Neurocomputing 71 (2008) 2489–2498

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Semi-supervised and active learning with the probabilistic RBF classifier

Constantinos Constantinopoulos *,[1], Aristidis Likas

Department of Computer Science, University of Ioannina, GR 45110 Ioannina, Greece

ARTICLE INFO

Available online 7 May 2008

Keywords:
Probabilistic RBF network
Active learning
Semi-supervised learning
EM algorithm
Unlabeled data

ABSTRACT

The probabilistic RBF network (PRBF) is a special case of the RBF network and constitutes a generalization of the Gaussian mixture model. In this paper we propose a semi-supervised learning method for PRBF, using labeled and unlabeled observations concurrently, that is based on the expectation–maximization (EM) algorithm. Next we utilize this method in order to implement an incremental active learning method. At each iteration of active learning, we apply the semi-supervised method, and then we employ a criterion to select an unlabeled observation and query its label. This criterion identifies points near the decision boundary. In order to assess the effectiveness of our method, we propose an adaptation of the well-known Query by Committee (QBC) algorithm for the active learning of the PBFR, and present experimental comparisons on several data sets that indicate the effectiveness of the proposed method.

1. Introduction

The active learning of a classifier constitutes a special learning problem, where the training data are available as a stream of classified observations and are *actively collected* by the classifier during training. At each learning iteration, the learning system determines regions of interest in the data space, asks for labeled training data that lie in these regions, and exploits the acquired class labels to improve its classification performance.

The importance of active learning is well established, see [3] for a study on the increase of classifier's accuracy as the number of labeled data increases. Various active learning methods have been suggested for almost all types of classifiers; in [5] a learning method for Gaussian mixture models [15] has been proposed, that selects data minimizing the variance of the learner. The Query by Committee (QBC) algorithm has been proposed in [17,9] for the active learning of a committee of classifiers, which picks those data for which the committee members disagree. Based on this selection method, in [13] they proposed the exploitation of available unlabeled data by employing the EM algorithm [8] to form a better selection criterion, that is used to train a naive Bayes classifier. In [25] Gaussian random fields and harmonic functions are trained, while data selection is based on the estimated expected classification error. In [23] an active learning methodology for support vector machine (SVM) classifiers has been proposed with applications to text classification.

In this work we focus on *pool-based* active learning which is well-studied active learning problem [13,23,25]. In this case a set of labeled and unlabeled observations is available right from the start. At each training iteration we are allowed to query the labels of unlabeled points, and use the acquired labels to improve the classifier (see Fig. 1). In practice this learning scenario is important in two cases: (a) when querying a field expert is expensive, as in medical diagnosis, and (b) when there is a huge quantity of unlabeled data and is difficult to manually characterize all of them, as for example in document classification and e-mail filtering [23,11].

An intuition behind pool-based learning is that the unlabeled data can be exploited to construct a more detailed generative model for the data set. Thus this problem is closely related to *semi-supervised* learning. Algorithms for semi-supervised learning have been proposed for Gaussian mixtures in [10,18], as well as for the RBF network [16]. So it has been established that unlabeled data reveal useful information for the distribution of the labeled data (see [4] for an informative recent survey book on semi-supervised learning).

In order to implement active learning the following issues must be addressed:

- Define an effective *criterion* for selecting the unlabeled points to query their label.
- Use an *incremental* training algorithm so that learning does not start from scratch at each iteration.

* Corresponding author. Tel.: +30 26510 98810; fax: +30 26510 98882.
E-mail addresses: ccostas@cs.uoi.gr (C. Constantinopoulos), arly@cs.uoi.gr (A. Likas).

[1] This research was co-funded by the European Union in the framework of the program "Heraklitos" of the "Operational Program for Education and Initial Vocational Training" of the 3rd Community Support Framework of the Hellenic Ministry of Education, funded by 25% from national sources and by 75% from the European Social Fund (ESF).

0925-2312/$ - see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.neucom.2007.11.039

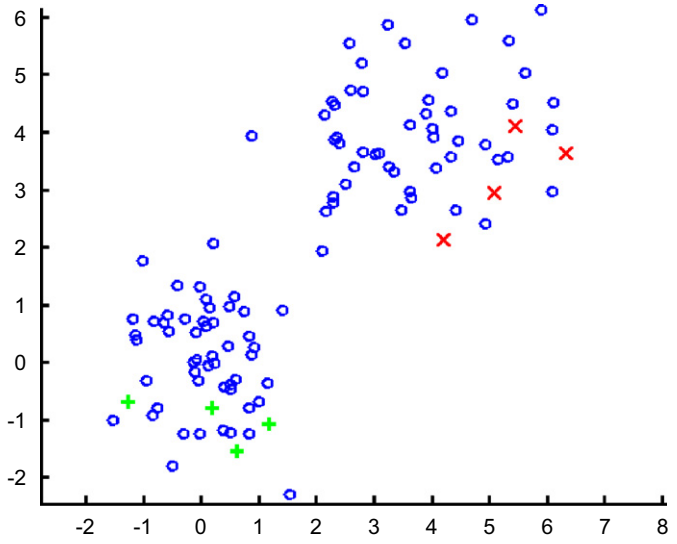

Fig. 1. Available data in a pool-based active learning scenario: "○" denotes unlabeled data, "+", "×" denote class labels. As learning proceeds "○" points are selected and their label ("+" or "×") is revealed.

- Exploit during training not only the labeled but also the unlabeled data (use *semi-supervised* learning training algorithms).

In this work we treat the problems of semi-supervised learning and pool-based active learning for the probabilistic RBF (PRBF) classifier [20,22]. This is a special case of the RBF network [2] that computes at each output unit the density function of a class. It adopts a cluster interpretation of the basis functions, where each cluster can generate observations of any class. This is a generalization of a Gaussian mixture model [15,2], where each cluster generates observations of only one class. In [6] an incremental learning method based on expectation–maximization (EM) for supervised learning is proposed that provides classification performance comparable to SVM classifiers.

The core of our work is the proposition of a *semi-supervised* learning method for the PRBF network, which is based on the EM algorithm for maximization of the *joint likelihood* of both the labeled and unlabeled data. Thus we obtain closed form update equations for the network parameters. We are facilitated by the fact that each node of the PRBF describes the local density of potentially all the classes. In order to handle the unlabeled data, it is possible to marginalize the class labels from the update equations of EM, thus using both labeled and unlabeled data in parameter estimation. Building on this method, we further present an incremental method for semi-supervised training. Exploiting the later method, we develop an *active* learning framework, by defining a suitable criterion for selecting the unlabeled observations, and asking for their label. Our contribution is concluded with the application of the QBC algorithm for the active learning of the PRBF, and its experimental comparison with the proposed algorithm.

In the following section we describe the PRBF network, and in Section 3 we present the semi-supervised training method based on the EM algorithm. In Section 4 we first propose an incremental semi-supervised training method, and in the following we propose an active learning algorithm. In the same section we also present the application of the QBC algorithm for the active learning of the PRBF. Section 5 provides the results from our experimental study, while the discussion in Section 6 concludes this work.

2. The PRBF classification network

Consider a classification problem with K classes, where K is known and each pattern belongs to only one class. We are given a training set $X=\{(x^n,y^n), n=1,\ldots,N\}$ where $x^{(n)}$ is a d-dimensional pattern, and y^n is a label $k\in\{1,\ldots,K\}$ indicating the class of pattern x^n. The original set X can be partitioned into K independent subsets X_k, so that each subset contains only the data of the corresponding class. Let N_k denote the number of patterns of class k, i.e. $N_k=|X_k|$.

