NEURAL ARCHITECTURES FOR PARAMETRIC ESTIMATION OF A POSTERIORI PROBABILITIES BY CONSTRAINED CONDITIONAL DENSITY FUNCTIONS

Juan Ignacio Arribas†, Jesus Cid-Sueiro†, Tülay Adali†† and Anibal R. Figueiras-Vidal†††

†Dept. of Teoría de la Señal, Comunicaciones e Ing. Telemática, Universidad de Valladolid, Valladolid, Spain.

†† Dept. of Computer Science and Electrical Engineering, University of Maryland Baltimore County, Maryland, USA.

††† Dept. of Tecnologías de las Comunicaciones, Universidad Carlos III de Madrid, Madrid, Spain.

Voice: +34 983 423000, Fax: +34 983 423667, email: jarribas@tel.uva.es

Abstract. A new approach to the estimation of 'a posteriori' class probabilities using neural networks, the Joint Network and Data Density Estimation (JNDDE), is presented in this paper. It is based on the estimation of the conditional data density functions, with some restrictions imposed by the classifier structure; the Bayes' rule is used to obtain the 'a posteriori' probabilities from these densities. The proposed method is applied to three different network structures: the logistic perceptron (for the binary case), the softmax perceptron (for multi-class problems) and a generalized softmax perceptron (that can be used to map arbitrarily complex probability functions). Gaussian mixture models are used for the conditional densities. The method has the advantage of establishing a distinction between the network architecture constraints and the model of the data, separating network parameters and the model parameters. Complexity on any of them can be fixed as desired. Maximum Likelihood gradient-based rules for the estimation of the parameters can be obtained. It is shown that JNDDE exhibits a more robust convergence characteristics than other methods of a posteriori probability estimation, such as those based on the minimization of a Strict Sense Bayesian (SSB) cost function.

INTRODUCTION

It is well known that when a neural network is trained in order to minimize a mean square error [1] or the cross entropy [2] between the targets and the network

0-7803-5673-X/99/$10.00 © 1999 IEEE

outputs, the network provides, after convergence, estimates of the 'a posteriori' probabilities of the classes; many applications in pattern recognition can take advantage of this property: medical diagnosis, financial data analysis, digital communications, among others. In [3], [5], [6], any cost function providing *a posteriori* class probabilities is called *Strict Sense Bayesian* (SSB). General conditions for SSB cost functions can be found in [3], [4] and [6] .

When the *a posteriori* probabilities are estimated using SSB cost functions, no previous assumptions about the data distribution are required: probabilities are estimated without computing the conditional density functions of the classes. It is known, however, that this poses some problems:

- The estimation of *a posteriori* probabilities in well-separated data requires very large training sets.
- Prior knowledge about the data distribution can not be used.
- It is difficult to use unlabeled data to improve learning.

These difficulties are of major importance in many real-world applications. In fact, although the probability estimation property is usually considered as a potential advantage of neural networks, it is rarely used in practice. An alternative to *a posteriori* probability estimation with SSB cost functions is Density Estimation (DE): it consists on estimating the conditional density functions of the data samples corresponding to different classes. Using Bayes' rule:

$$p(j|\mathbf{x}) = \frac{p(j) f(\mathbf{x}|j)}{\sum_{i=1}^{N} p(i) f(\mathbf{x}|i)} \qquad (1)$$

where N is the number of classes, $p(j)$ is the *a priori* probability of class j, $\mathbf{x}$ is an arbitrary point of the sample space and f denotes a density function; the computation of *a posteriori* probabilities from conditional densities is straightforward. Unlike SSB cost-based methods, the DE approach requires to specify some parametric model for the data densities; in spite of this, we will see that we can take some advantages of it.

In this paper we try to establish a link between DE and SSB-based methods: a density model is assumed, but it can be made as general as desired, without increasing the complexity of the classifier after learning. In the following, we will refer to this method as the *Joint Network and Data Density Estimation* (JNDDE).

