Deformed Kernel Based Extreme Learning Machine

Zhang Chen
School of Computer Science and Technology,China University of Mining and Technology, XuZhou,221116,China
Email: zc@cumt.edu.cn

Xia Shi Xiong and Liu Bing
School of Computer Science and Technology,China University of Mining and Technology, XuZhou,221116,China
Email: xiasx@cumt.edu.cn, liubing@cumt.edu.cn

***Abstract*—The extreme learning machine (ELM) is a newly emerging supervised learning method. In order to use the information provided by unlabeled samples and improve the performance of the ELM, we deformed the kernel in the ELM by modeling the marginal distribution with the graph Laplacian, which is built with both labeled and unlabeled samples. We further approximated the deformed kernel by means of random feature mapping. The experimental results showed that the proposed semi-supervised extreme learning machine tends to achieve outstanding generalization performance at a relatively faster learning speed than traditional semi-supervised learning algorithms.**

***Index Terms*—extreme learning machine (ELM); random feature mapping; semi-supervised learning; Reproducing Kernel Hilbert Spaces (RKHS).**

I. INTRODUCTION

Lately, extreme learning machine ELM has been attracting a lot of attention from an increasing number of researchers [1]-[5]. It was originally developed for the single-hidden layer feedforward neural networks (SLFN) [6]-[8], which was extended to the "generalized" SLFNs, i.e., may not be neuron alike [9, 10]. ELM has three main learning features: (1) ELM was originally proposed to apply random computational nodes in the hidden layer. Thus, the hidden layer of the ELM does not need be tuned. (2) ELM incorporates the smallest training error and the norm of output weights into the objective function. Hence, it controls the complexity of decision functions by means of regularization. (3) Unlike LS-SVM and SVM that only provide one type of computational need, ELM provides a unified solution to different practical applications (e.g., regression, binary, and multiclass classifications).

ELM is a supervised learning method. In many applications, however, there are little labeled data and a large amount of unlabeled data available. Semi-Supervised Learning (SSL) methods are proposed to solve this problem. ELM can be naturally extended to the unsupervised scenario, where the "cluster" and "manifold" assumptions are used to learn input-output mapping functions. The "cluster" assumption refers to that points in a data cluster have similar labels. The "manifold" assumption corresponds to high-dimensional data distributed on a low-dimensional manifold and the samples in each local region have similar labels. There are many approaches based on the "cluster" assumption, which uses techniques such as local combinatorial searches[12], branch-and-bound algorithms[13,14], gradient descent[15], semi-definite programming [16-19], continuation techniques[20], non-differentiable methods[21], concave-convex procedures[22,23], and deterministic annealing[24]. However, the time complexity of these methods scales at least quadratically with the dataset size, which makes them inapplicable to large-scale datasets. In [25], a cutting plane semi-supervised support vector machine algorithm (CutS3VM) was proposed to reduce the number of iterations, but it still takes time O(sn) to converge with guaranteed accuracy in the linear case, where n is the total number of samples in the dataset and s is the average number of non-zero features. Sindhwani et al.[26] proposed two kinds of large-scale semi-supervised linear SVMs: the transductive modified finite newton linear L_2-SVM (L_2-TSVM-MFN) and the deterministic annealing L_2-SVM-MFN method (DA L_2-SVM-MFN). L_2-TSVM-MFN is converged after having been switched many times and DA L_2-SVM-MFN needs a number of iterations to compute the corresponding parameters of unlabeled data. Besides the "cluster" assumption, many regularization frameworks based on the "manifold" assumption have been designed by adding a manifold regularization term. In [27], Belkin et al. proposed a general Manifold Regularization (MR) framework for a full range of learning problems from unsupervised and semi-supervised, to supervised. The MR framework adds an additional penalty term to the traditional regularization, from which the Laplacian Regularization Least Square Classification (LapRLSC) and the Laplacian SVM (LapSVM) methods are derived and have been shown to be efficient in semi-supervised learning problems. Additionally, the Discriminatively Regularization Least Square Classification (DRLSC) method and the Sparse Regularized Least Square Classification (S-RLSC) algorithm[29] were proposed, which improves the MR framework further. Although these frameworks can handle semi-supervised learning problems and the analytic solutions can also be derived, they still involve

doi:10.4304/jcp.8.6.1602-1609

expensive computation when training large-scale data sets.

To improve the performance of the ELM, it is essential to use the information provided by both labeled and unlabeled samples. We construct the deformed kernel for the ELM, which is adapted to the geometry of the underlying distribution. Based on the deformed kernel, we propose a deformed kernel-based extreme learning machine (DKELM) to provide a unified solution for regression, binary, and multiclass classifications (like ELM). To address large-scale data training, we approximate the deformed kernel by random feature mapping, so that the proposed DKELM does not need parameter tuning and has less computational complexity, as well as a natural out-of-sample extension for novel examples. We demonstrate the relationship between the traditional kernel-based learning approach and ELM, and our approach can be used by other kernel-based methods and a sequence of fast learning algorithms can be derived.

The rest of this paper is organized as follows. Some previous works are introduced in Section II. The method of constructing and approximating the deformed kernel is discussed in Section III. In Section IV, we first demonstrate the relationship between the traditional kernel-based learning approach and ELM, and propose the deformed kernel based extreme learning machine. The experiments using benchmark real-world data sets are reported in Section V. Finally, we conclude this paper in Section VI.

