VSP International
Science Publishers
P.O. Box 346, 3700 AH Zeist
The Netherlands

Lecture Series on Computer and Computational Sciences
Volume 1, 2004, pp. 1-5

Evolutionary Adaptive Schemes of Probabilistic Neural Networks

V.L. Georgiou[1], **N.G. Pavlidis**[2], **K.E. Parsopoulos**[3], **Ph.D. Alevizos**[4], **M.N. Vrahatis**[5]

Department of Mathematics,
University of Patras Artificial Intelligence Research Center (UPAIRC),
University of Patras, GR–26110 Patras, Greece

Received 31 July, 2004; accepted in revised form 5 September, 2004

Abstract: Self–adaptive probabilistic neural networks have already been proposed in the literature. Typically, the kernel of the probabilistic neural network is an n–dimensional identity matrix multiplied by the scalar spread parameter, σ. This prevents the network from properly fitting the data. Another approach is to use a diagonal spread matrix, allowing each dimension to assume its own spread value. This approach increases the degrees of freedom of the network and thus allows it to fit better to the available data. However, since the optimization procedure is now multivariate instead of univariate, it is harder and computationally more demanding. To address this optimization problem we employ state-of-the-art computational intelligence optimization algorithms, like Particle Swarm Optimization, Differential Evolution and Evolution Strategies.

Keywords: Probabilistic Neural Networks, Spread Parameter, Particle Swarm Optimization, Differential Evolution Algorithm, Evolution Strategies.

Mathematics Subject Classification: 65C35, 65C60, 92B20.

1 Introduction

In this contribution an effort is attempted to combine a classical statistical technique with state-of-the-art computational intelligence methods. A common task in numerous scientific fields is the classification and prediction of class membership. Many approaches have been proposed to this end, including discriminant analysis, artificial neural networks, probabilistic neural networks, naive Bayes classifiers, and projection pursuit regression among others.

Probabilistic neural networks (PNNs) are known in the statistical literature as kernel discriminant analysis [9]. PNNs are used in many fields such as bioinformatics, medicine, and engineering with not always very promising results. Recently, PNNs have been successfully applied for the prediction of protein cellular localization sites [2]. For this purpose, the Particle Swarm Optimization (PSO) algorithm has been employed for the optimization of a spread parameter which is crucial for the PNN's operation.

[1]Corresponding author. E-mail: vlg@math.upatras.gr
[2]E-mail: npav@math.upatras.gr
[3]E-mail: kostasp@math.upatras.gr
[4]E-mail: philipos@math.upatras.gr
[5]E-mail: vrahatis@math.upatras.gr

2 Probabilistic Neural Networks

The PNN was introduced by Specht [9] as a new neural network type, although it was already widely known in the statistical literature as kernel discriminant analysis [3]. What Specht introduced was the neural network approach of kernel discriminant analysis which incorporates the Bayes decision rule and the non–parametric density function estimation of a population [7].

The training procedure of the PNN requires only a single pass of the patterns of the training data, which results in a very small training time. In fact, the training procedure is just the construction of the PNN from the available data. The structure of the PNN has always four layers; the *input layer*, the *pattern layer*, the *summation layer*, and the *output layer*. An input feature vector, $X \in \mathbb{R}^n$, is applied to the n input neurons and is passed to the pattern layer. The pattern layer is organized into K groups, where K is the number of classes present in the data set. The ith neuron in the jth group of the pattern layer computes its output using a Gaussian kernel of the form,

$$f_{i,k}(X) = \frac{1}{(2\pi\sigma^2)^{n/2}} \exp\left(-\frac{\|X - X_{i,k}\|^2}{2\sigma^2}\right), \tag{1}$$

or in matrix form,

$$f_{i,k}(X) = \frac{1}{(2\pi)^{n/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(X - X_{i,k})^T \Sigma^{-1} (X - X_{i,k})\right), \tag{2}$$

where $X_{i.k} \in \mathbb{R}^n$ is the center of the kernel and σ is the spread (smoothing) parameter of the kernel. In the matrix form of the kernel, Σ equals $\sigma^2 I_n$ so there is only one parameter to estimate. The summation layer has K neurons and estimates the conditional class probabilities as follows:

$$G_k(X) = \sum_{i=1}^{M_k} w_{ik} f_{i,k}(X), \quad k \in \{1, \ldots, K\}, \tag{3}$$

where M_k is the number of patterns of class k and w_{ik} is the prior probability of class k. So a vector X is classified to the class that has the maximum output of its summation neuron.

