EXPLOITING GRAPH EMBEDDING IN SUPPORT VECTOR MACHINES

Georgios Arvanitidis and Anastasios Tefas

Aristotle University of Thessaloniki, Depeartment of Informatics
Box 451, 54124 Thessaloniki, Greece
Email: tefas@aiia.csd.auth.gr

ABSTRACT

In this paper we introduce a novel classification framework that is based on the combination of the support vector machine classifier and the graph embedding framework. In particular we propose the substitution of the support vector machine kernel with sub-space or sub-manifold kernels, that are constructed based on the graph embedding framework. Our technique combines the very good generalization ability of the support vector machine classifier with the flexibility of the graph embedding framework resulting in improved classification performance. The attained experimental results on several benchmark and real-life data sets, further support our claim of improved classification performance.

Index Terms— Support Vector Machines, Graph Embedding, Laplacian Matrix

1. INTRODUCTION

In classification, the main objective is to train a classifier, which generalizes well on unseen data samples. Several classifiers have been proposed in the literature. However, the *Support Vector Machines (SVM)* has been the most popular due to its state-of-the-art performance and its generalization capabilities.

The standard SVM is a binary classifier that tries to find a hyperplane that separates two classes of data points. The resulting decision surface has the maximum margin between the two classes. The margin is defined as the distance between the hyperplane and the samples that lie closest to this hyperplane. A quadratic convex optimization problem is formulated, which can be solved optimally. Whereas the SVMs are designed for linear classification problems, they can also construct a non-linear decision surface. This is achieved by mapping the initial points into a (usually higher or even infinite) dimensional *Hilbert space* through a mapping $\phi : \mathcal{X} \rightarrow \mathcal{H}$, where $\mathcal{X}$ is the original domain and $\mathcal{H}$ is the Hilbert space. In this high dimensional space, the data points can be separable and the maximum margin hyperplane can be found. The decision surface can be found without having to explicitly compute the mapping function ϕ, but only by computing dot products in the Hilbert space by means of the *kernel trick* [1].

Lately, the Graph Embedding Framework [2] has been proposed. Under the Graph Embedding Framework many dimensionality reduction algorithms such as Principal Component Analysis [3] and Linear Discriminant Analysis [4] can be formulated as graph relationships and thus, resulting in solving generalized eigenvalue problems.

In this paper, we combine the advantages of the SVM formulation with properties of the graphs that represent the various dimensionality reduction methods. A new SVM formulation is derived that corresponds to a standard SVM with a novel *precomputed kernel*. The resulting *Graph Embedded SVM (GESVM)* is shown to obtain competitive results with comparison to the standard Linear and RBF SVMs in several benchmark datasets.

Our paper is organized as follows. In Section 2, we make a short introduction to the SVM classifier as well as the Graph Embedding Framework. In Section 3 we describe in detail our proposed Graph Embedded SVM method. Experimental results are presented in Section 4. Finally, in Section 5 we conclude our work.

2. PRIOR WORK

2.1. Support Vector Machines

The *Support Vector Machine* is a binary classifier, which finds a hyperplane that has the maximum margin between the two classes. In most of the problems the classes are not linearly separable, so the SVM tries to find the hyperplane that has the maximum margin but also that minimizes the training error. Suppose that we have a binary problem with a data set $\{\mathbf{x}_i, y_i\}_{i=1}^n$ where n is the number of samples and the labels of each sample $y_i \in \{-1, +1\}$. Therefore, we can formulate the hyperplane as:

$$\begin{aligned} &\mathbf{w}^\top \mathbf{x} - b = 0 \\ \text{s.t. } &y_i(\mathbf{w}^\top \mathbf{x}_i - b) \geq 1 - \xi_i \\ &\xi_i \geq 0, \quad i = 1, \ldots, n \end{aligned} \tag{1}$$

where $\mathbf{w}$ is a vector, perpendicular to the hyperplane, b is the offset and ξ_i the penalty for the miss-classification, if $\xi_i > 1$, then $\mathbf{x}_i$ is not on the correct side of the separating plane. The margin between the two classes is equal to $\frac{2}{\|\mathbf{w}\|}$ so the problem is to maximize this margin which is equivalent to minimize $\frac{\mathbf{w}^\top \mathbf{w}}{2}$, so the problem is formulated as:

$$\begin{aligned} &\min_{\mathbf{w},b,\xi} \frac{1}{2}\mathbf{w}^\top \mathbf{w} + C\sum_{i=1}^{n} \xi_i \\ &\text{s.t. } y_i(\mathbf{w}^\top \mathbf{x}_i - b) \geq 1 - \xi_i \\ &\quad \xi_i \geq 0, \quad i = 1, \dots, n \end{aligned} \tag{2}$$

where C is a penalty parameter. Moreover, this problem can be solved using Lagrange multipliers and using the KKT conditions we obtain:

$$\mathbf{w} = \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i \tag{3}$$

where α_i are the Lagrange multipliers and most of them are equal to zero. The samples $\mathbf{x}_i$ that do not have zero Lagrange multipliers are called *support vectors*.

