ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL OF SCIENCE ENGINEERING AND TECHNOLOGY

IMPROVING THE PERFORMANCE OF SELF-ORGANIZING MAP

M.Sc. THESIS

Faranak NOURI

Department of Electronics and Communication Engineering

Electronics Engineering Programme

Thesis Advisor: Assoc. Prof. Dr. Neslihan Serap ŞENGÖR

December 2012

ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL OF SCIENCE ENGINEERING AND TECHNOLOGY

IMPROVING THE PERFORMANCE OF SELF-ORGANIZING MAP

M.Sc. THESIS

Faranak NOURI
504091242

Department of Electronics and Communication Engineering

Electronics Engineering Programme

Thesis Advisor: Assoc. Prof. Dr. Neslihan Serap ŞENGÖR

December 2012

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

ÖZ-DÜZENLEMELİ AĞ YAPISININ PERFORMANSININ İYİLEŞTİRİLMESİ

YÜKSEK LİSANS TEZİ

Faranak NOURI
504091242

Elektronik ve Haberleşme Mühendisliği Anabilim Dalı

Elektronik Mühendisliği Programı

Tez Danışmanı: Doç. Dr. Neslihan Serap ŞENGÖR

Aralık 2012

Faranak-Nouri, a **M.Sc.** student of ITU **Graduate School of Science Engineering and Technology** student ID 504091242, successfully defended the **thesis** entitled "**IMPROVING THE PERFORMANCE OF SELF-ORGANIZING MAP**", which she prepared after fulfilling the requirements specified in the associated legislations, before the jury whose signatures are below.

Thesis Advisor :	**Assoc. Prof. Dr. Neslihan S. ŞENGÖR** Istanbul Technical University	.....
Jury Members :	**Assoc. Prof. Dr. Şule Gündüz ÖĞÜDÜCÜ** Istanbul Technical University	.....
	Assist. Prof. Dr. İsa YILDIRIM Istanbul Technical University	.....

Date of Submission : 17 December 2012
Date of Defense : 24 January 2013

FOREWORD

This thesis was carried out in close collaboration with my supervisor Neslihan Serap Şengör over the last two years.I would like to thank her for introducing me the field,giving advices,sharing her opinions and listening to me.It is certain that this thesis would not be possible without her generous support.

December 2012 Faranak NOURI

TABLE OF CONTENTS

Page

ABBREVIATIONS

SOM **:** Self Organazing Map
BMU **:** Best Matching Unit
RL **:** Reinforcement Learning
LVQ **:**Learning Vector Quantization

LIST OF TABLES

Page

LIST OF FIGURES

Page

IMPROVING THE PERFORMANCE OF SELF-ORGANIZING MAP

SUMMARY

Clustering data with self organizing maps is a widely used method. In this thesis, self-organizing map and learning vector quantization methods are considered for solving clustering problems. Firstly, self organizing map is modified to use class information like learning vector quantization method. Also, inspired by the hybrid methods proposed recently which combine reinforcement learning with self organizing map, a novel approach is proposed where a simple reinforcement learning algorithm is used both in self organizing map and self organizing map with class information. Self organizing map and all the proposed approaches are realized in MATLAB and applied to clustering set of two-dimensional points and Iris data set. The discussion on the simulation results are based on the results obtained for training and test sets and ideas are given to improve these results.

The main purpose of this thesis is to propose a new learning method and explain how to implement the new method, so the problems choosen to be easily manufactured and widely used for educational purposes. So clustering points on a two-dimensional surface and Iris data set are choosen as benchmark problems and in order to have statistically significant conclusion, the evaluation measures are used as sensitivity, specificity, accuracy, precision and similarity.

At the end S-SOM structure gives the best result amongst the three methods that have been suggested. However, due to the reward mechanism of RL-SOM structure, which chooses the winner based on the criteria of choosing the best fit makes it possible to distinguish similar data in different classes. Even though, the proposed methods provided some improvement in the data set of 2-dimensional points, the results for Iris data set were not satisfactory as expected. Using in a way more knowledge in tarining would help to imrove the performance of already exisiting methos. One way is have different reinforcement learning methods implemented in conventional methods.

ÖZ-DÜZENLEMELİ AĞ YAPISININ PERFORMANS İYİLEŞTİRİLMESİ

ÖZET

Öbekleme problemleri, verilerin benzerliğne dayanmaktadır bu problemlerin analizinde bu benzerlik veya mesafe hesaplamak zorunludur. Bu yüzdenveri çok büyük ve ya dağınık bir şekilde ise bu verilerin düzenlenmi oldukça zordur ve asıl sorun hesaplamaların çok uzun zaman almasıdır. Bu çalışmaöz-düzenlemeli ağ yapısının bu sorununu çozmek için önerilerde bulunmuştur. İlk olaraköz-düzenlemeli ağ ile vektör kuantalama yapıları birlikte ele alınmıştır ve özdüzenlemeli ağ yapısında, vektör kuantalama yapısında olduğu gibi sınıf bilgisi kullanılmıştır. Ayrıca, son senelerde pekiştirmeli öğrenme ile birlikte öz-düzenlemeli yapıların ele alındığı hibrit yaklaşımlardan esinlenilmiştir ve özdüzenlemeli ağ ve sınıf bilgisinin kullanıldığı özdüzenlemeli ağın eğitimi basit bir pekiştirmeli öğrenme yaklaşımı ile uyarlanarak yeni bir yaklaşım önerilmiştir. Öz-düzenlemeli ağ yapısı ve önerilen yaklaşımlar MATLAB ortamında gerçeklenmiş ve basit bir nokta sınıflama problemine ve Iris çiçekleri veri kümesine uygulanmıştır. Elde edilen benzetim sonuçları eğitim ve test kümelerindeki başarımları ele alınarak irdelenmiştir ve önerilen yöntemlerin geliştirilmesi için önerilerde bulunulmuştur. Bu çalışmadaki asıl amaç yeni bir öğrenme yöntemi önerip nasıl uygulanacağını açıklamak olduğundan yöntemin sınandığı problem kolaylıkla üretilebilen ve eğitim amaçlı çokça kullanılan iki boyutlu noktaların öbeklenme problemi ve çokça kullanılan sınama problemi olan Iris veri kümesi olarak seçilmiştir.

Vektör Kuantalamaöğrenme yapısı (Learning Vector Quantization-(LVQ)), SOM yapısı gibi verilerin öbeklenmesinde yaygın kullanılan bir yöntemdir. Bu yontemlerin her ikiside T.Kohonen tarafından önerilmiştir [1]. Kohonen yapısının diğer öbekleme yapılarina göre en önemli farki, sadece kazanan hücre değil, aynı zamanda komşu hücrelerinde sadece giriş verileri temsil edecek şekilde güncelleştirilmesi olmasıdır. Vektör Kuantalama yapısında komşuluktanımlı değildir ve bu nedenle yapı, kümeleme sürecinde sınıf bilgileri kullandığı içinbir denetimli öğrenme olarak kabul edilir [1,2]. SOM yapının yaygın kullanımına rağmen, iki dezavantajı vardır : yakın verileri bir birindan ayırt edememe ve yavaş yakınsama özelliği.

Bu tezde, asıl amaç özellikle bu iki dezavantajı göz önüne alarak SOM yapısını geliştirmekdir. Yani, öncelikle SOM yapısında sınıf bilgisi kulanılır daha sonra basit bir pekiştirmeli-öğrenmekuralı ile SOM yapıda öğrenme kuralı değiştirilecektir. SOM yapısı gibi Pekiştirmeli Öğrenme yaklaşımı da eğiticisiz öğrenme yaklaşımıdır ve yapının öğrenmesi aracının davranışlarının ortamda değerlendirilip ödülendirilmesine dayanmaktadır.

Böylece yeni öğrenme kuralı hem SOM ve hem sınıf bilgileri ile SOM için önerilmiştir ve bu önerilen kurallar MATLAB programında iki öbekleme problemi için uygulanmaktadır. Bu tezin temel amacı yeni öğrenme yöntemleri önermek ve bu yöntemlerin nasıl uygulanıcağnı açıklamaktır.

Önerilen yöntemler net bir anlayış vermek için, dikkate alan problemler kolayca elde edile bilirve eğitim amaçlı kullanılabilir problemlerdir. Bu yüzden ilk olarak iki boyutlu bir uzayda farklı ortalama değerler ve standart sapma ile olan random noktalar için uygulanmiştir, istatistiksel olarak anlamlı bir sonuç elde etmek için, ilgili m-dosyaları rastgele her deneme için 30 deneme olarak farklı eğitim setlerin performans ölçütlere göre değrlendirilip daha sonra ikinci bir problem olarak Iris veri kumesi için uygulanilmiştitr ve yine aynı ölçütleregöre değrlendiriliştir.

Bu tez kapsamında yapılan çalışma, SOM yapısını gerçekleyip, komşuluk bilgisinden faydalanarak öbekleme problemini çözmek, pekiştirmeli öğrenme yapısının en azından ödül mekanizmasını kullanarak, elde edilen SOM yapısına ilişkin yeni bir öğrenme yaklaşımı önermeyi kapsamaktadır. SOM diğer yöntemler arasında öbekleme problemini için kullanıldığının sebebi komşuluk avantajıdır. SOM yapısının genel performansını arttırırken komşulukdan yararlanmak için yapı aynı tutulması gerekiyor. Bu uygulamada ilk olarakVektör Kuantalama öğrenme kuralı olarak düşünüldi. Bu nedenle, kümeler sınıf bilgisi kullanılarak oluşturulur. Daha sonra, pekiştirmeli öğrenme ile öz-düzenlemeli ağ yapıları birleştirerek, ödül etkisi eğitim sırasında göz önüne olındı.

Önerilen eğitim algoritmaları SOM ve önerilen üç farklı yapıların etkinliğini test etmek için MATLAB ortamında uygulanmaktadır. Örnek olarak genellikle iki farklı problem düşunülmüştür ve sonuç olarak genel bir tartışma gerçekleştirip ve bu sorunlar için simülasyon sonuçları her bir bölümde verilmiştir. Kümeleri, giriş verileri daha iyi temsil etmesi içinkümelerin oluşumundan sonra, 100 adımlar üzerine kadar eğitim aşaması devam edilir.Bu işlemle veri gruplamada kazanan nöronun ortalama mesafelerde binde biri fark sağlanır.

Yakın zamanda önerilen kimi çalışmalarda pekiştirmeli öğrenme ile özdüzenlemeli yapılar bir arada kullanılmaktadır. Bunlardan bazıları önerilen eğitim kuralında, SOM yapısı pekiştirmeli öğrenmede yeri olan aracı için ilk eğitim verilerini az sayıdaki eğitim kümesinden oluşturmakta kullanılmıştır. Böylece eğitim sürecinin hızlanması sağlanmıştır. Diyer önerilen metod da ise kamera görüntülerinden oluşmuş eğitim kümesi verilerinin sınıflanması SOM ve pekiştirmeli öğrenme içiçe kullanılarak sağlanmıştır. [6] ise sinirbilim ile ilişkili olarak primatların görme sistemini SOM yapısını bazı ampirik kurallar ile pekiştirmeli öğrenme çerçevesinde uyarlayarak modellemeyi amaçlamaktadır.

Sonucu olarak bu yöntemleri uyguladiktan sonra Iris veri seti için yöntemleri arasında çok fark yok iken belirlenen 2 boyutlu nokta verileri için daha hızlı yakınsama olmuştur. Önerilen yöntemler arasinda random noktalar için S-SOM en iyi sonuç veren yadıdır. Ancak ödül mekanizması daha iyi kurgulandığında RL-SOM yapısınında birbirine çok benzer verileri farklı öbeklere yerleştirebilme yeteneğinden faydalana bilmenin mümkün olabileceği görülmektedir. Yöntemlerin kararlı davrandığı tamamen farklı veriler ile aynı verilerin kullanıldığı durumlarda sonuçların çok değişmemesinden sonuçlarda izlenebilir. Nöron sayısının artması sonuçları belli bir değere kadar olumlu etkilerken 10 nöron için elde edilen sonuçlar her zaman diğer sonuçlardan daha iyi olmamıştır. Dolayısıyla nöron sayısının çok fazla artması sonuçleı herzaman olumlu etkilememektedir.

1. INTRODUCTION

Both Self-Organizing Map (SOM) and Learning Vector Quantization (LVQ), used especially in reducing the size of data clustering widely, are proposed by T.Kohonen [1]. The most important difference of SOM structure compared to other clustering structures is that not only the winner cell, but also the neighboring cells in one or two-dimensional predefinedlattice are updated in a way to represent the input data. In the LVQ structure neighborhood is not identified and because of using class information in the process of clustering this structure is often treated as a structure that includes supervised learning [1,2]. Despite the widespread use of the SOM structure, it has two drawbacks: the inability to distinguish between the affinity data and slow convergence feature. In this thesis, especially these two drawbacks are considered and the aim is to improve these aspects of SOM. So, primarily class information will be used in SOM structure and then with a simple reinforcement-learning rule, learning rule for SOM will be amended. New learning rules are proposed for both SOM and SOM with class information and these proposed rules areimplemented in MATLAB softwareand these are applied to two clustering problems. The main purpose of this thesis is to propose new learning methods and explain how to implement thesenew methods. In order to give a clear understanding of the proposed methods, the problems considered are theones that can be easily obtained and used for educational purposes [2]. So, clustering set of points with different mean values and standard deviations in a two-dimensional space is considered first, then clustering the Iris data is considered as a second problem [2].

The scope of this thesis covers implementation of SOM structure to solve clustering problems and to improve this structure by proposing new training approaches. SOM is preferred amongst the other methods used for solving clustering problems as it uses the advantage of neighborhood. To improve the performance of SOM by keeping its structure same to make use of neighboring, first the learning rule of LVQ is considered. Therefore, the clusters are formed using the class information. Then, combining self-organizing structure with reinforcement learning, the effect of the reward is considered during training. To test the efficiency of the proposed training

algorithms SOM and the proposed three different structures are implemented in MATLAB environment. As examples two different problems are considered and the evaluation measures resultsas sensitivity, specificity, accuracy, precision and similarity for these problems are given in each Chapter while in the conclusion an overall discussion is carried.

In the second Chapter, first clustering problem is defined and recent works , where Self-Organizing Map is used in combination with reinforcement learning [20-22] are reviewed. In one of these studies [20], the proposed training rule uses SOM structure to create a small number of data for the agent in the reinforcement learning as initial training data. Thus, the training process has been accelerated. In [21], a method is proposed for the classification of the training data, which are formed by the camera images. SOM and reinforcement learning areaccomplished together in a nested training structure. In [22] based on the knowledge obtained about the primate visual system from neuroscience studies, SOM structure is used with reinforcement learning to adapting some empirical rules governing the visual perception.Then, performance measures used in evaluating the methods proposed in this thesis are summarized.

In the third Chapter, a review of SOM is given and the algorithm used in forming m-file is discussed in detail. This Chapter is concluded with simulation results obtained for both clustering problems considered in this thesis. In the fourth Chaptergeneral information about reinforcement learning is given. In Chapter 5, the three training rules proposed in this thesis are introduced and simulation results for the considered problems are given for all three proposed structures. In the fifth Chapter, first the simulation results for all four approaches are summarized with tables and comparison of these approaches are given, thenChapter is concluded with discussion about how to improve the results further. Some results are also given in apendix.

2. APPROACHES AND EVALUATION MEASURESUSED IN CLUSTERING PROBLEM

2.1 Clustering Problem:

Clustering refers to the task of grouping a set of objects into meaningful clusters in such way that objects in the same cluster are more similar to each other than to those in other clusters. It is used for identifying hidden patterns and disclosing crucial knowledge from large data collections. The application areas of clustering include exploratory data mining, machine learning, pattern recognition, image segmentation, information retrieval, document classification, transaction analysis, web usage tracking and bioinformatics.

