Simple and Effective Connectionist Nonparametric Estimation of Probability Density Functions

Edmondo Trentin

Dipartimento di Ingegneria dell'Informazione
Università di Siena, V. Roma, 56 - Siena (Italy)

Abstract. Estimation of probability density functions (pdf) is one major topic in pattern recognition. Parametric techniques rely on an arbitrary assumption on the form of the underlying, unknown distribution. Nonparametric techniques remove this assumption In particular, the Parzen Window (PW) relies on a combination of local window functions centered in the patterns of a training sample. Although effective, PW suffers from several limitations. Artificial neural networks (ANN) are, in principle, an alternative family of nonparametric models. ANNs are intensively used to estimate probabilities (e.g., class-posterior probabilities), but they have not been exploited so far to estimate pdfs. This paper introduces a simple neural-based algorithm for unsupervised, nonparametric estimation of pdfs, relying on PW. The approach overcomes the limitations of PW, possibly leading to improved pdf models. An experimental demonstration of the behavior of the algorithm w.r.t. PW is presented, using random samples drawn from a standard exponential pdf.

1 Introduction

One major topic in pattern recognition is the problem of estimating probability density functions (pdf) [5]. Albeit popular, parametric techniques (e.g. maximum-likelihood for Gaussian mixtures) rely on an arbitrary assumption on the form of the underlying, unknown distribution [8]. Nonparametric techniques (e.g. k_n-nearest neighbors [4]) remove this assumption and attempt a direct estimation of the pdf from a data sample. The Parzen Window (PW) is one of the most popular nonparametric approaches to pdf estimation, relying on a combination of local window functions centered in the patterns of the training sample [5]. Although effective, PW suffers from several limitations, including:

(i) the estimate is not expressed in a compact functional form (i.e., a probability law), but it is a sum of as many local windows as the size of the sample;
(ii) the local nature of the window functions tend to yield a fragmented model, which is basically "memory based" and (by definition) is prone to overfitting;

F. Schwenker and S. Marinai (Eds.): ANNPR 2006, LNAI 4087, pp. 1–10, 2006.

(iii) the whole training sample has to be kept always in memory in order to compute the estimate of the pdf over any new (test) patterns, resulting in a high complexity of the technique in space and time;
(iv) the form of the window function chosen has a deep influence on the eventual form of the estimated model, unless an asymptotic case (i.e., infinite sample) is considered;
(v) the PW model heavily depends on the choice of an initial width of the local region of the feature space where the windows are centered.

Artificial neural networks (ANN) are, in principle, an alternative family of nonparametric models [6]. Given the "universal approximation" property [2] of certain ANN families (multilayer perceptrons [10] and radial basis function networks [9]), they might be a suitable model for any given (continuous) form of data distributions. While ANNs are intensively used for estimating probabilities [7] (e.g., posterior probabilities in classification tasks [2]), they have not been exploited so far for estimating pdfs. For instance, Bourlard and Morgan apply ANNs in order to estimate probabilistic quantities within a hidden Markov model (HMM) framework [3], but they use the ANN as a sate-posterior probability model, instead of modeling the emission-likelihoods that are required in the formal definition of standard HMMs. One of the main rationales behind this fact is that connectionist modeling of probabilities is easily (and somewhat heuristically) achieved by standard supervised backpropagation [10], once 0/1 target outputs are defined for the training data [2] (along the line of the Widrow-Hoff algorithm for linear discriminants [4]). Moreover, it is also simple to introduce constraints within the model that ensure the ANN may be interpreted in probabilistic terms, e.g. using sigmoid output activations (that range in the $(0, 1)$ interval), along with a softmax-like mechanism [1] which ensures that all the outputs sum to 1. Learning a pdf, on the contrary, is an unsupervised and far less obvious task.

This paper introduces a neural-based algorithm for unsupervised, nonparametric density estimation. The algorithm is presented in detail in Section 2, along with a concise review of the PW technique which is used within the ANN training scheme. The approach overcomes the limitations of PW, and it may lead to better pdf models than the PW itself. An experimental demonstration of the behavior of the algorithm w.r.t. PW is presented in Section 3, using random samples drawn from a standard exponential pdf.

