Nonlinear Metric Learning for Semi-Supervised Learning via Coherent Point Drifting

***Abstract*—In this paper, a nonlinear metric learning framework is proposed to boost the performance of semi-supervised learning (SSL) algorithms. Formulated under a constrained optimization framework, the proposed method learns a smooth nonlinear feature space transformation that makes the input data points more linearly separable in Laplacian SVM (LapSVM). Coherent point drifting (CPD) is chosen as the geometric model with the consideration of its remarkable representation power in generating sophisticated yet smooth deformations. Under CPD, larger smoothness weights can be assigned to labeled data points, allowing them to exert more significant influences in SSL. Our framework has broad applicability, and it can be integrated with many other SSL classifiers than LapSVM. Experiments performed on both synthetic and real world datasets show the effectiveness of our CPD-LapSVM over the state-of-the-art metric learning solutions in SSL.**

I. Introduction

In many modern applications, including image search, information retrieval and genomics, while unlabeled data are abundant, labeled instances are scarce as they may be difficult or expensive to acquire. Semi-supervised learning (SSL) aims to solve the classification problem of such inputs by augmenting labeled data with large amount of unlabeled data to build better classifiers. A variety of SSL methods have been proposed in the literature, which include generative models [1], self-training [2], multi-view [3], transductive support vector machines (TSVM) [4], graph-based models [5] and neural network based models [6], among others.

To make up for the lack of labeled samples, SSL solutions are commonly designed under certain assumption regarding the distribution of input data. For example, generative models assume the data follow an identifiable mixture distribution, and seek to determine the components through labeled samples [1]. The success of TSVM [4] depends on the validity of the assumption that unlabeled data from different classes are linearly separable under the feature space. Graph-based SSL methods [5] construct graphs where nodes represent samples (both labeled and unlabeled), and edges (may be weighted) reflect the similarity between samples. Nodes connected with large-weight edges are assumed highly likely to be assigned with the same labels.

The aforementioned assumptions in SSL, if violated in practice, could easily result in limited validity of the models and subsequently poor classification performance. Geometrically transforming the input points to make them in accordance with the assumed data distribution would provide a remedy. For distance/similarity based algorithms, learning such a transformation is equivalent to learning a new distance metric from the training samples. Distance metric learning (DML) has been extensively studied under supervised setting [7], [8]. However, it is not trivial to generalize supervised DML solutions, especially those developed under nearest neighbor (NN) paradigm, to handle SSL problems.

DML solutions for semi-supervised learning/classification commonly focus on formulating new regularizations to impose desired membership coherence throughout the data domain. LRML [9] and IDML [10] both assume the data lie approximately on a manifold of much lower dimension than the input space. The regularization term in LRML is a graph Laplacian, while IDML algorithm minimizes the harmonic energy over the data graph. Using a similar objective function as in LRML, SSM-DML [11] learns multiple Mahalanobis metrics for different feature sets. In SERAPH [12], an information-theoretic regularization is used to specify neighborhood relationship. OLapSVM [13] parameterizes graph weights through learning a Mahalanobis distance metric under Laplacian SVM. Despite the reported improvements, the existing semi-supervised DML solutions are mostly linear models performing under either input space or kernel space, which limit their capabilities in dealing with complex data.

In light of the limitations and drawbacks, we exploit the power of geometric space transformations to address the gap between model assumptions and actual data distributions. More specifically, we apply a deformation model called *Coherent Point Drifting* (CPD) [14] to make the transformed data points well conform to the underlying SSL assumptions. We choose Laplacian SVM (LapSVM), a classic graph-based SSL model, as the host solution to develop and demonstrate the effectiveness of our approach. Tailored to semi-supervised learning problems, we assign labeled points with larger influence ranges than the unlabeled. The choice of CPD is with two considerations: 1) its remarkable capability in generating high-order yet smooth deformations; and 2) the available mechanism within CPD for assigning different levels of smoothness to data points. To the best of our knowledge, this is the first attempt of utilizing globally smooth, nonlinear, dense spatial transformation models in semi-supervised learning. It should also be noted that, while the work present in this paper is based on LapSVM, our model has a broad applicability and can be readily extended to many other SSL solutions.

The rest of this paper is organized as follows. Section 2 introduces the background and related work of metric learning for SSL. The CPD transformation model is described in Section 3. Section 4 presents our CPD-LapSVM metric learning model integrating CPD transformation into semi-supervised

classification procedure. Experimental results are presented in Section 5 to validate our solutions with both synthetic and real world datasets. Section 6 concludes this paper.

II. Background and Related Work

Distance Metric Learning (DML) A metric over set $\mathcal{X}$ is a mapping function D: $\mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$. The goal of supervised metric learning is to learn a "better" metric, with the aid of training vectors $\mathbf{x_1}, \mathbf{x_2}, ..., \mathbf{x_n} \in \mathcal{X}$. The Mahalanobis distance has become one of the most widely studied metrics in supervised DML research [7], [8]. It is defined as: $D_{\mathbf{M}}(\mathbf{x_i}, \mathbf{x_j}) = \sqrt{(\mathbf{x_i} - \mathbf{x_j})^T \mathbf{M} (\mathbf{x_i} - \mathbf{x_j})}$, where $\mathbf{M}$ is a positive semi-definite (PSD) matrix (denoted as $\mathbf{M} \succeq 0$). Since $\mathbf{M}$ can always be decomposed as $\mathbf{M} = \mathbf{L}^T\mathbf{L}$, the Mahalanobis distance $D_{\mathbf{M}}$ can be rewritten as:

$$D_{\mathbf{M}}(\mathbf{x}_i, \mathbf{x}_j) = \sqrt{(\mathbf{L}\mathbf{x}_i - \mathbf{L}\mathbf{x}_j)^T (\mathbf{L}\mathbf{x}_i - \mathbf{L}\mathbf{x}_j)}. \quad (1)$$

Eqn. (1) indicates that the Mahalanobis metric with matrix $\mathbf{M}$ is essentially the Euclidean distance after a linear transformation $f : \mathbf{x} \rightarrow \mathbf{L}\mathbf{x}$. Learning a Mahalanobis metric therefore is equivalent to learning a linear transformation that ensures the resulted Euclidean distances would very well conform to the supervisory information. This observation also applies to nonlinear metric learning, and therefore the focus to learn a metric can be shifted to learn a geometric mapping.

In this paper, we propose a nonlinear SSL space transformation model for LapSVM. To help readers understand our approach, we introduce two of the related models, MSVM and OLapSVM, as follows.