The PRBF network has the same architecture as the typical RBF network, i.e. an input layer with d units for the input vector $x=(x_1,\ldots,x_d)$, an output layer with K units (one for each class) and a single hidden layer with an arbitrary number M of component functions (hidden units), which are probability density functions. In the PRBF network all component density functions $p(x|j)$, $(j=1,\ldots,M)$ are utilized for estimating the conditional densities of all classes by considering the components as a common pool [19,20]. The k-th PRBF output models the class conditional density function $p(x|k)$ of class k as a mixture model of the form

$$p(x|k)=\sum_{j=1}^{M}p(j|k)p(x|j),\quad k=1,\ldots,K, \tag{1}$$

where $p(x|j)$ denotes the component density j, while the mixing coefficient $p(j|k)$ represents the prior probability that a pattern has been generated from the density function of component j, given that it belongs to class k. The priors take positive values and satisfy the following constraint:

$$\sum_{j=1}^{M}p(j|k)=1,\quad k=1,\ldots,K. \tag{2}$$

Once the outputs $p(x|k)$ have been computed, the class of data point x is determined using the Bayes rule, i.e. x is assigned to the class with maximum posterior $p(k|x)$ computed by

$$p(k|x)=\frac{p(x|k)p(k)}{\sum_{\ell=1}^{K}p(x|l)p(\ell)}. \tag{3}$$

The class priors $p(k)$ are computed as $p(k)=N_k/N$, according to the maximum likelihood solution.

Also using Bayes theorem, the posterior probabilities $p(j|x,k)$ that component j generated a pattern x given its class k can be easily computed:

$$p(j|x,k)=\frac{p(j|k)p(x|j)}{\sum_{i=1}^{M}p(i|k)p(x|i)}. \tag{4}$$

In the following, we assume Gaussian component densities of the general form:

$$p(x|j)=\frac{1}{(2\pi)^{d/2}|\Sigma_j|^{1/2}}\exp\left\{-\frac{1}{2}(x-\mu_j)^{\mathrm{T}}\Sigma_j^{-1}(x-\mu_j)\right\}, \tag{5}$$

where $\mu_j\in\Re^d$ represents the mean of component j, while Σ_j represents the corresponding $d\times d$ covariance matrix. The whole adjustable parameter vector of the model consists of the mixing coefficients $p(j|k)$ and the component parameters (means μ_j and covariance matrices Σ_j) and we denote it by Θ.

It is apparent that the PRBF model is a special case of the RBF network, where the outputs correspond to probability density functions and the second layer weights are constrained to represent the prior probabilities $p(j|k)$. Furthermore, the separate mixtures model [14] can be derived as a special case of PRBF, if each component j generates only patterns of the class ℓ, and we set $p(j|k)=0$ for all classes $k\neq\ell$.

Training the PRBF network is performed using the EM algorithm for maximization of the likelihood with respect to the parameters [20]:

$$L(\Theta) = \sum_{k=1}^{K}\sum_{x\in X_k}\log p(x|k) = \sum_{k=1}^{K}\sum_{x\in X_k}\log\sum_{j=1}^{M}p(j|k)p(x|j). \quad (6)$$

EM is an iterative procedure with two steps at each iteration. During the *expectation* step, posterior probabilities $p(j|x,k)$ are computed using the current estimates of $p(j|k)$, μ_j and Σ_j, according to

$$p(j|x,k) = \frac{p(j|k)p(x|j;\mu_j,\Sigma_j)}{\sum_{i=1}^{M}p(i|k)p(x|i;\mu_i,\Sigma_i)}. \quad (7)$$

During the *maximization* step the new estimates of the component parameters are updated according to

$$\mu_j = \frac{\sum_{k=1}^{K}\sum_{x\in X_k}p(j|x,k)x}{\sum_{\ell=1}^{K}\sum_{x\in X_\ell}p(j|x,\ell)}, \quad (8)$$

$$\Sigma_j = \frac{\sum_{k=1}^{K}\sum_{x\in X_k}p(j|x,k)(x-\mu_j)(x-\mu_j)^{\mathrm{T}}}{\sum_{\ell=1}^{K}\sum_{x\in X_\ell}p(j|x,\ell)}, \quad (9)$$

$$p(j|k) = \frac{1}{|X_k|}\sum_{x\in X_k}p(j|x,k), \quad k=1,\ldots,K. \quad (10)$$

The EM updates eventually will converge to a maximum of the likelihood.

2.1. Component splitting

In [21] a hierarchical method for classification has been proposed based on the PRBF network that contains two stages: in the first stage (*EM-stage*), a PRBF network with a fixed number of components M is trained using the EM update equations (7)–(10). After the completion of this stage of training, there may be components of the network with mean vectors located on the decision boundary. In these regions of the data space there is an overlapping among classes. This happens if we have underestimated the maximum allowed number of components. In order to increase the generalization performance of the network, it is suggested in [21] to split each component. This happens in the second stage (*splitting stage*) of the training method, where every PRBF component j is splitted into subcomponents jk corresponding to the classes of the problem. This is achieved by evaluating the posterior probability $p(j|x,k)$ for a component to define if it is responsible for patterns of more than one class. So we compute $p(j|x,k)$ for every pattern $x\in X$, and check if $\sum_{x\in X_k}p(j|x,k)>0$ for more than one class k. If this happens, then we remove it from the network, and add a separate component for each class. So finally every subcomponent describes only one class. Splitting a component j, the resulting subcomponent of class k is a Gaussian probability density function $p(x|j,k)$, with mean μ_{jk}, covariance matrix Σ_{jk} and mixing weight $p(j|k)$. These parameters are estimated according to

$$p(j|k) = \frac{1}{|X_k|}\sum_{x\in X_k}p(j|x,k), \quad (11)$$

$$\mu_{jk} = \frac{\sum_{x\in X_k}p(j|x,k)x}{\sum_{x\in X_k}p(j|x,k)}, \quad (12)$$

$$\Sigma_{jk} = \frac{\sum_{x\in X_k}p(j|x,k)(x-\mu_{jk})(x-\mu_{jk})^{\mathrm{T}}}{\sum_{x\in X_k}p(j|x,k)}. \quad (13)$$

After splitting, the class conditional density is

$$p(x|k) = \sum_{j=1}^{M}p(j|k)p(x|j,k), \quad k=1,\ldots,K. \quad (14)$$

It has been shown in [21] that the addition of the splitting stage, on the one hand guarantees the increase of the likelihood and on the other hand leads to considerable improvement in generalization performance compared to the PRBF network obtained in the first stage. This two-stage approach is called the *PRBF-split* method.

Fig. 2 provides a characteristic example illustrating the effect of the splitting operation. A remark that can be made is that, from a classification point of view, a full effect exploitation of the splitting operation is achieved, if the components of the PRBF network have been placed in regions containing the decision boundary between classes (see Fig. 3). This remark led to the development of an incremental method [6] for placing the components of the PRBF network constructed in the first stage (EM training).