The structure of the paper is as follows: first, we describe the basic ideas behind the new method, defining the *implicit set* concept; second, we consider the binary logistic problem, which is going to help us to gain some insight in understanding the nature of the JNDDE. Third, we discuss more complex networks, to make it suitable to more general problems and show two general structures: in the first one, we employ a simple multi-output architecture and the second one is a general classification structure that can map arbitrarily complex probability maps. Fourth,

we describe the results obtained through computer simulations. The last section states some conclusions.

FUNDAMENTALS

Consider the N-output neural network whose output is given by $\mathbf{y}=g(\mathbf{x},\mathbf{w})$, where $\mathbf{x}$ is the input and $\mathbf{w}$ is the parameter set. In order to interpret outputs as probabilities, we assume that

$$0 \le y_i \le 1, \quad i = 1,\ldots,N \quad \sum_{i=1}^{N} y_i = 1 \tag{2}$$

This restriction will be adopted in the following, but it is not strictly necessary. In a binary case, a single output network can be used to estimate the *a posteriori* probability of one of the classes, the other one is readily available by a simple subtraction. Also, a network with N independent outputs can be interpreted as a collection of N independent binary classifiers. According to Eq.(1), if the network is able to estimate the *a posteriori* class probabilities in some particular problem, it should be verified that

$$y_j = g_j(\mathbf{x},\mathbf{w}) = \frac{p(j)f(\mathbf{x}|j)}{\sum_{i=1}^{N} p(i)f(\mathbf{x}|i)} = p(j|\mathbf{x}) \quad j = 1,\ldots,N \tag{3}$$

DEFINITION: *Conditional density functions $\{f(\mathbf{x}|j),\ j=1...N\}$ form an implicit set for a network given by outputs $g_j(\mathbf{x},\mathbf{w})$ if equation (3) are satisfied for some parameter vector $\mathbf{w}$.*

From this definition, it is evident that a neural network can compute exactly the *a posteriori* probabilities of the classes for some parameter vector $\mathbf{w}$ if and only if the conditional density functions form an implicit set. For binary problems, word *set* may be substituted by *pair*, resulting an *implicit pair.*

The main idea of this paper is to establish a parametric model for the family of the possible implicit sets and afterwards to find the implicit set in this family that best fits the data in a Maximum Likelihood sense. The conditional densities will depend on the network parameters, but they may depend on an additional parameter set $\mathbf{v}$, hence we write $f(\mathbf{x}|i)=f_{\mathbf{w},\mathbf{v}}(\mathbf{x}|i)$. The use of over-parametric densities (i.e. with more parameters than the network itself) is essential to allow us to estimate complex data density models without increasing the complexity of the classifier structure. This will be made clear later.

Note that this approach is network-dependent: for each neural network a parametric model of their implicit density sets must be found. For networks with several hidden layers, this may be a difficult task; in this paper, we restrict our

analysis to a wide family of networks based on the logistic and the softmax functions.

THE BINARY LOGISTIC PROBLEM (LOP)

Sigmoidal perceptrons

We will describe our method starting with a simple case: the JNDDE binary *Logistic Problem* (Fig. 1). Let us consider a binary problem, with class labels 0 and 1, and a network that computes the logistic or sigmoid function given by:

$$y = \frac{1}{1+e^{-\mathbf{w}^t\mathbf{x}-w_0}} \tag{4}$$

where $\mathbf{x}$ is the sample vector, and $\mathbf{w}^t$ indicates the transposed vector of $\mathbf{w}$. In Fig. 1, we show the binary perceptron, consisting of a soft decision step (given by the sigmoid) and a hard decision step (given by the WTA), which assigns one to the winning class and zero to the other (arg max y_i), forming the output vector $\mathbf{d}$.

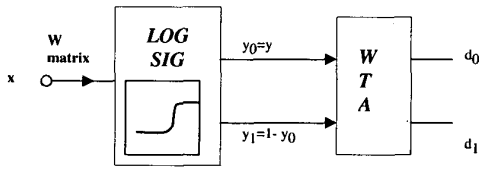

Fig. 1.- The binary Logistic network

We are interested here in training the logistic network in such a way that $y=p(1|\mathbf{x})$; The logistic non-linearity has been found by some authors ([8] is a good example) to be a specially suitable function for the case of Exponential Family distributions, (which includes the Gaussian distribution): it can be shown that the densities of the exponential family form implicit sets of the logistic network [2]. Here, we are interested in the class of all the implicit density sets for the logistic perceptron. It is provided by the following Theorem.

