Contribution to graph-based manifold learning with application to image categorization

Ruifeng Zhu

▶ **To cite this version:**

Ruifeng Zhu. Contribution to graph-based manifold learning with application to image categorization. Other [cs.OH]. Université Bourgogne Franche-Comté; Universidad del País Vasco. Facultad de ciencias, 2020. English. NNT : 2020UBFCA015 . tel-02980983

HAL Id: tel-02980983
https://theses.hal.science/tel-02980983

Submitted on 27 Oct 2020

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

JOINT PHD DEGREE

To obtain the degree of Doctor issued by

UNIVERSITY OF BOURGOGNE FRANCHE-COMTÉ
Specialty: Informatique

and

UNIVERSITY OF THE BASQUE COUNTRY
Specialty: Informatics Engineering

Presented and defended by

Ruifeng ZHU
July 08, 2020

Contribution to Graph-based Manifold Learning with Application to Image Categorization

Committee Members

Mr. Denis Hamad	HDR-Full Professor, University of Littoral Opal Coast, Reviewer
Mr. Franck Davoine	HDR-CNRS Researcher, University of Technology of Compiegne, Reviewer
Mr. Hichem Snoussi	HDR-Full Professor, University of Technology of Troyes, Examiner
Mr. Vincent Hilaire	HDR-Full Professor, University of Bourgogne Franche-Comté, Examiner
Mr. Fadi Dornaika	Research Professor, University of the Basque Country, Supervisor
Mr. Yassine Ruichek	HDR-Full Professor, University of Bourgogne Franche-Comté, Supervisor

Contents

Abstract

Contribution to graph-based manifold learning with application to image categorization

Ph.D. Candidate:
Ruifeng ZHU
University of Burgundy Franche-Comté, Belfort (UBFC), France, 2016-2020
University of the Basque Country, San Sebastián (EHU/UPV), Spain, 2017-2020

Supervisors:
Yassine RUICHEK (Professor, UBFC)
Fadi DORNAIKA (Professor, EHU/UPV)

Graph-based Manifold Learning algorithms are regarded as a powerful technique for feature extraction and dimensionality reduction in Pattern Recogniton, Computer Vision and Machine Learning fields. These algorithms uzilize sample information contained in the item-item similarity and weighted matrix to reveal the intrinstic geometric structure of manifold. It exhibits the low dimensional structure in the high dimensional data. This motivates me to develop Graph-based Manifold Learning techniques on Pattern Recognition, specially, application to image categorization. The experimental datasets of thesis correspond to several categories of public image datasets such as face datasets, indoor and outdoor scene datasets, objects datasets and so on. Several approaches are proposed in this thesis: 1) A novel nonlinear method called Flexible Discriminant graph-based Embedding with feature selection (FDEFS) is proposed. We seek a non-linear and a linear representation of the data that can be suitable for generic learning tasks such as classification and clustering. Besides, a byproduct of the proposed embedding framework is the feature selection of the original features, where the estimated linear transformation matrix can be used for feature ranking and selection. 2) We investigate strategies and related algorithms to develop a joint graph-based embedding and an explicit feature weighting for getting a flexible and inductive nonlinear data representation on manifolds. The proposed criterion explicitly estimates the feature weights together with the projected data and the linear transformation such that data smoothness and large margins are achieved in the projection space. Moreover, this chapter introduces a kernel variant of the model in order to get an inductive nonlinear embedding that is close to a real nonlinear subspace for a good approximation of the embedded data. 3) We propose the graph convolution based semi-supervised Embedding (GCSE). It provides a new perspective to non-linear data

embedding research, and makes a link to signal processing on graph methods. The proposed method utilizes and exploits graphs in two ways. First, it deploys data smoothness over graphs. Second, its regression model is built on the joint use of the data and their graph in the sense that the regression model works with convolved data. The convolved data are obtained by feature propagation. 4) A flexible deep learning that can overcome the limitations and weaknesses of single-layer learning models is introduced. We call this strategy an Elastic graph-based embedding with deep architecture which deeply explores the structural information of the data. The resulting framework can be used for semi-supervised and supervised settings. Besides, the resulting optimization problems can be solved efficiently.

KEY WORDS: Machine Learning, Manifold Learning, Graph-based Embedding, Semi-supervised Learning, Feature Selection, Image Categorization, Pattern Recognition, Computer Vision

I

Scientific Background

1

Introduction

1.1/ Background

Feature extraction from high-dimensional data is a difficult problem in several fields such as pattern recognition, machine learning, and computer vision. In order to tackle these difficulties, Graph-based Manifold Learning techniques [104] [92] [2] are developed to extract relevant features from the raw ones.

Two of the most popular feature extraction algorithms for dimensionality reduction applications might be Principal Component Analysis (PCA) [105] and Linear Discriminant Analysis (LDA) [1] before Graph-based Manifold Learning methods came out. Principal Component Analysis uzilizes eigenvectors of sample data's covariance to perform dimensionality reducing. It projects the original d-dimensional data onto the m-dimensional ($m \ll d$) subspace. PCA focuses on finding mutually orthogonal basis functions for obtaining the maximum variance's directions in sample data. It will preserve pairwise Euclidean distances. LDA is based on Fisher Score [26] and generates separated categories in low-dimensional subspace when sample data are linearly separable. But in real-world, datasets may contain nonlinear structures. They are invisible to such as PCA and LDA classical dimensionality reduction methods.

This motivates us to consider graph-based manifold learning techniques for feature extraction. Various of Graph-based Manifold Learning techniques, such as the most classical methods that ISOMAP [104], Locally Linear Embedding (LLE) [92] and Laplacian Eigenmap (LE) [2] have been proposed.

All of these algorithms metioned above are based on manifold space and graph theory. The manifold is a locally Euclidean topological space, i.e., each point of a manifold has a neighborhood that can be continuously mapped to the Euclidean space of the same dimensions and vice-versa. In this sense, manifold learning refers to a class of techniques that learn a low-dimensional embedding of the data points lying in a high dimensional space, while preserving original characteristics of the data. These techniques are motivated by suggestions that any interesting high-dimensional sample data could be regarded as genmetrically set. The points of set are near to surface of low-dimensional manifold structure. Manifold learning techniques have been primarily used for unsupervised dimensionality reduction transformations at beginning, where the local relationships between the data points in the original space are preserved during the transformation. Fig. 1.1 illustrates the concept of manifold smoothness. This figure shows the classical "Swiss Roll" experiment introduced by Locally Linear Embedding firstly [92]. The left part

of this figure depicts the data that live in a 3D space. The right part depicts a non-linear embedding of the data in a low dimensional space (a 2D space). The manifold smoothness concept stipulates that any pair of data samples that are close in the original space they should be also close in the low-dimensional space. The graph-based for feature extraction methods could be explained as follows.

The other basic is graph theory. We regard $\mathbf{G}(\mathbf{V}, \mathbf{E})$ as Graph, where $\mathbf{V}$ ($\mathbf{V} = \{\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_N\}$) denotes the set of vertices or nodes and $\mathbf{E}$ is the set of edges. If each node $\mathbf{v}_i$ has the relationship with the other node $\mathbf{v}_j$, which also means that their edge weight $S_{ij} > 0$. On the contrary, if $\mathbf{v}_i$ and $\mathbf{v}_j$ are non-connected then $S_{ij} = 0$. When the graph is undirected, $S_{ij} = S_{ji}$. In general, S_{ij} is regarded as a measure of similarity between the vertices $\mathbf{v}_i$ and $\mathbf{v}_j$. Besides, more similar two vertices $\mathbf{v}_i$ and $\mathbf{v}_j$, usually, the higher weigh value of S_{ij}. The Graph Embedding is treated as a mapping of f: $\mathbf{v}_i \in \mathbb{R}^d \rightarrow \mathbf{z}_i \in \mathbb{R}^m$, where $i \in (1, 2, \cdots, N)$ and $d \gg m$. For the data depicted in Fig. 1.1, an embedding can map each node to a low-dimensional feature vector and tries to preserve the connection strengths between vertices. The mapping function f reserves proximity measure defined on Graph $\mathbf{G}(\mathbf{V}, \mathbf{E})$.