The dual problem is formulated as:

$$\begin{aligned} &\max_{\boldsymbol{\alpha}} \boldsymbol{\alpha}^\top \mathbf{e} - \frac{1}{2}\boldsymbol{\alpha}^\top \mathbf{Q}\boldsymbol{\alpha} \\ &\text{s.t. } 0 \leq \alpha_i \leq C, \quad i = 1, \dots, n \\ &\mathbf{y}^\top \boldsymbol{\alpha} = 0 \end{aligned} \tag{4}$$

where $Q_{ij} = y_i k(\mathbf{x}_i, \mathbf{x}_j) y_j$ and $\mathbf{e} = [1, \dots, 1]^\top$. The kernel $k(\mathbf{x}_i, \mathbf{x}_j)$ is defined as $k(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T \mathbf{x}_j$ in the linear case. By exploiting the kernel trick [1] we can use a non-linear function $\phi(\mathbf{x}_i)$ to represent the samples in a higher dimensional space where they can be linearly separable. As a result the solution of (4) finds the optimal linear hyper plane in the high dimensional Hilbert space that corresponds to a non-linear surface in the initial space.

2.2. Graph Embedding

In this section we provide a short description of the *Graph Embedding* framework [2]. Prior work has shown that many dimensionality reduction algorithms can be integrated into this framework. The Graph Embedding framework is based on the introduction of the undirected weighted graph $\mathbf{G} = (\mathbf{X}, \mathbf{W})$, whose vertex set consists of the data matrix $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_n]^\top \in \mathbb{R}^{n \times d}$, and the similarity matrix $\mathbf{W} \in \mathbb{R}^{n \times n}$, whose entries can be positive, negative or zero. The graph embedding of the graph $\mathbf{G}$ is, therefore, an algorithm to find the low dimensional representation of the data that best preserves the relationships between the vertex pairs of $\mathbf{G}$. The graph $\mathbf{G}$ can be seen as an intrinsic graph. Furthermore, a penalty graph $\mathbf{G}^p = (\mathbf{X}, \mathbf{W}^p)$ can also be defined, whose corresponding weight matrix penalizes specific characteristics of the relationships between the data points. For projections to one dimension, assuming that $\mathbf{z} = [z_1, \dots, z_n]^\top$ is the vector of the projections of each data sample $\mathbf{x}_i$, the graph objective function to be optimized is:

$$\begin{aligned} \mathbf{z}^* &= \arg \min_{\mathbf{z}^\top \mathbf{C}\mathbf{z}=c} \sum_{i,j=1}^{n} \|z_i - z_j\|^2 W_{ij} \\ &= \arg \min_{\mathbf{z}^\top \mathbf{C}\mathbf{z}=c} \mathbf{z}^\top \mathcal{L}\mathbf{z} \end{aligned} \tag{5}$$

where $\mathcal{L}$ is the graph Laplacian defined as $\mathcal{L} = \mathbf{D} - \mathbf{W}$ and $\mathbf{D}$ is the diagonal degree matrix defined as $D_{ii} = \sum_{j=1}^{n} W_{ij},\ i = 1, \dots, n$. $\mathbf{C}$ is a constraint matrix to avoid trivial solutions and is typically a diagonal matrix for scale normalization, or the graph Laplacian of $\mathbf{G}^p$, that is $\mathbf{C} = \mathbf{L}^p = \mathbf{D}^p - \mathbf{W}^p$ and c is a constant. If we assume that the vector $\mathbf{z}$ is a result of the linear projection $\mathbf{z} = \mathbf{X}\mathbf{w}$, where $\mathbf{w} \in \mathbb{R}^d$ is the projection vector, then the objective to be optimized becomes

$$\begin{aligned} \mathbf{w}^* &= \arg \min_{\substack{\mathbf{w}^\top \mathbf{X}^\top \mathbf{C}\mathbf{X}\mathbf{w}=c \\ \text{or } \mathbf{w}^\top \mathbf{w}=c}} \sum_{i,j=1}^{n} \|\mathbf{w}^\top \mathbf{x}_i - \mathbf{w}^\top \mathbf{x}_j\|^2 W_{ij} \\ &= \arg \min_{\substack{\mathbf{w}^\top \mathbf{X}^\top \mathbf{C}\mathbf{X}\mathbf{w}=c \\ \text{or } \mathbf{w}^\top \mathbf{w}=c}} \mathbf{w}^\top \mathbf{X}^\top \mathcal{L}\mathbf{X}\mathbf{w} \end{aligned} \tag{6}$$

