DETECTION AND CLASSIFICATION OF TUMER

Muneer Ahamed
Research scholar, Electronics & Communication Engineering Department, Mewar University, Rajasthan, India

Abstract

Medical image Classification will play a very important role in diagnostic and teaching functions in drugs. For these functions completely different imaging modalities are used. There are several classifications created for medical pictures victimization each grey-scale and color medical pictures. a method is to search out the feel of the photographs and have the analysis. Texture classification is a picture process technique by that completely different regions of a picture are known supported texture properties. Second manner is by victimization neural network classification techniques and therefore the final one is by victimization the information mining classification schemes. Neural networks play a significant role in classification, with the assistance of, supervised and unsupervised techniques. The word data processing refers to, extracting the data from giant amounts of information. It's one amongst the realm that uses applied math, machine learning, image and alternative information manipulation with data extraction techniques.

Index: *Artificial neural network, perceptron, Scale conjugate gradient algorithm, Support vector machine, Statistical learning theory.*

1. Introduction

Medical pictures kind a significant element of a patient's health record and are related to manipulation, process and handling of information by computers. This makes the premise for the computer-assisted radiology development. More developments are related to the utilization of call support systems that helps to come to a decision, the relevant data for designation

In this chapter two techniques for image classification are mentioned.

1.1 ARTIFICIAL NEURAL NETWORKS

The human brain processes, stores and retrieves info in a completely totally different thanks to the standard electronic computer. Although, a computer system} is a lot of correct and quicker than the brain once processing numerical knowledge, nevertheless in advanced machine tasks – like face recognition and alternative tasks involving pattern recognition – the brain outperforms any known system. This is often chiefly owing to variety of options the human brain has that will be terribly helpful in artificial systems. The brain may be an extremely advanced, nonlinear and parallel system of easy biological process units, the neurons, (approximately 10^(10)having over (10)^(13) interconnections [1]). Neural events occur at milliseconds whereas events in typical computers occur in nanoseconds. The brain but, overcomes this slow speed through its huge range of neurons and interconnections. What is more, it's strong, fault tolerant and versatile and might contend with fuzzy, probabilistic, noisy, or inconsistent info.

This recognition of the brain's power has crystal rectifier to the interest within the development of Artificial Neural Network (ANN) technology that is impressed by the approach the brain processes info. Originally trying to model the human brain and its learning capabilities, Artificial Neural Networks or simply Neural Networks (NNets) haven't been given nevertheless a universally accepted definition, however the general public within the field would agree that a neural network consists of units (processors, nodes) that are interconnected with many alternative such units that operate severally of the input they're given and their native knowledge [2]. Some NNets attempt to model biological neural networks by victimization parallel computing design supported brain-like IP models and intrinsically they'll exhibit brain-

like behaviors like learning, association, categorization, generalization, rule extraction, improvement, etc.1.1.1Biological & Artificial Neuron

A biological somatic cell will be connected to concerning 〖10〗^4 different neurons through axons, conjunction junctions (synapses) and dendrites (tree-like networks of nerve fibres). A diagram of one somatic cell is shown in fig. 1.1. At the cell body, the soma, a typical somatic cell collects signals from others through the dendrites and sends out spikes of electrical activity through a protracted, skinny stand called the nerve fiber. The nerve fiber splits into branches at the top of that, a conjunction converts the activity from the nerve fiber into electrical effects that inhibit or excite activity within the connected neurons. once a somatic cell receives excitative input that's sufficiently massive compared with its restrictive input, it sends a spike of electrical activity down its nerve fiber. Learning happens once modifications square measure created to the effective coupling between somatic cells at the conjunction junction in order that the influence of 1 neuron on another changes.

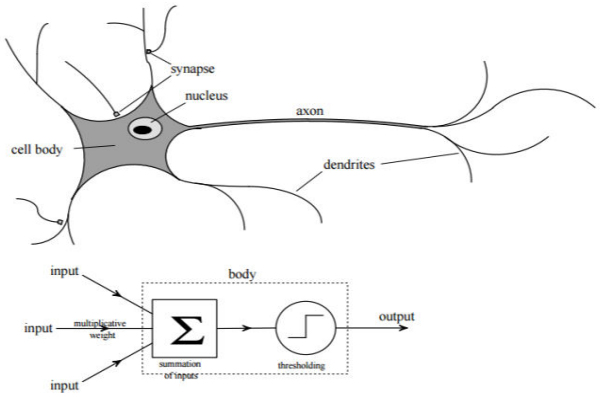

Fig. 1.1Schematic Illustrations of the Basic Features of a Biological Neuron and its Basic Model by Mcculloch and Pitts

The brain cells are able to perform more complex tasks than summation of the inputs they receive, but this is a reasonable approximation. The first attempts to model the operation of a biological neuron were made by McCulloch and Pitts in 1943 [3] based on their understanding of neurobiology [3]. Their model made several assumptions and was based on simple neurons which were considered to be binary devices with fixed thresholds. The results of this model were simple logic functions such as “a or b” and “a and b”. Considering a neural network with n such McCulloch-Pitts neurons (model neurons, units or nodes), each having two states (+1 if excited and -1 ifinhibited), the updating rule of the model can be given by

$$s_i^t = sign(\emptyset_i^t) \qquad (1.1)$$

Assuming the threshold is zero for simplicity. In equation (1.1) $\emptyset_i^t$ is the membrane potential (sum of all inputs) of the neuron *i* at time *t*, $sign(x) = +1$, for $x > 0$ and $sign(x) = -1$for $x > 0$. $sign$ is called the sign-function and it describes the processing task of the neuron. This function is usually called an updating function or activation function. If (1, 0) is used instead of (+1,-1), then the step function or Heaviside function can be used in place of equation (1.1):

