the CI models in a number of ways. The EHCI model is constructed by two steps, one is designing the ensemble members and the other is combining their predictions.

5.3.1. Designing the Ensemble Members

Combining the output of several classifiers is useful only if they disagree on some inputs. Theoretical and empirical work showed that an effective ensemble should consist of a set of networks that are not only highly correct, but ones that make their errors on different parts of the input space as well [**10**,**63**]. Generally, the approaches for employing diversity while designing the networks can be conducted into three groups [**39**] as follows:

5.3.1.1. Difference in Ensemble Members' Structures

Diverse individuals can be obtained by adopting different model structure. In case of Neural Network different types of models can be obtained by having different network types, number of neuron in hidden layer, learning algorithm and initial state in weight space. For SVR it can be different kernel function and kernel parameters as well as different C, ε and λ values. On the other hand for ANFIS it would be the methodology of creating initial FIS, the different types of ANFIS structure, etc.

5.3.1.2. Difference in Training Set

Diversity can be supported by training the EHCI members on different training datasets which can be achieved by bagging, boosting or cross validation [**11**,**63**,**68**]. Both the first one and the second one generate a group of networks which are error uncorrelated directly. Partridge [**72**] experimentally compared the capabilities of the method above and concluded that varying the net type and the training data are the two best ways for creating ensembles of networks making different errors.

5.3.1.3. Difference in Training Inputs

Different input parameters can be given to different base learners thus having a diverse knowledge overall the problem domain. In this case different base learners are expert in different portion of the solution space and improve the generalization ability of the combined model.

5.3.1.4. <u>Selecting Uncorrelated Ensemble Members</u>

Another popular way to have diversity is to generate a large number of initial networks from which several uncorrelated networks are selected as a member of the ensemble. Opitz and Shavlik [**73**] proposed an approach based on generic algorithm, searching for highly diverse set of accurate trained networks. Lazarevic and Obradoric [**74**] proposed a pruning algorithm to eliminate redundant classifier. Zhou et al. [**46**] described a selective constructing approach for ensemble; clustering-based selective neural network ensemble.

5.3.2. Enforcing Diversity in EHCI Models

We have emphasized on the first two ways to enforce diversity in our EHCI models. Basically we have followed three ways to enforce diversity.

5.3.2.1. <u>Heterogeneous Ensemble</u>

Heterogeneous Ensemble consists of members having multiple type base learning algorithms. In this case ensemble members can be different by the structure. We developed one heterogeneous ensembles model having GA optimized CI models of type MLP, SVR and ANFIS. At first we provided the input in MLP. We selected the badly predicted training data by MLP and provide it to train the SVR and later on the badly predicted training data by SVR is provided to ANFIS for training. In this way the model would become diverse by having training datasets and one HCI model handled those cases which cannot be handled by the other HCI model.

5.3.2.2. <u>Homogeneous Ensemble</u>

Homogeneous Ensemble consists of members having a single type base learning algorithm. In this case ensemble members can be different by the structure. We developed three homogeneous ensemble models and each has three HCI models of same type. As we have used three types of HCI models, we come up with three heterogeneous EHCI models.

5.3.2.3. <u>Selecting Training Set for Each EHCI Member</u>

We have selected different sizes of training set for different types of output. At first we divided the whole datasets into training and testing. Around 80% of the whole datasets is used for training and the 20% of the relevant datasets were used for testing. The training size for the ensemble members varies from 60% - 80% of the whole training set. We took the idea of sampling from the concept of boosting, especially in the case of classification. After the first run of the algorithm, in each of the following run we have selected the same amount of training data as selected in the first run which are badly predicted by the CI model of the current run.

5.3.3. Combing the Outputs of EHCI Members

To combine the outputs of the ensemble we have used linear and non-linear approaches. When the ensemble is used in classifying, voting is usually been used for combining outputs and when the ensemble is used in regression, simple average and weighted average are always been used [**39**]. Opitz and Shavlik [**73**] has pointed out that simple averaging outperforms since optimizing the combining weights can easily lead to the problem of over-fitting. Perrone and Copper [**75**] considers weighted average has a better performance as each network can avoided over-fitting by using a cross-validatory stopping rule. Sollich and Krogh [**76**] found that in large ensembles, one should use the simple averaging. In this way, the globally optimal generalization error on the basis of all the available data can be reached by optimizing the training set sizes of the individual member. For ensembles of more realistic size, optimizing the ensemble weights can still yield substantially better generalization performance than an optimally chosen single network trained on all data with the same amount of training noise.

The outputs of the EHCI members goes as an input into CI models and these models are trained after the training phase completion of the members of EHCI. We found that in some cases non-linear combiner performed well while in some cases the linear combiner performed better results. Among the linear approaches we have used simple average and weighted average methods to combine the outputs of EHCI members. We have also used many non-linear approaches to combine the outputs [**41**]. We have chosen CI models such as ANN, SVR, FIS created with Fuzzy C-means Clustering (FCM) and Subtractive Clustering (Subclust) as a combiner. For ANN combiner we have used MLP with one neuron in the hidden layer with logsigmoidal activation function. In the output layer we have used tansigmoidal activation function and we have used Rprop training algorithm. For SVR we have used "gaussian" type kernel with γ value 5. The other parameters e.g. C

= 0.5, lambda = 1e-7 and epsilon = 0.0001. For creating FIS, we have used FCM with 6 clusters and radius of 0.3 for Subclust.

5.4. Ensemble of Hybrid Computation Intelligence Models Development Steps

The Ensemble of Hybrid Computational Intelligence (EHCI) models building steps are stated as follows -

1. Determine the CI models' parameters to be optimized by observing models' accuracy and complexity.
2. Develop an Ensemble Model -
    a. Randomly choose X% of the training datasets.
    b. Optimize CI model using GA
    c. Training CI model on this X% datasets;
    d. Predict 100% training datasets;
    e. Choose the X% of badly predicted training datasets.
    f. Perform the steps b, c and d for N times on the data availed by step e. (N = number of ensemble members).

At first, we divide the datasets randomly into training and testing set. We have used 80% of the datasets for training and 20% for testing. To make homogeneous EHCI model same kind of CI model with different fixed parameters is chosen in step b of each run. Performing optimization by GA with different fixed parameters results into a completely different architecture of the CI model in each run. Consequently, though the homogeneous EHCI models have similar type of CI models, their architecture is completely different. Furthermore, these HCI members of the EHCI models are trained by different portion of the training datasets and thus EHCI models are enforced to be diverse enough in order to substantiate better generalization. On the other hand to make a heterogeneous EHCI model a different CI model must be chosen at step b in each run. The algorithm can be continued to N runs so as to have an ensemble of N members. The training and testing phase in EHCI model building steps are described below.

5.4.1. EHCI Model Development Steps: Training Phase with Linear Combiner

Figure 49: EHCI Model Building Steps - Training Phase with Linear Combiner

To train the EHCI models we have selected X% of the training data randomly with replacement to perform training of the base CI model in the first run. The percentages of training size are varying from 60%-90% for different gas components in order to achieve better performance. In the training phase (Figure 49: (a)), at first the parameters of the CI models were optimized using GA and then the training is performed. In the subsequent steps we predicted the whole training set as a part of “local testing” and chose X% of the badly predicted data from the whole training set to perform training in the next run. To perform combining of the EHCI members output linearly we have used simple average and weighted average method (Figure 49: (b)). To assign weight of the members of ECHI model we predicted the whole training data to measure each member’s performance in terms of RMSE. The formula for weighted average method is

$$\frac{\sum_{i=1}^{n} w_i x_i}{\sum_{i=1}^{n} w_i}$$

Where the weight calculation formula is

$$W_i = \frac{\sum_{i=1}^{n} RMSE_i - RMSE_i}{(n-1) \times \sum_{i=1}^{n} RMSE_i}$$

5.4.2. EHCI Model Development Steps: Training Phase with Non-Linear Combiner

The training phase is similar as stated above. In order to combine with non-linear approach we predicted the whole training data. The predicted output of the EHCI members are used as input and the actual outputs are used as output of the non-linear models. We have used NN, SVR, FIS-Subclust and FIS-FCM non-linear models as combiner (Figure 50: (b)). We have trained these non-linear models and used them for prediction in the testing phase.

Figure 50: EHCI Model Building Steps - Training Phase with Non-Linear Combiner

5.4.3. EHCI Testing Phase

The test data is predicted using the EHIC members and then combined the outputs by linear and non-linear models (Figure 51).

Figure 51: EHCI Testing Phase

5.5. Development Tools

The experiments are conducted by High Performance Computing (HPC) of ITC at KFUPM. Some experiments are also accomplished in high speed Intel Xeon quad core 2.8GHz platform. The EHCI models are implemented using MATLAB codes and

MATLAB toolboxes of Neural Network, ANFIS, GA, etc. The SVM-KM package is used for SVR. The CPCP dongle is used to produce result of the test data to use it as a benchmark.

CHAPTER 6

EXPERIMENTAL RESULTS & DISCUSSION

6.1. PERFORMANCE EVALUATION

In this thesis commonly used techniques for measuring regression problem will be applied to evaluate the performance of the results. They are explained as follows:

6.1.1. Correlation Coefficient

The Correlation Coefficient (CC) measures the statistical correlation between the predicted and actual values. CC shows how good the prediction is i.e. how strongly the relation is between the actual and predicted output. This method is unique, in the sense that it does not change with a scale in values. The value "1" means perfect statistical correlation and a "0" means no correlation at all. This performance measure is only used for numerical input and output.

The formula: $$\frac{\sum\left(x-x'\right)\left(y-y'\right)}{\sqrt{\sum\left(x-x'\right)^2\sum\left(y-y'\right)^2}}$$

Where x and y are the actual and predicted values while x' and y' are the mean of the actual and predicted values.

A good prediction model should have significant level (p-value) within 5%. A p-value is a measure to show the evidence against the null hypothesis. The null hypothesis represents the hypothesis of no change or no effect. P-value represents the probability of finding a co-relation by chance. In the sense of statistical significance the lower the p-value, the less likely the result is if the null hypothesis is true, and consequently the more "significant" the result is. The null hypothesis is often rejected when the p-value is less than 0.05 or 0.01.

6.1.2. Root Mean-Squared Error

The root mean-squared error is one of the most commonly used measures of success for numeric prediction. This value is computed by taking the average of the squared differences between each predicted value x_n and its corresponding actual value y_n. The root mean-squared error is simply the square root of the mean squared error. The root mean-squared error gives the error value the same dimensionality as the actual and predicted values.

$$\textit{The formula:} \quad \sqrt{\frac{(x_1 - y_1)^2 + (x_2 - y_2)^2 + \ldots (x_3 - y_3)^2}{n}}$$

Where n is the size of data.

6.1.3. Training Time

Comparing training time with prediction time, we found that prediction time is fraction of a second and negligible amount. So we discard prediction time and consider only training time as it differs excessively with training algorithm. It is computed as follows:

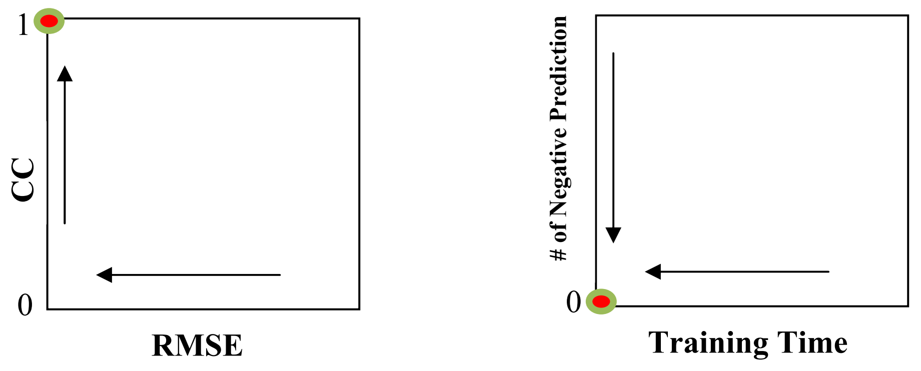
$$T_2 - T_1$$

Where T_2 is the CPU time at the end of the run and T_1 is the CPU time at the beginning of training. The prediction time is very less comparing the training time. Though it takes huge time to train, once the training is done the model can be used for prediction in no time.

6.1.4. Number of Negatively Predicted Values

As we are predicting mole fraction of gas compositions, the predicted values should not contain negative values. We counted the frequency of negative values predicted by each HCI and EHCI models and consider it as a performance measure.

We have used two metrics two represent results so as to easily compare the outcomes of the models. One metric of CC vs. RMSE and the other is number of negative prediction vs. training time. In the first metric the upper left most point indicates the best performance as we can see from the Figure 52: (a) that the upper left most point have 0 RMSE errors with highest CC value 1. In the second metric of Figure 52: (b), we can observe that the lower left corner represents the highest performance as we can see it has lowest training time with no negative predicted values.



(a): CC vs. RMSE (b): Negative Prediction vs. Training Time

Figure 52: Metrics for Performance Measure

We gave most importance to error measure of a model that is the RMSE values as long as it has an accepted CC value. In statistics CC value greater than 0.75 represents strong correlation between the predicted output and original values. We don't give too much importance to training time as long as we have lower RMSE value because once the model is trained, prediction require insignificant amount of time.

6.2. Experimental Setup

- **Individual CI Models:** MLP, SVR, ANFIS
- **Hybrid CI Models:** GA+MLP, GA+SVR, GA+ANFIS
- **Ensemble of Hybrid CI models:**
    - **Homogeneous EHCI models:**

        - Ensemble of 3 GA+MLP models
        - Ensemble of 3 GA+SVR models
        - Ensemble of 3 GA+ANFIS models
    - **Heterogeneous EHCI model:**
        - Ensemble of GA+MLP, GA+SVR and GA+ANFIS models.

The Table 4 shows the Training data percent that is randomly selected from the training set to train the each model. The rest of the training data is used for Table 5 to Table 8 shows the optimized parameters for the CI models obtained by GA and the corresponding GA parameters.

Table 4: Training Data Percent (X%) from Training Dataset

Component	GA + CI	EN_of_NN+SVR+ANFIS	EN_of_MLP	EN_of_SVR	EN_of_ANFIS
N2	80	70	70	70	70
CO2	90	70	70	70	70
H2S	80	80	80	80	80
C1	70	80	80	80	80
C2	80	80	80	80	80
C3	90	60	60	60	60

Table 5: Optimized Parameters for ANFIS

Parameters	GA+ANFIS				
Component	**radius**	**# of Rules Generated**	**pop**	**gen**	**crfn**
N2	0.2998	65	10	5	0.65
CO2	0.6120	13	50	20	0.65
H2S	0.7959	9	50	20	0.65
C1	0.6062	33	50	10	0.65
C2	0.6141	32	50	20	0.5
C3	0.5533	27	10	5	0.9

Table 6: Optimized Parameters for MLP

Parameters	GA+MLP									
Component	**Hidden Nodes**	**HL Act Fn**	**OL Act Fn**	**Tr Alg**	**Epoche**	**Lr Rate**	**Error Goal**	**pop**	**gen**	**crfn**
N2	56	logsig	tansig	trainlm	10	0.001	0.00001	10	5	0.65
CO2	21	tansig	tansig	trainlm	9	0.001	0.00001	50	20	0.65
H2S	17	logsig	tansig	trainlm	10	0.001	0.00001	50	20	0.65
C1	5	tansig	tansig	trainlm	13	0.001	0.00001	50	10	0.65
C2	26	logsig	tansig	trainlm	14	0.001	0.00001	50	20	0.65
C3	8	logsig	tansig	trainlm	15	0.001	0.00001	10	5	0.9

Table 7: Optimized Parameters for SVR

Parameters	GA+SVR								
Component	**C**	**λ**	**ε**	**# of SV**	**Kernel**	**Kernel Op**	**pop**	**gen**	**crfn**
N2	0.9763	0.000666782	0.1754	17	poly	0.5	10	5	0.65
CO2	6.2202	0.000503941	0.0352	54	poly	0.5	50	20	0.65
H2S	0.8796	0.000435202	0.0001	77	poly	0.5	50	20	0.65
C1	1.2912	0.000788126	0.2410	33	poly	0.5	50	10	0.65
C2	0.4764	1.95766E-06	0.0541	51	poly	0.5	50	20	0.65
C3	1.9763	0.000666782	0.1912	48	poly	0.5	10	5	0.9

6.3. Results & Discussions

In this work, the non-hydrocarbons and the hydrocarbons that occupy most of the volume out of twelve in a multi-stage separator are predicted. The non-hydrocarbons Nitrogen (N_2), Carbon dioxide (CO_2), Hydrogen Sulfide (H_2S) and the mostly dense hydrocarbons Methane (CH_4 as C1), Ethane (C_2H_6 as C2) and Propane (C_3H_8 as C3), i.e. altogether 6 gas components are predicted. We have showed the performance of each model in the following figures. The upper two figures of each box mainly depict the performance in terms of the metrics RMSE vs. CC and # of negative prediction vs. training time. The black square spot (■) in the Figures represents the performance of the benchmark model CPCP. The lower Figures of each boxes show the regression analysis of the prediction of training data as well as the test data for the best performed model.