The above objective can be extended to non-linear projections by using the kernel trick [1]. The input data are mapped to a high dimensional Hilbert space $\mathcal{H}$ using a map $\phi : \mathbf{x} \rightarrow \mathcal{H}$. The projection vector takes the form $\mathbf{w} = \sum_{i=1}^{n} \alpha_i \phi(\mathbf{x}_i)$. By defining the kernel matrix $\mathbf{K} \in \mathbb{R}^{n \times n}$ as $K_{ij} = \phi(\mathbf{x}_i)^\top \phi(\mathbf{x}_j)$, the objective in (6) can be written as

$$\begin{aligned} \boldsymbol{\alpha}^* &= \arg \min_{\substack{\boldsymbol{\alpha}^\top \mathbf{K}^\top \mathbf{C}\mathbf{K}\boldsymbol{\alpha}=c \\ \text{or } \boldsymbol{\alpha}^\top \mathbf{K}\boldsymbol{\alpha}=c}} \sum_{i,j=1}^{n} \|\boldsymbol{\alpha}^\top \mathbf{k}_i - \boldsymbol{\alpha}^\top \mathbf{k}_j\|^2 W_{ij} \\ &= \arg \min_{\substack{\boldsymbol{\alpha}^\top \mathbf{K}^\top \mathbf{C}\mathbf{K}\boldsymbol{\alpha}=c \\ \text{or } \boldsymbol{\alpha}^\top \mathbf{K}\boldsymbol{\alpha}=c}} \boldsymbol{\alpha}^\top \mathbf{K}^\top \mathcal{L}\mathbf{K}\boldsymbol{\alpha} \end{aligned} \tag{7}$$

where $\mathbf{k}_i$ is the i-th row of matrix $\mathbf{K}$.

The solutions of (5), (6) and (7) can be obtained by solving the generalized eigenvalue problem

$$\mathbf{A}\mathbf{v} = \lambda \mathbf{B}\mathbf{v} \tag{8}$$

where $\mathbf{A} = \mathcal{L}, \mathbf{X}^\top \mathcal{L}\mathbf{X}, \mathbf{K}^\top \mathcal{L}\mathbf{K}$ and $\mathbf{B} = \mathbf{I}, \mathbf{C}, \mathbf{X}^\top \mathbf{C}\mathbf{X}, \mathbf{K}, \mathbf{K}^\top \mathbf{C}\mathbf{K}$, depending on the type of the problem defined.

As has been showed in [2], there are several dimensionality reduction algorithms that can be reformulated within the graph embedding framework. The computation of the similarity matrix $\mathbf{W}$ is different for each of these algorithms. We will review these algorithms and show how they can be combined with the SVMs.

2.3. Dimensionality reduction algorithms formulation.

Principal Component Analysis (PCA) [3] is an algorithm that transforms the data into a new coordinate system such that the projected samples have maximum variance. Moreover, PCA finds and removes the projection direction with the minimum variance, that is

$$\mathbf{w}^* = \arg \min_{\mathbf{w}^\top \mathbf{w}=1} \mathbf{w}^\top \mathbf{C}\mathbf{w} \tag{9}$$

where

$$\mathbf{C} = \frac{1}{n}\sum_{i=1}^{n}(\mathbf{x}_i - \bar{\mathbf{x}})(\mathbf{x}_i - \bar{\mathbf{x}})^\top = \frac{1}{n}\mathbf{X}(\mathbf{I} - \frac{1}{n}\mathbf{e}\mathbf{e}^\top)\mathbf{X}^\top \tag{10}$$

In the above, $\bar{\mathbf{x}}$ is the mean of all samples, $\mathbf{C}$ is the covariance matrix, $\mathbf{I}$ is an identity matrix and $\mathbf{e}$ is an n-dimensional vector of all ones.

Linear Discriminant Analysis (LDA) [4] is used to find the most discriminative projection directions that effectively separate two or more classes. LDA uses two scatter matrices, the within-class scatter $\mathbf{S}_W$ and the between-class scatter $\mathbf{S}_B$, that is

$$\begin{aligned} \mathbf{S}_W &= \mathbf{X}(\mathbf{I} - \sum_{k=1}^{c}\frac{1}{n_k}\mathbf{e}^k\mathbf{e}^{k\top})\mathbf{X}^\top \\ \mathbf{S}_B &= \mathbf{X}(\sum_{k=1}^{c}\frac{1}{n_k}\mathbf{e}^k\mathbf{e}^{k\top} - \frac{1}{n}\mathbf{e}\mathbf{e}^\top)\mathbf{X}^\top \end{aligned} \tag{11}$$

In the above, $\mathbf{e}^c$ is an n-dimensional vector with $e_i^k = 1$ if $k = k_i$, 0 otherwise. The objective minimized by LDA is the ratio between the within-class scatter and the between-class scatter.