Clustering itself is not one exact algorithm, but the general task to be solved. Since a different type of clustering methods have been progressed, clustering keeps on a challenging task due to the fact that a clustering algorithm behaves differently depending on the preferred characteristic of the data set and the parameter values of the algorithm [18]. Therefore, it is important to have some detached measures to survey the quality of the clustering. While clustering problems are with no prior knowledge, a quantitative evaluation of the clustering algorithm serves as an important reference for different kind of tasks, for instance getting knowledge of the distribution of a data set, recognizing the clustering model that is most relevant for a problem scope, and determining the optimal parameters for a specific clustering method. Favored characters of clusters include groups with small distances among the cluster members, compact areas of the data space, interspaces or especial statistical distributions so in this way clustering can be devised as a multi-objective optimization problem [19-29].

Usually clustering is failed to distinguish with classification, while there is a differencebetweeneach other. In classification the objects are allocated to predefined classes, whereas in clustering the classes are also to be defined.

2.1.1 Different methods

There are different kind of clustering methods and each of them brings various results for grouping of dataset. Choosing anappropriate method will rely on the way of output desired. Considering with overall characteristic, clustering methods can be divided into two main category such as hierarchical and partitional clustering methods since whithin each category there exists a substantially of subtypes and different algorithms for discovering the clusters [19].

Hierarchical clustering goes on by either amalagmating smaller clusters into larger ones, or by dividing larger clusters into smaller. The clustering methods depend on the rule by which it is decided which of two small clusters should beassimilated or which large cluster should bedivided. At the end, there is a tree of clusters which is called a dendrogram, that shows how the clusters are associated. Clustering the data sets into disjoint groups can be obtained by cutting the dendrogram at a desired level [19].

Partitioning methods are trying to directly crumble the data items into a set of separategroups. The main factor the clustering algorithm attempts is to minimize the local structure of the data, as by allocateing clusters to peaks in the probability density function, or minimize the global structure. In general the global factorinclude minimizing some value of contrast in the samples within each cluster, while maximizing the difference of various clusters [29]. In this category, K-Means is a commonly used algorithm [31-19]. The main purpose ofK-Means clustering is the optimisation of an objective function that is described by thee quation

$$E = \sum_{i=1}^{c} \sum_{x \in C_i} d(x, m_i) \qquad \textbf{(2.1)}$$

In the above equation, m_i is the center of cluster C_i, whereas $d(x, m_i)$ is the Euclidean distance between a point x and m_i . Thus, the criterion function E tries to minimize the distance of each point from the center of the cluster to which the point belongs. Firstly the algorithms start by initialising a set of C cluster centers. After that,it allocates each sample of the dataset to the cluster whose center is the nearest, and recalculates the centers. The process continues until no noticeable changes in the centers of the clusters observed [30]. Another algorithm of this category is Adaptive Resonance Theory (ART) which is a theory inspied by how the brain processes information. There are two mainstructures of training ART which are slow and

fastmethods. In the slow one, the degree of training of the recognition neuron's weights towards the input vector is computed to continuous values with differential equations and therefore the input vector presentation is relying on the length of time. While for fast learning, algebraic equations are used to compute the degree of weight adjustments to be made, and binary values are used. Since fast learning is effective and efficient for diffrent kind of assignments, the slow learning method is more biologically conceivable and can be used with continuous time [37]. Third algorithm of the partitioning category is Self-Organizing Map (SOM) as it iswidely used for clustering and visualization. The patricular feaure of SOM is that the clusters in a SOM network are organized in a multi-dimensionalmap. During learning, the network updates not only the winner's weight,but also the weights of the winner's neighbors. Then these results areplaced in an output map in a way that the similar clusters are kept together [38]. In the next Chapters it willbe discussed more about the SOM structure.

2.1.2 Hybrid methods

On some occasions to overcome some of the short comings of techniques, there is need to havea hybrid method obtained with merging different techniques as I done in this thesis. Several researchers have surveyed Self-Organizing Map (SOM), Reinforcement Learning (RL) and Learning Vector Quantizatiton (VQ). This Chapter arranged to collect some experimental and analytical studies from the literature.

Recently there are some works, where Self-Organizing Map is used in combination with reinforcement learning [20-22]. In one of these studies [20], the proposed training rule uses SOM structure to create a small number of data for the agent in the reinforcement learning as initial training data. Thus, the training process has been accelerated. In [21], the proposed method is for the classification of the training data, which are formed by the camera images, SOM and reinforcement learning area ccomplished together in a nested training structure. In [22] based on the knowledge obtained about the primate visual system from neuron science studies, SOM structure is used with reinforcement learning to adapting some empirical rules governing the visual perception. Buendia-Ramon et al. (2011) proposed a qualitative approach to

obtain the relationship between the parameters of the methods (e.g., the learning rate in a neural network) and the performance evaluation of those methods by using Self-Organizing Maps (SOMs). While this work has introduced the SOM as a device for visual data mining it is used to conclude the rules that depend on the different parameters of a Reinforcement Learning method to its related performance evaluation. The methodology can be spread to other Machine Learning models, thus helping non-experts to optimize the paramaters of a certain model [23]. Smith proposed a new model based on the SOM of Kohonen which lets either the one-to-one, many-to-one or one-to-many method of the desired state-action mapping to be recorded. In spite of the fact that as presented there for tasks concluding actual reward, the method is easily extended to deferred reward. Then he conclude that the SOM is useful for providing real-time, on-line generalisation in reinforcement learning problems in which the concealed dimensionalities of the state and action spaces are small [24]. Somervuo and Kohonen (2004) constructed the SOM and Learning Vector Quantization (LVQ) algorithms for variable-length and complicated feature sequences. The novel approach is to combine an entire feature vector, as a model with each SOM node. Dynamic time wraping is used to obtain time-normalized distance between sequences with various extends. The proposed models, can then be used for the identification as well as structure of pattern [25]. Dutech (2012) presented a developmental reinforcement learning structurein order toexaminwealthy, complicated and large sensorimotor spaces. The central part of that architecture is made of a function approximator lies on a Dynamic Self-Organizing Map (DSOM). The life-long online learning feature of the DSOM lets us to take a progressive approachto learning a robotic task. This structureis examined on a robotic assignment that looks simple but is still an endurance task forreinforcement learning [28].

As it mentioned before clustering is based on similarity. In clustering analysis it is imperative to compute the similarity or distance. So when data is too large or data arranged in a scattered manner it is quite difficult to properly arrange them in a group and the main problem is that the calulation takes long time. So to overcome this problem in SOM method a new algorithm is proposed in this thesis, which performs two methods to decline the time of processing of clustering. We expected thatthe accuracy of result is much greater as compared to SOM algorithm. So the aim of this thesis is to implement the SOM structure to solve clustering problems and to

improve this structure by proposing new training approaches. So relying on this fact, firstly the learning rule of LVQ is considered. Therefore, the clusters are formed using the class information. Then, combining of self-organizing structures with reinforcement learning, the effect of the reward is considered during the training.

2.2 Evaluation Measures

Evaluation of clustering results sometimes is referred to as cluster validation. There have been several suggestions for a measure of similarity between two clusterings. Such a evaluation can be used to compare how well different data clustering algorithms perform on a set of data. These measures are usually lied to the type of factor being considered in assessing the quality of a clustering method. The evaluation measures which are used in this thesis are as sensitivity, specificity, accuracy, precision and similarity.

Sensitivity and specificity are statistical measures of the peformance of binary classification test, also known in statistics as classifiction function. Sensitivity measures the proportion of actual positives which are correctly identified. Specificity measures the proportion of negatives which are correctly identified. The accuracy of a measurement system is the degree of closeness of measurements of a quantity to that quantity's actual value. The precision of a measurement system, also called reproducibility or repeatability, is the degree to which repeated measurements under unchanged conditions show the same results [16-17]. Since similarity is fundamental to the definition of a cluster, a measure of the similarity between two patterns drawnfrom the same feature space is essentialto most clustering procedures. Because of the variety of feature types and scales, the distance measure (or measures) must be chosen carefully. It is most common to calculate the dissimilarity between two patterns using a distance measure defined on the featurespace. So in this thesis, we focused on the well-known distance measure which is Euclidean distance [26]. There are several terms that are commonly used along with the description of sensitivity, specificity, accuracy and precision. They are true positive (TP), true negative (TN), false negative (FN), and false positive (FP). The terms positive and negative refer to the classifier's prediction (sometimes known as the expectation), and the terms true and falserefer to whether that prediction

corresponds to the external judgment (sometimes known as the observation). Both false positive and false negativeindicate that the test results are opposite to the actual condition. In general, Positive defines the identified and negative defines the rejected. Therefore, True Positive is equal to correctly identified, False Positive is equal to incorrectly identified, True Negative is equal to correctly rejected and False Negative defines the incorrectly rejected. In that setting in this thesis TP,FP,TN,FN are defined as below:

Positive = classified in a class, Negative = classified out of a class

True Positive = correctly classified in a class

False Positive = incorrectly classified in a class

True Negative = correctly classified out of a class

False Negative = incorrectly classified out of a class

Sensitivity, specificity, accuracy and precision are described in terms of TP, TN, FN and FP.

$$\text{Sensitivity} = \frac{TP}{(TP+FN)} \tag{2.2}$$

$$\text{Specificity} = \frac{TN}{(TN+FP)} \tag{2.3}$$

$$\text{Accuracy} = \frac{(TN+TP)}{(TN+TP+FN+FP)} \tag{2.4}$$

$$\text{Precision} = \frac{TP}{(\text{TP}+\text{FP})} \tag{2.5}$$

As suggested by above equations, sensitivity is the proportion of true positives that are correctly identified by a diagnostic test. It shows how good the test is at detecting a class. Specificity is the proportion of the true negatives correctly identified by a diagnostic test. It suggests how good the test is at identifying normal (negative) condition. Precision and accuracy are terms used to describe systems and methods that measure, estimate, or predict. In all these cases, there is some parameter you wish to know the value of. This is called the true value, or simply, truth. The method provides a measured value, that you want to be as close to the true value as possible. Precision and accuracy are ways of describing the error that can exist between these two values [15].

And we defined similarity in this thesis as :

$$\text{Similarity} = \frac{\sum_{i=0}^{n} (\, norm\,(\, input_i - w_j\,))^2}{n} \quad \textbf{(2.6)}$$

Where$n$ is number of input sample and w_j denotes the winning neuron.

3. SELF-ORGANIZING MAP

The Self-Organizing Map (SOM), also known as Kohonen map, is one of the most popular neural network models and is developed by Professor Teuvo Kohonen in 1982. It is inspired by the sensory and motor mappings in the mammal brain, which appear to be automatically organizing the information derived through senses in a topologically organized way. SOMs organize the data into clusters on their own through unsupervised competitive learning. The word "Maps" is used because SOMs attempt to map their weights to conform to the given input data. The weights defining the nodes in a SOM network attempt to become as the inputs presented to them through training. In this sense, the network is trained unsupervised. They are also called Feature Maps, as in Self-Organizing Feature Maps. Retaining the principle features of the input data is a fundamental principle of SOMs, and one of the reasons that makes them so valuable. Specifically, the topological relationships between input data are preserved when mapped to a SOM network [8]. This has a pragmatic value of representing complex data and makes SOMs different from other artificial neural networks designed for clustering in the sense that they use a neighborhood function that preserve the topological properties of the input space. The property of topology preserving means that the mapping preserves the relative distance between the points. Points that are near each other in the input space are mapped to nearby units in the map used in SOM. SOM is used both to cluster data and to transform an incoming signal pattern of arbitrary dimension into a one or two dimensional discrete map, and to perform this transformation adaptively in a topologically ordered fashion therefore it can be named as a data visualization technique since it helped to solve data visualization problem which is important as humans simply cannot visualize high dimensional data.

Self-Organizing Map has proven to be useful in many technical applications. In an industrial setting, the SOM has been applied, for example, in process and machine state monitoring, fault identification and in robot control [5]. In process and systems

identification, the use of SOM is well motivated. The number of state variables may exceed the number of measurements by an order of magnitude. In addition, the state variables may be non-linearly related. In this case, the analytical model of the plant in question would not be identifiable from the measurements. In fault diagnosis, the SOM has two functions. Firstly, the SOM can be used in detecting the fault and in identifying it. One can detect faults even if there are no measurements of the faulty states by investigating the quantization error between the SOM and the measurements. If the quantization error exceeds a pre-defined limit, a faulty state has occurred. If one needs also to identify faults, representative examples of the faulty situations must have been recorded. In an application, measurements are made from a computer system in a network environment. The system is measured in terms of utilization rates of the central processing unit and traffic volumes in the network. It is not known a priori what the characteristic states in the operation of the system would be. So SOM can be used in clustering data measured from the system, or to form a representation of the characteristic states. Such a representation would be useful in monitoring the current state of the system [5].

3.1 Structure of SOM

SOM is set up by placing neurons at the determined nodes of a one or two or higher dimensional lattice. As one of the aim of SOM is to provide a mean for visualizing high dimensional data, lattices with higher dimensional are not so common. SOM structure consisting of two layers of neurons is illustrated in Figure 3.1[3].The first layer is not actually a processing layer; it only receives the input data and transfers it to the second layer. Each neuron of the second layer connects with each neuron of the first layer but no neuron of second layer connects to each other. Since each neuron has a location in the gird chosen, it also has neighboring neurons that are close to it.

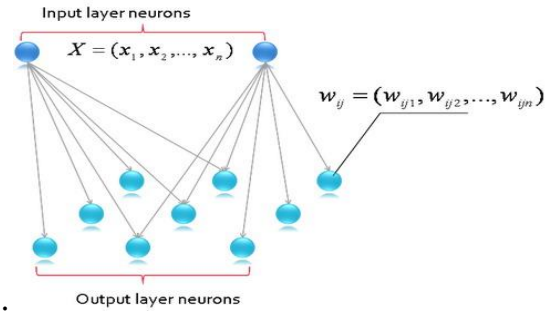


Figure 3.1: The Architecture of SOM [3]

By calculating, the distance of each neuron weight to the considered input datum we can determine that which neuron is more similar to the input considered by determining the smallest distance between neuron weights and the input. In this configuration, each node on the map has a unique (i, j) coordinate. This makes it easy to refer a node in the network, and to calculate the distances between nodes, which will be used when we attempt to find the radius of the neighborhood of the winning node while updating its weights.

The number of neurons in the second layer can be chosen arbitrarily, and differs from task to task. Each neuron of the second layer defined by its own weight vector and its dimension is equal to the dimension of the input vector. The neurons are adjacent to other neurons by a neighboring relation, which dictates the topology or structure of the map. Such a neighborhood relation is assigned by a special function called a topological neighborhood. Therefore, the topology of the lattice defines a neighborhood structure on the neurons, like those illustrated below in Figure 3.2 and Figures 3.3 [8].

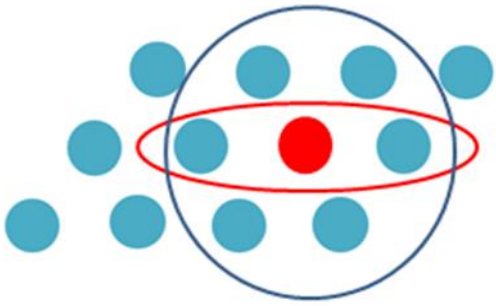


Figure 3.2 :2-dimensional topology and two possible neighborhoods [8].

Figure 3.3 :1- dimensional topology with small neighborhood [8].

Here to explain better, how the topological relations between input data is preserved during the training of a self-organizing map is illustrated in Figure 3.4. In Figure 3.4 [4] the blue blob is the distribution of the training data, and the small white lattice is the current training sample drawn from that distribution. At first (left) the SOM nodes are arbitrarily placed at the nodes of lattice. The node nearest to the training node (highlighted in yellow) is selected, and is moved towards the training datum, as

(to a lesser extent) are its neighbors on the grid. After many iterations, the grid tends to approximate the data distribution (right) [4].

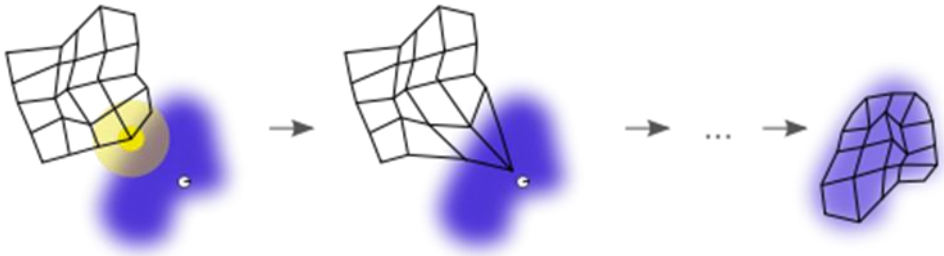


Figure 3.4 : Training SOM by preserving the topological relations [4].