A. Related Work

MSVM The Metric learning with SVMs (MSVM) algorithm [15] is formulated under the margin-radius-ratio bounded SVM paradigm. It simultaneously learns a SVM classifier and a Mahalanobis metric, expressed via a linear transformation $\mathbf{L}$, through solving the following constrained problem:

$$\begin{aligned} \min_{\mathbf{L},\mathbf{w},b} \quad & J = \frac{1}{2} R(\mathbf{L})^2 \|\mathbf{w}\|_2^2 + C \sum_{i=1}^{n} \xi_i \\ \text{s.t.} \quad & y_i(\mathbf{w}^T \mathbf{L}\mathbf{x}_i + b) \geq 1 - \xi_i, \quad \xi_i \geq 0, \quad \forall i = 1, .., n. \end{aligned}$$

Here, $R(\mathbf{L})$ is the radius of minimum enclosing ball (MEB) of data points in the transformed space. The MEB is the smallest ball that encloses all the data points, and its radius $R(\mathbf{L})$ can be obtained by:

$$R(\mathbf{L}) = \min_{r} \quad r, \quad \text{s.t.} \quad r \geq \|\mathbf{L}\mathbf{x}_i - \mathbf{L}\mathbf{x}_c\|_2, \quad \forall i = 1, .., n.$$

where $\mathbf{x}_c$ denotes the center of all the data points. It should be noted that, setting margin-radius-ratio bounds in SVMs is aimed to limit the transformation extension, and therefore ensure the convergence of the optimization procedure. From the transformation perspective, our work could be regarded as a nonlinear extension of MSVM, with Laplacian regularization targeting the SSL problems.

OLapSVM OLapSVM [13] is a metric learning model designed based on LapSVM. It aims to optimize graph Laplacians and learns task-specific similarity metrics from the labeled samples. With a transformation matrix $\mathbf{L}$ of size $m \times m$, where m is the dimension of the data, the edge weight between two data samples x_i and x_j is defined as:

$$w_{ij}(\mathbf{L}) = \exp(-\|\mathbf{L}(x_i - x_j)\|^2)$$

The optimal matrix $\mathbf{L}^*$ in OLapSVM is determined through minimizing the following objective function:

$$Q(\mathbf{L}) = \sum_{(i,j)\in\mathbf{F}} w_{ij}(\mathbf{L}) - \sum_{(i,j)\in\mathbf{S}} w_{ij}(\mathbf{L}) - \lambda \sum_{i=1}^{l+u} \sum_{j\in N_i} w_{ij}(\mathbf{L})$$

where $\mathbf{S}$ and $\mathbf{F}$ are equivalence and non-equivalence constraints defined by labels. N_i specifies the k nearest neighbors of $\mathbf{x}_i$ in the Euclidean space. l and u are the numbers of labeled and unlabeled data, respectively, and λ is a weighting factor.

III. Transformation Regularization and CPD Transformation Model

In transformation based applications (e.g. point matching), regularization is commonly required to ensure the well-posedness of the problem, as well as to generate a simple and smooth deformation field.

Let $v(\mathbf{x}) : \mathbb{R}^d \rightarrow \mathbb{R}^d$ denote a displacement function that moves dataset $\mathbf{x}$ towards $\mathbf{y}$. Estimation of an optimal $v(\cdot)$ has been commonly formulated under Tikhonov regularization framework,

$$\begin{aligned} \mathcal{R}[v] &= \mathcal{R}_{emp}[v] + \lambda \mathcal{R}_{reg}[v] \\ &= \frac{1}{2} \sum_{i=1}^{N} [\mathbf{y}_i - (\mathbf{x}_i + v(\mathbf{x}_i))]^2 + \frac{1}{2} \lambda ||\mathbf{D}v||^2 \end{aligned} \quad (2)$$

where N is the number of instances in dataset. $||\cdot||$ is the norm operator. λ is a regularization parameter that controls the trade-off between the data term and the regularization term. $\mathbf{D}$ is a linear differential operator.

The regularization functional on v is to ensure certain extent of smoothness, which can be essentially determined by the norm operator of the functional space. Different norm operators would lead to different smoothness functionals. A general norm of v in the Hilbert space $\mathbb{H}^m$ is given in [16]:

$$\|v\|_{\mathbb{H}^m}^2 = \int_{\mathbb{R}} \sum_{k=0}^{m} \| \frac{\partial^k v}{\partial x^k} \|^2 dx \quad (3)$$

Alternatively, the norm in the Reproducing Kernel Hilbert Space (RKHS) can also be defined as:

$$\|v\|_{\mathbb{H}^m}^2 = ||v||_K = \int_{\mathbb{R}^d} \frac{|\tilde{v}(\mathbf{s})|^2}{\tilde{K}(\mathbf{s})} d\mathbf{s} \quad (4)$$

where K is the kernel function associated to the RKHS, and $\tilde{K}$ is the Fourier transform of K. $\tilde{v}$ denotes the Fourier transform of the function v and $\mathbf{s}$ is a frequency domain variable.

According to [16], the optimal solution that minimizes Eqn. (2) is given by linear combination of particular kernel

functions on each instance $\mathbf{x}$, plus the term $\phi(\mathbf{z})$ in the null space of $\mathbf{D}$:

$$v(\mathbf{z}) = \sum_{i=1}^{N} \psi_i K_{\hat{\mathbf{D}}\mathbf{D}}(\mathbf{z}, \mathbf{x}_i) + \phi(\mathbf{z}) \tag{5}$$

where $v(\mathbf{z})$ stands for the displacement of an arbitrary position $\mathbf{z}$ in the same vector space. The kernel function $K_{\hat{\mathbf{D}}\mathbf{D}}$ is a Green's function of the self-adjoint operator $\hat{\mathbf{D}}\mathbf{D}$, where $\hat{\mathbf{D}}$ is the adjoint operator of $\mathbf{D}$. ψ_i (size $d \times 1$) is the weight of the kernel functions.

In this work, the *coherent point drifting* (CPD) model [14] is chosen to transform a feature space smoothly and nonlinearly. It was originally used as a registration solution to match point sets. CPD model is formulated using a particular regularization term where the kernel function K is chosen as a Gaussian low-pass filter G in Eqn. (4):

$$\mathcal{R}_{reg}[v] = \int_{\mathbb{R}^d} \frac{|\tilde{v}(\mathbf{s})|^2}{\tilde{G}(\mathbf{s})} d\mathbf{s} \tag{6}$$

This Gaussian choice is motivated by several considerations. First, the corresponding Green's function (as in Eqn. (5)) of this regularization term is also a Gaussian. Second, the Gaussian kernel is positive definite and the null space term is $\mathbf{0}$. Third, by choosing appropriately sized Gaussian functions we have the flexibility to control the ranges of the filtered frequencies and thus different amount of spatial smoothness at each data point. As the null space term of the Gaussian kernel becomes $\mathbf{0}$, the optimal solution $v(\cdot)$ for CPD is given by:

$$v(\mathbf{z}) = \sum_{i=1}^{N} \psi_i G(\mathbf{z}, \mathbf{x}_i) = \sum_{i=1}^{N} \psi_i G(||\mathbf{z} - \mathbf{x}_i||) = \sum_{i=1}^{N} \psi_i \ e^{-\frac{(\mathbf{z}-\mathbf{x}_j)^2}{2\sigma^2}} \tag{7}$$

where the Green's kernel function becomes a Gaussian $G(\cdot, \cdot)$. σ is the width of the Gaussian filter, and it controls the overall level of smoothness in the deformation field.

