The One-Hidden Layer Non-parametric Bayesian Kernel Machine

Sotirios P. Chatzis, Dimitrios Korkinof, and Yiannis Demiris

Department of Electrical and Electronic Engineering
Imperial College London
Email: {s.chatzis, d.korkinof10, y.demiris}@imperial.ac.uk

Abstract—**In this paper, we present a nonparametric Bayesian approach towards one-hidden-layer feedforward neural networks. Our approach is based on a random selection of the weights of the synapses between the input and the hidden layer neurons, and a Bayesian marginalization over the weights of the connections between the hidden layer neurons and the output neurons, giving rise to a kernel-based nonparametric Bayesian inference procedure for feedforward neural networks. Compared to existing approaches, our method presents a number of advantages, with the most significant being: (i) it offers a significant improvement in terms of the obtained generalization capabilities; (ii) being a nonparametric Bayesian learning approach, it entails inference instead of fitting to data, thus resolving the overfitting issues of non-Bayesian approaches; and (iii) it yields a full predictive posterior distribution, thus naturally providing a measure of uncertainty on the generated predictions (expressed by means of the variance of the predictive distribution), without the need of applying computationally intensive methods, e.g., bootstrap. We exhibit the merits of our approach by investigating its application to two difficult multimedia content classification applications: semantic characterization of audio scenes based on content, and yearly song classification, as well as a set of benchmark classification and regression tasks.**

Index Terms—**Nonparametric Bayesian inference, kernel machines, neural networks.**

I. Introduction

Feedforward neural networks (FNNs) constitute a significant nonlinear approach for approximating arbitrarily complex unknown functions [1, 2]. However, FNN training algorithms based on direct optimization of the network weights have led to less than satisfactory results [3]; they usually exhibit slow convergence combined with high computational requirements, and they often yield bifurcations and suboptimal estimates of the model parameters (local optima of the optimized objective functions). These issues can be largely attributed to the ill-posed nature of the FNN training problem, since parameter (weight) estimation involves inversion of a nonlinear system estimated from limited and noisy data [4].

A way of ameliorating these issues can be traced to [5, 6], and consists in randomly creating the hidden layer of the treated neural network, and using the desired output signal to train a linear output layer attached to the hidden layer, which computes a linear combination of the neuron outputs from the input-excited hidden layer. The weights of the output layer are typically computed using a least-mean squares method based, e.g., on applying a Moore–Penrose's generalized inverse [7]. As a consequence of this setup, the computational cost of the resulting model turns out to be considerably lower than the costs of other classical learning algorithms, such as gradient-descent methods or global search approaches (genetic algorithms, particle swarm optimization, etc.) [8, 9].

In this paper, we devise a novel nonparametric Bayesian approach towards feedforward neural networks, sharing some common concepts with the aforementioned approaches, especially extreme learning machines (ELMs) [10] (Fig. 1). We begin our analysis considering a random selection of the weights of the synaptic connections between the inputs and the hidden layer neurons of a postulated one-hidden-layer FNN, and the imposition of a suitable prior distribution over the (trainable) weights of the synaptic connections between the hidden layer neurons and the output layer neurons. Additionally, to endow our model with the capability of coping with observable datasets contaminated by noise and outliers, we introduce the fundamental assumption that the target values in the modeled populations comprise the superposition of some noiseless latent function of the outputs of the hidden layer neurons *(hidden layer state)*, that the postulated model can learn, plus an independent white Gaussian noise signal. Based on this prior configuration, we eventually derive the predictive posterior distribution of the network output, by effectively *marginalizing out* the network trainable weights. As we show, this construction eventually gives rise to a form of a nonparametric Bayesian kernel machine, where the employed kernels are defined over the outputs of the hidden layer neurons (hidden layer states). We dub our kernel-based model the *one-hidden-layer nonparametric Bayesian kernel machine (1HNBKM)*.

As we shall discuss in the following sections, the introduced nonparametric Bayesian model yields better generalization performance than the existing approaches, while retaining the stunning computational efficiency of ELMs, and also providing a full predictive distribution (instead of mere point-predictions) which can be of utility in several applications, e.g. active learning. The efficacy of the proposed approach will be evaluated on both synthetic applications, using well-known benchmark datasets, and real-life applications. The remainder of this paper is organized as follows: In Section II, our proposed approach is introduced, the model inference algorithms are derived, and its connections to existing approaches are discussed. In Section

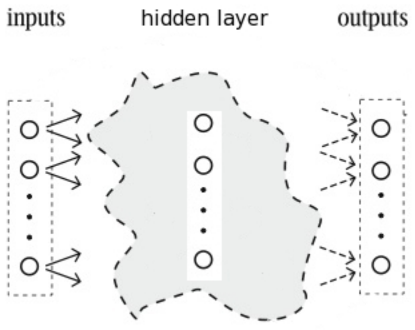

Figure 1. Schematic overview of the ELM approach.