3 Established Computational Intelligence Algorithms

Particle Swarm Optimization: Particle Swarm Optimization is a population–based, stochastic, optimization algorithm [6]. It exploits a population of individuals to synchronously probe promising regions of the search space. The population is called a *swarm* and the individuals (i.e., the search points) are called *particles.* Each particle moves with an adaptable velocity within the search space, and retains a memory of the best position it ever encountered. In the *global* variant of PSO, the best position ever attained by all individuals of the swarm is communicated to all the particles at each iteration [5].

Let X_i be the n–dimensional feature vector of the ith particle, $X_i \in S \subset \mathbb{R}^n$ and V_i its velocity in the search space. Then, the swarm is manipulated by the equations,

$$\begin{aligned} V_i(t+1) &= w\,V_i(t) + c_1\,r_1\big(P_i(t) - X_i(t)\big) + \; c_2\,r_2\big(P_{best_i}(t) - X_i(t)\big), && (4)\\ X_i(t+1) &= X_i(t) + V_i(t+1), && (5) \end{aligned}$$

where $i = 1, \ldots, NP$; P_i is the best position it ever attained; w is a parameter called *inertia weight*; c_1 and c_2 are two positive constants called *cognitive* and *social* parameter, respectively; and r_1, r_2, are random vectors uniformly distributed within $[0,1]^n$. Alternatively, the velocity update can be performed through the following equation [1],

$$V_i(t+1) = \chi \left[V_i(t) + c_1\, r_1 \big(P_i(t) - X_i(t)\big) + \ c_2\, r_2 \big(P_{best_i}(t) - X_i(t)\big) \right], \tag{6}$$

where χ is a parameter called *constriction factor*, giving rise to a different version of PSO.

Differential Evolution: Differential Evolution (DE) is also a population–based, stochastic algorithm that exploits a population of potential solutions, *individuals*, to probe the search space. Each new individual is generated by the combination of randomly chosen individuals of the population. This combination will be called mutation. So, for each individual $w_g^k,\ k = 1,\ \ldots,\ NP$, where g denotes the current generation, a new individual v_{g+1}^i is generated according to one of the following equations:

$$\begin{aligned}
v_{g+1}^i &= w_g^{best} + F\left(w_g^{r_1} - w_g^{r_2}\right), && (7)\\
v_{g+1}^i &= w_g^{r1} + F\left(w_g^{r_2} - w_g^{r_3}\right), && (8)\\
v_{g+1}^i &= w_g^{i} + F\left(w_g^{best} - w_g^{i}\right) + F\left(w_g^{r_1} - w_g^{r_2}\right), && (9)\\
v_{g+1}^i &= w_g^{best} + F\left(w_g^{r_1} - w_g^{r_2}\right) + F\left(w_g^{r_3} - w_g^{r_4}\right), && (10)\\
v_{g+1}^i &= w_g^{r_1} + F\left(w_g^{r_2} - w_g^{r_3}\right) + F\left(w_g^{r_4} - w_g^{r_5}\right), && (11)\\
v_{g+1}^i &= \left(w_g^{r_1} + w_g^{r_2} + w_g^{r_3}\right)/3 + (p_2 - p_1)\left(w_g^{r_1} - w_g^{r_2}\right) + (p_3 - p_2)\left(w_g^{r_2} - w_g^{r_3}\right) + \\
&\quad (p_1 - p_3)\left(w_g^{r_3} - w_g^{r_1}\right), && (12)
\end{aligned}$$

where w_g^{best} is the best individual of the previous generation; $F > 0$ is a real parameter, called mutant constant and r_1, r_2, r_3, r_4, r_5 and $r_6 \in \{1, 2, \ldots, k-1, k+1, \ldots, NP\}$ are random integers mutually different.

