A PROBABILTY NEURAL NETWORK FOR CONTINUOUS AND CATEGORICAL DATA

Shuang Cang[1] and Hongnian Yu[2]

[1] *Department of Computer Science, University of Wales, Aberystwyth, Y23 3DB, UK*

[2] *Faculty of Computing, Engineering and Technology, Staffordshire University, Stafford, ST18 0DG, UK*

Abstract: In most application of the data classifications, the data sets contain both continuous and categorical variables. In other word, multivariate data sets containing mixtures of continuous and categorical variables arise frequently in practice. This paper presents a novel Probability Neural Network (**PNN**) which can classify the data for both continuous and categorical input data types. The case with either continuous or categorical input variables is a special case of the mixtures of continuous and categorical input variables. Therefore, the proposed **PNN** can be also applied to these two special cases. Expectation Maximisation (**EM**) algorithm is widely used for mixture models of continuous variables, but not applicable for categorical variables. A mixture model of continuous and categorical variables is used to construct a Probability Density Function (**PDF**) which is the key part for the **PNN**.

The proposed **PNN** has two advantages comparing with the conventional algorithms such as the Multilayer Perceptron (**MLP**) Neural Network. One advantage is that the **PNN** can produce better results comparing with the **MLP** Neural Network, even using the normalized input variables for the **MLP**. Normally, the normalized input variables generate a better result than the non-normalized input variables for the **MLP** Neural Network. Another advantage is that the **PNN** does not need the cross validation data set and does not produce the over training like the **MLP** neural network does. These have been proven in our experimental study. The proposed **PNN** can also be used to perform the unsupervised cluster analysis. The superiority of **PNN** in comparing the **MLP** neural network is demonstrated by applying them to a real-life data set, the Trauma data set which includes both continuous and categorical variables.

Keywords: Neural Networks, Probability Density Function (**PDF**), Classification, Pattern Recognition, Mixture Models, Expectation Maximisation (**EM**) algorithm.

1. INTRODUCTION

Probability density functions (**PDF**) play an important role in pattern recognition. If we know **PDF** for each class, then the probability of the new pattern belonging to each class can be obtained by using the Bayes' rule. Mixture models (Yuille 1994, Michalis 2001, Bishop 1995) are widely used to approximate a true density function for continuous variables. The Parzen window estimator (Parzen 1962) is a fundamental technique for estimating **PDF**. The mixture model for the binary variables was studied by Yang (2001). However, it is quite often that the data set of a real application contains both continuous variables and categorical variables with values 0,1,2,…,m ($m > 0$). It is noted that the binary variables are the special case of categorical variables (m=1).

This paper presents a new approach of finding **PDF** for a mixture of continuous and categorical input variables, which we call the probability neural network (**PNN**) for the mixtures of continuous and categorical input variables. Up to now, most of the works are dealing with either the binary variables (m=1) or continuous variables which are special cases of our approach. The number of components can be determined by using the algorithms proposed

by Cang (2001), which has successively be applied for determining the number of components in **PDF**.

This paper consists of five sections. In section 2, the widely used Expectation Maximisation (**EM**) algorithm is briefly described. Section 3 presents a new mixture model for the mixture of continuous and categorical variables. The probability neural network and the procedures of training probability neural network are presented in Section 4. An experimental result is presented in Section 5. The conclusions are presented in Section 6.

2. GAUSSIAN MIXTURE MODELS FOR CONTINUOUS VARIABLES

The EM algorithm is widely used to estimate parameters in mixture models for continuous variables. For M components, the mixture density for a d dimensional vector X can be written as a linear combination of component density functions $p(X|j)$ in the form

$$p(X) = \sum_{j=1}^{M} p(X \mid j)P(j) \qquad (2.1)$$

where $P(j)$ are the parameters in the mixture model and satisfy the following conditions

$$\sum_{j=1}^{M} P(j) = 1\,,\ 0 \le P(j) \le 1 \qquad (2.2)$$

The component density functions $p(X|j)$ in (2.1) satisfy

$$\int_{-\infty}^{\infty} p(X \mid j)dX = 1 \qquad (2.3)$$

The most widely used distribution for each component density is the Gaussian distribution. The Gaussian mixture model is only applicable for continuous variables. The form of the Gaussian density function for each component is

$$p(X|j) = \frac{1}{(2\pi)^{d/2} \mid \Sigma_j \mid^{1/2}} e^{-(X-\mu_j)^T \Sigma_j^{-1}(X-\mu_j)/2} \qquad (2.4)$$

where the parameters μ_j and Σ_j are the means of a d-dimensional vector and a $d \times d$ covariance matrix, respectively. Values for the parameters $P(j)$, μ_j and Σ_j can be determined in each component using the EM algorithm (Bishop 1995) as follows.