The matrix format of $v(\mathbf{z})$ can be written as:

$$v(\mathbf{z}) = \mathbf{\Psi} \begin{pmatrix} G(\mathbf{z}, \mathbf{x}_1) \\ \cdots \\ G(\mathbf{z}, \mathbf{x}_n) \end{pmatrix} = \mathbf{\Psi} \vec{G}(\mathbf{z}, \mathbf{x}), \tag{8}$$

where $\mathbf{\Psi}$ (size $d \times n$) is the weight matrix for the Gaussian kernel functions. n is the number of instances in $\mathbf{x}$.

In the original point-matching CPD algorithm, a uniform σ was used in deformation field estimation. However, different σ_i can be used to specify the stiffness of the deformation field around $\mathbf{x}_i$. A large σ corresponds to a smoother neighborhood and therefore a uniform deformation across a larger range. In other words, σ defines the influence power of each data point. In this work, we take advantage of this flexibility equipped in the CPD model, and assign different influence ranges to labeled and unlabeled data points. More specifically, a larger σ is assigned to labeled points allowing them to exert more significant influence in producing desired classification output. More details will be presented later in section IV-B.

IV. CPD BASED NONLINEAR DML FRAMEWORK FOR SSL

Unlike the existing semi-supervised DML methods, which are mostly focused on linear transformations, our proposed method seeks a smooth global nonlinear transformation that drives labeled and unlabeled data points together towards a better linear separability. Our solution consists of two versions: the linear model maximizes the linear separability under the original input space, and the kernel version strives for the same goal under the kernel space.

A. CPD based DML with LapSVM (CPD-LapSVM)

Laplacian SVM (LapSVM) is a popular graph-based SSL solution. Formulated based on the standard SVM models, LapSVM solves the classification problem by employing two regularization terms: one for SVM maximal margin classification, and the other for label smoothness across the graph - neighboring nodes should have identical or similar labels. LapSVM seeks to find the best class separation (maximal margin) while taking account of the graph structure that reflects the intrinsic similarities among data points.

Let $\mathcal{X} = \{\mathbf{x}_i | \ \mathbf{x}_i \in \mathbb{R}^d, i = 1, \cdots, l+u\}$ denote the whole training dataset. $\{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^{l}$ are labeled data with labels $\mathbf{y}_i \in \{-1, +1\}$, and unlabeled data instances are the remaining $\{\mathbf{x}_i\}_{i=l+1}^{l+u}$. LapSVM learns a classifier $f(\mathbf{x})$ from the training set $\mathcal{X}$, by solving the following optimization problem [5]:

$$\begin{aligned} \min_{f \in \mathcal{H}_\mathcal{K}} \quad & J = \frac{1}{l} \sum_{i=1}^{l} \xi_i + \gamma_A \|f\|_K^2 + \gamma_I \sum_{j,k=1}^{l+u} D_{jk} (f(\mathbf{x}_j) - f(\mathbf{x}_k))^2 \\ \text{s.t.} \quad & y_i f(\mathbf{x}_i) \geq 1 - \xi_i, \quad \xi_i \geq 0, \quad \forall i = 1 \dots l; \end{aligned} \tag{9}$$

where D_{ij} is the weight of the edge connecting $\mathbf{x}_i$ and $\mathbf{x}_j$ in the data adjacency graph. ξ_i is the slack variable from SVM. γ_A and γ_I are the hyper-parameters for the regularization terms. $\|f\|_K^2$ is the squared norm of f in $\mathcal{H}_\mathcal{K}$, the RKHS.

Formulation of our CPD-LapSVM Similarly as in SVML [8], our framework jointly learns a spatial transformation and a classifier at the same time. The main distinction is that the metrics learned in our model are expressed with nonlinear global deformations, regulated via the CPD model. For each data point $\mathbf{x}_i$, let $\mathbf{x}_i^0$ be its initial coordinate. Through the displacement $v(\mathbf{x})$ in Eqn. (8), $\mathbf{x}_i$ will be moved from $\mathbf{x}_i^0$ to $\mathbf{x}_i^1$:

$$\mathbf{x}_i^1 = \mathbf{x}_i^0 + v(\mathbf{x}_i^0) = \mathbf{x}_i^0 + \mathbf{\Psi} \vec{G}(\mathbf{x}_i^0, \mathbf{x}^0) \tag{10}$$

where $\mathbf{x}^0$ is the initial dataset. For each test data point $\mathbf{z}$, $\vec{G}(\mathbf{z}, \mathbf{x}^0)$ will be calculated based on Eqn. (7). The weight matrix $\mathbf{\Psi}$ captures the data abstraction from the training samples, and needs to be estimated in the training stage.

As we take LapSVM as the host algorithm, the classifier to be learned is a LapSVM classifier $f(\mathbf{x}_i) = \mathbf{w}^T \mathbf{x}_i^1 + b$ under the CPD-transformed space. The kernelized version of both LapSVM and our CPD-LapSVM can lead to nonlinear decision boundaries in the input space, though they are still hyperplanes under the kernel space. The CPD transformation can be applied in both the input space and the kernel space.

The latter is the kernel version of our CPD-LapSVM model, which will be presented later in section IV-C.

Under the input space, our linear version CPD-LapSVM (note: "linear" refers to decision boundary; CPD transformation is nonlinear) is built upon the LapSVM objective function in Eqn. (9). First, we use a quadratically smoothed hinge loss function as the slack variable item:

$$\xi_i = \max[0, 1 - y_i f(\mathbf{x}_i^1)]^2 \tag{11}$$

The choice of quadratic form is motivated by the mathematical convenience in computing the derivatives w.r.t. $f(\cdot)$ and $\mathbf{\Psi}$. Second, the squared Frobenius norm of $\mathbf{\Psi}$, denoted as $\|\mathbf{\Psi}\|_F^2$, is added to impose a smoothness constraint onto the estimated transformations. As a return, the chance of overfitting would be reduced.