Locally Linear Embedding (LLE) [5] finds a low dimensional representation of the data in which the relationship between the neighboring samples is preserved. Each data sample is reconstructed only by its K neighbors. The obtained reconstruction coefficient matrix $\mathbf{M}$ is calculated as it has been shown in [2]. LLE follows the Graph Embedding formulation with similarity matrix $\mathbf{W} = \mathbf{M} + \mathbf{M}^\top + \mathbf{M}^\top\mathbf{M}$ and penalty matrix the identity. We define the corresponding Laplacian as $\mathcal{L}_L = \mathbf{D} - \mathbf{W}$ where $D_{ii} = \sum_{j=1}^{n} W_{ij}$.

Laplacian Eigenmaps (LE) [6] find a low dimensional representation in which the similarities between neighboring points are preserved. By defining a K-NN graph with weight matrix entries

$$W_{ij} = \exp(-\|\mathbf{x}_i - \mathbf{x}_j\|^2/t) \text{ if } i \in N_K(j) \text{ or } j \in N_K(i) \tag{12}$$

we construct the corresponding Laplacian matrix $\mathcal{L}_E = \mathbf{D} - \mathbf{W}$ where $D_{ii} = \sum_{j=1}^{n} W_{ij}$. LE solves the generalized eigenvalue problem

$$\mathcal{L}\mathbf{u} = \lambda\mathbf{D}\mathbf{u} \tag{13}$$

It is obvious that LE follows the Graph Embedding framework with similarity matrix $\mathbf{W}$ defined in (12) and $\mathbf{D}$ as the penalty matrix.

In Table 1 we summarize the similarity and penalty matrices that should be used in the graph embedding framework in order to implement all the previous algorithms.

Table 1. Laplacian Matrices Definitions

Algorithm	$\mathbf{W}$ & $\mathbf{B}$ Definition
PCA	$W_{ij} = 1/n; \mathbf{B} = \mathbf{I}$
LDA	$W_{ij} = \delta_{k_i,k_j}/n_k; \mathbf{B} = \mathbf{I} - 1/n\mathbf{e}\mathbf{e}^\top$
LLE	$\mathbf{W} = \mathbf{M} + \mathbf{M}^\top + \mathbf{M}^\top\mathbf{M}; \mathbf{B} = \mathbf{I}$
LE	$W_{ij} = \exp(-\|\mathbf{x}_i - \mathbf{x}_j\|^2/t)$ if $i \in N_K(j)$ or $j \in N_K(i)$; $\mathbf{B} = \mathbf{D}$

3. GRAPH EMBEDDED SUPPORT VECTOR MACHINES (GESVM)

The Graph Embedded SVM is a method that combines the Graph Embedding framework with the SVM classifier. In this Section we describe our proposed method in detail.

3.1. Proposed method

We assume that we have n data points assembled in a matrix $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_n]^\top \in \mathbb{R}^{n\times d}$ with label vector $\mathbf{y} = [y_1, \ldots, y_n]^\top$, where $y_i \in \{-1, +1\}$, $i = 1, \ldots, n$. As mentioned before, the graph embedding framework enables the reformulation of existing dimensionality reduction algorithms into a unified view and allows the desin of novel algorithms. Each method is associated with a corresponding laplacian matrix $\mathcal{L}$. Different laplacian matrices correspond to different approaches which optimize specific properties of the low dimensional representation of the initial data points.

Following the ideas proposed in [7], [8] for semi-supervised learning, we propose a novel supervised regularization term defined as

$$\|\mathbf{f}\|_I^2 = \sum_{i,j=1}^{n} (f(\mathbf{x}_i) - f(\mathbf{x}_j))W_{ij} = \mathbf{f}^\top\mathcal{L}\mathbf{f} \tag{14}$$

where $\mathbf{f} = [f(\mathbf{x}_i), \ldots, f(\mathbf{x}_n)]^\top$ is the vector containing the values of the function f on the data points. $\|\mathbf{f}\|^2$ can be considered as a smoothness term that corresponds to the marginal distribution of the data $\mathbf{X}$. Instead of using the Laplacian only for encoding the unsupervised graph structure of the data as proposed in [7], [8] for semi-supervised learning, we use a fully supervised framework. That is, we exploit the Laplacian Matrix in order to represent several Dimensionality Reduction Criteria that enhance discrimination and/or generalization ability as it will be explained in the following.

Considering a linear SVM, where the decision function takes the form

$$f(\mathbf{x}) = \mathbf{w}^\top \mathbf{x} - b$$

the penalizer is written as

$$\|\mathbf{f}\|_I^2 = \mathbf{w}^\top \mathbf{X}^\top \mathcal{L} \mathbf{X} \mathbf{w}$$