6.3.1. Nitrogen (N_2)

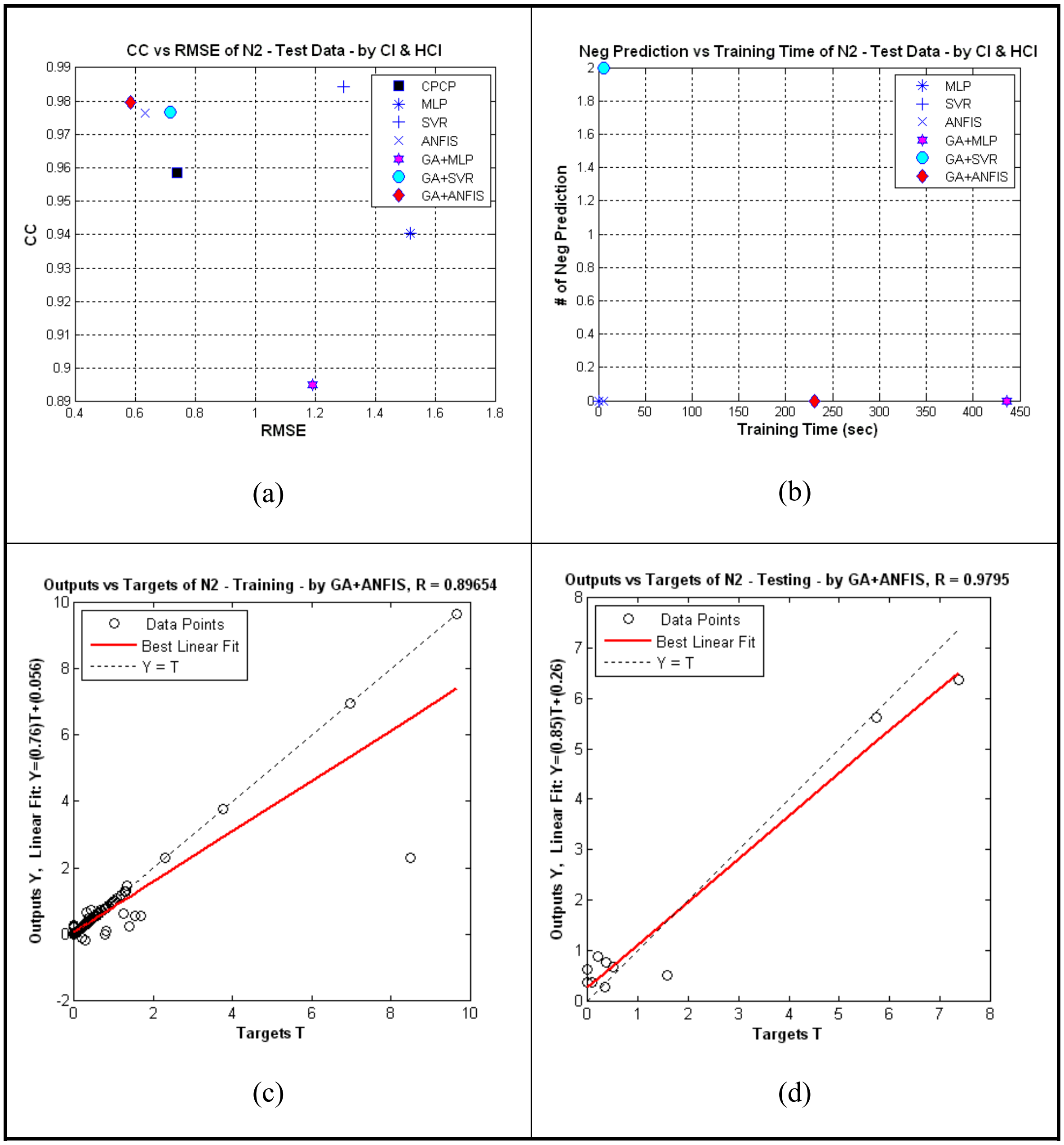


Figure 53: Performance of CI and HCI for N_2 Prediction

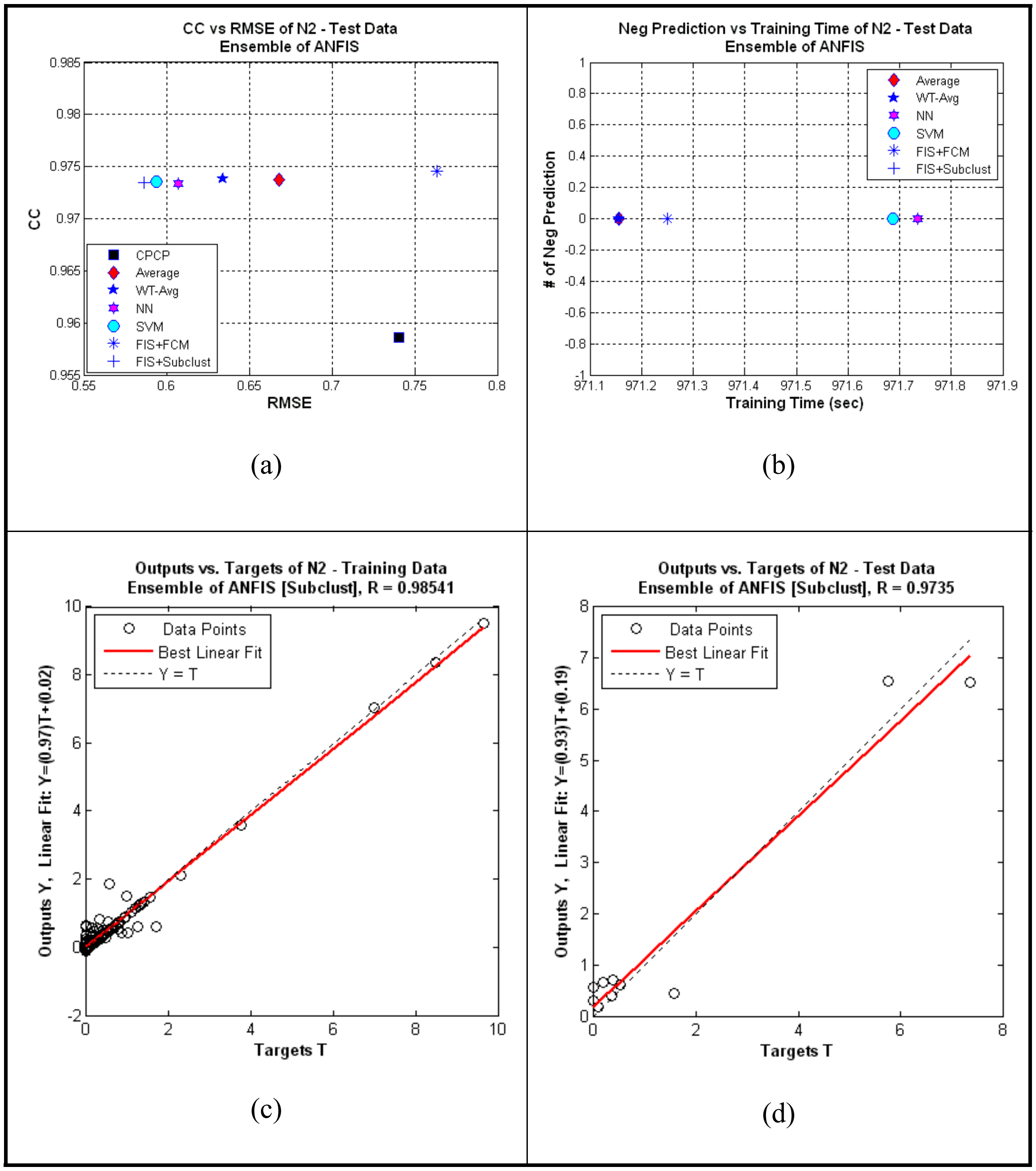


Figure 54: Performance of Ensemble of ANFIS for N_2 Prediction

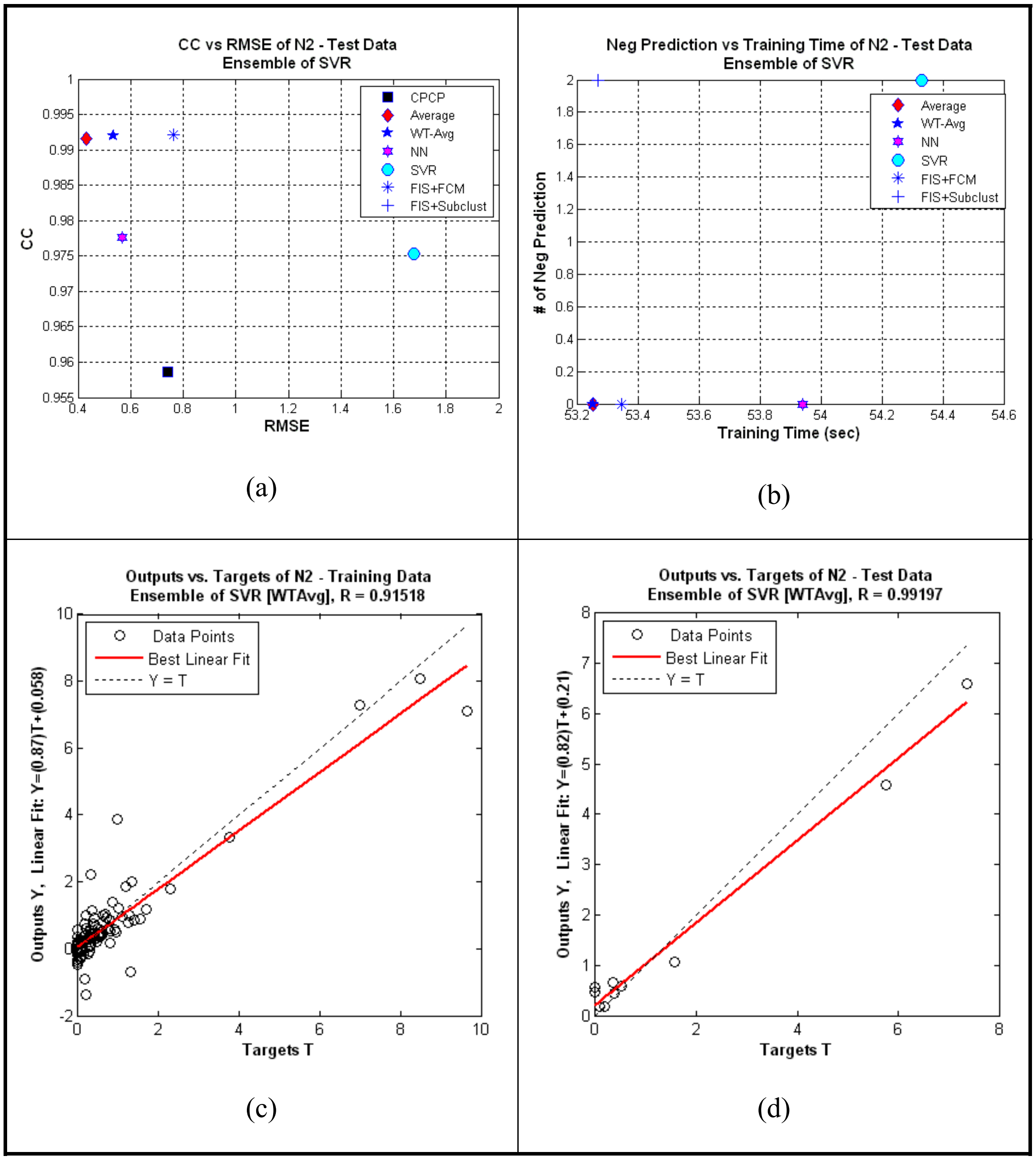


Figure 55: Performance of Ensemble of SVR for N_2 Prediction

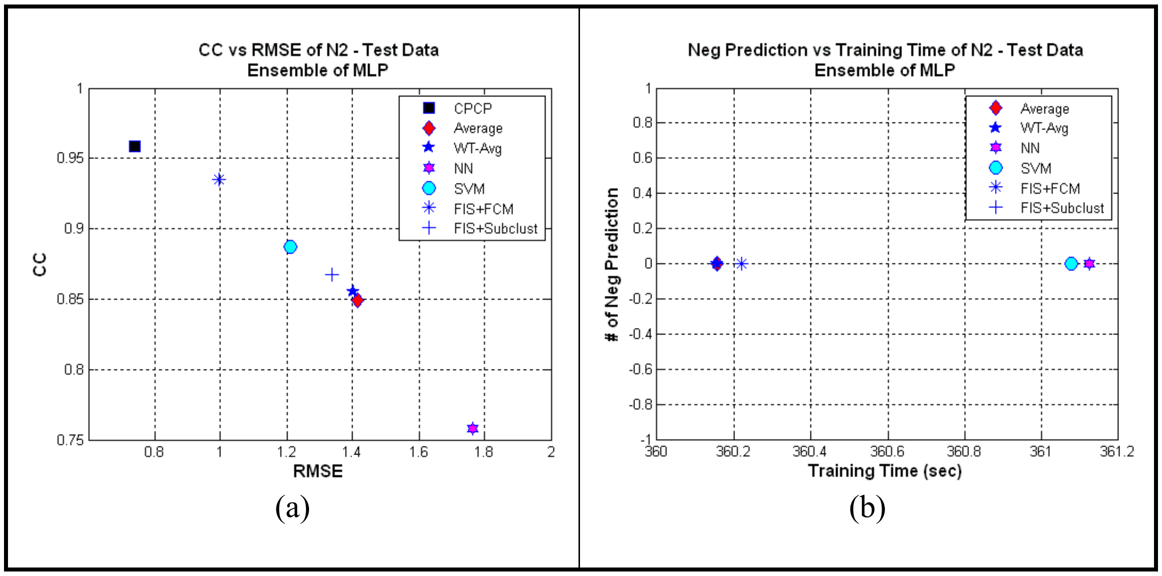


Figure 56: Performance of Ensemble of MLP for N_2 Prediction

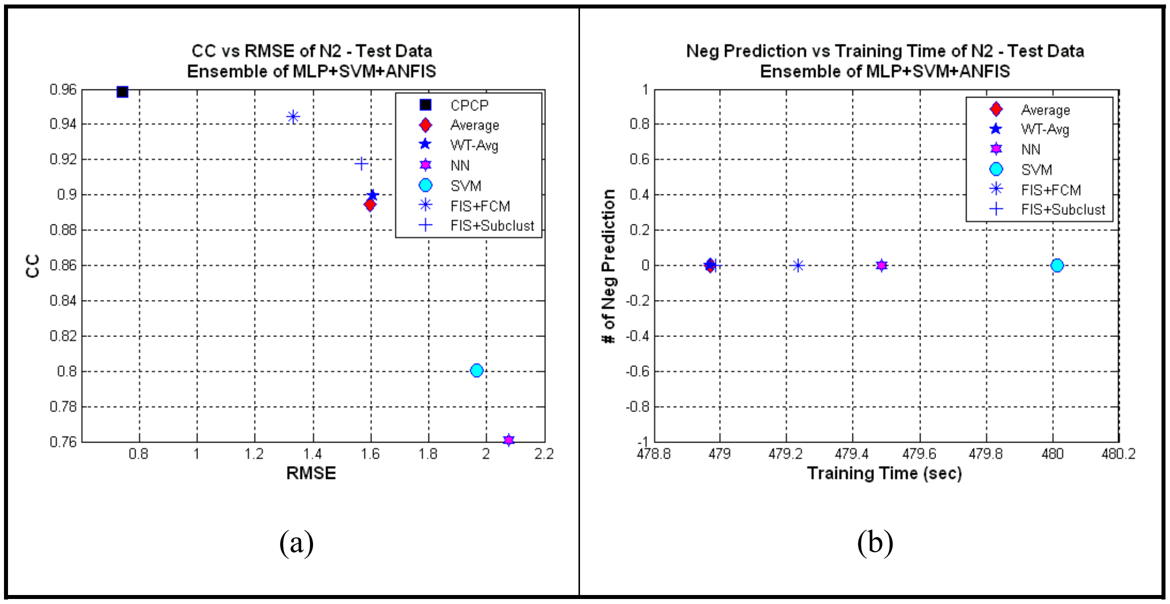


Figure 57: Performance of Heterogeneous Ensemble for N_2 Prediction

Figure 53: (a) shows that the performance of HCI model GA+ANFIS outperforms other CI, HCI and CPCP models for N2 prediction. It is noticeable that the HCI models perform better than the corresponding CI models. Figure 53: (b) shows that GA+ANFIS took less time than GA+MLP and did not predict any negative value. The regression analysis of GA+ANFIS in figure 53: (c, d) on training and testing data shows that the prediction is strongly correlated with the original values.

Figure 54: (a) shows that the EHCI model of ANFIS combined with FIS-Subclust performed better than other combiner as well as CPCP. The error RMSE value of the best model GA+ANFIS in Figure 53: (a) is near to 0.6 whereas the EHCI model of ANFIS combined with FIS-Subclust is much lower than 0.6. Nevertheless EHCI model of SVR with average combining method in Figure 55: (a) shows that the RMSE value is near to 0.4 which is much lower than the previous models.

Figures 56 and 57 show the other ECHI models' performance on predicting N2 in separators gas compositions.