$$s_i^t = \theta(\emptyset_i^t) \qquad (1.2)$$

Where $\theta = \begin{cases} 1 & x > 0 \\ 0 & x < 0 \end{cases}$

The membrane potential $\emptyset_i^t$ of the McCulloch-Pitts model is the weighted sum of all the inputs to the neuron *i* at time *t*, i.e

$$\emptyset_i^t = \sum_{j=1}^{n} w_{ij} s_j^t \qquad (1.3)$$

Where *i, j*= 1, 2, ..., n. The weight w_{ij}represents the strength of the synapse connecting neuron *j* to neuron *i* and can be excitatory (positive) or inhibitory (negative). By combining equation (1.1) and (1.3) the following equation can be obtained:

$$s_i^{t+1} = sign\left(\sum_{j=1}^{n} w_{ij} s_j^t\right) \qquad (1.4)$$

1.1.2 Neural Network Architectures

Artificial Neural Networks can be divided into two main categories based on their processing structure (known as architecture or topology): feed-forward and feedback or recurrent neural networks (fig. 1.2).

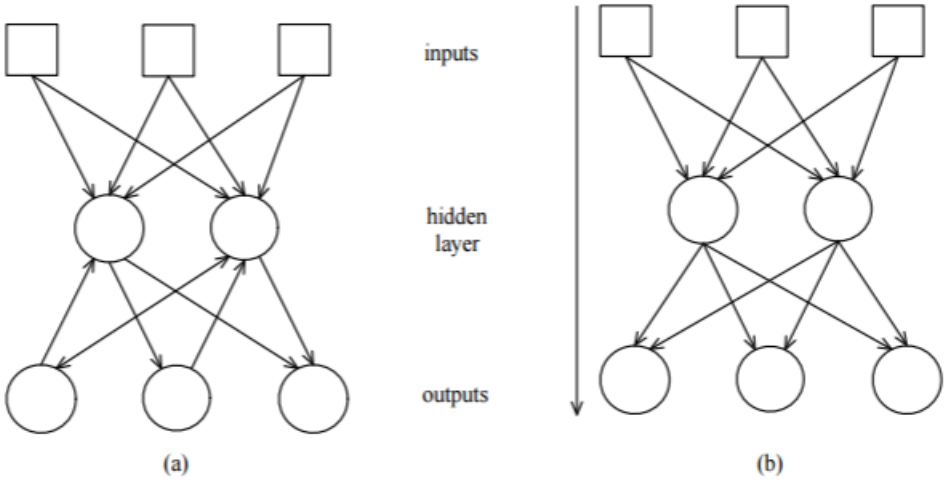

Fig. 1.2 (a) Feedback (b) Feed-Forward Neural Network

Feed-forward neural networks (fig. 1.2b) allow signals to travel one way only; from input to output. There is no feedback (loop) connections. Feed-forward networks tend to be straight forward models that associate inputs with outputs. They are extensively used in pattern recognition. This type of organization is also referred to as bottom-up or top-down.

In feedback k neural networks signals will travel in each direction. Therefore the network has loops in its topology Networks with this structure can be very complicated and powerful, as they can continuously change state until an equilibrium in reached - this balanced state is then kept up until changes in the input data occur. These architectures are also called recurrent although this usually refers to feedback connections in networks with a single layer of nodes.

These architectures can be further subdivided into two main classes based on the number of layers of processing nodes used in the model. In a single-layered network there is a layer of input nodes and a layer of output nodes, which are the only processing nodes in the model. In a multi-layered network there is one or more layers between the input and the output nodes called hidden layers. The nodes belonging to these layers are usually referred to as hidden nodes.

1.1.2.1 The Perceptron

In 1958, Frank Rosenblatt designed the first artificial neural network by interconnecting a number of McCulloch-Pitts neurons in a simple fashion [4]. He studied networks of model neurons with threshold activation functions and called them perceptron. The procedure that enables perceptron models to learn and be trained on specific exemplar data was introduced by Rosenblatt in 1962 and is known as the Perceptron Learning Algorithm [5]. This is guaranteed to converge on a suitable set of weights if a solution exists [1]. The algorithm can be summarized as follows:

1. Initialize weights and thresholds to random small values.
2. Present input example and desired output.
3. Calculate the actual output (eq. (6.3)):$o^t = f\sum_{i=0}^{n} w_i^t x_i^t$, where n is the total number of inputs, $\updownarrow w_i^t$ is the weight from input i at time t and *f(x)* is the activation function.
4. Update the weights to reinforce correct decisions and discourage incorrect decisions (reduce error):

$$if\ correct\ \ w_i^{t+1} = w_i^t$$
$$if\ output\ 0\ should\ be\ 1 \quad w_i^{t+1} = w_i^t + x_i^t$$
$$if\ output\ 1\ should\ be\ 0 \quad w_i^{t+1} = w_i^t - x_i^t$$

5. Present the next example and repeat from step 3.

The main difficulty in the above algorithm is properly updating the weights (step 4) so that the difference between the actual and the target output is as small as possible. Based on this idea, a lot of modifications have been suggested for the basic perception algorithm in order to improve its performance. One modification to the algorithm is to introduce a multiplicative factor$\eta\epsilon[0,1]$ into the weight adaptation term. This slows down the change in the weights, forcing the network to take smaller steps towards the solution. Hence, step 4 of the basic algorithm above is replaced by:

$$if\ correct\ \ w_i^{t+1} = w_i^t$$
$$if\ output\ 0\ should\ be\ 1 \quad w_i^{t+1} = w_i^t + \eta x_i^t$$
$$if\ output\ 1\ should\ be\ 0 \quad w_i^{t+1} = w_i^t - \eta x_i^t$$

Where $0 \leq \eta \leq 1$, a positive gain term that controls the adaptation rate (learning rate). Another very important modification was introduced by Widrow and Hoff, known as Widrow-Hoff rule or delta rule [6]. Windrow and Hoff realised that it would be best to have variable weight adjustments, depending on how long away is the actual output from the desired value. The delta rule calculates the difference between the weighted sum and the target output and calls that the error. The error term Δ can be written $\Delta = y^t - o^t$, where y^t is the desired response of the system and o^t is the actualresponse. Step 4 of the basic algorithm above can therefore be replaced by:

$$if\ correct\ \ w_i^{t+1} = w_i^t$$
$$otherwise\ calculate\ \ \Delta = y^t - o^t$$
$$and\ update\ the\ weights \quad w_i^{t+1} = w_i^t + \eta \Delta x_i^t$$

Where $0 \leq \eta \leq 1$,a positive gain term that controls the adaptation rate.