With these two added terms, our linear CPD-LapSVM learns a nonlinear transformation and a linear classifier simultaneously through the minimization of the updated objective function:

$$\begin{aligned}\min_{\mathbf{\Psi},\mathbf{w},b} J = \frac{1}{l}\sum_{i=1}^{l}\max[0, 1 - y_i(\mathbf{w}^T\mathbf{x}_i^1 + b)]^2 + \gamma_A\|\mathbf{w}\|_K^2 + \gamma_L\|\mathbf{\Psi}\|_F^2 \\ + \gamma_I \sum_{j,k=1}^{l+u} D_{jk}(\mathbf{w}^T\mathbf{x}_j^1 - \mathbf{w}^T\mathbf{x}_k^1)^2 \\ \text{s.t. } y_i(\mathbf{w}^T\mathbf{x}_i^1 + b) \geq 1 - \max[0, 1 - y_i(\mathbf{w}^T\mathbf{x}_i^1 + b)]^2 \quad \forall i = 1 \dots l;\end{aligned} \tag{12}$$

where γ_A, γ_I and γ_L are trade-off hyper-parameters.

Optimization strategy The objective function of CPD-LapSVM is parameterized with a transformation matrix $\mathbf{\Psi}$ and classifier parameters $\{\mathbf{w}, b\}$. To search for an optimal solution, we adopt an EM-like iterative minimization strategy that updates $\mathbf{\Psi}$ and $\{\mathbf{w}, b\}$ alternatingly. The matrix $\mathbf{\Psi}$ is initialized with all 0 entries, so are $\mathbf{w}$ and b.

With $\mathbf{\Psi}$ fixed, Eqn. (12) reduces to the original LapSVM objective, performing on the transformed dataset $\mathbf{x}^1$. It can be easily optimized using the LapSVM solver in [5] (a standard SVM solver with the quadratic forms). With $\{\mathbf{w}, b\}$ fixed, the classification decision boundary becomes explicit. We can then further update the deformation to make the transformed dataset better conform to the membership assigned via the decision boundary. Now the objective function is only on $\mathbf{\Psi}$ (note: $\mathbf{x}_i^1$ is also a function of $\mathbf{\Psi}$, as in Eqn. (10)), reformulated as follow,

$$\begin{aligned}\min_{\mathbf{\Psi}} \ J = \frac{1}{l}\sum_{i=1}^{l}\max[0, 1 - y_i(\mathbf{w}^T\mathbf{x}_i^1 + b)]^2 \\ + \gamma_L\|\mathbf{\Psi}\|_F^2 + \gamma_I \sum_{j,k=1}^{l+u} D_{jk}(\mathbf{w}^T\mathbf{x}_j^1 - \mathbf{w}^T\mathbf{x}_k^1)^2\end{aligned} \tag{13}$$

While CPD is capable of producing rather sophisticated deformations, the smoothness term $\|\mathbf{\Psi}\|_F^2$ in this objective greatly regularizes the deformations that can be generated. In this paper, we used full graphs as the adjacency graphs, where each pair of points are connected. The edge weight D_{jk} between $\mathbf{x}_j$ and $\mathbf{x}_k$ is assigned as $D_{jk} = \exp(-\frac{1}{2\alpha^2}(\|\mathbf{x}_j^1 - \mathbf{x}_k^1\|^2))$, where α is the parameter in heat kernel function. As the objective function Eqn. (13) is differentiable w.r.t. $\mathbf{\Psi}$, a gradient based constrained optimization solver [1] is used to seek its local minima, as well as the optimal solutions of $\mathbf{\Psi}$. The gradient $\frac{\partial J}{\partial \mathbf{\Psi}}$ is given as follows:

$$\begin{aligned}\frac{\partial J}{\partial \mathbf{\Psi}} = -\frac{2}{l}\sum_{i=1}^{l}\max[0, 1 - y_i(\mathbf{w}^T\mathbf{x}_i^1 + b)]y_i\mathbf{w}\vec{G}^T(\mathbf{x}_i^0, \mathbf{x}^0) + 2\gamma_L\mathbf{\Psi} \\ + 2\gamma_I \sum_{j,k=1}^{l+u} D_{jk}(1 - \frac{1}{2\alpha^2}\|\mathbf{x}_j^1 - \mathbf{x}_k^1\|^2)(\mathbf{x}_j^1 - \mathbf{x}_k^1)(\vec{G}^T(\mathbf{x}_j^0, \mathbf{x}^0) \\ - \vec{G}^T(\mathbf{x}_k^0, \mathbf{x}^0))\mathbf{w}^\mathbf{T}\mathbf{w}\end{aligned} \tag{14}$$

The above derivations are based on a particular classifier, LapSVM. Integrating CPD with other SVM based models, e.g, TSVM, would be rather straightforward. In general, for SSL solutions formulated under an optimization framework, we can commonly utilize CPD to parameterize data points at new locations, and use the two-stage EM procedure to optimize the transformation and the classifier in an alternating fashion.

B. SSL mechanism in CPD: assign larger influence to labeled samples

As described in section III, by choosing an appropriately sized Gaussian function, we have the flexibility to control the amount of spatial smoothness around each point. In the CPD registration algorithm [14], all points were treated equally and assigned the same Gaussian width. Such uniform assignment, if adopted in SSL, would fail to stress the importance of the membership certainty residing in labeled points. To exploit such certainty, we assign wider Gaussian widths to labeled data to amplify their control ranges. In this way, more samples will move coherently along with labeled samples to make the overall data set more separable. This concept is illustrated in Fig. 1. Black dots represent unlabeled data, and red and blue dots are labeled instances with opposite labels. Labeled samples have larger influence ranges (radii of the circles) than the unlabeled.

$$G(\mathbf{x}_i, \mathbf{x}_j) = \begin{cases} e^{-\frac{(\mathbf{x}_i - \mathbf{x}_j)^2}{2\sigma_u^2}} & \forall x_i, x_j \in \text{ unlabeled data;} \\ e^{-\frac{(\mathbf{x}_i - \mathbf{x}_j)^2}{2\sigma_l^2}} & \text{otherwise.} \end{cases} \tag{15}$$

Mapping this concept to implementation, we assign different σs to the Gaussian function $G(\mathbf{x}_i, \mathbf{x}_j)$ in Eqn. (7). If $\mathbf{x}_i$ and $\mathbf{x}_j$ are both unlabeled samples, $G(\mathbf{x}_i, \mathbf{x}_j)$ is computed with a smaller width σ_u. Otherwise, a wider

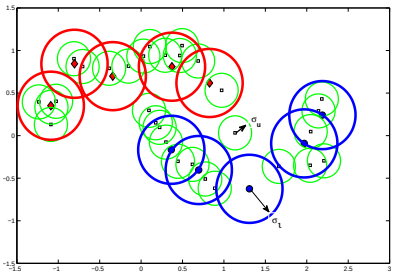


Fig. 1. Illustration of the concept of assigning different σs to labeled and unlabeled points. Refer to text for details.