II. Brief of the Extreme Learning Machine

The output function of ELM for generalized SLFNs in the case of one output node case is

$$f_L(x) = \sum_{i=1}^{L} \beta_i h_i(x) = h(x)\beta \qquad (1)$$

where $\beta = [\beta_1, \dots, \beta_L]^T$ is the vector of the weights between a hidden layer of L nodes and the output node. Note that $h(x) = [h_1(x), \dots, h_L(x)]$ is the output (row) vector of the hidden layer with respect to the input x. In fact, $h(x)$ maps the data from the d-dimensional input space to the L-dimensional hidden-layer feature space (ELM feature space) H. Different from traditional learning algorithms [11], ELM is meant to minimize the training error as well as the norm of the output weights [7]

$$\text{Minimize: } \|H\beta - T\|^2 \text{ and } \|\beta\| \qquad (2)$$

where H is the hidden-layer output matrix, which is denoted by

$$H = \begin{bmatrix} h(x_1) \\ h(x_2) \\ \vdots \\ h(x_n) \end{bmatrix} = \begin{bmatrix} h_1(x_1) & \dots & h_L(x_1) \\ h_1(x_2) & \dots & h_L(x_2) \\ \vdots & \vdots & \vdots \\ h_1(x_n) & \dots & h_L(x_n) \end{bmatrix}. \qquad (3)$$

As with SVM for the binary classification, to minimize the norm of the output weights $\|\beta\|$ is actually used to maximize the distance of the separating margins of the two different classes in the ELM feature space: $2/\|\beta\|$. The norm controls the complexity of the function in the ambient space, which will be elaborated later.

If a feature mapping h(x) is unknown to users, the output function of ELM classifier is

$$f(x) = h(x)H^T \left(\frac{I}{C} + HH^T\right)^{-1} T = [k(x, x_1), \dots, k(x, x_n)] \left(\frac{I}{C} + M\right)^{-1} T \qquad (4)$$

where $M = HH^T$, $m_{i,j} = k(x_i, x_j)$ and $k(x, y)$ is a kernel function.

If a feature mapping h(x) is known, we have

$$h(x) = [G(a_1, b_1, x), \dots, G(a_L, b_L, x)] \qquad (5)$$

where $G(a, b, x)$ is a nonlinear piecewise continuous function satisfying ELM universal approximation capability theorems [7],[30] and $\{(a_i, b_i)\}_{i=1}^{L}$ are randomly generated according to any continuous probability distribution. The output function of ELM classifier is

$$f(x) = h(x)\beta = h(x)H^T \left(\frac{I}{C} + HH^T\right)^{-1} T, \qquad (6)$$

or

$$f(x) = h(x)\beta = h(x) \left(\frac{I}{C} + H^T H\right)^{-1} H^T T, \qquad (7)$$

where $T = \begin{bmatrix} t_{11} & \dots & t_{1m} \\ t_{21} & \dots & t_{2m} \\ \vdots & \vdots & \vdots \\ t_{n1} & \dots & t_{nm} \end{bmatrix}$ and m is the number of classes.

III. Deforming the Kernel by Warping an RKHS

For a Mercer kernel K: $X \times X \to \square$, there is an associated RKHS H_K of functions $X \to \square$ with the corresponding norm $\| \ \|_K$. Given a set of l labeled examples $\{(x_i, y_i)\}_{i=1}^{l}$ and a set of u unlabeled examples $\{x_j\}_{j=l+1}^{l+u}$, where $x_i \in \mathbb{R}^N$ and $y \in \{+1. -1\}$, the classical kernel-based learning approach is based on solving the regularization problem given by

$$f = \text{argmin}_{f \in H_K} \left\{\frac{1}{l}\sum_i^l V(f, x_i, y_i) + \gamma_A \|f\|_k^2\right\}, \qquad (8)$$

where V is some loss function, such as the squared loss $(y_i - f(x_i))^2$ for RLS and the hinge loss function $\max[0, 1 - y_i f(x_i)]$ for SVM; $\|f\|_k^2$ is the norm of the classification function f in the reproducing kernel Hilbert space H_K, and γ_A controls the complexity of function f. The Representer Theorem [27] states that a solution can be found in the form $f(x) = \sum_{i=1}^{l} \alpha_i K(x, x_i)$. In order to avoid confusion, we list main notations of this paper in Table I.

TABLE I.

Notations

Notation	Explanation
$\mathbb{R}^d$	The input d-dimensional Euclidean space
X	$x = [x_1, \dots, x_l, \dots, x_{l+u}] \in \mathbb{R}^{d \times (l+u)}$ is the training data matrix. $\{x_i\}_{i=1}^{l}$ are labeled points, and $\{x_j\}_{i=l+1}^{l+u}$ are unlabeled points.
m	The number of classes that the samples belong to
Y	$Y = (y_1, \dots, y_l) \in \mathbb{R}^{m \times l}$ is the 0-1 label matrix. $y_i \in \mathbb{R}^m$ is the label vector of x_i, and all components of y_i are 0 s except one being 1.
F(·)	$F(x) = (f_1(x), \dots, f_m(x))^T$ is the discriminative vector function. The index of the class which x

	belongs to is that of the component with the maximum value.
k(x, y)	Kernel function of variables x and y
K	Kernel matrix $K = \{k(x_i, x_j)\} \in \mathbb{R}^{(l+u)\times(l+u)}$ or $\mathbb{R}^{l\times l}$
B	$B = (\beta_1, \dots, \beta_l) \in \mathbb{R}^l$. Its columns are the coefficients of the kernel function to represent the discriminative function F(·).
$\| \ \|_K$	Norm in the Hilbert space H_K
$\langle \ \rangle_K$	Inner product in the Hilbert space H_K
tr(M)	The trace of the matrix M , that is, the sum of the diagonal elements of the matrix M.
span{k(x,·)|x ∈ χ}	subspace expanded by k(x,·)

In the implementation of this kernel-based learning approach, we often use the Radial Basis Function or Gaussian (RBF) as kernel, and the kernel k defines a unique RKHS. Since the Gaussian kernel is isotropic, it has a spherical symmetry. That is, it generally does not conform to the particular geometry of the underlying classes. In other words, the underlying data structure is obviated. Finally, it is unable to provide an accurate decision surface. To address these limitations, it is crucial to define a new kernel that is adapted to the geometry of the data distribution well.