III, the experimental evaluation of our method is conducted. Finally, in the last section of this paper, our conclusions are drawn and our results are discussed.

II. Proposed Approach

A. Model Formulation

Let us consider an FNN with one hidden layer comprising N neurons, and a linear output layer which maps the hidden layer states

$$\boldsymbol{x}(t) = h(\boldsymbol{W}_{in}\boldsymbol{u}(t)) \tag{1}$$

to the actual M-dimensional outputs

$$\boldsymbol{y}(t) = \boldsymbol{W}^{\mathrm{T}}\boldsymbol{x}(t) \tag{2}$$

where $h(\cdot)$ is the neuron activation function, and $\boldsymbol{u}(t)$ the network input. In constructing the network, we choose to randomly select the weights $\boldsymbol{W}_{in}$ of the synapses between the D-dimensional input and the hidden layer neurons. Let $\boldsymbol{y}(t) = [y_j(t)]_{j=1}^{M}$; then, from (2) and assuming that the observations $\boldsymbol{y}(t)$ are contaminated with white Gaussian noise, we have

$$y_j(t) = \boldsymbol{w}_j^{\mathrm{T}}\boldsymbol{x}(t) + \epsilon \tag{3}$$

where

$$\epsilon \sim \mathcal{N}(0, \sigma^2) \tag{4}$$

σ^2 is the noise variance, and $\boldsymbol{w}_j$ is the jth column of $\boldsymbol{W}$. Let us now impose a spherical Gaussian prior over the weights vectors $\boldsymbol{w}_j$, such that

$$\boldsymbol{w}_j \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}) \tag{5}$$

Under this setting, we obtain the following results for the mean and the covariance of the network component responses $y_j(t)$

$$\mathbb{E}\left[y_j(t)\right] = \mathbb{E}\left[\boldsymbol{w}_j^{\mathrm{T}}\right]\boldsymbol{x}(t) = 0 \tag{6}$$

and

$$\mathbb{E}\left[y_j(t_1)y_j(t_2)\right] = \boldsymbol{x}(t_1)^{\mathrm{T}}\mathbb{E}\left[\boldsymbol{w}_j\boldsymbol{w}_j^{\mathrm{T}}\right]\boldsymbol{x}(t_2) = \boldsymbol{x}(t_1)^{\mathrm{T}}\boldsymbol{x}(t_2) \tag{7}$$

Then, it turns out that $y_j(t_1)$ and $y_j(t_2)$ are jointly Gaussian with zero mean and covariance given by the dot-product $\boldsymbol{x}(t_1)^{\mathrm{T}}\boldsymbol{x}(t_2)$, for any $j \in \{1, \dots, M\}$, and $\forall t_1, t_2$. In other words, under our Bayesian approach, and by marginalizing out the weights $\boldsymbol{w}_j$, we eventually obtain:

$$[y_j(t)]_{t=t_1}^{t_T} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{K}_r(\boldsymbol{X}, \boldsymbol{X})) \tag{8}$$

where $\boldsymbol{K}_r$ is given by

$$\boldsymbol{K}_r(\boldsymbol{X}, \boldsymbol{X}) \triangleq \begin{bmatrix} k_r(\boldsymbol{x}(t_1), \boldsymbol{x}(t_1)) & \dots & k_r(\boldsymbol{x}(t_1), \boldsymbol{x}(t_T)) \\ \vdots & \vdots & \vdots \\ k_r(\boldsymbol{x}(t_T), \boldsymbol{x}(t_1)) & \dots & k_r(\boldsymbol{x}(t_T), \boldsymbol{x}(t_T)) \end{bmatrix} \tag{9}$$

$$\boldsymbol{X} \triangleq [\boldsymbol{x}(t)^{\mathrm{T}}]_{t=1}^{T} \tag{10}$$

with the $\boldsymbol{x}(t)$ computed using (1), and

$$k_r(\boldsymbol{x}(t_1), \boldsymbol{x}(t_2)) \triangleq \boldsymbol{x}(t_1)^{\mathrm{T}}\boldsymbol{x}(t_2) \tag{11}$$

Apparently, the obtained expression of the joint distribution (8) essentially constitutes a nonparametric kernel-based prior over the network outputs, since the covariance matrix $\boldsymbol{K}_r(\boldsymbol{X}, \boldsymbol{X})$ is the gram matrix for a simple linear (dot-product) kernel.