First, a K-means clustering method is used for a fixed number of components M in (2.1) to determine parameters $P(j)$, μ_j and Σ_j for each component $p(X|j)$ in (2.4). Clearly the condition (2.3) is satisfied. Then, the parameters $P(j)$, μ_j and Σ_j are obtained for each component using the following recursive formulas.

$$P(j) = \frac{1}{N}\sum_{n=1}^{N} w_j^n, \qquad \mu_j = \frac{\sum_{n=1}^{N} w_j^n X^n}{\sum_{n=1}^{N} w_j^n},$$

$$\Sigma_j = \frac{\sum_{n=1}^{N} w_j^n (X^n - \mu_j)(X^n - \mu_j)'}{\sum_{n=1}^{N} w_j^n}, \qquad (2.5)$$

where N is the size of the data set and the weight is

$$w_j^n = \frac{p(X^n \mid j)P(j)}{\sum_{j=1}^{M} p(X^n \mid j)P(j)} \qquad (2.6)$$

$p(X^n|j)$ is defined in (2.4). Iterate E-step, M-step, stable point for the parameters in the mixture model can be reached.

The **EM** algorithm presented above requires that all the variables are continuous, and the **EM** can not be applied to the cases with a mixture of continuous and categorical variables. We propose a modified **EM** algorithm which can deal with the case for a mixture of continuous and categorical variables in next section.

3. MIXTURE MODELS FOR CONTINUOUS AND CATEGORICAL VARIABLES

3.1 Mixture Models

In (2.1), we assume that X is a d-dimensional vector, which contains d_c continuous variables, d_1 binary variables, d_2 variables with 3 categorical values, and in general, d_{m-1} variables with m categorical values, thus, $d = d_c + \sum_{i=1}^{m-1} d_i$. The EM algorithm described in section 2 can not be applied directly. We propose a new model to handle this issue. The X can be represented as

$$X = [x_1^{(c)}, \cdots, x_{d_c}^{(c)}, x_1^{(2)}, \cdots, x_{d_1}^{(2)}, \cdots, x_1^{(m)}, \cdots, x_{d_{m-1}}^{(m)}]\,.$$

Assuming that the categorical variables are independent of each other, for a mixture of continuous and categorical variables in mixture model $p(X)$ (2.1), we propose to represent component $p(X|j)$ as the following form

$$p(X/j) = p(X_c/j) \prod_{i_1=1}^{d_1} p_{ji_1 0}^{f_0^{(1)}(x_{i_1})} (1 - p_{ji_1 0})^{f_1^{(1)}(x_{i_1})}$$

$$\prod_{i_2=1}^{d_2} p_{ji_2 0}^{f_0^{(2)}(x_{i_2})} p_{ji_2 1}^{f_1^{(2)}(x_{i_2})} (1 - p_{ji_2 0} - p_{ji_2 1})^{f_2^{(2)}(x_{i_2})}$$

$$\cdots$$

$$\prod_{i_m=1}^{d_{m-1}} p_{ji_m 0}^{f_0^{(m)}(x_{i_m})} p_{ji_m 1}^{f_1^{(m)}(x_{i_m})} \cdots p_{ji_m m-1}^{f_{m-1}^{(m)}(x_{i_m})} (1 - \sum_{k=0}^{m-1} p_{ji_m k})^{f_m^{(m)}(x_{i_m})} \qquad (3.1)$$

where X_c are the continuous variables and $f_k^{(m)}(x)$ (k = 0,1,2,…,m) is a function of variable x. We can show that the component density functions $p(X|j)$ satisfy $\sum \cdots \sum \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} p(X \mid j)dX = 1$. Two special cases are:

- If the variables are all continuous variables, (3.1) reduces to $p(X|j)=p(X_c|j)$ and $p(X_c|j)$ satisfies Gaussian distribution (2.4).