Limitations of the Perceptron

In 1969, Minsky and Papert pointed in their book, "Perceptrons", that single-layered networks' capabilities are limited, as this architecture (no hidden nodes) can only solve linearly separable problems[78]. A simple example that is commonly used to demonstrate this limitation is the exclusive-OR (X-OR) logic function which accepts two inputs (0 or 1) and produces an output of 1 only if either input is 1; otherwise it outputs 0 (fig. 1.3). If the output is plotted in a two-dimensional space, it can be shown that it is impossible to draw a single straight line between the two output classes. In [1] it is easily shown that the exclusive-or problem cannot be simulated by a single-layered network, since there is no set of weights that completely satisfies the truth table of the X-OR function.

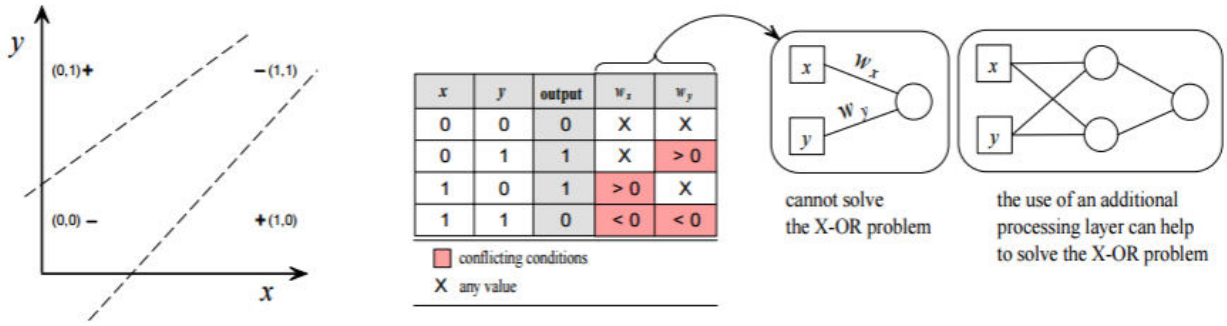

Fig. 1.3 The Exclusive-OR (X-OR) Logic Function [1]

Many classification problems are not linearly separable and this leads to the use of multi-layered networks that are capable of dealing with non-linear classification problems (it can form more complex decision regions) due to the use of additional layers of processing nodes.

1.1.2.2 Multi-layer Perceptron (MLP)

Multi-layer perceptron (MLP) is the architecture that most people prefer for its capability of performing arbitrary classifications. The following section gives a description of one of the most commonly used learning algorithms for the MLP, suggested by Rumelhart and McClelland [8].

The Back-Propagation algorithm (BP) The learning algorithm used for multi-layered perceptrons is called generalized delta rule or back-propagation rule and it was developed by Rumelhart, McClelland and Williams in 1986 and their work is delineated in their book titled "Parallel Distributed Processing". The back-propagation rule generalizes the delta rule delineated in section 1.1.2.1.Convergence is not guaranteed and it is frequently slow even when it converges. The algorithm is a gradient descent method and uses an error function that represents the difference between the network's calculated output and the desired output. In order to learn successfully, the network's output should approach the target output by continuously reducing the value of this error. The generalized delta rule calculates the error for a specific input and so back-propagates the error from one layer to the previous one. The association weights between the nodes area unit adjusted in line with the back-propagated error in order that the error operate is reduced and therefore the network learns.

Scaled Conjugate Gradient Algorithm (SCG)

The Scaled Conjugate Gradient (SCG) algorithm is a member of the class of conjugate gradient methods. These are general purpose second-order techniques that minimize goal functions. Unlike the back-propagation algorithm, which is a first-order technique, these methods make use of the second derivatives of the goal function. This allows them to find a better way to a minimum than a first-order technique, but at a higher computational cost. In a similar fashion to the back-propagation algorithm, conjugate gradient methods try to get closer to the minimum of the function, but instead of always proceeding down the gradient of the error function, they follow a direction conjugate to the directions of the previous steps. Most minimization methods use first-order derivatives combined with line search methods along selected direction, a sin back-propagation.

1.1.3 Design Issues & Training of ANNs

When it comes to deciding which ANN architecture to use and which algorithm to apply to this structure, there is no easy answer. Selection of a suitable combination of these two design components is often done experimentally and it is problem dependent (i.e. the difference between a classification and a time-series prediction problem will probably result in the use of a feed-forward neural network for the first and a recurrent neural network for the second problem, respectively). In general, the specification and design of an ANN should aim to produce the system with the optimum overall performance, although this, together with what should beconsidered 'optimum' performance, is also problem dependent. In most applications – and especially in medical problems – ANNs should be used to supplement conventional methods. Furthermore, in some cases it might be necessary for the neural network not to try and improve the overall performance only, but mainly focus on the improvement or restriction only of the performance over one or more specific classes of the data.

There are several issues to be considered when designing a neural network application, aiming to achieve a good generalization performance – i.e. to have the outputs of the network approximate well target values given inputs that are not in the training set. The design of a machine learning system in general usually involves the following tasks:

(a) Data collection:It is a necessary condition for good generalization that the training cases be a sufficiently large and representative subset (the sample) of the set of all cases that need to generalize to (the population). For unseen cases that are nearby training examples in the feature space the trained network needs to interpolate, which can often be done reliably. For unseen cases that may lie farther away from training examples (inside or outside the range of the training set) the network needs to extrapolate, which is notoriously unreliable. Hence, it is important to have sufficient training data to avoid the need for extrapolation.