6.3.2. Carbon dioxide (CO_2)

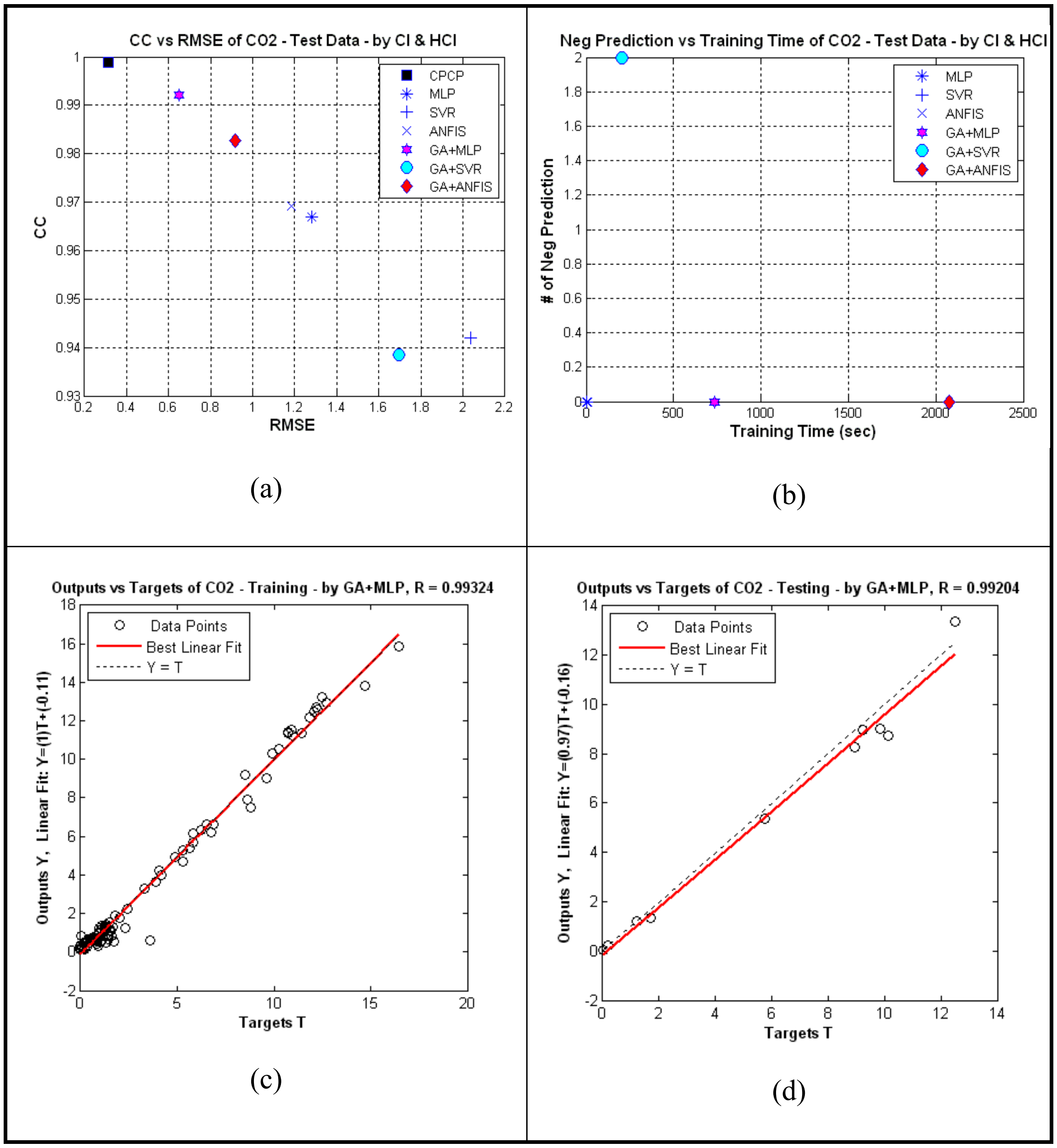


Figure 58: Performance of CI and HCI for CO_2 Prediction

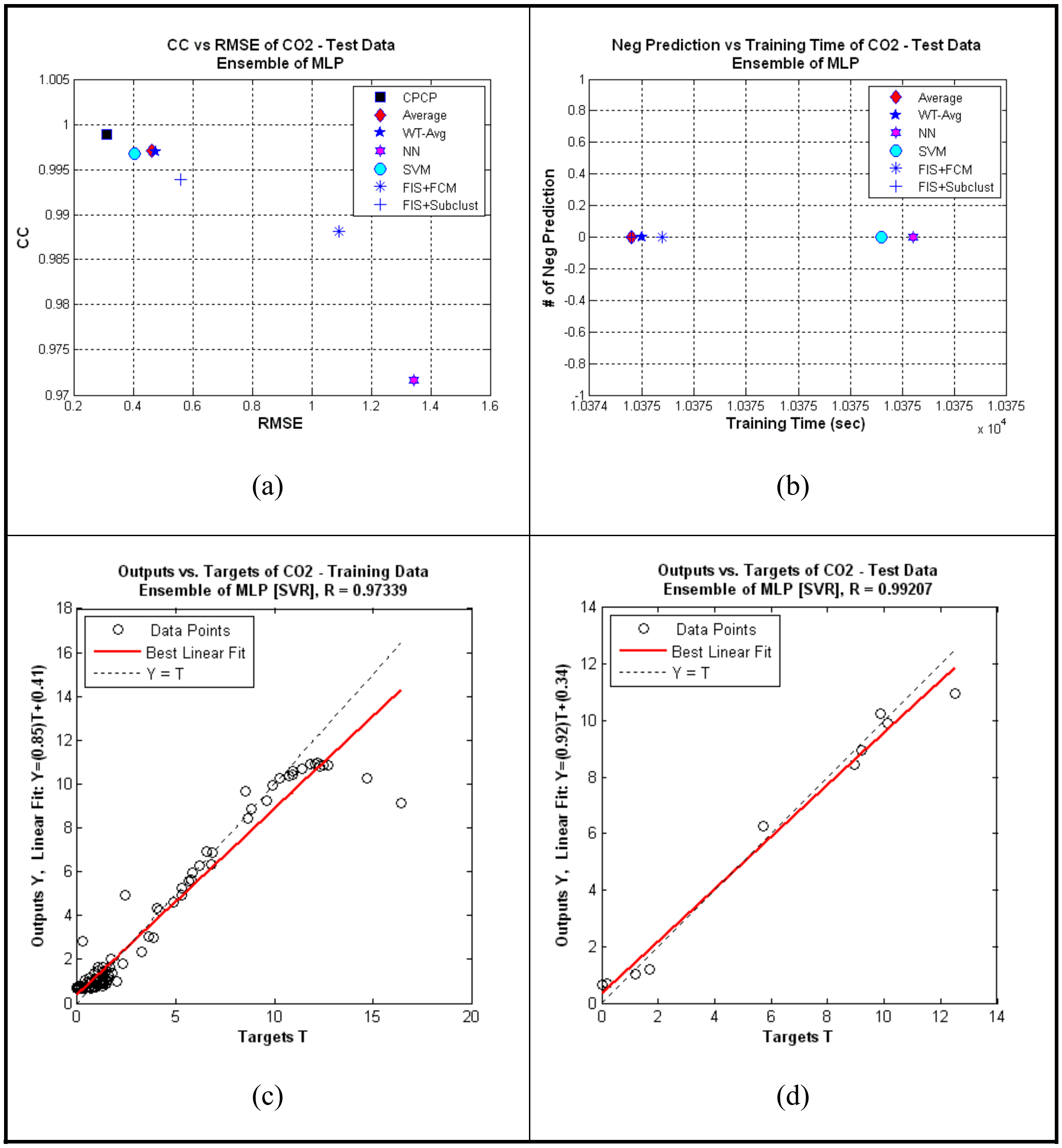


Figure 59: Performance of Ensemble of MLP for CO_2 Prediction

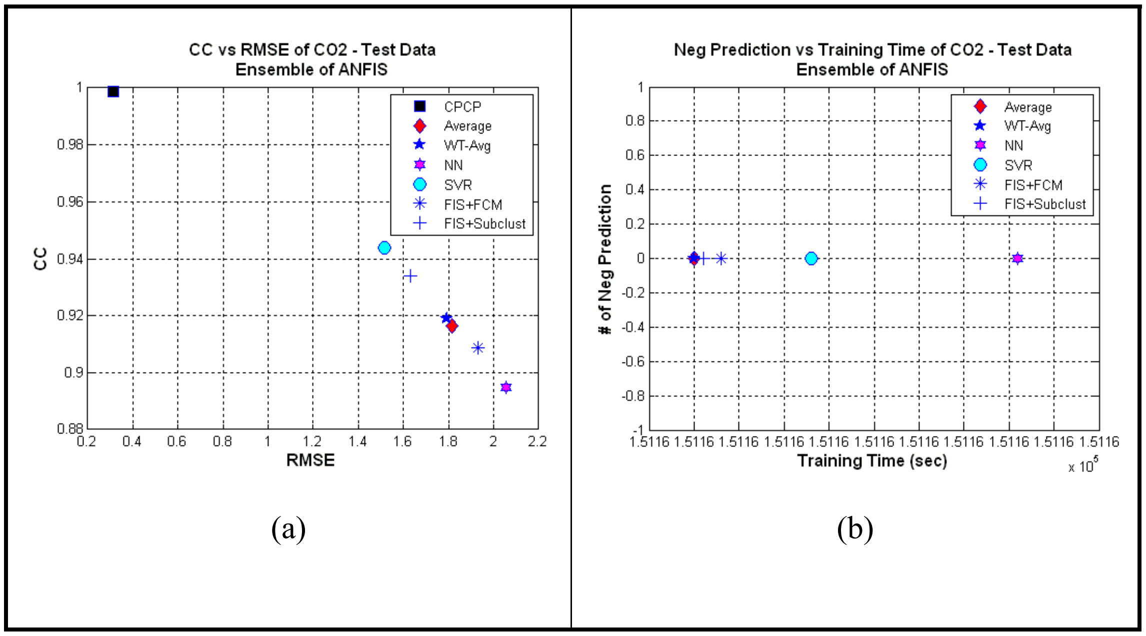


Figure 60: Performance of Ensemble of ANFIS for CO_2 Prediction

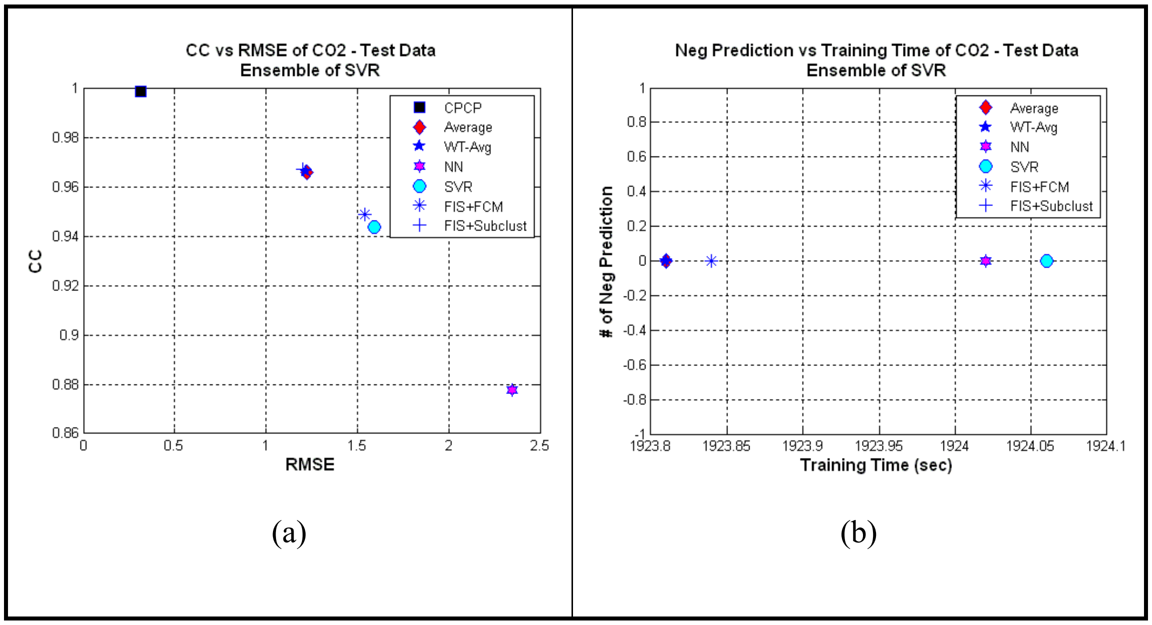


Figure 61: Performance of Ensemble of SVR for CO_2 Prediction

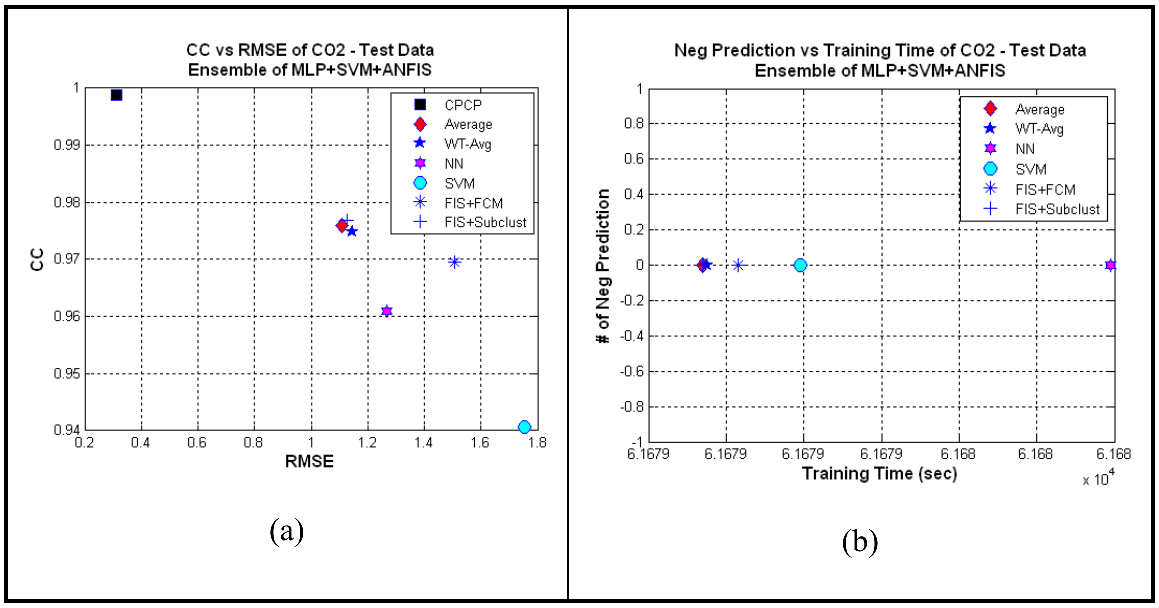


Figure 62: Performance of Heterogeneous Ensemble for CO_2 Prediction

Figure 58: (a) shows that the performance of HCI model GA+MLP outperforms other CI, HCI models for CO_2 prediction. It is noticeable that the HCI models perform better than the corresponding CI models. Figure 58: (b) shows that GA+MLP took less time than GA+ANFIS and did not predict any negative value. The regression analysis of GA+MLP in Figure 53: (c, d) on training and testing data shows that the prediction is strongly correlated with the original values.

Figure 59: (a) shows that the EHCI model of MLP combined with SVR performed better than other combiner. The error RMSE value of the best model GA+MLP in Figure 58: (a) is above 0.6 whereas the EHCI model of MLP combined with SVR is 0.4.

Figures 60, 61 and 62 show the other ECHI models' performance on predicting CO_2 in separators gas compositions.