THEOREM 1: *Density functions $f_0(\mathbf{x})$ and $f_1(\mathbf{x})$ form an implicit set for the logistic perceptron network, if and only if they can be expressed in the form*

$$f_0(\mathbf{x}) = \frac{q_0}{p_0} e^{-\frac{\mathbf{w}^T\mathbf{x}+w_0}{2}} f_c(\mathbf{x}); \quad f_1(\mathbf{x}) = \frac{q_1}{p_1} e^{\frac{\mathbf{w}^T\mathbf{x}+w_0}{2}} f_c(\mathbf{x}) \tag{5}$$

where f_c is an arbitrary density function that will be called the Central Density.

Proof: The sufficient condition is straightforward: replacing (3) and (5) in the Bayes' rule, we get a logistic function of the input. The necessity condition can also be proved easily. If (5) holds, we can write

$$p_0 f_0(\mathbf{x}) e^{+\frac{\mathbf{w}^T\mathbf{x}+w_0}{2}} = p_1 f_1(\mathbf{x}) e^{-\frac{\mathbf{w}^T\mathbf{x}+w_0}{2}} \tag{6}$$

where p_1 and p_0 denote the *'a priori'* class probabilities, $f_0(\mathbf{x})=f(\mathbf{x}|0)$ and $f_1(\mathbf{x})=f(\mathbf{x}|1)$ are the conditional class probability density functions. Equation (6) shows the conditions that any implicit pair must verify. Defining

$$f_c(\mathbf{x}) \equiv \frac{p_0 f_0(\mathbf{x})}{q_0} e^{\frac{\mathbf{w}^T\mathbf{x}+w_0}{2}} = \frac{p_1 f_1(\mathbf{x})}{q_1} e^{-\frac{\mathbf{w}^T\mathbf{x}+w_0}{2}} \tag{7}$$

where q_0 , q_1 are some normalizing factors ensuring that $f_c(\mathbf{x})$ is a p.d.f. and. expressing f_0 and f_1 as functions of f_c, the proof is completed. ■

Selecting different central densities, we can generate arbitrary pairs of implicit density functions. For instance, if $f_c(\mathbf{x})$ is a zero-mean Gaussian function, we obtain a Gaussian pair.

Estimating implicit pairs

The JNDDE method is based on making hypotheses about the implicit pair and estimating its parameters using the data. It can be summarized as follows:

1. Assume a model the central p.d.f. $f_c(\mathbf{x},\mathbf{v})$ where $\mathbf{v}$ is the parameter vector to be estimated.
2. Express f_0 and f_1 as a function of $\mathbf{w}$ and $\mathbf{v}$
3. Obtain MLE of $\mathbf{w}$, $\mathbf{v}$. (Gradient-based learning rules can be obtained, but the details are not shown in this paper).
4. Use estimated $\mathbf{w}$ as the perceptron weights, in such a way that y are estimates of *a posteriori* probabilities.

The first step is a key point of the method. The central density should satisfy two important requirements:

1. Since the computation of the conditional densities from the central density requires to integrate the product of the central density by a exponential function, we must select a model f_c providing closed form solutions for these integrand operations.
2. The central density model should be general enough to map arbitrarily complex densities.

The first requirement is satisfied if $f_c(\mathbf{x},\mathbf{v})$ is a Gaussian function. It is easy to show that, in such a case, the class densities are also Gaussian functions, whose parameters depend on **w** and **v**. To satisfy the second requirement, we will assume that $f_c(\mathbf{x})$ is a *Gaussian Mixture*,

$$f_c(\mathbf{x}) = q_1 N(\mathbf{x}, \bar{v}_1) + ... + q_m N(\mathbf{x}, \bar{v}_m) \tag{8}$$

In such a case, both classes are Gaussian mixtures.