3.2 Training Process of Self-Organizing Map

The way that SOMs go about organizing themselves is by competition between neurons for representation of the samples. The weights of neurons are allowed to change themselves through training to become more like samples. By adapting their weights, they are expected to be winners in the next competition. This selection of winner and training process makes the weights organize themselves into a map representing similarities.

As we mentioned a SOM does not need a priori given target output to be specified unlike many other types of artificial neural network structures. Instead, when the node weights match the input vector, that area of the lattice is selectively optimized according to the neighborhood function to provide close resembling of the data for the class from which the input data is chosen. Either from an initial distribution of random weights, or weights randomly chosen from the data set and over many iterations, the SOM eventually settles into a map of stable zones. Each zone is effectively a feature classifier, so you can think of the graphical output as a type of feature map of the input space.

By using a very simple algorithm and over many iterations, one can order weight vectors in such a way that, they will represent the similarities of the sample vectors. In the next Chapter, SOM algorithm used in this thesis will be introduced.

3.2.1 SOM Algorithm

The Self-Organizing Map algorithm can be summarized in 6 steps:

Sampling : The first step in constructing a SOM is to choose a vector at random from the set of input data and present it to the network. Let m denote the dimension of the input then the input vector that is selected at randomly from the input space can be denoted by

$$x = [x_1, x_2, \dots, x_m] \qquad (\mathbf{3.1})$$

Initialization : In this step, the weight vectors of the neurons that are placed at the chosen lattice must be initialized. As we mentioned before these weight vectors have the same dimensions as the sample vectors. There are a number of ways to initialize the weight vectors, here we just mention three different types of network initializations method that Kohonen presented: random initialization, initialization using initial samples and linear initialization [5].

1. **Random Initialization :** Random initialization means simply that random values are assigned to weight vectors. This is the case if nothing or little is known about the input data at the time of the initialization.
2. **Initialization Using Initial Samples :** Initial samples of the input data set can be used for weight vector initialization. This has the advantage that the neurons automatically will lay in the same part of the input space with the data.
3. **Linear Initialization :** Another initialization method takes advantage of the principal component analysis (PCA) of the input data. The weight vectors are initialized to lie in the same input space that is spanned by two eigenvectors corresponding to the largest eigenvalues of the input data. This has the effect of stretching the Self-Organizing Map to the same orientation as the data that is the most significant.

A synaptic-weight vector of j^{th} neuron is denoted by[1]

$$w_j = [w_{j1}, w_{j2}, \dots, w_{jm}]^T \quad , \ j=1, 2, \dots, l \qquad \mathbf{(3.2)}$$

Where l is total number of neurons in this thesis and m denoting the dimension of the input data set.

Similarity Matching : Chosen an input from data set, every node corresponding to a neuron is checked out to test whether its weights are similar the input vector or not. The winning node is commonly named as the Best Matching Unit (BMU). The BMU is determined by running through all weight vectors and calculating the distance

from each weight to the chosen sample vector. The neuron that has weight with the shortest distance to the input vector is the winner, as it is most similar neuron to the considered input amongst all neurons. If there is more than one neuron with the same distance, then the winning weight is chosen randomly among the weights with the shortest distance. There are a number of different ways to determine the distance between weights and the input. Actually, mathematically this corresponds to which norm is used, however, the most commonly used one is the *Euclidean Distance*. The Euclidean distance is given as [1]:

$$Dist = \sqrt{\sum_{j=1}^{j=n}(x_j - w_j)^2}, \qquad j = 1,2,\dots,l \tag{3.3}$$

Where x_j is the data value at the ith data member of a sample, n is the number of dimensions to the sample vector and w_i is the synaptic-weight vector of jth neuron.

Topological Neighborhood : Since each weight vector has a location in the gird chosen, it also has neighboring neurons that are close to it. The weight of the BMU is updated to become more like to the randomly selected sample vector. In addition to this updating, the weights of the neighboring neurons to the BMU are also updated and so their weights become more like to the chosen sample vector.

In this step to determine the neighboring neurons, the neighborhood radius of the BMU is calculated. The value of the updating term for neighboring neurons is largest for the nearest neighbor and gets smaller as the distance to the BMU unit is increased. Typically, the radius of the neighbors in the lattice decreases over time. Any node within this radius is deemed to be inside the BMU's neighborhood. Let us denote the Topological Neighborhood by $h_{j,i}$ centered on winning neuron i and encompasses a set of excited neurons, a typical one is denoted by j. $d_{j,i}$ denote the lateral distance between winning neuron i and the excited neuron j in the out-put space rather than on some distance measure in the original input space. In the case of two-dimensional lattice it is defined by the following expression [1]

$$d^2_{j,i=\|r_j - r_i\|} \tag{3.4}$$

The topological neighbourhood is symmetric about maximum point defined by $d_{j,i} = 0$. In other words, it attains maximum value at winning neuron i for which the distance $d_{j,i}$ is zero. The amplitude of it decreases monotonically with increasing lateral decaying to zero. Therefore, good choice of $h_{j,i}$ that satisfies these requirements are the Gaussian function [1].

$$h_{j,i(x)}(n) = exp\left(-\frac{d_{j,i}^2}{2\sigma(n)^2}\right) \qquad n = 0,1,2,\ldots \qquad \textbf{(3.5)}$$

The parameter σ is the "effective width" of topological neighborhood. It measures the degree to which excited neuron is approximately the winning neuron participate in the learning process.

Figure 3.5 shows a plot of the neighborhood function used. As the time progresses, the base goes towards the center, so there are less neighbors as time progresses. The initial radius is set high, some value near the width or height of the map [6].

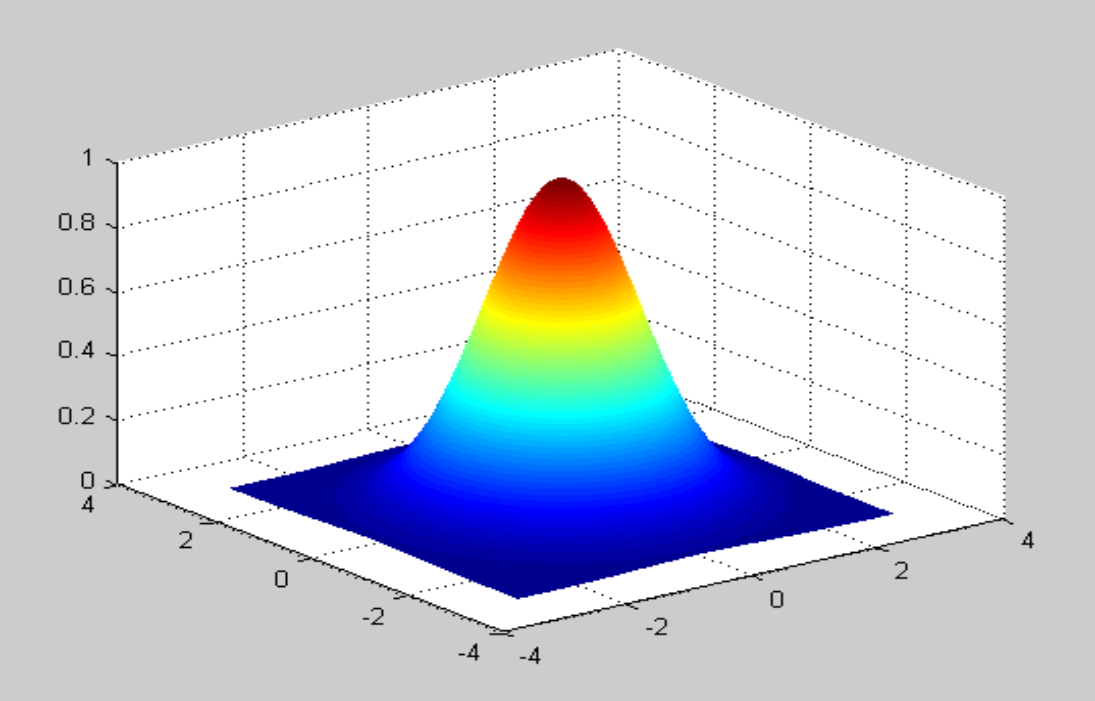


Figure 3.5 : Gaussian neighborhood function

Figure 3.6 [7] illustrates an example of the size of a typical neighborhood close to the commencement of training.

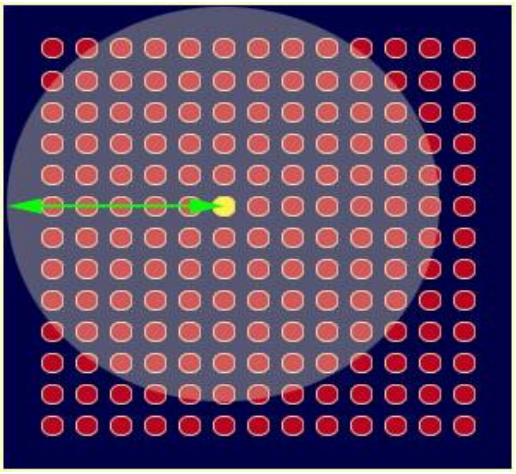


Figure 3.6 : The BMU's neighborhood [7]

One can recognize that the neighborhood shown above is centered on the BMU and encompasses most of the other nodes. The arrow shows the radius. A unique feature of the Kohonen map's learning algorithm is that the neighborhood shrinks over time. This is accomplished by using the exponential decay function [1]

$$\sigma(n) = \sigma_0 \exp\left(-\frac{n}{\tau_1}\right) \quad , \quad n = 0,1,2,\ldots \qquad \textbf{(3.6)}$$

Where σ_0 is the value of σ at the initiation of SOM algorithm and τ_1 is a time constant to be chosen by designer. Figure 3.7 [7] show how the neighborhood in Figure 3.6 decreases over time. The figure is drawn assuming the neighborhood remains centered on the same node, in practice the BMU will move around according to the input vector being presented to the network.

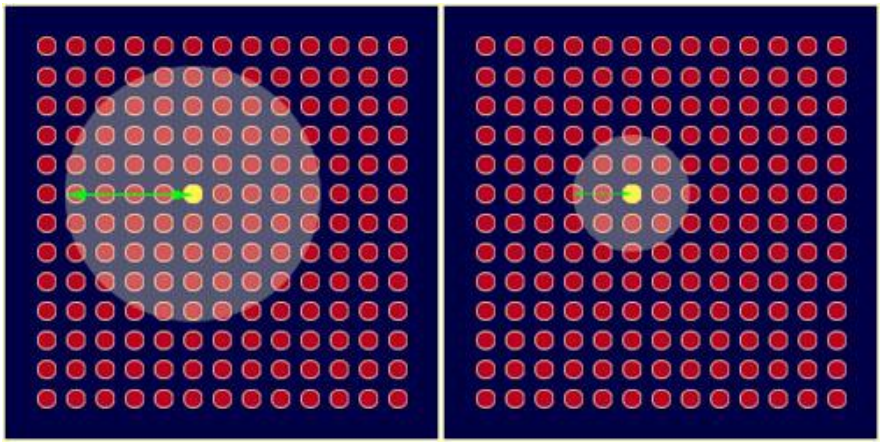


Figure 3.7 : The ever shrinking radius [7]

Updating : Each neighboring node's weights are adjusted to make them more like the input vector. The closer a node to the BMU, the more its weights is altered. Using

the synaptic-weight vector $w_j(n)$ of neuron j at time n we can define the updated weight vector $w_j(n+1)$ at time $(n+1)$ by [1]

$$w_j(n+1) = w_j(n) + \eta(n)h_{j,i(x)}(n)(x(n) - w_j(n) \quad (\mathbf{3.7})$$

Where n represents the time-step and η is a small variable called the learning rate, which decreases with time, in particular it should start at some initial value η_0 and then decrease gradually with increasing time n. This requirement can be satisfied by the following heuristic [1]:

$$\eta(n) = \eta_0 \, exp\left(-\frac{n}{\tau_2}\right) \quad , \quad n = 0,1,2,\ldots \quad (\mathbf{3.8})$$

Where τ_2 is another time constant of the SOM algorithm.

Continuation : Repeat step 2 until no noticeable changes in the feature map are observed.

3.3 Discussion on SOM

Probably the best thing about SOM is that it is very easy to understand and because it used unsupervised learning method no external knowledge is needed during the training. Another great thing is its excellent capability to visualize high- dimensional data onto one or two dimensional space that makes it unique. Especially its topology preserving property of topology during dimensionless this helped humans to visualize high dimensional data easily. But one major problem with SOMs is getting the right data. Unfortunately one needs a value for each member of samples in order to generate a map. Sometimes this simply is not possible and often it is very difficult to acquire all of the data so this is a limiting feature of SOMs often referred to as missing data. Another problem is that every SOM is different and finds different similarities among the sample vectors. SOMs organize sample data so that in the final product, the samples are usually surrounded by similar samples, however similar samples are not always near each other or near samples always are not belong to same cluster [6].

The final major problem with SOM is since the training is an iterative process through time, it requires a lot of computational effort and thus is time-consuming thesis and also the number of neurons affects the performance of the algorithm also.

As the number increases the computation increases which results in increasing computational time.

Here in this thesis to improve clustering property of SOM firstly, self-organizing map is modified to use class information like learning vector quantization method. Also, inspired by the hybrid methods proposed recently which combine reinforcement learning with self-organizing map, a novel approach is proposed where a simple reinforcement learning algorithm is used both in self-organizing map and self-organizing map with class information [10].

3.4 Clustering Applications Using SOM:

As it is mentioned previously in this thesis, all the methods considered are applied to the simple clustering problems. In this Chapter, these problems are presented and solved with SOM. Two different problems are considered. In the first problem two sets of 2-dimensional random data points are generated. Since it is easy to follow two-dimensional problem, this problem is a characteristic example in the clustering problems [1].

As a second problem, SOM applied to a benchmark problem. This benchmark problem deals with real data set composed of features of Iris flowers and Sir Ronald Fisher (1936) as an example of discriminate analysis introduces it. This data set sometimes called Anderson's Iris data set, because Edgar Anderson collected the data to quantify the morphologic variation of Iris flowers of three related species [18].

3.4.1 Random data points

In this example, we have constructed a simple data set with two sets of two dimensional data points. Each class consists of 10 data points. For the first class random numbers are chosen from a normal distribution with mean (0,0) and standard deviation one and for the second class random numbers are chosen from a normal distribution with mean (2,2) and standard deviation two. Here, SOM is realized in MATLAB environment and with this m-file clustering of these two dimensional points are realized. In Figure 3.8, the points in the data set are given for one trial. To

give statistical meaningful results, the problem is solved for 30 times with different sets created with same statistical properties.

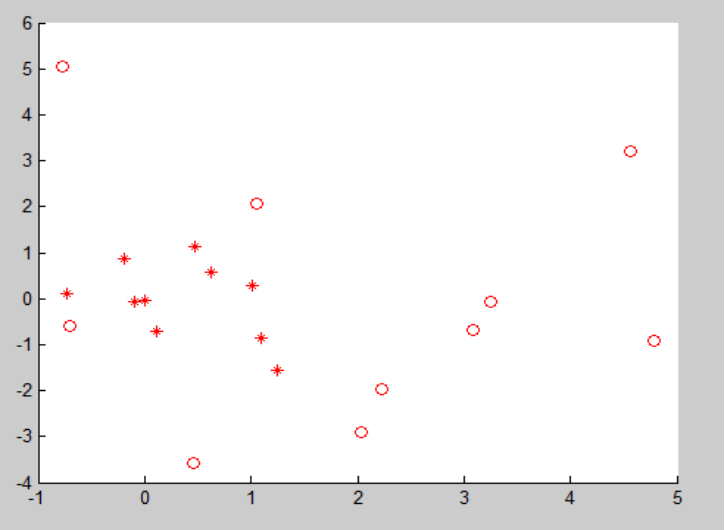


Figure 3.8 : The plot of training data set

Table 3.1 denotes the parameter values that have been used for SOM structure for this example. Here *k* parameter denotes the number of training data points, which is 20 in this example, where 10 points belongs to class one, the other 10 belongs to class two, and for the test data set, we have 3 points from each class so we have totally 6 points in the test set. Here l is the number of neurons whereas in this problem 4, 6, 8 and 10 neurons are considered. Since m is the dimension of data set, so in this example it is equal to two, η_0 is the initial value of SOM's learning rate and σ_0 is the initial value of the effective width of topological neighborhood.