6.3.3. Hydrogen Sulfide (H_2S)

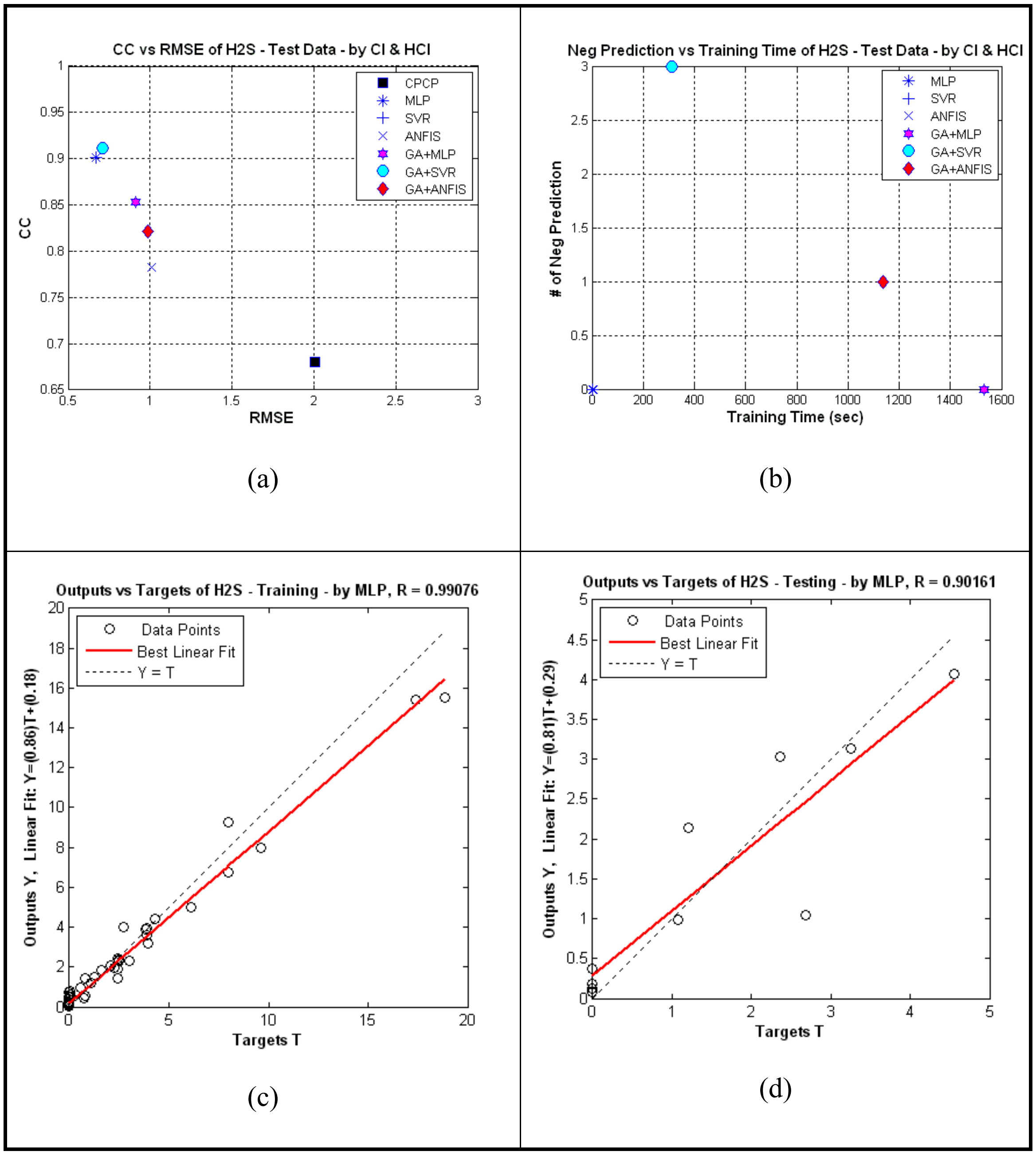


Figure 63: Performance of CI and HCI for H_2S Prediction

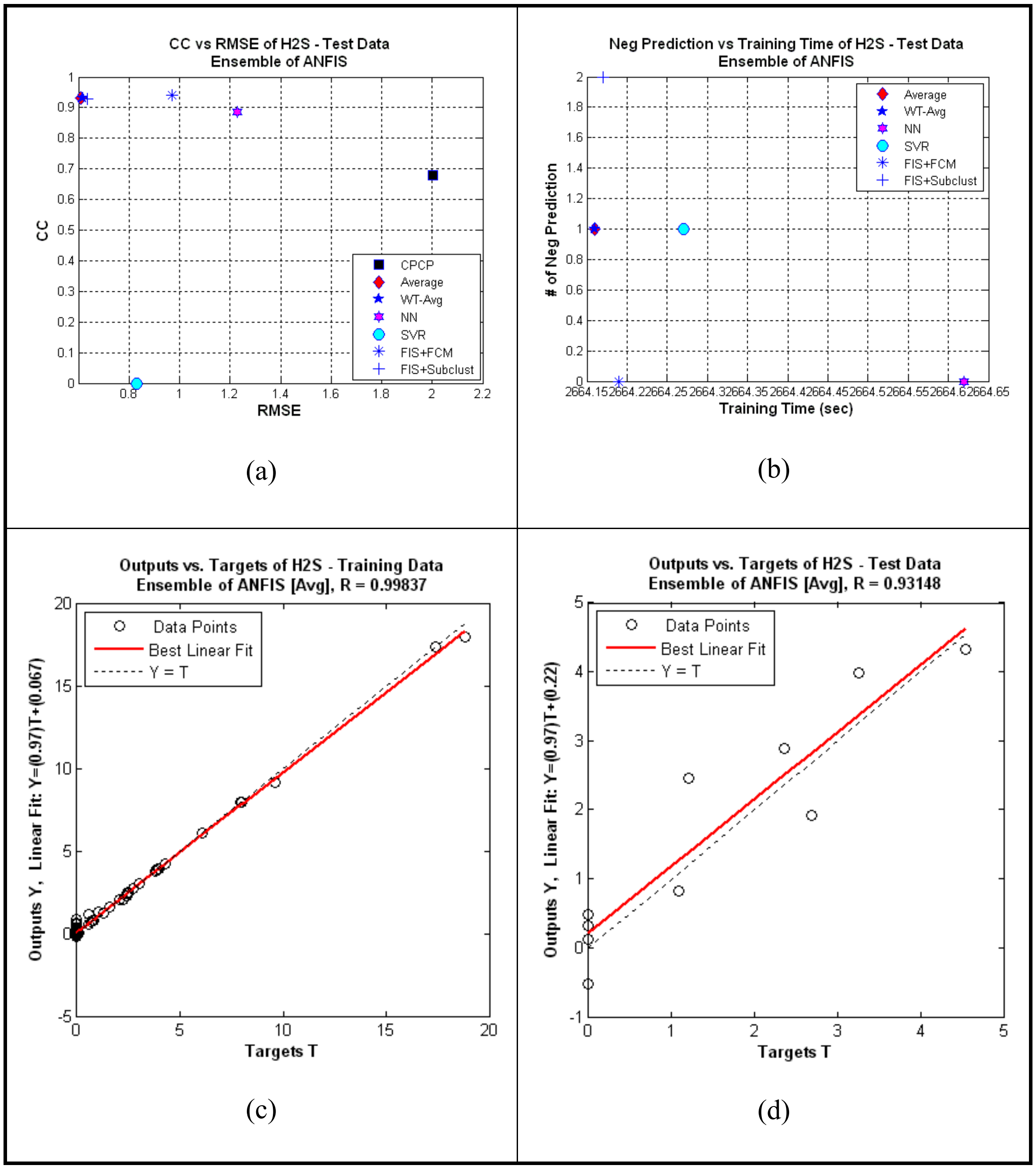


Figure 64: Performance of Ensemble of ANFIS for H_2S Prediction

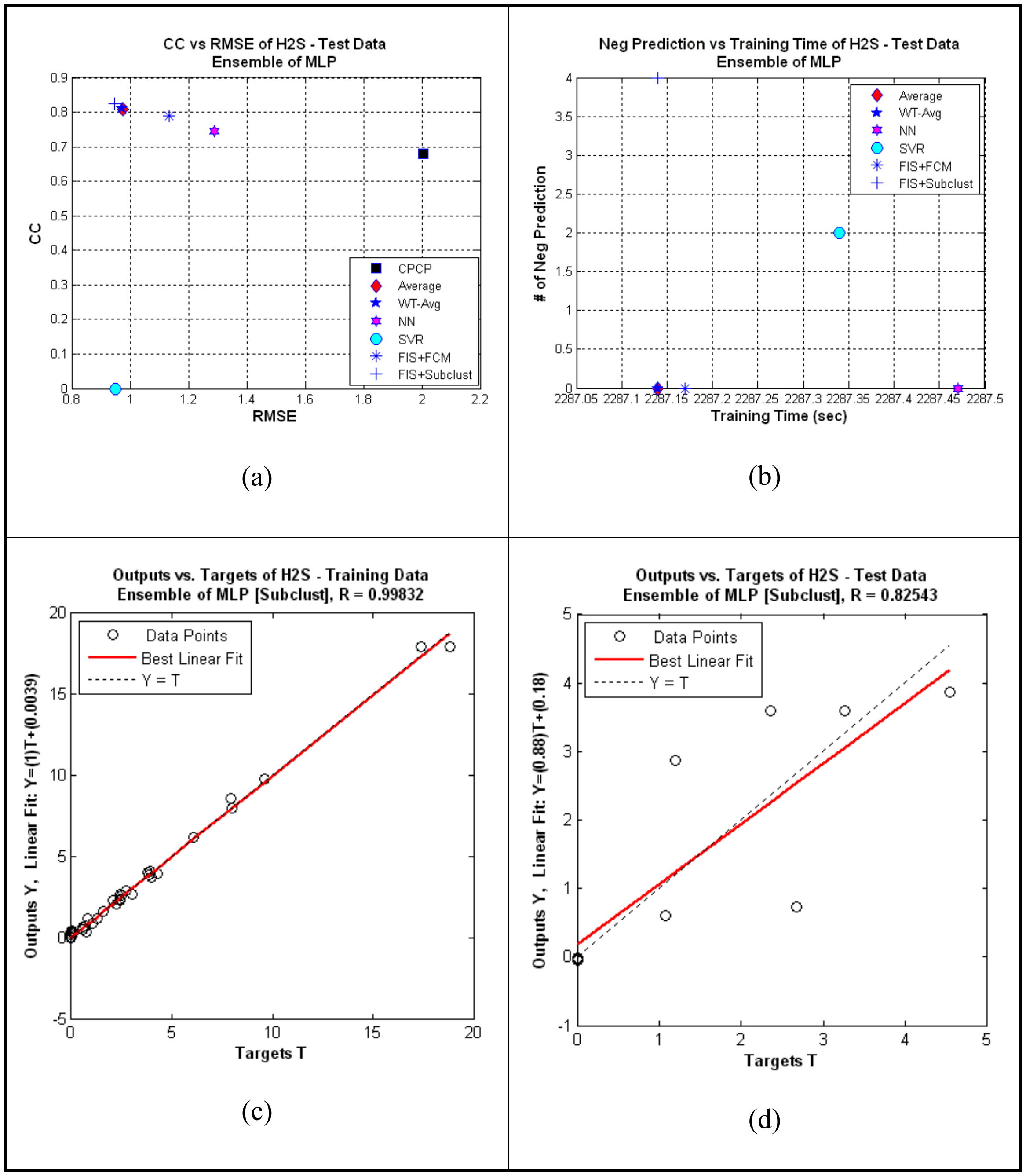


Figure 65: Performance of Ensemble of MLP for H_2S Prediction

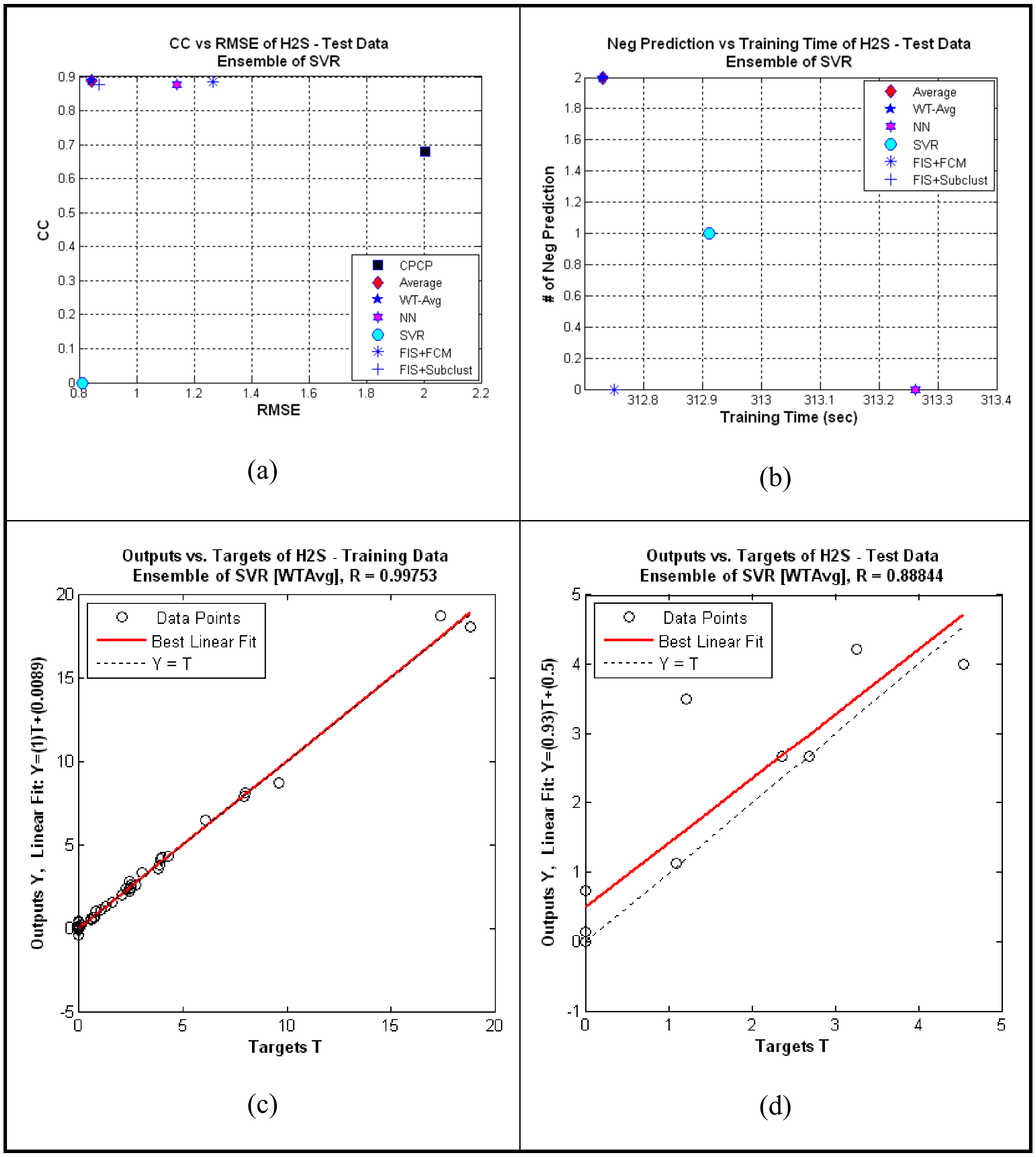


Figure 66: Performance of Ensemble of SVR for H_2S Prediction

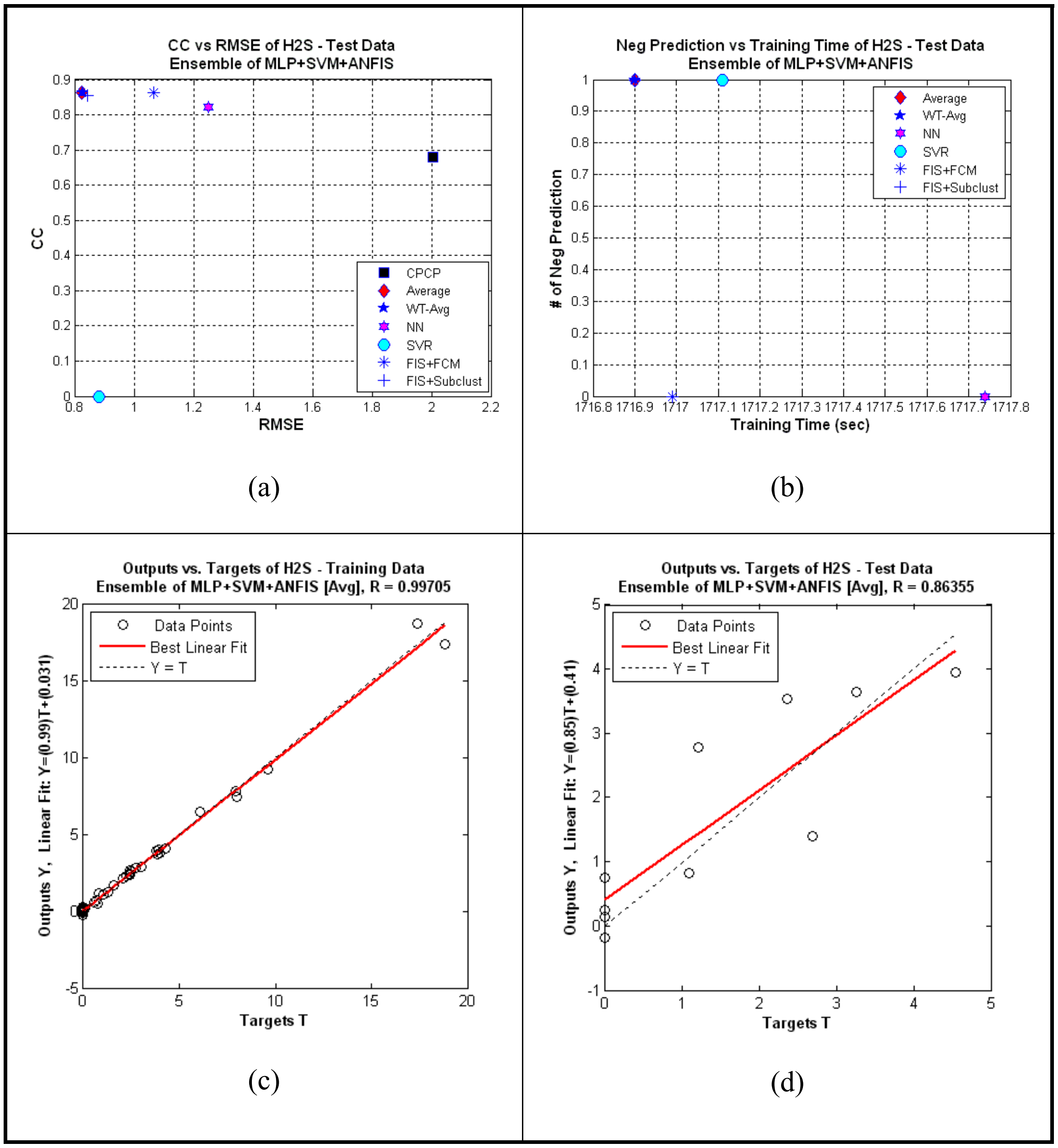


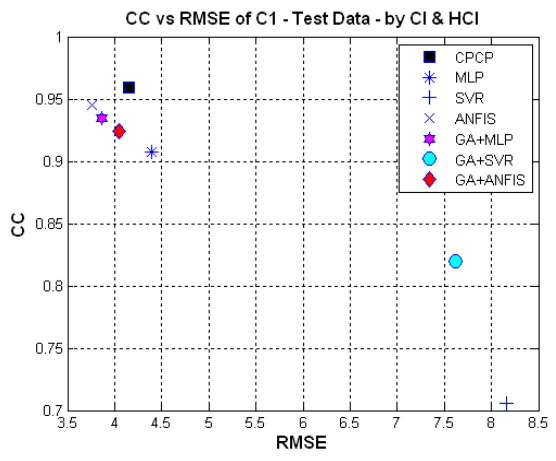
Figure 67: Performance of Heterogeneous Ensemble for H_2S Prediction

Figure 63: (a) shows that the performance of CI model MLP outperforms other CI, HCI and CPCP models for H_2S prediction. HCI models except GA+MLP performs better than the corresponding CI models. Figure 63: (b) shows that CI model MLP took minimum time comparing other CI and HCI models and did not predict any negative value. The regression analysis of MLP in Figure 63: (c, d) on training and testing data shows that the prediction is strongly correlated with the original values.

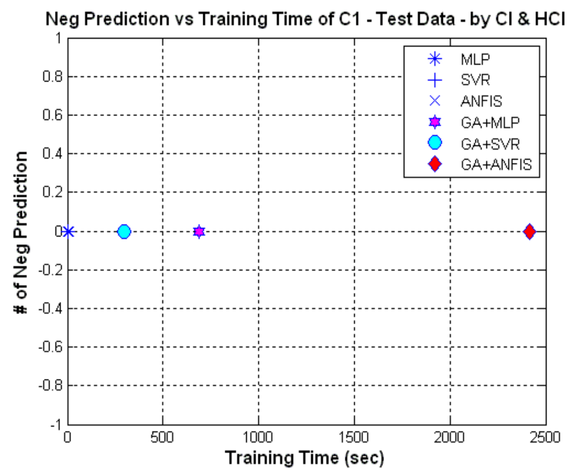
Figure 64: (a) shows that the EHCI model of ANFIS combined with simple average method performed better than other combiner as well as CPCP. The error RMSE value of the best CI model MLP in Figure 63: (a) is near to 0.7 whereas the EHCI model of ANFIS is about 0.6 with 1 negative prediction (Figure 64: (b)).

On the other hand EHCI model of MLP with FIS-Subclust combining method in Figure 65: (a), EHCI model of SVR with weighted average method (Figure 66: (a)) and heterogeneous EHCI model (Figure 67: (a)) performance are not as good as EHCI model of ANFIS but all the models performs better than CPCP.

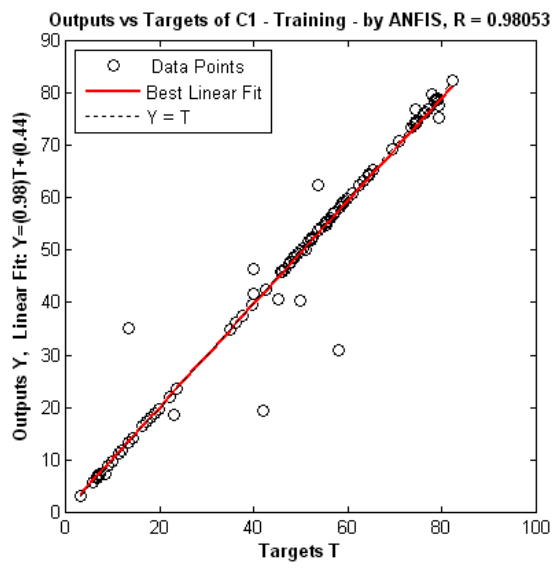
6.3.4. Methane (CH_4 as C1)



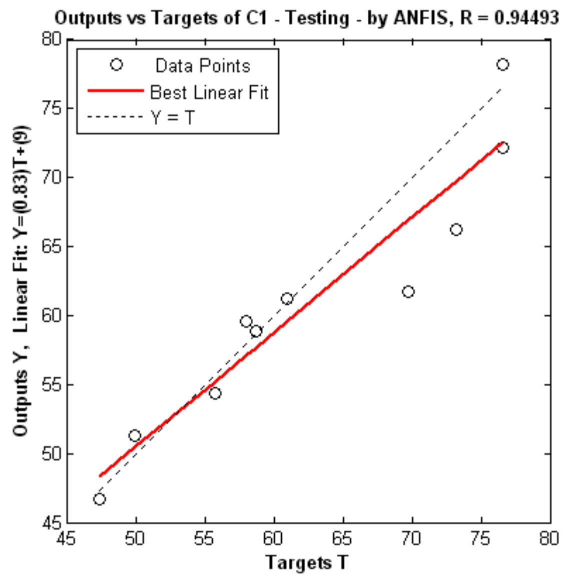
(a)



(b)

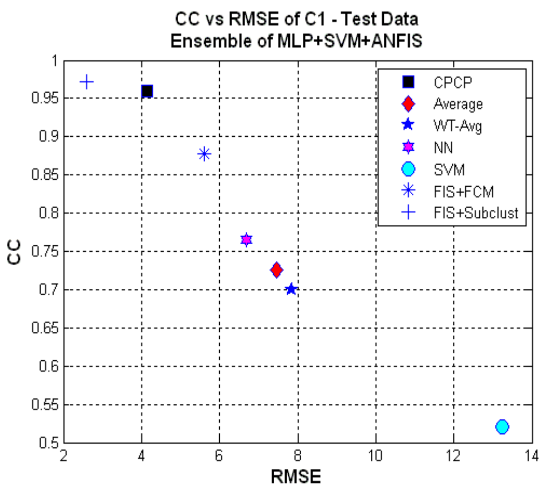


(c)

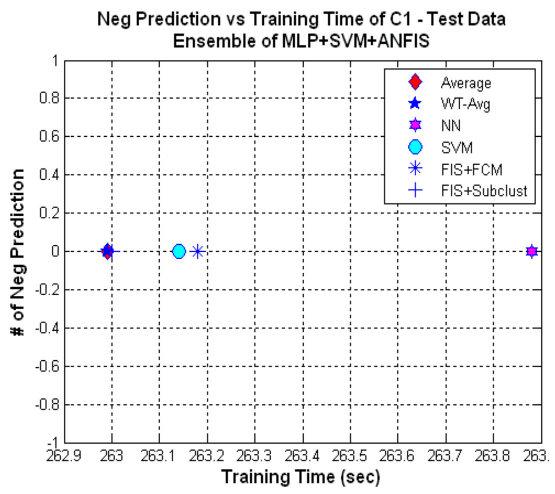


(d)

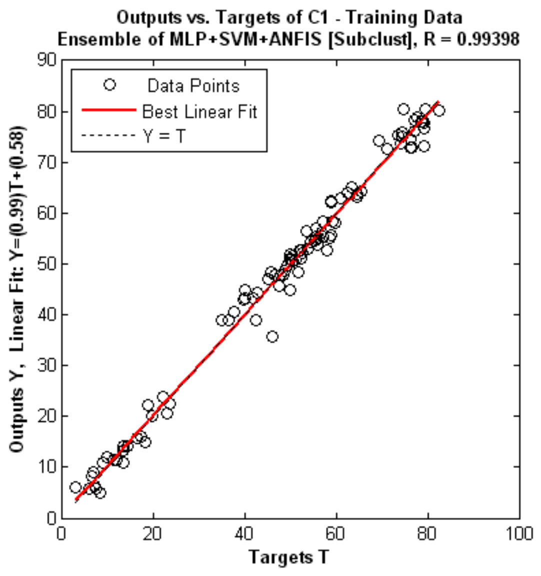
Figure 68: Performance of CI and HCI for C1 Prediction



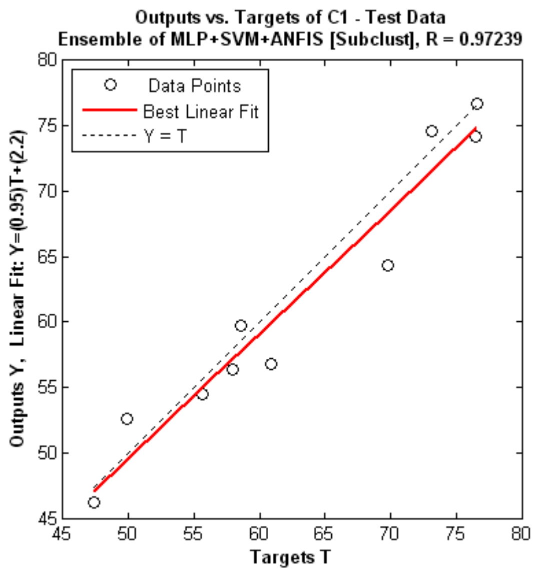
(a)



(b)

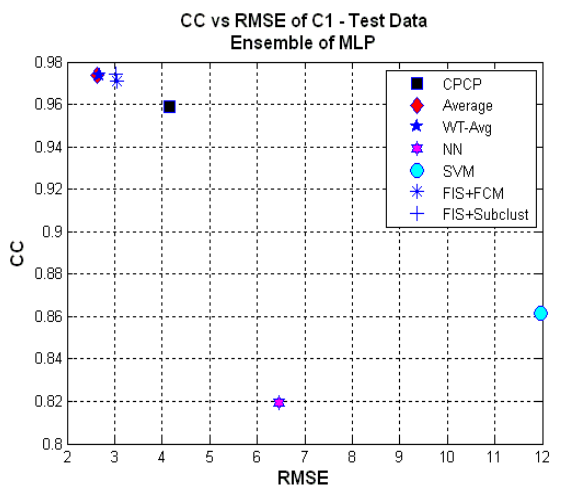


(c)

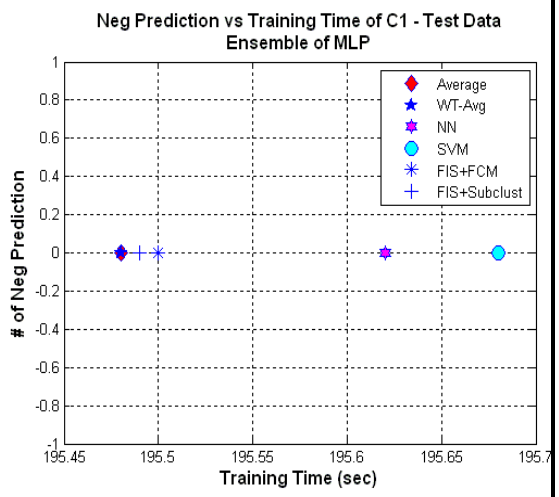


(d)