- If the variables are all binary variables, (3.1) reduces to $p(X \mid j) = \prod_{i_1=1}^{d_1} p_{ji_10}^{f_0^{(1)}(x_{i_1})}(1-p_{ji_10})^{f_1^{(1)}(x_{i_1})}$.

Therefore, the proposed model (3.1) includes the continuous or binary models, which are widely applied as a special case. Next, we show that how to determine the parameters in (3.1) for a mixture of continuous and categorical variables (any number of categorical variables). The general form of the functions $f_k^{(m)}(x)$ is investigated in section 3.2.

3.2 Determining $f_k^{(m)}(x)$ in Mixture Model (3.1)

In this section, we will develop a general form for the functions $f_k^{(m)}(x)$ in (3.1).

For m = 1, $f_k^{(m)}$(x) is a binomial probability distribution, thus $f_0^{(1)}$(x)=1-x, $f_1^{(1)}$(x)=x. For m = 2, $f_0^{(2)}(x) = 1-\frac{3}{2}x+\frac{1}{2}x^2$, $f_1^{(2)}$(x)=2x-x^2 and $f_2^{(2)}(x) = -\frac{1}{2}x+\frac{1}{2}x^2$.

In general, functions $f_k^{(m)}(x)$ (k = 0,1, 2,…,m) can be represented as an m order polynomial

$$f_k^{(m)}(x) = a_{k0}^{(m)} + a_{k1}^{(m)}x + a_{k2}^{(m)}x^2 + ... + a_{km}^{(m)}x^m \quad (k = 0, 1, 2,\ldots,m) \qquad (3.2)$$

We can rewrite the function $f_k^{(m)}(x)$ (k = 0,1, 2,…,m) as follows:

$$f_k^{(m)}(x) = A_k^{(m)}X^T = (a_{k0}^{(m)}, a_{k1}^{(m)}, a_{k2}^{(m)}, ...a_{km}^{(m)})\begin{pmatrix}1\\ x\\ \vdots\\ x^m\end{pmatrix} \qquad (3.3)$$

where $A_k^{(m)} = (a_{k0}^{(m)}, a_{k1}^{(m)}, a_{k2}^{(m)}, ...a_{km}^{(m)})$ and $X = (1 \quad x \quad \cdots \quad x^m)$.

If variable x takes the categorical value i, $i \in \{0,1,2,\cdots,m\}$, then let $f_k^{(m)}(x=i) = \begin{cases}1 & k=i\\ 0 & k\neq i\end{cases}$. So the coefficients $a_{ki}^{(m)}$ (i = 0,1,2,…,m and k = 0,1,2,…,m) satisfy the following equations.

$$\tilde{X}_{(m+1)^2\times(m+1)^2} \times \tilde{A}_{(m+1)^2\times 1} = \tilde{C}_{(m+1)^2\times 1} \qquad (3.4)$$

where

$$\tilde{X} = \begin{bmatrix}
1 & x_0 & \cdots & x_0^m & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 1 & x_0 & \cdots & x_0^m & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & x_0 & \cdots & x_0^m\\
1 & x_1 & \cdots & x_1^m & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 1 & x_1 & \cdots & x_1^m & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & x_1 & \cdots & x_1^m\\
\cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots\\
1 & x_m & \cdots & x_m^m & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 1 & x_m & \cdots & x_m^m & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & x_m & \cdots & x_m^m
\end{bmatrix}$$

$$\tilde{A} = \begin{bmatrix}A_0^{(m)} & A_1^{(m)} & \cdots & A_m^{(m)}\end{bmatrix}^T = \begin{bmatrix}a_{00}^{(m)}\\ a_{01}^{(m)}\\ \vdots\\ a_{0m}^{(m)}\\ a_{10}^{(m)}\\ a_{11}^{(m)}\\ \vdots\\ a_{1m}^{(m)}\\ \vdots\\ \vdots\\ \vdots\\ \vdots\\ a_{m0}^{(m)}\\ a_{m1}^{(m)}\\ \vdots\\ a_{mm}^{(m)}\end{bmatrix} \text{ and } \tilde{C} = \begin{bmatrix}1\\ 0\\ \vdots\\ 0\\ 0\\ 1\\ \vdots\\ 0\\ \vdots\\ \vdots\\ \vdots\\ \vdots\\ 0\\ 0\\ \vdots\\ 1\end{bmatrix},$$

and x_i (i = 0,1,2,…,m) are the elements of matrix $\tilde{X}$ and are discrete values, such as x_i=i.