Thus, if the input-output function to be learned is smooth, the correct output for a test example that lies close to some training cases, will be close to the correct outputs of those training cases. Hence, with proper training and given a sufficient sample for the training set, a neural network will be able to generalize reliably to the population.

(b) Data pre processing: This step is usually needed in order to extract information (in the form of features) related to the problem. That might be necessary if the original (or raw) data is in a format not suitable for use by the neural network. For example, in a medical vision problem it might be necessary to capture images from the media that is normally used by the

medical personnel (e.g. microscope, X-rays, MRIs, etc.) and then digitize them, if they are not already digitized. In addition, because of hardware and software restrictions, it might also be considered necessary to pre-processes the images so that significant features of image objects related to the decision making process can be extracted and numerically expressed.

(c) Feature Extraction: The inputs to the network should contain sufficient information pertaining to the target, so that there exists a mathematical function relating correct outputs to inputs with the desired degree of accuracy. For example, when classifying image objects using a set of outdoor scene images, just the use of color might not be sufficient input for a proper categorization of the objects; additional information such as shape, size and texture might be necessary for discriminating between objects of different categories. Finding good inputs for a classifier and collecting enough training data often take more time and effort than training the network. The features are chosen by the designer based on its own knowledge and experience or the knowledge of an expert on the field of the problem. In most cases it is usual to find redundant or ineffective features, if they exist, and try to eliminate them (feature selection) as this would reduce the complexity and training time of the final system. Furthermore, sets of features that are most significant can be determined by comparative analysis.

Specific to the design of a neural network system the following tasks are necessary:

(d) Selection of an ANN type and architecture: As already mentioned at the beginning of the section, the architecture or topology of the network depends on the problem and the nature of the data. For example, a feedback architecture is usually required when the data is time depended and each pattern may require as input the output generated from the previous training example. Feedback NNets are usually more difficult to train than feed-forward NNets while the latter type are the most often used in practical applications. This is because feed-forward NNets usuallyproduce a response to an input quickly and can be trained using a wide variety of efficient conventional numerical methods in addition to algorithms developed by Neural Network researchers.

Other issues closely related to the network topology include the number of hidden layers and hidden units to be used. In a wide variety of applications a linear or generalized linear model can be sufficient for developing a good classification system. In that case no hidden layers will be needed. However, in applications where a nonlinear model is needed, an MLP with one hidden layer with arbitrary large number of units typically suffices for a good approximation. The optimum number of hidden units depends on the number of input features, the size of the training set, the amount of noise present in the data, the architecture and the choice of functions and parameters in the algorithm. Although a lot of attempts have been made in order to determine a general ‘rule of thumb’ that give the most appropriate number of hidden units to be used for training the network [Sarle, 2000], none of them can be considered reliable enough or true for any situation. Among approaches that attempt to help with this problem is the use of early stopping.

(e) ANN training, testing and validation: Designing and training an ANN also involves a minimum of three sets of independent input/output vector pairs representative of the process. This is part of a method known as early stopping, which is usually employed in order to prevent over fitting on the sample data and improve generalization.

1.1.4 Complexity &Generalization Performance of an ANN

The complexity of an ANN mainly depends on the number of adaptable parameters (weights and biases) in the system. The level of complexity can greatly affect the overall performance of the system. MLPs are often used for classifying data in a high dimensional feature space by defining a decision boundary between the different classes sampled in the feature space. Generally, the higher the complexity of the MLP being trained the higher the non-linearity of the decision boundary. If the MLP’s complexity is larger than the optimum needed, then the boundary formed is sensitive to noise and over fit to the training data giving poor generalization performance (classification of unseen data). Lower complexity of the MLP than the one needed for the task will result in the opposite effect of over fitting, under fitting giving again poor generalization performance. This is because the system is unable to form the decision boundaries needed to completely separate the classes. For the trained MLP to classify as accurately as possible new, unseen data, an optimum complexity must be found. There are several methods that can be used at different stages of the above mentioned process in order to

find a near optimum complexity. As already mentioned earlier, in the data pre processing stage, both the feature extraction and feature selection are significant factors in the training of an ANN since they can determine the number of input features to be used for the design and therefore directly affect the complexity of the network. For the training of the network, also a number of different techniques could be used in order to try and increase its generalization performance. These may include early stopping, cross-validation, and varying the size of the training data that must be a representative sample of the 'population'.

1.1.5 Feature Selection

In many real-life problems, features relevant to the problem are unknown. Therefore, in feature extraction it is usually the case where a large number of candidate features is chosen to better represent the domain of the problem. This results in many of the extracted features being either irrelevant or redundant, which in any case they do not affect the output of the network. However, the running time of the learning process is increased and in some cases, depending on the size of the training data set, learning might not work well without removing the unwanted features. An extensive analytical work on feature selection for classification has been presented in [9] while the work of Hall [10] with a number of publications on the subject also provide a good reference.

Feature selection methods search for the best subset of features through the competing 2^ncandidate subsets according to an evaluation function. The selection is such that the classification accuracy does not significantly decrease and the resulting class distribution given the selected feature subset is as close as possible to the class distribution given the whole feature set. Searching for the best feature subset is an exhaustive procedure even for a medium-sized feature set. Over the past few years, a number of search algorithms have been designed to prevent an exhaustive search of subsets and reduce computational complexity. A typical feature selection method involves the following tasks:

1. Generation of next candidate subset
2. Evaluation of generated subset
3. Stopping criterion
4. Validation of selected subset

A number of generation (or search) and evaluation algorithms exist. The most widely known generation procedures include the Branch & Bound algorithm [82], RELIEF [83], RELIEF-F [14], and the wrapper method [13]. Branch & Bound does a complete search through the feature subsets, but avoids an exhaustive search by exploiting the mono tonicity principle of a monotonic evaluation function. The rest of the above mentioned methods use a heuristic search where generation of subsets is incremental (increasing or decreasing). Other well-known heuristic approaches include the Sequential Forward Selection (SFS) and Sequential Backward Selection (SBS). SFS starts from an empty set and in each iteration generates subsets by adding a feature selected by an evaluation function, while SBS works backwards, i.e. it starts from the complete feature set and in each iteration generates a subset by discarding a feature selected by an evaluation function.