Generalizing the above results to allow for the utilization of kernels of any other appropriate form (i.e., satisfying Mercer's conditions [11]), we introduce the *one-hidden-layer nonparametric Bayesian kernel machine (1HNBKM).* For example, in case a Gaussian radial basis function (RBF) kernel is considered, the definition of the 1HNBKM model yields a prior distribution of the form (8) with its kernel function given by

$$k_r(\boldsymbol{x}(t_1), \boldsymbol{x}(t_2)) \triangleq \exp\left[-\left|\left|\boldsymbol{x}(t_1) - \boldsymbol{x}(t_2)\right|\right|^2 / 2\lambda^2\right] \tag{12}$$

B. Model Inference

Let us consider a set of example data points $\mathcal{D} = \{\boldsymbol{x}(t), \boldsymbol{y}(t)\}_{t=1}^{T}$, where the $\boldsymbol{x}(t)$ are generated according to (1), using the input signal values $\{\boldsymbol{u}(t)\}_{t=1}^{T}$. Let $\boldsymbol{y}_j \triangleq [y_j(t)]_{t=1}^{T}$, and y_{j*} be the jth network output for hidden layer state value $\boldsymbol{x}_*$ (generated given an input $\boldsymbol{u}_*$). Then, from (8) and (3) we have

$$\begin{bmatrix} \boldsymbol{y}_j \\ y_{j*} \end{bmatrix} \sim \mathcal{N}\left(\boldsymbol{0}, \begin{bmatrix} \boldsymbol{K}_r(\boldsymbol{X}, \boldsymbol{X}) + \sigma^2\boldsymbol{I}_T & \boldsymbol{k}_r(\boldsymbol{x}_*) \\ \boldsymbol{k}_r(\boldsymbol{x}_*)^{\mathrm{T}} & k_r(\boldsymbol{x}_*, \boldsymbol{x}_*) \end{bmatrix}\right) \tag{13}$$

where

$$\boldsymbol{k}_r(\boldsymbol{x}_*) \triangleq [k_r(\boldsymbol{x}(t_1), \boldsymbol{x}_*), \dots, k_r(\boldsymbol{x}(t_T), \boldsymbol{x}_*)]^{\mathrm{T}} \tag{14}$$

From this result, the following expression for the predictive density of the model is derived

$$p(y_{j*}|\boldsymbol{x}_*, \mathcal{D}) = \mathcal{N}(y_{j*}|\mu_{j*}, \sigma_*^2) \tag{15}$$

where

$$\mu_{j*} = \boldsymbol{k}_r(\boldsymbol{x}_*)^{\mathrm{T}}\left(\boldsymbol{K}_r(\boldsymbol{X}, \boldsymbol{X}) + \sigma^2\boldsymbol{I}_T\right)^{-1}\boldsymbol{y}_j \tag{16}$$

and

$$\begin{aligned} \sigma_*^2 =& k_r(\boldsymbol{x}_*, \boldsymbol{x}_*) \\ &- \boldsymbol{k}_r(\boldsymbol{x}_*)^{\mathrm{T}}\left(\boldsymbol{K}_r(\boldsymbol{X}, \boldsymbol{X}) + \sigma^2\boldsymbol{I}_T\right)^{-1}\boldsymbol{k}_r(\boldsymbol{x}_*) \end{aligned} \tag{17}$$

Similar, the model log evidence (log marginal likelihood) will be given by

$$\log p(\boldsymbol{y}|\boldsymbol{X}) = \sum_{j=1}^{M} \frac{1}{2}\Big\{ - T\log 2\pi - \log\left|\boldsymbol{K}_r(\boldsymbol{X},\boldsymbol{X}) + \sigma^2 \boldsymbol{I}_T\right| - \boldsymbol{y}_j^{\mathrm{T}}\left(\boldsymbol{K}_r(\boldsymbol{X},\boldsymbol{X}) + \sigma^2\boldsymbol{I}_T\right)^{-1}\boldsymbol{y}_j\Big\} \tag{18}$$

Estimation of the model hyperparameters, that is of the noise hyperparameter σ^2 as well as of the hyperparameters of the employed kernel functions $k(\cdot,\cdot)$, is conducted by means of type-II maximum likelihood, i.e. by optimization of the model evidence (18). To perform this optimization task, we here employ the scaled conjugate gradient (SCG) descent algorithm [12].

C. Relations to existing models

Recently, [13] proposed a Bayesian methodology dubbed the Bayesian ELM (BELM). Their method relies on the utilization of Bayesian linear regression as a means of obtaining a posterior distribution over the columns $\boldsymbol{w}_j$ of the trainable weights matrices $\boldsymbol{W}$. Then, using the expression of the obtained posterior over the $\boldsymbol{w}_j$ and the expression (3), the predictive distribution of the model can be shown to yield a Gaussian with mean

$$\mu_{j*} = \boldsymbol{x}_*^{\mathrm{T}}\boldsymbol{A}^{-1}\boldsymbol{X}^{\mathrm{T}}\boldsymbol{y}_j \tag{19}$$

and variance

$$\sigma_*^2 = \sigma^2\boldsymbol{x}_*{}^{\mathrm{T}}\boldsymbol{A}^{-1}\boldsymbol{x}_* \tag{20}$$

where

$$\boldsymbol{A} = \boldsymbol{X}^{\mathrm{T}}\boldsymbol{X} + \sigma^2\boldsymbol{I} \tag{21}$$