[1]The sequential quadratic programming based constrained optimizer "fmincon" in Matlab Optimization Toolbox is utilized.

range σ_l is used, as shown in Eqn. (15). In this way, the constructed Green's kernel matrix in Eqn. (5) would still be maintained as symmetric.

The widths σ_u and σ_l are determined as follows. Let $d_{i\min}$ denote the distance from an instance $\mathbf{x}_i$ to its nearest neighbor. σ_u is computed as the mean of all $d_{i\min}$ values across the entire training set. σ_l is calculated as the mean of $d_{i\min}$ values for only the labeled data. Since the labeled data are more sparsely positioned compared with the whole training set, the width σ_l is always larger than σ_u. In addition, σ_l increases along with the decrease of the number of labeled samples.

C. Kernelization of CPD-LapSVM

The CPD-LapSVM model we introduced as far works under the input space. Many machine learning algorithms, including various DML solutions, can be kernelized, and the idea is to embed the input features into a higher dimensional space, with the hope that the transformed data would have certain desired property under the new domain.

SVMs and LapSVMs can be naturally kernelized as their dual formulations and solutions can both be expressed with inner products. Our CPD-LapSVM, with $\mathbf{\Psi}$ as a parameter matrix, cannot be directly kernelized the same way – computation of $\mathbf{\Psi}$ requires the location information of the transformed samples. Therefore, we adopt a kernel principal component analysis (KPCA) based framework proposed in [17]. Given a chosen kernel function, we first project the input samples into a kernel feature space introduced by KPCA. We then train the CPD-LapSVM model under the kernel space to learn both the transformation and classifier. Proven to be equivalent to the traditional kernel trick, this KPCA based framework requires no derivation of any new mathematical formula. If a low-rank KPCA is used, this approach also provides a convenient way to accelerate a learner. For more technical details, we refer readers to [17].

V. Experimental Results

We performed experiments on a synthetic dataset, seven benchmark UCI datasets, and a real world dataset for Alzheimer's Disease (AD) diagnosis. Comparisons are made with state-of-the-art SSL solutions.

A. Experiments on synthetic dataset

The first set of experiments are conducted on the two-moon synthetic dataset [2]. This dataset consists of 200 examples, equally divided into two classes (see Fig. 2). 10% of each class are chosen randomly as labeled samples in the following experiments. Both linear and kernel versions of our CPD-LapSVM were tested.

Results from linear version CPD-LapSVM This experiment is designed to show the ability of CPD-LapSVM in transforming data points for better separability under the input space. The effectiveness of assigning larger weights to labeled data for SSL is also demonstrated. Comparison is made with LapSVM, the host algorithm, to show the improvements.

[2]http://manifold.cs.uchicago.edu/manifold_regularization/data.html

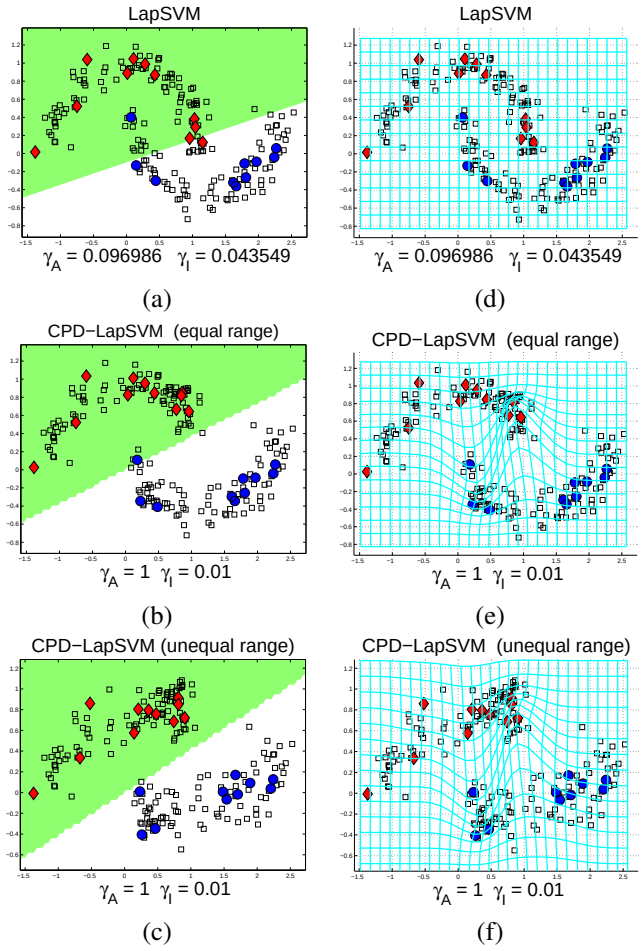


Fig. 2. First column: classification results of linear LapSVM (a), CPD-LapSVM with equal smoothness range (b), and CPD-LapSVM with larger smoothness range for labeled samples (c), respectively. Second colume (d) - (f): initial deformation field, deformation field of (b) and deformation field of (c).

Fig. 2 (a) and 2 (c) show the classification results of LapSVM and CPD-LapSVM, respectively. As a comparison, we also show in 2 (b) the result of a modified version of CPD-LapSVM, where equal Gaussian weights are assigned to both labeled and unlabeled samples. It is evident that, linear LapSVM cannot handle the non-separability in the data, while our CPD-LapSVM achieves a 100% accuracy by making the data points linearly separable through space transformation. Assigning equal width (Fig. 2.(b)) can also deform the space, but it does not work as effectively as CPD-LapSVM (Fig. 2.(c)). The corresponding deformation fields of Fig. 2 (b) and 2 (c) are shown in Fig. 2 (e) and 2 (f). From the comparison of the two fields, one can tell that the field of CPD-LapSVM appears smoother, labeled points are more linearly separated and the unlabeled points follow more closely with labeled samples. This can serve as a supporting evidence that assigning larger Gaussian width to the labeled points indeed allows them to exert amplified influences of their label certainty.

Results from kernel version CPD-LapSVM In this experiment, the two-moon dataset is used to simulate linearly

inseparable cases in the feature spaces induced by RBF kernels. Fig. 3 (a), 3 (b) and 3 (c) show the best classification results of LapSVM using RBF kernels with different widths. When an optimal or appropriate kernel width is in place, as in Fig. 3 (a), LapSVM can have the two classes well separated. However, it performs poorly when sub-optimal widths are used, as in 3 (b) and 3 (c). Finding an optimal width through cross-validation often entails a large number of width candidates and therefore many iterations. Our kernel CPD-LapSVM can greatly ease this procedure – deforming the kernel space through CPD provides a supplementary force to the RBF kernel in making the data points more linearly separable, just as it does under the input space. The effects are demonstrated in Fig. 3 (e) and 3 (f): CPD-LapSVM uses the same RBF kernels as in Fig. 3 (b) and 3 (c), but managed to obtain better classification accuracies. Due to the difficulty of visualization in high-dimensional space, the decision boundaries, which are hyperplanes under the kernel spaces, are shown under the original input space.