THE SOFTMAX PERCEPTRON (SP)

A similar analysis can be done for the multi-class problem. The softmax nonlinearity is the natural generalization of the logistic function for multiple-output problems, see [9] and [10]; consider the *Softmax Perceptron* (SP) in Fig. 2, whose outputs are given in (9).

$$y_j = softmax(\mathbf{w}_j{}^T \mathbf{x}) = \frac{e^{\mathbf{w}_j{}^T \mathbf{x}}}{\sum_{i=1}^{N} e^{\mathbf{w}_i{}^T \mathbf{x}}} = \frac{1}{1 + \sum_{i \neq j}^{N} e^{(\mathbf{w}_i - \mathbf{w}_j)^T \mathbf{x}}} \quad j = 1,..., N \tag{9}$$

where N is the number of outputs, i.e., number of classes, y_j is the output of our network for class j, $\mathbf{x}$ its input vector and $\mathbf{w}_j$ is the weight vector.

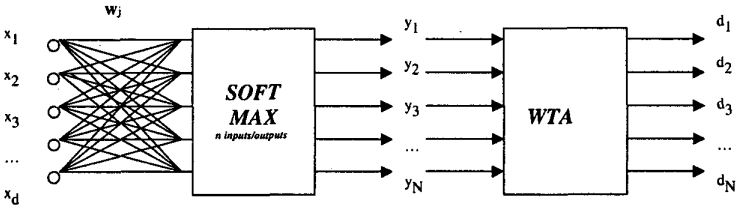

Fig. 2.- The multi-class SP network (**d** is the dimensionality of input data)

We wish y_j to be the *a posteriori* class probability of input sample **x** belonging to class j, so $f_j(\mathbf{x})$ would constitute an *implicit set.*

THEOREM 2: *Density functions $f_j(\mathbf{x})$ form an implicit set for the SP network, if and only if they can be expressed in the form*

$$f_j(\mathbf{x}) = \frac{f_c(\mathbf{x}) e^{\mathbf{w}_j{}^T \mathbf{x}}}{\int f_c(\mathbf{x}) e^{\mathbf{w}_j{}^T \mathbf{x}} d\mathbf{x}} \quad \forall j \tag{10}$$

where j is class index, x input sample, w network weights and $f_c(x)$ an arbitrary probability density function.

Proof: The proof is easy and follows the same steps than those in Theorem 1. ■

Thus, in (10) we have the conditional probability density functions for each class in terms of the *Central Density*, constrained by the chosen network structure, SP in this case. Later on, one would compute the partial derivatives for the gradient based learning rules for all the parameters, details not shown in this paper, to obtain Maximum Likelihood Estimates (MLE).

THE GENERALIZED SOFTMAX PERCEPTRON (GSP)

The SP network is not a universal classifier. A more general structure can be obtained by computing the class outputs as the sum of several softmax functions (see Fig. 3).

Again, applying Bayes' formula and defining outputs Z_i as estimates of *a posteriori* class probabilities, constituting densities f(x|i) an *implicit set*, we obtain:

$$Z_j \equiv \frac{\sum_{k=1}^{M_j} e^{\mathbf{w}_{j,k}{}^T \mathbf{x}}}{\sum_{i=1}^{N} \sum_{k=1}^{M_i} e^{\mathbf{w}_{i,k}{}^T \mathbf{x}}} = \frac{p_j f(\mathbf{x} \mid i)}{\sum_{i=1}^{N} p_i f(\mathbf{x} \mid j)} \quad j = 1, \ldots, N \tag{11}$$

where M_j are the number of elements within j-th class, also called the subclass number, and N is the number of classes.