For example, if m=1, x_0=0 and x_1=1 then $\tilde{X} = \begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 0 & 1 & 0\\ 1 & 1 & 0 & 0\\ 0 & 0 & 1 & 1\end{bmatrix}$ and $\tilde{C} = \begin{bmatrix}1\\ 0\\ 0\\ 1\end{bmatrix}$

If m = 2, x_0=0, x_1=1 and x_2=2, then $\tilde{X} = \begin{bmatrix}1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0\\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1\\ 1 & 2 & 4 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 1 & 2 & 4 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 2 & 4\end{bmatrix}$ and $\tilde{C} = \begin{bmatrix}1\\ 0\\ 0\\ 0\\ 1\\ 0\\ 0\\ 0\\ 1\end{bmatrix}$.

The coefficients of vector $\tilde{A}$ can be obtained using equation (3.4)

$$\tilde{A} = \tilde{X}^{-1}\tilde{C} \qquad (3.5)$$

The coefficients $A_k^{(m)} = (a_{k0}^{(m)}, a_{k1}^{(m)}, a_{k2}^{(m)}, ...a_{km}^{(m)})$ (k=1,2,…,m) for any m can be calculated by using (3.5). It is noted that equation (3.5) is easy to apply. We can show that the inverse of $\tilde{X}$ exists and the

computation is not an issue since it will be computed off-line.

3.3 Maximum Likelihood and EM Algorithm for Mixture Model

In this section, we determine the parameters in the *jth* (j=1,2,…,M) component defined in (3.1). Gaussian components are used for the continuous variables. Then the mixture model (3.1) contains the following adjustable parameters: $P(j)$, μ_j and Σ_j from Gaussian components, $p_{ji_1 0}$ (i_1=1,2,…,d_1) from binary components, $p_{ji_2 0}$ and $p_{ji_2 1}$ (i_2=1,2,…,d_2) from ternary values components, and so on. The negative log-likelihood (Bishop, 1995) for the set of N patterns $\{X_n, n=1,2,\ldots,N\}$ is

$$E = -\sum_{n=1}^{N} \ln p(X^n) = -\sum_{n=1}^{N} \ln\{\sum_{j=1}^{M} p(X^n \mid j)P(j)\} \quad (3.6)$$

From (3.6), it is easy to see that maximizing the likelihood $\sum_{n=1}^{N} p(X^n)$ is equivalent to minimizing E.

Setting the derivatives with respect to each parameter in (3.6) to zero, we obtain

$$P(j) = \frac{1}{N}\sum_{n=1}^{N} w_j^n,\ \mu_j = \frac{\sum_{n=1}^{N} w_j^n X_c^n}{\sum_{n=1}^{N} w_j^n},$$

$$\Sigma_j = \frac{\sum_{n=1}^{N} w_j^n (X_c^n - \mu_j)(X_c^n - \mu_j)^{'}}{\sum_{n=1}^{N} w_j^n} \quad (3.7)$$

where N is the size of the data set and the weight is

$$w_j^n = p(j / X^n) = \frac{p(X^n \mid j)P(j)}{\sum_{j=1}^{M} p(X^n \mid j)P(j)} \quad (3.8)$$

where $p(X^n|j)$ is defined in (3.1) as $p(X|j)$.

It is noted that equations (3.7) and (3.8) are the same as equations (2.5) and (2.6), but the calculation of weights w_j^n defined in (3.8) need X^n which is a mixture of continuous and categorical variable in (3.7) and (3.8).

For a general categorical value which has m parameters $p_{ji_m 0}, p_{ji_m 1}, \cdots, p_{ji_m m-1}$ (i_m=1,2,…,d_m, k = 0,1…,m and j = 1,2,…,M) in (3.1), setting the derivatives with respect to each of the parameters $p_{ji_m 0}, p_{ji_m 1}, \cdots, p_{ji_m m-1}$ in (3.6), we obtain the following equations

$$\sum_{n=1}^{N} w_j^n \left(\frac{f_0^{(m)}(x_{i_m})}{p_{ji_m 0}} - \frac{f_m^{(m)}(x_{i_m})}{1-\sum_{k=0}^{m-1} p_{ji_m k}}\right) = 0$$

$$\sum_{n=1}^{N} w_j^n \left(\frac{f_1^{(m)}(x_{i_m})}{p_{ji_m 1}} - \frac{f_m^{(m)}(x_{i_m})}{1-\sum_{k=0}^{m-1} p_{ji_m k}}\right) = 0 \quad (3.9)$$