2 The Proposed Estimation Algorithm

The algorithm is introduced by reviewing the basic concepts of PW estimation (refer to [4]). Let us consider a pdf $p(\mathbf{x})$, defined over a real-valued, d-dimensional feature space. The probability that a pattern $\mathbf{x}' \in \mathcal{R}^d$, drawn from $p(\mathbf{x})$, falls in a certain region R of the feature space is $P = \int_R p(\mathbf{x})d\mathbf{x}$. Let then $\mathcal{T} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ be an unsupervised sample of n patterns, identically and independently distributed (i.i.d.) according to $p(\mathbf{x})$. If k_n patterns in $\mathcal{T}$ fall within

R, an empirical estimate of P can be obtained as $P \simeq k_n/n$. If $p(\mathbf{x})$ is continuous and R is small enough to prevent $p(\mathbf{x})$ from varying its value over R in a significant manner, we are also allowed to write $\int_R p(\mathbf{x})d\mathbf{x} \simeq p(\mathbf{x}')V$, where $\mathbf{x}' \in R$, and V is the volume of region R. As a consequence of the discussion, we can obtain an estimated value of the pdf $p(\mathbf{x})$ over pattern $\mathbf{x}'$ as:

$$p(\mathbf{x}') \simeq \frac{k_n/n}{V_n} \tag{1}$$

where V_n denotes the volume of region R_n (i.e., the choice of the region width is explicitly written as a function of n), assuming that smaller regions around $\mathbf{x}'$ are considered as the sample size n increases. This is expected to allow Equation (1) to yield improved estimates of $p(\mathbf{x})$, i.e. to converge to the exact value of $p(\mathbf{x}')$ as n (hence, also k_n) tends to infinity (a discussion of the asymptotic behavior of nonparametric models of this kind can be found in [4]).

The basic instance of the PW technique assumes that R_n is a hypercube having edge h_n, such that $V_n = h_n^d$. The edge h_n is usually defined as a function of n as $h_n = h_1/\sqrt{n}$, in order to ensure a correct asymptotic behavior. The value h_1 has to be chosen empirically, and it heavily affects the resulting model. The formalization of the idea requires to define a unit-hypercube window function in the form

$$\varphi(\mathbf{y}) = \begin{cases} 1 & \text{if } \mid y_j \mid \leq 1/2, j = 1, \ldots, d \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

such that $\varphi(\frac{\mathbf{x}'-\mathbf{x}}{h_n})$ has value 1 iff $\mathbf{x}'$ falls within the d-dimensional hypercubic region R_n centered in $\mathbf{x}$ and having edge h_n. This implies that $k_n = \sum_{i=1}^{n} \varphi(\frac{\mathbf{x}'-\mathbf{x}_i}{h_n})$. Using this expression, from Equation (1) we can write

$$p(\mathbf{x}') \simeq \frac{1}{n} \sum_{i=1}^{n} \frac{1}{V_n} \varphi(\frac{\mathbf{x}' - \mathbf{x}_i}{h_n}) \tag{3}$$

which is the PW estimate of $p(\mathbf{x}')$ from the sample $\mathcal{T}$. The model is usually refined by considering smoother window functions $\varphi(.)$, instead of hypercubes, e.g. standard Gaussian kernels with zero mean and unit covariance matrix.