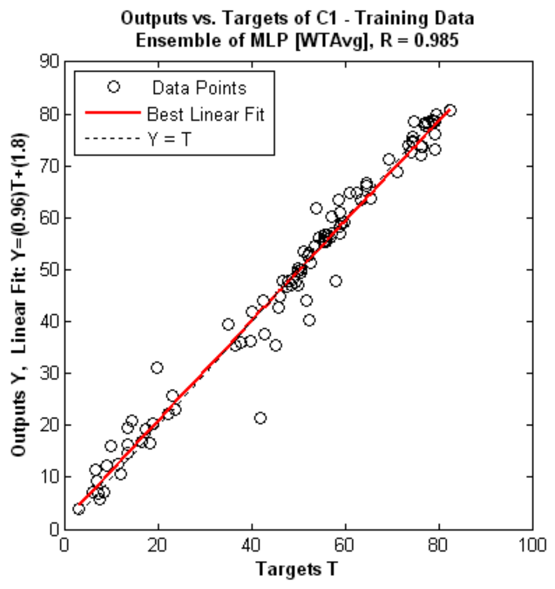
Figure 69: Performance of Heterogeneous Ensemble for C1 Prediction



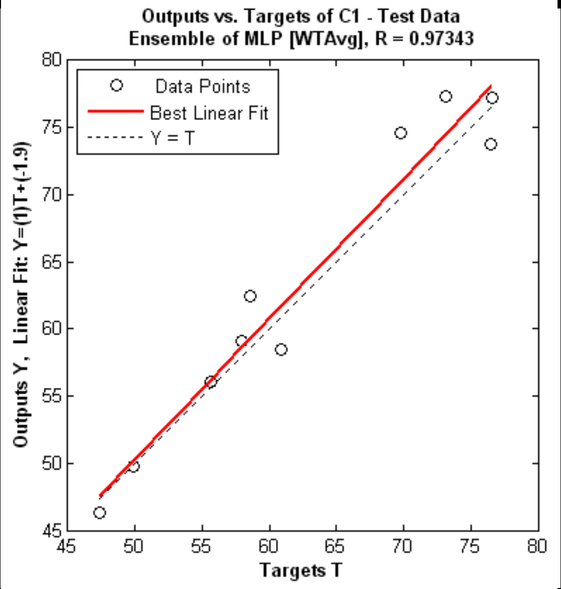
(a)



(b)



(c)



(d)

Figure 70: Performance of Ensemble of MLP for C1 Prediction

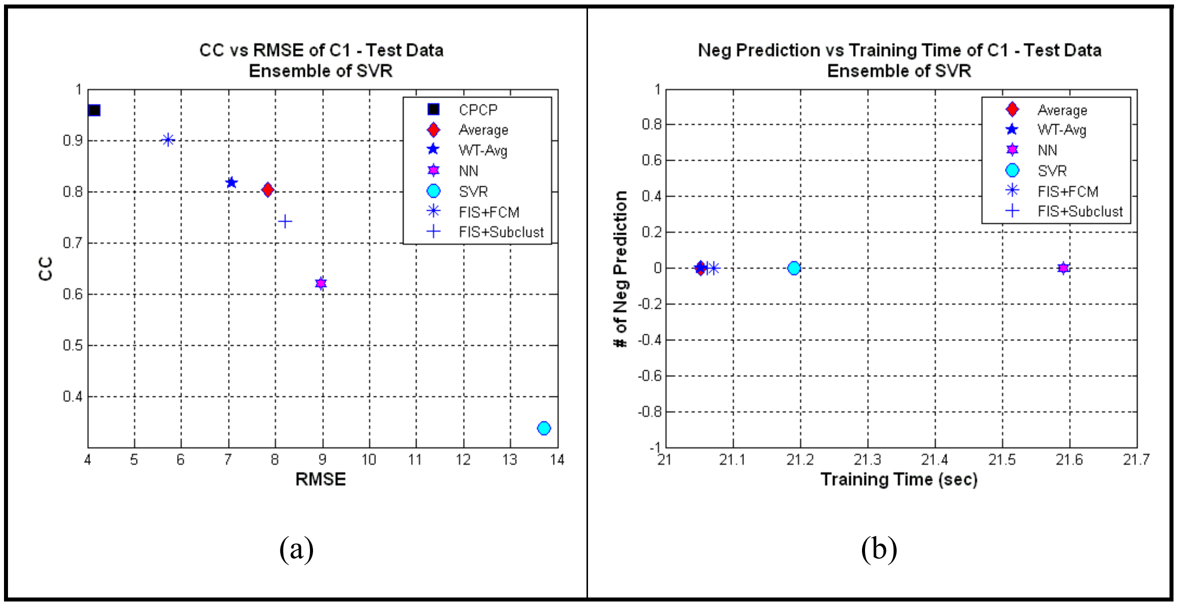


Figure 71: Performance of Ensemble of SVR for C1 Prediction

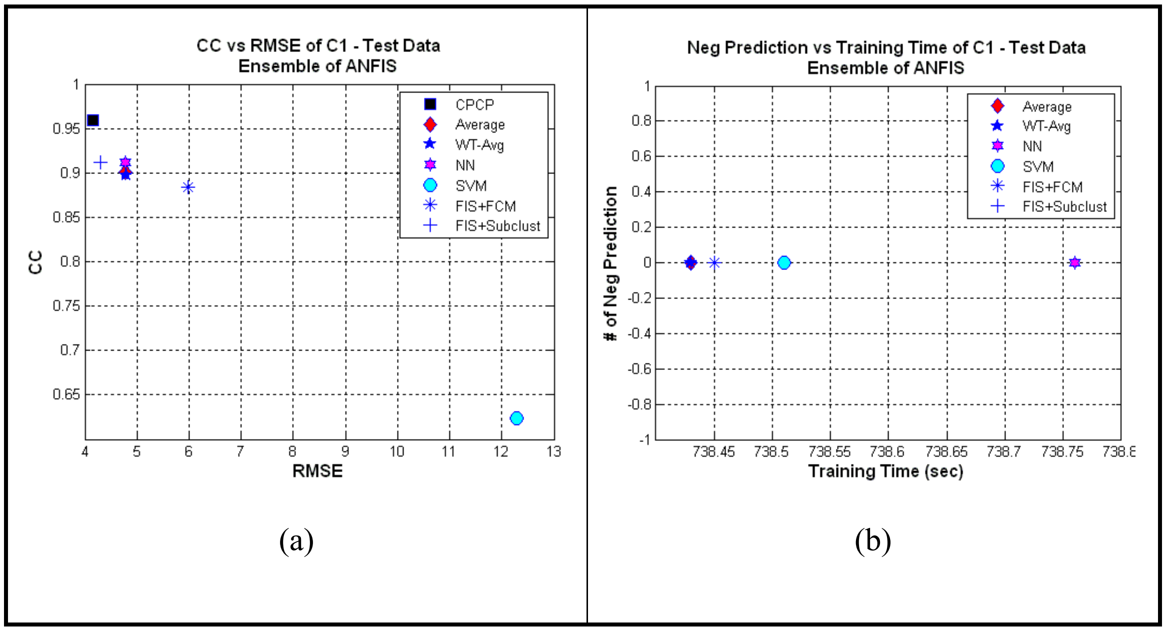


Figure 72: Performance of Ensemble of ANFIS for C1 Prediction

Figure 68: (a) shows that the performance of CI model ANFIS outperforms all the CI, HCI and CPCP models for C1 prediction. It is noticeable that the HCI models perform better than the corresponding CI models except ANFIS. Figure 68: (b) shows that ANFIS took less time along with MLP and SVR than the HCI models with no negative prediction. The regression analysis of GA+ANFIS in Figure 68: (c, d) on training and testing data shows that the prediction is strongly correlated with the original values.

Figure 69: (a) shows that the heterogeneous EHCI model combined with FIS-Subclust performed better than other combiner as well as CPCP. The error RMSE value of the best model ANFIS in Figure 68: (a) is near to 3.75 whereas the RMSE of heterogeneous EHCI model is about 2.5 (Figure 69: (a)). Nevertheless EHCI model of MLP with weighted average combining method in Figure 70: (a) shows that the RMSE value is near to 2.5.

Figures 71 and 72 show the other ECHI models' performance on predicting C1 in separators gas compositions.