3. Semi-supervised PRBF training

In this section we develop an EM algorithm for semi-supervised training of the PRBF network. We assume a set of labeled observations $X=\{(x^n,y^n)|n=1,\ldots,N\}$ and a set of

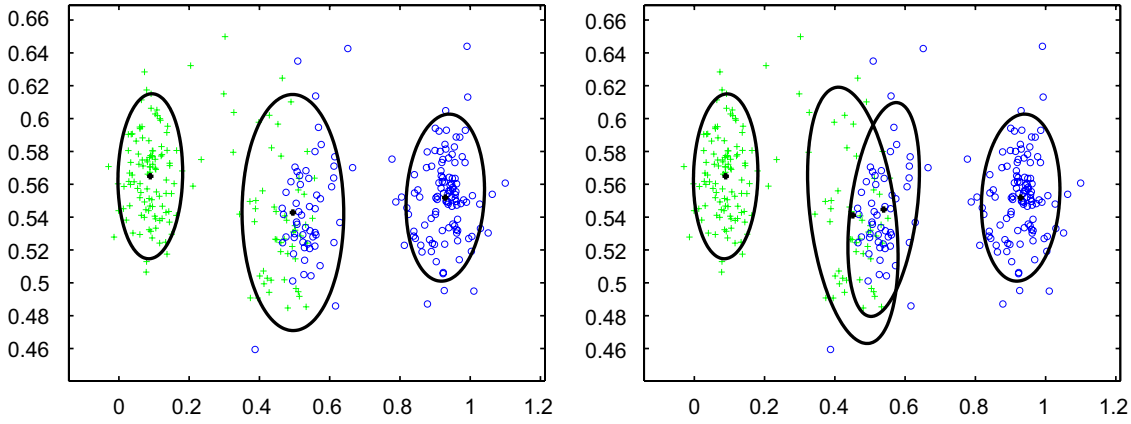

Fig. 2. An example of the component splitting process. The component in the middle is located in a region with data of two classes and is splitted into two subcomponents each describing the data of one class.

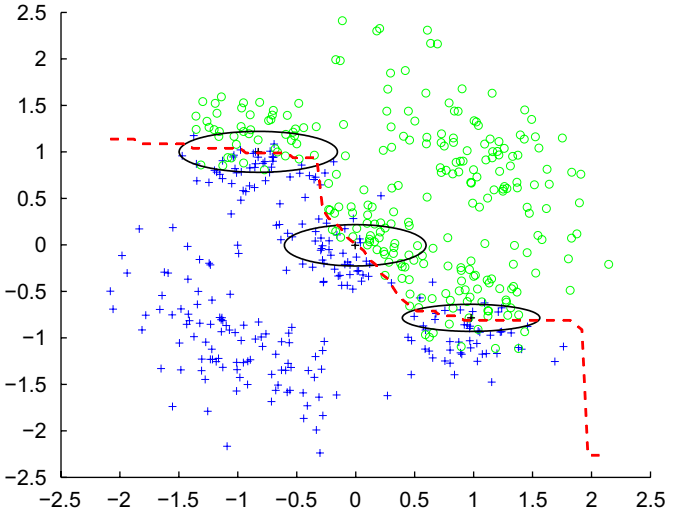

Fig. 3. Desirable placement of components on the decision boundary (shown with dashed line) in the first stage on hierarchical training. A subsequent splitting of each component will provide a satisfactory solution to the classification problem.

unlabeled observations $X_U = \{x^n | n = 1, \ldots, N_U\}$. The labeled observations have an "input" part $x \in \Re^d$, and an "output" part $y \in \{1, \ldots, K\}$ in the case of a classification problem with K classes. This "output" part (called label) assigns an observation to one class, and in the case of unlabeled observations is missing. Let Ω be the joint set of labeled and unlabeled observations, i.e. $\Omega = X \cup X_U$. Moreover we can separate X according to the class labels in K disjoint sets $X_k = \{(x^n, y^n) | y^n = k, n = 1, \ldots, N_k\}$ one for each class, then $\Omega = \bigcup_k X_k \cup X_U$.

Adopting Bayesian reasoning, a classifier assigns a new unlabeled observation x to the class $k^\star$ with maximum posterior probability. If we drop the part of the posterior that depends only on x, then

$$k^\star = \arg\max_k p(x|k)p(k), \tag{15}$$

where $p(x|k)$ is the class conditional distribution of observations from class k, and $p(k)$ is the prior probability of this class. For two classes k and k' there is a *decision boundary* $p(x|k)p(k) = p(x|k')p(k')$ that divides the space of the observations.

To describe class conditional distributions we employ the PRBF network with M basis functions. For input x the class conditional probability $p(x|k)$ is the k-th output of the PRBF

$$p(x|k) = \sum_{j=1}^{M} p(j|k)p(x|j), \tag{16}$$

where each basis function is a Gaussian with mean $\mu_j \in \Re^d$ and covariance matrix Σ_j. In order to find estimates for the parameters of the network

$$\Theta = \{p(k), p(j|k), \mu_j, \Sigma_j | j = 1, \ldots, M, k = 1, \ldots, K\}$$

we maximize the *joint likelihood*, as in [16]. Assuming i.i.d. observations, the joint log-likelihood $\mathcal{L}$ of labeled and unlabeled data is

$$\begin{aligned}\mathcal{L} &= \log\left\{\prod_k \prod_{x \in X_k} p(x,k) \prod_{x \in X_U} p(x)\right\} \\ &= \sum_k \sum_{x \in X_k} \log p(k) \sum_j p(j|k)p(x|j) \\ &\quad + \sum_{x \in X_U} \log \sum_k p(k) \sum_j p(j|k)p(x|j).\end{aligned} \tag{17}$$

The density $p(x)$ of the unlabeled data was computed by marginalizing the class label from the joint density $p(x,k) = p(x|k)p(k)$ as follows:

$$p(x) = \sum_k p(x,k) = \sum_k p(k)p(x|k) = \sum_k p(k) \sum_j p(j|k)p(x|j). \tag{18}$$

For the maximization of $\mathcal{L}$ we use the EM algorithm [8]. The EM is an iterative algorithm that is guaranteed to converge at a local maximum of the likelihood surface. It is employed in problems where *hidden variables* exist. These variables determine the solution of the problem, although they are not observable. In our case the hidden variables define the node of the network that generated an observation, and the label of an unlabeled observation. In order to be able to apply the EM algorithm, it should be possible to compute the *posterior probability* of the hidden variables given the observations. Based on this remark, in the following we formally derive the EM update equations for semi-supervised PRBF training.

For each unlabeled observation x, we introduce a hidden variable $z^{(x)}$ that assigns this observation to one class k and one basis function j. Thus, each $z^{(x)}$ is a binary $M \times K$ matrix, where $z^{(x)}_{jk} = 1$ if x is assigned to the k-th class and the j-th node. This assignment is unique, so that $\sum_j \sum_k z^{(x)}_{jk} = 1$. Moreover for a labeled observation (x,k), the corresponding $z^{(x)}$ is constrained so that $z^{(x)}_{j\ell} = 0$ for every class $\ell \neq k$ and for all j. Thus a hidden variable can assign a *labeled* observation to any node but only one class. This does not hold for the case of *unlabeled* observations that can be assigned to any class and any node. Given the set of hidden variables $Z = \{z^{(x)} | \forall x \in \Omega\}$, we define the *complete* log-likelihood, as the logarithm of the joint density $p(\Omega, Z)$ of both the observations and the hidden variables:

$$\mathcal{Q} = \log \prod_{x \in \Omega} \prod_k \prod_j [p(k)p(j|k)p(x|j)]^{z^{(x)}_{jk}}. \tag{19}$$

Then the EM algorithm requires the computation of the expectation $\langle \mathcal{Q} \rangle$ w.r.t. the distribution of Z. Since the expected value of $z^{(x)}_{jk}$ is equal to the joint posterior probability $p(j,k|x)$ that x is assigned to the j-th node and the k-th class, it follows that

$$\langle \mathcal{Q} \rangle = \sum_{x \in \Omega} \sum_k \sum_j p(j,k|x) \log\{p(k)p(j|k)p(x|j)\}. \tag{20}$$

The EM algorithm iterates two steps until convergence. During the *E-step*, the posterior probability of the hidden variables given the observations is computed, in order to define the expectation of the complete log-likelihood $\langle \mathcal{Q} \rangle$, given the current estimate for the parameter vector Θ. During the *M-step* it provides estimates Θ that maximize $\langle \mathcal{Q} \rangle$. This procedure is guaranteed to converge at a local maximum of the joint log-likelihood $\mathcal{L}$ (17).