Let us now consider a feedforward ANN that we wish to train in order to learn a model of the probability law $p(\mathbf{x})$ from the unsupervised dataset $\mathcal{T}$. The idea is to use the PW model as a target output for the ANN, and to apply standard backpropagation to learn the ANN connection weights. A unbiased variant of this idea is proposed, according to the following unsupervised algorithm (expressed in pseudo-code):

```
Input: T = {x_1, ..., x_n}, h_1.
Output: p̃(.) /*the connectionist estimate of p(.) */

1. Let h_n = h_1/√n
2. Let V_n = h_n^d
3. For i=1 to n do /* loop over T */
```

3.1 Let $\mathcal{T}_i = \mathcal{T} \setminus \{\mathbf{x}_i\}$
3.2 Let $y_i = \frac{1}{n}\sum_{\mathbf{x}\in\mathcal{T}_i}\frac{1}{V_n}\varphi(\frac{\mathbf{x}_i-\mathbf{x}}{h_n})$ /* target output */
4. Let $\mathcal{S} = \{(\mathbf{x}_i, y_i) \mid i = 1, \ldots, n\}$ /* supervised training set */
5. Train the ANN via backpropagation over $\mathcal{S}$
6. Let $\tilde{p}(.)$ be equal to the ANN
7. Return $\tilde{p}(.)$

We call the ANN trained this way the Parzen Neural Network (PNN). Since the PNN output is assumed to be an estimate of a pdf, it must be non-negative. This is granted once standard sigmoids (in the form $y = \frac{1}{1+e^{-x}}$) are used in the output layer. Standard sigmoids range in the $(0, 1)$ interval, while pdfs may take any positive value. For this reason, sigmoids with adaptive amplitude λ (i.e., in the form $y = \frac{\lambda}{1+e^{-x}}$), as described in [11], should be taken into consideration. A direct alternative is using linear output activation functions, forcing negative outputs to zero once training is completed. Nevertheless, as in the k_n-nearest neighbor technique [4], the PNN is not necessarily a *pdf* (in general, the integral of $\tilde{p}(.)$ over the feature space is not 1).

There are two major aspects of the algorithm that shall be clearly pointed out. First, the PW generation of target outputs (steps 3-3.2) is unbiased. Computation of the target for i-th input pattern $\mathbf{x}_i$ does not involve $\mathbf{x}_i$ in the underlying PW model. This is crucial in smoothing the local nature of PW. In practice, the target (estimated pdf value) over $\mathbf{x}_i$ is determined by the concentration of patterns in the sample (different from $\mathbf{x}_i$) that occurs in the surroundings of $\mathbf{x}_i$. In particular, if an isolated pattern (i.e., an outlier) is considered, its exclusion from the PW model turns out to yield a close-to-zero target value. This phenomenon is evident along the possible tails of certain distributions, and it is observed in the experiments described in Section 3.

A second relevant aspect of the algorithm is that it trains the ANN only over the locations (in the feature space) of the patterns belonging to the original sample. At first glance, a different approach could look more promising: once the PW model has been estimated from $\mathcal{T}$, generate a huge supervised training set by covering the input interval in a "uniform" manner, and by evaluating target outputs via the PW model. A more homogeneous and exhaustive coverage of the feature space would be expected, as well as a more precise ANN approximation of the PW. As a matter of fact, training the ANN this way reduces its generalization capabilities, resulting in a more "nervous" surface of the estimated pdf, since the PW model has a natural tendency to yield unreliable estimates over regions of the feature space that are not covered by the training sample (again, refer to the experimental demonstration in Section 3).

It is immediately seen that, in spite of the simplicity of its training algorithm, the PNN is expected to overcome most of the PW limitations listed[1] in Section 1. The following Section highlights that, in addition, the PNN may turn out to be more accurate than the PW estimate.

[1] Response of the PNN w.r.t. point (i) in that list depends on the complexity of the ANN architecture that is used.