The evaluation function basically determines the outcome of the feature subset selection, since each function uses a different measure in order to determine whether a feature or a subset of features will be selected or rejected. Feature selection methods are grouped into two categories: filter methods which are independent of the inductive algorithm that will use the selected features and wrapper methods which use the inductive algorithm as the evaluation function [10]. Based on [9] different types of evaluation functions use one of the following measures: distance, uncertainty, dependence, consistency and classifier error rate, with the latter being used by wrapper methods.

Feature sub set selection is a major design issue not only for Artificial Neural Networks but for any other machine learning algorithm that might be used. In a comparative study between different algorithms it might be necessary and simpler to choose a filter method as this is independent of the inductive algorithm and might give results faster. Alternatively, the use of a wrapper method might provide additional information to be used for the comparison between the algorithms under consideration.

1.1.6 Early Stopping

This technique is used for improving the generalization of a neural network during training and preventing it from over fitting. It achieves that by splitting the available data into three subsets which are used separately for training, validation and testing. These sets are used in the following manner: At each iteration through the training set, the ANN runs a test over the validation set; this is done iteratively while the performance of the network continues to improve. When the performance does not show further improvement with more iterations then the training is completed and the ANN runs a test over the test set. The result gives the final performance of the classifier.

The basic concept is that the performance of the ANN against the training set will continue to improve with more training, and this is also true for its performance against the validation set. However, if the ANN starts over fitting the training set, its performance then against the validation set will start to decrease. Therefore, increase of the validation error (i.e. the error against the validation set) is indicative of poor generalization and can be used for stopping the classifier from further training. The weights giving best performance against the validation set can then be accepted as the set of weights that gives the ANN best generalization ability.

As the error surface is not always smooth, it is not certain at this point of the training if the validation error is the global minimum. So, it is common practice to continue training for several iterations (based on the judgement of the designer) in case the ANN performance against the validation set improves again. At the same time the configuration and set of parameters of the ANN that gave best results until this point are always stored. Because the validation set has been utilized during the training process, after the training is finalized, it is important to test the trained ANN against a separate set (the test set), which hasn't been used at all during training and will give an unbiased estimate of the generalization error of the network.

A practical alternative use of early stopping is to 'rotate' the validation set, i.e. divide the data (after a test set has been selected and excluded from it) into k subsets and use a different subset for validation and the remaining subsets for training, leading to k different combinations of training and validation sets. This approach has the advantage of avoiding over fitting or under fitting as a result of a bad split of the data into training and validation sets. Early stopping can then be applied k times and after each separate training has been completed, the generalization error can be estimated against the test set. As each different training has been tested against the same test set, the one that gave the best performance can be chosen.

Concerning the architecture of the ANN and the number of hidden units to be used, in early stopping, it is essential to use lots of hidden units to avoid bad local optima. There seems to be no restriction on the number of hidden units, other than that imposed by computer time and memory requirements [2].

1.1.7 Cross-Validation

Cross-validation may be a methodology for estimating generalization error supported resampling [15 - 17] and therefore the obtained results area unit usually used for selecting among numerous models, like completely different network architectures. In k-fold cross-validation, the info is split into k (approximately) equally sized subsets. The ANN is trained k times, whenever omitting one among the subsets, that is employed for testing. If k equals the sample size (i.e. there are a unit as several subsets as information examples) the strategy is named "leave-one-out" cross-validation (LOOCV). "Leave-v-out" cross-validation (LVOCV) involves effort out all potential subsets of v cases. The ultimate generalization performance of the system will then be calculable by averaging between the results obtained from every set.

Cross-validation is also wont to estimate the generalization error of a given model, or for model choice by selecting the model that has the littlest calculable generalization error. Hence, cross-validation can be wont to select the amount of hidden units, or a set of the input options (feature set selection). Moreover, this methodology is right for little information sets that can't be split into coaching and check subsets. Cross-validation permits for the complete information set to be used for coaching whereas it still provides freelance estimate of the generalization error.

1.1.8 Sensitivity versus Specificity

The performance of a binary classifier is typically quantified by its accuracy throughout the take a look at part, i.e. the fraction of misclassified points on the take a look at set. the employment of the general classification accuracy as Associate in Nursing analysis metric is adequate provided the category distribution among examples is constant and comparatively balanced. Moreover, this analysis approach conjointly assumes equal error prices, i.e. that a false positive error is equally vital as a false negative error. Sadly, in real-life issues, these assumptions aren't forever true. Consequently, the performance of such systems area unit best represented in terms of their sensitivity and specificity quantifying their performance associated with false positive and false negative instances. These metrics area unit supported the thought that a take a look at purpose forever falls into one among the subsequent four categories: False Positive (FP) if the system labels a negative purpose as positive; False Negative (FN) if the system labels a positive purpose as negative; True Positive (TP) and True Negative (TN) if the system properly predicts the label. Within the following TP, TN FP, FN area unit won't to denote the amount of true positives, true negatives, false positives and false negatives, severally.

1.2 SUPPORT VECTOR MACHINE

Conventional neural network ways have incontestable difficulties finding a decent generalization performance as a consequence of the algorithms used for parameter choice and also the applied mathematics measures used for choosing the model with the "optimum performance" [18]. Convergence of a neural network to a worldwide minimum of the error operate involves the fine calibration of variety of parameters together with crucial the quantity of hidden units, the worth of the educational rate, etc. thanks to these difficulties, within the previous few years the scientific community has been experimenting with Support Vector learning. This less dimmed learning technique was recently introduced and has started receiving hyperbolic attention as a result of the easy concepts it's supported and also the high performance it's incontestable in sensible applications.