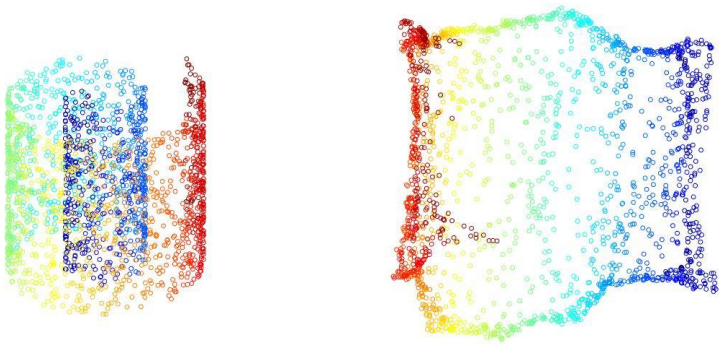


Figure 1.1: The classical Synthetic "Swiss Roll" example. The left part of this figure depicts the data that live in a 3D space. The right part depicts a non-linear embedding of the data in a low dimensional space (a 2D space). The manifold concept stipulates that any pair of data samples that are close in the original space they should be also close in the low-dimensional space.

We utilize the manifold theory and graph-based learning theory to develop our research. This thesis focuses on graph-based manifold learning with application to image categorization. According to the availability of label information, Learning-based Pattern Recognition algorithms usually could be classified into Unsupervised Learning, Supervised Learning and Semi-supervised Learning algorithms. The standard concept of Supervised Learning, Unsupervised Learning and Semi-supervised Learning are as follows.

Supervised Learning. The learning system observes a labeled training set consisting of features and labels, denoted by $\mathbf{X} = [\mathbf{x}_1, \cdots, \mathbf{x}_N] \in \mathbb{R}^{d \times N}$ and $y_i \in \{1, 2, ..., C\}$, $i = 1, 2, ...N$, respectively. Where y_i is the label of $\mathbf{x}_i$, d is that dimensionality of original data, N is the number of data samples and C is the number of classes. The goal is to predict the label y_i for any new input with feature $\mathbf{x}_i$.

Unsupervised Learning. The learning system observes an unlabeled set of items,

represented by their features $\mathbf{X} = [\mathbf{x}_1, \cdots, \mathbf{x}_N] \in \mathbb{R}^{d \times N}$. We do not know the labels $y_i \in \{1, 2, ..., C\}$ of the features in advance. The goal is to organize the items. Typical unsupervised learning tasks include clustering, dimensionality reduction and so on.

Semi-supervised Learning. The training data consists of both l labeled samples and u unlabeled samples, denoted by $\mathbf{X}_l = [\mathbf{x}_1, \cdots, \mathbf{x}_l] \in \mathbb{R}^{d \times l}$ and $\mathbf{X}_u = [\mathbf{x}_{l+1}, \cdots, \mathbf{x}_{l+u}] \in \mathbb{R}^{d \times u}$, respectively. The whole dataset could be regarded as $\mathbf{X} = [\mathbf{X}_l, \mathbf{X}_u] \in \mathbb{R}^{d \times (l+u)}$, where $N = l + u$. Usually, we regard that the ublabeled sample data are much more than labeled sample data, namely, $u \gg l$ or $u > l$. Semi-supervised classification learning aims at training a classifier from unlabeded sample data and labeled sample data together. It avoids expensive human labour for labelling data. So semi-supervised learning is more suitable for applications in real-world than supervised learning.

The Graph-based Manifold Learning techniques on Pattern Recognition, specially, application to image categorization are the fields that we focus on in the thesis. The related topics are Graph Construction and Large-Scale Graphs, Unsupervised Graph-Based Manifold Learning methods, Semi-supervised/Supervised Graph-Based Manifold Learning methods, Label Propagation methods, Dimensionality Reduction methods, Feature Selection methods. We also introduce several of state-of-the-art topics on Graphs, such as Deep Learning on Graphs and Manifolds, Graph-based application on Learning to Hash.

As described above, Graph Construction is a crucial step in graph-based manifold learning which is the conversion of data into a weighted graph. Supervised, Semi-supervised and Unsupervised Learning tasks can uzilize graphs in order to estimate their models. Graph-based Label Propagation algorithms rely on generating the graph where the labeled and unlabeled data points from the nodes $\mathbf{V}$ of Graph $\mathbf{G}(\mathbf{V}, \mathbf{E})$ and similarities between points are edges $\mathbf{E}$ of Graph. We uzilize labeled sample data for propagating information to unlabel sample data though the Graph. Dimensionality reduction learns to project high-dimensional sample data onto low-dimensional subspace and avoiding losing discriminant information as much as possible. Feature Selection focuses on selecting a subset of features. Specially, Feature Selection minimizes redundancy and maximizes relevance to the target. Deep Learning on Graphs and Manifolds aims at generalizing deep learning apporachs to non-Euclidean domains (e.g. graph theory and manifold learning). Learning to Hash are motivated by Graph-based Manifold Learning algorithm (Laplacian Eigenmaps algorithm [2]) and Semantic Hashing [93] for efficient nearest neighbor search in massive databases [116]. We will introduce all of these Graph-based Manifold Learning techniques in Chapter 2 for details.

The experimental datasets of the thesis correspond to several categories of applications. The thesis findings deployed diverse public image datasets such as Face Datasets, Indoor and Outdoor Scene datasets, Objects datasets, etc. Figs. 1.2, 1.3, 1.4 show some samples in three different datasets. The supervised learning handwritten digits recognition will be used as an illustrative example of problem statement that is adopted in image categorization. Consider handwritten digits recognition, image samples are shown in in Fig. 1.5. We assume that the size of handwritten digits images is 28×28. If it is represented by a vector $\mathbf{x}_i \in \mathbb{R}^{d \times N}$, where $d = 28 \times 28$ and $i = 1, 2, \cdots, N$. There are N images in the whole dataset ($\mathbf{X} = [\mathbf{x}_1, \cdots, \mathbf{x}_N]$). We aim at generating a classifier that uzilizes the input sample $\mathbf{x}_i$ to produce the sample label $(0, 1, 2, 3, 4, 5, 6, 7, 8, 9)$. How to obtain a better recognition result? We uzilize machine learning techniques that dataset $\mathbf{X} = [\mathbf{x}_1, \cdots, \mathbf{x}_N]$ is regarded as training set, and it is used for estimating the parameters

of suitable model. The labels of digits in training set are known. The target label y_i is regarded as the category of digit.

Figure 1.2: Several images from the 8 Sports Event Categories Dataset. The images in the first row are from Bocce Sport category and the ones in the second row are from the Rock Climbing category.

Figure 1.3: Several examples from the ORL Face Dataset. The images contain frontal images under differentfacial expression and pose per individual.

Figure 1.4: Several examples from the COIL-20 Object Dataset.

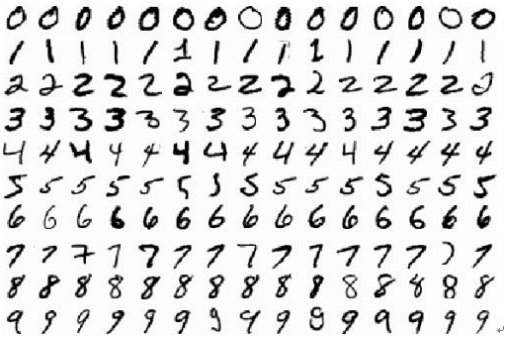


Figure 1.5: Several examples from the MNIST handwritten digits Dataset.

1.2/ Contributions and Outline

In this thesis, we focus on the study of the Graph-based Manifold Learning and its Applications to image categorization. More precisely, we target the graph-based semi-supervised learning for categorizing images. The thesis introduces several flexible schemes that can provide data representations using both linear and nonlinear projections. Several of our proposed methods also integrate the concept of original feature ranking in order to make a better use of the data. The main contributions are summarized as follows:

- A novel nonlinear method called Flexible Discriminant graph-based Embedding with feature selection (FDEFS). We seek a non-linear representation of the data that can be suitable for generic learning tasks such as classification and clustering. Besides, a byproduct of the proposed embedding framework is the feature selection of the original features, where the estimated linear transformation matrix $\mathbf{W}$ can be used for feature ranking and selection.