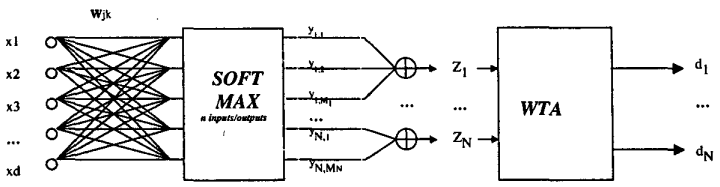

Fig. 3.- The GSP Network

***THEOREM 3**: Density functions $f_j(\mathbf{x})$ form an implicit set for the GSP network, if and only if*

$$f_j(\mathbf{x}) = \frac{f_c(\mathbf{x})\sum_{k=1}^{M_j} e^{\mathbf{w}_{j,k}{}^T\mathbf{x}}}{\int f_c(\mathbf{x})\sum_{k=1}^{M_j} e^{\mathbf{w}_{j,k}{}^T\mathbf{x}}\,d\mathbf{x}} \quad j=1,\ldots,N \tag{12}$$

where j is class index, M_j is the number of elements within j-th class, $\mathbf{x}$ the input sample, $\mathbf{w}$ the network weights and $f_c(\mathbf{x})$ an arbitrary probability density function.

Proof: The proof is easy and also omitted. Again, gradient based learning rules can be obtained which are use to recursively adapt all the parameters, both network and data for the GSP net. ■

RESULTS

We tested JNDDE through computer simulation. Here we consider just a simple example to illustrate the main difference between this method and those based on minimizing a SSB cost. Other simulation examples analyzing convergence speed of stochastic gradient algorithms can be found in [5], [6] and [11].

Consider a zero-mean Gaussian two-dimensional central density f_c with variance matrix $\sigma^2\mathbf{I}$. We took σ=1. Consider the implicit pair of conditional densities resulting from this central density and a logistic perceptron with parameter vector $\mathbf{w}$=(-1,3). Data were generated according to this density pair. The simulation work followed three steps

1) Estimate $\mathbf{w}$ minimizing a cross entropy cost function. Stochastic gradient learning was used. The algorithm was stopped after 16384 iterations. At this time the algorithm has almost reached the final convergence values.
2) Estimate $\mathbf{w}$ by MLE of the conditional densities. Since classes are Gaussian, the JNDDE approach reduces, in this case, to estimate mean, variances and a priori probabilities of each classes, which can be done using standard estimates.
3) Estimate $\mathbf{w}$ by MLE, assuming that the true variance of each class is known.

Simulations were carried out for different training set sizes: 4,8,16,32,... up to 2^{13}. Since all methods make true hypothesis about the data, all of them would obtain a perfect estimate of $\mathbf{w}$ if unlimited data were available. For these finite size examples, we compared the estimation performance using the *Weight Relative Square Error* (WRSE) measure given by

$$WRSE = \frac{\left|\mathbf{w}_{opt} - \mathbf{w}_{est}\right|^2}{\left|\mathbf{w}_{opt}\right|^2} \tag{13}$$

where $\mathbf{w}_{opt}$ is the optimal value of network vector parameter, and $\mathbf{w}_{est}$ the estimated value.

The results obtained, are shown in Fig. 4, which is the average of 50 simulations. As we could expect, JNDDE gets a lower WRSE for the same amount of training data. This is due to the fact that it makes true hypothesis about the data distribution, thus reducing the space search. Note that the difference in WRSE, in a log-log scale, remains constant no matter what is the size of the training set.

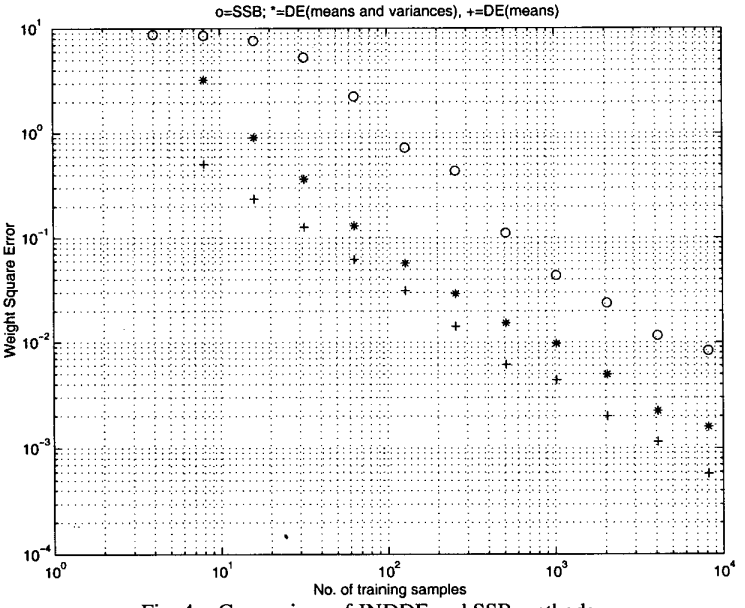

Fig. 4 .- Comparison of JNDDE and SSB methods.