Therefore, we can define a supervised SVM as follows [7]:

$$\begin{aligned} &\min_{\mathbf{w},b,\xi} \frac{1}{2}\|\mathbf{w}\|_2^2 + C\sum_{i=1}^{n} \xi_i + \frac{\lambda}{2}\mathbf{w}^\top \mathbf{X}^\top \mathcal{L}\mathbf{X}\mathbf{w} \\ &\text{s.t. } y_i(\mathbf{w}^\top \mathbf{x}_i - b) \geq 1 - \xi_i, \quad i = 1, \ldots, n \\ &\quad \xi_i \geq 0, \quad i = 1, \ldots, n \end{aligned} \tag{15}$$

where λ is a trade off parameter between the two regularization terms of $\mathbf{w}$ satisfying $\lambda \geq 0$. The dual problem is formulated as [7]:

$$\begin{aligned} &\max_{\boldsymbol{\alpha}} \boldsymbol{\alpha}^\top \mathbf{e} - \frac{1}{2}\boldsymbol{\alpha}^\top \mathbf{Y}\mathbf{X}(\mathbf{I} + \lambda \mathbf{X}^\top \mathcal{L}\mathbf{X})^{-1}\mathbf{X}^\top \mathbf{Y}\boldsymbol{\alpha} \\ &\text{s.t. } \boldsymbol{\alpha}^\top \mathbf{y} = 0 \\ &\quad 0 \leq \alpha_i \leq C \end{aligned} \tag{16}$$

where $\mathbf{I} \in \mathbb{R}^{d \times d}$ is the identity matrix and $\mathbf{Y}$ is defined as diagonal matrix with $diag(\mathbf{Y}) = \mathbf{y}$. For simplicity of notation we define the matrix $\mathbf{A} = (\mathbf{I} + \lambda \mathbf{X}^\top \mathcal{L}\mathbf{X})$. The dual problem (16) corresponds to an SVM with a *linear precomputed kernel* defined as

$$\mathbf{Q}_0 = \mathbf{X}\mathbf{A}^{-1}\mathbf{X}^\top \tag{17}$$

This way we integrate assumptions about the underlying graph structure of the data or the desired discriminant graph embedding into the kernel matrix of the SVM.

The above concept can be easily extended into the nonlinear case using the *kernel trick* [1]. The kernel matrix can be written as $\mathbf{K} = \mathbf{\Phi}\mathbf{\Phi}^\top$, where $\mathbf{\Phi} \in \mathbb{R}^{n \times m}$ is the matrix of the mapped data points $\mathbf{X} \in \mathcal{X}$ into a Hilbert space $\mathcal{H}$ through the mapping $\phi : \mathcal{X} \to \mathcal{H}$ and m is the unknown dimensionality of the feature space. For non-linear projections, we define the projection vector $\mathbf{w} \in \mathbb{R}^m$. Since this vector belong to $\mathbb{R}^m$ (the column space of $\mathbf{\Phi}$), we can restrict this vector to be in the range of $\mathbf{\Phi}$. Therefore, $\mathbf{w}$ can be represented as

$$\mathbf{w} = \sum_{i=1}^{n} \alpha_i \phi(\mathbf{x}_i) = \mathbf{\Phi}^\top \boldsymbol{\alpha} \tag{18}$$

The decision function of a non-linear SVM has the form:

$$f(\mathbf{x}) = \mathbf{w}^\top \phi(\mathbf{x}) - b = \boldsymbol{\alpha}^\top \mathbf{k}_i - b \tag{19}$$

where $\mathbf{k}_i$ is the i-th column of the kernel matrix. The corresponding regularization term becomes now:

$$\|\mathbf{f}\|_I^2 = \boldsymbol{\alpha}^\top \mathbf{K}\mathcal{L}\mathbf{K}\boldsymbol{\alpha} \tag{20}$$

then the supervised SVM can be defined as:

$$\begin{aligned} &\min_{\boldsymbol{\alpha},b,\xi} \sum_{i=1}^{n} \xi_i + \boldsymbol{\alpha}^\top \mathbf{K}\boldsymbol{\alpha} + \lambda \boldsymbol{\alpha}^\top \mathbf{K}\mathcal{L}\mathbf{K}\boldsymbol{\alpha} \\ &\text{s.t. } y_i(\boldsymbol{\alpha}^\top \mathbf{k}_i - b) \geq 1 - \xi_i, \quad i = 1, \ldots, n \\ &\quad \xi_i \geq 0, \quad i = 1, \ldots, n \end{aligned} \tag{21}$$

using the Lagrangian multipliers β_i, ζ_i:

$$\begin{aligned} &L(\boldsymbol{\alpha}, \xi_i, b, \boldsymbol{\beta}, \boldsymbol{\zeta}) = \\ &\quad = \sum_{i=1}^{n} \xi_i + \frac{1}{2}\boldsymbol{\alpha}^\top (2\mathbf{K} + 2\lambda \mathbf{K}\mathcal{L}\mathbf{K}\boldsymbol{\alpha}) \\ &\quad - \sum_{i=1}^{n} \beta_i (y_i(\boldsymbol{\alpha}^\top \mathbf{k}_i - b) - 1 + \xi_i) - \sum_{i=1}^{n} \zeta_i \xi_i \end{aligned} \tag{22}$$

and by the KKT conditions we get:

$$\begin{aligned} \frac{\partial L}{\partial b} &= 0 \Longrightarrow \sum_{i=1}^{n} \beta_i y_i = 0 \\ \frac{\partial L}{\partial \xi_i} &= 0 \Longrightarrow 1 - \beta_i - \zeta_i = 0 \\ &\Longrightarrow 0 \leq \beta_i \leq 1 \end{aligned}$$