Instead of solving (8) like a traditional kernel-based learning approach, we modify (or deform) the original kernel K in order to adapt it to the underlying distribution geometry. Defining a new deformed kernel $\tilde{k}$, the new problem to be solved becomes

$$\tilde{f} = \operatorname{argmin}_{f\in\tilde{H}_K} \left\{\frac{1}{l}\sum_i^l V(f, x_i, y_i) + \gamma_A \|f\|_{\tilde{k}}^2\right\}, \quad (9)$$

(8) and (9) solved with different kernels, and thus in different RKHS.

The solution of (9) is

$$\tilde{f}(\mathrm{x}) = \sum_{i=1}^{l} \alpha_i \tilde{k}(x, x_i), \quad (10)$$

that should be appropriate for real setting.

To "deform" the original kernel and adapt it to the geometry of the underlying distribution, let $\mathcal{V}$ be a linear space with positive semi-definite inner product, and let $S: H_K \rightarrow \mathcal{V}$ be a bounded linear operator. Defining $\widetilde{H}_K$ to be the space with the same functions as H_K and its inner product defines

$$\langle f, g\rangle_{\tilde{k}} = \langle f, g\rangle_k + \langle Sf, Sg\rangle_{\mathcal{V}}, \quad (11)$$

It is proved in [27] that $\widetilde{H}_K$ is a valid RKHS. In this specific problem, it is required that S and $\mathcal{V}$ should depend on the data. Therefore, let $\mathcal{V}$ be $\mathbb{R}^{l+u}$, and define S as the evaluation map $S(f) \equiv f = (f(x_1, \dots, f(x_{l+u}))$ .Using a symmetric positive semi-definite matrix M, the semi-norm on $\mathbb{R}^{l+u}$ can be written as $\|Sf\|_{\mathcal{V}}^2 = f^T M f$. With such a norm, the regularization problem in (9) becomes

$$\tilde{f} = \operatorname{argmin}_{f\in\tilde{H}_K} \left\{\frac{1}{l}\sum_i^l V(f, x_i, y_i) + \gamma_A(\|f\|_k^2 + f^T M f)\right\} \quad (12)$$

where f includes both labeled and unlabeled data and the matrix M encodes smoothness w.r.t. the graph or manifold.

Let $M = \gamma_I/\gamma_A\, L$ and substitute it into (12), we have

$$\tilde{f} = \operatorname{argmin}_{f\in\tilde{H}_K} \left\{\frac{1}{l}\sum_i^l V(f, x_i, y_i) + \gamma_A\|f\|_k^2 + \gamma_I f^T L f)\right\} \quad (13)$$

where $\gamma_I/\gamma_A \in (0, \infty]$ is a free parameter that controls the "deformation" of the original kernel. Thus, Equation (13), in fact, is a graph-based semi-supervised learning problem based on the manifold assumption; it can be indirectly set out using (12) and solved using (10).

To utilize the geometry information of the data distribution, a graph G can be constructed using labeled and unlabeled pixels. The graph Laplacian of G is a matrix defined as $L = \mathrm{D} - \mathrm{W}$, where W is the adjacency matrix. The elements W_{ij} are measures of the similarity between pixels x_i and x_j , and the diagonal matrix D is given by $D_{ii} = \sum_{j=1}^{l+u} W_{ij}$. The graph Laplacian L measures the variation of the function f along the graph built upon all labeled and unlabeled samples. By fixing $\gamma_I/\gamma_A = 0$, the original (undeformed) kernel is obtained.

Next, we discuss the computation of the deformed kernel $\tilde{k}$. In [27], the resulting new kernel was computed explicitly in terms of labeled and unlabeled data. It is proved that

$$k(x,\cdot) - \tilde{k}(x,\cdot) \in span\{\tilde{k}(x_1,\cdot), \dots, \tilde{k}(x_{l+u},\cdot)\} \quad (14)$$

and

$$span\{\tilde{k}(x_i,\cdot)\}_{i=1}^{l+u} \equiv span\{k(x_i,\cdot)\}_{i=1}^{l+u} \quad (15)$$

Thus, the two spans are same and we have

$$\tilde{k}(x,\cdot) = k(x,\cdot) + \sum_j \beta_j(x) k(x_j,\cdot) \quad (16)$$

where the coefficients β_j depend on x, let $\beta(x) = (\beta_1(x) \dots \beta_{l+u}(x))^T$.

We can compute $k(x_i,\cdot)$ at x:

$$k_{x_i}(x) = \langle k(x_i,\cdot), \tilde{k}(x,\cdot)\rangle_{\tilde{k}}$$

$$= \langle k(x_i,\cdot), k(x,\cdot) + \sum_j \beta_j(x) k(x_j,\cdot)\rangle_{\tilde{k}}$$

$$= \langle k(x_i,\cdot), k(x,\cdot) + \textstyle\sum_j \beta_j(x) k(x_j,\cdot)\rangle_k + K_{x_i}{}^T M g \quad (17)$$

Where $K_{x_i} = (k(x_i, x_1) \dots k(x_i, x_{l+u}))^T$ and g is the vector given by components $g_k = k(x, x_k) + \sum_j \beta_j(x) k(x_j, x_k)$. Therefore, it can be derived from (17) that

$$(\mathrm{I}+\mathrm{MK})\,\beta(x) = -MK_x \quad (18)$$

where K is the kernel matrix $K_{ij} = K(x_i, x_j), 1 \le i, j \le l+u$, $K_x = (k(x_1, x) \dots k(x_{l+u}, x))^T$ and **I** is the identity matrix. Finally, we obtain the following explicit form for $\tilde{k}$:

$$\tilde{k}(x, z) = k(x, z) - K_x^T (I + MK)^{-1} M K_z \quad (19)$$

where $M = \gamma_I/\gamma_A\, L$. It satisfies the Mercer's conditions, being a valid kernel. If $\gamma_I/\gamma_A = 0$, the deformed kernel is degenerated into the original

(undeformed) kernel. When M is singular, one adds a small ridge term to M and uses a continuity argument.