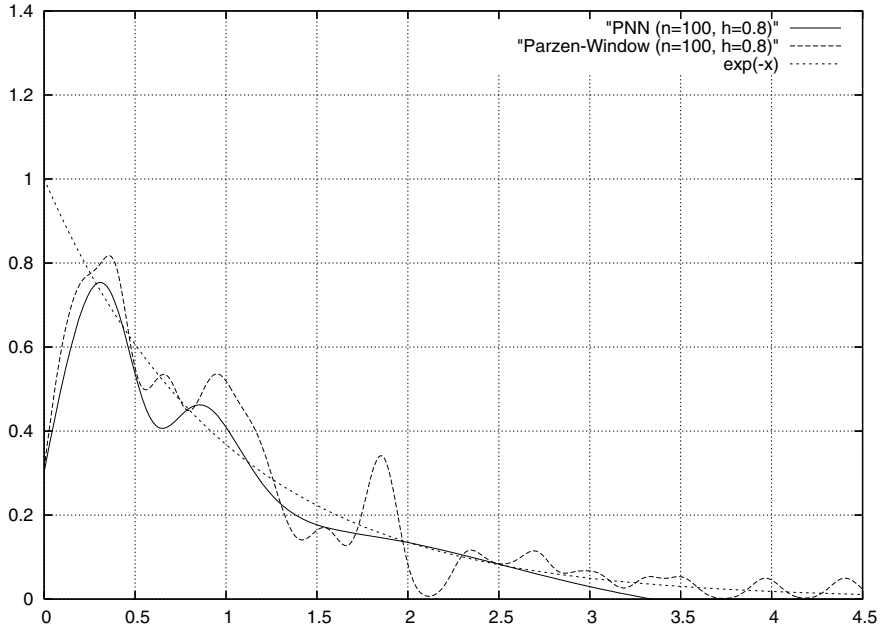

Fig. 1. Nonparametric estimates of the pdf from a random sample of 100 points drawn from the standard exponential distribution. Parzen-Window and PNN are estimated using an initial edge $h_1 = 0.8$.

3 Experimental Demonstration

A simple, illustrative estimation task is considered. Samples in this Section are randomly drawn from the popular exponential pdf, defined as

$$p(x) = \frac{1}{\beta} e^{-(x-\alpha)/\beta} \quad \text{with} \quad x \geq \alpha, \beta > 0 \tag{4}$$

where α is the location and β is the scale. This distribution is unlikely to be modeled under a parametric assumption (e.g., via maximum-likelihood with a mixture of Gaussian densities), unless the form of Equation (4) is known in advance. The application of nonparametric techniques is sought. In the following experiments, a *standard* exponential distribution is used, having $\alpha = 0$ and $\beta = 1.0$.

In the first instance, a random sample of $n = 100$ points was generated according to $p(x)$. Figures 1–4 show the resulting PW and PNN models, estimated from the sample and plotted against the original pdf, for decreasing values of h_1. The latter ranged from 0.8 to 0.1. As expected, the PW estimates are heavily affected by the choice for h_1, and the shape of the corresponding estimated pdf is unnaturally peaked around the individual points in the training dataset. The overall trend of the original distribution is better fit for small values of h_1, but

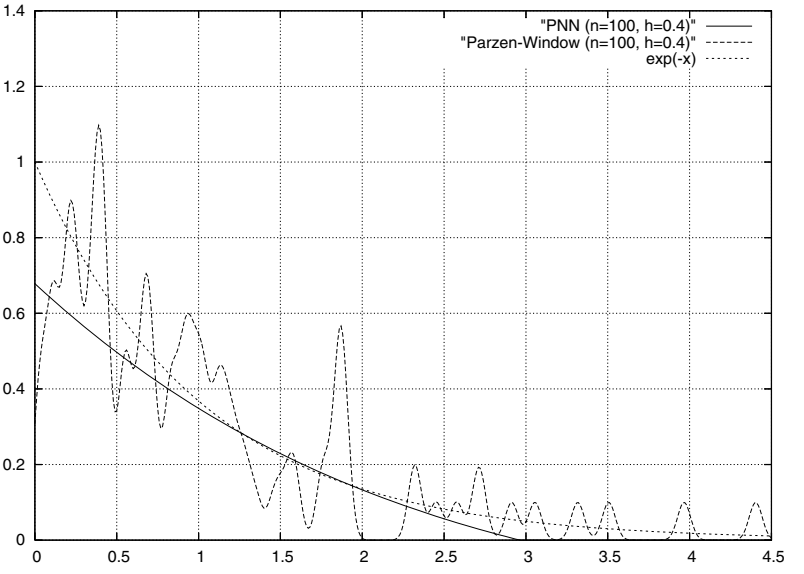

Fig. 2. Nonparametric estimates of the pdf from a random sample of 100 points drawn from the standard exponential distribution. Parzen-Window and PNN are estimated using an initial edge $h_1 = 0.4$.