Then the reduced Lagrange problem is formulated as:

$$\begin{aligned} &L'(\boldsymbol{\alpha}, \boldsymbol{\beta}) = \\ &\quad = \frac{1}{2}\boldsymbol{\alpha}^\top (2\mathbf{K} + 2\lambda \mathbf{K}\mathcal{L}\mathbf{K})\boldsymbol{\alpha} - \sum_{i=1}^{n} \beta_i y_i (\boldsymbol{\alpha}^\top \mathbf{k}_i - 1) \\ &\quad = \frac{1}{2}\boldsymbol{\alpha}^\top (2\mathbf{K} + 2\lambda \mathbf{K}\mathcal{L}\mathbf{K})\boldsymbol{\alpha} - \boldsymbol{\alpha}^\top \mathbf{K}\mathbf{Y}\boldsymbol{\beta} + \mathbf{e}^\top \boldsymbol{\beta} \end{aligned} \tag{23}$$

Taking derivative of the reduced Lagrangian:

$$\frac{\partial L'}{\partial \boldsymbol{\alpha}} = (2\mathbf{K} + 2\lambda \mathbf{K}\mathcal{L}\mathbf{K})\boldsymbol{\alpha} - \mathbf{K}\mathbf{Y}\boldsymbol{\beta} \tag{24}$$

This implies that the unknown parameteres $\boldsymbol{\alpha}$ are given by:

$$\boldsymbol{\alpha}^* = (2\mathbf{I} + 2\lambda \mathcal{L}\mathbf{K})^{-1}\mathbf{Y}\boldsymbol{\beta}^* \tag{25}$$

Table 2. Rrecomputed Kernels $\mathbf{Q_1}$ definition

Kernel	$\mathbf{Q_1}$
$\mathbf{Q}_C$	$\mathbf{K}(2\mathbf{I} + 2\lambda(\mathbf{I} - \frac{1}{n}\mathbf{e}\mathbf{e}^\top)\mathbf{K})^{-1}$
$\mathbf{Q}_W$	$\mathbf{K}(2\mathbf{I} + 2\lambda(\mathbf{I} - \sum_{k=1}^{c} \frac{1}{n_k}\mathbf{e}^k\mathbf{e}^{k\top})\mathbf{K})^{-1}$
$\mathbf{Q}_B$	$\mathbf{K}(2\mathbf{I} + 2\lambda(\sum_{k=1}^{c} \frac{1}{n_k}\mathbf{e}^k\mathbf{e}^{k\top} - \frac{1}{n}\mathbf{e}\mathbf{e}^\top)\mathbf{K})^{-1}$
$\mathbf{Q}_L$	$\mathbf{K}(2\mathbf{I} + 2\lambda\mathcal{L}_L\mathbf{K})^{-1}$
$\mathbf{Q}_E$	$\mathbf{K}(2\mathbf{I} + 2\lambda\mathcal{L}_E\mathbf{K})^{-1}$

where the $\boldsymbol{\beta}^*$ can be found from the dual problem which is formulated as [8]:

$$\begin{aligned} &\max_{\boldsymbol{\beta}} \boldsymbol{\beta}^\top\mathbf{e} - \frac{1}{2}\boldsymbol{\beta}^\top\mathbf{Y}\mathbf{K}(2\mathbf{I} + 2\lambda\mathcal{L}\mathbf{K})^{-1}\mathbf{Y}\boldsymbol{\beta} \\ &\text{s.t. } \boldsymbol{\beta}^\top\mathbf{y} = 0 \\ &\quad 0 \leq \beta_i \leq C \end{aligned} \tag{26}$$

By defining the matrix $\mathbf{B} = (2\mathbf{I} + 2\lambda\mathcal{L}\mathbf{K})$, with $\mathbf{I} \in \mathbb{R}^{n\times n}$ the identity matrix, the new dual problem corresponds to a quadratic optimization problem with a *non-linear precomputed kernel* defined as:

$$\mathbf{Q}_1 = \mathbf{K}\mathbf{B}^{-1} \tag{27}$$

Fast algorithms for solving the above problem in the primal space and at the dual have been recently proposed in [9]. In Table 2 we show the formulation of the $\mathbf{Q}_1$ precomputed kernels using various graph embeddings presented in Section 2.2. The above solution produces a non-linear SVM where all the samples are support vectors. In order to overcome this issue that renders the proposed non-linear solution computationally expensive we propose in the following the use of the SVD for decomposing the kernel matrix.

The effect of the $\mathcal{L}$ matrix on the data is different for the various graph embeddings. The SVM tries to maximize the margin between the two classes, but the regularization parameter inserts an additional optimization problem that is defined by the Laplacian matrix.