IV. ELM Based on Deformed Kernel

In regularization problem (8), if yi is an m-dimensional label vector with the elements 0 or 1, where m is the number of classes, and xi belongs to the k-th class, then the k-th component of yi takes the value 1 and the rest components take the value 0. The corresponding vector function is defined as $F(x) = (f_1(x), \dots, f_m(x))^T$. Then the extended regularization problem estimates an unknown vector function by minimizing

$$F^* = \operatorname{argmin}_{F\in S}\left\{\frac{1}{l}\sum_i^l V(F, x_i, y_i) + \gamma_A\|F\|_k^2\right\} \quad (20)$$

where. $S = \{ f_1(x), f_2(x), \dots, f_m(x) | f_i(x) \in$ RKHS $H_K, 1 \le \mathrm{i} \le \mathrm{m}\}$ $F = (F(x_1), \dots, F(x_l)) \in \mathbb{R}^{m\times l}$

In (20), if we introduce a deformed kernel $\tilde{k}$, the problem to be solved becomes

$$\tilde{F}^* = \operatorname{argmin}_{\tilde{F}\in\tilde{S}}\left\{\frac{1}{l}\sum_i^l V\left(\tilde{F}, x_i, y_i\right) + \gamma_A\left\|\tilde{F}\right\|_{\tilde{k}}^2\right\} \quad (21)$$

where $\tilde{S} = \{ f_1(x), f_2(x), \dots, f_m(x) | f_i(x) \in$ RKHS $H_{\tilde{k}}, 1 \le \mathrm{i} \le \mathrm{m}\}$, $\tilde{F} = (F(x_1), \dots, F(x_l)) \in \mathbb{R}^{m\times l}$. The solution of (21) is $\tilde{F}(\mathrm{x}) = \sum_{i=1}^l \beta_i \tilde{k}(x, x_i)$, where $\beta_i = (\beta_{1i}, \beta_{2i}, \dots, \beta_{mi})^{\mathrm{T}}$

Based on what is introduced above, the regularization problem for DKELM with multioutput nodes can be formulated as

$$\tilde{F}^* = \operatorname{argmin}_{\tilde{F}\in\tilde{H}_K}\left\{\frac{C}{2}\sum_i^l\left(y_i - \tilde{F}(x_i)\right)^2 + \frac{1}{2}\left\|\tilde{F}\right\|_{\tilde{k}}^2\right\} \quad (22)$$

The solution of the optimization problem (22) is given by

$$\mathrm{B}^* = \mathrm{Y}\left(\frac{\mathbf{I}}{C} + \tilde{K}\right)^{-1},$$

where **I** is the identity matrix. $\mathrm{B}^{*\mathrm{T}} = \left(\frac{\mathbf{I}}{C} + \tilde{K}\right)^{-1}\mathrm{Y}^{\mathrm{T}}$.

Let $\tilde{H} = \left(\tilde{\emptyset}(x_1), \tilde{\emptyset}(x_2), \dots \tilde{\emptyset}(x_l)\right)^{\mathrm{T}}$ and $\tilde{\emptyset}(x) = \frac{1}{\sqrt{L}}\left(\tilde{k}_{b_1}(x, a_1), \tilde{k}_{b_2}(x, a_2), \dots, \tilde{k}_{b_L}(x, a_L)\right)$, so $\tilde{K} \approx \tilde{H}\tilde{H}^T$ and $\mathrm{B}^{*\mathrm{T}} = \left(\frac{\mathbf{I}}{\mathrm{C}} + \tilde{H}\tilde{H}^T\right)^{-1}\mathrm{Y}^{\mathrm{T}}$

,where $\tilde{K} = (\tilde{k}(x_i, x_j)) \in \mathbb{R}^{l\times l}$ is the deformed kernel matrix over labeled points.

If the number of labeled samples is not huge, the output function is

$$F^*(x)^{\mathrm{T}} = \tilde{\emptyset}(x)\tilde{H}^T\mathrm{B}^{*\mathrm{T}} = \tilde{\emptyset}(x)\tilde{H}^T\left(\tilde{H}\tilde{H}^T + \frac{\mathbf{I}}{\mathrm{C}}\right)^{-1}\mathrm{Y}^{\mathrm{T}}, \quad (23)$$

if the number of labeled samples is huge, according to the Sherman-Morrison-Woodbury(SMW) formula for matrix inversion, we have

$$F^*(x)^{\mathrm{T}} = \tilde{\emptyset}(x)\left(\frac{\mathbf{I}}{\mathrm{C}} + \tilde{H}^T\tilde{H}\right)^{-1}\tilde{H}^T\mathrm{Y}^{\mathrm{T}}, \quad (24)$$

where $Y = (\boldsymbol{y_1}, \dots, \boldsymbol{y_l}) \in \mathbb{R}^{m\times l}$ is the label matrix with elements 0 or 1, and $\boldsymbol{y_i}$ $(1 \le i \le l)$ is an m-dimensional label vector with the elements 0 or 1. In a semi-supervised case, the number of labeled samples is small, so (40) should be used to compute the output function.