…

$$\sum_{n=1}^{N} w_j^n \left(\frac{f_{m-1}^{(m)}(x_{i_m})}{p_{ji_m m-1}} - \frac{f_m^{(m)}(x_{i_m})}{1-\sum_{k=0}^{m-1} p_{ji_m k}}\right) = 0$$

Defining $F_k^{(m)}(x) = \sum_{n=1}^{N} w_j^n f_k^{(m)}(x)$ (k = 0, 1, 2,…,m), from equation (3.9), we obtain

$$p_{ji_m k} = \frac{F_k^{(m)}(x_{i_m})}{\sum_{u=0}^{m} F_u^{(m)}(x_{i_m})} \quad (3.10)$$

where i_m=1,2,…,d_m, k = 0,1…,m and j = 1,2,…,M.

Notice that $\sum_{k=0}^{m} p_{ji_m k} = 1$. For case m = 1, which is a binomial probability distribution, we have

$$1 - p_{ji_1 0} = \frac{\sum_{n=1}^{N} w_j^n x_{i_1}^n}{\sum_{n=1}^{N} w_j^n} \quad (i_1=1,2,\ldots,d_1) \quad (3.11)$$

In the modified **EM** algorithm for the maximum likelihood (3.6), we can use (3.7), (3.8) and (3.10) to update the parameters. The mixture model for continuous and categorical Variables is determined.

4. PROBABILITY NEURAL NETWORK

The architecture of probability neural network is described in Figure 1. Typically, this probability neural network has d inputs and k outputs (one for each class). The main different with a classical neural network lies on the specific functional form of the base functions which are considered to be density functions for each class as well as on some constraints involving from the hidden to the output layer.

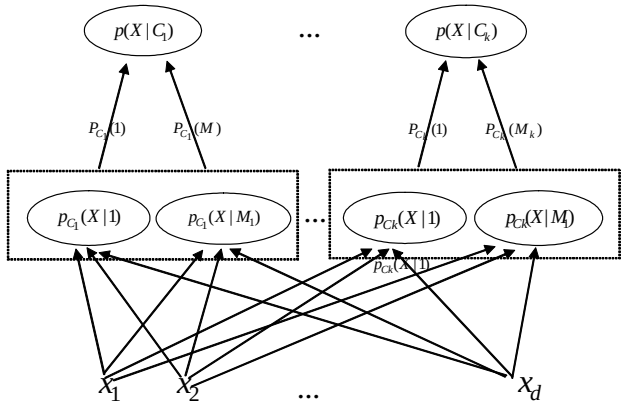

Figure 1: Probability Neural Network

Following the discussion in section 3 and Figure 1, we summarise the computing algorithm for (3.1) as below.

Modified EM Algorithm:

(1). For the supervised learning, divide the data set into two parts, the training and test data sets. The size of the training data set is *N*.

(2). Partition the training data set for each class. Thus we obtain a number of sub-training data sets. Each sub-training data set belongs to one class. The size of the sub-training data set is N_{C_k} for class C_k ($k \in [1,2,\cdots,K]$) and *K* is a total number of classes.

(3). For each sub-training data set which belongs to one class, determining the number of components using the algorithms proposed by Cang (2001) and then compute the parameters for each components in (3.1) using (3.7), (3.8) and (3.10). Then obtain the mixture models $p(X/C_k)$ defined in (2.1) for each class, where C_k indicates class *k*.

(4). Compute the prior probability for each class $P(C_k)$, where $P(C_k) = \frac{N_{C_k}}{N}$.

(5). For each new pattern *X*, (pattern in test data set), compute the values $p(X/C_k)P(C_k)$ ($k \in [1,2,\cdots,K]$) and determine $X \in C_k$ by taking $\max(p(X/C_k)P(C_k))$.

5. EXPERIMENT STUDY

In this section, we apply the new **PNN** to a realistic data set (Trauma data set) and we also compare **PNN** with the standard **MLP** neural network. The cross validation method is used for these problems. The data is partitioned into a fixed number (*N*) of partitions or folds, each fold is hold out as test data in turn, while the other *N*-1 folds are measured and then the *N* estimates are averaged to give a final accuracy.