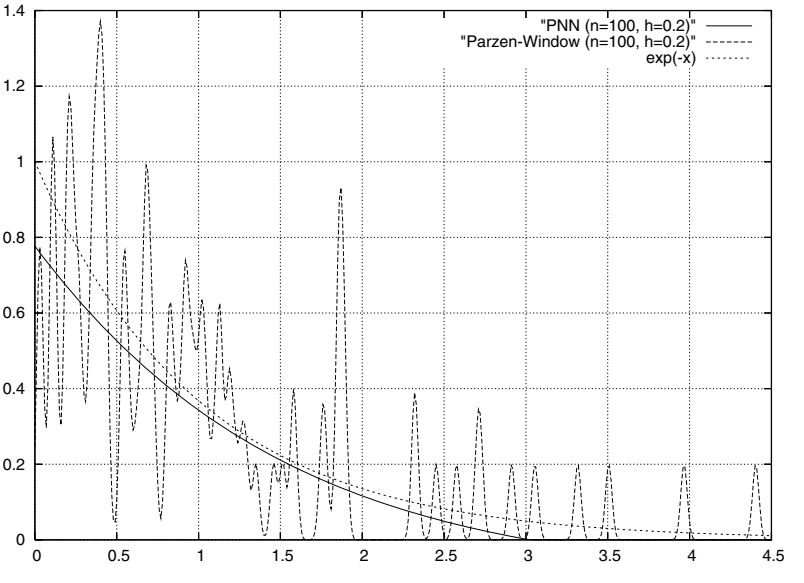

Fig. 3. Nonparametric estimates of the pdf from a random sample of 100 points drawn from the standard exponential distribution. Parzen-Window and PNN are estimated using an initial edge $h_1 = 0.2$.

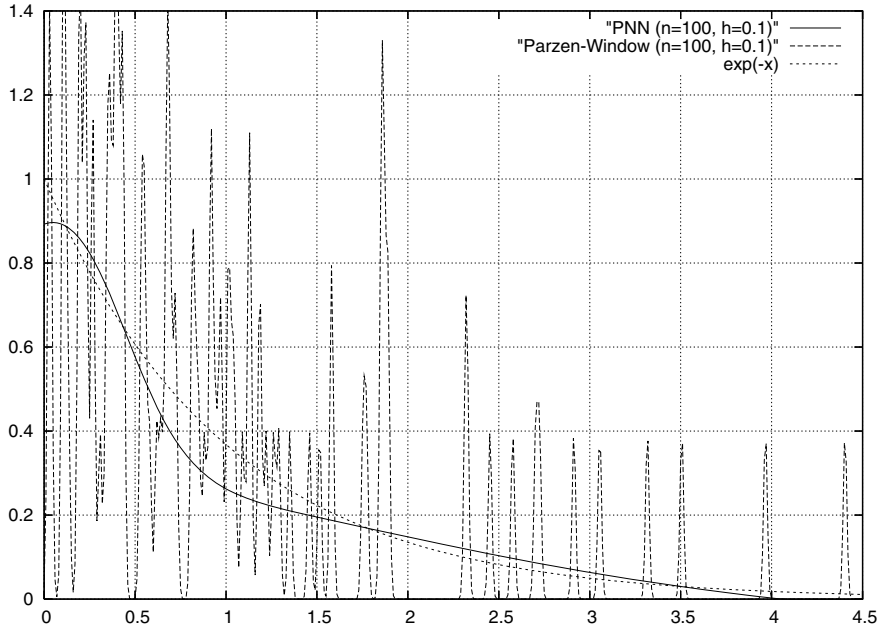

Fig. 4. Nonparametric estimates of the pdf from a random sample of 100 points drawn from the standard exponential distribution. Parzen-Window and PNN are estimated using an initial edge $h_1 = 0.1$.

the function obtained turns out to be very sensitive to local sampling irregularities. Moreover, the PW model includes peaks along the tail of the pdf, due to the presence of isolated points (within the sample) belonging to low-probability regions. The presence of local peaks basically violates the natural shape of the underlying pdf. Finally, as far as concerns complexity, the PW model requires the combination of $n = 100$ window functions, i.e. a total of 100 free parameters that had to be determined from the data.