Table 3.1 : List of parameters and their values

Parameter	η_0	σ_0	η	m	l	k
Parameter value	1	3	2.5	2	4,6,8,10	Training : 20 Test : 6

In Figure 3.9, 2-dimensional lattice is given when four neurons are used. To determine the neighborhood the coordinate of each neuron is specified. The coordinate of the first neuron which is representing the first cluster in 2-dimensional geometry is (4, 2) and the coordinates of the neurons which are representing the second, third fourth cluster are (4, 4), (3, 3), (2, 2), respectively.

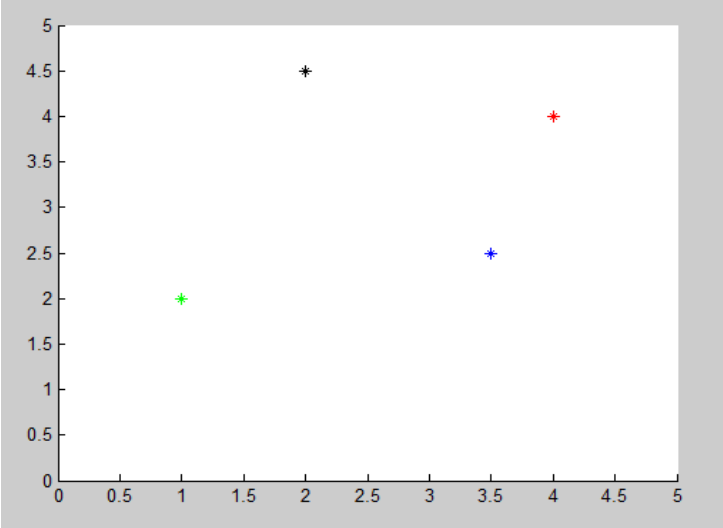


Figure 3.9 : The coordiantes of four neurons in 2-dimensional lattice.

3.4.1.1 Simulation results

By applying SOM to the data setgiven in Figure 3.8, the clusters are formed as in Figure 3.9 at the end of the training the information related to each cluster and each class, are provided in Figure 3.10. As it can be followed from the Figure 3.10, there are two incorrect points in the cluster1 that are provided by SOM. It can be seen easily from the figure that these two points belongs to class1 that must not be in the same cluster with class2 points, so as it mentioned in previous Chapters SOM cannot be recognize similar points and in this thesis, we would try to solve this problem of SOM.

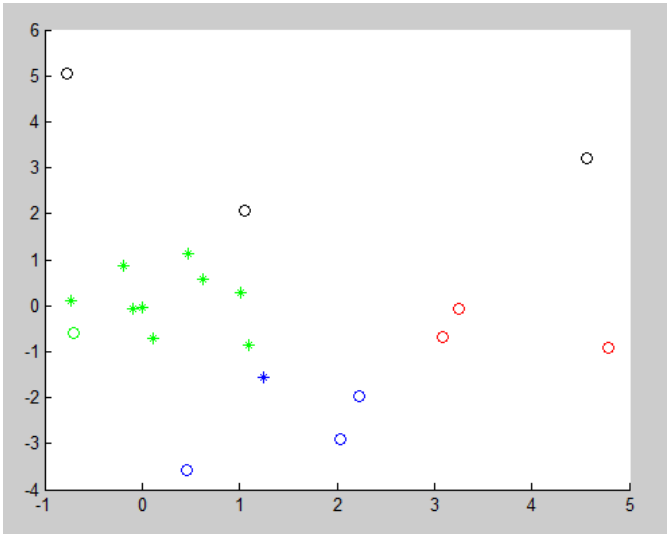


Figure 3.10 : Result of SOM structure for the data set given in Figure 3.8.

In order to have statistically significant conclusion, the related m-files are executed30 times in two different ways. At first for different training sets, which are produced randomly with same statistical properties for 30-timesare considered then keeping the training and test data same, 30 different training processes are completed and in both cases for comparison the performance of SOM, averages of theevaluationmeasuresare given. In Table 3.2 and Figure 3.11 theresults for 30 different trials with randamtraining data are given. In this table at first SOM for 4 neurons then 6, 8 and 10 neurons are considered andn the Table 3.3 the results of the test sets considered. All the outcomes for the second type of training and test sets are given in the appendix.

Table 3.2 : Results of SOM structure for the training set. The average is taken over 30 trials. The training set is regenerated randomly in each trial with same statistical properties.

Number Of Neuron	Sensitivity	Specificity	Accuracy	Precision
4 Neuron	0.229±0.017	0.9403±0.0242	17.1596±1.4629	0.823±0.073
6 Neuron	0.151± 0.010	0.9720±0.0126	18.0064± 1.299	0.860±0.065
8 Neuron	0.115±0.008	0.9854±0.0096	18.7981±1.3648	0.897±0.068
10 Neuron	0.090±0.006	0.9875±0.0070	18.6243±1.2859	0.887±0.064

The sensitivity of the test given the proportion of the data that are classified correctly in a class thus according to the Table 3.2 which illustrates the outcomes of SOM. The Sensitivity for the 4 neurons is approximately 23%, so the method is able to cluster almost less than a quarter of the points correctly in a class.

The Specificity gives the probability of a negative test, it means that it gives the data which are classifed correctly out of a class. The test has almost 94% Specificity for the 4 neuron. In other words, more than four-fifth of the tarining samples classified correctly out of the classes.

The accuracy shows the proportion of true results (both true positives and true negatives) in the population so theaccuracy refers to the closeness of a measured value to a standard or known value. In this test the accuracy for the 4 neuron is about 17 which means that 17 points out of 20 ones correctly have been classified.

Precision is the degree to which several measurements provide answers very close to each other. It is an indicator of the scatter in the data. The lesser the scatter, higher the precision thus according to the Table 3.2 the precsion of SOM for 4 neurons is just over the four fifth of the training test. So SOM has a good accuracy and a good precison. Further more according to the table as the number of neuron increase the Sensitivity declines but the Specificity, accuracy and precision become better.

Table 3.3 : Results of SOM structure for the test set. The average is taken over 30 trials. The test set is regenerated randomly in each trial with same statistical properties.

Number Of Neuron	Sensitivity	Specificity	Accuracy	Precision
4 Neuron	0.359±0.0879	0.932±0.0809	5.6431±0.847	0.8722±0.143
6 Neuron	0.218±0.042	0.969± 0.030	5.8750±0.628	0.8889± 0.11
8 Neuron	0.1513±0.025	0.9878±0.016	6.2278±0.525	0.9278±0.095
10 Neuron	0.113±0.017	0.984±0.016	5.9994± 0.691	0.8833±0.117

Table 3.3 illustrates the all the evaluation results for the test sets in which it can be seen that all the outcomes are approximately close to the training set outcomes.

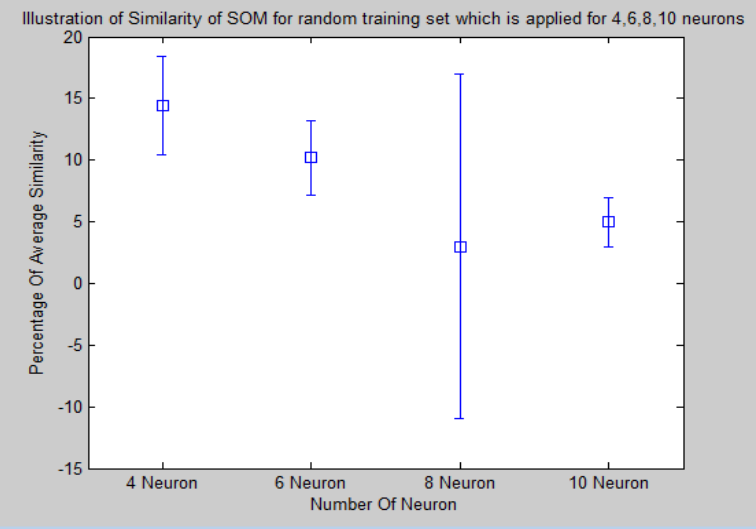


Figure 3.11 : Illustration of average similarity outcomes of SOM for random dataset over 30 trails.

Figure 3.11 denotes the average percentage similarity of the training set after the 30 trails. It is clear from the table that the for the 4 neuron the similarity, which shows the proportion of the similarity of the neurons to the input sample, is almost near 15% but as the number of neuron increase the average converge to zero instead of the

10 neuron. It means that by increasing the number of neuron at the end of the training neuron become more similar to the input samples.

The averge number of steps to conclude the formation of clustering in 30 randomly generated training sets is about five. The main of this thesis is to improve this.

3.4.2 Iris data set

This famous (Fisher's or Anderson's) Iris data set consists of 50 samples from each of three species of Iris which are Iris setosa, versicolor, and virginica. Four features were measured from each sample: the length and the width of the sepals and petals, in centimetres. So the input data set is a four dimensional set and based on the combination of these four features, we would try to cluster them.

3.4.2.1 Simulation results

Now we will try to cluster these points using SOM structure. Table 3.4 shows the parameter values that have been used in this example. Where k parameter is the number of training data ponits, which is 120 in this example, where 40 points belongs to class one, 40 belongs to class two and last 40 belong to class three and for test data set we have 10 points from each class so we have 30 points in the test set. $\boldsymbol{l}$ is the number of neurons that is used. For this example, the problem is solved by 4, 6, 8 and 10 neurons. Here m is the dimension of data set, which is 4 in this example, η_0 is the initial value of SOM's learninig rate and σ_0 is the initial value of the effective width of topological neighborhood.

Table 3.4: List of parameters and their values

Parameter	η_0	σ_0	η	m	l	k
Parameter value	1	3	2.5	4	4,6,8,10	Training : 120 Test : 30

The results obtained by applying SOM to the Iris data set are provided in Table 3.5 and 3.6. In these tables the neurons are generated randomly. Since, Iris data set is a 4-dimensional set so we can not plot it like previous example. As metioned before at first SOM is applied for 4 neurons then 6, 8 and 10 neurons. The values defining the neuron'sweighs are initialized randomly.

Table 3.5 : Results of SOM structure for the Iris training set over the 30 trials. The neurons are regenerated randomly in each trial with same statistical properties.

Number Of Neuron	Sensitivity	Specificity	Accuracy	Precision
4 Neuron	0.2265±0	0.9415±0	99.7042±0	0.8250±0
6 Neuron	0.1537± 0.003	0.9746±0.0047	105.736±2.77	0.8744±0.023
8 Neuron	0.128±0.0015	0.9869±0	110.783± 0.01	0.9167±0
10 Neuron	0.087±0.0027	0.9817±0.0032	101.21±3.457	0.8361±0.029

Table 3.5 denotes all the outcomes of SOM structure after 30 trails. At the first glance it is clear that as the number of neuron increase Sensitivity decrease while Accuracy, Specificity and Precision approximately increase.

Sensitivity, which shows the proportion of the data that are classified correctly, for 4 neuron is about 22% and the Specificity of that is almost 94%. The Accuracy of Iris data set is just over the 99. It means that 99 points out of 120 number of data set clustered correctly and the Precision of the SOM for Iris data set is about 82% so as in the random data set, SOM is a structure with good accuracy and good Precision so the results almost are same as the results of random data points.

Table 3.6 : Results of SOM structure for the Iris test set over the 30 trials. The neurons are regenerated randomly in each trial with same statisticalproperties.

Number Of Neuron	Sensitivity	Specificity	Accuracy	Precision
4 Neuron	0.8654±0	0.8654±0	23.3750±0	0.7667±0
6 Neuron	0.9807±0.002	0.9807±0.002	28.5715±0.0595	0.9333±0
8 Neuron	0.9551±0.007	0.9551±0.007	24.1354± 1.028	0.786± 0.035
10 Neuron	0.9934±0.005	0.9934±0.005	29.4064±0.7526	0.959±0.023

Table 3.6 shows the all the evaluation results for the Iris test sets over the 30 trails in which it can be seen that all the outcomes are approximately close to the training set outcomes.

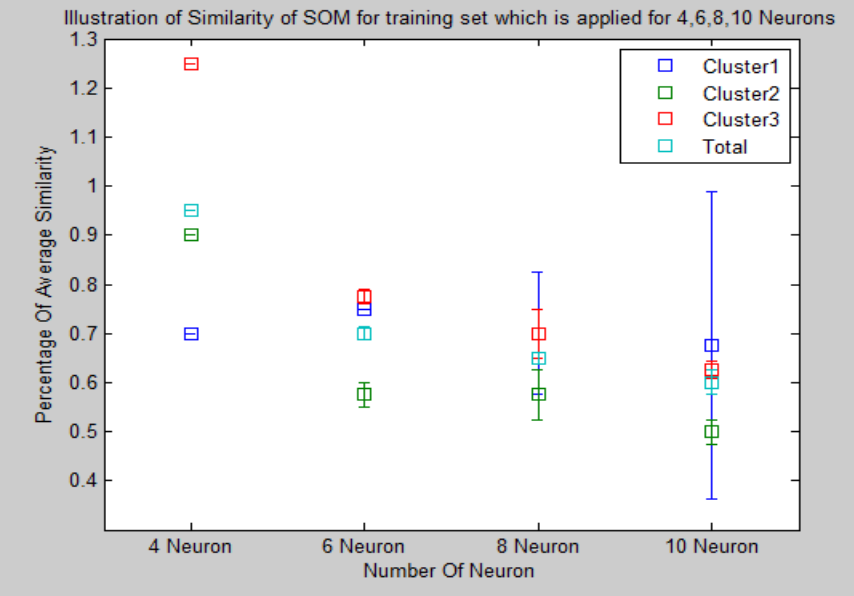


Figure 3.12 : The average similarity results of SOM for Iris data set over 30 trails. Initial weight of neurons aregenereted randomly.

Figure 3.12 shows the average similarity of Iris data clusters separately. According to the figure as the number of neuron, increase the total average similarity decrease it means that the neurons became more similar to training set.

As it mention in Chapter 3.2.1, initial samples of the input data set can be used for weight vector initialization. In Table A.1 and A.2 the results are obtained when initial weight of neurons are selected from training set. Just three neurons are considered because of the three different classes that we have in the training set.

Table 3.7 : The average number of convergence steps obtained for 30 trials. For each trial, the initial weight of neurons is generated randomly.

SOM	4 neuron	6neuron	8neuron	10neuron
Average Number of steps	7±0	12.36±3.01	6.5±0.5	8.1±1.30

In the Table 3.7, average number of convergence steps to conclude the formation of clustering is considered. It is clear from the table that convergence in Iris data set does not depend on the number of neuron.

4. REINFORCEMENT LEARNING

Reinforcement Learning (RL) is an approach in machine intelligence to solve many problems like flight control systems for aircraft, automated manufacturing systems, robot learning to dock on battery charger, playing backgammon and poker in which the leaner learns an approximately optimal policy through trial and error interactions with the environment. Historically, the earliest use of a RL method was in Samuel's Checkers player program [12-13]. In RL, there is no teacher to train the learner, instead experience is the only teacher and the goal of the learner is to find out adequate actions to control the process. In this method, as there is no teacher the learner is not told which actions to take, as in most forms of machine learning, but instead must discover which actions yield the best result by trying them. The best result gives a reward and in the most interesting and challenging cases, actions may affect not only the immediate reward but also the next situation and, through that, all subsequent rewards [10].

Reinforcement learning is different from supervised learning, where the learning progress with examples provided by a knowledgeable external supervisor in other words, it requires sample input-output pairs from the function to be learned. Unlike reinforcement learning, supervised learning is not adequate for learning from trial and error interactions.