- We investigate strategies and related algorithms to develop a joint graph-based embedding and feature weighting for getting a flexible and inductive nonlinear data representation on manifolds. The proposed criterion explicitly estimates the feature weights together with the projected data and the linear transformation such that data smoothness and large margins are achieved in the projection space. Moreover, the chapter introduces a kernel variant of the model in order to get an inductive nonlinear embedding that is close to a real nonlinear subspace for a good approximation of the embedded data.

- We propose the graph convolution based semi-supervised Embedding (GCSE). It provides a new perspective to non-linear data embedding research, and makes a link to signal processing on graph methods. The proposed method utilizes and exploits graphs in two ways. First, it deploys data smoothness over graphs. Second, its regression model is built on the joint use of the data and their graph in the sense

that the regression model works with convolved data. The resulting scheme can solve and address the problem of over-fitting on local neighborhoods for image data of various types like faces, outdoor scenes, and man-made objects.

- The flexible deep learning that can overcome the limitations and weaknesses of single-layer learning models is introduced, that we call this strategy an Elastic graph-based embedding with deep architecture which deeply explores the structural information of the data. The resulting framework can be used for semi-supervised and supervised settings. Besides, the resulting optimization problems can be solved efficiently.

The thesis is organized as follows: Graph-based Manifold Learning techniques are introduced in the Chapter 2. Chapter 3 discusses a novel nonlinear method called Flexible Discriminant graph-based Embedding with feature selection (FDEFS). Chapter 4 introduces a joint graph-based embedding and explicit feature weighting for getting a flexible and inductive nonlinear data representation on manifolds. Chapter 5 proposes an algorithm of graph convolution based semi-supervised Embedding (GCSE). Chapter 6 describes an Elastic graph-based embedding with deep architecture which deeply explores the structural information of the data. Finally, the conclusions of the work and some perspectives are given in the Chapter 7.

2

Related Work

In this chapter, we introduce some backgrounds on Graph-based Manifold Learning for Pattern Recognition applications, including Graph Construction and Large-Scale Graphs, Unsupervised Methods, Graph-based Manifold Learning on Semi-supervised Methods, Label Propagation Methods, Dimensionality Reduction Methods, Feature Selection Methods and the other state-of-the-art topics on Graphs in recent years (e.g., Deep Learning on Graphs and Manifolds, Graph-based application on Learning to Hash).

First of all, we introduce the notations adopted in this chapter and the whole thesis. We define the graph $\mathbf{G}(\mathbf{V}, \mathbf{E})$ with vertices (or nodes) $\mathbf{V}$ and edges $\mathbf{E}$. Let sample data matrix as $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_l, \mathbf{x}_{l+1}, \ldots, \mathbf{x}_{l+u}] \in \mathbb{R}^{d\times(l+u)}$, where $\mathbf{x}_i \big|_{i=1}^{l}$ and $\mathbf{x}_i \big|_{i=l+1}^{l+u}$ are the labeled training samples and unlabeled test samples, respectively, l and u are the numbers of labeled train samples and unlabeled test samples data, respectively, and d is the sample dimension. Let $N = l+u$ be the total number of samples data and n_c is the total number of labeled samples in the cth class. We represent the labeled training samples by the matrix $\mathbf{X}_l = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_l] \in \mathbb{R}^{d\times l}$. The label of each sample $\mathbf{x}_i$ is denoted by $y_i \in \{1, 2, ..., C\}$ $i = 1, 2, \cdots, l$, where C is the total number of classes. $\mathbf{X}_u = [\mathbf{x}_{1+1}, \mathbf{x}_{l+2}, \ldots, \mathbf{x}_{l+u}] \in \mathbb{R}^{d\times u}$ denotes the unlabeled test data matrix.

Define similarity matrix $\mathbf{S}$, a Laplacian matrix $\mathbf{L}$ can be computed. If we assume that the similarity matrix $\mathbf{S}$ is symmetric, then the classic Laplacian matrix is given by $\mathbf{L} = \mathbf{D} - \mathbf{S}$ where $\mathbf{D}$ is a diagonal matrix whose elements are the row or column (since the matrix is symmetric) sums of $\mathbf{S}$ matrix, namely $\mathbf{D} = diag(\mathbf{S} \cdot \mathbf{1})$. $\mathbf{1}$ or $\mathbf{0}$ mean that the vector with all ones or zeros. The normalized Laplacian matrix $\mathbf{L}$ is defined by $\mathbf{L} = \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{S}\mathbf{D}^{1/2}$ where $\mathbf{I}$ denotes the identity matrix. Tab. 2.1 summarizes the main notations. For the other notations used in the thesis, we usually emphasize their meanings near by the equations or at the beginning of Subsection, and keep their meanings consistently in the whole thesis as possible.

Table 2.1: Main notations used in the thesis.

Notation	Description
d	Dimensionality of original data
N	Number of data samples
l	Number of labeled samples
u	Number of unlabeled samples
C	Number of classes
n_c	Number of labeled samples in the c-th class
$\mathbf{x}_i \in \mathbb{R}^d$	The i-th original data sample
$y_i \in \{1, 2, ..., C\}$	The label of $\mathbf{x}_i$
$\mathbf{X}_l = [\mathbf{x}_1, \cdots, \mathbf{x}_l] \in \mathbb{R}^{d \times l}$	Labeled training samples matrix
$\mathbf{X}_u = [\mathbf{x}_{1+1}, \cdots, \mathbf{x}_{l+u}] \in \mathbb{R}^{d \times u}$	Unlabeled test samples matrix
$\mathbf{X} = [\mathbf{X}_l, \mathbf{X}_u] \in \mathbb{R}^{d \times (l+u)}$	Original data matrix
$\mathbf{D}$	Diagonal matrix
$\mathbf{I}$	Identity matrix
$\mathbf{L}$	Laplacian matrix
$\mathbf{1}$ or $\mathbf{0}$	Vector with all ones or zeros entries

2.1/ Graph Construction and Large-Scale Graphs

For researchers in graph theory field, they mainly focus on analyzing and mining information patterns from graph. Graph construction will be defined or provided at first. If we uzilize graph in real world scenarios, it becomes not certain, especially, the sample data in real world scenarios are noise, multi-distribution, high-dimensionality and uncertain definition. So the graph construction becomes an important research in manifold learning field. In this Section, we focus on three aspects to introduce Graph Construction methods which are classical Graph Construction algorithms, data self representation Graph Construction and Large-Scale Graphs.

Classical Graph Construction algorithms were introduced by Tenenbaum, J. B., De Silva, V., and Langford, J. C. in [104]. k-nearest neighbors graph (kNN) and ϵ-neighborhoods graph are two of the most typical methods for Graph Construction. We focus on any two points to compute their distance ($d(\mathbf{x}_i, \mathbf{x}_j)$) or simlarity ($sim(\mathbf{x}_i, \mathbf{x}_j)$), which could be given by:

$$W_{ij} = \begin{cases} \text{sim}\left(\mathbf{x}_i, \mathbf{x}_j\right) if\ \mathbf{x}_i\ is\ the\ nearest\ neighbor\ of\ \mathbf{x}_j\ or\ vice\ versa, \\ 0\ \ otherwise. \end{cases} \tag{2.1}$$

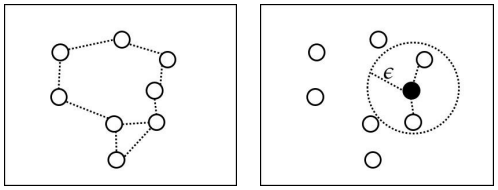
According to Eq. 2.1, $\mathbf{W}$ is a symmetric matrix since $\mathbf{x}_i$ is the nearest neighbor of $\mathbf{x}_j$ or $\mathbf{x}_j$ is the nearest neighbor of $\mathbf{x}_i$. We have to say, kNN graph needs a large of computing ability and resource when dataset is huge and k is a big number. So we need the more novel and advantage algorithms to compute the large-scale graph that will be discussed in the end of this section (such as Anchor Graph [68]).