The PNN is trained on the same values of h_1 as the corresponding PW. A compact ANN topology was used, namely 6 sigmoids in the hidden layer and a linear output unit. The latter is forced to non-negative values at test time, by converting negative outputs to zero. It is seen from the graphics that the PNN estimate is smoother, much closer to the reference exponential pdf than the PW is. Sensitivity to the choice of h_1 is significantly decreased w.r.t. PW. It is worth noticing the difference between the PNN curve and the form of the PW model that, roughly speaking, constitutes the target output for PNN during training. The unbiased nature of the training algorithm (steps 3-3.2) is further exploited in the avoidance of individual peaks, especially along the tail of the distribution. Complexity of the model is reduced in a substantial manner, as well, namely 18 free parameters to be learned from the data: 6 input-to-hidden weights, 6 hidden-to-output weights, and 6 individual bias (i.e. diagonalization) values for

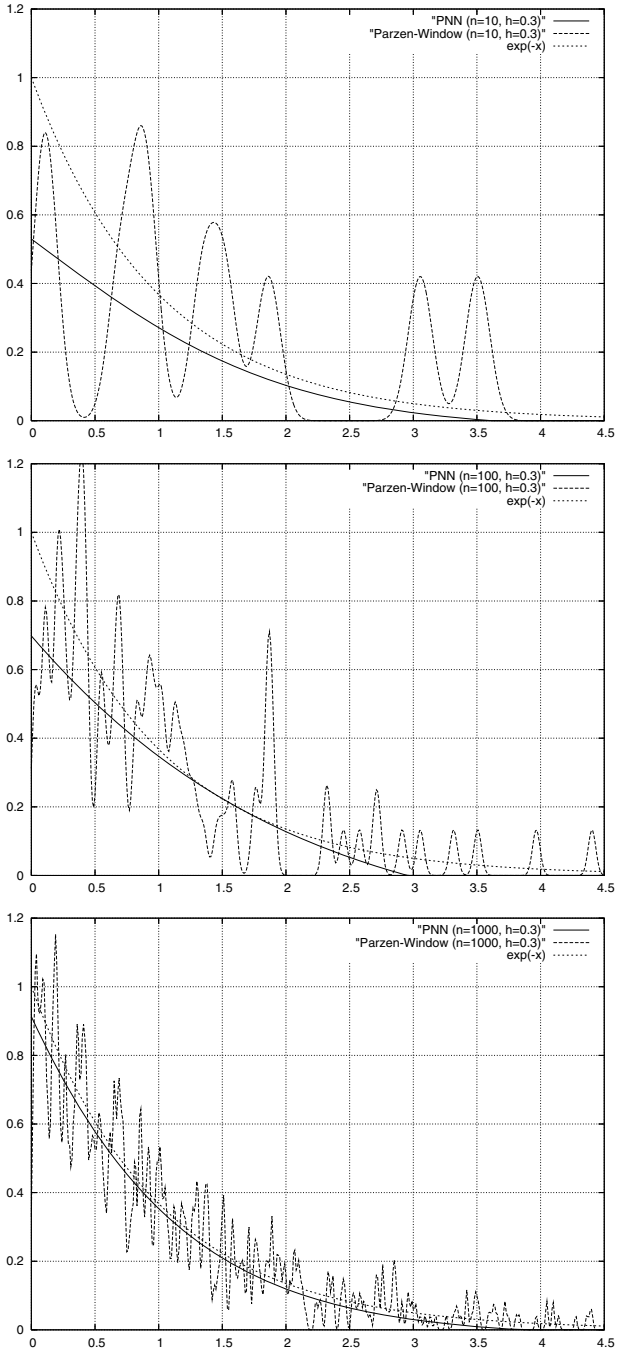

Fig. 5. Estimates of the pdf for an increasing sample size: $n = 10$ (top), $n = 100$ (mid), $n = 1000$ (bottom). Initial edge $h_1 = 0.3$.