One of the challenges that arise in reinforcement learning and not in other kinds of learning is the trade-off between exploration and exploitation. To obtain a lot of reward, in reinforcement learning learner must prefer actions that it has tried in the past and found to be effective in producing reward. However, to discover such actions, it has to try actions that it has not selected before. The learner has to exploit what it already knows in order to obtain reward, but it also has to explore in order to make better action selections in the future [11].

4.1 The Components of Reinforcement Learning

The standard RL model is intended to operate in a learning environment composed by two subjects: The learning agent (learner) and a dynamic process. At each discrete time, the agent sensing the environment, selects an action and applies it back to the process. The action changes the environment in some manner and this change is communicated to the agent through a scalar reinforcement signal, indications about how well it is performing the required task. These signals are usually associated to some dramatic condition accomplishment of a subtask (reward) or complete failure (punishment), and the learner's goal is to optimize its behavior based on some performance measure (usually minimization of a cost function). The crucial point is that in order to do that, in the RL framework the learning agent must learn the conditions (associations between observed states and chosen actions) that lead to rewards or punishments. In other words, it must learn how to assign credit to past actions and states by correctly estimating costs associated to these events [10].

As a result beyond the agent and the environment, one can identify four main sub elements of a reinforcement learning system: a policy, a reward (reinforcement) function, a value function and optionally a model of the environment.

A *policy* defines the learning agent's way of behaving at a given time. The policy is the core of reinforcement learning in the sense that it alone is sufficient to determine behavior. In general, policies may be stochastic.

A *reward function* defines the goal in reinforcement learning problem. A reinforcement learning agent's goal is to learn to perform actions that will maximize the total reward it receives in the long run. The reward function defines what the good and bad events are for the agent. A system designer must be selecting a proper reward function through to the goal of the system.

The value function is a mapping from states to state values and can be approximated using any type of function approximator. The value of a state is defined as the sum of the reinforcements received when starting in that state and following some fixed policy to a terminal state. Whereas a reward function indicates what is good in an immediate sense, a value function specifies what is good in the long run. Rewards are in a sense primary, whereas values, as predictions of rewards, are secondary. Without rewards, there could be no values, and the only purpose of estimating values is to achieve more reward. Action choices are made based on value judgments. We

seek actions that bring about states of highest value, not highest reward, because these actions obtain the greatest amount of reward for us over the long run. Unfortunately, it is much harder to determine values than it is to determine rewards. Rewards are basically given directly by the environment, but values must be estimated and re estimated from the sequences of observations an agent makes over its entire lifetime. In fact, the most important component of almost all reinforcement learning algorithms is a method for efficiently estimating values.

The fourth and final element of some reinforcement learning systems is a model of the environment. For example, given a state and action, the model might predict the resultant next state and next reward [11].

Figure 3.1 shows how the RL agent interacts with the process. The agent observes the states of the system through its sensors and chooses actions based on its experience and according to the action that it has been choosed it gives a reward or punishment signal. In practical terms, this means that the agent's sensors must be good enough to produce correct and unambiguous observations of the process states.

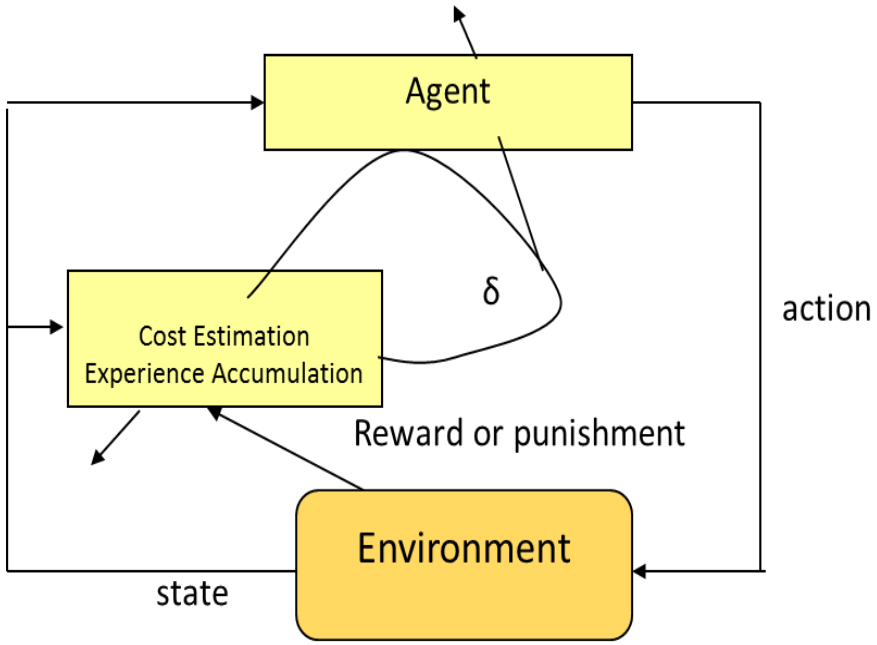


Figure 4.1 : Interaction between agent and dynamic process in Reinforcement learning.

5. IMPROVING SOM TRAINING PROCESS

Here in this Chapter we defined the three structure which are prooposed to improve the SOM.

5.1 Adapting Reinforcement Learning to SOM

In this thesis, the training phase of SOM will be modified to include RL approach. The motivation is to use the concept of award in RL during the training of SOM. The winning of a neuron for each point in the data set is considered as a behavior which changes the environment. This behavior can be evaluated and based on this evaluation it can be decided whether the weight of the winning neuron should be changed or not. This proposed structure named as RL-SOM.

With this proposed adaptation, the only change will be in the process of weight updating in SOM algorithm, which is given in Chapter 2.3. At this point, the similarity of the winning neuron for the kth data will be considered and if this similarity is less than the similarity determined in the previous iteration winning neuron for the kth data, weight of the winning neuron would not be updated, else it would be updated. Therefore, the selected neuron in SOM structure is evaluated and if its similarity is not better than the neuron which is determined in the previous iteration step, its weight will not be updated as a punishment, but if it is more similar than the previous winner, as a reward its weight will be updated.

As it is mentioned above the only change in the algorithm, will be in the process of updating in SOM algorithm.

Updatingthe Weight of the Winning Neuron and its Neighbors : According to the determined neighborhood function $h_{j,i(x)}$, the weight of the winner neuron and its neighbor's weight will be updated, as the weight of the winning neuron is more. This updating equation is given as below:

$$W_j(n+1) = W_j\,(n) + \delta^{reward}(n)\eta(n)h_{j,i(x)}\,(n)(x(n) - W_j(n) \qquad (\mathbf{5.1})$$

Where δ^{reward} parameter in equation (4.1) defined as below:

$$\delta^{reward}(n) = \begin{cases} 1 & Dist_{j,i}(n) \leq Dist_{j,i(}(n-1) \\ 0 & Dist_{j,i}(n) > Dist_{j,i}(n-1) \end{cases} \quad (\mathbf{5.2})$$

5.1.1 Simulation results

As it is mentioned previously, in this thesis all the proposed methods is applied to the simple clustering problems. As in the previous Chapter on simulation results firstly, RL-SOM is applied to a set of random data points then it is applied to Iris data set.

5.1.1.1 Random data points

The results obtained when RL-SOM is applied to the data set given in Figure 3.8 are provided in Figure 5.1. In this application the same parameters value are used which are in Table 3.1. As it can be followed from the Figure 5.1, there are eight incorrect points in this figure. It can be seen easily from the figure that two points in cluster 1, two points in cluster 2, three points in cluster 3 and one point in cluster 4 classified wrongly. Although the overall performance of RL-SOM is poor, if the results of SOM and RL-SOM checked carefully it can be seen that RL-SOM can allocate the close data of separate class correctly in different clusters which SOM couldn't distinguish.

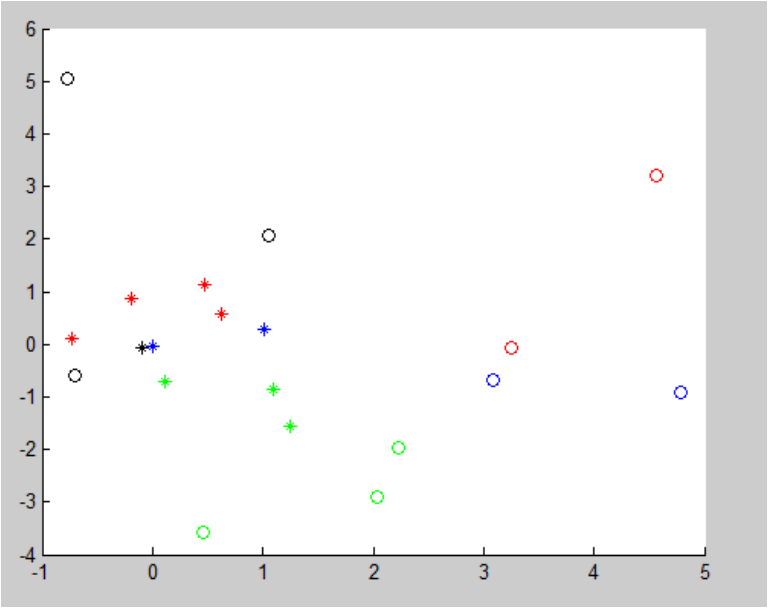


Figure 5.1 : Results of RL-SOM structure for the data set given in Figure 3.8

In order to have statistically significant conclusion, the related m-files are executed in two different ways as in Chapter 3.4.1.1. In Table 5.1 and 5.2 the average results

of 30 trials is taken over the random training and test sets and the results for the same type of random data set are given in appendix.

Table 5.1 : Results of RL-SOM structure for the training set. The average is taken over 30 trials. The training set is regenerated randomly in each trial with same statistical properties.

Number of neuron	Sensitivity	Specificity	Accuracy	Precision
4 Neuron	0.2196±0.0347	0.8861± 0.036	14.3883±1.566	0.6883±0.0773
6 Neuron	0.1368±0.0135	0.9395±0.0161	14.9842±1.4222	0.7117±0.070
8 Neuron	0.1061±0.0108	0.9591±0.0101	15.4756±1.3382	0.7350±0.0671
10 Neuron	0.0813±0.0078	0.9695±0.0092	15.6012±1.5770	0.7383± 0.078

Table 5.1 gives the all results of RL-SOM structure after 30 trails in which Sensitivity of RL-SOM for 4 neuron is about 22% which is approximately same as the SOM structure. Specificity of RL-SOM is about 88% which is less than the Specificity of the SOM structure. It means that the number of the points that are incorrectly classified out of a class is more than the SOM structure. It can be seen from the table that the accuracy and precision of the RL-SOM are also worse than the SOM. However, it is clear that except the sensitivity as the number of neuron goes up the results become better.

Table 5.2 : Results of RL-SOM structure for the test set. The average is taken over 30 trials. The test set is regenerated randomly in each trial with same statistical properties.

Number Of Neuron	Sensitivity	Specificity	Accuracy	Precision
4 Neuron	0.3299±0.1014	0.8926±0.0684	5.2028±0.7739	0.7944±0.1362
6 Neuron	0.1798±0.0345	0.9565±0.0344	5.5648±0.8535	0.8222±0.1447
8 Neuron	0.1304±0.0280	0.9738±0.0208	5.7444±0.7705	0.8389± 0.135
10 Neuron	0.1053±0.0177	0.9838±0.0143	5.9811±0.6572	0.8722±0.1132

Table 5.2 denotes the all the evaluation results for the test sets in which it can be seen that all the outcomes are approximately close to the training set outcomes.

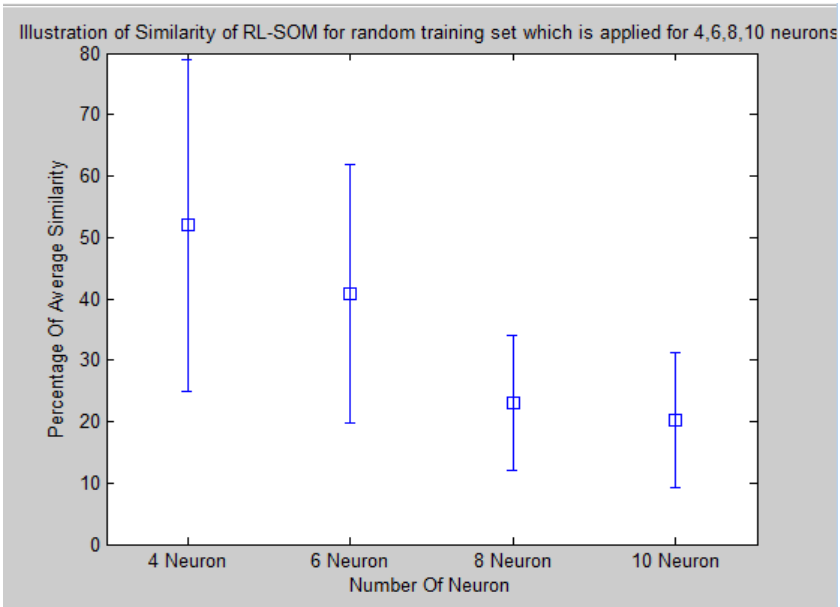


Figure 5.2 : The average similarity results of RL-SOM for random data set over 30 trails.

Figure 5.2 illustrates the average percentage similarity of the training set after the 30 trails. It is clear from the table that the for the 4 neuron the similarity, which shows the proportion of the similarity of the neurons to the input sample, is justover 50% but as the number of neuron increase the average converge to zero it means that by increasing the number of neuron at the end of the training neuron become more similar to the input samples but all the same, these values are worse than the SOM values.

The average number of steps to conclude the formation of clustering in 30 randomly generated training sets is about three which is less than the value which is obtained by SOM. It means that by mixing RL and SOM we could improve convergence performance of SOM.

5.1.1.2 Iris data points

Now we will try to cluster Iris data points by using RL-SOM structure. In this example the same parameters value has been used which are given in Table.2. For this example as in the previous example, the problem is solved with 4, 6, 8 and 10 neurons.

The results obtained by applying RL-SOM to the Irisdata set are provided in Table 5.3 and 5.4.

Table 5.3 : Results of RL-SOM structure for the Iris training set over the 30 trials. The neurons are regenerated randomly in each trial with same statistical properties.

Number Of Neuron	Sensitivity	Specificity	Accuracy	Precision
4 Neuron	0.1912± 0.0194	0.8570±0.0230	70.9177±8.946	0.5858±0.0745
6 Neuron	0.1218±0.0120	0.9311±0.0137	79.0154±8.818	0.6519±0.0735
8 Neuron	0.0919±0.0069	0.9470±0.0046	77.9957±4.588	0.6433± 0.0384
10 Neuron	0.0708± 0.005	0.9622±0.0065	80.1021±6.992	0.6603±0.0582

Table 5.3 denotes all the outcomes of SOM structure after 30 trails. At the first glance it is clear that as the number of neuron increase, as in SOM, Sensitivity decrease but the other evaluation measures incresae and these values, in contrast to SOM's values, are more better.

Table 5.4 : Results of RL- SOM structure for the Iris test set over the 30 trials. The neurons are regenerated randomly in each trial with same statistical properties.

Number Of Neuron	Sensitivity	Specificity	Accuracy	Precision
4 Neuron	0.8703±0.0311	0.8703± 0.031	23.6922± 2.21	0.7778± 0.075
6 Neuron	0.9964±0.0074	0.9964±0.0074	30.1502± 0.79	0.9867± 0.027
8 Neuron	0.9861±0.0138	0.9861± 0.014	28.4472± 2.19	0.9278±0.0738
10 Neuron	0.9773±0.0164	0.9773± 0.016	26.2788±3.074	0.8556±0.1026

Table 5.4 shows the all the evaluation results for the Iris test sets over the 30 trails.