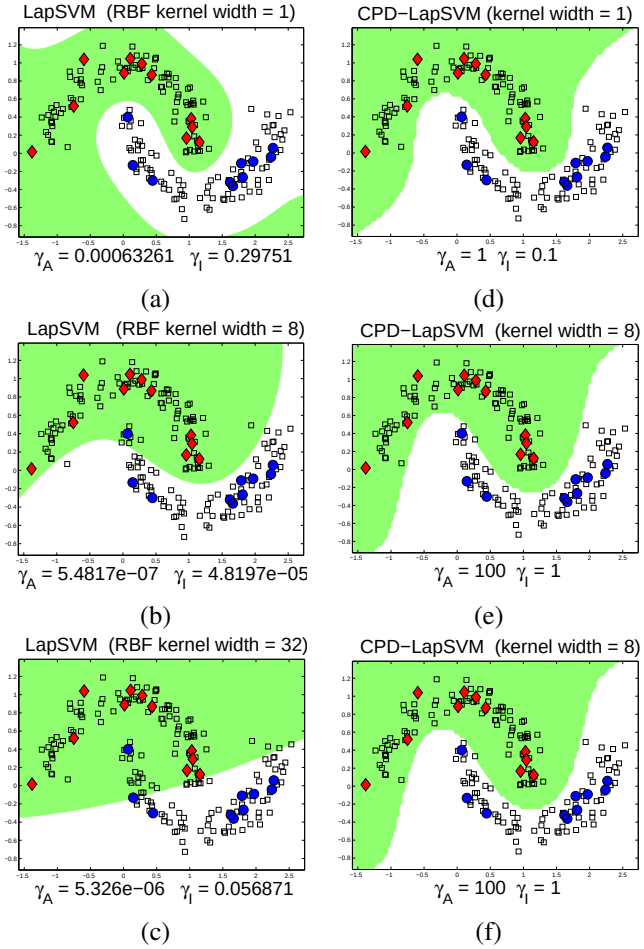


Fig. 3. First column: classification results of kernel LapSVM with RBF kernels width = 1, 8 and 32. Second column: results of kernel version CPD-LapSVM with RBF kernels width = 1, 8 and 32.

B. Experiments on UCI datasets

In this section, we employ the UCI machine learning repository datasets to evaluate our CPD-LapSVM for semi-supervised classification. Experiments are conducted to explore: 1) performance of CPD-LapSVM on datasets with a small number of labeled samples; and 2) the impact of the number of labeled samples on the classification accuracy. Seven UCI benchmark datasets were used in this study, and their basic info is summarized in Table I. All datasets have been preprocessed through normalization.

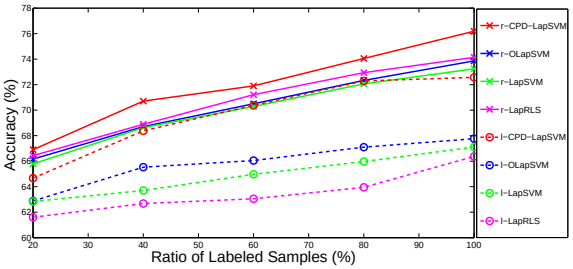
TABLE I
SEVEN UCI BENCHMARK DATASETS USED IN EXPERIMENTS. COLUMNS SHOW THE NAME, NUMBERS OF SAMPLES, ATTRIBUTES AND CLASSES OF EACH DATASET.

Datasets	# Samples	# Attributes	# Classes
Balloons	76	4	2
Haberman	306	3	2
Liver	345	6	2
Breast-cancer	286	9	2
Heart-statlog	270	13	2
Diabetes	768	8	2
Sonar	208	60	2

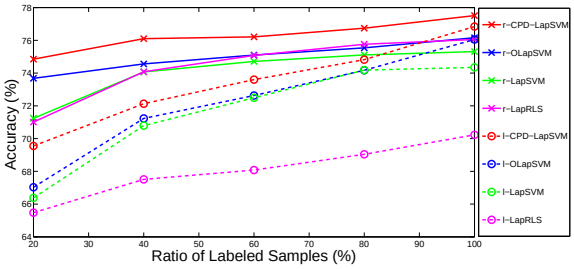
Three semi-supervised methods, Laplacian regularized least squares (LapRLS), Laplacian SVM (LapSVM) and OLapSVM [13], are utilized in all experiments as the competing solutions. These methods are tested with both linear and RBF kernels. Many SSL DML solutions have been proposed in recent years, under both SVM and k-NN paradigms. OLapSVM [13] is chosen as a comparison due to its close relevance to our solution, as it 1) also uses LapSVM as the host algorithm; 2) learns metrics to change the graph edge weights; and 3) has both linear and kernel versions.

Semi-supervised classification UCI datasets are all labeled. To simulate the SSL data scenario, 30% of the training data are randomly selected as labeled samples and the rest are treated as unlabeled. To better compare the classification performance, we run the experiment 50 times with different random 4-fold splits of each dataset, three for training and one for testing. The hyper-parameters γ_A, γ_I and γ_L are determined through cross-validation (CV) from $\{2^{-5} \sim 2^{15}\}$. γ_A and γ_I are the slackness tradeoff and graph regularization parameter used in all comparison models. γ_L is the CPD regularization parameter used only in CPD-LapSVM. All the kernel methods have an additional parameter to tune: the RBF kernel width σ, which is also chosen through CV from $\{2^{-5} \sim 2^{10}\}$ in the experiments. The mean and standard deviation of each competing algorithm are calculated over the 50 runs and summarized in Table II.

To better evaluate the relative performance of each algorithm, a pairwise Student's t-test with a p-value 0.05 is conducted among the tested methods for each dataset. Then, a schema from [18] is applied to rank the tested algorithms. Each solution is compared with all competing methods: scores 1 if it is significantly better than one opponent in statistic; 0.5 points if there is a tie (no significant difference), and 0 if it performs worse. Table II summarizes the classification accuracies and comparison scores. Highest accuracy for each dataset has been identified in Boldface. It is evident that our CPD-LapSVM outperforms all other methods with significant margins in statistic. It achieves the highest ranking score in both linear and kernel groups. It is noteworthy that the linear CPD-LapSVM obtains comparable results (score 38.5) with the



(a) Liver



(b) Breast-cancer

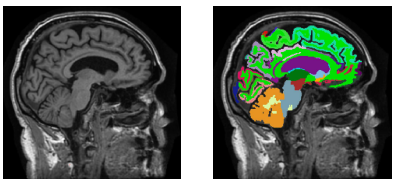
Fig. 4. Classification accuracies of tested methods w.r.t. different prevalence levels of labeled data. The prefix l denotes linear methods and r denotes kernel methods using RBF kernels.

other competing methods using RBF kernels. Furthermore, our proposed CPD-LapSVM achieves significantly improvements over the host method LapSVM.

