Volume 5, Issue 4, 2015 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Introduction to Probabilistic Neural Network –Used For Image Classifications

[1]Shreepad S. Sawant, [2]Preeti S. Topannavar
Department of Electronics and Telecommunication Engineering
[1]PG Student, JSPM's Imperial College of Engineering and Research, Pune (M.H.) India
[2]Assistant Professor, JSPM's Imperial College of Engineering and Research, Pune (M.H.) India

Abstract: In the defect identification classification of an image plays an important role. There are different classifiers available for the image classification. But the neural network based classification has advantages over the other method. Neural networks are predictive models loosely based on the action of biological neurons. Neural network classifier also subdivided into different types classifiers. Self organizing map (SOM) neural network, Multilayer Perceptron (MLP) neural network, radial basis function (RBF) neural network and Probability neural network (PNN) these are different types of neural network classifiers. In this paper we are going to study about Probability neural network (PNN) classifier. In this study, PNN classifier is explained with its working and its advantages.

Keywords:- Self Organizing Map (SOM), Probabilistic Neural Network (PNN), Radial Basis function (RBF), multilayer Perceptron (MLP).

I. INTRODUCTION

Classifiers are used for object recognition and classification use. Classification includes image sensors, object detection, image preprocessing, feature extraction, object classification, and object segmentation [3]. Using the features of the defect is one of the most widely used techniques for the classification and detection [7]. In the classification classifiers senses the object properties or features such as energy, entropy, correlation, homogeneity entropy etc. We have seen various kinds of classifiers in past many years. Neural network classifiers are one of them. Neural networks are predictive models loosely based on the action of biological neurons. The Neural network (NN) is seen as information paradigm inspired by the way the human brain processes information [5]. The selection of the name "neural network" was one of the great PR successes. It certainly sounds more exciting than a technical description such as "A network of weighted and additive values with nonlinear transfer functions". Though despite the name neural networks are far from "artificial brains" or "thinking machine". Generally a typical artificial neural network might have a hundred neurons. As compare to the human nervous system which is believed to have about $3x10^{10}$ neurons, it is very less.

The original "Perceptron" model was developed by Frank Rosenblatt in 1958. This Rosenblatt's model consisted of three layers. In first layer a "retina" that distributed inputs to the second layer. During second layer "association units" which combine the inputs with weights and trigger a threshold step function which feeds to the output layer. In third layer the output layer which combines the values. Unluckily, the use of a step function in the neurons made the perceptions difficult or impossible to guide. A critical analysis of perceptrons published in 1969 by Marvin Minsky and Seymore Papert pointed out a number of critical weaknesses of perceptrons and for a period of time interest in perceptrons waned [10].

Interest in neural networks was revived in 1986 when David Rumelhart, Ronald Williams and Geoffrey Hinton published "Learning Internal Representations by Error Propagation". These scientists projected a multilayer neural network with nonlinear but differentiable transfer functions that avoided the pitfalls of the original perceptron's step functions.

Neural network is the best tool in recognition and discrimination between different sets of signals. For getting best results using the neural network, we have to choose a suitable architecture and learning algorithm. Unluckily there is no guaranteed method to do this. The best way to do that is to choose what is expected to be suitable according to our previous experience and then to expand or shrink the neural network size until a reasonable output is obtained.

II. THE MULTILAYER PERCEPTRON NEURAL NETWORK MODEL

This network has an **input layer** (on the left) with three neurons in it, one **hidden layer** (in the middle) with three neurons and an **output layer** (on the right) with three neurons.

There is one neuron in the input layer for each predictor variable. For categorical variables, to represent the *N* categories of the variable, *N*-1 neurons are used

