KNOWLEDGE ENGINEERING: PRINCIPLES AND TECHNIQUES
Proceedings of the International Conference on Knowledge Engineering,
Principles and Techniques, KEPT2007
Cluj-Napoca (Romania), June 6–8, 2007, pp. 107–113

A HYBRID INCREMENTAL/MONTE CARLO SEARCHING TECHNIQUE FOR THE "SMOOTHING" PARAMETER OF PROBABILISTIC NEURAL NETWORKS

FLORIN GORUNESCU[1], MARINA GORUNESCU[2], KENNETH REVETT[3], AND MARIUS ENE [4]

ABSTRACT. The only control factor that needs to be selected for Probabilistic Neural Network training to cause a reasonable amount of overlap is the smoothing parameter σ. The shape of the decision boundary can be made as complex as necessary by choosing an appropriate value of σ. It has been shown that the decision boundary varies continuously from a hyperplane to a very nonlinear surface according to σ and it has also been suggested several techniques to choose this parameter. The aim of this paper is to introduce a hybrid technique, sequentially using an incremental search as a first raw approach, followed by a Monte Carlo search to fine tune the first one. An application to a medical data concerning the hepatic cancer is also considered.

1. Introduction

The probabilistic neural networks (PNN), introduced by Specht [4], represent supervised neural networks (NN) widely used in the area of classification, pattern recognition, nonlinear mapping etc. PNN are essentially based on the well-known Bayesian classification technique, that is a strategy allowing the minimization of the expected risk. They constitute a class of NN combining some of the best attributes of statistical pattern recognition and feedforward NN, representing the neural network implementation of kernel discriminant analysis. The greatest advantages of PNN are the fact that the output is probabilistic (easy interpretation of output) and the training speed. PNN requires small training time, training PNN actually consisting mostly of copying training cases into the network. On the other hand, the main criticism of PNN is the very rapid increase in memory and computing time when the input sample dimension and the training set size increase ("curse" of dimensionality).

2000 *Mathematics Subject Classification.* 68T05, 90B15.
Key words and phrases. Probabilistic neural network, smoothing parameter, incremental search, Monte Carlo search, hybrid search.

The classification performance of PNN is largely influenced by the smoothing parameter σ (i.e. the radial deviation of the Gaussian functions). In this paper we propose a hybrid two-step sequentially algorithm to optimize the smoothing factor of the PNN. This technique consists in using, as a first step, an incremental search to estimate local optima of the cost function given by the percentage of well classified patterns. The second step, the fine tuning process, consists in using a Monte Carlo technique to estimate the best value of σ, in order to maximize the classification accuracy. This hybrid searching model is then applied to a medical data set, concerning the hepatic cancer.
The paper is organized as follows: in the following Section the basic concepts of PNN are described. Next, the proposed searching approach is presented. Subsequently, the experimental setup including the description of the data set and the sampling technique used are presented. In the next Section, a presentation of the obtained results is reported. The paper ends with conclusions.

2. Probabilistic Neural Networks

The PNN paradigm is based on the Bayes strategy for pattern recognition. Consider a q-category situation in which the state of nature θ is known to be θ_k, k=1, 2,..., q. If it desired to decide whether $\theta = \theta_k$ based on a set of measurements represented by the p-dimensional samples (vectors $\mathbf{x}$=$(x_1, x_2, ..., x_p)$), the Bayes decision rule formally becomes:

- Decision θ_k: "State of nature is θ_k";
- Given measurement $\mathbf{x}$ if the decision is θ_k then the error is $P(error|x) = 1 - P(\theta_k|x)$;
- Minimize the probability error;
- Bayes decision rule: "*Decide* θ_k *if* $P(\theta_k|x) > P(\theta_j|x),\ \forall j \neq k$" or, equivalently, "*Decide* θ_k *if* $P(x|\theta_k)P(\theta_k) > P(x|\theta_j)P(\theta_j),\ \forall j \neq k$"