6.3.5. Ethane (C_2H_6 as C2)

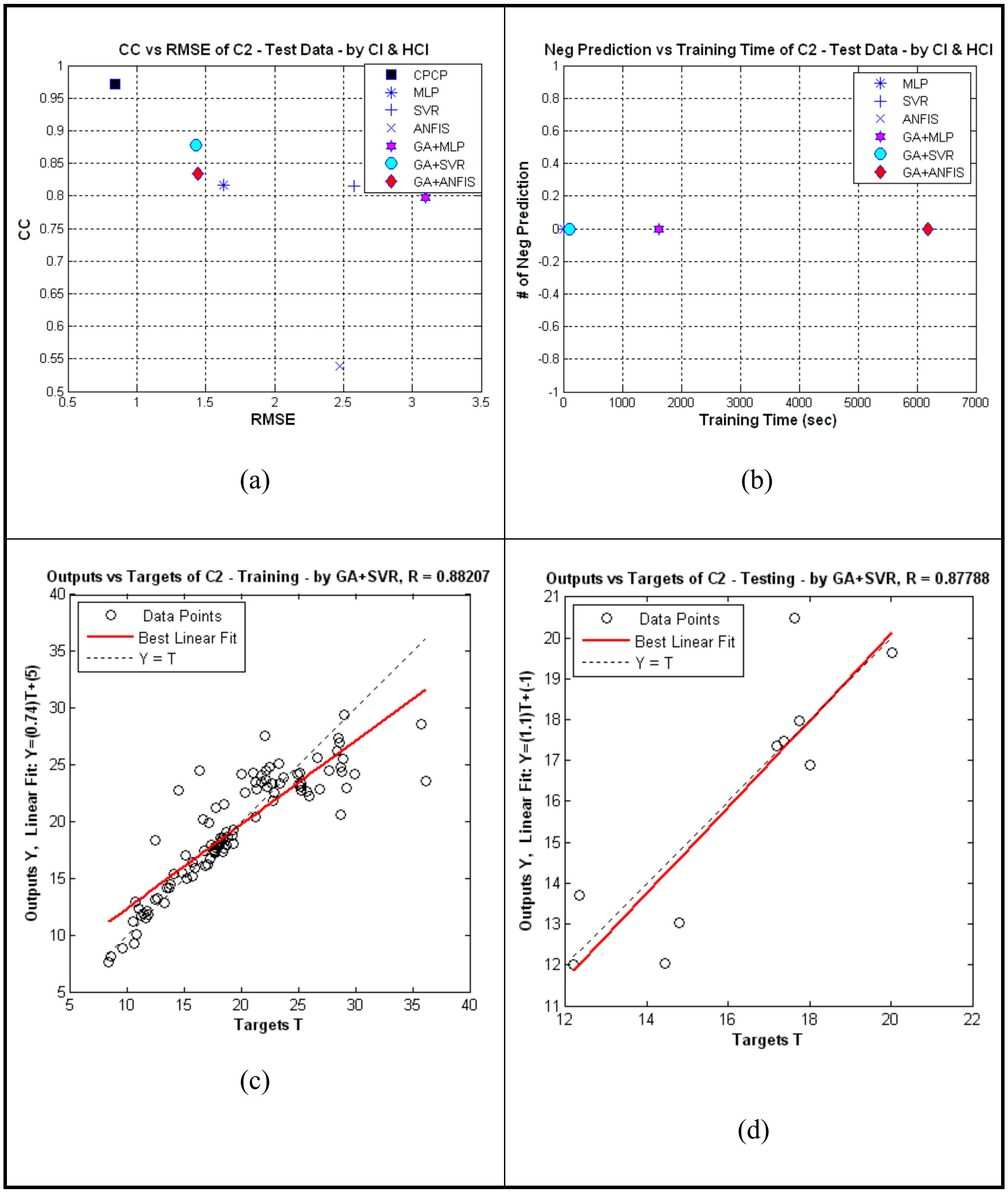


Figure 73: Performance of CI and HCI for C2 Prediction

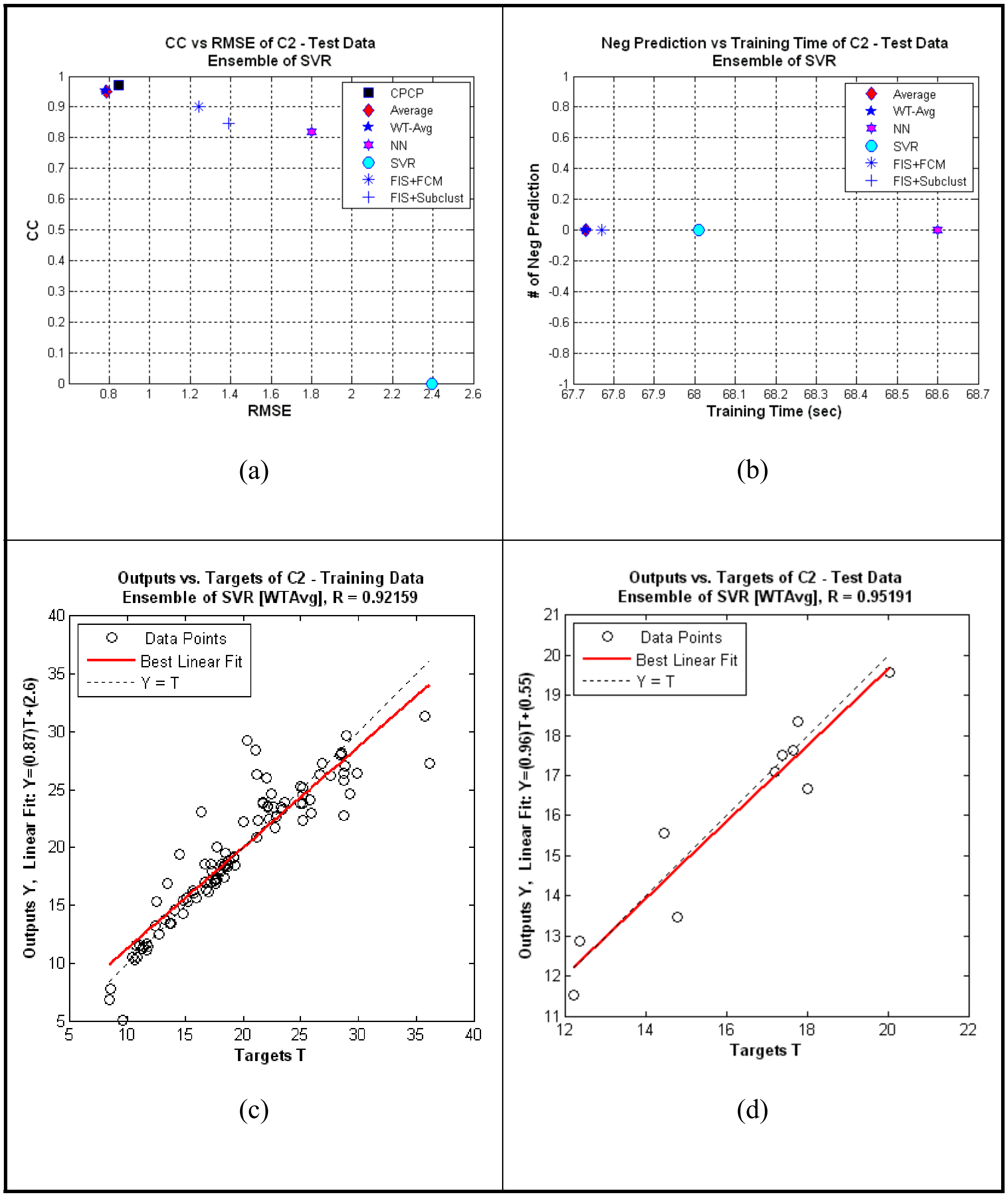


Figure 74: Performance of Ensemble of SVR for C2 Prediction

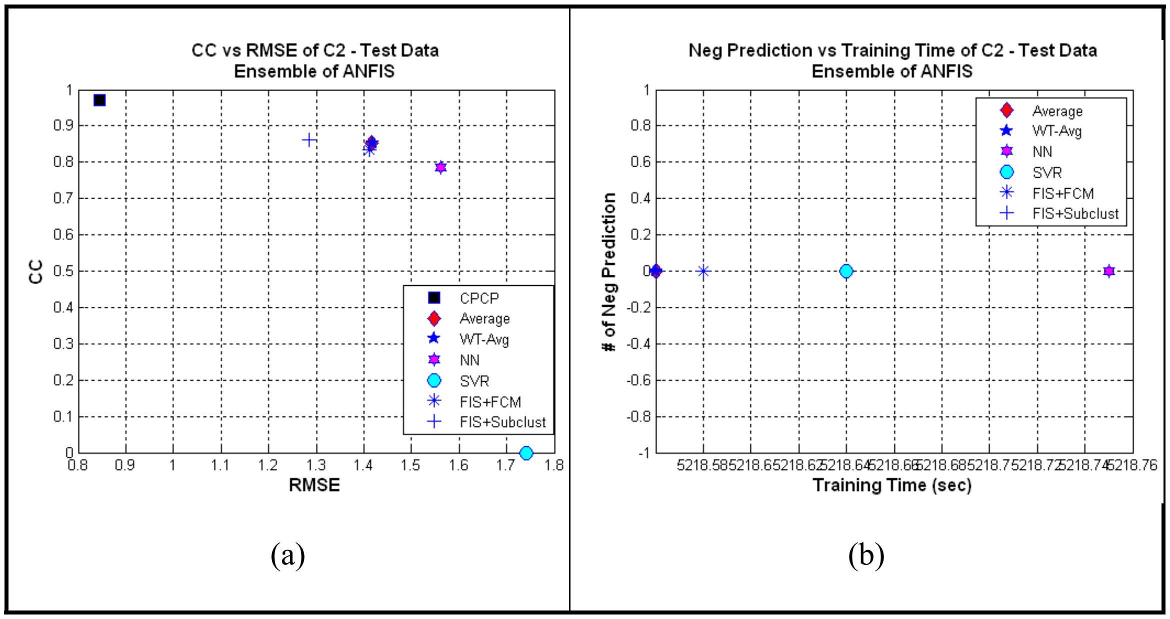


Figure 75: Performance of Ensemble of ANFIS for C2 Prediction

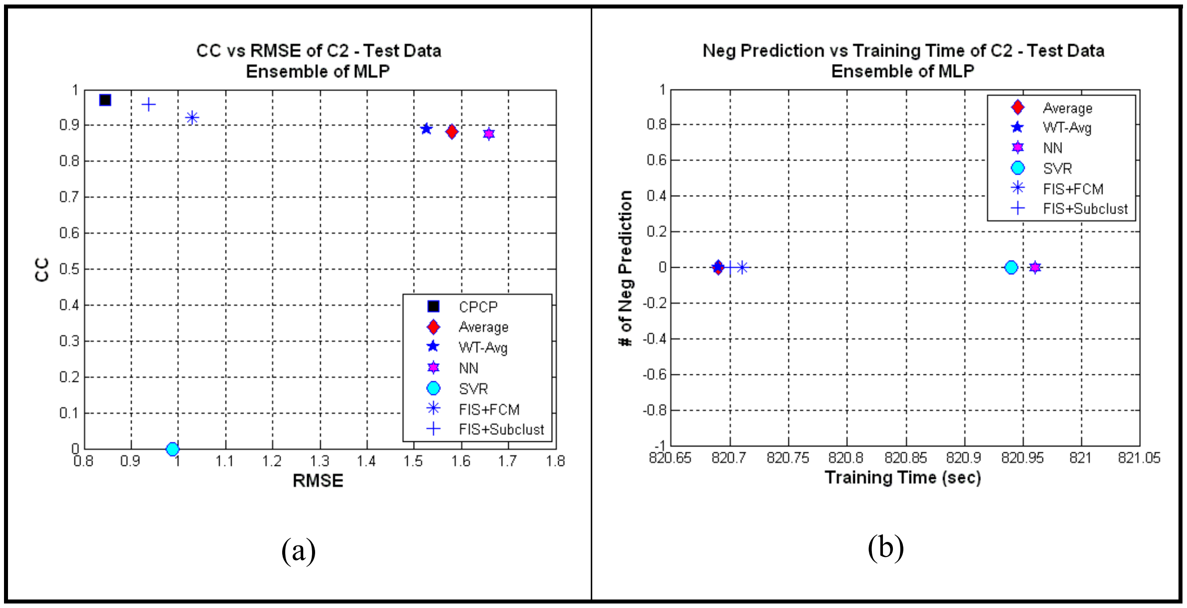


Figure 76: Performance of Ensemble of MLP for C2 Prediction

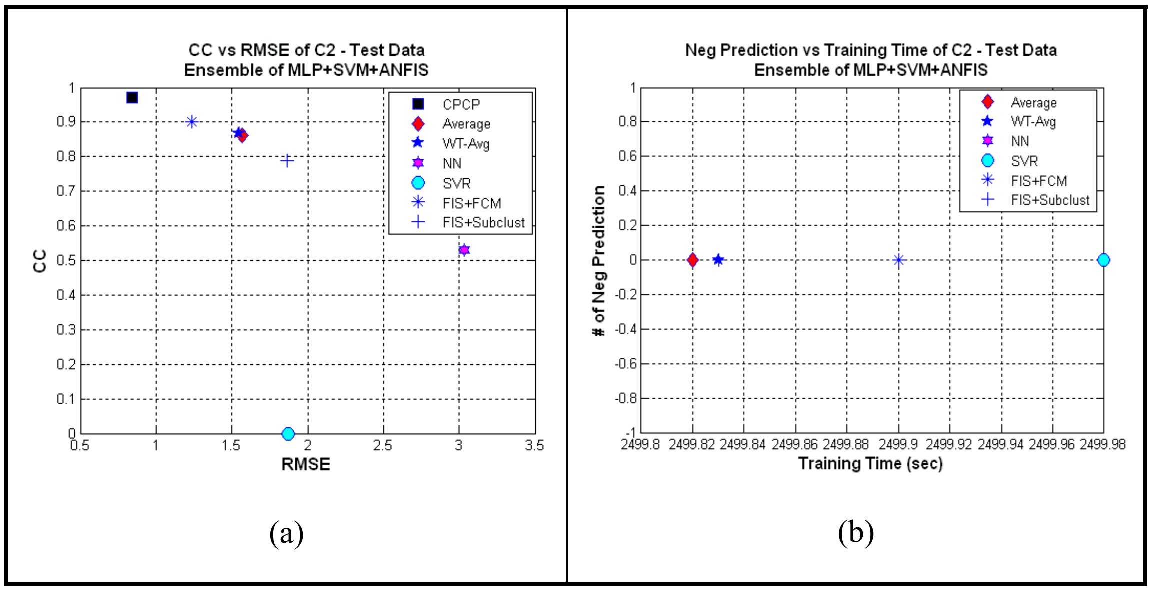


Figure 77: Performance of Heterogeneous Ensemble for C2 Prediction

Figure 73: (a) shows that the performance of HCI model GA+SVR outperforms the other CI, HCI models for C2 prediction. It is noticeable that the HCI models perform better than the corresponding CI models except GA+MLP. Figure 73: (b) shows that GA+SVR took less time than other HCI models and did not predict any negative value. The regression analysis of GA+SVR in Figure 73: (c, d) on training and testing data shows that the prediction is strongly correlated with the original values.

Figure 74: (a) shows that the EHCI model of SVR combined with weighted average method performed better than other combiner. The error RMSE value of the best model GA+SVR in Figure 73: (a) is near 1.5 whereas the EHCI model of SVR has RMSE about 0.8. Figures 75, 76 and 77 show the other ECHI models' performance on predicting CO_2 in separators gas compositions.

6.3.6. Propane (C_3H_8 as C3)

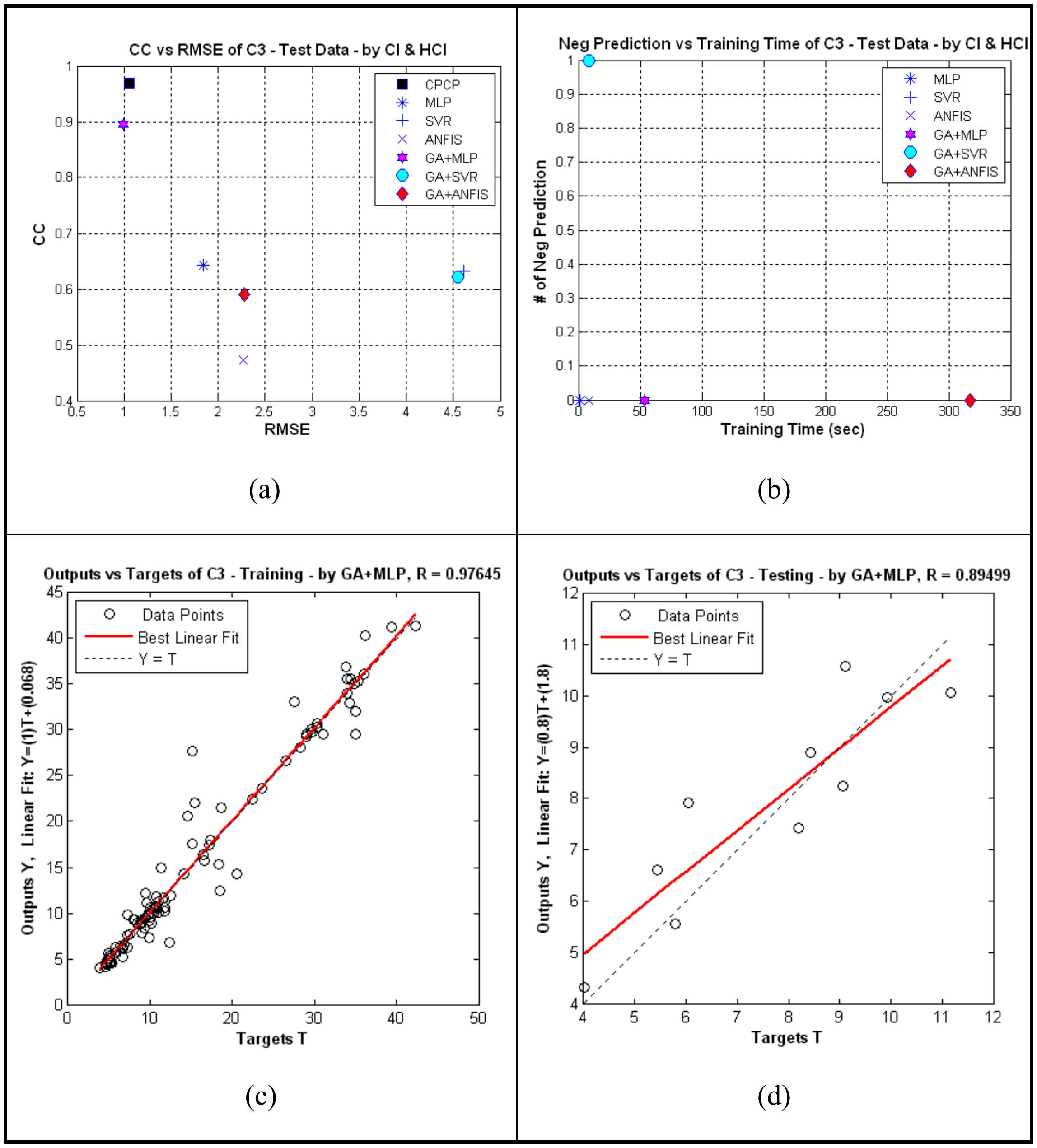


Figure 78: Performance of CI and HCI for C3 Prediction

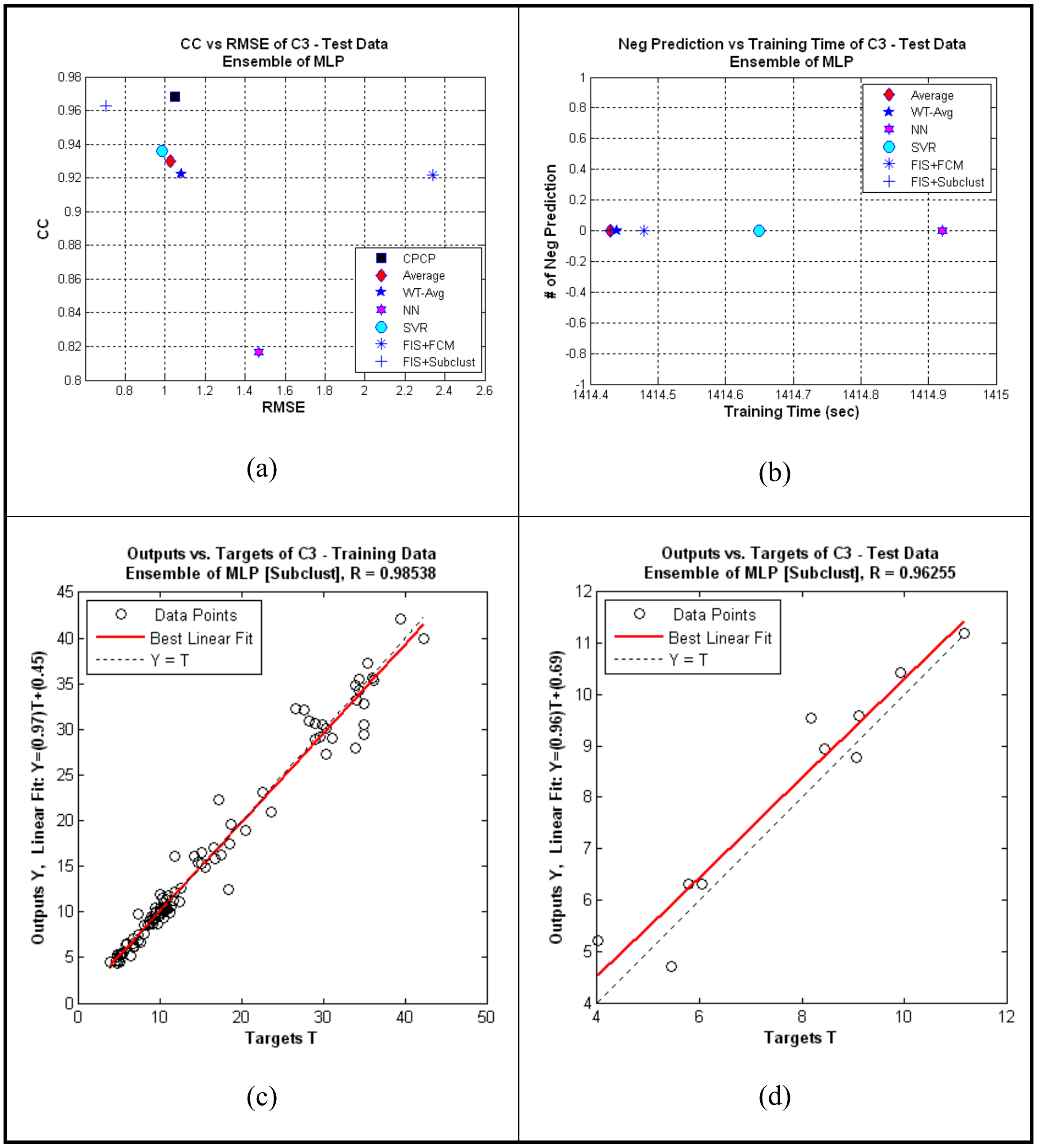


Figure 79: Performance of Ensemble of MLP for C3 Prediction

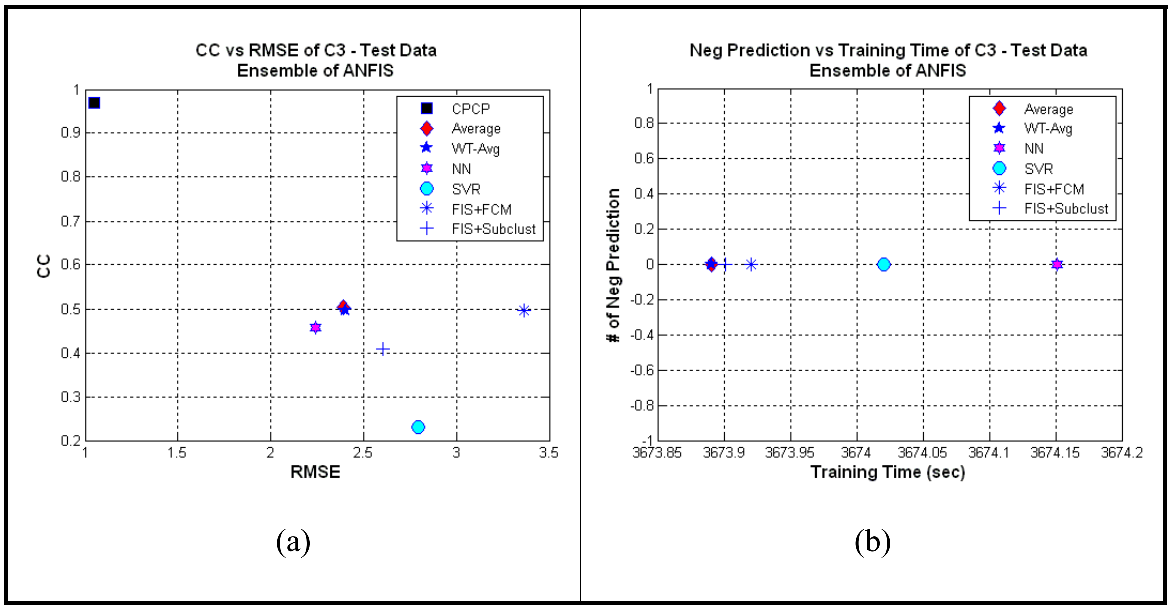


Figure 80: Performance of Ensemble of ANFIS for C3 Prediction

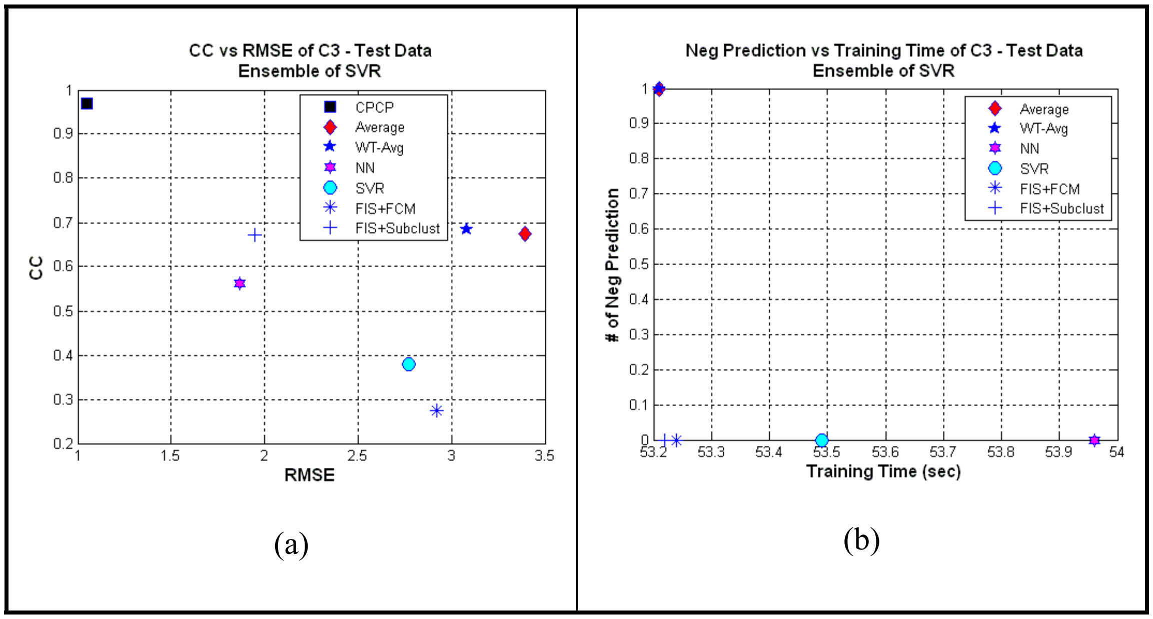


Figure 81: Performance of Ensemble of SVR for C3 Prediction

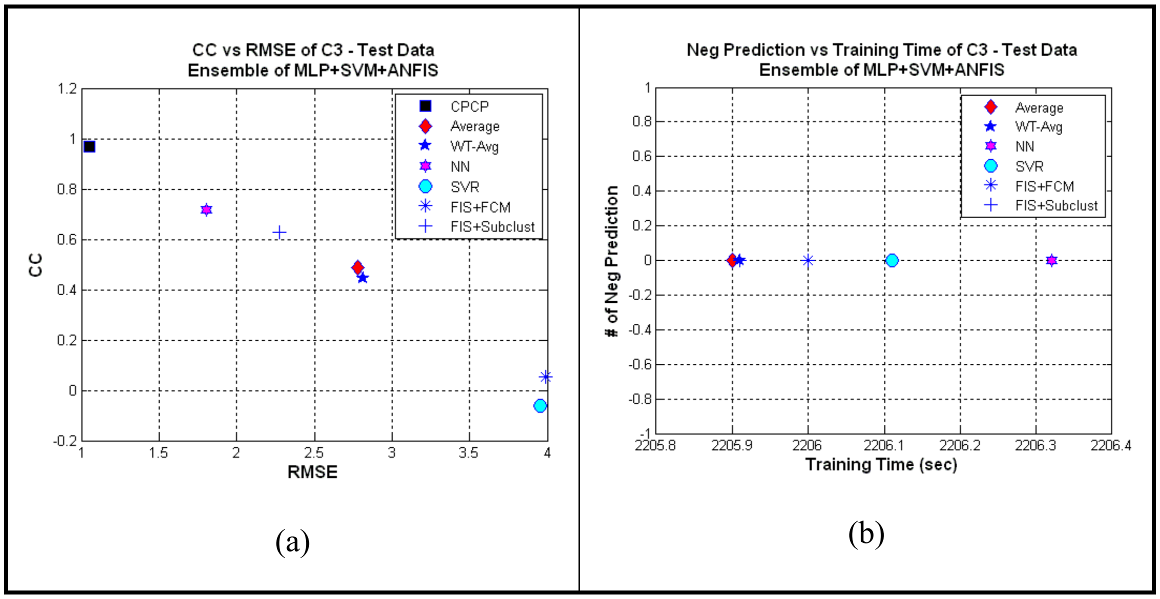


Figure 82: Performance of Heterogeneous Ensemble for C3 Prediction

Figure 78: (a) shows that the performance of HCI model GA+MLP outperforms the other CI, HCI models for C3 prediction. It is noticeable that the HCI models perform better than the corresponding CI models. Figure 78: (b) shows that GA+MLP took less time than GA+ANFIS and did not predict any negative value. The regression analysis of GA+SVR in Figure 78: (c, d) on training and testing data shows that the prediction is strongly correlated with the original values.

Figure 79: (a) shows that the EHCI model of MLP combined with FIS-Subclust method performed better than other combiner. The error RMSE value of the best model GA+MLP in Figure 78: (a) is near 1 whereas the EHCI model of MLP has RMSE about 0.7.

Figures 80, 81 and 82 show the other ECHI models' performance on predicting CO_2 in separators gas compositions.