Support Vector Machines (SVMs) were initial introduced by Vladimir Vapnik between the late seventies and early eighties [19] and that they are supported the Structural Risk step-down (SRM) principle from applied mathematics learning theory [20]. In antithesis to the Empirical Risk step-down (ERM) principle that is employed by neural networks to attenuate the error on the coaching knowledge SRM minimizes a certain on the take a look at error, therefore permitting SVMs to generalize higher than standard neural networks [18]. excluding the matter of poor generalization and over fitting, SVMs additionally address the issues of potency of coaching, potency of testing and rule parameter calibration [21], issues oftentimes encountered in artificial neural network ways.

SVMs were developed to unravel classification issues and initial work was centered on optical character recognition (OCR) applications [22]. Some recent applications and extensions of SVMs embrace isolated written digit recognition [23], visual perception [24], identification [25,], face detection in pictures [26 ,27] and text categorization [28]. All of the on top of cases ar samples of SVMs used for classification. However, SVMs have additionally been planned and applied to variety of various varieties of issues together with regression estimation, novelty detection, and resolution of inverse issues. This can be as a result of the success of the tactic and also the got to adapt it to specific issues. In regression and statistic prediction applications, SVMs are compared to variety of competitor ways and their generalization performance was found to either match or be considerably higher than these ways [29, 20]. The employment of SVMs for density estimation [31] and ANOVA decomposition [32] has additionally been studied. Relating to extensions, the fundamental SVMs contain no previous information of the matter and far work has been done on incorporating previous information into SVMs [34]. Though SVMs have sensible generalization performance, they'll be terribly slow in take a look at section, a tangle addressed in [33,37]. Recent work has generalized the fundamental concepts [35, 36] and shown however these concepts are often incorporated in a very wide selection of alternative algorithms.

In the literature the term Support Vector Machine has been accustomed describe classification victimization support vector ways whereas the term Support Vector Regression Machine has been accustomed describe regression with support vector ways. During this thesis the term support vector machine (SVM) can ask support vector ways used for

either classification or regression and also the terms Support Vector Classification (SVC) and Support Vector Regression (SVR) are going to be used for specification.

The sections that follow provides a transient description of the fundamental ideas of applied mathematics learning theory associated structural risk step-down followed by an introduction to SVMs within the settings of each classification and regression. For added material and a a lot of elaborated description of SVMs one will ask the works of V. Vapnik [19,20], C. Burges [38], and A. Smola [34,36].

1.2.1 Statistical Learning Theory

Generally, for a given learning task, with a given finite quantity of coaching knowledge, the simplest generalization performance are going to be achieved if the proper balance is reached between the accuracy earned on the actual coaching set, and therefore the ability of the machine to be told any coaching set while not error, that is, its capability. A trained machine with very large capacity will over fit on the training data and will be able to identify only previously seen examples. While a machine with very small capacity will not be able to identify even previously seen data, i.e. learning of the training data will be incomplete. Neither can generalize well and therefore, the relation between the capacity and the performance of a learning machine must be controlled to achieve the right balance.

1.2.1.1 Empirical Risk Minimization

Suppose, for the case of two-class pattern recognition, l observations are given. Each observation consists of a pair: a vector $x\epsilon R^d, i = 1,2, \dots \dots, l$ and the associated label y_i , given by the source of information. Ultimately, the task of a learning machine is to estimate a function $f_\alpha: R \to \{\pm 1\}$ using these examples, such that f_α will correctly classify unseen examples (x, y). So, assuming that there exists some unknown probability distribution $P(x, y)$ drawn from the given data, the aim is to estimate the function f_α(x) = y for examples (x, y)that were generated from the same underlying probability distribution $P(x, y)$as the training data. Actually, the learning machine can be defined by a set **F** of possible mappings $x_i \to f_\alpha(x)$, where the functions f_α themselves are labelled by the adjustable parameters α. The machine is assumed to be deterministic: for a given input x, and choice of α, it will always give the same output f_α(x). A particular choice of α generates a trained machine. As only the training data is given, a measure that could be used to decide which of the possible functions is preferable, is the training error or the empirical risk $R_{emp}\ [f_\alpha]$, which is defined to be the measured mean error rate on the training set (for a fixed, finite number of observations).

$$R_{emp}\ [f_\alpha] = \frac{1}{2l}\sum_{i=1}^{l}|y_i - f_\alpha x_i| \qquad (1.5)$$

$R_{emp}\ [f_\alpha]$ is a fixed number for a particular choice of α and for a particular training set(x_i, y_i). The quantity $y_i - f_\alpha x_i$ is called the loss and for the case described here it can only take the values {-1} and {+1}.

1.2.1.2 Structural Risk Minimization

VC (Vapnik-Chervonenkis) theory, a part of Statistical learning theory, shows that the set of functions **F** where f_α is chosen from must be restricted to one which has a capacity that is suitable for the amount of the available training data. To do so, VC theory introduces bounds on the expected risk that depend on both the empirical risk and the capacity off. The minimization of these bounds leads to the structural risk minimization principle. According to this principle, for all the functions in**F** (i.e. for any α) and $l > h$ with a probability of at least$1 - \eta(0 \le \eta \le 1)$, the following bound holds:

$$R\ [f_\alpha] \le\ R_{emp}\ [f_\alpha] + \sqrt{\frac{h\left(log\left(\frac{2l}{h}\right)+1\right)-\log\left(\frac{\eta}{4}\right)}{l}} \qquad (1.6)$$

The parameter h is a non-negative integer called the VC dimension, which is defined as the largest number of points that can be separated in all possible ways using functions of the given set. In effect, the VC dimension provides a measure of the notion of capacity. The right hand side of (6.6) is a bound on $R\ [f_\alpha]$ and it holds only with a certain probability. The second term of this bound is called the VC confidence which is a monotonic increasing function of h for every value of l. This bound is independent of $P(x, y)$, andif **η**is known, it can be easily computed. Conversely,

the left hand side is difficult to compute. Thus, given some selection of learning machines whose empirical risk is zero and choosing a fixed, sufficiently small **η**, one must choose the learning machine whose associated set of functions **F** has minimal VC dimension. This leads to a better upper bound on the expected risk. In general, for non-zero empirical risk, one wants to choose that learning machine which minimizes the right hand side of equation (6.6). This is the essential idea of SRM.