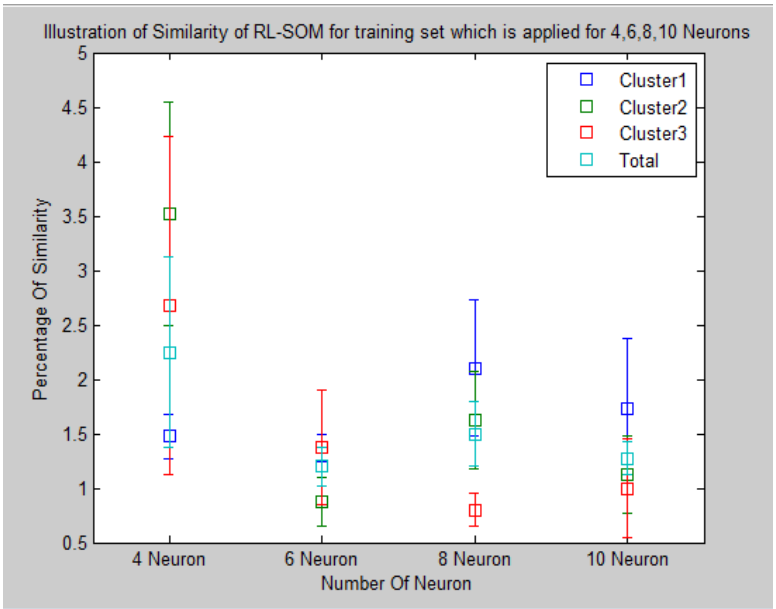


Figure 5.3 : The average similarity results of RL-SOM for Iris data set over 30 trails. Initial weights of neurons are generated randomly.

Figure 5.3 shows the average similarity of Iris data clusters separately. According to the figure as the number of neuron increase the total average similarity approximately decrease it means that the neurons became more similar to training set but to compare with SOM the results are not better.

In Table 5.5 the convergence speed of the method is given.

Table 5.5 : The average number of convergence steps obtained for 30 trials. For each trial, the initial weights of neurons are generated randomly.

RL-SOM	4neuron	6neuron	8neuron	10neuron
Average number of steps	8.53±1.1	10.07±1.2	9.23±2.19	10.07±1.36

5.2 Adapting Class Information to SOM

Learning Vector Quantization (LVQ) is one of the structure which again proposed by T.Kohonen. The Difference of LVQ from SOM is in learning method where during training class information is used however neighboring relation is not taken into account and only the winning neuron's weight is updated. During this updating if the winning neuron’s label same as the input data’s label in that case the neuron's weight is increased else the weight is reduced. For example, a data set labeled with three

classes 3rd, 5th, 7th neurons can have the first neuron class label, 1st, 2nd, 4th neurons can have the second neuron class label, 6th and 8th neurons can have the third neuron class label, etc..

In this case thesis the class information for the data set will be considered during the training of SOM and the proposed algorithm obtained by this change will be named as Supervised Self-Organizing Map (S-SOM).

The algorithm steps of this new proposed structure will be remained the same as the algorithm of the SOM structure which is given in 2.2.1 Chapter, except the "determination of the initial values" and "updating the weight of the winning neuron and its neighbors".

The First Assignment of Values : Weight vectors are randomly assigned. Thus,initial value of the l number of neuron will be determined. This l number of neuron will be labled with r number of classes.

Updating the Weight of the Winning Neuron and its Neighbors : According to the determined neighborhood function $h_{j,i(x)}(n)$ the weight of the winner neuron an ditsneighbor's weight will be updated. This updating equation is given as below :

$$W_j(n+1) = W_j(n) + \delta^{class}(n)\eta(n)h_{j,i(x)}(n)\left(x(n) - W_j(n)\right) \qquad (\mathbf{5.3})$$

Where δ^{class} parameter in equation (5.1) defined as below:

$$\delta^{reward}(n) = \begin{cases} 1 & if\ nth\ input\ data's\ label\ is\ the\ same\ as \\ & the\ winning\ neuron' label \\ 0 & if\ nth\ input\ data's\ label\ is\ not\ the\ same\ as \\ & the\ winning\ neuron' label \end{cases} \qquad \mathbf{(5.4)}$$

5.2.1 Simulation results

5.2.1.1 Random data points

By applying S-SOM through the data set which is given in Figure 3.8 at the end of the training the information related to each cluster and each class are provided in .

As it can be followed from the Figure 5.4 that the clusters composed without any error. Thus S-SOM structure can solve the problem of SOM which cannot distinguish the close data that are not belonging to a same class.

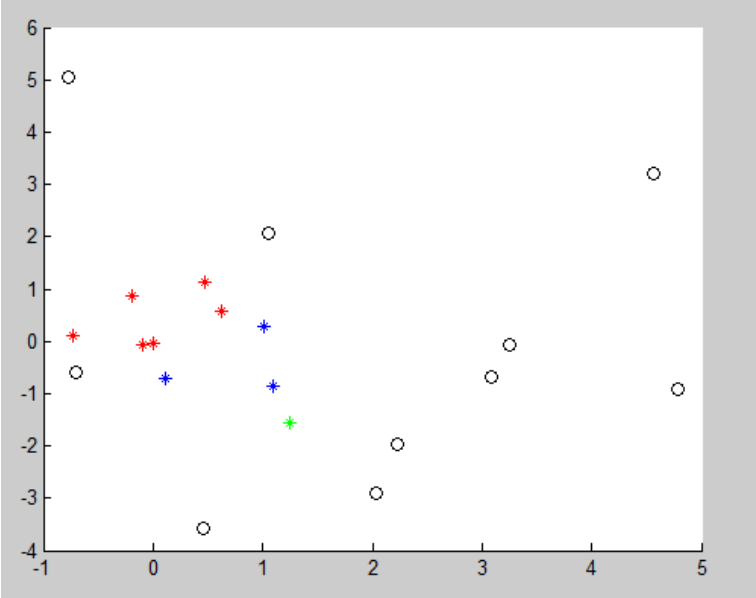


Figure 5.4 : Results of S-SOM structure for the data set given in Figure 3.8

In order to have statistically significant conclusion, the related m-files are executed for the different training sets, which are produced randomly for eachtrial. 30 trials are considered and the performance of S-SOM is given in Tables 5.6. and 5.7. Table 5.6., is giving the results for 30 different trials where the training and test sets select randomly in each trail. In this table at first S-SOM applied to 4 random neurons then 6, 8 and 10 random neurons.

Table 5.6 : Results of S-SOM structure for the training set. The average is taken over 30 trials. The test set is regenerated randomly in each trial with same statistical properties.

Number Of Neuron	Sensitivity	Specificity	Accuracy	Precision
4 Neuron	0.2551±0.0409	0.8948±0.0414	15.5275±1.5876	0.7483±0.0771
6 Neuron	0.1656± 0.014	0.9536±0.0243	16.8858±1.7802	0.8100±0.0855
8 Neuron	0.1146±0.0097	0.9708±0.0122	17.0550±1.4890	0.8133±0.0730
10 Neuron	0.0925± 0.007	0.9818± 0.009	17.8003± 1.484	0.8483±0.0725

Table 5.6 gives the all results of S-SOM structure after 30 trails in which Sensitivity of S-SOM for 4 neuron is about 25% which is approximately better than both SOM and RL-SOM structures. Specificity of S-SOM is about 89% which is less than the Specificity of the SOM structure but not less than RL-SOM one. It means that the number of the points that are incorrectly classified out of a class is more than the SOM structure and is less than the RL-SOM. It can be seen from the table that the accuracy and precision of the RL-SOM are also worse than the SOM. However it is

clear that except the Sensitivity as the number of neuron goes up the results become better.

Table 5.7 : Results of S-SOM structure for the test set. The average is taken over 30 trials. The test set is regenerated randomly in each trial with same statistical properties.

Number Of Neuron	Sensitivity	Specificity	Accuracy	Precision
4 Neuron	0.3095±0.0741	0.9073±0.0635	5.3417±0.7323	0.8111±0.8111
6 Neuron	0.2052± 0.039	0.9615±0.0311	5.7352±0.6395	0.8611±0.1080
8 Neuron	0.1448±0.0249	0.9799± 0.023	5.9972±0.7572	0.8889±0.1263
10 Neuron	0.1141± 0.014	0.9906±0.0110	6.2911±0.4801	0.9278± 0.084

Table 5.4 shows the all the evaluation results for the Iris test sets over the 30 trails.

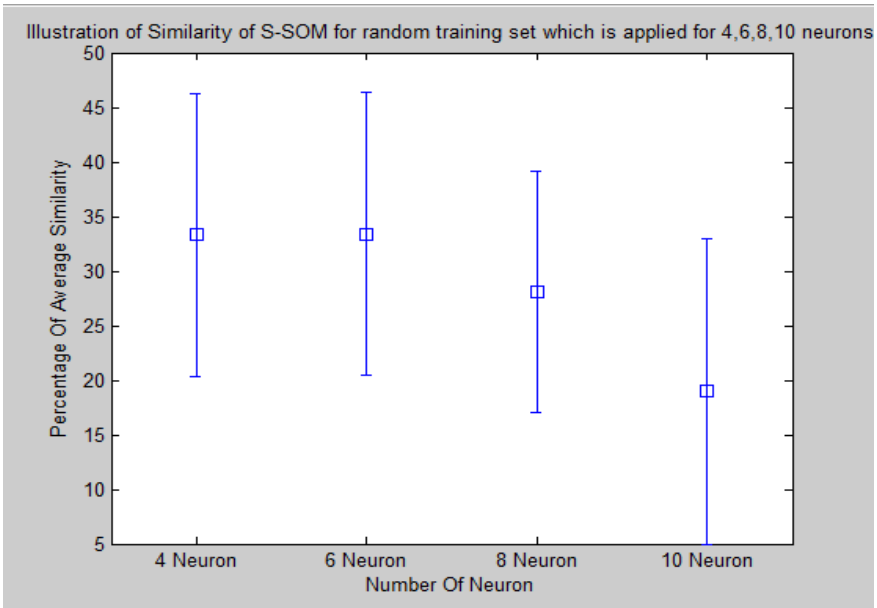


Figure 5.5 : The average similarity results of S-SOM for random data set over 30 trail.

Figure 5.5 illustrates the average percentage similarity of the training set after the 30 trails. It is clear from the table that the for the 4 neuron the similarity, which shows the proportion of the similarity of the neurons to the input sample, is justbelow 35% while as the number of neuron increase the average converge to zero. It means that by increasing the number of neuron at the end of the training neuron become more similar to the input samples but all the same, these values are worse than the SOM values and are better than RL-SOM.

The average number of convergence steps obtained for S-SOM over 30 trials is about 5. For each trial the training set is generated randomly.

5.2.1.2 Iris data points

Now we will try to cluster Iris data points by using S-SOM structure. In this example the same parameters value has been used as in Table 3.4. For this example as the previous example, the problem is solved using 4, 6, 8 and 10 neurons.

The results obtained by applying S-SOM to the Iris data set are provided in Table 5.8 and 5.9. They are giving the average pecentage of evaluation mesuares for 30-times training of data sets and test sets for 4,6,8 and 10 neurons.

Table 5.8 : Results of S-SOM structure for the Iris training set over the 30 trials. The neurons are regenerated randomly in each trial with same statistical properties.

Number Of Neuron	Sensitivity	Specificity	Accuracy	Precision
4 Neuron	0.2465±0.034	0.7670± 0.061	63.4790±9.576	0.526±0.079
6 Neuron	0.1246±0.019	0.9142± 0.017	72.9464±9.197	0.602±0.077
8 Neuron	0.1057±0.012	0.9431±0.0132	80.4083±7.621	0.664±0.063
10 Neuron	0.0770±0.009	0.9637±0.0108	83.723 ±9.927	0.691±0.083

Table 5.8 denotes all the outcomes of S-SOM structure after 30 trails. At the first glance it is clear that as the number of neuron increase Sensitivity decrease while accuracy, Specificity and precision approximately increase.

Sensitivityfor the 4 neuron is about 24% and the Specificity of that is almost 76%. The accuracy of Iris data set is just over the 63. It means that 69 points out of 120 number of data set clustered correctly and the precision of the SOM for Iris data set is about 52% .

Again, as in previous examples the evaluation measures are given considering the all of training process, Acoording to Table 5.9 the results for test set are better since the training phase is complete and the clusters are determined.

Table 5.9 : Results of S-SOM structure for the Iris test set over the 30 trials. The neurons are regenerated randomly in each trial with same statistical properties.

Number Of Neuron	Sensitivity	Specificity	Accuracy	Precision
4 Neuron	0.766±0.125	0.7665±0.125	20.047±5.649	0.658± 0.189
6 Neuron	0.902±0.025	0.902±0.025	20.766± 2.618	0.676±0.088
8 Neuron	0.958±0.044	0.958±0.044	23.995± 7.188	0.780± 0.240
10 Neuron	0.976±0.019	0.976 ± 0.02	25.8153± 4.19	0.839 ±0.14

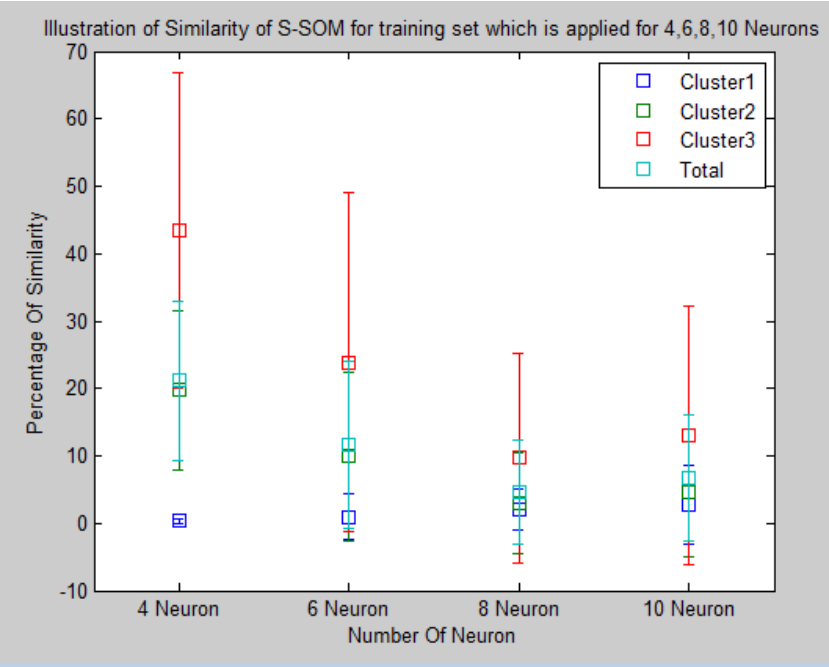


Figure 5.6 : The average similarity results of S-SOM for Iris data set over 30 trails. Initial weights of neurons are generated randomly.

Figure 5.6 shows the average similarity of Iris data clusters separately. According to figure as the number of neuron, increase the total average similarity decrease it means that the neurons became more similar to training set.

Table 5.10 : The average number of steps obtained for 30 trials. For each trial, the initial weights of neurons are generated randomly

S-SOM	4 Neuron	6 Neuron	8 Neuron	10 Neuron
Average number of steps	10.4±1.71	7.73±1.25	8.3±1.51	8.23±1.36

5.3 Adapting Reinforcement Learning and Class Information Together to SOM

This structure is the combination of two previous approaches, which are proposed for the first time in Chapter 4.1 and 4.2 of this thesis, and it named as RL-S-SOM. In this structure, the winning neuron and its neighbors are updated considering the class information and reinforcement learning together.

The only change will be appear in the process of weight updating in SOM algorithm.

Updating the Weight of the Winning Neuron and its Neighbors : According to the determined neighborhood function $h_{j,i(x)}(n)$ the weight of the winner neuron and its neighbor's weight will be updated as the weight of the winning neuron is more. This updating equation is given as below:

$$W_j(n+1) = W_j(n) + \delta^*(n)\eta(n)h_{j,i(x)}(n)\left(x(n) - W_j(n)\right) \quad \textbf{(5.5)}$$

Where δ^{class} parameter in equation (5.1) defined as below:

$$\delta^*(n) = \begin{cases} 1 & Dist_{j,i}(n) \leq Dist_{j,i(}(n-1)\delta^{class} = 1 \\ 0 & Dist_{j,i}(n) \leq Dist_{j,i(}(n-1)\delta^{class} = 0 \\ 0 & Dist_{j,i}(n) > Dist_{j,i(}(n-1) \end{cases} \quad \textbf{(5.6)}$$

5.3.1 Simulation results

5.3.1.1 Random data points

By applying RL-S-SOM to the data set which is given in Figure 3.8, the reults given in Figure 5.7 is obtained at the end of the training. The information related to each cluster and each class are provided in Figure 5.7. As it can be followed from the Figure 5.7 the clusters composed without any error. Thus RL-S-SOM structure can solve the problem of SOM which cannot distinguish the close data that do not belong to a same class.