In the following analyses, the definitions for the confusion matrix, sensitivity, specificity, true positive rate (TPR) and false positive rate (FPR) for two classes are given as follows.

Table 1: The confusion matrix for two classes:

	True Class Label		
Prediction Class Label		Class 1	Class 2
	Class 1	A_{11}	A_{12}
	Class 2	A_{21}	A_{22}

where Sensitivity = Number of true positive decisions/Number of actually positive cases = $A_{22}/(A_{12}+A_{22})$.

Specificity = Number of true negative/Number of actually negative cases = $A_{11}/(A_{11}+A_{21})$.

True positive rate (TPR) = Sensitivity.

False positive rate (FPR) = 1 – Specificity.

The Helicopter Emergency Medical Service (HEMS) attached to the Royal London Hospital has gathered data from pre-hospital trauma patients over a ten-year period. The size of the trauma data is 1044 excluding 321 patterns that contain missing data somewhere. The outcome is a *lived/died* prediction on individual patients. The trauma data is an unbalanced data with only 158 *died* cases among 1044 patterns. There are 16 features including 5 continuous features in this data set. The features are described as in Table 2, where 'Con' indicates a Continuous feature and 'Cat' Indicates a categorical feature.

Table 2: Features in the trauma data set

Name	Type	Values	Description
Age	Con.		Age from 0 to 100
Gender	Cat.	0 1	Male=1,Female=0
Injury Type	Cat.	0 1	Blunt=1,Penetrating=0
Head	Cat.	0 1 2 3 4 5 6	Head injury
Facial	Cat.	0 1 2 3 4	Facial injury
Chest	Cat.	0 1 2 3 4 5 6	Chest injury
Abdominal	Cat.	0 1 2 3 4 5	Abdominal or pelvic contents injury
Limbs	Cat.	0 1 2 3 4 5	Limbs or bony pelvis injury
External	Cat.	0 1 2 3	External injury
Respiration Rate	Con.		Respiration rate
Systolic Blood	Con.		Systolic blood pressure
GCS Eye	Cat.	0 1 2 3 4	Glasgow coma score (GCS) eye response
GCS Motor	Cat.	0 1 2 3 4 5 6	GCS motor response
GCS Verbal	Cat.	0 1 2 3 4 5	GCS verbal response
Oximetry	Con.		Oximetry (% red blood cell O2 saturation)
Heart Rate	Con.		Heart rate
Class	Cat.	0, 1	Classification , 1 is died and 0 is lived.

In order to test every single pattern and make comparisons for these methods, we divided the trauma data into five almost equally size fold without overlapping each other. The sizes of each fold are 208, 208, 208, 210 and 210, respectively. We used one fold as the test data set and the rest of four fold as the training data set in turn. This has guaranteed that every pattern in the trauma data would test once. This random partitioning was done 10 times. For each partition data, the means of the sensitivity, specificity and classification rates for all 5 folds of the test data are calculated by using two methods. One is the new **PNN**. The other method is the standard **MLP** neural network with one hidden layer. The results of the sensitivity, specificity and classification rate set are shown in Table 3 for the test data.

Table 3: The training and test performance results on each data partition (there are 10 random partitions) (SE=sensitivity, SP=specification, CL=classification)