We can show that the expressions of the predictive density of our model reduce to (19)-(20) in case we employ a simple linear (dot product) kernel of the form (11). Indeed, if we consider a kernel of the form (11), the resulting expression of $\boldsymbol{K}_r(\boldsymbol{X},\boldsymbol{X})$ will turn out to be essentially low-rank by construction: Expanding (16) and (17) in terms of $\boldsymbol{K}_r(\boldsymbol{X},\boldsymbol{X})$ and $\boldsymbol{x}(t)$, and using the matrix inversion lemma, the expressions of the 1HNBKM predictive mean and variance can be restated in the forms (19) and (20), respectively. Therefore, our approach comprises a generalization of the BELM network [13], incorporates it as a special case, and reduces to it when a linear kernel function is considered.

It is also interesting to compare the computational complexity of the 1HNBKM predictive algorithm with the complexities of the ELM and BELM algorithms. Indeed, predictions under both the BELM and ELM algorithms reduce to multiplication of a matrix $\boldsymbol{H}$ with the point (hidden layer state value) $\boldsymbol{x}_*$ where prediction is to be performed ($\boldsymbol{H} = \boldsymbol{A}^{-1}\boldsymbol{X}^{\mathrm{T}}\boldsymbol{y}_j$ in the case of the BELM). However, this is exactly the case also for the 1HNBKM algorithm. Indeed, from Eq. (16) we observe that prediction under our approach consists in the multiplication of the previously computed (and stored) matrix $\boldsymbol{\Pi} = \left(\boldsymbol{K}_r(\boldsymbol{X},\boldsymbol{X}) + \sigma^2\boldsymbol{I}_T\right)^{-1}\boldsymbol{y}_j$ with a costless transformation of the point where prediction is to be carried out, $\boldsymbol{k}_r(\boldsymbol{x}_*)$. Hence, prediction under the proposed 1HNBKM algorithm imposes computational costs of the same order of magnitude as the ELM and BELM approaches.

III. Applications

In the following section, we provide a thorough experimental evaluation of the 1HNBKM model, considering both classical benchmark tasks and real-world applications. In our experimental evaluations, we consider analog hidden neurons, with simple sigmoid transfer functions

$$h(z) = \frac{1}{1+\exp(-z)} \tag{22}$$

To demonstrate the advantages of our approach, we also evaluate ELM networks, BELMs implemented as described in [13], back-propagation (BP) networks trained using the Levenberg–Marquardt algorithm [11], run with the standard settings of its implementation in the Neural Networks Toolbox of MATLAB, SVM models with ϵ-insensitive loss functions [14], as implemented in the LIBSVM MATLAB interface [15], and Gaussian processes (GPs) with RBF kernels, employing the Laplace approximation with softmax likelihood functions to effect classification tasks [16]. The number of neurons of the evaluated neural networks is selected so as to maximize their performance in generalization accuracy terms. The proposed 1HNBKM approach, as well as the SVMs, are evaluated considering Gaussian RBF kernels, similar to the evaluated GP models. Selection of the hyperparameter values of the RBF kernels employed by the evaluated 1HNBKM and GP models is conducted by means of type-II maximum-likelihood, while in the case of the SVM we use cross-validation. The matrices $\boldsymbol{W}_{in}$ in (1) are drawn from the standard Gaussian distribution. Application of the evaluated neural network algorithms to classification tasks was performed by considering C output neurons, where C is the number of classes in the task; each output takes values in the interval $[0,1]$, with the higher the output value, the higher the probability of the corresponding class.

Evaluation and comparison of the performances of the considered algorithms is conducted on the grounds of their classification error rates, in cases of classification tasks, and obtained root mean square errors, in cases of regression tasks. Comparison of the computational efficiency of the considered algorithms is not straightforward, due to the following reasons:

1) ELM-inspired algorithms (including our method) do not entail training in the sense of the iterative procedures employed in the case of BP neural networks. Hence, comparison of the times or repetitions needed for training algorithm convergence is not applicable.
2) The used SVM libraries are based on an underlying C code, hence being highly computationally efficient. In contrast, our implementations of the ELM-inspired algorithms were purely in MATLAB. Thus, comparison of execution times would definitely be biased against the ELM-inspired algorithms.

For these reasons, we avoid providing such comparisons. However, regarding comparison of the ELM algorithm with our 1HNBKM, we would like to underline that prediction under both algorithms imposes computational costs of the same order of magnitude, as already highlighted in Section II.C.

A. Semantic Characterization of Audio Scenes Based on Content

The significance of audio content in the semantic characterization of multimedia has recently motivated development of various techniques for content-based scene classification in audio signals. Audio streams, in general, contain a lot of artifacts, noise, and outliers, that cannot be easily eliminated by a potential model training sample. Furthermore, to allow for the effective semantic classification of audio data, usually a large number of audio features has to be extracted, thus increasing significantly the dimension of the formulated feature space over which classification or categorization algorithms are carried out. Therefore, application of pattern recognition methodologies offering high computational efficiency and, simultaneously, being highly effective in terms of their pattern recognition effectiveness, is expected to be of significant value to real-life audio scene categorization systems.