Explicitly described, during the E-step we compute $p(j,k|x)$ for every $x \in \Omega$, $j \in \{1, \ldots, M\}$ and $k \in \{1, \ldots, K\}$ according to

$$p(j,k|x) = p(j|k,x)p(k|x). \tag{21}$$

If x is unlabeled then we compute $p(k|x)$ and $p(j|k,x)$ for every class k using Bayes theorem

$$p(k|x) = \frac{p(x|k)p(k)}{\sum_\ell p(x|\ell)p(\ell)}, \tag{22}$$

$$p(j|k,x) = \frac{p(j|k)p(x|j)}{\sum_i p(i|k)p(x|i)}. \tag{23}$$

If x is labeled, then we exploit the information of the label and set

$$p(k|x) = \begin{cases} 1 & \text{if } x \in X_k, \\ 0 & \text{if } x \notin X_k \end{cases} \tag{24}$$

and we compute $p(j|k,x)$ similarly

$$p(j|k,x)=\begin{cases}\dfrac{p(j|k)p(x|j)}{\sum_i p(i|k)p(x|i)} & \text{if } x\in X_k,\\ 0 & \text{if } x\notin X_k.\end{cases} \tag{25}$$

During the M-step we maximize $\langle\mathcal{Q}\rangle$ w.r.t. θ, given the current estimation of the joint posteriors. Setting the derivatives equal to zero, we get the following solution for every $j\in\{1,\ldots,M\}$ and $k\in\{1,\ldots,K\}$ is

$$\mu_j=\frac{\sum_{x\in\Omega}\sum_k p(j,k|x)x}{\sum_{x\in\Omega}\sum_k p(j,k|x)}, \tag{26}$$

$$\Sigma_j=\frac{\sum_{x\in\Omega}\sum_k p(j,k|x)(x-\mu_j)(x-\mu_j)^{\mathrm{T}}}{\sum_{x\in\Omega}\sum_k p(j,k|x)}, \tag{27}$$

$$p(j|k)=\frac{\sum_{x\in\Omega}p(j,k|x)}{N_k+\sum_j\sum_{x\in X_{\mathrm{U}}}p(j,k|x)}, \tag{28}$$

$$p(k)=\frac{N_k+\sum_j\sum_{x\in X_{\mathrm{U}}}p(j,k|x)}{N+N_{\mathrm{U}}}. \tag{29}$$

It must be noted that the class priors $p(k)$ are updated during EM iterations, while in the supervised case they are fixed equal to the proportion of the data of each class.

3.1. Semi-supervised component splitting

As in the typical supervised case described in Section 2.1, after the semi-supervised training described above, there may be nodes of the network located to regions with overlapping among classes. For this reason, we can apply semi-supervised node splitting using both labeled and unlabeled observations. First we evaluate the joint posterior probabilities $p(j,k|x)$ for a node, and define if it is responsible for observations of more than one class. If $\sum_{x\in\Omega}p(j,k|x)>0$, then we remove it from the network, and add a separate node for the k-th class. So finally each node is responsible for only one class. Splitting a node $p(x|j)$, the resulting node for class k is a Gaussian $p(x|j,k)$ with mean μ_{kj}, covariance Σ_{kj} and mixing weight $p(j|k)$. These parameters are estimated according to

$$\mu_{kj}=\frac{\sum_{x\in\Omega}p(j,k|x)\,x}{\sum_{x\in\Omega}p(j,k|x)}, \tag{30}$$

$$\Sigma_{kj}=\frac{\sum_{x\in\Omega}p(j,k|x)\,(x-\mu_{kj})(x-\mu_{kj})^{\mathrm{T}}}{\sum_{x\in\Omega}p(j,k|x)}, \tag{31}$$

$$p(j|k)=\frac{\sum_{x\in\Omega}p(j,k|x)}{N_k+\sum_j\sum_{x\in X_{\mathrm{U}}}p(j,k|x)}. \tag{32}$$

Consequently the class conditional density is estimated as

$$p(x|k)=\sum_j p(j|k)p(x|j,k). \tag{33}$$

In the case of a training set where all the points are labeled, the class conditional likelihood is increased for all classes after splitting as proved in [21]. However, in the semi-supervised case we cannot guarantee that splitting increases the joint likelihood.

It can be observed that in the case where there are no unlabeled data both the EM update equations and the equations for component split reduce to the supervised training equations described in Section 2.

4. Pool-based active learning using PRBF

As noted in the Introduction in order to develop an efficient algorithm for pool-based active learning, it is desirable to have an incremental training algorithm that exploits both labeled and unlabeled data. In the following subsection, based on the results of Section 3 and the supervised incremental algorithm presented in [6], we present an incremental algorithm for semi-supervised learning of PRBF.

4.1. Incremental semi-supervised addition of nodes

The incremental semi-supervised training algorithm contains two stages: the EM-stage where a PRBF network is incrementally constructed and the split-stage, where the nodes of the resulting network are splitted using the equations for semi-supervised split.

Consider a PRBF network with M components in the first stage of training. In order to construct a network with $M+1$ components, the procedure of component addition involves search in the parameter space, to define the parameters of the new component. More specifically, the algorithm searches among a set of candidate regions in the data space to place the new component, and selects the most appropriate candidate according to certain criteria. Then, the semi-supervised EM algorithm is used to adjust the parameters of the resulting network with $M+1$ components. This procedure of sequential component addition starts with one component and is repeated until some stopping condition is met.

Assuming a network with M components and parameter vector Θ_M, the conditional density of each class k is $p(x|k;\Theta_M)$. In the case where a new component $j=M+1$ is added with density $f_{M+1}(x)$, each new class conditional density $p(x|k;\Theta_{M+1})$ is defined as a mixture of the current model $p(x|k;\Theta_M)$ and the new component $f_{M+1}(x)$:

$$p(x|k;\Theta_{M+1})=(1-\alpha_k)p(x|k;\Theta_M)+\alpha_k f_{M+1}(x), \tag{34}$$

where α_k $(k=1,\ldots,K)$ are the mixing weights for the new component and $\alpha_k\in(0,1)$. This is analogous with the incremental training procedure called greedy-EM proposed in [24] for unsupervised probability density estimation. Using the above combination formula, the resulting network is again a PRBF network. The log-likelihood $\mathcal{L}(\Theta_{M+1})$ of the model with $M+1$ components is

$$\mathcal{L}(\Theta_{M+1})=\sum_{k=1}^{K}\sum_{x\in X_k}\log\{(1-\alpha_k)p(x|k;\Theta_M)+\alpha_k f_{M+1}(x)\}. \tag{35}$$

The crucial task during component addition is related to the determination of the parameters of the new component. This is accomplished through a search procedure among *a set of candidate solutions*, since it is not possible to directly specify a single good component to add. Thus we define a set of *candidate initial* component parameters and the best parameter values (μ,Σ,a_k) (obtained according to a specific criterion) are considered as the final component parameters.