In k-nearest neighbors graph, we regard that $\mathbf{x}_i$ and $\mathbf{x}_j$ are connected, while $\mathbf{x}_j$ is among k nearest neighbors of $\mathbf{x}_i$ or $\mathbf{x}_i$ is among k nearest neighbors of $\mathbf{x}_j$. In ϵ-neighborhoods graph, we regard that $\mathbf{x}_i$ and $\mathbf{x}_j$ are connected, while $\left\|\mathbf{x}_i - \mathbf{x}_j\right\|^2 < \epsilon$ where $\|\cdot\|$ is the Euclidean norm in $\mathbb{R}$. An example of k-nearest neighbors graph is shown in Figure 2.1(a)

for k = 2. Another example of ϵ-neighborhoods graph is shown in Figure 2.1(b). After obtaining adjacency graph, heat kernel usually is chosen to be the function of weights as following:

$$W_{\text{ij}} = \begin{cases} e^{-\frac{||\mathbf{x}_i - \mathbf{x}_j||^2}{t}} & if\ nodes\ i\ and\ j\ are\ connected \\ 0 & otherwise \end{cases} \tag{2.2}$$

where $t \in \mathbb{R}$.



(a) (b)

Figure 2.1: Illustrations of k-nearest neighbors graph and ϵ-neighborhoods graph. (a) k-nearest neighbors graph with k = 2. (b) ϵ-neighborhoods graph.

The other typical classical method is Simple-minded. There is no parameter in this algorithm. Define $W_{\text{ij}} = 1$ if and only if vertices $\mathbf{x}_i$ and $\mathbf{x}_j$ are connected by an edge. Otherwise, $W_{\text{ij}} = 0$. As the following shown,

$$W_{\text{ij}} = \begin{cases} 1\ if\ nodes\ \mathbf{x}_i\ and\ \mathbf{x}_j\ are\ connected \\ 0 \qquad\qquad otherwise \end{cases} \tag{2.3}$$

In Graph Construction field, most of algorithms are based on data self representation methods, including with the k-nearest neighbors graph and ϵ-neighborhoods graph. As the mathmatics model of graph is descripted in the previous section, to define the graph $\mathbf{G}(\mathbf{V}, \mathbf{E})$ with vertices (or nodes) $\mathbf{V}$ and edges. $\mathbf{E}$. A weight W_{ij} is assigned to each edge for measuring the link strength, which is usually a necessary step for graph learning. Therefore, it also describes a graph by a three-tuple $\mathbf{G}(\mathbf{V}, \mathbf{E}, \mathbf{W})$. In the graph, data is associated with the vertices. So two main steps are generally conducted to construct a graph which to effect the data self representation Graph Construction. The first is to Determine the topological structure of the graph, especially for the edge set $\mathbf{E}$. Then, Based on the current edge set, determine the weight matrix $\mathbf{W}$ [87].

Different from ϵ-neighborhoods graph and k nearest neighbors graph (kNN), ℓ_1-Graph [4] aims at automatic sparsity, better robustness for the noise and adaptive neighborhood. ℓ_1-norm optimization problem is used for robust sparse coding. Define sample set $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N]$ and $\mathbf{x}_i \in \mathbb{R}^d$. In Robust Sparse Representation [117], suppose $\mathbf{x}_i = \Upsilon\alpha$, where $\mathbf{x}_i \in \mathbb{R}^d$ is needed to be approximate, $\alpha \in \mathbb{R}^m$ is unknown reconstruction coefficients, $\Upsilon \in \mathbb{R}^{d \times m}$ is the overcomplete dictionary ($d < m$). So ℓ_1-Graph construction is descripted as following:

$$\min_{\alpha^i} \left\|\alpha^i\right\|_1, \quad s.t. \quad \mathbf{x}_i = \mathbf{B}^i\alpha^i \tag{2.4}$$

where $\alpha^i \in \mathbb{R}^{d+N-1}$ and $\mathbf{B}^i = [\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_{i-1}, \mathbf{x}_{i+1}, \cdots, \mathbf{x}_N, I] \in \mathbb{R}^{d\times(d+N-1)}$.

The graph weight setting is to define $\mathbf{G} = \{\mathbf{X}, \mathbf{W}\}$ as the ℓ_1-based graph. $\mathbf{W}$ is graph weight matrix and sample data $\mathbf{X}$ is graph nodes. We regard that $W_{i,j} = \alpha^i_j$ if $i > j$ and $W_{i,j} = \alpha^i_{j-1}$ if $i < j$. After obtaining $\mathbf{W}$, it could be used for ℓ_1-Graph weight.

Motivated by ℓ_1-based graph, a ℓ_2-based graph construction [84] via a sparse similarity graph method is proposed. It is used for robust subspace learning and subspace clustering. This method measures the similarity among data points through the reconstruction coefficients with ℓ_2-norm. To further capture the global data structure, Liu et al. [66] propose the LRR-graph, which seeks a Low-Rank Representation(LRR) of the data. By jointly obtaining the representation of all the data under the low-rankness assumption, LRR-graph effectively impose global constraints on the data structure (e.g., multiple subspaces). Moreover, since each sample can be used to represent itself, there always exist a feasible solution for LRR-graph even if the data sampling is insufficient. These properties make LRR-graph become a good candidate for various learning tasks including Semi-Supervised Learning. Non-Negative Low-Rank and Sparse (NNLRS) Graph is another optional data self representation graph construction for semi-supervsied learning method [135]. In NNLRS-graph, the weights in graph are generating via finding a non-negative low-rank and sparse matrix. The NNLRS-graph could obtain joint global mixture of subspace structure and locally linear structure of sample set. So its advantages include generative and discriminative. In [136], it explicitly incorporates the label information into ℓ_1-graph, LRR-graph and NNLRS-graph to extend graph construction methods for Semi-Supervised(SS) learning, these algorithms are SS ℓ_1-graph, SSLRR-graph and SSNNLRS-graph.

The Graph Construction decides neighborhood structure and strength of association between the nodes in the whole graph $\mathbf{G}(\mathbf{V}, \mathbf{E})$. The typical Graph Construction algorithms mentioned above usually can not be used for Large Scale Graph Construction of large dataset. A series of algorithms proposed to solve Large Scale Graph Construction problems.

Among all of these methods, Anchor Graph is one of the typical representative as large graph construction algorithm proposed by W. LIU et al. in 2010 [68]. In Anchor Graph, it defines a small categories of anchor points that could cover the whole point nodes. The anchor points are able to nonparametric regression. As locally weighted average of labels, the processing on anchor points could predict the label for each data point.

The anchor graph tries to use smaller set of k points, which are called by anchors. They are defined by $\mathcal{U} = \{u_1, u_2, \cdots, u_k\} \in \mathbb{R}^{d\times k}$ for approximating the structure of neighborhood underlying sample data $\mathbf{X}$= $\{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_l, \mathbf{x}_{l+1}, \cdots, \mathbf{x}_{l+u}\} = \{\mathbf{X}_l, \mathbf{X}_u\} \in \mathbb{R}^{d\times N}$, where $k \ll N$.. After computing theh similarities of the whole sample points with respect to k anchors by linear time $O(d \cdot k \cdot N)$. We approximate the true affinity matrix $\mathbf{S}_{true} \in \mathbb{R}^{N\times N}$ via uziliz-ing these affinities. Detailedly, the anchor graph exploits the nonlinear data-to-anchor projection, namely, $\mathbb{R}^d \to \mathbb{R}^k$ as follow:

$$\mathbf{z}(\mathbf{x}) = \frac{\left[\delta_1 \exp\left(-\frac{dist^2(\mathbf{x},u_1)}{t}\right), \cdots, \delta_k \exp\left(-\frac{dist^2(\mathbf{x},u_k)}{t}\right)\right]^T}{F} \tag{2.5}$$

where t means band width ($t > 0$), $\delta_j \in (1, 0)$ and $\delta_j = 1$ if and only if the anchor u_j is one of $s \ll k$ closest anchors of x in $\mathcal{U}$ according to distance function $dist(\cdot)$(such as ℓ_2

distance). $F = \sum_{j=1}^{k} \delta_j \exp\left(-\frac{dist^2(\mathbf{x},\mathbf{u}_1)}{t}\right)$ leads to $\|z(\mathbf{x})\|_1 = 1$.