The dataset used to carry out our tests consists of 487, 20 min. audio samples extracted from several movie genres. Each sample has been divided into semantically coherent audio segments (scenes), and a groundtruth semantic classification has been assigned to each one of these scenes by human experts; we consider 8 semantic classes of audio scene content: *Music*, *Speech*, *Others1* (low environmental sounds: wind, rain etc), *Others2* (sounds with abrupt changes, like a door closing), *Others3* (louder sounds, mainly machines and cars), *Gunshots*, *Fights*, and *Screams*. This selection of the considered semantic classes aims to provide highly detailed audio class descriptors, while also focusing on defining audio classes of violent content, so that the system can be also used as a detector of violence in audio information, with applications, e.g. in systems developed to protect sensitive groups of the population (e.g. children) from violent multimedia content.

Each audio scene is represented by a 12-dimensional feature vector, computed by deriving from each segment

- the spectral rolloff median (SRM);
- the zero crossing rate (ZCR), measuring the number of time-domain zero crossings, divided by the frame's length;
- two spectrogram features, the standard deviation and the maximum value of the means obtained over the spectrogram windows;
- two chroma features [17], expressing the deviation between the obtained chroma coefficients of each segment, and the deviation between successive frames for each chroma element;
- the spectral rolloff [18];
- the energy entropy;
- the pitch;
- and, four Mel-frequency Cepstral Coefficient (MFCC)-related features, namely the maximum value and the maximum to mean ratio of the first MFCC, the standard deviation of the second, and the median value of the third.

The results of the experimental evaluation are presented in Table I, for 2, 4 and 8 classes of audio clips. In the latter case, the full dataset is utilized, while in the 2- and 4-class experiments we have utilized the Speech-Music and Speech-Music-Gunshots-Fights & Screams classes, respectively. In the provided results, we have omitted the performance of the BELM method, as it turned out to be identical to that of the ELM method in all cases.

In all cases, we utilized all the dataset datapoints, a total of 490 per class. We segmented the available data into randomly chosen training and testing sets constituting 75% and 25% of the total points respectively. Each algorithm was evaluated for 25 sets of 4 runs, a total of 100 executions, and the means and standard deviations of the obtained error rates were computed. Each one of the 25 sets corresponds to a different random segmentation of the dataset and the 4 runs correspond to all combinations of training and test sets.

As we can observe, all the evaluated methods yield satisfactory results, especially in the case of two learned classes, where the success rate approaches or exceeds 95%. We can also see that the proposed algorithm consistently outperforms all rival methods, especially in the most demanding experiment, namely the one with 8 classes of audio to distinguish; in that case, the proposed method yields less than half the error rate of the SVM, and a significant improvement over the BP algorithm. This is a very significant finding, as it shows that our method retains its robustness when trying to discriminate between a higher number of less easily discernible classes, contrary to its comparators the performance of which deteriorates significantly, as we noticed especially in the case of the SVM.

B. Yearly Song Classification Using Audio Features

Automatic generation of missing metadata from media content is recently receiving increasing attention. Media files are usually enriched with metadata information, which is utilized for the purpose of search or classification. There is, however, increasing demand to deal with missing or false metadata, a task that usually cannot be performed manually due to the impressive abundance of media files. To this end, interesting approaches include, but are not limited to, content-based retrieval, genre prediction, artist recognition, musical piece recognition, and document topic detection.

A very interesting application that has remained unaddressed through the years is the automatic prediction of the track's release year. To the best of our knowledge, an approach towards this direction has yet to emerge and the reason becomes obvious after a close inspection of the problem. Indeed, we can observe that the problem in question entails surprisingly challenging complexity issues, deriving from the great diversity of style and genre of the songs released each year. In fact, it almost seems that the diversity is too severe for

Table I
SEMANTIC CHARACTERIZATION OF AUDIO SCENES BASED ON CONTENT: OBTAINED CLASSIFICATION ERROR RATES (MEANS AND STANDARD DEVIATIONS).

#Audio scenes classes	ELM (%)	GP (%)	1HNBKM (%)	SVM (%)	BP (%)	#Neurons
2	4.589 (1.26)	4.777 (1.25)	3.329 (0.60)	5.159 (1.27)	5.150 (1.61)	50
4	13.702 (1.94)	11.920 (0.45)	11.281 (0.39)	15.307 (1.60)	13.186 (1.62)	50
8	30.429 (1.23)	28.320 (1.12)	18.476 (0.89)	39.252 (1.35)	25.527 (1.56)	100
$\#Datapoints/class$	490					

the extraction of any pattern or credible prediction just from a track's audio features. In this work, we are able to demonstrate that the application of the 1HNBKM can provide an algorithm able to generate reasonable predictions.