Next, we discuss the computation of the deformed kernel $\tilde{k}$, according to

$$\tilde{k}(x, z) = k(x, z) - K_x^t(I + MK)^{-1}MK_z \ ,$$

where $K = (k(x_i, x_j)) \in \mathbb{R}^{(l+u)\times(l+u)}$ is the kernel matrix over labeled and unlabeled samples, we have to compute a matrix inversion of size $(l+u)\times(l+u)$. Note that this inversion scales exponentially with the number of samples. If the number of labeled and unlabeled samples is huge, it is difficult to compute. So we further approximate the kernel matrix K, letting $H = (\emptyset(x_1), \emptyset(x_2), \dots \emptyset(x_{l+u}))^{\mathrm{T}}$ and $\emptyset(x) = \frac{1}{\sqrt{L}}\left(k_{b_1}(x, a_1), k_{b_2}(x, a_2), \dots, k_{b_L}(x, a_L)\right)$, then $K \approx HH^T$, $K_x^T \approx \emptyset(x)H^T$ and $K_Z \approx H\emptyset^T(z)$, so we achieve

$$\begin{aligned}\tilde{k}(x, z) &\approx k(x, z) - K_x^T(I + MHH^T)^{-1}MK_z \\ &= k(x, z) - \emptyset(x)H^T(I + MHH^T)^{-1}MH\emptyset^T(z) \\ &= k(x, z) - \emptyset(x)H^T[I - MH(I + H^TMH)^{-1}H^T]MH\emptyset^T(z) \\ &= k(x, z) - \emptyset(x)[H^TMH - H^TMH(I + H^TMH)^{-1}H^TMH]\emptyset^T(z) \\ &= k(x, z) - \emptyset(x)[H^TMH(I + H^TMH)^{-1}]\emptyset^T(z) \\ &= k(x, z) - \emptyset(x)[Q(I + Q)^{-1}]\emptyset^T(z), \end{aligned} \quad (25)$$

where $Q = H^TMH, M = \gamma_I/\gamma_A L = 2\gamma_I L$.

Correspondingly, in a semi-supervised case, the output function of DKELM with a single output is

$$f^*(x) = \tilde{\emptyset}(x)\tilde{H}^T\left(\tilde{H}\tilde{H}^T + \frac{\mathbf{I}}{\mathrm{C}}\right)^{-1}Y'^{\mathrm{T}} \quad (26)$$

where Y' is an l dimensional label vector given by: $Y' = (y_1, \dots, y_l)$.

The formula (25) and (26) all involve the inversion of a matrix of order $L \times L$, as long as L is large enough, the generalization performance of DKELM is not sensitive to the dimensionality of the feature space (L) and good performance can be reached, which will be verified later in Section 5. The DKELM algorithm is summarized in the Table II.

TABLE II.

THE DESCRIPTION OF DKELM ALGORITHM BASED ON DEFORMED KERNEL

DKELM Algorithm based on deformed kernel
Input: l labeled examples$\{(x_i, y_i)\}_{i=1}^l$, u unlabeled examples $\{x_j\}_{j=l+1}^{l+u}$.
Output: Estimated function $F^*(x) = (f_1^*(x), \dots, f_2^*(x), \dots, f_m^*(x))^{\mathrm{T}}$.
Step 1: Construct data adjacency graph with (l+u) nodes using k nearest neighbors or a graph kernel. Choose edge weights Wij, for example, for binary weights or heat kernel weights $W_{ij} = e^{-\|x_i - x_j\|^2/4t}$.
Step 2: Compute graph Laplacian matrix: $L_G = D - W$, where D is a diagonal matrix given by $D_{ii} = \sum_{j=1}^{l+u} W_{ij}$.
Step 3: Choose a kernel function $k(x, y)$. Choose γ_I, C and L (the number of sample points), randomly generate $\{(a_i, b_i)\}_{i=1}^L$.
Step 4: if the number of the training data sets is very large $(l+u) \gg L$, compute $H = (\emptyset(x_1), \emptyset(x_2), \dots \emptyset(x_{l+u}))^{\mathrm{T}}$, $\emptyset(x) = \frac{1}{\sqrt{L}}\left(k_{b_1}(x, a_1), k_{b_2}(x, a_2), \dots, k_{b_L}(x, a_L)\right)$, select (25) for computing the deformed kernel; Otherwise, use (19) .
Step 5: Select (23) for computing the output function $F^*(x)$ of DKELM

with multioutputs or select (26) for computing the output function $f^*(x)$ of DKELM with single output (m=1).

Step 6: Output $F^*(x) = (f_1^*(x), \ldots, f_2^*(x), \ldots, f_m^*(x))^{\mathrm{T}}$.

Like ELM, DKELM has the unified solutions for regression, binary and multiclass classification. But we mainly discuss DKELM for the classification problems in this paper.

A DKELM classifier with a single-output node (m = 1): For multiclass problems, among all the multiclass labels, the predicted class label of a given testing sample is closest to the output of a DKELM classifier. The decision function of the DKELM classifier is

$$f^*(x) = \widetilde{\emptyset}(x)\widetilde{H}^T(\widetilde{H}\widetilde{H}^T + \frac{\mathbf{I}}{\mathrm{C}})^{-1}Y'^{\mathrm{T}}. \quad (27)$$

For the binary classification, the decision function of DKELM classifier is

$$f^*(x) \approx \operatorname{sign}\left(\widetilde{\emptyset}(x)\widetilde{H}^T(\widetilde{H}\widetilde{H}^T + \frac{\mathbf{I}}{\mathrm{C}})^{-1}Y'^{\mathrm{T}}\right). \quad (28)$$