Let M be the current number of PRBF components. In order to determine the candidate initial component parameters, we partition the labeled data set X into M subsets $S_j=\{x|P(j|x)>P(i|x),\forall i\neq j\}$, based on the posterior probabilities $P(j|x)$ obtained by marginalizing class labels:

$$P(j|x)=\sum_{k=1}^{K}P(j|x,k)P(k), \tag{36}$$

with $P(k)=|X_k|/|X|$ being the prior probability of class k. For each of the M sets S_j, a subset of candidate components is created by partitioning its data using the *kd-tree* approach. A *kd-tree* [1] defines a recursive partitioning of the data space into disjoint subspaces. It is a binary tree, where the data associated with any

non-terminal node are partitioned using a *cutting hyperplane*, to specify the successor's nodes. To partition the data points of a node we have followed the approach used in [12]: the cutting hyperplane is defined to be perpendicular to the direction of the principal component of the data corresponding to the node. Fig. 4 illustrates the partitioning stages for an artificial data set. The procedure of recursive partitioning is applied until level four (tree depth), and we consider all tree nodes (not only leaf nodes) to define overlapping subsets of S_j (i.e. 14 subsets for each component j). The statistics (sample mean and covariance) of each of the $14M$ subsets constitute candidate initial parameter values for the component $M+1$. The values of α_k are set equal to $\pi_{jk}/2$ for the subsets derived from partitioning the subset S_j.

As previously mentioned, we wish the new component to be placed at regions in the data space containing examples of more than one class (see Fig. 3). In this case, a subsequent component split would provide a sensible placement of the resulting subcomponents on the decision boundary. A way to quantify the degree to which a candidate component satisfies this property is to compute *the change of the log-likelihood for class k*, caused by the addition of the candidate new component l with density $p(x;\theta^l)$, according to (34). So we define the change $\Delta\mathscr{L}_k^l$ for class k as

$$\Delta\mathscr{L}_k^l = \frac{1}{|X_k|}(\mathscr{L}_k(\Theta_{M+1}^l) - \mathscr{L}_k(\Theta_M)) = \frac{1}{|X_k|}\sum_{x\in X_k}\log\left\{1-\alpha_k+\alpha_k\frac{p(x;\theta^l)}{p(x|k;\Theta_M)}\right\}, \quad (37)$$

where $\Theta_{M+1}^l = \Theta_M \cup \theta^l$. Based on the values $\Delta\mathscr{L}_k^l$, we search among the candidate components l to determine those whose addition causes an *increase in the log-likelihood for at least two classes*. Such candidates lie in a region containing data of more than one class, consequently on a decision boundary. In order to find the best candidate, we retain the components that increase the log-likelihood of at least two classes and discard the rest. For each retained component l, we add the positive $\Delta\mathscr{L}_k^l$ terms to compute the *total increase of the log-likelihood* $\Delta\mathscr{L}_l$. The candidate $l^\star$ whose value $\Delta\mathscr{L}_{l^\star}$ is maximum is added to the current model PRBF(M), if this maximum value is higher than a predefined threshold (set equal to 0.01 in all experiments). Otherwise, we consider that the attempt to add a new component is unsuccessful and terminate the first stage of training with a PRBF model with M components.

After the successful addition of a new node we apply the semi-supervised EM to the PRBF network with $M+1$ components, as described in the previous section. It must be noted that in this case the use of unlabeled data provide an additional important benefit. This is related to the avoidance of singular solutions (i.e. the variance of some component tends to zero), that is a typical problem of EM when the data set in sparse. In the first iterations of active learning, the available labeled are few and there may be numerical problems when applying the typical EM algorithm for supervised learning (Section 2). The exploitation of unlabeled data in the semi-supervised case assists the EM algorithm in avoiding singular solutions.

The above incremental procedure of node addition followed semi-supervised training can be applied many times, in order to add the desired number of nodes to the given network. Fig. 5 illustrates the addition of the first two nodes. The initial network with only one node is illustrated in Fig. 5(a). The six candidate nodes and the chosen node are illustrated in Fig. 5(b) and (c) correspondingly. Fig. 5(d) illustrates the network after the application of semi-supervised EM.

After the stage of adding nodes, the equations for semi-supervised node splitting are applied to provide the final solution to the incremental semi-supervised PRBF training problem.

4.2. The active learning algorithm

In the previous subsection we described an incremental algorithm for training a PRBF network using labeled and unlabeled observations. In the following we incorporate the algorithm in an active learning framework, where we iteratively select an unlabeled point and query its label. After its label is given, we add the labeled point in the labeled set and train the network again. The only issue that remains to be specified is related to the specification of the criterion used to select the unlabeled data point to ask for its label. We propose the selection of a point that lies near the classification boundary. In this way we facilitate the iterative addition of basis functions on the decision boundary, as described in the previous section.

As a criterion for selecting a suitable point we propose the ratio of class posteriors. For each unlabeled observation $x \in X_U$ we compute the class posterior $p(k|x)$ for every class, and then find the two classes with the largest posterior values:

$$\kappa_1^{(x)} = \arg\max_k p(k|x), \quad \kappa_2^{(x)} = \arg\max_{k\neq\kappa_1^{(x)}} p(k|x). \quad (38)$$

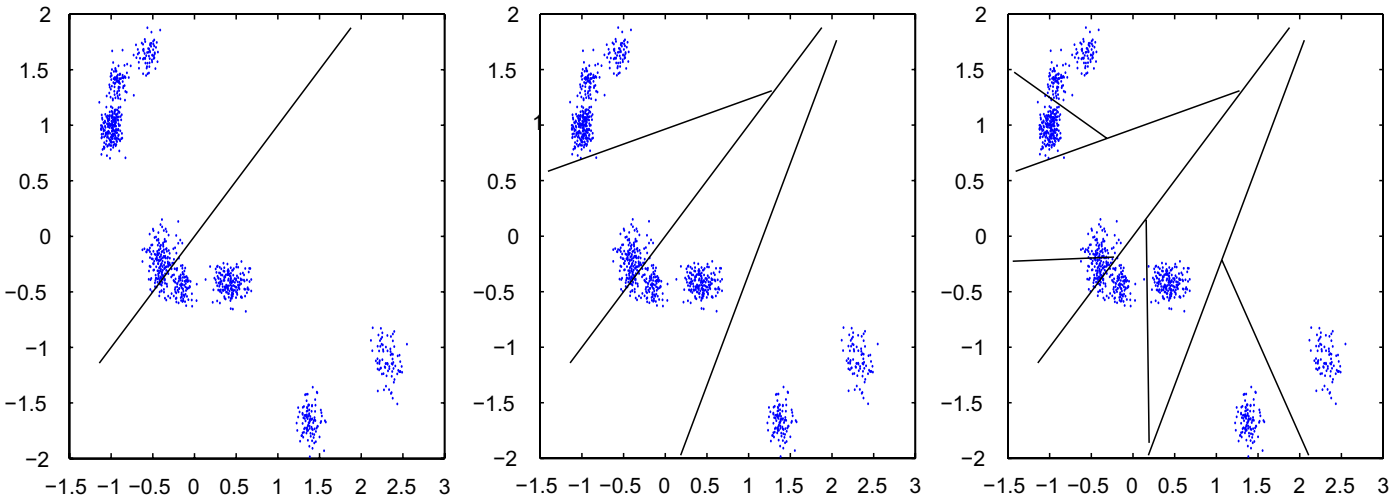

Fig. 4. Successive partitioning of an artificial data set overlapping partitions using the *kd*-tree method. All the 14 partitions illustrated in the three graphs are considered to specify candidate parameter vectors.