We have utilized a subset of the "Million song dataset" [19], which comprises 515345 tracks with available release year information. The tracks are mostly western, commercial tracks ranging from 1922 to 2011, with a peak in the year 2000 and onwards. We have effectively made use of those released between the years 1980 to 2010. Apart form the year, the dataset provides 90 additional representative features; of these 90 attributes, 12 are timbre average and 78 are timbre covariance, all extracted from the timbre features using the Echo Nest API [20].

In the experimental evaluation of our algorithm, we have respected the following train/test split: The first 463715 tracks are used for training and the remaining 51630 for testing. This is necessary in order to avoid the "producer effect," by making sure no given artist ends up with their songs being included in both the training and test set. We have indeed observed better recognition performance for all evaluated methods in the opposite case.

In our experiments, our goal was to detect the decade a song was written. For this purpose, we randomly selected 1000 songs released in years 1980-1982, 1000 songs released in 1995-1997, and 1000 songs released in 2008-2010 from the above mentioned datasets, and created the training and test datasets for three classes, respectively: *Decade of 1980*, *decade of 1990*, and *decade of 2000.* We chose to limit our datasets to only 1000 samples from each decade as this constitutes the ceiling (in terms of computational tractability) for the back-propagation algorithm to which our method is compared.

In Table II, we provide the classification rates (means and standard deviations) obtained from the evaluated algorithms for 10 different random selections of the data. As we observe, the proposed approach outperforms all its comparators, in terms of the obtained classification error rate. Note also that our method is among the most computationally efficient of the evaluated ones, since it does not entail the cumbersome procedures that back-propagation relies on. Interestingly enough, the back propagation model required 1 hour and 3 min to fully train and test on a desktop computer with a quad-core Intel $i7$ $3.4GHz$ CPU and $16GB$ RAM, and consumed virtually all the memory of the machine, while all the other methods did not require more than a minute. Note also that the BELM once again did not yield any improvement over the ELM method; in fact the obtained results are identical to the ELM and this is the reason why we have omitted them from Table II.

C. Regression Using Benchmark Datasets: California Housing

Here, we evaluate our method in a regression task and compare its performance to its comparators. For this purpose, we consider a benchmark regression dataset, namely *California Housing.* The database comprises 20,460 8-dimensional samples, of which we use 8000 samples for training and the rest 12,460 for testing. The obtained results (averages and standard deviations) over fifty trials of simulations (from different random initializations) for the evaluated methods are illustrated in Table III. As we observe, our method outperforms all the evaluated alternatives in terms of the average obtained root mean square error (RMSE), with BP being the worst performing method.

D. Diabetes Diagnosis

In this experiment, we consider application of our method in a medical diagnosis task (diabetes diagnosis), using the *Pima Indians Diabetes Database* produced in the Applied Physics Laboratory, Johns Hopkins University, 1988. The diagnostic, binary-valued variable investigated is whether the patient shows signs of diabetes according to World Health Organization criteria. The database consists of 768 women over the age of 21 resident in Phoenix, Arizona. All examples belong to either positive or negative class. All the input values are within [0,1]. In our experiments, we randomly choose the 75% of the available samples for training and the rest 25% for testing. We conduct fifty trials (for different random selections of the data) for all the evaluated algorithms, and provide the average performance (and its standard deviation) in Table IV. As we observe, our algorithm yields better generalization performance compared to the competition. It is also noteworthy that BP yields the lowest generalization performance in this experiment, and that the BELM does not yield any performance improvement over ELM.

E. Classification Tasks with Application of Active Learning

Finally, in this experiment we consider application of active learning in the context of the 1HNBKM. Active learning is based on the notion that the performance of the learners might be considerably improved if the learners could actively participate in the learning process. That is, contrary to conventional supervised learning, where the learner "passively" receives the labeled data and generates a learned model, we would like to introduce a framework for identifying a subset of a pool

Table II
YEARLY SONG CLASSIFICATION: OBTAINED CLASSIFICATION ERROR RATES (MEANS AND STANDARD DEVIATIONS).

Method	ELM (%)	GP (%)	1HNBKM (%)	SVM (%)	BP (%)
Performance	41.92 (1.38)	40.81 (1.09)	39.06 (1.03)	43.44 (1.63)	51.91 (1.28)
#Neurons	250	-	250	-	250
$\#Datapoints/class$	1000				

Table III
PERFORMANCE COMPARISON IN REGRESSION APPLICATION (CALIFORNIA HOUSING): OBTAINED RMSES (MEANS AND STANDARD DEVIATIONS) OF THE EVALUATED ALGORITHMS.