A DKELM classifier with multioutput nodes (m > 1) is: For multiclass cases, the predicted class label of a given testing sample is the index number of the output node, which has the highest output value for the given testing sample. The decision function of the DKELM classifier is

$$F^*(x)^{\mathrm{T}} \approx \widetilde{\emptyset}(x)\widetilde{H}^T(\widetilde{H}\widetilde{H}^T + \frac{\mathbf{I}}{\mathrm{C}})^{-1}\mathrm{Y}^{\mathrm{T}}. \quad (29)$$

The predicted class label of sample x is

$$label(x) = \arg\max_{i\in\{1,\ldots,m\}} f_i^*(x). \quad (30)$$

The deformed kernel in both cases is computed by

$$\tilde{k}(x,z) = k(x,z) - K_x^t(I + MK)^{-1}MK_z,$$

which is applied to moderate scale training samples, or

$$\tilde{k}(x,z) \approx k(x,z) - \emptyset(x)[Q(I+Q)^{-1}]\emptyset^T(z), \quad (31)$$

which is applied to large scale training samples, where $Q = H^T MH, M = 2\gamma_I L_G$.

V. EXPERIMENTS

In this section, we will validate the performance of the proposed DKELM algorithm on a number of real-world data sets. In particular, we studied the sensitivity of DKELM to the number of labeled samples. All the experiments are performed with MATLAB 7.0.1 environment on a 3.10GHZ Intel CoreTM i5-2400 with 3-GB RAM.

A. Data Sets

We used different scale data sets from the UCI machine learning repository (satellite, Ionosphere), and another benchmark repository (Extended Yale B, USPS). For the satellite data sets, there are multiple class labels; we used their first two classes only. For USPS, we randomly selected 250 data points from each class for our experiments. The basic information about these data sets is summarized in Table III.

TABLE III.

DESCRIPTION OF THE DATA SETS

Data	Size (n)	Feature (d)	Class
SatelliteC1-C2	2236	36	2
Ionosphere	351	34	2
Extended Yale B	2114	1024	38
USPS	2500	256	10

B. Parameter selection and experimental settings

Comparisons are made with four important classification methods: CutS^{3}VM[25], L_2-TSVM-MFN[26], DA L_2-SVM-MFN[26] and S-RLSC algorithm[29]. In our experiments, binary edge weights are chosen and the neighborhood size k is set to be 12 for all the data sets. DKELM algorithm needs to choose the feature mapping, the cost parameter C and the number of hidden nodes L, since ELM algorithm achieves good generalization performance as long as L and C are large enough[30]. Thus we let C =500. The regularization parameters γ_A and γ_I are split into the ratio 1:9, and we let $CA = 1/C$, $CI = 2\gamma_I/((l+u)^2 \cdot C) = 9/C$, which is set in the same way as in [27]. We select Gaussian functions as the hidden-node output functions.

We test L_2-TSVM-MFN with multiple switchings and DA L_2-SVM-MFN with parameter $\lambda = 0.001$ and $\lambda' = 1$ on all datasets. We also test CutS3VM with parameters $C_l = C_u = 1$, and set r in the balancing constraint to the true ratio of the positive points in the unlabeled set. The S-RLSC methods also have regularization parameters γ_A and γ_I. Let $CA = \gamma_A l$, $CI = \gamma_I l/(l+u)^2$, and also use the Gaussian kernel function. In our experiments, we also set CA=0.005, CI=0.01 and $\sigma = 0.5$ for all data sets, which is set in the same way as in [29].

For each data set T, 15% of the data points are left for out-of-sample extension experiment. We denote by T_r the rest data points of the data set T. In each class of T_r, we randomly label l data points to train every algorithm. For DKELM, S-RLSC, L_2-TSVM-MFN, DA L_2-SVM-MFN and CutS^{3}VM, the training set consists of the whole T_r, including the labeled and the unlabeled data points. For L_2-TSVM-MFN, DA L_2-SVM-MFN and CutS^{3}VM, multiclass datasets are learned using a one-versus-rest approach.

C. Experimental results

For simplicity, we used DKELM with a single output and 800 hidden nodes; the recognition result of all the algorithms is shown in Table 4–6, respectively. For each dataset, classification accuracy and training time averaged over 20 independent trials. The number of l (in each class) of the labeled data points varies from 5 to 250 for the Satellite data set, from 5 to 150 for the Ionosphere and from 5 to 40 for the Extended Yale B data set.

In Tables 4–6, for several values of m, the best classification results are in boldface for each fixed value of m. As can be seen from the tables, the classification accuracy is lower for all algorithms when l is small. With the increase of labeled data, the discriminative ability of the DKELM algorithm is much better than the other algorithms, since it utilizes the manifold structure of labeled and unlabeled samples. The recognition result of the S-RLSC algorithm is very close to that of the

DKELM algorithm, but it runs much slower than our algorithm. For the Satellite and Ionosphere data sets, the performance of the DKELM algorithm is worse than that of Extended Yale B data set, since the manifold structure is less salient than that of face images. As can be seen from Table 6, the recognition accuracy of the L2-TSVM-MFN, DA L_2-SVM-MFN and CutS3VM classifiers decreases with the increase of the number of classes, since these classifiers are constructed with a one-versus-rest approach, which has a great influence on the accuracy. Moreover, this kind of multiclass classification approach also increases the running time of these algorithms. With the increase of the number of the feature dimensions of data sets, the running time of the CutS3VM increases dramatically, since its time complexity depends on the average number of non-zero features. In contrast, as can be seen from Table IVandVI, the speed of the DKELM algorithm is not sensitive to the number of classes and the feature dimensions of data sets. It can perform well by means of the intrinsic geometry of data distribution.

TABLE IV.