To illustrate the way the Bayes decision rule is applied to PNN, consider the general case of a q-category classification problem, in which the states of nature are denoted by $\Omega_1, \Omega_2, ..., \Omega_q$. The goal is to determine the class (category) membership of a multivariate sample data (i.e. a p-dimensional random vector $\mathbf{x}$) into one of the q possible categories $\Omega_1, \Omega_2, ..., \Omega_q$, that is, we have to make the decision $D(\mathbf{x}) = \Omega_i, i = 1, 2, ..., q$, where $\mathbf{x}$ represents the sample (data vector). If we know the (multivariate) probability density functions $f_1(\mathbf{x}), f_2(\mathbf{x}), ..., f_q(\mathbf{x})$, associated with the categories $\Omega_1, \Omega_2, ..., \Omega_q$, the *a priori* probabilities $h_i = P(\Omega_i)$ of occurrence of patterns from categories Ω_i and the *loss* (or *cost*) parameters l_i, associated with all incorrect decisions given $\Omega = \Omega_i$, then, according to the Bayes decision rule, we classify $\mathbf{x}$ into the category Ω_i if the following inequality holds true:

$$l_i h_i f_i(\mathbf{x}) > l_j h_j f_j(\mathbf{x}), i \neq j.$$

The boundaries between every two decision classes Ω_i and Ω_j, $i \neq j$, are given by the hypersurfaces:

$$l_i h_i f_i(\mathbf{x}) = l_j h_j f_j(\mathbf{x}), i \neq j.$$

and the accuracy of the decision depends on the accuracy of estimating the corresponding p.d.f's. only.

The key to using the Bayes decision rule to PNN is represented by the technique chosen to estimate the p.d.f's $f_i(\mathbf{x})$ corresponding to each decision class Ω_i, based upon the training samples set. The classical approach uses a sum of small multivariate Gaussian distributions, centered at each training sample, that is:

$$f_i(\mathbf{x}) = \frac{1}{(2\pi)^{p/2}\sigma^p} \cdot \frac{1}{m_i} \cdot \sum_{j=1}^{m_i} \exp\left(-\frac{\|x-x_j\|^2}{2\sigma^2}\right), i = 1, 2, ..., q,$$

where m_i is the total number of training patterns in Ω_i, $\mathbf{x}_j$ is the j-th training pattern from category Ω_i, p is the input space dimension and σ is the adjustable "*smoothing*" parameter using the training procedure.

Bayes decision rule: For each $\mathbf{x} \in \Omega_i$ compare $f_i(\mathbf{x})$ and $f_j(\mathbf{x})$ for all $i \neq j$, following the algorithm:

"IF $l_i h_i f_i(\mathbf{x}) > l_j h_j f_j(\mathbf{x})$ (for all $j \neq i$) THEN $\mathbf{x} \in \Omega_i$ ELSE IF $l_i h_i f_i(\mathbf{x}) \leq l_j h_j f_j(\mathbf{x})$ (for some $j \neq i$) THEN $\mathbf{x} \notin \Omega_i$"

The standard PNN training procedure requires a single pass over all the samples of the training set, rendering PNN faster to train compared to feedforward NN.

Basically, the architecture of PNN might be limited to three layers: the *input/pattern* layer, the *summation* layer and the *output* layer. Each input/pattern node forms a product of the input pattern vector $\mathbf{x}$ with a weight vector W_i and then performs a nonlinear operation, that is $\exp\left[-(W_i - \mathbf{x})(W_i - \mathbf{x})^\tau/(2\sigma^2)\right]$ (assuming that both $\mathbf{x}$ and W_i are normalized to unit length), before outputting its activation level to the summation node. Each summation node receives the outputs from the input/pattern nodes associated with a given class and simply sums the inputs from the pattern units that correspond to the category from which the training pattern was selected, that is $\sum_i \exp\left[-(W_i - \mathbf{x})(W_i - \mathbf{x})^\tau/(2\sigma^2)\right]$. The output nodes produce binary outputs by using the inequality:

$$\sum_i \exp\left[-(W_i - \mathbf{x})(W_i - \mathbf{x})^\tau/(2\sigma^2)\right] > \sum_j \exp\left[-(W_j - \mathbf{x})(W_j - \mathbf{x})^\tau/(2\sigma^2)\right]$$

related to two different categories Ω_i and Ω_j.

Note. Since the PNN paradigm is based on the Bayes decision rule, the binary outputs above are based on finding the maximum of all sums.

3. Hybrid incremental/Monte Carlo searching technique

The key factor in PNN is therefore the way to determine the value of σ, since this parameter needs to be estimated to cause reasonable amount of overlap. Commonly, the smoothing factor is chosen heuristically. If σ is too large or too small the corresponding probability density functions will lead to the increase in misclassification rate. Thus, too small deviations cause a very spiky approximation which cannot generalize and, on the other hand, too large deviations smooth out the details.