Training Data					
Probability neural network (**PNN)**			Multilayer Perceptron neural network (**MLP)**		
SE	**SP**	**CL**	**SE**	**SP**	**CL**
0.7489	0.9410	0.9119	0.8157	0.9786	0.9538
0.7639	0.9467	0.9191	0.8129	0.9771	0.9523
0.7641	0.9427	0.9157	0.8321	0.9796	0.9571
0.7561	0.9458	0.9172	0.8309	0.9814	0.9586
0.7790	0.9422	0.9174	0.8522	0.9842	0.9643
0.7806	0.9444	0.9195	0.8037	0.9774	0.9512
0.7758	0.9402	0.9152	0.8243	0.9813	0.9576
0.7705	0.9441	0.9179	0.8496	0.9817	0.9617
0.7876	0.9438	0.9200	0.8355	0.9785	0.9569
0.7701	0.9407	0.9150	0.7731	0.9766	0.9459
Mean					
0.7697	**0.9432**	**0.9169**	**0.8230**	**0.9796**	**0.9559**
Standard Division					
0.0118	**0.0022**	**0.0025**	**0.0233**	**0.0024**	**0.0053**
Test Data					
Probability neural network (**PNN)**			Multilayer Perceptron neural network (**MLP)**		
SE	**SP**	**CL**	**SE**	**SP**	**CL**
0.6178	0.9241	0.8773	0.7386	0.8808	0.8573
0.6187	0.9336	0.8869	0.6956	0.8876	0.8572
0.5991	0.9277	0.8792	0.7239	0.8841	0.8562
0.6269	0.9256	0.8813	0.7097	0.8847	0.8592
0.6313	0.9268	0.8822	0.6706	0.8927	0.8611
0.6128	0.9223	0.8745	0.6803	0.8888	0.8554
0.6135	0.9454	0.8946	0.7095	0.8985	0.8697
0.6139	0.9313	0.8831	0.6991	0.8903	0.8564
0.5776	0.9293	0.8736	0.6635	0.8869	0.8554
0.5878	0.9211	0.8718	0.6907	0.8844	0.8536
Mean					
0.6099	**0.9287**	**0.8804**	**0.6981**	**0.8879**	**0.8581**
Standard Division					
0.0169	**0.0070**	**0.0068**	**0.0233**	**0.0051**	**0.0046**

From Table 3, we can see that our new approach, **PNN** with 88.04% classification rate in overall mean, give a better result for the test data set than the **MLP** neural network with 85.81%, even use the nomalized input variables for **MLP** neural network. For the training data, we can see that there is a over training for **MLP** neural network, while there is no problems on this matter for the **PNN**.

6 CONCLUSIONS

This paper has proposed a new probability neural network (**PNN**) for a mixture of continuous and categorical variables inputs. We have used a realistic data set (Trauma data set) to demonstrate that the proposed method is more reliable, and more accurate than the standard **MLP** neural network. It can be applied in many problems. The main problem for the **MLP** neural network is the over training, while the **PNN** can overcomes this problem, and the **PNN** does not need to use the cross validation data set.

The **PNN** can be applied to multi-class and high dimensional data set. In order to reduce computation time for a high dimensional data set, we can reduce dimension by using wider range of feature selection algorithms or principal components analysis (**PCA**) first, then apply **PNN** to this reducing dimensional data set as input.

The **PNN** can be used as unsupervised learning as well. We can obtain the clusters without considering the class label and measure the similarity and different among the clusters. The number of the clusters can be determined by the algorithms proposed by Cang (2001). We found that for unsupervised learning using the **PNN** is more accurate and more robust than the principle component analysis (**PCA**) for the most complicate data sets.

REFERENCES

S. Cang and D. Partridge (2001), Determining the number of components in mixture models Using Williams' Statistical Test, Proc. of the 8th International Conference on Neural Information Processing, China, 828-834.

Parzen E. (1962) On estimation of a probability density function and mode. Annals of Mathematics Statistics, **3**, 1065-1076.

Yuille A. L., Stolorz P. and Utans J. (1994) Statistical physics, mixtures of distributions, and EM algorithms. Neural Computation, **6**, 334-340.

Michalis K. Titsias and Aristidis C. Likas (2001) Shared kernel models for class conditional density estimation, IEEE Transactions on Neural Networks, VOL. 12, NO. 5, 987-997.

Z. R. Yang (2001) A binary probabilistic model and genetic algorithm for HIV protease cleavage sites prediction and search, Proc. of the 8th International Conference on Neural Information Processing, China,847-852.

Y. Young and G. Coraluppi (1970), Stochastic estimation of a mixture of normal density functions using an information criterion. IEEE Trans. On Information Theory, **16**, 258-263.

Carreira-Perpinan, M. A. (2000), Mode-finding for mixtures of Gaussian distributions, IEEE Trans. on Pattern Analysis and Machine Intelligence **22** (**11**), 1318-1323.

Richardson, S. and Green,P.J (1997) On Bayesian analysis of Mixtures with an Unknown Number of Components. Journal of the Royal Statistical Society, B59, 731-792.

Detrano (1989) American Journal of Cardiology, **64**, 304-310.

C. Bishop (1995), Neural networks for pattern recognition, Oxford Press.