Impact of prevalence of labeled samples We also explore the impact of varying prevalence levels of labeled samples on the performance of the tested algorithms. In this experiment, we used UCI liver and breast-cancer datasets as examples. The percentages of labeled data in training sets are set to an ascending sequence $\{20\%,\ 40\%,\ 60\%,\ 80\%\ ,\ 100\%\}$. The same experimental setting and hyper parameters selection procedure are adopted from the previous experiments. The classification results are shown in Fig. 4. Solid lines in both subfigures depict the results from kernel versions, while dash lines are for linear models. It is evident that CPD-LapSVM models consistently outperforms the other competing methods. Kernel CPD-LapSVM, the solid red lines in both subfigures, achieves the highest classification accuracies among all solutions throughout all different label prevalence levels. In addition, linear CPD-LapSVM, dash read lines, often produces comparable performance with other competing methods using RBF kernels.

C. Applications in Alzheimer's Disease diagnoses

In this section, our proposed CPD-LapSVM is applied to solve a real world problem in medicine – identify potential Alzheimer's Disease (AD) patients based on neuroimage data. AD is the most common form of dementia, affecting more than 44 million people worldwide. Mild cognitive impairment (MCI) is often considered as the early stage of AD. While approximately $5\% - 10\%$ MCI patients will develop into AD each year, the others remain in this stage and never convert. Clinically, the former is called progressive MCI (pMCI) if the conversion happens within 3 years after baseline diagnosis, and the latter is called stable MCI (sMCI). Differentiating pMCI from sMCI at the baseline time, if with high accuracy, can potentially lead to early diagnosis of AD, which is of great importance to initiate treatments early, as well as understand the disease mechanism. As MCI-to-AD conversion takes 3 years to detect, and many patients do not have follow-up diagnosis, a large number of MCI subjects are labeled as "Unknown MCI" (uMCI). With such unlabeled data, SSL would be suitable to produce better predictions.



(a) (b)

Fig. 5. (a): brain MRI of an ADNI subject. (b): anatomical segmentation of (a). Colors indicate different brain structures.

Data and setup The data used in this experiment are obtained from the Alzheimer's Disease Neuroimaging Initiative (ADNI) database [3]. Overall, 110 patients labeled as pMCI, 38 with sMCI and 94 with uMCI (242 subjects in total) are used in our experiments.

The features utilized in this study are the volumes of 113 cortical and subcortical brain structures extracted from the subjects' Magnetic Resonance Imaging (MRI) scans. The anatomical structures include left/right hippocampi, left/right caudates, etc, as shown in Fig. 5. All features have been normalized by the corresponding whole brain volumes. The same experimental setting and hyper parameters selection approach in the previous experiments are adopted here. The uMCI patients are shared as unlabeled samples over the 4-fold cross validation.

Experimental results and comparisons The classification accuracies are reported in Table III . Compared to the host LapSVM algorithm, the classification accuracies is improved in both linear and kernel versions of CPD-LapSVM. The highest accuracy, 78.27%, is produced by the kernel CPD-LapSVM. We also summarize several state-of-the-art works in MCI-to-AD prediction in Table IV. They are all SSL solutions, utilizing MRI datasets under ADNI. "Moradi *et al.* (i)" on row 4 is the solution proposed in [19] with age related effect, and "Moradi *et al.* (ii)" is the solution with age effect removed. The accuracy obtained through our CPD-LapSVM kernel version is at least 3.5% higher than all other solutions. While direct comparisons of the methods are not feasible, as different subjects, features and classifiers were used, the high accuracies from our model nevertheless can be regarded as an indirect evidence for the power of the proposed framework.

VI. Conclusions

The proposed CPD-LapSVM model learns a globally smooth nonlinear transformation to improve the performance of LapSVM classifier. CPD is used as the transformation model in part because of its inherent mechanism to assign data samples with different influence ranges. Our framework

[3] www.loni.usc.edu/ADNI

TABLE II

MEAN AND STANDARD DEVIATION OF CLASSIFICATION ACCURACIES OF EACH TESTED METHOD ON SEVEN BENCHMARK UCI DATASETS. THE PREFIX l DENOTES THE LINEAR VERSION AND r DENOTES KERNEL VERSION. BOLDFACE INDICATES THE HIGHEST CLASSIFICATION ACCURACY FOR EACH DATASET. THE LOWER NUMBER IN THE PARENTHESIS DENOTES THE RANKING SCORE OF EACH METHOD ON THE GIVEN DATASET.

Algorithms	**Balloons**	**Haberman**	**Liver**	**Breast Cancer**	**Heart Statlog**	**Diabetes**	**Sonar**	**Total Score**
l-LapRLS	82.67 ± 5.62 (2.5)	53.56 ± 7.05 (0.5)	62.10 ± 4.03 (1.0)	67.18 ± 3.79 (1.5)	78.87 ± 3.14 (2.0)	74.08 ± 2.79 (2.5)	69.01 ± 3.66 (1.0)	11
l-LapSVM	83.33 ± 5.21 (2.5)	56.57 ± 5.52 (1.5)	63.00 ± 3.64 (1.5)	68.53 ± 4.21 (2.5)	79.96 ± 2.85 (2.5)	74.19 ± 2.39 (2.5)	70.00 ± 3.48 (1.5)	14.5
l-OLapSVM	88.67 ± 4.98 (7.0)	65.15 ± 6.53 (3.5)	64.81 ± 3.07 (3.5)	69.15 ± 4.21 (3.5)	81.33 ± 2.59 (6.0)	74.97 ± 2.56 (3.5)	71.14 ± 3.34 (2.5)	29.5
l-CPD-LapSVM	86.67 ± 7.56 (5.5)	64.19 ± 6.31 (3.5)	67.81 ± 3.56 (6.0)	70.68 ± 3.70 (4.0)	$\mathbf{82.78 \pm 2.68}$ (9.0)	76.39 ± 3.45 (7.0)	72.07 ± 3.18 (3.5)	38.5
r-LapRLS	84.00 ± 6.05 (3.0)	71.21 ± 5.17 (7.5)	69.29 ± 3.37 (8.0)	72.76 ± 4.67 (7.0)	79.06 ± 3.92 (2.5)	76.78 ± 2.39 (8.0)	75.86 ± 3.62 (7.5)	43.5
r-LapSVM	87.33 ± 6.81 (7.0)	70.45 ± 5.43 (7.0)	67.71 ± 3.49 (6.0)	72.57 ± 3.29 (6.5)	80.33 ± 3.10 (4.5)	75.31 ± 2.03 (5.0)	75.59 ± 3.24 (7.0)	43
r-OLapSVM	89.33 ± 4.92 (7.5)	70.76 ± 5.33 (7.0)	68.10 ± 3.69 (6.5)	74.03 ± 3.13 (7.5)	81.41 ± 3.24 (6.0)	75.49 ± 2.37 (5.0)	75.80 ± 3.04 (7.5)	47
r-CPD-LapSVM	$\mathbf{89.67 \pm 6.44}$ (7.5)	$\mathbf{72.32 \pm 4.66}$ (7.5)	$\mathbf{69.95 \pm 2.27}$ (8.5)	$\mathbf{75.63 \pm 3.51}$ (9.0)	81.50 ± 3.09 (6.0)	$\mathbf{76.95 \pm 3.12}$ (8.0)	$\mathbf{76.71 \pm 3.05}$ (7.5)	**53.5**