Although an appropriate figure is easily chosen by experiment, by selecting a number which produces a low selection error, and fortunately PNN are not too sensitive to the precise choice of smoothing factor, the smoothing parameter σ is critical for the classification accuracy. Therefore, there are some approaches to assess this important PNN issue [1], [2], [3], [5]. This work deals with the estimation of a (near) optimum value of σ using a two-step method, combining both a deterministic raw detection technique (incremental search) and a stochastic fine detection (Monte Carlo search).

The complete hybrid searching algorithm consists in three steps (sub-algorithms):

- Algorithm for estimating the searching domain D_σ, using statistical tools.
- Algorithm for raw estimating of local optima of the cost function, using an incremental search.
- Algorithm for fine estimating of the optimum value of σ, using a Monte Carlo search.

We synthesize below the three algorithms.

A. Algorithm to estimate D_σ

Input. Consider q classes of objects (p-dimensional vectors) $\Omega_1, \Omega_2, ..., \Omega_q$. Each decision class Ω_i contains a number of m_i vectors (or training patterns), that is $\Omega_i = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_{m_i}\}$.

1) For each class $\Omega_i, i = 1, 2, ..., q$, compute the (Euclidian) distance between any pair of vectors and denote these distances by $d_1, d_2, ..., d_{r_i}$, where $r_i = C^2_{m_i}$.

2) For each class $\Omega_i, i = 1, 2, ..., q$, compute the corresponding average distances and standard deviations $D_i = \frac{\sum_{j=1}^{r_i} d_j}{r_i}$, $SD_i = \sqrt[2]{\frac{\sum_{j=1}^{r_i} (d_j - D_i)^2}{r_i}}$.

3) For each class $\Omega_i, i = 1, 2, ..., q$, consider the corresponding 99.7% confidence interval $I_{\Omega_i} = (D_i - 3SD_i, D_i + 3SD_i)$ for the average distances.

Output. $D_\sigma = (\cup I_{\Omega_i}) \cap R_+$ represents the searching domain for the smoothing parameter σ.

B. Algorithm for incremental search

Input. The searching domain D_σ.

1) Divide the searching domain D_σ by N dividing knots $\Delta_j, j = 1, 2, ..., N$ into $(N + 1)$ equal sectors.

2) Repeat Bayes decision rule algorithm by assigning $\sigma = \Delta_j$.

3) Compute the maximum value of the cost function.

Output. The values σ's corresponding to the local optima of the cost function.

C. Algorithm for Monte Carlo search

Input. The values σ's corresponding to the local optima of the cost function.

1) Consider heuristically a neighborhood for each σ (i.e. an interval centered in σ).

2) Generate in each interval a number M of random dividing points $\{P_1, P_2, ..., P_M\}$, uniformly distributed.

3) Repeat Bayes decision rule algorithm by assigning $\sigma = P_k, k = 1, 2, ..., M$.

4) Compute the maximum value of the cost function in each case.

Output. The value σ corresponding to the global optimum of the cost function represents the optimal value of the smoothing parameter.

Note. In the incremental search, the number of dividing knots N was chosen heuristically. It has been experimentally proven that for $N > 300$, the accuracy graph became flat, ending thus a further search. The number of training patterns that are classified in the right way represents the cost function of the PNN algorithm.

4. Experimental results

The model was fitted to real data consisting of 299 individuals (both patients and healthy people) from the Department of Internal Medicine, Division of Gastroenterology, University Emergency Hospital of Craiova, Romania. This group of individuals consists of 60 patients with chronic hepatitis (CH), 179 patients with liver cirrhosis (LC), 30 patients with hepatocellular carcinoma (HCC) and 30 healthy people (HP).

The hybrid PNN algorithm has been applied to data in order to classify the group of individuals into two categories, depending on the diagnosis type: Ω_1 = hepatic cancer (HCC) and Ω_2 = non-hepatic cancer. In this way, the physician benefits from an efficient tool, provided by this approach, seen as a computer-aided diagnostic of the hepatic cancer.