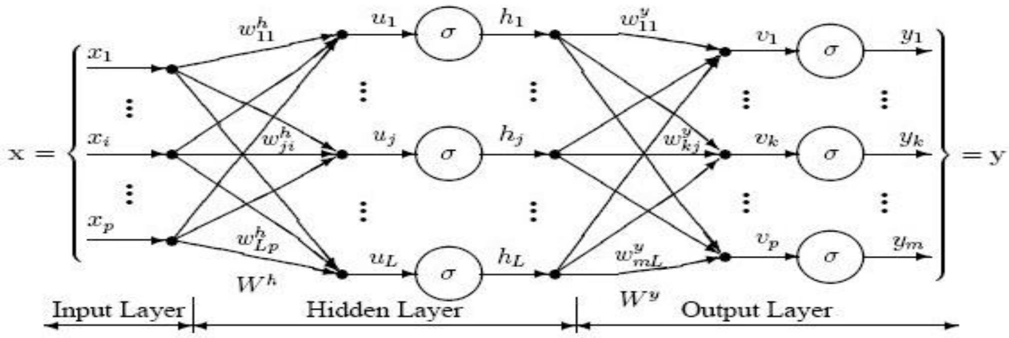


Fig 1: Perceptron network with three Layers

.

A. Input Layer

A vector of predictor variable values ($x_1...x_p$) is presented to the input layer. The input layer (or processing before the input layer) standardizes these values so that the range of each variable is -1 to 1. The input layer distributes the values to each of the neurons in the hidden layer. Additionally to the predictor variables, a constant input of 1.0 called the *bias* that is fed to each of the hidden layers; the bias is multiplied by a weight and added to the sum going into the neuron.

B. Hidden Layer

Arriving at a neuron in the hidden layer, value from each input neuron is multiplied by a weight (w_{ji}) and the resulting weighted values are added together producing a combined value u_j. The weighted sum (u_j) is fed into a transfer function (σ) which outputs a value (h_j). Outputs from the hidden layer are distributed to the output layer.

C. Output Layer

Arriving at a neuron in the output layer, value from each hidden layer neuron is multiplied by a weight (w_{kj}), and the resulting weighted values are added together producing a combined value v_j. The weighted sum (v_j) is fed into a transfer function (σ) which outputs a value (y_k). The *y* values are the outputs of the network.

If a regression analysis is being performed with a continuous target variable then there is a single neuron in the output layer and it generates a single y value. There are *N* neurons in the output layer producing *N* values, single for each of the *N* categories of the target variable, for classification problems with categorical target variables. Various kinds of neural-network architecture including multilayer perceptron (MLP) neural network, radial basis function (RBF) neural network, self-organizing map (SOM) neural network and probabilistic neural network (PNN) have been proposed [1]. PNN has become an effective tool for solving many classification problems because of ease of training and a sound statistical foundation in Bayesian estimation theory. PNN uses Baye's strategy for decision making with non parametric estimator for obtaining probability density function.

.

III. PROBABILISTIC NEURAL NETWORKS (PNN):

A probabilistic neural network is predominantly a classifier [8]**.** PNN uses a supervised training set to develop probability density functions within a pattern layer [5]. This is a model based on competitive learning with a 'winner takes all attitude' and the core concept based on multivariate probability estimation [2]. Probabilistic (PNN) and General Regression Neural Networks (GRNN) have similar architectures, except there is a fundamental difference. General regression neural networks perform regression where the target variable is continuous, whereas Probabilistic networks perform classification where the target variable is categorical.

A. Architecture of a PNN:

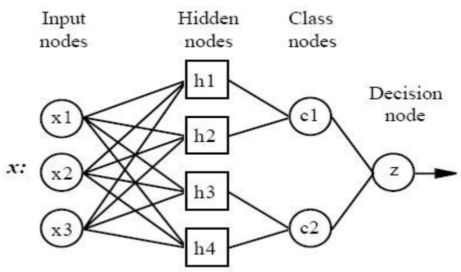


Fig 2: Architecture of PNN

All PNN networks have four layers:

1. *Input layer:* There is one neuron in the input layer for each predictor variable. For the case of categorical variables *N*-1 neurons are used where *N* is the number of categories. Input neurons (or processing before the input layer) standardizes the range of the values by subtracting the median and dividing by the inter quartile range. Then input neurons feed the values to each of the neurons in the hidden layer.