CONCLUSIONS

This paper presents a novel approach to the estimation of 'a posteriori' probabilities with neural networks which is based on estimating the conditional data densities. The JNDDE can reduce the data requirements of the estimation process, at the cost of making assumptions about the sample distribution. The method is based on using two different models, one for the posteriori probabilities (which is given by the network structure) and the other for the conditional data

densities; both of them can be made arbitrarily complex. This states an important question for future work: for a given sample, how to identify the complexity of both models. On the other hand, since JNDDE uses a data model, it has the potential capability to extract information from unlabeled data. Use of JNDDE in hybrid learning, both labeled an unlabeled data, is a matter for future research.

ACKNOWLEDGEMENTS: This work has been partially supported by the Spanish Government under C.I.C.Y.T. grant numbers TIC96-0500-C10-03, TIC96-0500-C10-09 and TIC97-0772 . T. Adali's work was partially supported by the National Science Foundation Career Award, NSFNCR-9703161.

REFERENCES

[1] D.W.Ruck, S.K.Rogers, M.Kabrisky, M.E.Oxley and B.W.Suter, "The multilayer perceptron as an approximation to a Bayes optimal discriminant function", **IEEE Trans. on Neural Networks**, vol. 1, no. 4, pp. 296-298, 1990.

[2] S. Amari, "Backpropagation and stochastic gradient descent method", **Neurocomputing**, No.5, pp. 185-196, 1993.

[3] J.Cid-Sueiro, A.R. Figueiras-Vidal, "Cost Functions to Estimate Class Probabilities", **Proceedings of the European Conference on Signal Analysis and Prediction (ECSAP'97)**, pp. 113-116, Praha, 24-27 June, 1997.

[4] P. Smyth, J.W. Miller, R. Goodman, "Objective functions for probability estimation", **Proceedings of the Int. Joint Confefernce on Neural Networks**, pp. 881-886, 1991.

[5] J. I. Arribas , J. Cid-Sueiro, "Bayesian Approaches to the Estimation of A Posteriori Probabilities", **Procs of the IASTED Int. Conf. on Signal Processing and Communications**, pp. 107-110, Canary Islands, Spain, Feb. 1998.

[6] J. Cid-Sueiro, J. Ignacio Arribas, A. R. Figueiras-Vidal, "Cost Functions to Estimate A Posteriori Probability in Multi-Class Problems", to appear in **IEEE Trans. on Neural Networks**.

[7] David Miller and Hasan S. Uyar, "A Mixture of Experts Classifier with Learning Based on Both Labelled and Unlabelled Data", **Neural Information Processing Systems 9**, pp. 571-577, 1996.

[8] Michael I. Jordan, "Why the logistic function? A turotial discussion on probabilities and neural networks", M.I.T., *Computational Cognitive Science Technical Report 9503*, August 13, 1995.

[9] C.M. Bishop, Neural Networks for pattern recognition, Oxford Press,1995.

[10] J.S. Bridle, "Probabilistic interpretation of feedforward classification network outputs with relationship to statistical pattern recognition". In F. Fogelman Soulie and J. Herault (Eds.), **Neurocomputing: Algorithms, Architectures and Applications**, pp. 227-236. New York: Springer Verlag, 1990.

[11] Juan Ignacio Arribas, Jesus Cid-Sueiro, Tulay Adali and Anibal R. Figueiras-Vidal, "Neural networks to estimate ML multi-class constrained conditional probability density functions", **International Joint Conference on Neural Networks**, Washington D.C., USA, July 1999.