Algorithms	Mean	Std.	#Neurons
1HNBKM	0.0808	0.0014	10
ELM	0.1267	0.0033	80
BELM	0.1101	0.0021	30
BP	0.1285	0.0026	10
SVM	0.1180	0.0011	-
GP	0.0104	0.0008	-

Table IV
PERFORMANCE COMPARISON IN REAL MEDICAL DIAGNOSIS APPLICATION (DIABETES DIAGNOSIS): OBTAINED SUCCESS RATES (MEANS AND STANDARD DEVIATIONS) OF THE EVALUATED ALGORITHMS.

Algorithms	Training	Set	Test	Set	#Neurons
	Mean (%)	Std. (%)	Mean (%)	Std (%)	
1HNBKM	78.23	1.04	79.12	1.52	20
ELM	78.68	1.18	77.57	2.85	20
BELM	78.68	1.18	77.57	2.85	20
BP	86.63	1.7	74.73	3.2	20
SVM	78.76	0.91	77.31	2.35	-
GP	78.51	1.05	78.27	1.65	-

of unlabeled examples that would be most informative if the associated labels were available and incorporate them in the learning procedure. Hence, an active learning methodology comprises two basic procedures: first, selection of the most informative samples from a pool of unlabeled data; and, second, labeling of these samples and introduction into the inference procedure of the 1HNBKM.

With non-probabilistic classification schemes, a popular heuristic for establishing the confidence of estimates and identifying points for active learning is to simply use the distance from the classification boundary (margin). This approach could be also used with 1HNBKM-based classification models, by inspecting the magnitude of the predictive means $\mu_{j*} = \mathbb{E}[y_{j*}|\boldsymbol{x}_*, \mathcal{D}]$, and choosing the data point $\bar{\boldsymbol{x}}_*$ such that

$$\bar{\boldsymbol{x}}_* = \underset{\boldsymbol{x}_*}{\operatorname{argmin}}\{\underset{\mathrm{j}}{\max}\mathbb{E}[y_{j*}|\boldsymbol{x}_*, \mathcal{D}]\} \tag{23}$$

However, 1HNBKM-based classification provides us with both the predictive mean as well as the predictive variance for the unknown label of the $\boldsymbol{x}_*$. We therefore propose an approach which considers both the predictive mean as well as the predictive variance. Specifically, we select the next point according to a criterion which chooses the unlabeled point where the classification is the most uncertain:

$$\bar{\boldsymbol{x}}_* = \underset{\boldsymbol{x}_*}{\operatorname{argmin}}\frac{\mathbb{E}[y_{\lambda_* *}|\boldsymbol{x}_*, \mathcal{D}]}{\mathbb{V}[y_{\lambda_* *}|\boldsymbol{x}_*, \mathcal{D}]} = \underset{\boldsymbol{x}_*}{\operatorname{argmin}}\frac{\mu_{\lambda_* *}}{\sigma_*^2} \tag{24}$$

where

$$\lambda_* = \underset{\mathrm{j}}{\operatorname{argmax}}\mathbb{E}[y_{j*}|\boldsymbol{x}_*, \mathcal{D}] \tag{25}$$

To assess the performance of our approach, we repeat the experiments of Section III.B (yearly song classification based on audio features) as follows: We begin with the trained models of Section III.B, and add an extra N unlabeled samples, where $N \in [1, 100]$, to perform active learning with. Active learning is conducted under both the criteria (23) and (24). Figure 2 illustrates how the obtained classification rates of the 1HNBKM vary with the number of unlabeled samples under both approaches. As we observe, utilization of the predictive uncertainty information under criterion (24) yields a clear competitive benefit for the 1HNBKM algorithm over criterion (23), which ignores this information. Hence, obtaining a *full predictive density* instead of mere *point predictions* turns out to be of significance for the active learning algorithm.

IV. CONCLUSIONS

In this paper, we presented a nonparametric Bayesian approach towards one-hidden-layer feedforward neural networks. Our approach is based on a random selection of the weights of the synapses between the input and the hidden layer neurons, and a Bayesian marginalization over the weights of the connections between the hidden layer neurons and the output neurons, giving rise to a kernel-based nonparametric Bayesian inference procedure for feedforward neural networks. Being a nonparametric Bayesian learning approach, our method entails inference instead of fitting to data, thus resolving the overfitting issues of non-Bayesian approaches. It also yields a full predictive posterior distribution, thus naturally providing a

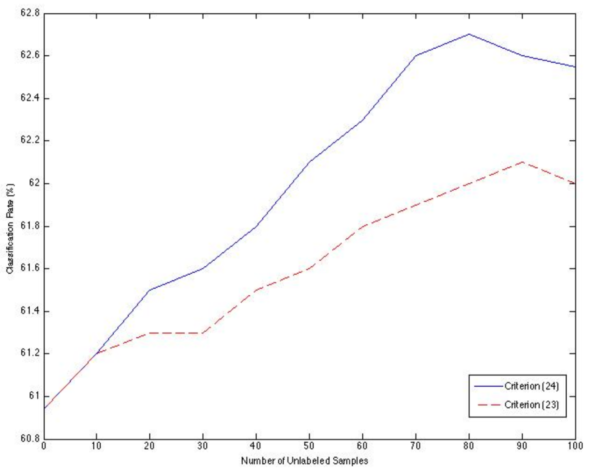

Figure 2. Application of active learning to 1HNBKM-based classification.

measure of uncertainty on the generated predictions (expressed by means of the variance of the predictive distribution), without the need of applying computationally intensive methods, e.g., bootstrap.