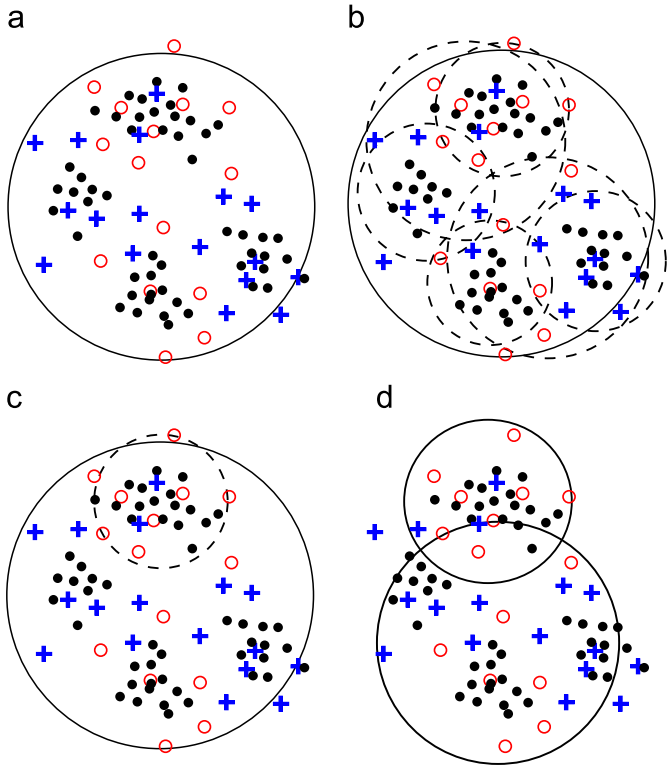

Fig. 5. Addition of the first two basis functions. The nodes of the network are drawn with solid lines, and the candidate nodes with dashed lines. The dots represent the unlabeled observations in a two-class problem.

We choose to ask for the label of $\hat{x}$ that exhibits the smallest ratio of largest class posteriors:

$$\hat{x} = \arg\min_{x \in X_U} \frac{p(\kappa_1^{(x)}|x)}{p(\kappa_2^{(x)}|x)}. \tag{39}$$

In this way we pick the unlabeled observation that lies closer to the decision boundary of the current classifier. Note that according to (15), we classify observations to the class with the maximum class posterior. Thus for some x on the decision boundary holds that $p(\kappa_1^{(x)}|x) = p(\kappa_2^{(x)}|x)$. Consequently if an observation approaches the decision boundary between two classes, then the corresponding logarithmic ratio of class posteriors tends to zero.

Summarizing the presented methodology, we propose the following active learning algorithm:

1. Input: The set X of labeled observations, the set $X_\emptyset$ of unlabeled observations, and a degenerate network $PRBF_{J=1}$ with one basis function.
2. For $s = 0, \ldots, S-1$:
   (a) Add one node to the network $PRBF_{J+s}$ to form $PRBF_{J+s+1}$.
   (b) Apply EM until convergence for semi-supervised training of $PRBF_{J+s+1}$.
3. For $s = 0, \ldots, S$:
   (a) Split the nodes of $PRBF_{J+s}$ to form $PRBF_{J+s}^{\text{split}}$.
4. Select the network $PRBF_{J^\star}^{\text{split}} \in \{PRBF_J^{\text{split}}, \ldots, PRBF_{J+S}^{\text{split}}\}$ that maximizes the joint likelihood.
5. Set the current network: $PRBF_J = PRBF_{J^\star}$.
6. If $X_\emptyset$ is empty go to step 7, else
   (a) Pick an unlabeled observation $\hat{x}$ according to (39), and ask its label $\hat{y}$.
   (b) Update the sets: $X = X \cup \{(\hat{x}, \hat{y})\}$ and $X_\emptyset = X_\emptyset \backslash \{\hat{x}\}$.
   (c) Go to step 2.
7. Output: Split the nodes of $PRBF_J$ to form the output network $PRBF_J^{\text{split}}$.

In all our experiments we use $S = 1$, thus we try to add one node at each iteration of the active learning.

4.3. The QBC algorithm

In order to compare the proposed algorithm with another active learning technique, in the following we briefly discuss the QBC method [17,9], and we describe its application to the PRBF classifier. At each iteration of the QBC method, it constructs a committee of classifiers, and uses its members to classify unlabeled observations. Using some measure of disagreement, the method selects the observation for which the members of the committee disagree the most. The label of this observation is added in the training set, and the procedure is repeated.

The crucial task for the QBC method is the construction of the committee. In [9] they assume a distribution over the classifiers, and they sample the members of the committee from the subset of valid classifiers, which classify correctly the current training set. Thus, when QBC selects to add in the training set a label that causes disagreement, the number of valid classifiers is reduced.

The use of the QBC with a probabilistic classifier has been proposed in [7] for training hidden Markov models for natural language processing, and also in [13] for the pool-based active learning of a naive Bayes classifier in the context of text classification. At each iteration of QBC they train a classifier, and they propose the sampling of the parameters for each member of the committee w.r.t. to the posterior distribution of the parameters of the current classifier given the current training set. During training, in [7] they employ the EM to predict the missing labels and improve the accuracy of the classifier, exploiting the semi-supervised learning. It is easy to adapt this approach for the PRBF classifier. Although we have to examine two important stages of the method, namely the creation of the committee members, and the measure of disagreement between them.

We sample the parameters of each member of the committee w.r.t. to the posteriors of the parameters of the current PRBF classifier. After sampling, we apply EM for semi-supervised training of the members, in order to improve their generalization performance. So we can make some assumptions to facilitate the sampling. Although the Gaussian components of the PRBF are not independent, we make the simplifying assumption that we can sample the parameters of each Gaussian independently of the rest. The next assumption is that the features of the data are independent, so that the covariance matrix of each component is diagonal. Thus we can sample each of the d components of the mean vector independently, and the same holds for the d diagonal components of the covariance matrix. We also assume conjugate priors, consequently for the j-th Gaussian component the prior of each mean component $\mu_{j(d)}$ is the standard normal distribution, the prior of each inverse variance component $\tau_{j(d)} = 1/\sigma_{j(d)}^2$ is gamma $\mathcal{G}(\tau_{j(d)}|\hat{\alpha}, \hat{\beta})$, and the prior of the mixing weights $\{\pi_{jk} = p(j|k) | j = 1, \ldots, M\}$ given class k is Dirichlet $\mathcal{D}(\pi_{1k}, \ldots, \pi_{Mk}|\hat{\theta}_1, \ldots, \hat{\theta}_M)$. We set all parameters of the priors equal to unit, so that $\hat{\alpha} = \hat{\beta} = \hat{\theta}_1 = \cdots = \hat{\theta}_M = 1$. Consequently, we sample the parameters of the j-th Gaussian component from thefollowing posterior distributions:

$$\mu_{j(d)} \sim \mathcal{N}(\mu_{j(d)}|m_j, s_j), \tag{40}$$

$$\tau_{j(d)} \sim \frac{1}{\Gamma(\alpha_j)} \beta_j^{\alpha_j} \tau_{j(d)}^{\alpha_j - 1} \exp(-\beta_j \tau_{j(d)}) \tag{41}$$

and we sample the weights of the k-th class according to

$$\pi_{1k},\ldots,\pi_{Mk}\sim\frac{\Gamma(\sum_j\theta_{jk})}{\prod_j\Gamma(\theta_{jk})}\prod_{j=1}^{M}\pi_{jk}^{\theta_{jk}-1}. \quad (42)$$