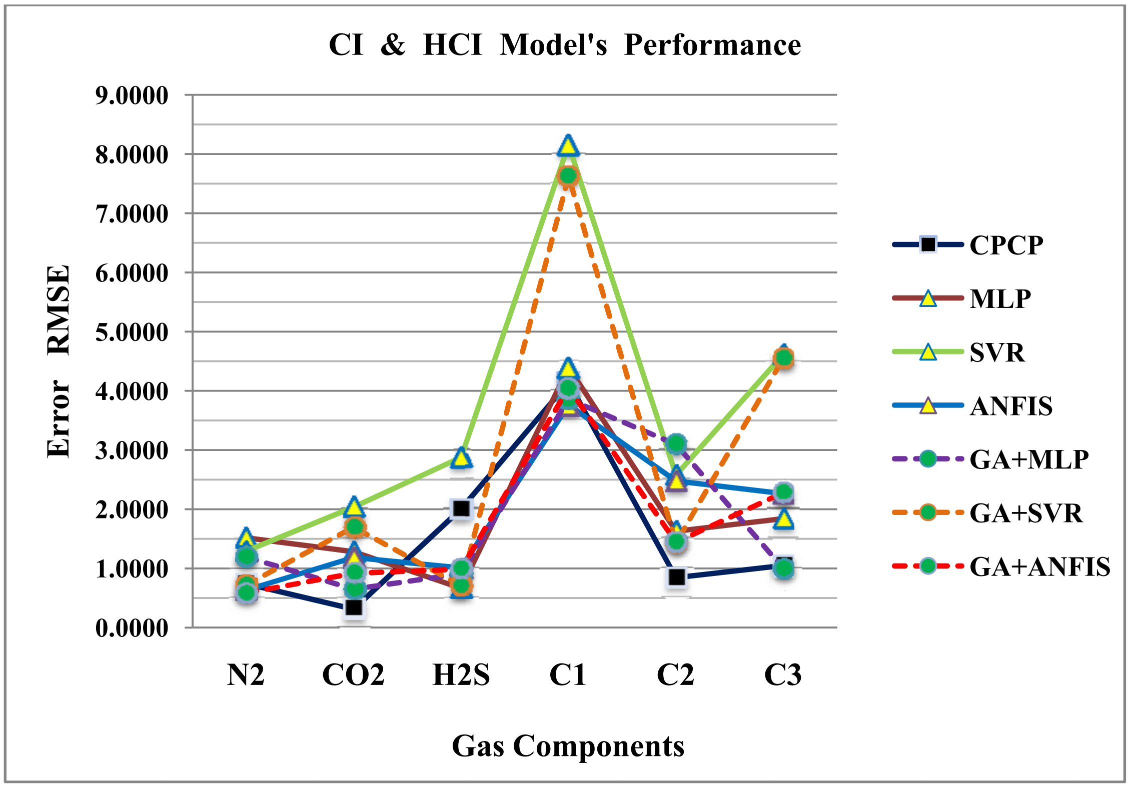


Figure 83: Performance of CI and HCI Models

Figure 83 shows the performance of gas components prediction in terms of RMSE. The black square (■) represents the performance of CPCP. The triangular symbol represents CI models performance and the circular symbols represent the performance of HCI models. It is expected that the circular symbols should appear lower part of the graph. In the case e.g. H2S we see that the performance of CI model is better than HCI models. But in most of the cases the performance of HCI is better than CI models. Some CI model may perform better than other HCI models but it is expected that on average the HCI models accuracy would be higher than CI models.

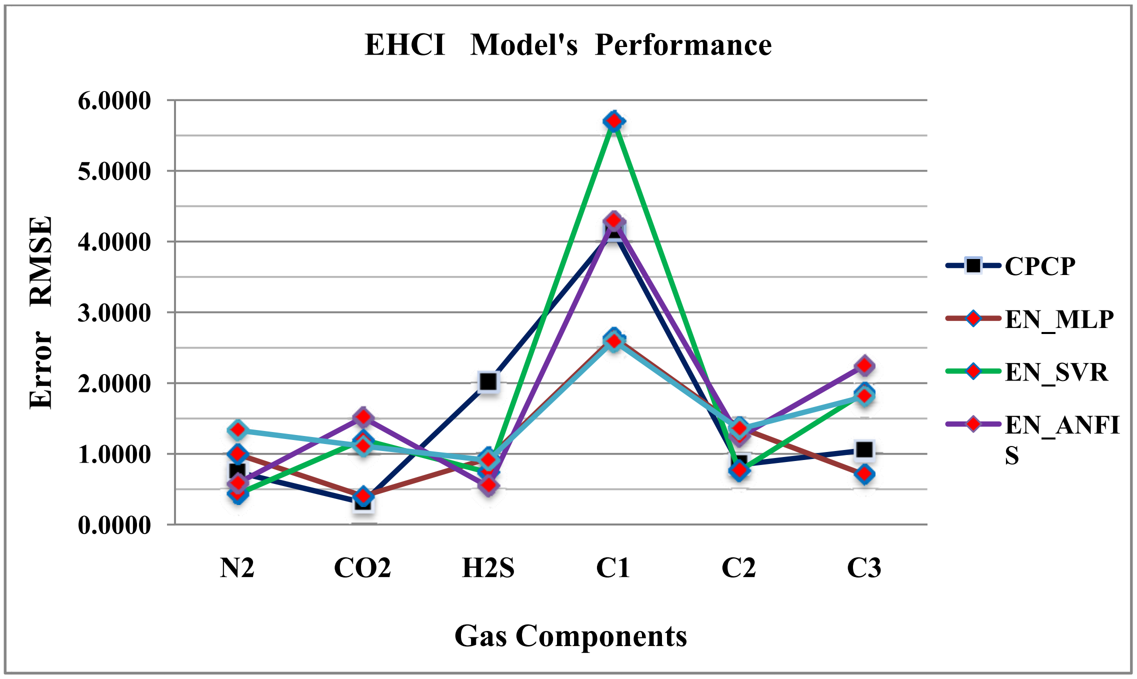


Figure 84: Performance of EHCI Models

The Figure 84 demonstrates the EHCI model's performance. By comparing with Figure 83 we can see that the EHCI models RMSE values are much lower than the CI and HCI model. In case of CO_2 the performance of CI and HCI (Figure 83) are far from the expected value but in case of EHCI the RMSE value is closer to CPCP. In case of C2 (Figure 83) the performance of CI and HCI could not outperform CPCP but EHCI model of SVR outperforms CPCP (Figure 84). For all other components the RMSE is much lower in the ECHI models.

Table 8: Performance of CI Models on Training Data

Training	CP		MLP			SVR			ANFIS		
Component	**CC**	**RMSE**	**CC**	**P-value**	**RMSE**	**CC**	**P-value**	**RMSE**	**CC**	**P-value**	**RMSE**
N2	0.8008	1.1383	0.9233	0.0000	0.8834	0.8142	0.0000	1.2879	0.9426	0.0000	0.5110
CO2	0.9978	0.2926	0.9910	0.0000	0.6219	0.9865	0.0000	2.3149	0.9976	0.0000	0.2927
H2S	0.9947	0.5334	0.9908	0.0000	0.5512	0.9799	0.0000	2.9719	0.9944	0.0000	0.3241
C1	0.9611	6.7795	0.9613	0.0000	6.3902	0.8810	0.0000	12.0963	0.9805	0.0000	4.4701
C2	0.8063	3.9949	0.9190	0.0000	2.5136	0.8782	0.0000	3.5870	0.9236	0.0000	2.3067
C3	0.9480	3.8902	0.9641	0.0000	3.0871	0.8832	0.0000	6.0333	0.9895	0.0000	1.5100

Table 9: Performance of CI Models on Test Data

Testing	CP		MLP			SVR			ANFIS		
Component	**CC**	**RMSE**	**CC**	**P-value**	**RMSE**	**CC**	**P-value**	**RMSE**	**CC**	**P-value**	**RMSE**
N2	0.9586	0.7402	0.9403	0.0001	1.5183	0.9843	0.0000	1.2947	0.9764	0.0000	0.6331
CO2	0.9989	0.3114	0.9669	0.0000	1.2819	0.9419	0.0000	2.0379	0.9693	0.0000	1.1854
H2S	0.6800	2.0037	0.9016	0.0004	0.6673	0.9149	0.0002	2.8769	0.7828	0.0074	1.0078
C1	0.9592	4.1464	0.9073	0.0003	4.3953	0.7058	0.0226	8.1564	0.9449	0.0000	3.7612
C2	0.9719	0.8453	0.8166	0.0039	1.6302	0.8143	0.0041	2.5780	0.5389	0.1080	2.4759
C3	0.9684	1.0511	0.6426	0.0451	1.8437	0.6319	0.0500	4.6158	0.4720	0.1684	2.2636

Table 10: Performance of HCI Models on Training Data

Training	CP		GA+MLP			GA+SVR			GA+ANFIS		
Component	**CC**	**RMSE**	**CC**	**P-value**	**RMSE**	**CC**	**P-value**	**RMSE**	**CC**	**P-value**	**RMSE**
N2	0.8008	1.1383	0.9239	0.0000	0.6173	0.7954	0.0000	0.9110	0.8965	0.0000	0.6655
CO2	0.9978	0.2926	0.9932	0.0000	0.5030	0.9890	0.0000	0.6285	0.9991	0.0000	0.1845
H2S	0.9947	0.5334	0.9786	0.0000	0.7012	0.9963	0.0000	0.2969	0.9990	0.0000	0.1388
C1	0.9611	6.7795	0.9737	0.0000	5.3129	0.9716	0.0000	5.5233	0.9847	0.0000	3.9868
C2	0.8063	3.9949	0.9496	0.0000	1.8938	0.8821	0.0000	2.8458	0.9694	0.0000	1.4813
C3	0.9480	3.8902	0.9764	0.0000	2.3174	0.8912	0.0000	4.7318	0.9926	0.0000	1.2935

Table 11: Performance of HCI Models on Test Data

Testing	CP		GA+MLP			GA+SVR			GA+ANFIS		
Component	**CC**	**RMSE**	**CC**	**P-value**	**RMSE**	**CC**	**P-value**	**RMSE**	**CC**	**P-value**	**RMSE**
N2	0.9586	0.7402	0.8953	0.0005	1.1926	0.9767	0.0000	0.7183	0.9795	0.0000	0.5851
CO2	0.9989	0.3114	0.9920	0.0000	0.6492	0.9385	0.0001	1.6950	0.9827	0.0000	0.9162
H2S	0.6800	2.0037	0.8527	0.0017	0.9093	0.9119	0.0002	0.7088	0.8211	0.0036	0.9878
C1	0.9592	4.1464	0.9346	0.0001	3.8597	0.8195	0.0037	7.6243	0.9242	0.0001	4.0493
C2	0.9719	0.8453	0.7971	0.0058	3.0978	0.8779	0.0008	1.4300	0.8339	0.0027	1.4430
C3	0.9684	1.0511	0.8950	0.0005	0.9948	0.6227	0.0545	4.5514	0.5898	0.0727	2.2791

Table 12: Performance of EHCI Models of Heterogeneous and MLP on Training Data

Training	CP		EN_of_MLP+SVR+ANFIS				EN_of_MLP			
Component	**CC**	**RMSE**	**CC**	**P-value**	**RMSE**	**Combiner**	**CC**	**P-value**	**RMSE**	**Combiner**
N2	0.8008	1.1383	0.8404	0.0000	0.8733	FCM	0.8458	0.0000	0.8740	FCM
CO2	0.9978	0.2926	0.9946	0.0000	0.4420	Avg	0.9734	0.0000	1.0653	SVR
H2S	0.9947	0.5334	0.9970	0.0000	0.2342	Avg	0.9983	0.0000	0.1756	Subclust
C1	0.9611	6.7795	0.9940	0.0000	2.4706	Subclust	0.9850	0.0000	3.8961	WT_Avg
C2	0.8063	3.9949	0.9849	0.0000	1.0394	Subclust	0.9476	0.0000	1.9194	Subclust
C3	0.9480	3.8902	0.9574	0.0000	3.0316	NN	0.9854	0.0000	1.7741	Subclust

Table 13: Performance of EHCI Models of Heterogeneous and MLP on Test Data

Testing	CP		EN_of_MLP+SVR+ANFIS				EN_of_MLP			
Component	**CC**	**RMSE**	**CC**	**P-value**	**RMSE**	**Combiner**	**CC**	**P-value**	**RMSE**	**Combiner**
N2	0.9586	0.7402	0.9445	0.0000	1.3335	FCM	0.9354	0.0001	0.9959	FCM
CO2	0.9989	0.3114	0.9760	0.0000	1.1102	Avg	0.9921	0.0000	0.6455	SVR
H2S	0.6800	2.0037	0.8635	0.0013	0.8243	Avg	0.8254	0.0033	0.9471	Subclust
C1	0.9592	4.1464	0.9724	0.0000	2.5963	Subclust	0.9734	0.0000	2.6864	WT_Avg
C2	0.9719	0.8453	0.7901	0.0065	1.8677	Subclust	0.9579	0.0000	0.9375	Subclust
C3	0.9684	1.0511	0.7165	0.0197	1.8074	NN	0.9626	0.0000	0.7051	Subclust

Table 14: Performance of EHCI Models of SVR and ANFIS on Training Data

Training	CP		EN_of_SVR				EN_of_ANFIS			
Component	**CC**	**RMSE**	**CC**	**P-value**	**RMSE**	**Combiner**	**CC**	**P-value**	**RMSE**	**Combiner**
N2	0.8008	1.1383	0.9152	0.0000	0.5971	WT_Avg	0.8604	0.0000	0.2509	Subclust
CO2	0.9978	0.2926	0.9951	0.0000	0.4142	Subclust	0.9694	0.0000	1.1126	SVR
H2S	0.9947	0.5334	0.9975	0.0000	0.2155	WT_Avg	0.9984	0.0000	0.1904	Avg
C1	0.9611	6.7795	0.9655	0.0000	6.8267	FCM	0.9975	0.0000	1.5971	Subclust
C2	0.8063	3.9949	0.9216	0.0000	2.3371	WT_Avg	0.9790	0.0000	1.2252	Subclust
C3	0.9480	3.8902	0.9835	0.0000	1.8862	Subclust	0.8759	0.0000	5.1539	NN

Table 15: Performance of EHCI Models of SVR and ANFIS on Test Data

Testing	CP		EN_of_SVR				EN_of_ANFIS			
Component	**CC**	**RMSE**	**CC**	**P-value**	**RMSE**	**Combiner**	**CC**	**P-value**	**RMSE**	**Combiner**
N2	0.9586	0.7402	0.9920	0.0000	0.5339	WT_Avg	0.9735	0.0000	0.5858	Subclust
CO2	0.9989	0.3114	0.9539	0.0000	1.4027	Subclust	0.9253	0.0001	1.7695	SVR
H2S	0.6800	2.0037	0.8884	0.0006	0.8431	WT_Avg	0.9315	0.0001	0.6100	Avg
C1	0.9592	4.1464	0.9005	0.0004	5.7066	FCM	0.9113	0.0002	4.2918	Subclust
C2	0.9719	0.8453	0.9519	0.0000	0.7818	WT_Avg	0.8604	0.0014	1.2852	Subclust
C3	0.9684	1.0511	0.8394	0.0024	1.3132	Subclust	0.4582	0.1830	2.2419	NN

The Tables 8 to 15 show the numerical values of the CC, RMSE and P-values of all the models. The CC value represents how good the prediction is and the P-value shows how significant the prediction is. The CC above 0.75 represents statistically acceptable correlation and the P-value less than or equal to 0.05 means the significance level is within 5%. In the Tables 8 to 15 we can see that the P-value is less than 0.05 except one or two cases which shows that the prediction of models are significant.

CHAPTER 7

CONCLUSION AND FUTURE WORK

The outcomes indicate that the performances of EHCI models are anticipating. Various types of EHCI models are equipped with different gas components. The performance of CPCP is ameliorating only for CO_2. Nevertheless it should be noted that if the fraction of gas components is low comparing to other gas components then a relatively small difference in prediction would cause a higher error in calculation. Although CPCP performed well, the results obtained by CPCP are fixed. On the other contrary, the performances of HCI or EHCI models are still have options to be optimized. Fine tuning to GA operator and other parameters of the models can improve the HCI or EHCI model's performance.

Different EHCI models perform well for different gas components. It cannot be extrapolated to use particular types of EHCI model for all the gas components. Furthermore the combining techniques are also important and its performance for different gas components varies. We have used EHCI model consisting of only 3 members. In general there is no ensemble method which surpasses other ensemble methods consistently. It is anticipated that using more base learners will lead to a better performance, yet Zhou et al. proved the "many could be better than all" theorem which points that this may not be the fact. Though ensemble having more members might have better accuracy, we have got better results than CPCP benchmark by using EHCI model having only three members. So we didn’t include more members in ensemble so as to obtain the simple EHCI model. We can further improve the EHCI model by incorporating new HCI members of EHCI models. Moreover new HCI can be included as a member such as different types of ANN, Type II Fuzzy Logic, GHDH based model, Extreme Learning Machine (ELM) etc. To make the HCI member diverse, enough data can be divided intelligently by flocking the training sets or the dominant input parameters. The parameters of GA can be finely tuned so that the accuracy of EHCI models can be improved to greater degree. Moreover EHCI members can be chosen from a group of well diverse and accurate HCI models. We do not perform post processing so that very small value counted as negative. Post processing of the output can improve the performance by eliminating negative predicted value. Furthermore, there are still 6 more gas components left to be predicted.

REFERENCES

[1] Włodzisław Duch, "What is Computational Intelligence and what could it become? Department of Informatics," Nicolaus Copernicus University, Grudzia¸dzka 5, Toru´n, Poland, and School of Computer Engineering, Nanyang Technological University,Singapore.

[2] G. Pampara, Andries P. Engelbrecht, T. Cloete, "CIlib: A Component-based Framework for Plug-and-Simulate Hybrid Computational Intelligence Systems," in *International Joint Conference on Neural Networks (IJCNN)*, 2008.

[3] Istvan Juhos, Laszlo Makra, Balazs Toth, "Forecasting of traffic origin NO and NO2 concentrations by Support Vector Machines and neural networks using Principal Component Analysis," *Simulation Modelling Practice and Theory*, vol. 16, pp. 1488–1502, 2008.

[4] Mahmut Bayramoglu, Yilmaz Yildirim, "Adaptive Neuro-Fuzzy based Modelling for Prediction of Air Pollution Daily Levels in City of Zonguldak," *Chemosphere*, vol. 63, pp. 1575–1582, 2006.

[5] Alex J. Smola, Bernhard Scholkopf, "A Tutorial on Support Vector Regression," NeuroCOLT Technical Report, TR-98-030, September 30, 2003.

[6] "The European Network on Intelligent Technologies for Smart Adaptive Systems: EUNITE," , Aachen, Germany, pp. 16-17, March 2001.

[7] Chih-Hung Wu,Gwo-Hshiung Tzeng, Rong-Ho Lin, "A Novel Hybrid Genetic Algorithm for Kernel Function and Parameter Optimization in Support Vector Regression," *Expert Systems with Applications*, vol. 36, pp. 4725–4735, 2009.

[8] Robi Polikar, "Bootstrap-Inspired Techniques in Computational Intelligence," in *IEEE SIGNAL PROCESSING MAGAZINE*, 2007.

[9] R. Harris, X. Yao, G. Brown, J. L. Wyatt, "Diversity Creation Methods: A Survey and Categorization," *Journal of Information Fusion*, vol. 6, no. 1, pp. 5-20, March 2005.

[10] Hansen L.K., Salamon P., "Neural Network Ensembles," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, no. 10, pp. 993–1001, 1990.

[11] R. E. Schapire, "The Boosting Approach to Machine Learning: An Overview," *Lecture Notes in Statistics*, pp. 149-172, 2003.

[12] Atalla F. Sayda, James H. Taylor, "Modeling and Control of Three-Phase Gravity Separators in Oil Production Facilities," in *Department of Electrical & Computer Engineering, University of New Brunswick*, Canada.

[13] S.G. Alborzi, J. Hargreaves, "Design optimizes sour-crude production facility," *Oil & Gas Journal*, Aug 2010.