2. *Hidden layer:* This layer has one neuron for each case in the training data set. Neuron stores the values of the predictor variables along with the target value for the case. When presented with the *x* vector of input values from the input layer, the Euclidean distance of the test case from the neuron's center point is computed by hidden neuron and then using the sigma value(s) apply the RBF kernel function. Resulting value is passed to the neurons in the pattern layer.
3. *Pattern layer / Summation layer:* The next layer in the network is different for PNN and for GRNN. For PNN networks for each category of the target variable there is one pattern neuron. The actual target category of each training case is stored with each hidden neuron; the weighted value coming out of a hidden neuron is fed only to the pattern neuron that corresponds to the hidden neuron's type. Pattern neurons add the values for the class they represent (hence, it is a weighted vote for that category).
   For GRNN networks, in the pattern layer there are only two neurons. One neuron is the denominator summation unit the other is the numerator summation unit. The weight values coming from each of the hidden neurons adds up by denominator summation unit. Numerator summation unit adds up the weight values multiplied by the actual target value for each hidden neuron.
4. *Decision layer:* The decision layer is different for PNN and GRNN. For PNN networks the decision layer compares the weighted votes for each target category accumulated in the pattern layer and uses the largest vote to predict the target category.

PNN is pattern classification algorithm which falls into the broad class of "nearest-neighbor-like" algorithms [6]. Although the implementation is very different, PNN are conceptually similar to *K-Nearest Neighbor* (k-NN) models. Basic idea is that a predicted target value of an item is likely to be about the same as other items that have close values of the predictor variables. Let's consider this figure:

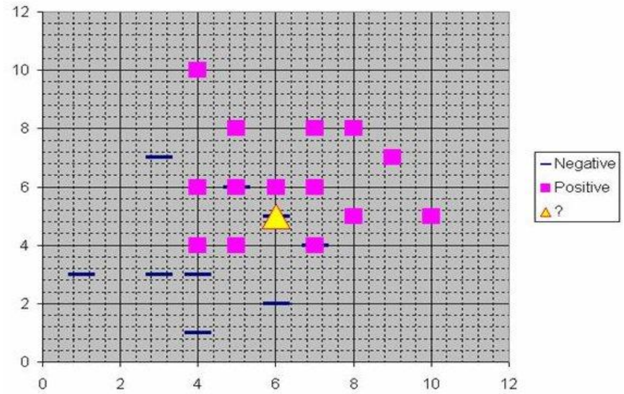


Fig 3: Different cases are plotted using *x* and *y* coordinates

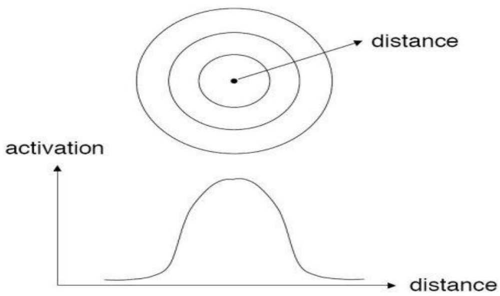
Assume that each case in the training set has *x* and *y* these two predictor variables. There cases are plotted using their *x* and *y* coordinates as shown in the figure. We can also assume that the target variable has two categories, first is negative which denoted by a dash and positive which is denoted by a square. Now, suppose we are trying to predict the value of a new case represented by the triangle with predictor values x=6 and y=5.1.

Notice that the triangle is position almost exactly on top of a dash representing a negative value. However that dash is in a fairly unusual position compared to the other dashes which are clustered below the squares and left of center. Therefore it could be that the underlying negative value is an odd case.

Depends on how many neighboring points are considered the nearest neighboring classification performed. The new point should be classified as negative since it is on top of a known negative point, if 1-NN is used and only the closest point is considered. Alternatively, if 9-NN classification is used and the closest 9 points are considered, at that time the effect of the surrounding 8 positive points may overbalance the close negative point.

A probabilistic neural network builds on this foundation and generalizes it to consider all of the other points. The PNN classifier has sometimes been accepted as belonging to the class of radial basis function [2]. The distance is computed from the point being evaluated to each of the other points, and a *radial basis function* (RBF) (also called a *kernel function*) is applied to the distance to compute the weight (influence). The radial basis function is so named because the radius distance is the argument to the function.