We employed our novel method to two difficult problems of multimedia content classification: semantic characterization of audio scenes based on content, and yearly song classification, as well as a set of benchmark problems. As we showed, our approach offers a considerable performance improvement over existing methodologies combined with exceptional computational efficiency, and manages to outperform all its comparators in all the considered experimental cases.

Few open issues of the 1HNBKM that we aim to address in the near future include examining the utility of different kinds of kernels apart from the Gaussian RBF kernel used in our experiments. Demo source codes written in MATLAB allowing for replication of the here presented results shall be made available through the website of the authors: http://www.iis.ee.ic.ac.uk/ sotirios.

Acknowledgment

This work has been funded by the EU FP7 ALIZ-E project (grant 248116).

References

[1] H. T. Siegelmann and E. D. Sontag, “Turing computability with neural nets,” *Applied Mathematics Letters*, vol. 4, pp. 77–80, 1991.

[2] G. Deco and B. Schurmann, “Neural learning of chaotic dynamics,” *Neural Process Lett.*, vol. 2, no. 2, pp. 23–26, 1995.

[3] F. Kianifardand and W. Swallow, “A review of the development and application of recursive residuals in linear models,” *J. Amer. Statist. Assoc.*, vol. 91, no. 443, pp. 391–400, 1996.

[4] S. Haykin and J. Principe, “Making sense of a complex world,” *IEEE Signal Process. Mag.*, vol. 15, no. 3, pp. 66–81, 1998.

[5] G.-B. Huang and C.-K. Siew, “Extreme learning machine with randomly assigned RBF kernels,” *Int. J. Inf. Technol.*, vol. 11, no. 1, pp. 16–24, 2005.

[6] H. Jaeger, “The “echo state” approach to analysing and training recurrent neural networks,” German National Research Center for Information Technology, Bremen, Tech. Rep. 148, 2001.

[7] C. R. Rao and S. K. Mitra, *Generalized Inverse of Matrices and Its Applications*. New York: Wiley, 1972.

[8] J. Kim, H. S. Shin, K. Shin, and M. Lee, “Robust algorithm for arrhythmia classification in ECG using extreme learning machine,” *BioMed. Eng. Online*, vol. 8, no. 31, pp. 1–12, 2009.

[9] R. Zhang, G.-B. Huang, N. Sundararajan, and P. Saratchandran, “Multi-category classification using an extreme learning machine for microarray gene expression cancer diagnosis,” *IEEE/ACM Trans. Comput. Biol. Bioinf.*, vol. 4, no. 3, pp. 485–495, 2007.

[10] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, “Extreme learning machine: Theory and applications,” *Neurocomputing*, vol. 70, no. 1-3, pp. 489–501, 2006.

[11] C. M. Bishop, *Pattern Recognition and Machine Learning*. New York: Springer, 2006.

[12] M. F. Moller, “A scaled conjugate gradient algorithm for fast supervised learning,” *Neural Networks*, vol. 6, pp. 525–533, 1993.

[13] E. Soria-Olivas, J. Gómez-Sanchis, J. D. Martín, J. Vila-Francés, M. Martínez, J. R. Magdalena, and A. J. Serrano, “BELM: Bayesian extreme learning machine,” *IEEE Trans. Neural Netw.*, vol. 22, no. 3, pp. 505–509, 2011.

[14] Z. Shi and M. Han, “Support vector echo-state machine for chaotic time-series prediction,” *IEEE Transactions on Neural Networks*, vol. 18, no. 2, pp. 359–372, 2007.

[15] C.-C. Chang and C.-J. Lin, *LIBSVM: a library for support vector machines*, 2001, software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm.

[16] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*. MIT Press, 2006.

[17] M. A. Bartsch and G. H. Wakefield, “Audio thumbnailing of popular music using chroma-based representations,” *IEEE Trans Multimedia*, vol. 7, no. 1, pp. 96–104, 2005.

[18] T. Giannakopoulos, D. Kosmopoulos, A. Aristidou, and S. Theodoridis, “Violence content classification using audio features,” in *Advances in Artificial Intelligence*, 2006, pp. 502–507.

[19] T. Bertin-Mahieux, D. P. Ellis, B. Whitman, and P. Lamere, “The million song dataset,” in *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR 2011)*, 2011, (submitted).

[20] Echo-Nest. The Echo Nest API. [Online]. Available: http://developer.echonest.com/