1.2.2 Support Vector Classification

Having introduced the basic concepts of statistical learning theory, the basis needed for describing support vector machines is set. In the previous section, based on a binary classification case, it is shown that one needs to find a class of functions whose capacity can be computed in order to design a learning machine. Vapnik and Chervonenkis in [39, 40] considered the class of hyper planes

$$w.x + b = 0, w \epsilon R^d, b \epsilon R \quad (1.7)$$

Corresponding to decision functions

$$O(x) = sign(w.x + b) \quad (1.8)$$

And represented the Generalized Portrait rule for constructing separating hyperplanes from empirical knowledge. This learning rule was projected for severable issues and it's supported the actual fact that among all potential separating hyper planes there exists a novel one with most margin of separation from the categories (a best margin hyper plane) and also the capability decreases with increasing margin

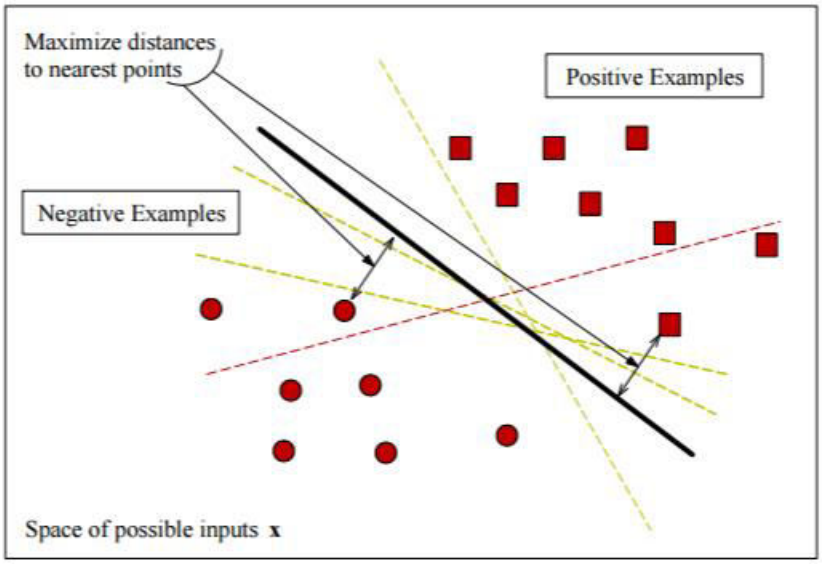

Fig. 1.4 Optimal Margin Hyperplane for the Separable Case

1.3 CONCLUSION

In this chapter introduction to different type of biomedical image classification techniques are introduced and special focus on support vector machine. And how classification accuracy can be achieved depending on the changes and other neural approaches. Neural when combined with fuzzy logic also gives an efficient classifier. The classifiers described in this paper categorizes the image as normal and abnormal and present the location of tumour via clustering. The manual procedure that pathologist choose for diagnosis is microscopic detection which is often time consuming and causes fatigue to them, hence this proposed system is quite beneficial . But there are several forms in which a tumour is categorized depending upon the size and location of tissues inside the brain. Types of brain images like benign and malignant types and their features are discussed.