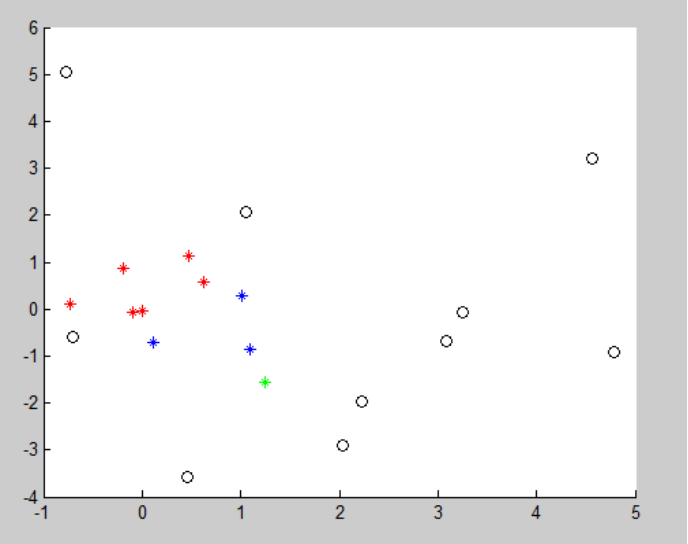


Figure 5.7 : Results of RL-S-SOM structure for the data set given in Figure 3.8

The results for RL-S-SOM is summarized in Table 5.11 and 5.12 and Increasing the number of neurons improves the results. Not only the number of incorrect data decreases, but also the similarity gets better.

Table 5.11 : Results of RL-S-SOM structure for the training set. The average is taken over 30 trials. The training set is regenerated randomly in each trial with same statistical properties.

Number Of Neuron	Sensitivity	Specificity	Accuracy	Precision
4 Neuron	0.2720±0.0407	0.8898±0.0510	15.6887±1.6851	0.7583±0.081
6 Neuron	0.1673±0.0149	0.9429±0.0186	16.0414± 1.428	0.770± 0.070
8 Neuron	0.1188±0.0062	0.9696±0.0146	17.0921±1.5787	0.8167±0.076
10 Neuron	0.0941±0.006	0.9828±0.0100	17.9930± 1.569	0.8583±0.077

Table 5.12 shows the all the evaluation results for the Iris test sets over the 30 trails.

Table 5.12 : Results of RL-S-SOM structure for the test set. The average is taken over 30 trials. The training set is regenerated randomly in each trial with same statistical properties.

Number Of Neuron	Sensitivity	Specificity	Accuracy	Precision
4 Neuron	0.3471±0.1055	0.8971± 0.089	5.2778±0.8339	0.811±0.143
6 Neuron	0.1946±0.0513	0.9608± 0.036	5.6306±0.8766	0.838±0.155
8 Neuron	0.1398±0.0284	0.9773±0.0211	5.8771±0.7321	0.8667±0.13
10 Neuron	0.1132±0.0178	0.9875±0.0149	6.1489±0.6633	0.9056±0.13

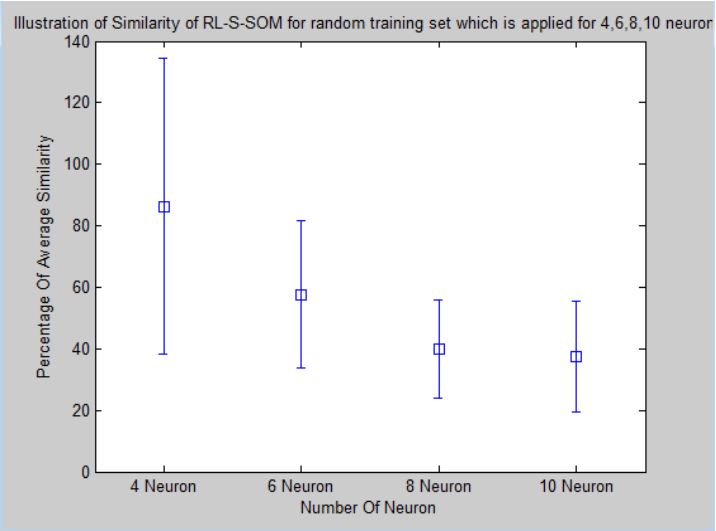


Figure 5.8 : The average similarity results of RL-S-SOM for random data set over 30 trails.

Figure 5.8 illustrates the average percentage similarity of the training set after the 30 trails. It is clear from the table that the for the 4 neuron the similarity, which shows the proportion of the similarity of the neurons to the input sample, is near the 85% but as the number of neuron increase the average converge to zero. It means that by increasing the number of neuron at the end of the training neuron become more similar to the input samples.

The average number of steps obtained for RL-S-SOM over the 30 trials is obout 4. For each trial, the training set is generated randomly.

5.3.1.2 Iris data points

Using RL-S-SOM, Iris data set is clustered. Again with increasing the number of neurons the results are improved. In order to see the effect of initial neuron values, not only randomly generated set are considered, but also the clustering with initial values taken from training set is considered. These results are summarized in Tables 5.13, 5.14 and Figure 5.9.

Table 5.13 : Results of RL-S-SOM structure for the Iris training set over the 30 trials. The neurons are regenerated randomly in each trial with same statistical properties.

Number Of Neuron	Sensitivity	Specificity	Accuracy	Precision
4 neuron	0.2206± 0.012	0.7488±0.0686	56.31± 10.453	0.4658±0.087
6 neuron	0.1447± 0.023	0.9149±0.0214	78.419±5.916	0.6481±0.049
8 neuron	0.1053±0.011	0.9420±0.0184	80.004±10.87	0.660±0.0902
10 neuron	0.0836±0.013	0.9618±0.0110	84.742± 7.401	0.6997±0.061

Table 5.13 denotes all the outcomes of RL-S-SOM structure after 30 trails. At the first glance it is clear that as the number of neuron increase Sensitivity decrease while accuracy, Specificity and precision approximately increase.

Sensitivity for the 4 neuron is about 22% which is almost the same as as the SOM and the Specificity of that is almost 74%. The accuracy of Iris data set is just over the 56. It means that 56 points out of 120 number of data set clustered correctly and the precision of the SOM for Iris data set is about 46% .

Table 5.14 : Results of RL-S-SOM structure for the Iris test set over the 30 trials. The neurons are regenerated randomly in each trial with same statistical properties.

Number of Neuron	Sensitivity	Specificity	Accuracy	Precision
4 neuron	0.751± 0.142	0.7514±0.1423	20.5644±3.4872	0.6756±0.114
6 neuron	0.928± 0.043	0.9278±0.0428	23.4956± 4.479	0.767± 0.151
8 neuron	0.961±0.034	0.9607±0.0336	24.5353±5.3962	0.7978±0.180
10 neuron	0.9736±0.02	0.9736± 0.020	25.3256± 3.977	0.8222±0.132

Table 5.14 shows the all the evaluation results for the Iris test sets over the 30 trails.

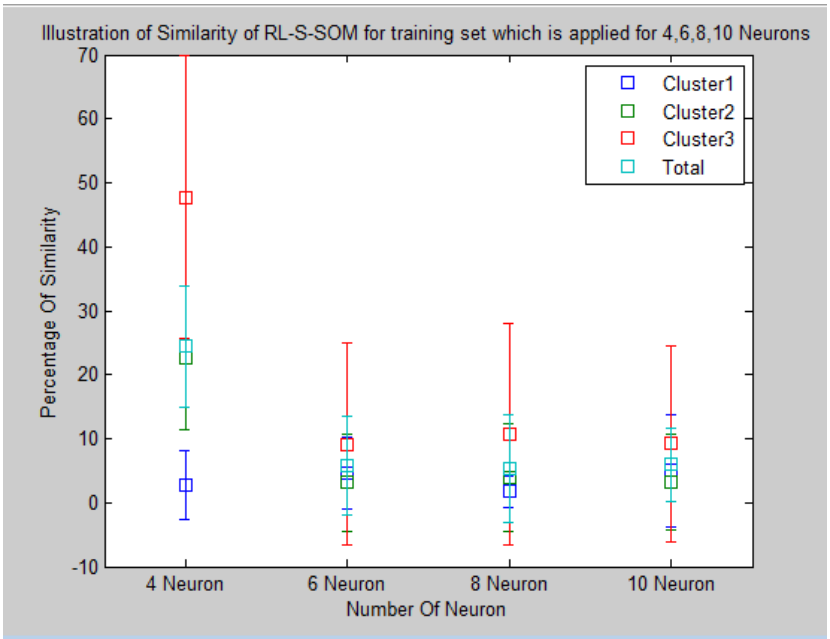


Figure 5.9 : The average similarity results of RL-S-SOM for Iris data set over 30 trails. Initial weights of neurons are generated randomly.

All these results reveal that the characteristic of class 1 causes clustering it easily, while it is not that easy to cluster class three as it is very similar to class 2.

6. CONCLUSION AND RECOMMENDATIONS

In this thesis, first SOM algorithm is given and then the new approaches in order to improve SOM's performanceare proposed in Chapter 4. These structures have been applied to simple problems which are Iris data set and random data points. All the simulation results are obtained by m-files prepared in MATLAB environment. The results are discussed after simulation results are given. In order to compare each of the four methods in random data points and to carry out this discussion in a statistically significant manner, the related m-files are executed for 30 timesand the results are summarized with the tables given here.

In Table 6.1 and 6.2, the converge properties of the methods are given for both problems considered throughout the thesis. As it can be followed from Table 6.1 the proposed methods converge faster for considering the 2-dimensional point data set while there is not much difference between methods for Iris data set, where the results are given in Table 6.2.

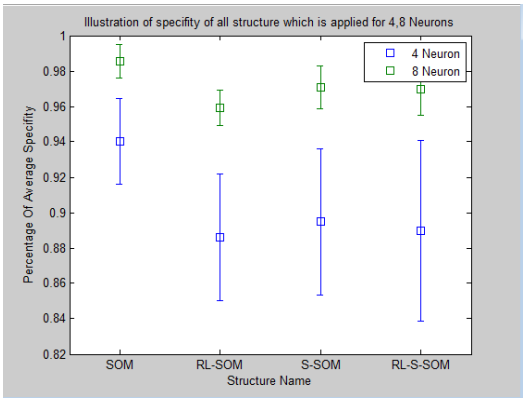
Table 6.1 : The average number of steps obtained for 30 trials. For each trial, the training set is generated randomly.

Method	SOM	S-SOM	RL-SOM	RL-S-SOM
Average Number of steps	5.43±0.9	4.6±1.0	3.63±0.8	3.76±0.8

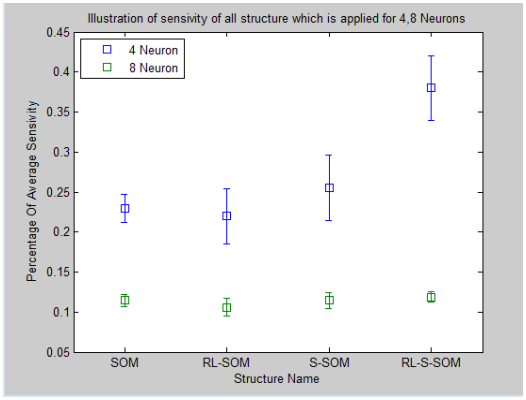
Table 6.2: The average number of steps for Iris data set.

Method / Average Step	SOM	RL-SOM	S-SOM	RL-S-SOM
4 Neuron	7.00±0	8.53±1.1	10.4±1.71	8.67±1.42
6 Neuron	12.36±3.01	10.07±1.2	7.73±1.25	8.9±2.10
8 Neuron	6.5±0.5	9.23±2.19	8.3±1.51	8.93±1.84
10 Neuron	8.1±1.30	10.07±1.36	8.23±1.36	9.07±1.93

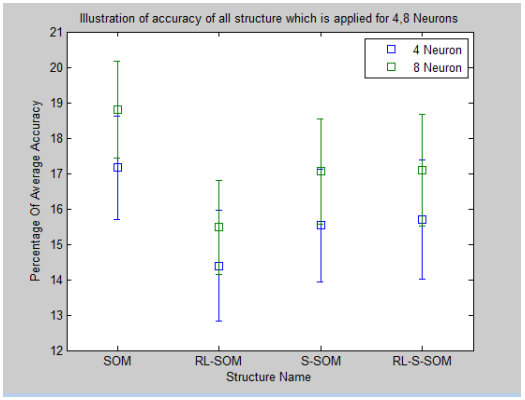
As it can be seen in Table 6.1 in RL-SOM structure, the cluster formations have been accelerated. After the formation of the clusters, in order to have a better represent action, the training phase is continued more. The clusters formed by continuing the training phase until 100 steps are over. This provided the grouping of data in the order of one-thousandth difference in the average distances of the winning neuron.



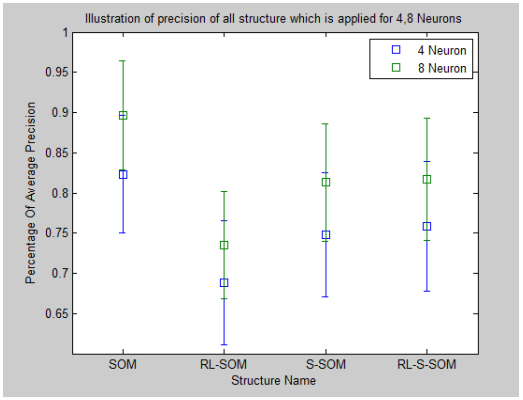
(a)



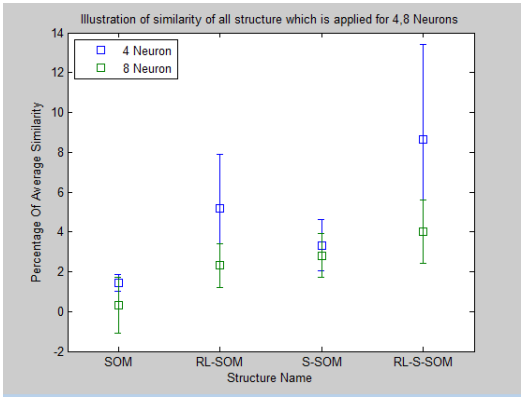
(b)



(c)



(d)

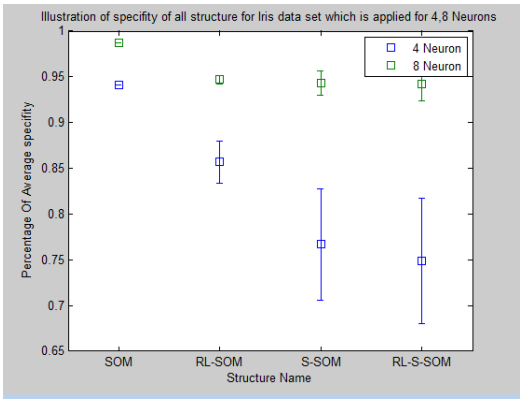


(e)

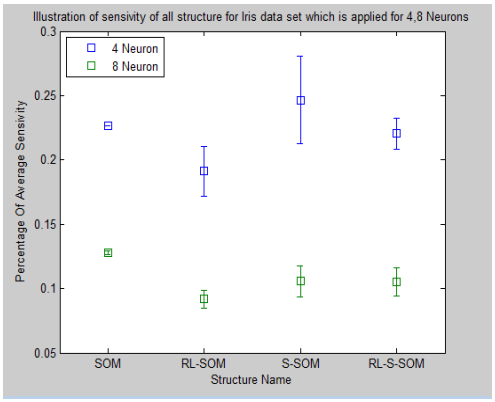
Figure 6.1 : Comparing the evaluation measures of four methods for random data set: (a)Specificity. (b)Sensitivity. (c)Accuracy. (d)Precision. (e)Similarity

According to the Figure 6.1 which summarizes the simulation results in Chapter 5.2, S-SOM structure gives the best result amongst the three methods that have been

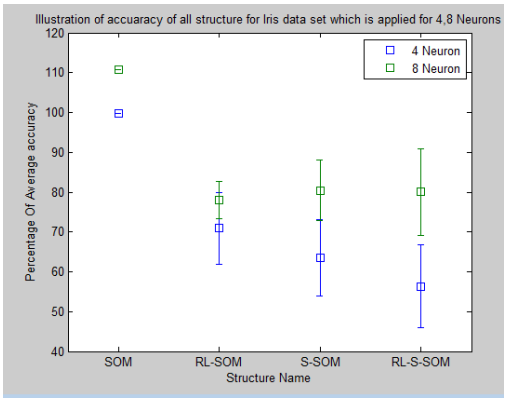
suggested. However, due to the reward mechanism of RL-SOM structure, which chooses the winner based on the criteria of choosing the best fit makes it possible to distinguish similar data in different classes.