The anchor graph generates data-to-anchor affinity matrix $\mathbf{Z} = [z(x_1), \cdots, z(x_n)]^\top \in \mathbb{R}^{N\times k}$ which defines highly sparse. In the end, anchor graph obtains the data-to-data affinity matrix $\mathbf{S} = \mathbf{Z}\mathbf{\Lambda}^{-1}\mathbf{Z}^\top \in \mathbb{R}^{N\times N}$ that $\mathbf{\Lambda} = \text{diag}(\mathbf{Z}^\top\mathbf{1}) \in \mathbb{R}^{k\times k}$. It could be used for approximating true affinity matrix $\mathbf{S}_{true}$.

Large scale graph construction also could be used for efficient nearest neighbor search in massive databases, which is also used in Learning to Hashing [114]. In [69], a graph-based hashing method which automatically discovers the neighborhood structure inherent in the data to learn appropriate compact codes and uzilize Anchor Graphs to obtain tractable low-rank adjacency matrices. Furthermore, a series of methods also utilized large scale graph construction on Graph-based Learning to Hashing algorithms [63] [50]. This will be discussed in Subsection 2.7.2.

2.2/ Unsupervised Graph-Based Manifold Learning

In this Section, we aim at two aspects to introduce Unsupervised Graph-Based Manifold Learning methods which are classical Unsupervised Graph-based Manifold Learning and the development of Unsupervised Manifold Learning algorithms.

2.2.1/ Classical Unsupervised Graph-based Manifold Learning Algorithms

We begin with a brief review of Locally Linear Embedding (LLE) [92], Laplacian Eigenmaps (LE) [2] and ISOMAP [104] that the most classical manifold learning techniques. They began a new era of Graph-based Manifold Learning domain for Pattern Recognition.

2.2.1.1/ Locally Linear Embedding

Locally Linear Embedding (LLE) [92] is the representative method of manifold learning on unsupervised learning. It aims on exploiting the local symmetries of linear reconstructions for learning the global structure of nonlinear manifolds. LLE computes neighborhood-preserving and low-dimensional embeddings of high-dimensional inputs.

Define neighbors to sample data $\mathbf{x}_i$ using the ϵ-neighborhoods or k nearest neighbors. We have already introduced the details about how to obtain a graph in Section 2.1. We define

$$\phi(\mathbf{W}) = \sum_{i=1}^{N} \left\| \mathbf{x}_i - \sum_{j=1}^{N} W_{ij}\mathbf{x}_j \right\|^2 \quad s.t. \sum_{j=1}^{N} W_{ij} = 1 \tag{2.6}$$

where W_{ij} is weighted matrix of $\mathbf{G}(\mathbf{V}, \mathbf{E})$. Hence, we can obtain the embedding matrix $\mathbf{Z} = (\mathbf{z}_1, \mathbf{z}_2, \cdots, \mathbf{z}_N) \in \mathbb{R}^{m\times N}$ by minimizing

$$\phi(\mathbf{Z}) = \sum_{i=1}^{N} \left\| \mathbf{z}_i - \sum_{j=1}^{N} W_{ij}\mathbf{z}_j \right\|^2 \quad s.t.\ \frac{1}{N}\mathbf{Z}^T\mathbf{Z} = \mathbf{I}, \quad \sum_i \mathbf{z}_i = 0 \tag{2.7}$$

The optimization problem of Eq. 2.7 could be solved by eigen decomposition.

2.2.1.2/ Laplacian Eigenmap

Laplacian Eigenmaps (LE) [2] method is joint graph laplacian, laplace beltrami operator on manifold and relations to heat equation. LE is a geometric method for tracing high-dimensional sample data, which focuses on maintaining the mapping of nodes as close as together when weighted matrix W_{ij} is high. Here $\mathbf{W}$ could be obtained by two steps that (i) Setting the edges, and (ii) Estimating the weights of those edges. The first step utilize k nearest neighbors method and ϵ-neighborhoods method. The second step that choosing the weighs could to utilize Heat Kernel method (only parameter $t \in \mathbb{R}$) and Simple-minded method (No parameter). Specifically, Laplacian Eigenmaps algorithm mainly minimize the following objective function:

$$\begin{aligned} \phi(\mathbf{Z}) &= \frac{1}{2}\sum_{i,j=1}^{N} \left\|\mathbf{z}_i - \mathbf{z}_j\right\|^2 W_{ij} \\ &= \mathrm{tr}\left(\mathbf{Z}^T\mathbf{L}\mathbf{Z}\right) \quad s.t.\ \mathbf{Z}^T\mathbf{D}\mathbf{Z} = \mathbf{I} \end{aligned} \tag{2.8}$$

where $\mathbf{L}$ is the Laplacian matrix. The optimization problem of Eq. 2.8 could be solved by eigen decomposition.

2.2.1.3/ A Global Geometric Framework for Nonlinear Dimensionality Reduction (ISOMAP)

A global geometric framework for nonlinear dimensionality reduction (ISOMAP) is a representive nonlinear manifold learning method. It focuses on finding optimal subspace which preserves geometric distance among sample data. ISOMAP obtains a globally optimal solution and is guaranteed to coverge asymptotically to real structure. ISOMAP aims on discovering the nonlinear degrees of freedom, which underlies in real-world scenarios. Let $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N) \in \mathbb{R}^{d\times N}$ denote a set of N points in the original d-dimensional space, and $\mathbf{Z} = (\mathbf{z}_1, \mathbf{z}_2, \cdots, \mathbf{z}_N) \in \mathbb{R}^{m\times N}$ be a set of the reduced representations in the m-dimensional space ($m \ll d$). Then, ISOMAP can perform manifold feature learning using the following.

To determine the nearest neighbors of each sample by using k-nearest neighbors graph and ϵ-neighborhoods, and set edge lengths equal to $d(\mathbf{x}_i, \mathbf{x}_j)$; Furthermore, to construct undirected graph $\mathbf{G}(\mathbf{V}, \mathbf{E})$, where each node $\mathbf{v}_i \in \mathbf{V}$ corresponds to a point $\mathbf{x}_i$. Define $d_G(\mathbf{x}_i, \mathbf{x}_j)$ as the shortest path distance between $\mathbf{x}_i$ and $\mathbf{x}_j$ over $\mathbf{G}$. The classical Dijkstra's algorithm and Floyd's algorithm can be applied to find the shortest paths. Finally, to obtain the low-dimensional embedding $\mathbf{Z}$ by solving the following problem minimizing:

$$\phi(\mathbf{Z}) = \sum_{i,j} \left(d\left(\mathbf{z}_i, \mathbf{z}_j\right) - d_G\left(\mathbf{x}_i, \mathbf{x}_j\right) \right)^2 \tag{2.9}$$

which can be similarly solved as the classical multidimensional scaling (MDS) algorithm in [104] for details.

2.2.2/ The development of Unsupervised Manifold Learning

The research of Unsupervised Learning or Clustering methods always accompany Graph-based representations of the relationships among data points. The typical algorithms include Spectral Clustering [73] and Normalized Cut [100].

Since Laplacian Eigenmap (LE) [2], ISOMAP [104] and Locally Linear Embedding (LLE) [92] came out, a series of Unsupervised Graph-Based Manifold Learning methods as follows. Locality Preserving Projection (LPP) [41] is a classical linear version of LE, where a linear transformation is recommended between the original data and their projections. On one hand, the problem of out-of-sample is naturally avoided. On the other hand, the efficiency using LPP for data classification is also improved. Unfortunately, due to introducing the linearity compulsively, the global nonlinear geometry in data may be destroyed, which makes failure to detect the nonlinear geometry structure and cannot well carry out geometry-aware learning. He et al. proposed a novel subspace learning algorithm that is Neighborhood Preserving Embedding (NPE) [39]. It focuses on preserving the structure of local neighborhood on the sample data manifold. NPE is not only defined on the train sample data, but also in the reproducing kernel Hilbert space into which data points are projected. Isometric Projection [8] is also a novel linear dimensionality reduction algorithm. It constructs a weighted data graph, and the weighs are discrete approximations of the geodesic distances on the data manifold. Int the end, a linear subspace is obtained via preserving the pairwise distance.

The follows two Unsupervised Graph-Based Manifold Learning Methods will be introduced details and some of these methods and formulas are also used for the other chapters.