[14] H. Sundgren, F. Wlnqulst, I. Lukkarl, I. Lundstrom, "Artificial Neural Networks and Gas Sensor Arrays: Quantification of Individual Components in a Gas Mixture," *Meas. Sci. Technol.*, vol. 2, pp. 464-469, 1991.

[15] James C. Howland, Mark S. Voss, "Natural Gas Prediction Using The Group Method of Data Handling," in *Computer Information Systems*, College of Technical Sciences, Montana State University, Montana, USA.

[16] Corpoven, S.A M.F. Briones and E.R. Martinez, U. de Oriente G.A. Rojas, and U. Central de Venezuela J.A. Moreno, "Application of Neural Networks in the Prediction of Reservoir Hydrocarbon Mixture Composition From Production Data," vol. SPE 28598.

[17] C. Tas¸altın, M.A. Ebeoglu, F. Tekce, A. Ozmen, Z.Z. O¨ ztu¨rk, "Finding the Composition of Gas Mixtures by a Phthalocyanine-coated QCM Sensor Array and an Artificial Neural Network," *Sensors and Actuators*, vol. B 115, pp. 450-454, 2006.

[18] J.A.Almarry, F.T.Al-Saadoon, "Prediction of Liquid Hydrocarbon Recovery from a Gas Condensate Reservoir," *SPE 13715*, 1985.

[19] Adel M. Elsharkawy, Salah G. Foda, "EOS Simulation and GRNN Modeling of the Constant Volume Depletion Behavior of Gas Condensate Reservoirs," *Energy & Fuels*, vol. 12, no. 2, pp. 353-364, 1998.

[20] Farhan A. Al-Farhan, Luis F. Ayala H, "Optimization of surface condensate production from natural gases using artificial intelligence," vol. 53, no. 135–147, 2006.

[21] Balan B., Ameri S., Mohaghegh S., "A Hybrid Neuro-Genetic Approach to Hydraulic Fracture Treatment Design and Optimization," *SPE 36602*, 1996.

[22] Sheng-wei Fei, Cheng-liang Liu, Yu-bin Miao, "Support Vector Machine with Genetic Algorithm for Forecasting of Key-gas Ratios in Oil Immeersed Transformer," *Expert Systems with Applications*, vol. 36, pp. 6326–6331, 2009.

[23] Fred Aminzadeh, "Applications of AI And Soft Computing for Challenging Problems in The Oil Industry," *Journal of Petroleum Science and Engineering*, vol. 47, pp. 5-14, 2005.

[24] Eissa M. El-M. Shokir, Hazim N. Dmour, "Genetic Programming (GP)-Based Model for the Viscosity of Pure and Hydrocarbon Gas Mixtures," *Energy & Fuels*, vol. 23, no. 7, pp. 3632-3636, June 19, 2009.

[25] Amar Khoukhi, Munirudine Oloso, Mostafa Elshafei, Abdulazeez Abdulraheem, "Support Vector Regression And Functional Networks For Viscosity And Gas/Oil Ratio Curves Estimation," in *SPE/EAGE Reservoir Characterization and Simulation*

Conference, Abu Dhabi, UAE, Oct 19-21, 2009.

[26] Tarek Helmy, Anifowose Fatai, "Hybrid Computational Intelligence Models for Porosity and Permeability Prediction of Petroleum Reservoirs," *International Journal of Computational Intelligence and Applications*, vol. 9, no. 4, pp. 313-337, 2010.

[27] Tarek Helmy, Anifowose Fatai, Kanaan Faisal, "Hybrid Computational Models for the Characterization of Oil and Gas Reservoirs," *Elsevier Journal of Expert Systems with Applications*, vol. 37, pp. 5353-5363, July, 2010.

[28] Fi-John Chang, Ya-Ting Chang, "Adaptive neuro-fuzzy inference system for prediction of water level in reservoir," *Advances in Water Resources*, vol. 29, pp. 1-10, 2006.

[29] S. Mohaghegh, S. Ameri, K. Aminian, "Predicting the Production Performance of Gas Reservoirs," *Scientia Iranica*, vol. 1, no. 3, 1994.

[30] N. Ueda, R. Nakano, "Generalization Error of Ensemble Estimators," in *In Proceedings of International Conference on Neural Networks*, pp. 90-95, 1996.

[31] Huanhuan Chen, "Diversity and Regularization in Neural Network Ensembles," School of Computer Science University of Birmingham, PhD Thesis October, 2008.

[32] Jean-Noel Jaubert, Laurent Avaullee, Jean-Francois Souvay, "A crude oil data bank containing more than 5000 PVT and gas injection data," *Journal of Petroleum Science and Engineering*, vol. 34, pp. 65-107, 2002.

[33] Saifur Rahman, "Experimental K-value Prediction", SPE/EAGE Reservoir Characterization and Simulation Conference," in *SPE 125413*, Abu Dhabi, UAE, Oct, 2009.

[34] M., Hernández Espinoza, M. Fernández Redondo, "Optimal use of trained neural network for input selection. International Work-Conference on Artificial and Natural Neural Networks ," in *IWANN'99*, Alicante, Spain, 1999, pp. 506–515.

[35] A. R. Moghadassi, F. Parvizian, S. M. Hosseini, A. R. Fazlali, "A New Approach for Estimation of PVT Properties of Pure Gases based on Artificial Neural Network Model," *Brazilian Journal of Chemical Engineering*, vol. 26, no. 01, pp. 199-206, March 2009.

[36] Deyi Xie, Dave Wilkinson, Tina Yu, San Ramon, "Permeability Estimation Using a Hybrid Genetic Programming and Fuzzy/Neural Inference Approach," in *SPE Annual Technical Conference*, Dallas, Texas, U.S.A., SPE 95167, October, 2005.

[37] Wafaa El-Shahat Afify, Alaa H. Ibrahim Hassan, "Permeability and Porosity Prediction from Wireline logs Using Neuro-Fuzzy Technique," *Ozean Journal of Applied Sciences*, vol. 3, no. 1, 2010.

[38] Ali Kadkhodaie-Ilkhchi, M. Reza Rezaee, Hossain Rahimpour-Bonab, "A committee neural network for prediction of normalized oil content from well log data: An example from South Pars Gas Field," *Journal of Petroleum Science and Engineering*, vol. 65, pp. 23-32, 2009.

[39] Ying Zhao, Jun Gao, Xuezhi Yang, "A Survey of Neural Network Ensembles," in *International Conference on Neural Networks and Brain*, October, 2005.

[40] Haibo He, Xiaoping Shen, "Bootstrap Methods for Foreign Currency Exchange Rates Prediction," in *Proceedings of International Joint Conference on Neural Networks*, Orlando, Florida, USA, pp. 12-17, August, 2007.

[41] Kin Keung Lai, Lean Yu, Shouyang Wang, Huang Wei, "A Novel Nonlinear Neural Network Ensemble Model for Financial Time Series Forecasting," *ICCS, Part I, LNCS 3991*, pp. 790-793, 2006.

[42] Yan-Shi Dong, Ke-Song Han, "Boosting SVM Classifiers by Ensemble," in *WWW*, Chiba, Japan, May 10-14, 2005.

[43] Prem Melville, Nishit Shah, Lilyana Mihalkova, Raymond J. Mooney, "Experiments on Ensembles with Missing and Noisy Data," *Proceedings of 5th International Workshop on Multiple Classifier Systems (MCS-2004), LNCS*, vol. 3077, pp. 293-302, 2004.

[44] N. Ancona, P. Blonda, C. Tarantino, G. Satalino, A. D'Addabbo G. Pasquariello, "Neural Network Ensemble and Support Vector Machine Classifiers for the Analysis of Remotely Sensed Data: a Comparison," in *CNR - I.E.S.I.,IEEE*, Via Amendola 166/5, 70126 Bari (Italy), 2002.

[45] Joaquin Torres-Sospedra, Carlos Hernandez-Espinosa Mercedes Fernandez-Redondo, "The Mixture of Neural Networks as Ensemble Combiner," *ANNPR, LNAI*, vol. 5064, pp. 168-179, 2008.

[46] Jian-Xin Wu, Yuan Jiang, Shi-Fu Chen, Zhi-Hua Zhou, "Genetic Algorithm based Selective Neural Network Ensemble," *Proceedings of the 17th International Joint Conference on Artificial Intelligence*, vol. 2, pp. 797-802, 2001.

[47] Song Yixu,Wen Qinghua, Yang Zehong, "Hybrid approaches for stock price prediction," in *ICCSA*, 2009.

[48] Mohammed E. El-Telbany, Sultan H. Aljahdali, "Genetic Algorithms for Optimizing Ensemble of Models in Software Reliability Prediction," *ICGST-AIML*, vol. 8, no. I, June 2008.

[49] Anazida Zainal, Mohd Aizaini Maarof, Siti Mariyam Shamsuddin, Ajith Abraham, "Ensemble of One-class Classifiers for Network Intrusion Detection System," Faculty of Computer Science and Information System, University Technology Malaysia, Skudai, Johor.

[50] Xin Yao, Arjun Chandra, "Evolving hybrid ensembles of learning machines for better generalization," *Neurocomputing*, vol. 69, pp. 686-700, 2006.

[51] Bernhard E. Boser, Isabelle Guyon, Vladimir Vapnik, "A Training Algorithm for Optimal Margin Classifiers," in *Fifth Annual ACM Conference on Computational Learning Theory (COLT)*, Pittsburgh, PA, USA, pp. 144-152, 1992.

[52] Vladimir Vapnik, Corinna Cortes, "Support-Vector Networks," *Machine Learning*, vol. 20, no. 3, pp. 273-297, Sept 1995.

[53] V. Vapnik, *The Nature of Statistical Learning Theory*. New York: Springer-Verlag, 1995.

[54] Stuart J. Russell, Peter Norvig, *Artificial Intelligence : A Modern Approach*, 2nd ed.: Prentice Hall/Pearson Education, 2003.

[55] Fuller A.D., *Neural Fuzzy Systems*. Abo, 1995.

[56] J.-S.R. Jang, "ANFIS: adaptive-network-based fuzzy inference systems," *IEEE Transactions on Systems*, vol. 23, no. 3, pp. 665-685, 1993.

[57] Tagaki T., Sugeno M., "Fuzzy Identification of Systems and its Application to Modelling and Control," *IEEE Transactions on Systems*, vol. Men and Cybernetics, no. 15, pp. 116-132, 1985.

[58] L. S. Admuthe, Dr. S.D. Apte, "Computational Model using ANFIS And GA : Application for Textile Spinning Process," in *International Conference on Computer Science and Information Technology, ICCSIT*, Beijing, Chaina, 8-11 Aug, 2009.

[59] Clarence De Silva, Fakhreddine O. Karray, *Soft Computing and Intelligent Systems Designe*, 1st ed.: Addison Wesley, 2004.

[60] Cheng-Lung Huang, Chieh-Jen Wang, "A GA-based feature selection and parameters optimization for support vector machines," *Expert System with Application*, vol. 3, pp. 231-240, 2006.

[61] S. Chiu, "Fuzzy Model Identification based on Cluster Estimation," *Journal of Intelligent & Fuzzy Systems*, vol. 2, no. 3, pp. 267-278, 1994.

[62] T. G. Dietterich, "An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization," *Machine Learning*, vol. 40, no. 2, pp. 139-158, 2000.

[63] A. Krogh, J. Vedelsby, "Neural Network Ensembles," in *Advances in Neural Information Processing Systems (NIPS 7)*, pp. 231-238, 1995.

[64] Gavin Brown, *Ensemble Learning, Encyclopedia of Machine Learning*, C.Sammut & G.I.Webb (Eds.), Ed.: Springer Press , 2010.

[65] Ethem Alpaydin, "Introduction to Machine Learning," *Lecture Notes, The MIT Press*, vol. 1.1, 2004.

[66] McCabe G. P., Moore D. S., *Introduction to the Practice of Statistics.*: Michelle Julet, 2002.

[67] S. Geman, E. Bienenstock, R. Doursat, "Neural networks and the bias/variance dilemma," *Neural computation*, vol. 4, no. 1, pp. 1-58, 1992.

[68] L. Breiman, "Bagging predictors," *Machine Learning*, vol. 24, no. 2, pp. 123–140, 1996.

[69] R. Schapire, Y. Freund, "Experiments with a New Boosting Algorithm," in *In International Conference on Machine Learning*, 1996.

[70] J. Friedman, T. Hastie, R. Tibshirani, "Additive Logistic Regression: A Statistical View of Boosting," *The Annals of Statistics*, vol. 28, no. 2, pp. 337–374, April 2000.

[71] D. Mease, A. Wyner, "Evidence Contrary to the Statistical View of Boosting," *Journal of Machine Learning Research*, vol. 9, pp. 131-156, 2008.

[72] D., Partridge, "Network Generalization Differences Quantified," *Neural Networks*, vol. 9, pp. 263-271.

[73] Opitz D.W., Shavlik J.W., "Actively searching for an effective neural network ensemble," *Connection Science*, vol. 8, pp. 337-353, 1996.

[74] A Lazarevic, Z. Obradovic, "Effective Pruning of Neural Network Classifier Ensembles," *International Joint Conference on Neural Networks*, vol. 2, pp. 796-801, 2001.

[75] L. N. Copper, M. P. Perrone, "When Networks Disagree: Ensemble Method for Neural Networks," *Artificial Neural Networks for Speech and Vision*, pp. 126-142, 1993.

[76] P. Sollich, A. Krogh, "Learning with ensembles: How over-fitting can be useful," *Advances in Neural Information Processing System*, vol. 8, pp. 190-198, 1996.

APPENDIX A: A RESERVOIR SAMPLE WITH SEPARATOR COMPOSITION

Table 16: Molar compositions related to reservoir of Fluid F3 [JAUBERT]

		First stage conditions: Tsep/ °C = 89.0 Psep/bar = 34.0		Second stage conditions: Tstock/ °C = 15.0 Pstock/bar = 1.01325 tank oil density (kg/m3) = 828.4		Properties of the cuts from C6 to C20+	
Compound	Reservoir fluid (F3)	Separator gas	Separator liquid	Residual gas	Stock tank oil	Molar weight $(g \cdot mol^{-1})$	density at 15 °C (kg/m3)
1. Hydrogen Sulfide	0.000	0.000	0.000	0.000	0.000		
2. Nitrogen	0.450	0.780	0.040	0.190	0.000		
3. Carbon dioxide	1.640	2.510	0.470	2.420	0.000		
4. Methane	45.850	73.180	8.280	42.510	0.000		
5. Ethane	7.150	9.870	3.740	16.340	0.690		
6. Propane	6.740	8.190	6.680	22.470	2.860		
Cut C4							
7. i-Butane	0.840	0.840	1.270	2.640	0.940		
8. n-Butane	3.110	2.620	5.250	8.220	4.530		
Cut C5							
9. i-Pentanes	1.030	0.560	2.070	1.570	2.190		
10. n-Pentane	1.650	0.750	3.440	1.930	3.810		
Cut C6							
11. i-Hexanes	1.280	0.200	2.540	0.510	3.030	86.0	672.8
12. n-Hexane	1.240	0.190	2.460	0.490	2.940		
Cut C7							
13. i-Heptanes	0.470	0.030	1.030	0.070	1.270	92.0	729.4
14. Benzene	0.240	0.020	0.520	0.030	0.640		
15. Cyclanes C7	2.230	0.150	4.950	0.320	6.060		
16. n-heptane	0.830	0.050	1.830	0.120	2.240		
Cut C8							
17. i-Octanes	0.720	0.010	1.370	0.030	1.700	106.0	750.9
18. Toluene	0.740	0.010	1.430	0.030	1.760		
19. Cyclanes C8	2.020	0.040	3.890	0.080	4.810		
20. n-Octane	0.800	0.010	1.530	0.030	1.900		
Cut C9							
21. i-Nonanes	0.620	0.000	1.390	0.000	1.720	120.0	773.9
22. Aromatics C9	0.920	0.000	2.060	0.000	2.550		
23. Cyclanes C9	0.640	0.000	1.420	0.000	1.770		
24. n-Nonane	0.520	0.000	1.150	0.000	1.430		
Cut C10							
25. i-Decanes	1.020	0.000	2.010	0.000	2.500	137.0	783.5
26. Aromatics C10	0.360	0.000	0.700	0.000	0.870		
27. n-Decane	0.310	0.000	0.620	0.000	0.770		
28. undecanes (cut C11)	1.810	0.000	4.130	0.000	5.120	146.0	796.8
29. dodecanes (cut C12)	1.470	0.000	3.350	0.000	4.160	159.0	805.7
30. tridecanes (cut C13)	1.450	0.000	3.320	0.000	4.130	172.0	815.1
31. tetradecanes (cut C14)	1.280	0.000	2.920	0.000	3.630	183.0	827.2
32. pentadecanes (cut C15)	1.150	0.000	2.640	0.000	3.270	198.0	843.7
33. hexadecanes (cut C16)	0.910	0.000	2.070	0.000	2.570	218.0	845.8
34. heptadecanes (cut C17)	0.820	0.000	1.860	0.000	2.320	233.0	844.9
35. octadecanes (cut C18)	0.800	0.000	1.830	0.000	2.270	249.0	849.1
36. nonadecanes (cut C19)	0.710	0.000	1.620	0.000	2.010	262.0	858.9
37. eicosanes plus (C20+)	6.180	0.000	14.110	0.000	17.540	474.0	925.3

RESUME

Muhammad Imtiaz Hossain
MS in ICS, Research Assistant, CCSE Department.
King Fahd University of Petroleum & Minerals, Dhahran 31261, Saudi Arabia.
E-mail: g200803120@kfupm.edu.sa , imtiaz.kfupm@gmail.com
Mobile: +966531828624
KFUPM Box. 1086, Dhahran – 31261, KSA.

B.S. SENIOR PROJECT

"Expert Student Advising System" - implemented using JAVA, JDBC, MS SQL Server 2003 and ASP.NET using C#.

PUBLICATIONS

1. **Muhammad Imtiaz Hossain**, Mohammad Tanvir Parvez, "*Offline Arabic Handwritten Digit Recognition using Neural Network*", Proceedings of the 1st Saudi Students Conference, Riyad, 2010.
2. **Muhammad I. Hossain**, Tarek Helmy, Amar Khoukhe, Abdulazeez Abdulrahim, "*Gas Compositions Prediction In Primary Separator Of Three Stage Separators Using Artificial Intelligence*", Proceedings of the 2nd Saudi Students Conference, Jedda, 2011**.**
3. Tarek Helmy, Sayed M. Rahman, **Muhammad Imtiaz Hossain**, Abdul Azeez A. R., "*Non-linear Heterogonous Ensemble Model for Permeability Prediction of Oil and Gas Reservoirs*", Submitted to the ISI Arabian Journal for Science and Engineering, January, 2011.
4. **M. Imtiaz Hossain**, Tarek Helmy, Md Rafiul Hassan, A. Abdulrahim, M. Elshafei, "*Non Hydrocarbons Prediction in Multistage Separator using Neural network*", Proceedings of the 5th Global Conference on Power Control and Optimization, Duabi, pp. to appear, 2011.

ONGOING RESEARCH WORK

1. Evolutionary Optimized Hybrid Computational Intelligence Models for Gas Components Prediction. (**MS Thesis Research)**
2. Hidden Markov Model based imbalanced data sampling for classification.
3. Classification of Bioinformatics Datasets using Machine Learning Techniques.
4. Transition Initiation Sites (TIS) Prediction in cDNA Sequences using Machine Learning.

5. Handwritten Arabic and Bengali Character and Digit Recognition using Artificial Intelligence.
6. Temperature Prediction using Artificial Intelligence.
7. Permeability, Porosity, Flow Zone Index Prediction of oil/gas reservoir.

AI TOOLS EXPERIENCE

1. **Matlab Toolbox** – Neural Network, Fuzzy Logic, ANFIS, Genetic Algorithm, SVM.

2. **Other Toolbox** – DTREG, WEKA, Rapid Miner, GMDH based Abductive Network.