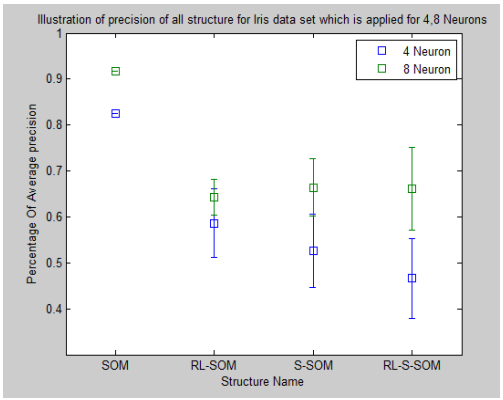
(a)



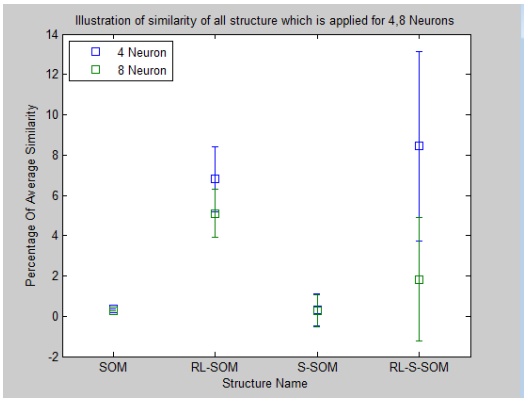
(b)



(c)



(d)



(e)

Figure 6.2 : Comparing the evaluation measures of four methods for Iris data set: (a)Specificity. (b)Sensitivity. (c)Accuracy. (d)Precision. (e)Similarity

Infigure 6.2 the results for Iris data set are summarized. As it can followed from these figures, implementing RL and LVQ in SOM did not help much in improving the results. While choosing the initial data from the training set improved the results still results are not better than SOM.

In this thesis, SOM is investigated and in order to improve its performance new training algorithms are proposed. SOM structure and the proposed algorithms are implemented in MATLAB environment and two problems are considered. Even though, the proposed methods provided some improvement in the data set of 2-dimensional points, the results for Iris data set were not satisfactory as expected.

Using in a way more knowledge in the training would help to improve the performance of already existing methods. One way could have different reinforcement learning methods implemented in conventional methods.

REFERENCES

[1] **Haykin, S.** (1999). Neural Network, A comprehensive Foundation.

[2] **Kohonen,T.** (1990). The self-organizing map. Proceedings of the IEEE, 9, 1464-1479.

[3] **Magomedov, B.** (2006). Self-Organizing Feature Maps (Kohonenmaps). http://www.codeproject.com/Articles/16273/Self-Organizing-Feature-Maps-Kohonen-maps, 7 November

[4] **Self organizing map.** *In Wikipedia.* Data received: 07.08.2012, address : http://en.wikipedia.org/wiki/Self-organizing_map

[5] **Hollmen, J.** (1996). Self-Organizing Map (SOM) http://users.ics.aalto.fi/jhollmen/dippa/node9.html .(Fri Mar 8 1996)

[6] **Germano, T.** (1999). Self-Organizing Maps http://davis.wpi.edu/~matt/courses/soms/.March 23

[7] **AI-Junkie.** (2007). SOM tutorial. http://www.ai-junkie.com/ann/som/som1.html. (Retrieved March 20)

[8] <www.di.unito.it/~cancelli/Dottorato_biologia/SOM.pdf>, date retrieved: 29.08.2012.

[9] **Guthikonda, S.** (2005). Kohonen Self-Organizing Maps. December

[10] **Ribeiro, C. H. C.** (1999). A Tutorial on Reinforcement Learning Techniques. *International Joint Conference on Neural Networks (IJCNN'99) - Tutorial Track on Learning, com o t´ituloA Tutorial on Reinforcement Learning Techniques.*

[11] **Sutton, R. A. and Barto, A. G.** (1998). Reinforcement Learning. A Bradford Book, the MIT Press Cambridge, Massachusetts London, England

[12] **Samuel, A. L. (**1959). Some studies in machine learning using the game of checkers. *IBM Journal on Research and Development, 3:210–229*

[13] **Mance, E. Harmon, Stephanie S. Harmon.** (1996). Reinforcement Learning: A Tutorial.

[14] **Precup, D.** (2005). Reinforcement Learning: A Brief Tutorial. McGill University, September.

[15] **Zhu, W., Zeng, N., Wang, N.** (2010). Sensitivity, Specificity, Accuracy, Associated Confidence Interval and ROCA analysis with Practical SAS. Implementations K&L consulting services, Inc, Fort Washington, PAO octagon Research Solutions, Wayne, PA.

[16] **Akobeng, A. K.** (2006). Understanding diagnostic tests 1: sensitivity, specificity and predictive Acta Pædiatrica ISSN 0803–5253 8 June, revised 4 December 2006; accepted 8 December 2006.

[17] **Sensitivity and Specificity.** *In Wikipedia.* Data received: 06.02.2013, address : http://en.wikipedia.org/wiki/Sensitivity_and_specificity.

[18] **Halkidi, M and Vazirgiannis, M.** (2001). A data set oriented approach for clustering algorithm selection. In Proceedings of the 5th European Conference on Principles of Data Mining and Knowledge Discovery (PKDD)

[19] **He, J. Tan, A. Tan, C. Sung, S.** (2002). On Quantitative Evaluation of Clustering Systems, Information Retrieval and Clustering W. Wu and H. Xiong (Eds.) pp. *Kluwer Academic Publishers.*

[20] **Tateyama, T. Kawata, S.Oguchi, T.** (2004). A teaching method using a self-organizing map for reinforcement learning, Artif. *Life Robotics*, Cilt no.7, sf.193-197.

[21] **Dozono, H. Fujiwara, R., ve Takahashi, T.** (2008). Visual Reinforcement Learning Algorithm using Self Organizing Map and Its Simulation in Open GL Environment, WSEAS Transactions on Information Science &Applications, Cilt no. 5/5, sf. 685-694.

[22] **Brohan, K., Gurney, K., Dudek, P.** (2010). Using Reinforcement Learning to Guide the Development of Self-organized Feature Maps for Visual Orienting. *International Conference on Artificial Neural Networks*, sf. 180-189.

[23] **Ram´on, V.Olivas, E. Guerrero, J. Montero, P. Jos´eM.** (2011). Analysis of a Reinforcement Learning algorithm using Self-Organizing Maps. *ESANN 2011 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*. Bruges (Belgium), 27-29 April 2011, i6doc.com publ., ISBN 978-2-87419-044-5.

[24] **Smith, A. J.** (2002) Applications of the Self-Organizing Map to Reinforcement Learning. Neural Networks - New developments in self-organizing maps. Volume 15 Issue 8-9, October 2002, Pages 1107 - 1124

[25] **Somervuo, P.** and **Kohenen, T.** (2004). Self Organizing Maps and Vector Quantization. Helsiki University of Technology,Nerural Network Research Center,P.O.Box 2200,FIN-02015-HUT,Finland

[26] **Kessler, W**. (2012) Evaluation of Text Classification,Institue of Maschinelle Sprachverarbeiutung

[27] **Jan, W. Owsiński and Mejza, M.** (2002). A Hybrid Clustering Method for General Purpose and Pattern Recognition. *Kluwer Academic Publishers.* Systems Research Institute, PolishAcademy of Sciences, Newelska 6, 01 447 Warsaw, Poland.

[28] **Dutech, A.** (2012). Self-organizing developmental reinforcement learning. LORIA – INRIA Campus Scientique - BP 23954506 Vandoeuvre les Nancy, France June 7.

[29] **He, J., Tan, A., Tan, C., Sung, S., Wu, W. and Xiong, H.** (2002). On Quantitative Evaluation of Clustering Systems. Information Retrieval and Clustering W. Wu and H. Xiong (Eds.) pp. 2002 Kluwer Academic Publishers

[30] **Batistakis, Y., Vazirgiannis, M.** (2001). On Clustering Validation Techniques. Journal of Intelligent Information Systems. Department of Informatics, Athens University of Economics & Business, Patision 76, 10434, Athens,Greece (Hellas).

[31] **Mac Queen, J.** (1967). Some Methods for Classification and Analysis of Multivariate Observations. In Proceedings of 5th Berkley Symposium on Mathematical Statistics and Probability, Volume I: Statistics, pp. 281–297.

[32] **Berkhin, P.** (2008). Survey of Clustering Data Mining Techniques. Accrue Software, 1045 Forest Knoll Dr., San Jose, CA, 95129

[33] **Kohonen, T., Ijain, A. K., Murty, M. N. and Flynnp, J. (**2001). Self-Organizing Maps. Springer Series.

[34] **Jain, A. k., Murty, M. N. and Flynn, P. J.** (1999). Data clustering: ACM Computing Surveys (CSUR) Volume 31 Issue 3, Sept. Pages 264-323

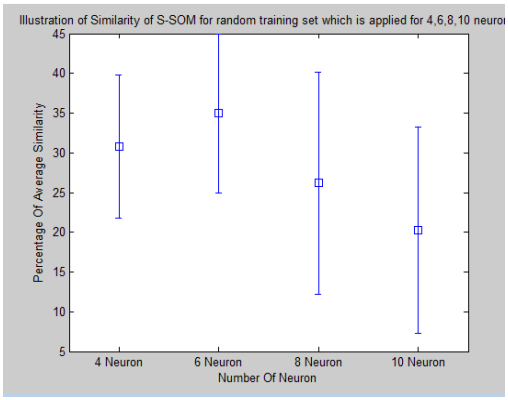
[35] **ZHANG, Y., FU, A.W., CAI, C.H., and Heng.** (2000). Clustering categorical data. In Proceedings of the 16th ICDE, 305, San Diego, CA.

[36] **Anitha, S., Elavarasi, Dr. Akilandeswari, J. and Sathiyabhama, B.** (2011). A survey on partition clustering algorithms. *International Journal of Enterprise Computing and Business Systems Vol. 1 Issue 1.*

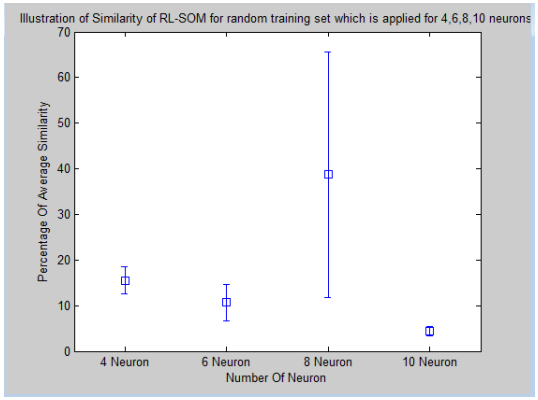
[37] **Carpenter, G.A. and Grossberg, S.** (2003). Adaptive Resonance Theory (http://cns.bu.edu/Profiles/Grossberg/CarGro2003HBTNN2.pdf).

[38] **Kohonen, T.** (1997). Self-Organization and Associative Memory. Springer-Verlag, Berlin, second edition.

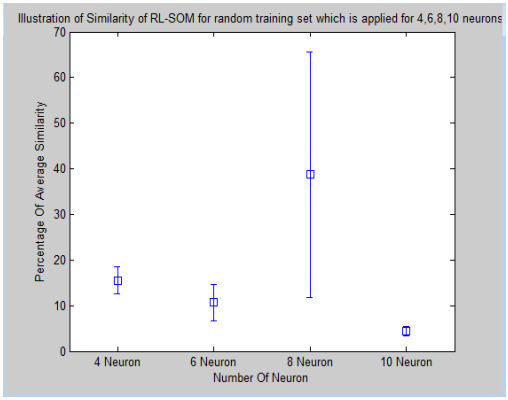
APPENDICES



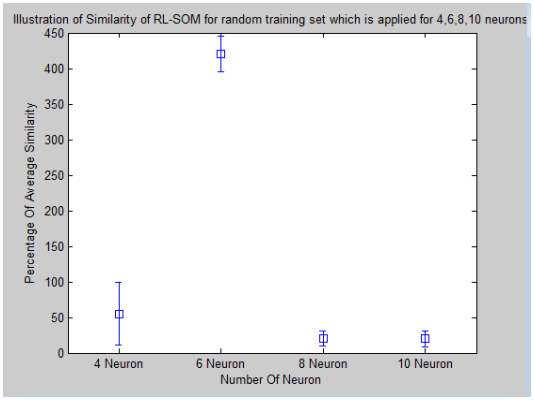
(a)



(b)



(c)



(e)

Figure A.1 : The average similarity results of the four structure for data set which are kept same over 30 trails.(a)SOM. (b)RL-SOM. (c)S-SOM. (d)RL-A-S-SOM.

Table A.5: The average similarity results of RL-SOM for Iris data set over 30 trails. Initial weights of neurons are selected from training set.

Number Of Neuron	AverageSimilarity			
	Class 1	Class 2	Class3	Total
3 neuron	0.35±0.03	1.2±0	1.18±0	0.91±0.008

Table A.6 : Results of S-SOM structure for the Iris training set over the 30 trials. Initial weights of neurons are selected from training set.

Number Of Neuron	Sensitivity	Specificity	Accuracy	Precision
3 neuron	0.4286±0	0.6197±0	66.2444±0	0.5500±0

Table A.7: Results of S-SOM structure for the Iris test set over the 30 trials. Initial weights of neurons are selected from training set.

Number Of Neuron	Sensitivity	Specificity	Accuracy	Precision
3 neuron	0.3750±0	0.3750±0	20.0667±0	0.6667±0

Table A.8: The average similarity results of S- SOM for Iris data set over 30 trails. Initial weights of neurons are selected from training set.

Number Of Neuron	AverageSimilarity			
	Class 1	Class 2	Class3	Total
3 neuron	0.10±0	10.91±0.0004	23.03±0.0006	11.35±0.0003

Table A.9: Results of RL-S-SOM structure for the Iris training set over the 30 trials. Initial weights of neurons are selected from training set.

Number Of Neuron	Sensitivity	Specificity	Accuracy	Precision
3 neuron	0.2667±0	0.8462±0	80.6111±0	0.6667±0

Table A.10: Results of RL-S-SOM structure for the Iris test set over the 30 trials. Initial weights of neurons are selected from training set.

Number Of Neuron	Sensitivity	Specificity	Accuracy	Precision
3 neuron	1±0	1±0	30.2222±0	1±0

Table A.11: The average similarity results of RL-S- SOM for Iris data set over 30 trails. Initial weights of neurons are selected from training set.

Number of Neuron	Average Similarity			
	Class 1	Class 2	Class 3	Total
3 neuron	0.26±0.17	9.83±4.14	20.46±7.86	10.18±3.94

Table A.12: The average number of convergence steps obtained for 30 trials. For each trial, the training set is generated randomly.

RL-S-SOM	4neuron	6neuron	8neuron	10neuron
Average number of step	8.67±1.42	8.9±2.10	8.93±1.84	9.07±1.93

CURRICULUM VITAE

Name Surname: Faranak Nouri

Place and Date of Birth: Urmia-Iran 1985

E-Mail: faranak.nouri1@gmail.com , fnouri@itu.edu.tr

B.Sc.: Islamic Azad University of Urmia

List of Publications and Patents:

PUBLICATIONS/PRESENTATIONS ON THE THESIS

- **Faranak Nouri.**, N.S. Sengor., 2012: Testing Different Approaches by Self Organizing Map. Electrical, Electronic and Computre Engineering Symposium, 29November-01December, 2012Bursa, Turkey