The parameters of the posteriors are the following:

$$m_j=\frac{\hat{\eta}_j\,\hat{\mu}_{j(d)}}{\hat{\eta}_j+\hat{\sigma}^2_{j(d)}},\quad \frac{1}{s_j}=1+\frac{\hat{\eta}_j}{\hat{\sigma}^2_{j(d)}}, \quad (43)$$

$$\alpha_j=\hat{\alpha}+\frac{\hat{\eta}_j}{2},\quad \beta_j=\hat{\beta}+\frac{\hat{\eta}_j}{2}\hat{\sigma}^2_{j(d)}, \quad (44)$$

$$\theta_{jk}=\hat{\theta}_j+\hat{\rho}_{jk}. \quad (45)$$

These parameters are functions of the maximum likelihood estimations of the mean $\hat{\mu}_{j(d)}$ and the variance $\hat{\sigma}^2_{j(d)}$, and the number $\hat{\rho}_{jk}$ of patterns that belong to class k and have been generated from the j-th component. If we have trained a PRBF using EM, then these estimations are already available:

$$\hat{\mu}_{j(d)}=\frac{1}{\hat{\eta}_j}\sum_k\sum_{x\in X_k}p(j|x,k)x_{(d)}, \quad (46)$$

$$\hat{\sigma}^2_{j(d)}=\frac{1}{\hat{\eta}_j}\sum_k\sum_{x\in X_k}p(j|x,k)(x_{(d)}-\hat{\mu}_{j(d)})^2, \quad (47)$$

$$\hat{\eta}_j=\sum_k\hat{\rho}_{jk}, \quad (48)$$

$$\hat{\rho}_{jk}=\sum_{x\in X_k}p(j|x,k). \quad (49)$$

Sampling with the above procedure gives initial estimates for the parameters of the PRBF, which in the following we train using semi-supervised EM. If we repeat this procedure many times we can construct a committee of PRBF classifiers. In our experiments we constructed committees with five members.

In order to measure the disagreement between members we adopt the Kullback–Liebler divergence to the mean, as proposed in [13]. Given an observation x, each member of the committee defines a posterior distribution over class labels $p_m(k|x)$. Given a committee with M_c members, we define the mean class distribution as

$$\bar{p}(k|x)=\frac{1}{M_c}\sum_{m=1}^{M_c}p_m(k|x). \quad (50)$$

The Kullback–Liebler divergence to the mean is the average of the divergence $\mathrm{d}(\cdot\|\cdot)$ of each class distribution to the mean class distribution

$$\frac{1}{M_c}\sum_{m=1}^{M_c}\mathrm{d}(p_m(k|x)\|\bar{p}(k|x)). \quad (51)$$

Using this measure we can quantify the disagreement for each unlabeled observation, using all the available unlabeled observations as a pool. We select to ask the label of the observation that causes the most disagreement, and we estimate again the parameters of the classifier.

In the following we summarize the QBC algorithm for pool-based active learning of the PRBF:

1. Input: The set X of labeled observations, the set X_U of unlabeled observations, and a degenerate network $PRBF_{M=1}$ with one basis function.
2. (a) Try to add one node to the network $PRBF_M$ to form $PRBF_{M+1}$. If the attempt is unsuccessful go to step 3.
   (b) Set $PRBF_M=PRBF_{M+1}$.
   (c) Apply EM until convergence for semi-supervised training of $PRBF_M$.
3. If X_U is empty go to step 4, else
   (a) For each committee member $PRBF_M^{(c)}$, where $c=1,\ldots,M_c$ do:
      (i) Sample the parameters of $PRBF_M^{(c)}$, according to (40)–(42).
      (ii) Apply EM until convergence for semi-supervised training of $PRBF_M^{(c)}$.
   (b) Pick the unlabeled observation $\hat{x}$ that maximizes (51), and ask its label $\hat{y}$.
   (c) Update the sets: $X=X\cup\{(\hat{x},\hat{y})\}$ and $X_U=X_U\backslash\{\hat{x}\}$.
   (d) Go to step 2.
4. Output: Split the nodes of $PRBF_M$ to form the final network $PRBF_M^{\mathrm{split}}$.

5. Experiments

For the experimental evaluation of our method we used four data sets, available from the UCI repository, namely the "segmentation", the "waveform", the "optical digits" and the "pen-based digits" data set. The characteristics of the data sets are summarized in Table 1. All the data sets were standardized, so that all their features exhibit zero mean and unit standard deviation.

In order to estimate the average generalization error we used five-fold cross-validation, and for each fold we computed the percentage of mis-classifications. In all the experiments, we formed the initial set X of labeled patterns selecting uniformly 50 labeled patterns from the training set, and we treated the rest as the pool X_U of unlabeled patterns. We actively selected 300 more, one at each iteration of the algorithm which we proposed in Section 4.2.

For comparison, we also estimated the average generalization error using the QBC method of active learning, which was described in Section 4.3. The average generalization error for both methods, after the addition of each label, is illustrated in Fig. 6. The final generalization error and the standard deviation after the addition of 300 labels are also summarized in Table 2. In all the cases, the decrease of the generalization error is rapid at the first 50 iterations, and the rate of decrease is comparable for the two methods. Although the proposed method exhibits a slightly steeper descent, and eventually results in a smaller generalization error at three cases.

6. Discussion

We have proposed a semi-supervised learning method for the PRBF network that is based on the EM algorithm for maximization of the joint likelihood of both the labeled and unlabeled data. The method uses closed-form update equations for the network parameters. We have also presented an incremental semi-supervised training method that is subsequently employed in a pool-based active learning approach that selects unlabeled points lying near the decision boundary.

Table 1
The number of patterns, features and classes for each data set

	Patterns	Features	Classes
Segmentation	2310	19	7
Waveform	5000	21	3
Optical digits	5620	62	10
Pen-based digits	10 992	16	10