The work described in [46] proposed a graph-based non-linear embedding framework for unsupervised feature selection, termed Joint Embedding Learning and Sparse Regression (JELSR). With this method, the embedding and sparse regression are jointly estimated. JELSR is an unsupervised method that aims to rank the original features via a simultaneous non-linear embedding and sparse regression estimation. Aiming at a graph-based embedding and sparse regression for feature ranking, JELSR solves the following optimization problem:

$$\underset{\mathbf{W},\mathbf{Z}\ s.t.\mathbf{Z}^T\mathbf{Z}=\mathbf{I}}{\arg\min} \quad tr(\mathbf{Z}^T\mathbf{L}\,\mathbf{Z}) + \beta\left(\left\|\mathbf{X}^{\mathbf{T}}\mathbf{W} - \mathbf{Z}\right\|_2^2 + \alpha\,\|\mathbf{W}\|_{2,1}\right) \tag{2.10}$$

where α and β are two regularization parameters. $\mathbf{W} = [\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_m] \in \mathbb{R}^{d\times m}$ is the linear transform matrix, m is the dimensionality of the embedding, and $\mathbf{Z} = [\mathbf{z}_1; \mathbf{z}_2; \ldots; \mathbf{z}_N] \in \mathbb{R}^{N\times m}$ denotes the data matrix of embedding (i.e., the non-linear projection of $\mathbf{X}$). The $\ell_{2,1}$ norm of $\mathbf{W}$ is given by $\|\mathbf{W}\|_{2,1} = \sum_{i=1}^{d} \|\hat{\mathbf{w}}_i\|_2$ where $\hat{\mathbf{w}}_i$ denotes the ith row of $\mathbf{W}$. This norm promotes row sparsity of the linear transform $\mathbf{W}$. According to [46], the JELSR

algorithm could be used for unsupervised feature selection by computing the scores for all features of data, where each score is given by the ℓ_2 norm of the corresponding row in the matrix $\mathbf{W}$. It also can be used for graph-based data embedding using the estimated $\mathbf{Z}$ matrix.

The Manifold Regularized Deep Learning Algorithm (MRDL) in [127] estimates a Non-linear Sparsity Preserving Projections [53]. It can be considered as an improved variant of the unsupervised Local Linear Embedding (LLE) method. The main differences are as follows. First, it is based on a kernel sparse graph. Second, it adopts a cascade of layers. In each layer, a sparse graph is computed and then the non-linear projections are estimated using sparsity preserving property.

The objective function allows the estimation of the similarity matrix $\mathbf{S}$ (the graph matrix) is given by:

$$\min_{\mathbf{S}\geq 0} \|\mathbf{X} - \mathbf{X}\,\mathbf{S}\|_2^2 + \alpha\,\|\mathbf{S}\|^{\frac{1}{2}} + \beta\,\|\mathbf{S}\|_2^2 \tag{2.11}$$

where α and β are two positive regularization parameters. Details on how to solve the above optimization can be found in [127].

Once the graph matrix $\mathbf{S}$ is computed, the non-linear projections $\mathbf{Z}$= ($\mathbf{z}_1$,$\mathbf{z}_2$,... $\mathbf{z}_N$) are estimated by solving the following optimization problem:

$$\begin{aligned} &\min_{\mathbf{Z}} \sum_i \left\| \mathbf{z}_i - \frac{1}{2} \sum_j (\mathbf{D}^{-1}(\mathbf{S}^T + \mathbf{S})_{ij}\mathbf{z}_j) \right\|_2^2 \\ &\Rightarrow \min_{\mathbf{Z}} tr(\mathbf{Z}\,\mathbf{D}^{\frac{1}{2}}(\mathbf{I} - \tilde{\mathbf{S}})^2\mathbf{D}^{\frac{1}{2}}\mathbf{Z}^T) \end{aligned} \tag{2.12}$$

where $tr(.)$ denotes the trace of a matrix.

The optimal solution is given by $\mathbf{Z}^* = \mathbf{G}^T\mathbf{D}^{-\frac{1}{2}}$ where $\tilde{\mathbf{S}} = \frac{1}{2}(\mathbf{S}^\mathbf{T} + \mathbf{S})\,\mathbf{D}^{-1}$, and $\mathbf{G}$ is given by the eigenvectors of $(\mathbf{I} - \tilde{\mathbf{S}})$ associated with its smallest eignevalues.

2.3/ Semi-supervised Graph-Based Manifold Learning

Supervised Learning and Unsupervised Learning are two traditional techniques in machine learning domain. When we talk about Supervised Learning, it means that each of data sample consists of some input data and a corresponding output value or label. It focuses on constructing a regressor or classifier which estimate the output values or labels for unseen inputs as truely as possible. In Unsupervised Learning domain, we should try to deduce a few of underlying structures from the inputs instead of specific output value provided. One of examples in Unsupervised Learning domain is clustering, it focuses on deducing a mapping from given inputs to group into the same cluster. Semi-supervised Learning is to combine supervised and unsupervised learning in machine learning field. Semi-supervised classification learning aims on training classifier via unlabeled and labeled data. It avoids to lots of experienced human annotators working ahead of schedule.

Semi-supervised Graph-Based Manifold Learning is one of the most important branches in Semi-supervised Learning domain. We thought that the sample data is embedded within a low dimensional manifold as a graph structure. Furthermore, every sample data is regarded as a vertex in weighted graph. It includes a measure of similarity

among different vertices. In general, how to solve Semi-supervised Learning problem via Graph-based Manifold Learning approaches need to complete a series of steps in the next: First and foremost, graph construction, such as k Nearest Neighbors Graph or ϵ-neighborhoods Graph. Moreover, give several of labels in a set of nodes. The last, to classify labels on the unlabeled nodes in the whole graph. Most of Semi-supervised Graph-Based Manifold Learning algorithms happen in the first and third step above, which are Graph Structure and extracting feature from labeled to classify unlabeled samples.

We also realize that why Semi-supervised Graph-Based Manifold Learning methods are attractive the researchers to develop related fields as following. First of all, **BEING GENERIC.** Categories of data naturally constructed by graphs, e.g. Internet Web, Social Media, Protein Structure and Communication Networks. Moreover, **EFFECTIVE.** Several of application backgrounds have already demonstrated the effective of graph-based manifold learning methods outperform semi-supervised and supervised learning techniques fields, such as face recognition, object recognition, text classification and so on. The third, **SCALABILITY.** Many application domains need to handle large scale dataset, especially for the computing ability is tremendously increasing since GPU-based deep learning technique developing. Fortunately, Graph-based techniques can be easily parallelized processing and to generate large scale graphs. Finally, **CONVEXITY.** Graph-based Manifold Learning on Semi-supervised machine learning methods usually need to consider that optimizing convex objective after obtaining objective function, and Graph-based Manifold Learning algorithms usually could be ideal optimization result.

Categories of selected topics in Semi-supervised Graph-Based Manifold Learning will be introduced as following.

We will introduce some graph-based semi-supervised inference that can be transductive methods or inductive methods. The goal of a transductive algorithm is to learn a function that is able to predict the labels for only the unlabeled data. The inductive methods use the graph structure to estimate a function which can then be applied to new data instances.

Semi-Supervised Learning Using Gaussian Fields and Harmonic Functions(GFHF) [134] is one of the typical algorithms in transductive semi-supervised methods that based on a Gaussian random field model. GFHF uzilizes Gaussian fields on the continuous state space, instead of random fields on a discrete label set. The result classification methods for Gaussian fields could be regarded as a form of k Nearest Neighbor (KNN) method, it means that the nearest labeled samples are obtained in terms of random walk over graph. The algorithms introduced here have relation with spectral graph, random walks, heat kernels, electric networks and normalized cuts. The GFHF minimize the objective function as following:

$$\min_{\mathbf{F}} \frac{1}{2} \sum_{i,j=1}^{N} \left\| \mathbf{F}_{i.} - \mathbf{F}_{j.} \right\|^2 S_{ij} + \lambda_{\infty} \sum_{i=1}^{l} ||\mathbf{F}_{i.} - \mathbf{Y}_{i.}||^2 \tag{2.13}$$

where $\mathbf{Y} = (\mathbf{Y}_l, \mathbf{Y}_u)^T \in {}^{(l+u)\times C}$ is a binary label matrix associated with the samples with $\mathbf{Y}(i, j) = 1$ if $\mathbf{x}_i$ has label $\mathbf{y}_j = j$; $\mathbf{Y}(i, j) = 0$, otherwise. We also define an unknown label matrix denoted by $\mathbf{F} = (\mathbf{F}_l, \mathbf{F}_u)^T \in {}^{(l+u)\times C}$. In a semi-supervised setting, $\mathbf{F}_l = \mathbf{Y}_l$ and $\mathbf{F}_u$ is unknown labels matrix. We denote $\mathbf{F}_i$ and $\mathbf{Y}_i$ as the ith row of$\mathbf{F}_{i.}$ and $\mathbf{Y}_{i.}$, respectively.