REFERENCES

[1] [Beale and Jackson, 1994] Beale, R. and Jackson, T. (1994). *Neural Computing: Anintroduction*. Institute of Physics Publishing Ltd., Bristol, UK.
[2] [Sarle, 2000] Sarle, W. S. (2000). Neural Network FAQ. FTP Archives. ftp://ftp.sas.com/pub/neural/FAQ.html [accessed July 2000].
[3] McCulloch, W. and Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, 5:115– 133.
[4] Rosenblatt, F. (1959). The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological Review*, 65:386–408.
[5] Rosenblatt, F. (1962). *Principles of neurodynamics*. Spartan Books, New York.
[6] Widrow, B. and Hoff, M. (1960). Adaptive switching cirduits. *IRE WESCON Convention Record*, 4:96–104.
[7] Minsky, M. and Papert, S. (1969). *Perceptrons*. MIT Press, Cambridge, MA.
[8] Rumelhart, D. and McClelland, J. (1986). *Parallel distributed processing: exploration in the microstructure of cognition*, volume 1 & 2.MIT Press, Cambridge, MA.
[9] Dash, M. and Liu, H. (1997). Feature Selection for Classification. *Intelligent Data Analysis*, 1(3):131–156. Elsevier Science Inc.
[10] Hall, M. (2000). Correlation-based feature selection of discrete and numeric class machine learning. In *Proceedings of the International Conference on MachineLearning*, pages 359–366, San Francisco, CA. Morgan Kaufmann Publishers.
[11] Narendra, P. M. and Fukunaga, K. (1977). A branch and bound algorithm for feature selection. *IEE Transactions on Computers*, C-29(9):917– 922.
[12] Kira, K. and Rendell, L. A. (1992). The feature selection problem: Traditional methods and a new algorithm. In *Proceedings of Ninth NationalConference on Artificial Intelligence*, pages 129–134.
[13] Kohavi, R. and Sommerfield, D. (1995). Feature subset selection using the wrapper method: Overfitting and dynamic search space topology. In *Proceedings of First International Conference on Knowledge Discovery and DataMining*, pages 192–197. Morgan Kaufmann.
[14] Kononenko, I. (1994). Estimating attributes: Analysis and extension of RELIEF. In *Proceedings of European Conference on Machine Learning*, pages 171–182. Morgan Kaufmann.
[15] Hjorth, J. (1994). *Computer Intensive Statistical Methods Validation, Model Selection, and Bootstrap*. Chapman & Hall, London.
[16] Plutowski, M., Sakata, S., and White, H. (1994). Crossvalidation estimates IMSE. In Cowan, J., Tesauro, G., and Alspector, J., editors, *Advancesin Neural Information Processing Systems*, volume 6, pages 391–398. Morgan Kaufman, San Mateo, CA.
[17] Weiss, S. and Kulikowski, C. (1991). *Computer Systems That Learn*. Morgan Kaufmann.
[18] Gunn, S. (1998). Support Vector Machines for Classification and Regression. Technical report, University of Southampton. Image, Speech and Intelligent Systems Research Group ISIS.
[19] Vapnik, V. (1982). *Estimation of Dependences Based on Empirical Data*. Springer-Verlag, New York.
[20] Vapnik, V. (1995). *The Nature of Statistical Learning Theory*. Springer- Verlag, New York.
[21] Cristianini, N. and Shawe-Taylor, J. (2000). *An introduction to Support Vector Machines and other kernel-based learning methods*.Cambridge University Press, Cambridge, UK.
[22] Cortes, C. (1995). *Prediction of Generalisation Ability in Learning Machines*. PhD thesis, Department of Computer Science.
[23] Cortes, C. and Vapnik, V. (1995). Support Vector Networks. *Machine Learning*, 20:273–297.
[24] Blanz, V., Sch¨olkopf, B., B¨ulthoff, H., Burges, C. J. C., Vapnik, V., and Vetter, T. (1996). Comparison of view-based object recognition algorithms using realistic 3d models. In von der Malsburg, C., von Seelen, W., Vorbrggen, J., and Sendhoff, B., editors, *Artificial Neural Networks – ICANN'96*, volume 1112 of *LectureNotes in Computer Science*, pages 251–256, Berlin. Springer.
[25] Schmidt, M. (1996). Identifying Speakers with Support Vector Machines. In *Proceedings of Interface '96, Sydney*.

[26] Osuna, E. E., Freund, R., and Girosi, F. (1997). Support Vector Machines: Training and Applications. A.I.Memo 1602, Massachusetts Institute of Technology.
[27] Osuna, E. E., Freund, R., and Girosi, F. (1997). Training Support Vector Machines: An Application to Face Detection. In *Proceedings of ComputerVision and Pattern Recognition '97*, pages 130–136.
[28] Joachims, T. (1997). Text categorization with support vector machines. Technical Report LS VIII Number 23, University of Dortmund.
[29] Drucker, H., Burges, C. J. C., Kaufman, L., Smola, A. J., and Vapnik, V. (1997). Support Vector Regression Machines. In Mozer, M., Jordan, M., and Petsche, T., editors, *Advances in Neural Information Processing Systems*, volume 9, pages 155–161, Cambridge, MA. MIT Press.
[30] Mattera, D. and Haykin, S. (1999). Support Vector Machines for Dynamic Reconstruction of a Chaotic System. In Sch¨olkopf, B., Burges, C. J. C., and Smola, A. J., editors, *Advances in Kernel Methods – Support VectorLearning*, pages 211–242. MIT Press, Cambridge, MA.
[31] Weston, J., Gammerman, A., Stitson, M. O., Vapnik, V., Vovk, V., and Watkins, C. (1997). Density estimation using support vector machines. TechnicalReport CSD-TR-97-23, Royal Holloway College.
[32] Stitson, M. O., Gammerman, A., Vapnik, V., Vovk, V., Watkins, C., and Weston, J. (1999). Support Vector Regression with ANOVA Decomposition Kernels. In Sch¨olkopf, B., Burges, C. J. C., and Smola, A. J., editors, *Advances inKernel Methods – Support Vector Learning*, pages 285–292. MIT Press, Cambridge, MA.
[33] Burges, C. J. C. (1996). Simplified support vector decision rules. In Saitta, L., editor, *Proceedings of the Thirteenth International Conference on MachineLearning*, volume 1112, pages 71–77, Bari, Italy. Morgan Kaufman.
[34] Burges, C. J. C. (1998). Building locally invariant kernels. In *Proceedings of the 1997 NIPS Workshop on Support Vector Machines*, volume 1112, pages251–256.
[35] Smola, A. J. (1998). *Learning with Kernels*. PhD thesis, Technische Universit¨at Berlin, Berlin.
[36] Smola, A. J. and Sch¨olkopf, B. (1998). On a kernel-based method for pattern recognition, regression, approximation and operator inversion. *Algorithmica*, 22:211–231.
[37] Osuna, E. E. and Girosi, F. (1998). Reducing the run-time complexity of support vector machines. In *International Conference on Pattern Recognition(ICPR'98)*, Brisbane, Australia.
[38] Burges, C. J. C. (1998). A Tutorial on Support Vector Machines for Pattern Recognition. *Data Mining and Knowledge Discovery*, 2(2):121–167.
[39] Vapnik, V. and Chervonenkis, A. (1964). A note on one class of perceptrons. *Automation and Remote Control*, 25.
[40] Vapnik, V. and Chervonenkis, A. (1974). *Theory ofPattern Recognition [in Russian]*. Nauka, Moscow. (German Translation: W. Wapnik & A. Tscherwonenkis, *Theorie der Zeichenerkennung*, Akademie–Verlag, Berlin, 1979).
[41] Gonzalez RC, Woods RE, 1992, "Digital image processing", Addison-Wesley.
[42] WHO, Feb 1998, "Fact Sheet No 104: Tuberculosis", WHO, Geneva.