Each individual in the data set is represented by a 15-dimensional vector $\mathbf{x} = (x_1, x_2, ..., x_{15})$, where the components represent some of the most important characteristics (serum enzymes) leading to the right medical diagnosis. Concretely, x_1 = TB (total bilirubin), x_2 = DB (direct bilirubin), x_3 = IB (indirect bilirubin), x_4 = AP (alkaline phosphatase), x_5 = GGT (gamma glutamyl transpeptidase), x_6 = LAP (leucine amino peptidase), x_7 = AST (aspartate amino transferase), x_8 = ALT (alanine amino transferase), x_9 = LDH (lactic dehydrogenase), x_{10} = PI (prothrombin index), x_{11} = Gamma, x_{12} = Albumin, x_{13} = Glycemia, x_{14} = Cholesterol, x_{15} = Age.

The first step to obtain a good classification using PNN is to optimally estimate the misclassification costs and the prior probabilities. Unfortunately, there is no definitive science to obtain them and must be assigned as a specific part of the problem definition. In our practical experiment we have estimated them heuristically. Thus, as far as the costs parameters are concerned, we have considered them to be inversely proportional to the average distances D_i, that is $l_i = 1/D_i$. As concerns the prior probabilities, they measure the membership probability in each group and, thus, we have considered them equal to each group size, that is $h_i = m_i$.

To avoid overfitting, the data set was randomly partitioned into two sets: the training set and the testing set. A number of 254 persons (85%) of the initial group were withheld from the initial group for the training process. Once the optimal smoothing parameter σ was obtained using the training set, the trained PNN was applied to the testing set. The procedure was repeated 10 times and the algorithms were coded in Java for the ease of implementation, using JDBC (*Java Database Connectivity*) for data processing. In Table 1 we have displayed the results of our experiment.

Table 1. Experimental results

No. N of dividing knots	Accuracy (%)			
	Incremental search		Hybrid search	
	Training	Testing	Testing	Sigma
100	90	83	86	488.065
150	92	85	90	557.786
300	96	90	95	581.278

The gain in classification accuracy is obviously, obtained without a significant loss in speed.

5. Discussion

A hybrid searching model of the smoothing parameter for probabilistic neural networks was proposed. The proposed approach incorporates an incremental (deterministic) search for the optima of the cost function and a Monte Carlo (stochastic) search for the global optimum of the smoothing parameter. The effectiveness of this PNN-based hybrid model is assessed on a medical data set related to hepatic cancer diagnosis. We used raw data (the only data available for the experiment) and we obtained reliable results providing the PNN ability and flexibility to learn from raw examples. Implementation is relatively rapid and it is an alternative to standard statistical approaches.

To evaluate further the capabilities of this model, comparisons with other searching techniques have to be done. Further work will also include an alternative to this approach, using evolutionary algorithms for the second step, instead of the Monte Carlo simulation.

References

[1] Bolat B., Yldirim T., 2003, Performance increasing methods for probabilistic neural networks, Pakistan Journal of Information and Technology, 2(3), pp. 250-255.

[2] Georgiou V.L., Pavlidis N.G., Parsapoulos K.E., Alevizos P.D., Vrahatis M.N. 2004, Optimizing the performance of probabilistic neural networks in bioinformatics task, Proceedings of the EUNITE Conference, pp. 34-40.

[3] Gorunescu F., Gorunescu M., El-Darzi E., Ene M., Gorunescu S., 2005, Statistical Comparison of a Probabilistic Neural Network Approach in Hepatic Cancer Diagnosis, Proceedings IEEE International Conference on "Computer as a tool"- Eurocon2005, Belgrade, Serbia, pp. 237-240.

[4] Specht D.F., 1990, Probabilistic neural networks, Neural Networks, vol. 3, pp. 109-118.

[5] Streit R.L., Luginbuhl T.E., 1994, Maximum likelihood training of probabilistic neural networks, IEEE Trans. Neural Networks 5(5), pp. 764-783.

(1) Universitatea de Medicina si Farmacie din Craiova, str. Petru Rares, Nr. 2-4
E-mail address: `fgorun@rdslink.ro`

(2) Universitatea din Craiova, Str. A.I. Cuza, Nr. 13
E-mail address: `mgorun@inf.ucv.ro`

(3) University of Westminster, Harrow School of Computer Science, London, UK
E-mail address: `revettk@westminster.ac.uk`

(4) Universitatea de Medicina si Farmacie din Craiova, str. Petru Rares, Nr. 2-4
E-mail address: `enem@umfcv.ro`