PERFORMANCE COMPARISON OF ALL T HE ALGORITHMS FOR THE SATELLITE DATA SET

Number of labeled data points *l*	DKELM Accuracy (%)	*S-RLSC* Accuracy(%)	*L_2-TSVM-MFN* Accuracy (%)	*DA L_2-SVM-MFN* Accuracy (%)	CutS^{3}VM Accuracy (%)
l =5	57.62	62.78	56.48	61.02	73.59
l =10	78.45	79.79	69.83	70.73	74.46
l =50	85.68	83.47	71.95	75.28	82.85
l =250	89.21	87.86	75.82	76.29	84.46
Number of labeled data points *l*	DKELM Training Time(s)	*S-RLSC* Training Time(s)	*L_2-TSVM-MFN* Training Time(s)	*DA L_2-SVM-MFN* Training Time(s)	CutS^{3}VM Training Time(s)
l =5	52.782	328.579	3.782	12.652	1.190
l =10	55.273	328.647	3.894	11.676	0.976
l =50	54.374	330.152	2.957	10.016	0.620
l =250	52.962	322.674	2.365	5.625	0.569

TABLEV.

PERFORMANCE COMPARISON OF ALL THE ALGORITHMS FOR THE IONOSPHERE DATA SET

Number of labeled data points *l*	DKELM Accuracy (%)	*S-RLSC* Accuracy(%)	*L_2-TSVM-MFN* Accuracy (%)	*DA L_2-SVM-MFN* Accuracy (%)	CutS^{3}VM Accuracy (%)
l =5	57.62	72.78	66.52	60.81	73.89
l =10	74.79	73.45	69.42	71.67	74.56
l =50	87.68	83.47	81.14	77.21	85.25
l =250	90.21	87.86	85.73	86.39	86.43
Number of labeled data points *l*	DKELM Training Time(s)	*S-RLSC* Training Time(s)	*L_2-TSVM-MFN* Training Time(s)	*DA L_2-SVM-MFN* Training Time(s)	CutS^{3}VM Training Time(s)
l =5	21.704	168.579	0.618	5.724	0.554
l =10	25.753	168.647	0.772	5.676	0.377
l =50	23.176	170.152	0.252	4.534	0.324
l =250	22.802	162.674	0.246	3.165	0.232

TABLEVI

PERFORMANCE COMPARISON OF ALL THE ALGORITHMS FOR THE EXTENDED YALE B DATA SET

Number of labeled data points *l*	DKELM Accuracy(%)	*S-RLSC* Accuracy(%)	*L_2-TSVM-MFN* Accuracy (%)	*DA L_2-SVM-MFN* Accuracy (%)	CutS^{3}VM Accuracy (%)
l =5	61.25	64.41	55.17	38.47	63.93
l =10	82.05	83.18	62.76	58.71	69.82
l =20	94.52	93.10	67.72	68.49	75.91
l =30	95.42	95.24	71.35	73.59	79.56
l =40	97.44	97.12	75.25	76.84	80.13
Number of labeled data points *l*	DKELM Training Time(s)	*S-RLSC* TrainingTime(s)	*L_2-TSVM-MFN* Training Time(s)	*DA L_2-SVM-MFN* Training Time(s)	CutS^{3}VM Training Time(s)
l =5	87.589	452.644	60.427	361.928	75.249
l =10	88.670	455.972	58.958	273.536	69.162
l =20	86.465	452.177	49.514	247.923	65.368
l =30	87.259	454.921	42.943	190.380	58.532
l =40	89.694	452.228	35.519	163.476	54.348

The out-of-sample extension result of the algorithms on larger USPS data sets is shown in Fig. 1. We perform the DKELM algorithm using 500 hidden nodes. As can be seen from Fig.1, the DKELM algorithm has best recognition results over any other algorithm. So our proposed DKELM algorithm tends to have better scalability and achieve best generalization performance at a relatively faster learning speed.

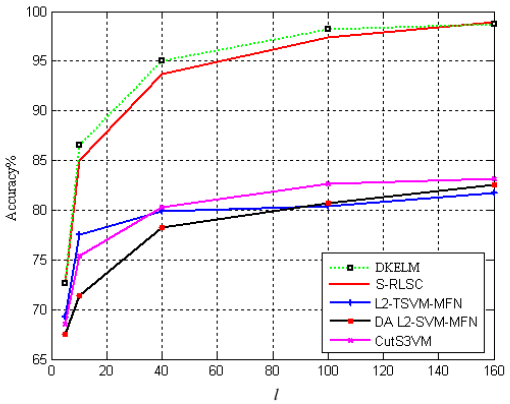

Figure 1. Out-of-sample extension classification results on the USPS data set

VI CONCLUSIONS

In this paper, we first extended the traditional kernel-based learning problem to multiclass cases in an Extreme Learning Machine context. To enhance the performance of ELM, a deformed kernel was proposed, which can make use of underlying information from both labeled and unlabeled samples. To speed up our algorithm, we further approximated the deformed kernel by means of random feature mapping. Our algorithm does not need kernel parameter tuning. The experimental results have shown that the DKELM algorithm achieves better generalization performance at a relatively faster learning speed than traditional semi-supervised classification algorithms. In the future, we will further optimize our proposed framework and study the sparse regularization problems in our framework.

ACKNOWLEDGMENTS

This work was supported by the National Natural Science Foundation of China under Grant Nos. 50674086.

REFERENCES

[1] X. Tang and M. Han, "Partial Lanczos extreme learning machine for single-output regression problems," Neurocomputing, vol. 72, no. 13–15, pp. 3066–3076, Aug. 2009.

[2] Q. Liu, Q. He, and Z.-Z. Shi, "Extreme support vector machine classifier," Lecture Notes in Computer Science, vol. 5012, pp. 222–233, 2008.

[3] B. Frénay and M. Verleysen, "Using SVMs with randomised feature spaces: An extreme learning approach," in Proc. 18th ESANN, Bruges, Belgium, Apr. 28–30, pp. 315–320,2010.