The PCA algorithm, projects the data in order to maximize their variance, but the (15) and (21) are defined as minimization problems so the $\mathbf{Q}_C$ precomputed kernel instead of maximization, minimizes the variance of the projected samples keeping them well-separated due to the SVM constraints.

The matrix $\mathbf{S}_W$ minimizes the interclass variance and $\mathbf{S}_B$ maximizes the distance between the classes, so as before, $\mathbf{Q}_W$ precomputed kernel tries to minimize the interclass sparseness and the $\mathbf{Q}_B$ to minimize the distance between the classes centers respectively, under the constraints of class separability given by the SVM formulation.

Furthermore, the algorithms LLE and Laplacian Eigenmaps preserve similarities between the samples, by maximizing the nearest neighbor density. However, the $\mathbf{Q}_L$ and $\mathbf{Q}_E$ kernels due to the minimization as mentioned before, try to maximize the similarity inside the graph structure under the separability constraints of SVM.

Table 3. Characteristics of Benchmark Datasets

Dataset	**Library**	**Samples**	**Attributes**	**Classes**
Australian	Statlog	690	14	2
Breast Cancer	UCI	683	10	2
Diabetes	UCI	768	8	2
German	UCI	1000	24	2
Heart	Statlog	270	13	2
Hepatitis	UCI	155	19	2
Sonar	UCI	208	60	2
Ionosphere	UCI	351	34	2
Liver	UCI	345	6	2
Transfusion	UCI	768	8	2

In conclusion, the proposed approach is a general framework that allows for several different criteria on the graph structure to be incorporated in the SVM solution. These criteria should enforce similarity inside the classes and dissimilarity between the classes which is already implied in terms of margin maximization by the SVM constraints.

4. EXPERIMENTAL RESULTS

4.1. Benchmark Datasets

We compared our proposed GESVMs against the standard Linear and RBF SVMs using 10 benchmark datasets from the UCI and Statlog (`http://archive.ics.uci.edu/ml/`) repositories. The characteristics of each dataset can be seen in Table 3.

All the features of each dataset were scaled to the interval $[-1, +1]$. To evaluate the test error we used 5-fold Cross Validation. Additionaly, an inner 5-fold cross validation loop is performed on the training set of the external fold to select the optimal regularization parameter λ from the grid $\{0.1, 0.2, \ldots, 10\}$ with step of 0.1. The cost parameter of the SVM was set in all cases to $C = 100$ and for the case of the RBF SVM the width of the kernel was set to $\gamma = 1$. In Table 4 we present our experimental results for the linear SVM for several graph embedding kernels and in Table 5 for the RBF SVM with the various graph embedding non linear kernels.

From the results we can observe that GESVM obtain in most of the cases better classification accuracy than the standard SVM classifier. For the linear case, in 7 out of 10 datasets the proposed approach gives better performance and in one case the linear SVM has better performance. However, there is no clear winner among the different graph embedding methods. This can be attributed to the internal graph structure of each dataset.

In the non-linear case with the $\mathbf{Q}_1$ precomputed kernel, we can see that the graph embedding methods, win 8 out of 10 datasets. Also here there is not a clear winner among the methods, but we can observe that the $\mathbf{Q}_W$ precomputed kernel has better classification accuracy than the classical SVM

Table 4. Benchmark results for the linear case with $\mathbf{Q_0}$ precomputed kernel.

	Linear SVM	$\mathbf{Q}_W$	$\mathbf{Q}_B$	$\mathbf{Q}_C$	$\mathbf{Q}_L$	$\mathbf{Q}_E$
australian	85.36(1.63)%	**85.51(1.58)%**	85.36(1.63)%	**85.51(1.58)%**	84.64(2.52)%	**85.51(1.58)%**
breast	**96.93(0.60)%**	96.78(0.64)%	96.78(0.64)%	96.78(0.83)%	96.78(0.66)%	96.78(0.64)%
diabetes	76.82(2.97)%	**77.21(2.62)%**	76.95(2.39)%	76.69(2.62)%	76.82(2.97)%	76.82(3.13)%
german	76.20(1.60)%	**77.20(1.10)%**	77.10(0.96)%	76.80(1.44)%	76.70(1.48)%	76.40(1.02)%
heart	**83.33(5.40)%**	82.96(6.33)%	82.22(6.75)%	82.59(7.12)%	79.26(10.59)%	82.59(5.94)%
hepatitis	**76.18(4.98)%**	**76.18(4.98)%**	**76.18(4.98)%**	**76.18(4.98)%**	**76.18(4.98)%**	**76.18(4.98)%**
ionoshpere	87.18(3.03)%	87.75(4.23)%	85.75(3.58)%	**88.61(3.76)%**	85.70(3.45)%	88.33(3.37)%
liver	67.25(6.28)%	67.54(5.85)%	**67.83(6.01)%**	67.54(5.85)%	67.25(6.28)%	67.25(6.03)%
sonar	69.70(14.00)%	67.78(13.45)%	68.25(11.98)%	**69.72(11.27)%**	68.76(10.95)%	65.81(16.19)%
transfusion	**76.34(0.52)%**	76.20(0.28)%	**76.34(0.52)%**	76.20(0.28)%	**76.34(0.52)%**	76.20(0.28)%

Table 5. Benchmark results for the non-linear case with $\mathbf{Q_1}$ precomputed kernel.