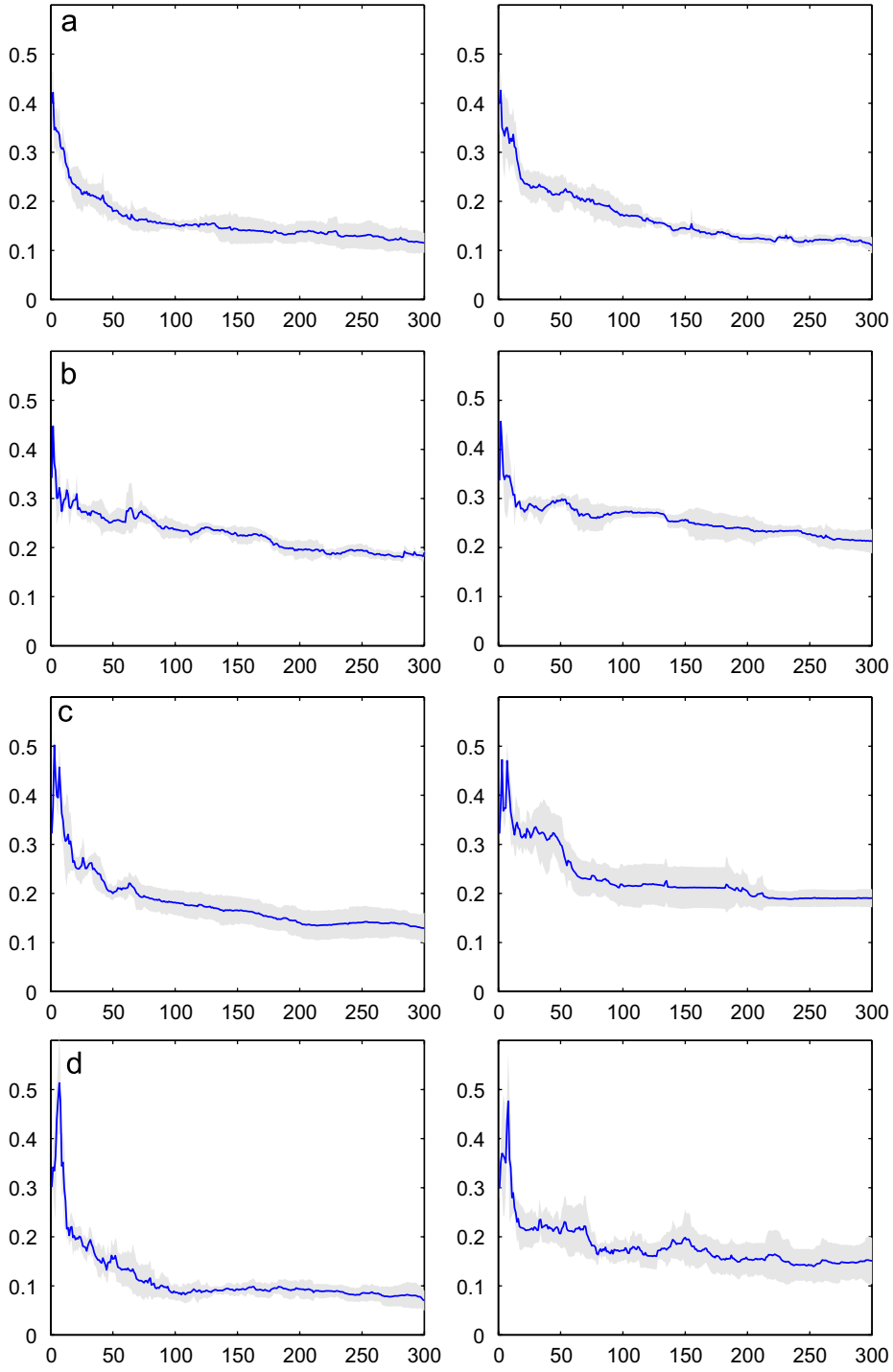

Fig. 6. The average generalization error w.r.t. the number of added labels, for pool-based active learning using the proposed method (left column) and the QBC method (right column). The shaded regions show the one standard deviation area. (a) Segmentation, (b) waveform (c) optical digits, (d) pen-based digits.

Table 2
The average generalization error and the standard deviation (in parenthesis) after the addition of 300 labels

	Active PRBF learning	QBC
Segmentation	0.115 (0.020)	0.111 (0.016)
Waveform	0.190 (0.005)	0.212 (0.024)
Optical digits	0.132 (0.031)	0.190 (0.018)
Pen-based digits	0.070 (0.020)	0.150 (0.046)

The experimental results are encouraging, since, using only a small percentage of the labeled data, the active learning method achieves classification performance comparable to that obtained when all labels are known in advance. Further research work could focus on studying the case where a bunch of unlabeled observations are selected and labeled at each iteration (instead of a single one). Another issue concerns the development and evaluation of other selection criteria for the active acquisition of class information. Also we plan to consider the problem of new

class discovery, as an analogous problem that we would like to tackle. Finally, we aim to apply the method in real world active learning applications, such as for example spam e-mail detection, and perform comparison with active learning methods that employ SVMs.

References

[1] J.L. Bentley, Multidimensional binary search trees used for associative searching, Commun. ACM 18 (9) (1975) 509–517.
[2] C.M. Bishop, Neural Networks for Pattern Recognition, Oxford University Press, Oxford, 1995.
[3] V. Castelli, T. Cover, On the exponential value of labeled samples, Pattern Recognition Lett. 16 (1995) 105–111.
[4] O. Chapelle, B. Scholkopf, A. Zien, Semi-supervised Learning, MIT Press, Cambridge, MA, 2006.
[5] D. Cohn, Z. Ghahramani, M. Jordan, Active learning with statistical models, J. Artif. Intell. Res. 4 (1996) 129–145.
[6] C. Constantinopoulos, A. Likas, An incremental training method for the probabilistic RBF network, IEEE Trans. Neural Networks 17 (4) (2006) 966–974.
[7] I. Dagan, S. Engelson, Committee-based sampling for training probabilistic classifiers, in: Proceedings of the Twelfth International Conference on Machine Learning, Morgan Kaufmann, Los Altos, CA, 1995, pp. 150–157.
[8] A. Dempster, N. Laird, D. Rubin, Maximum likelihood estimation from incomplete data via the EM algorithm, J. R. Statist. Soc. Ser. B 39 (1) (1977) 1–38.
[9] Y. Freund, H.S. Seung, E. Shamir, N. Tishby, Selective sampling using the query by committee algorithm, Mach. Learn. 28 (1997) 133–168.
[10] Z. Ghahramani, M. Jordan, Supervised learning from incomplete data via an EM approach, in: J.D. Cowan, G. Tesauro, J. Alspector (Eds.), Advances in Neural Information Processing Systems, vol. 6, Morgan Kaufmann, Los Altos, CA, 1994.
[11] S. Hoi, R. Lin, M. Lyu, Large-scale text categorization by batch mode active learning, in: Proceedings of the WWW2006, 2002.
[12] A. Likas, N.A. Vlassis, J.J. Verbeek, The global k-means clustering algorithm, Pattern Recognition 36 (2003) 451–461.
[13] A.K. McCallum, K. Nigam, Employing EM in pool-based active learning for text classification, in: J.W. Shavlik (Ed.), Proceedings of the 15th International Conference on Machine Learning, Morgan Kaufmann, Los Altos, CA, 1998.
[14] G. McLachlan, T. Krishnan, The EM Algorithm and Extensions, Wiley, New York, 1997.
[15] G. McLachlan, D. Peel, Finite Mixture Models, Wiley, New York, 2000.
[16] D. Miller, H. Uyar, Combined learning and use for a mixture model equivalent to the RBF classifier, Neural Comput. 10 (1998) 281–293.
[17] H.S. Seung, M. Opper, H. Sompolinsky, Query by committee, in: Proceedings of the Fifth Workshop on Computational Learning Theory, Morgan Kaufmann, Los Altos, CA, 1992, pp. 287–294.
[18] S. Tadjudin, A. Landgrebe, Robust parameter estimation for mixture model, IEEE Trans. Geosci. Remote Sensing 38 (2000) 439–445.
[19] M.K. Titsias, A. Likas, A probabilistic RBF network for classification, in: Proceedings of the International Joint Conference on Neural Networks, vol. 4, IEEE, New York, 2000, pp. 238–243.
[20] M.K. Titsias, A. Likas, Shared kernel models for class conditional density estimation, IEEE Trans. Neural Networks 12 (5) (2001) 987–997.
[21] M.K. Titsias, A. Likas, Mixture of experts classification using a hierarchical mixture model, Neural Comput. 14 (9) (2002) 2221–2244.
[22] M.K. Titsias, A. Likas, Class conditional density estimation using mixtures with constrained component sharing, IEEE Trans. Pattern Anal. Mach. Intell. 25 (7) (2003) 924–928.
[23] S. Tong, D. Koller, Support vector machine active learning with applications to text classification, J. Mach. Learn. Res. 2 (2001) 45–66.
[24] N.A. Vlassis, A. Likas, A greedy-EM algorithm for Gaussian mixture learning, Neural Process. Lett. 15 (2002) 77–87.
[25] X. Zhu, J. Lafferty, Z. Ghahramani, Combining active learning and semi-supervised learning using Gaussian fields and harmonic functions, in: Proceedings of the 20th International Conference on Machine Learning, 2003.

Constantinos Constantinopoulos received the B.Sc. degree in 2000, the M.Sc. degree in 2002 and the Ph.D. degree in 2006 from the Computer Science Department, University of Ioannina, Greece. He is currently a research associate at the same department. His research interests include neural networks, machine learning and Bayesian reasoning.

Aristidis C. Likas received the Diploma degree in electrical engineering in 1990 and the Ph.D. degree in electrical and computer engineering in 1994 both from the National Technical University of Athens, Greece.

Since 1996, he has been with the Department of Computer Science, University of Ioannina, Greece, where he is currently an associate professor. His research interests include neural networks, machine learning, statistical signal processing and bioinformatics.