Weight = RBF (*distance*)



B. Radial Basis Function

Different types of radial basis functions might be used; however the most common is the Gaussian function:

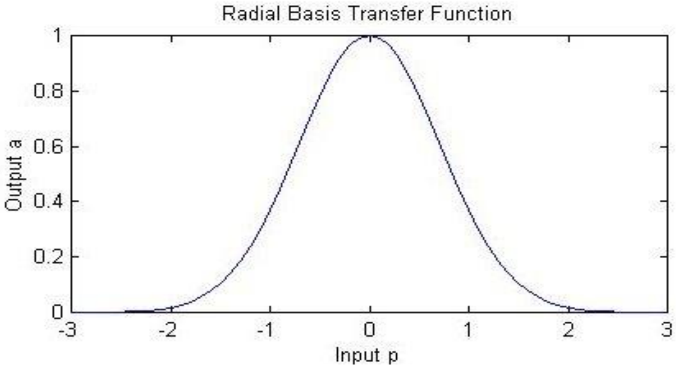


Fig : Radial based transfer function

The process based classification that differentiates PNN from RBF is that PNN works on estimation of probability density function (pdf) while RBF works on iterative function approximation [2].

C. Advantages of PNN networks:

1) It is usually much faster to train a PNN network than multilayer perceptron network.
2) PNN networks often are more accurate than multilayer perceptron networks.
3) PNN networks are relatively insensitive to outliers (wild points).
4) PNN networks create or generate accurate predicted target probability scores.
5) PNN networks approach Bayes optimal classification.

D. Disadvantages of PNN networks

1) PNN networks are slower than multilayer perceptron networks at classifying new cases
2) PNN networks want extra memory space to store the model

E. Removing unnecessary neurons

One of the disadvantages of PNN models compared to multilayer perceptron networks is that PNN models are large due to the fact that there is one neuron for each training line. Due to this the model to run slower than multilayer perceptron networks when using scoring to predict values for new rows.

Removing unnecessary neurons has three merits

1) The size of the stored model is reduced.
2) The time required to apply the model during scoring is reduced.
3) Removing neurons often improves the accuracy of the model.

F. Methodology:

A description of the derivation of the PNN classifier was specified. PNNs had been used for classification troubles. The PNN classifier presented very small training time, good accuracy, negligible retraining time and robustness to weight changes. There are 6 stages involved in the proposed model which are starting from the data input to output. First stage is should be the image processing system. Essentially in image processing system acquisition of image and enhancement of image are the steps that have to do. The proposed model requires converting the image into a format capable of being manipulated by the computer. MATLAB uses to convert the Images. Then the PNN is used to classify the images. Finally, performance based on the result will be analyzed at the end of the development phase.

IV. CONCLUSION

In this study we have seen basics of PNN classifiers. As well as we have seen its methodology and how it operates. PNN has more advantages over the other types of neural network classifiers. It gives satisfactory classification accuracy. After removing unnecessary neurons we get the speed of operation equal to the MLP. PNN has more accuracy over the other types of Neural Network classifiers.

REFERENCES

[1] K. Z. Mao, K. C. Tan and W. Ser "*Probabilistic Neural-network Structure Determination for Pattern Recognition* " IEEE Trans. On neural networks vol.11, No. 4, July 2000.

[2] Balasundaram Karthikeyan, Srinivasan Gopal, Srinivasan venkatesh, Subramanian saravanan, "*PNN and its Adaptive version-An Ingenious Approach to PD Pattern Classification compaired with BPA Network*" journal for electrical engineering, vol. 57, No. 3 , 2006, 138-145.

[3] Pooja Kamavisdar, Sonam Saluja, Sonu Agarwal, "*Survey on image classification Approaches and Techniques*" International journal of advanced research in computer and communication engineering, vol. 2, Issue 1, January 2013.

[4] Ameet Joshi, Shweta bapna, Sravanya Chunduri, "*Comparision Study of different pattern Classifiers*".

[5] Florin Gorunescu, "*Benchmarking Probabilistic neural network Algorithms* ", in proc.6th Intl. Conf. on Artificial intelligence and digital communications, Thessaloniki, Greece 2006 pp.1-7.

[6] David Montana, "A weighted Probabilistic Neural Network", NIPS4 1991: 1110-1117.

[7] Romeu Ricardo da silva and Domingo Mery, "*State of the art of Weld Seam Radiogarphic Testing Part II – Pattern Recognition*", www.ndt.net.

[8] Vincent cheung , Kevin cannons , "*An introduction to Probabilistic Neural Networks*", June 10 2002.

[9] Chapter 11 "*Image Processing-Classification*", www.jars1974.net.

[10] "*Multilayer Perceptron Neural Network*", www.dtreg.com.