	RBF SVM	$\mathbf{Q}_W$	$\mathbf{Q}_B$	$\mathbf{Q}_C$	$\mathbf{Q}_L$	$\mathbf{Q}_E$
australian	78.10(2.79)%	79.12(2.11)%	78.69(2.54)%	78.98(1.93)%	82.61(1.69)%	**84.49(2.41)%**
breast	94.58(0.79)%	95.17(1.29)%	95.61(1.35)%	95.17(1.29)%	95.55(1.19)%	**96.34(1.49)%**
diabetes	71.35(1.59)%	71.87(1.32)%	71.22(1.56)%	71.35(1.64)%	**73.30(1.58)%**	73.04(1.77)%
german	**70.70(1.68)%**	60.10(3.97)%	59.60(3.49)%	59.60(3.49)%	69.60(2.95)%	59.50(3.69)%
heart	73.70(2.41)%	**76.30(3.04)%**	75.93(3.70)%	75.93(3.70)%	67.41(12.40)%	75.93(5.56)%
hepatitis	**78.06(4.81)%**	77.41(6.49)%	76.76(6.24)%	77.41(6.49)%	76.14(4.78)%	76.76(6.24)%
ionoshpere	92.31(2.62)%	**92.60(2.77)%**	91.18(2.77)%	90.89(2.98)%	90.82(2.56)%	90.32(2.98)%
liver	68.99(8.79)%	69.86(9.58)%	**71.01(8.99)%**	69.86(8.41)%	70.72(8.03)%	69.28(8.16)%
sonar	73.07(9.02)%	**75.00(2.72)%**	**75.00(2.72)%**	**75.00(2.72)%**	70.63(3.92)%	**75.00(2.72)%**
transfusion	77.95(2.58)%	**78.88(3.78)%**	78.75(4.05)%	78.75(4.07)%	**78.88(4.36)%**	78.48(4.00)%

in most of the problems, but not the maximum of all kernels. A disadvantage of this type of precomputed kernel is that, it is computationally expensive and also renders all the training samples as support vectors due to (25)

5. CONCLUSION

In this paper we have proposed a novel classifier which combines the Graph Embedding Framework and the SVM. We added a new regularization term in the original SVM formulation, based on the Graph Embedding. This term incorporates knowledge about the underlying distribution of the data into the SVM optimization problem. The resulting dual corresponds to an original SVM with a new precomputed kernel. We examined both the linear and the kernel version of the SVM formulation. Experimental results on several benchmark datasets show that our method obtains improved classification accuracy in comparison with the standard SVM. Moreover, the proposed framework allows for new algorithms to be designed that define a graph structure to be imposed in the SVM classifier.

6. REFERENCES

[1] Bernhard Schölkopf and Alexander J. Smola, *Learning With Kernels Support Vector Machines, Regularization, Optimization, and Beyond*, MIT Press, December 2001.

[2] S. Yan, D. Xu, B. Zhang, H.J. Zhang, Q. Yang, and S. Lin, "Graph Embedding and Extensions: A General Framework for Dimensionality Reduction," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 1, pp. 40–51, January 2007.

[3] I. Joliffe, "Principal Component Analysis," *Springer-Verlag*, 1986.

[4] A.M. Martinez and A.C. Kak, "Pca versus lda," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 23, no. 2, pp. 228–233, February 2001.

[5] S. Roweis and L. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *Science*, vol. 290, no. 22, pp. 2323–2326, December 2000.

[6] M. Belkin and P. Niyogi, "Laplacian eigenmaps and spectral techniques for embedding and clustering," *Advances in Neural Information Processing System*, vol. 14, pp. 585–591, 2001.

[7] Z. Xu, I. King, M. Rung-Tsong Lyu, and R. Jin, "Discriminative Semi-Supervised Feature Selection via Manifold Regularization," *IEEE Transactions on Neural Networks*, vol. 21, no. 7, pp. 1033–1047, July 2010.

[8] M. Belkin, P. Niyogi, and V. Sindhwani, "Manifold Regularization: A Geometric Framework for Learning from Labeled and Unlabeled Examples," *Journal of Machine Learning Research*, vol. 7, pp. 2399–2434, November 2006.

[9] S. Melacci and M. Belkin, "Laplacian Support Vector Machines Trained in the Primal," *JMLR*, vol. 12, pp. 1149–1184, Mar 2011.