TABLE III

MEAN AND STANDARD DEVIATION OF CLASSIFICATION ACCURACIES OF EACH TESTED METHOD ON ADNI MCI DATASET.

Algorithms	Linear Kernel	RBF Kernel
LapRLS	67.09 ± 1.69	76.09 ± 3.29
LapSVM	66.37 ± 1.37	76.14 ± 3.87
OLapSVM	67.40 ± 2.20	76.54 ± 3.78
CPD-LapSVM	$\mathbf{69.12 \pm 1.19}$	$\mathbf{78.27 \pm 2.82}$

TABLE IV

COMPARISONS OF CPD-LAPSVM WITH OTHER STATE-OF-THE-ART SSL SOLUTIONS FOR MCI-TO-AD PREDICTIONS USING ADNI.

Methods	ACC(%)	SEN(%)	SPE(%)
Ye *et al.*[20]	56.10	94.10	40.80
Filipovych *et al.*[21]	$--$	79.40	51.70
Moradi *et al.*[19](i)	72.60	84.16	53.66
Moradi *et al.*[19](ii)	74.74	88.85	51.46
CPD-LapSVM (RBF)	**78.27**	86.38	52.32

has broad applicability, and it can be integrated with other SSL classifiers than LapSVM. Evaluations on UCI and ADNI datasets demonstrate the efficacy of our model in solving real world problems. Exploring other types of geometric models is the direction of our future efforts. Also, we are interested in applying the proposed framework to other machine learning problems.

REFERENCES

[1] K. Nigam, A. K. McCallum, S. Thrun, and T. Mitchell, "Text classification from labeled and unlabeled documents using em," *Machine learning*, vol. 39, no. 2-3, pp. 103–134, 2000.

[2] C. Rosenberg, M. Hebert, and H. Schneiderman, "Semi-supervised self-training of object detection models," 2005.

[3] U. Brefeld, T. Gärtner, T. Scheffer, and S. Wrobel, "Efficient co-regularised least squares regression," in *ICML*, pp. 137–144, ACM, 2006.

[4] T. Cristianini, "Convex methods for transduction," *NIPS*, vol. 16, p. 73, 2004.

[5] M. Belkin and P. Niyogi, "Semi-supervised learning on riemannian manifolds," *Machine learning*, vol. 56, no. 1-3, pp. 209–239, 2004.

[6] J. Weston, F. Ratle, H. Mobahi, and R. Collobert, "Deep learning via semi-supervised embedding," in *Neural Networks*, pp. 639–655, Springer, 2012.

[7] J. Goldberger, G. E. Hinton, S. T. Roweis, and R. Salakhutdinov, "Neighbourhood components analysis," in *NIPS*, pp. 513–520, 2004.

[8] Z. Xu, K. Q. Weinberger, and O. Chapelle, "Distance metric learning for kernel machines," *arXiv preprint arXiv:1208.3422*, 2012.

[9] S. C. Hoi, W. Liu, and S.-F. Chang, "Semi-supervised distance metric learning for collaborative image retrieval and clustering," *TOMM*, vol. 6, no. 3, p. 18, 2010.

[10] P. S. Dhillon, P. P. Talukdar, and K. Crammer, "Learning better data representation using inference-driven metric learning," in *Proceedings of the ACL 2010 Conference Short Papers*, pp. 377–381, Association for Computational Linguistics, 2010.

[11] J. Yu, M. Wang, and D. Tao, "Semisupervised multiview distance metric learning for cartoon synthesis," *Image Processing, IEEE Transactions on*, vol. 21, no. 11, pp. 4636–4648, 2012.

[12] G. Niu, B. Dai, M. Yamada, and M. Sugiyama, "Information-theoretic semi-supervised metric learning via entropy regularization," *Neural computation*, vol. 26, no. 8, pp. 1717–1762, 2014.

[13] Y. Gu and K. Feng, "Optimized laplacian svm with distance metric learning for hyperspectral image classification," *Applied Earth Observations and Remote Sensing*, vol. 6, no. 3, pp. 1109–1117, 2013.

[14] A. Myronenko and X. Song, "Point set registration: Coherent point drift," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 32, no. 12, pp. 2262–2275, 2010.

[15] X. Zhu, P. Gong, Z. Zhao, and C. Zhang, "Learning similarity metric with svm," in *IJCNN*, pp. 1–8, IEEE, 2012.

[16] Z. Chen and S. Haykin, "On different facets of regularization theory," *Neural Computation*, vol. 14, no. 12, pp. 2791–2846, 2002.

[17] C. Zhang, F. Nie, and S. Xiang, "A general kernelization framework for learning algorithms based on kernel pca," *Neurocomputing*, vol. 73, no. 4, pp. 959–967, 2010.

[18] J. Wang, A. Kalousis, and A. Woznica, "Parametric local metric learning for nearest neighbor classification," in *NIPS*, pp. 1601–1609, 2012.

[19] E. Moradi, A. Pepe, C. Gaser, H. Huttunen, J. Tohka, A. D. N. Initiative, *et al.*, "Machine learning framework for early mri-based alzheimer's conversion prediction in mci subjects," *NeuroImage*, vol. 104, pp. 398–412, 2015.

[20] D. H. Ye, K. M. Pohl, and C. Davatzikos, "Semi-supervised pattern classification: application to structural mri of alzheimer's disease," in *PRNI*, pp. 1–4, IEEE, 2011.

[21] R. Filipovych, C. Davatzikos, A. D. N. Initiative, *et al.*, "Semi-supervised pattern classification of medical images: application to mild cognitive impairment (mci)," *NeuroImage*, vol. 55, no. 3, pp. 1109–1119, 2011.