the sigmoids. Experiments with an increased number of hidden units did not lead to significant improvement in the modeling capabilities of the PNN, and the contour of the estimated pdf remained basically the same.

Figure 5 plots the results obtained for an increasing number n of training data. An intermediate value of $h_1 = 0.3$ was applied. The same PNN architecture described above was used. As expected, both PW and PNN models improve as the size of the sample increases. For $n = 1000$ the PNN (18 free parameters, 6 nonlinear functions to be evaluated and combined at test time) is close to the reference exponential pdf, PW (1000 free parameters, 1000 nonlinear functions to be evaluated and combined at test time), is still sensitive to individual training points and outliers (see, again, the tail of the distribution). In real-world applications, e.g. speech processing, the sample size (e.g., $n >> 1000$) may make the PW approach computationally infeasible at test time (too complex in space and time requirements), while the PNN (once the resource-demanding training process is completed) remains compact and efficient.

4 Conclusion

This paper presented a simple and effective neural-based nonparametric technique for the estimation of probability density functions. The estimation method can be seen as an unsupervised learning algorithm for neural networks, since it takes an unlabelled training sample and learns the proper ANN connection weights using the gradient method. A reference, unbiased PW estimate is used to drive the learning process. In this respect, it is worth underlining the strong difference in shape between the PW estimate and the corresponding PNN, i.e. the latter is not limited to approximate the PW closely. The PNN overcomes most of the limitations of classic nonparametric techniques, e.g. PW and k_n-nearest neighbor. As in the k_n-nearest neighbor case, the PNN is not in general a pdf, but the resulting model is expected to be close to the actual pdf and, eventually, useful. Standard regularization techniques [2] can be used to increase the generalization capabilities of the PNN, yielding even smoother pdf representations. We are currently carrying out experiments on real-world data for pattern classification problems.

References

1. Y. Bengio. *Neural Networks for Speech and Sequence Recognition.* International Thomson Computer Press, London, UK, 1996.
2. C. M. Bishop. *Neural Networks for Pattern Recognition.* Oxford University Press, Oxford, 1995.
3. H. Bourlard and N. Morgan. *Connectionist Speech Recognition. A Hybrid Approach*, volume 247. Kluwer Academic Publishers, Boston, 1994.
4. R. O. Duda and P. E. Hart. *Pattern Classification and Scene Analysis.* Wiley, New York, 1973.
5. K. Fukunaga. *Statistical Pattern Recognition.* Academic Press, San Diego, second edition, 1990.

6. S. Haykin. *Neural Networks (A Comprehensive Foundation)*. Macmillan, New York, 1994.
7. D. Husmeier. *Neural Networks for Conditional Probability Estimation*. Springer-Verlag, London, 1999.
8. A.M. Mood, F.A. Graybill, and D.C. Boes. *Introduction to the Theory of Statistics*. McGraw-Hill International, Singapore, 3rd edition, 1974.
9. J. Park and I. W. Sandberg. Universal approximation using radial-basis-function networks. *Neural Computation*, 3(2):246–257, 1991.
10. D.E. Rumelhart, G.E. Hinton, and R.J. Williams. Learning internal representations by error propagation. In D.E. Rumelhart and J.L. McClelland, editors, *Parallel Distributed Processing*, volume 1, chapter 8, pages 318–362. MIT Press, Cambridge, 1986.
11. E. Trentin. Networks with trainable amplitude of activation functions. *Neural Networks*, 14(4–5):471–493, May 2001.