Evolution Strategies: Evolution Strategies (ES) are population based search algorithms that have been developed by Rechenberg and Schwefel [11]. ES exploit a population of μ individuals to probe the search space. At each iteration of the algorithm, λ offsprings are produced by stochastic variation, called mutation, of recombinations of a set of individuals (called the *parents*) from the current population. Mutation is typically carried out by adding a realization of a normally distributed random vector. After the creation of the offspring individuals, selection takes place, and either the μ best individuals among the offspring population, or the μ best individuals among both the parent and the offspring populations are selected to form the parents in the next generation. These two selection schemes are denoted as (μ, λ)–ES and $(\mu + \lambda)$–ES, respectively.

ES use a set of parameters, called *strategy parameters*, to parameterize the normal distribution used in the mutation procedure. These parameters can either be fixed, or evolve during the evolution process resulting in self–adaptive ES. A special case of ES is the Covariance Matrix Adaptation Evolution Strategies (CMA–ES), which has been developed by Hansen and Ostermeier [4]. CMA–ES adapt the parameters of the algorithm such that strategy parameter settings that produce individuals that are selected are favored [4].

4 The Proposed Approach

The main disadvantage of PNNs is their sensitivity with respect to the choice of a parameter, called *spread parameter*, which influence the receptive field of each kernel. Usually, this parameter is set to standard default values suggested in the literature. However, this approach is not always appropriate due to the dependence of the spread parameter on the specific data at hand. Therefore, techniques for the automatic determination and adaptation of this parameter can be proved very useful. We investigate the sensitivity of the PNNs' performance with respect to the spread parameter and propose algorithms for the selection of an optimized value. The PSO, DE and ES algorithms are considered for this purpose.

Alternatively, a matrix of spread parameters can be used to enhance PNNs' performance by increasing the network's degrees of freedom. This approach, however, imposes a heavier computational burden since the optimization procedure becomes multivariate instead of univariate. Thus, although the matrix of spread parameters offers more flexibility to the model, the high dimensionality of the problem renders the optimization over–identified [8]. For example, if the problem is 100–dimensional, then the matrix would be 10000–dimensional, and huge amount of data would be necessary to obtain a reliable estimate of the matrix. To overcome this problem, a diagonal form of the matrix of spread parameters can be used. This matrix can be efficiently computed through the PSO, DE and ES algorithms.

Acknowledgment

We thank European Social Fund (ESF), Operational Program for Educational and Vocational Training II (EPEAEK II) and particularly the Program IRAKLEITOS for funding the above work.

References

[1] Clerc, M. and Kennedy, J., The particle swarm–explosion, stability, and convergence in a multidimensional complex space, *IEEE Trans. Evol. Comput.*, **6**(1), pp. 58–73, 2002.

[2] Georgiou, V. L., Pavlidis, N. G., Parsopoulos, K. E., Alevizos, Ph. D. and Vrahatis, M. N., Optimizing the performance of probabilistic neural networks in a bioinformatics task, In *Proceedings of the EUNITE 2004 Symposium*, pp. 34–40, Aachen, Germany, 2004.

[3] Hand, D. J., *Kernel Discriminant Analysis*, Research Studies Press, Chichester, 1982.

[4] Hansen, N. and Ostermeier, A., Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation*, **9**(2), pp. 159–195, 2001.

[5] Kennedy, J. and Eberhart, R. C., *Swarm Intelligence*, Morgan Kaufmann Publishers, 2001.

[6] Parsopoulos, K. E. and Vrahatis, M. N., On the computation of all global minimizers through particle swarm optimization, *IEEE Trans. Evol. Comput.*, **8**(3), pp. 211–224, 2004.

[7] Parzen, E., On the estimation of a probability density function and mode, *Annals of Mathematical Statistics*, **3**, pp. 1065–1076, 1962.

[8] Ripley, B. D., *Pattern Recognition and Neural Networks*, Cambridge University Press, Cambridge, 1996.

[9] Specht, D. F., Probabilistic neural networks, *Neural Networks*, **1**(3), pp. 109–118, 1990.

[10] Storn, R. and Price, K., Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces, *J. Global Optimization*, **11**, pp. 341–359, 1997.

[11] Schwefel., H. P., *Evolution and Optimum Seeking*, Wiley, New York, 1995.