Since λ_∞ is a large number such that $\sum_{i=1}^{l} \|\mathbf{F}_{i.} - \mathbf{Y}_{i.}\|^2 \Rightarrow 0$, or $\mathbf{F}_{i.}$=$\mathbf{Y}_{i.}$, Eq. 2.13 could be transformed into:

$$\min_{\mathbf{F}} trace\left(\mathbf{F}^T\mathbf{L}\mathbf{F}\right) \; s.t. \; \mathbf{F}_l = \mathbf{Y}_l \tag{2.14}$$

The goal is to derive the labels of unlabeled samples $\mathbf{F}_u$. It can be shown that the matrix of unknown labels are given by:

$$\mathbf{F}_u = -\mathbf{L}_{uu}^{-1}\mathbf{L}_{ul}\mathbf{Y}_l \in \mathbb{R}^{u\times C} \tag{2.15}$$

where $\mathbf{L}_{uu}$ and $\mathbf{L}_{ul}$ come from Laplacian matrix $\mathbf{L} = \begin{pmatrix} \mathbf{L}_{ll} & \mathbf{L}_{lu} \\ \mathbf{L}_{ul} & \mathbf{L}_{uu} \end{pmatrix} \in \mathbb{R}^{N\times N}$ that $\mathbf{L}_{ll} \in \mathbb{R}^{l\times l}$ and $\mathbf{L}_{uu} \in \mathbb{R}^{u\times u}$.

Local and Global Consistency (LGC) [132] is an optimal method for semi-supervised transductive learning. LGC aims on obtaining a classifying function, that is sufficiently smooth with respect to the intrinsic structure collectively revealed by known labeled and unlabeled points. LGC minimizes the objective function as following:

$$\min_{\mathbf{F}} \frac{1}{2}\sum_{i,j=1}^{N}\left\|\frac{\mathbf{F}_i}{\sqrt{D_{ii}}} - \frac{\mathbf{F}_{j.}}{\sqrt{D_{jj}}}\right\|^2 S_{ij} + \lambda\sum_{i=1}^{n}\|\mathbf{F}_{i.} - \mathbf{Y}_{i.}\|^2 \tag{2.16}$$

where $\mathbf{D}$ is a diagonal matrix that $\mathbf{D} = diag(\mathbf{S}\cdot\mathbf{1})$. $\mathbf{1}$ means the vector with all 1 entries. $\lambda > 0$ is the regularization parameter.

Eq. 2.16 could be transformed into:

$$\min_{\mathbf{F}}\left[\text{trace}\left(\mathbf{F}^T\widehat{\mathbf{L}}\mathbf{F}\right) + \mu\text{trace}\left((\mathbf{F}-\mathbf{Y})^T(\mathbf{F}-\mathbf{Y})\right)\right] \tag{2.17}$$

where the normalized Laplacian $\widehat{\mathbf{L}}$ is defined by $\widehat{\mathbf{L}} = \mathbf{I} - \mathbf{D}^{-\frac{1}{2}}\mathbf{S}\mathbf{D}^{-\frac{1}{2}}$. Eq. 2.17 could be solved by closed-form solution as follow:

$$\mathbf{F} = (\mathbf{I} + \frac{\hat{\mathbf{L}}}{\mu})^{-1}\mathbf{Y} \tag{2.18}$$

Recently, the Graph-based transductive semi-supervised methods always attract researchers. Predicting Labels And Neighbors with Embeddings Transductively Or Inductively from Data (Planetoid) [122] propose a novel graph-based semi-supervised learning framework. The ebmedding of instance in Planetoid merges context and class label of intance in the graph. In the next, to joint the hidden layers and embedding of the classifier. Then, to input a softmax layer for prediction. Planetoid obtains a good performance on distantly supervised entity extraction, entity classification, and text classification experiments. The Graph Convolutional Networks [52] propose a deep neural network approach for semi-supervised learning on graph-structured data. This approach provides a deep

architecture in order to learn the soft label in a transductive setting. We will introduce more details about related topics in Subsection 2.7.1. In [115], Graph-based Transductive Learning is used for brain disorder disease with multi-modal transductive semi-supervised classification learning method. Motivated from the sparse representation of graph, Minimum Tree Cut(MTC) [128] uzilize a spanning tree to approximate. It exploits a linear-time method to label the tree so that cut size of the tree minimizes. In [49], a scalable graph-based Semi-supervised Learning framework uzilizes sparse Bayesian model for defining the graph-based sparse prior. In graph-based semi-supervised inductive methods, Manifold Regularization [3] is the representative example of this class of methods. Manifold Regularization (MR) proposed a data-dependent regularization framework. It aims on exploiting non-Euclidean of the probability distribution, which to uzilize reproducing kernel Hilbert spaces to prove novel representer theorems. MR extends Support Vector Mechine (SVM) and ridge regression to Laplacian Support Vector Machine (LapSVM) and Laplacian Regularized Least Squares (LapRLS). We explain LapRLS and LapSVM as examples to review Manifold Regularization for semi-supervised learning inductive applications.

Let $\mathbf{f}(\cdot)$ be a function which projects the input to the output. Every Mercer kernel $K : \mathbf{X}\times\mathbf{X} \rightarrow \mathbb{R}$ has an associated RKHS ($\mathcal{H}_K$) of functions $\mathbf{X} \rightarrow \mathbb{R}$ with the corresponding norm ($\|\cdot\|_K$). $(\mathbf{x}_i, y_i), i = 1, 2, \cdots, l$ means labeled examples set, where l means labeled samples and u means unlabeled samples. The optimization problem for Laplacian Regularized Least Squares (LapRLS) is given by:

$$f^*(x) = \min_{f\in\mathcal{H}_K} \frac{1}{l}\sum_{i=1}^{l}\left(\mathbf{y}_i - f\left(\mathbf{x}_i\right)\right)^2 + \gamma_A\left\|f\right\|_K^2 + \frac{\gamma_I}{\left(u+l\right)^2}\mathbf{f}^T\mathbf{L}\mathbf{f} \tag{2.19}$$

$$f^*(x) = \sum_{i=1}^{l+u}\alpha_i{}^*\cdot K(\mathbf{x}, \mathbf{x}_i) \tag{2.20}$$

where γ_A and γ_I are hyperparameters that determine the relative importance of these penalties, $\left\|f\right\|_K^2$ is the regularizer that ensures the smoothness of possible solutions. $\mathbf{L}$ is laplacian matrix. The minimizer of above optimization problem accepts an expansion as Eq. 2.20, where $\alpha = [\alpha_1, \alpha_2, \cdots, \alpha_{l+u}]^T$ and $\mathbf{K} \in \mathbb{R}^{l\times l}$ is classical Gram matrix that $K_{ij} = K(\mathbf{x}_i, \mathbf{x}_j)$. According to the details of [3],

$$\alpha^* = \left(\mathbf{J}\cdot\mathbf{K} + \gamma_A l\cdot\mathbf{I} + \frac{\gamma_I\cdot l}{\left(u+l\right)^2}\cdot\mathbf{L}\cdot\mathbf{K}\right)^{-1}\cdot\mathbf{Y} \tag{2.21}$$

where $\mathbf{J}$ is $(l+u)\times(l+u)$ diagonal matrix given by $\mathbf{J} = diag(\underbrace{1, \cdots, 1}_{l}, \underbrace{0, \cdots, 0}_{u}) \in \mathbb{R}^{(l+u)\times(l+u)}$ and $\mathbf{Y}$ is $(l+u)$ dimensional label vector that $\mathbf{Y} = [y_1, \cdots, y_l, 0, \cdots, 0]$.