[4] Y. Miche, A. Sorjamaa, P. Bas, O. Simula, C. Jutten, and A. Lendasse, "OP-ELM: Optimally pruned extreme learning machine," IEEE Trans. Neural Netw., vol. 21, no. 1, pp. 158–162, Jan. 2010.

[5] W. Deng, Q. Zheng, and L. Chen, "Regularized extreme learning machine," in Proc. IEEE Symp. CIDM, Mar. 30–Apr. 2, pp. 389–395,2009.

[6] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Extreme learning machine: A new learning scheme of feedforward neural networks," in Proc. IJCNN, Budapest, Hungary, Jul. 25–29, vol. 2, pp. 985–990.,2004.

[7] 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, Dec. 2006.

[8] G.-B. Huang, L. Chen, and C.-K. Siew, "Universal approximation using incremental constructive feedforward networks with random hidden nodes," IEEE Trans. Neural Netw., vol. 17, no. 4, pp. 879–892, Jul. 2006.

[9] G.-B. Huang and L. Chen, "Convex incremental extreme learning ma-chine," Neurocomputing, vol. 70, no. 16–18, pp. 3056–3062, Oct. 2007.

[10] G.-B. Huang and L. Chen, "Enhanced random search based incremental extreme learning machine," Neurocomputing, vol. 71, no. 16–18, pp. 3460–3468, Oct. 2008.

[11] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagation errors," Nature, vol. 323, pp. 533–536, 1986.

[12] T. Joachims. Transductive inference for text classification using support vector machines. In ICML 16, pp:200-209, 1999.

[13] K. P. Bennett and A. Demiriz. Semi-supervised support vector machines. In NIPS, pages 368-374, 1999.

[14] O. Chapelle, V. Sindhwani, and S. S. Keerthi. Branch and bound for semi-supervised support vector machines. InNIPS, pages 217-224, 2006.

[15] O. Chapelle and A. Zien. Semi-supervised classification by low density separation. In AISTATS 10, 2005.

[16] O. Chapelle, B. Scholkopf, and A. Zien. Semi-Supervised Learning. MIT Press: Cambridge, MA, 2006.

[17] L. Xu, J. Neufeld, B. Larson, and D. Schuurmans. Maximum margin clustering. In NIPS, 2004.

[18] L. Xu and D. Schuurmans. Unsupervised and semi-supervised multi-class support vector machines. In AAAI, 2005.

[19] Z. Xu, R. Jin, J. Zhu, I. King, and M. Lyu. Efficient convex relaxation for transductive support vector machine. In NIPS, 2007.

[20] O. Chapelle, M. Chi, and A. Zien. A continuation method for semi-supervised svms. In ICML 23, pages 185-192,2006.

[21] A. Astorino and A. Fuduli. Nonsmooth optimization techniques for semi-supervised classification. IEEE Trans.Pattern Anal. Mach. Intell., 29(12):2135-2142, 2007.

[22] R. Collobert, F. Sinz, J. Weston, and L. Bottou. Large scale transductive svms. Journal of Machine Learning Research, 7:1687-1712, 2006.

[23] G. Fung and O. Mangasarian. Semi-supervised support vector machines for unlabeled data classification. Optimization Methods and Software, 15:29-44, 2001.

[24] V. Sindhwani, S. S. Keerthi, and O. Chapelle.Deterministic annealing for semi-supervised kernel machines. In ICML 23, pp: 841-848, 2006.

[25] Bin Zhao, Fei Wang, Changshui Zhang.CutS3VM: A Fast Semi-Supervised SVM Algorithm.The 14th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining(KDD).pp:830-838,August 24-27,2008.

[26] V. Sindhwani, S.S. Keerthi. Large Scale Semi-supervised Linear SVMs. 29th Annual International ACM SIGIR, Technical report, 2006.

[27] M. Belkin, V. Sindhwani, P.Niyogi, Manifold regularization: a geometric framework for learning from labeled and unlabeled examples, J. Mach. Learn. Res. 7:2399–2434,2006.

[28] H. Xue, S. Chen, Q. Yang, Discriminatively regularized least-squares classification, Pattern Recognition 42(1) pp:93–104,2009.

[29] Mingyu Fan, Nannan Gu, Hong Qiao, etc. Sparse regularization for semi-supervised classification .Pattern Recognition, 44(8,), pp: 1777-1784,2011

[30] G.-B. Huang, H.-M Zhou, X.-J. Ding. Extreme Learning Machine for Regression and Multiclass Classification IEEE Transactions on Systems, Man, and Cybernetics-PART B: Cybernetics, Vol. 42, no. 2, 513 – 529,2012.

[31] Huang et al. Extreme learning machines: a survey .International Journal of Machine Learning and Cybernetics. pp:107–122,2011.

Zhang Chen is current a Ph.D candidate at China University of Mining and Technology(CUMT), China. She received her MS degree in Computer Application Technology from CUMT in 2004, and her BS degree in Computer Science from CUMT in 2001. She is currently a lecture at school of Computer Science and Technology, CUMT. Her research interest is computation intelligence and machine learning et al.

Xia Shi Xiong is born in 1962, Ph.D. He is a professor at school of Computer Science and Technology in CUMT. He has published more than 60 research papers in journals and international conferences. His research interest is Wireless sensor networks and intelligent information processing et al.

Liu Bing is current a Ph.D candidate at China University of Mining and Technology(CUMT), China. She received her MS degree in Computer Application Technology from CUMT in 2005, and her BS degree in Computer Science from CUMT in 2002. She is currently a lecture at school of Computer Science and Technology, CUMT. His research interest is computation intelligence and machine learning et al.