Using a hinge loss instead of the square loss (box shown in Eq. 2.22), LapSVM solves optimization problem as followings:

$$f^*(x) = \min_{f \in \mathcal{H}_K} \frac{1}{l} \sum_{i=1}^{l} \boxed{(1 - y_i \cdot f(\mathbf{x}_i))_+} + \gamma_A \left\|f\right\|_K^2 + \frac{\gamma_I}{(u+l)^2} \mathbf{f}^T \mathbf{L} \mathbf{f} \tag{2.22}$$

$$f^*(x) = \sum_{i=1}^{l+u} \alpha_i^* \cdot K(\mathbf{x}, \mathbf{x}_i) \tag{2.23}$$

Unlike LapRLS, we do not have closed form solution for α_i in LapSVM. It could be implemented by using SVM [16] solvers the quadratic form. [3] could find the details.

A series of graph-based semi-supervised inductive methods have been proposed recent years [91] [32]. Inductive methods GraphSAGE presented by Hamilton et al [34], which provides categories of methods such as GraphSAGE-LSTM (aggregating by feeding the neighborhood features into an Long short-term memory (LSTM) [44]), GraphSAGE-GCN (which extends a graph convolution-style operation to the inductive setting), GraphSAGE-pool (taking the elementwise maximization operation of feature vectors transformed by a shared nonlinear multilayer perceptron) and GraphSAGE-mean (taking the elementwise mean value of feature vectors). Graph Attention Networks (GATs) [106] is also one of novel neural network architectures. Besides, categories of algorithms are used for two types of semi-supervised learning models: transductive learning models and inductive learning models [107] [106] [122] [90].

2.4/ Graph-based Label Propagation

Graph-based Label Propagation (LP) has been used for categories of real-world applications successfully. Based on the intrinsic geometry relationships of samples data, Label Propagation is a processing that propagating label information from labeled samples data to unlabeled data. The performance of LP is via two terms that manifold smoothness term makes Label Propagation to decide theh sample labels via label information from neighbours and label fitness term. Label fitness term is used for meaturing the uncertain soft labels.

Existing graph based LP algorithms usually can be divided into two categories that are transductive and inductive algorithms. The notion of transductive and inductive label propagation methods is similar as the classification ways of the Semi-supervised Graph-Based Manifold Learning (Subsection 2.3). In transductive label propagation field, we focus on classifing a particular set of sample data points rather than a general decision function for unseen examples classification problem, which means inductive label propagation field need to solve unseen examples.

One of the most typical algorithms in Graph-based transductive Label Propagation method is Gaussian Fields and Harmonic Function (GFHF) [134], that we have already introduced this algorithm in Subsection 2.3. In [134], GFHF algorithm uzilizes graph-based method to model the similarity between pairs of examples. This is called graph transduction technique. The other popular label embedding based transductive algorithms include Learning with Local and Global Consistency (LGC) [132], Special Label Propagation (SLP) [78], Linear Neighborhood Propagation (LNP) [110], Robust multi-class graph transduction (RMGT) [67] and Positive and Negative Label Propagation (PN-LP) [137].

Flexible Manifold Embedding (FME) [79] is one of representative Graph-based Inductive Label Propagation algorithms in recent years. FME can effectively utilize manifold structure from both labeled and unlabeled data. In addition, a simplified version of FME (called FME/U) for unsupervised dimensionality reduction. We can regared FME as a framework that merges LapRLS [3] and LGC [132] for solving inductive problems. We define that sample set $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_l, \mathbf{x}_{l+1}, \cdots, \mathbf{x}_{l+u}] \in \mathbb{R}^{d \times N}$, a binary label matrix $\mathbf{Y} \in \mathbb{B}^{N \times C}$ with $Y_{ij} = 1$ if $\mathbf{x}_i$ has label $y_i = j$; $Y_{ij} = 0$, otherwise. $\mathbf{F} \in \mathbb{R}^{N \times C}$ is prediction labels. The Flexible Manifold Embedding minimizes the following criterion:

$$\begin{aligned}(\mathbf{F}^*, \mathbf{W}^*, \mathbf{b}^*) = \arg\min_{\mathbf{F},\mathbf{W},\mathbf{b}} & \operatorname{trace}((\mathbf{F}-\mathbf{Y})^T\mathbf{U}(\mathbf{F}-\mathbf{Y})) \\ & + \operatorname{trace}(\mathbf{F}^T\mathbf{L}\mathbf{F}) + \mu(\mathbf{W}^2 + \gamma\left\|\mathbf{X}^T\mathbf{W} + \mathbf{1}\cdot\mathbf{b}^T - \mathbf{F}\right\|^2)\end{aligned} \tag{2.24}$$

where $\mathbf{U} \in \mathbb{R}^{N \times N}$ is a diagonal matrix whose first l diagonal elements are set to 1 and the rest $u = N - l$ elements are set to 0 and μ and γ are two balance parameters. Define $\mathbf{L}$ is graph Laplacian matrix that $\mathbf{L} = \mathbf{D} - \mathbf{S}$. $\mathbf{0}, \mathbf{1} \in \mathbb{R}^{N \times 1}$ as a vector with all elements as 0 and a vector with all elements as 1, respectively. $\mathbf{b} \in \mathbb{R}^{C \times 1}$ is the bias term. $\mathbf{W} \in \mathbb{R}^{d \times C}$ is the projection matrix.

Vanish the derivative of Eq. 2.24 $(\mathbf{F}^*, \mathbf{W}^*, \mathbf{b}^*)$ with respect to $\mathbf{W}$ and $\mathbf{b}$. We can obtain $\mathbf{F}$ and $\mathbf{W}$. In the next, vanish the derivative with respect to $\mathbf{F}$. These functions are given as followings:

$$\mathbf{b} = \frac{1}{N}\left(\mathbf{F}^T\mathbf{1} - \mathbf{W}^T\mathbf{X}\mathbf{1}\right) \tag{2.25}$$

$$\mathbf{W} = \gamma\left(\gamma\mathbf{X}\mathbf{H}_c\mathbf{X}^T + \mathbf{I}\right)^{-1}\mathbf{X}\mathbf{H}_c\mathbf{F} = \mathbf{A}\mathbf{F} \tag{2.26}$$

$$\mathbf{F} = (\mathbf{U} + \mathbf{L} + \mu\gamma\mathbf{H}_c - \mu\gamma^2\mathbf{M})^{-1}\mathbf{U}\mathbf{Y} \tag{2.27}$$

where $\mathbf{A} = \gamma\left(\gamma\mathbf{X}\mathbf{H}_c\mathbf{X}^T + \mathbf{I}\right)^{-1}\mathbf{X}\mathbf{H}_c$ and $\mathbf{H}_c = \mathbf{I} - \frac{1}{N}\mathbf{1}\mathbf{1}^T$ are used for centering the data by subtracting the mean of the data, $\mathbf{X}_c = \mathbf{X}\mathbf{H}_c$.

Except for FME, the other representative inductive label propagation algorithms recently include Laplacian Linear Discriminant Analysis (LapLDA) [102], Embedded Label Propagation (ELP) [13]. All of ELP, LapLDA and FME could solve theh out-of-sample problem via learning linear mapping explicitly. ELP and FME add regressive error term to encode the mismatch which are different from LapLDA. Consider that LapLda, ELP and FME are linear projection algorithms. So theh dimensionality of data will effect the computing. It is hard to deal with high-dimensional sample data for these methods.

2.5/ Graph-Based Dimensionality Reduction

Dealing with high-dimensional data is a difficult problem in several fields such as pattern recognition, machine learning, and computer vision. The goal of dimensionality reduction is to map high-dimensional data into lower dimensional subspace without losing discriminant information. Numerous dimensionality reduction approaches have been proposed in the past several decades. Principal Component Analysis (PCA) [105] and Linear Discriminant Analysis (LDA) [1] are well-known as linear dimensionality reduction methods.

Let us also consider a linear transformation mapping the original d-dimensional image space into m-dimensional feature space, where $m \ll d$. PCA learns a projection matrix such that the variance of low-dimensional data is maximized.

$$\mathbf{W}_{\text{opt}} = \arg\max_{\mathbf{W}} \left|\mathbf{W}^T \mathbf{S}_T \mathbf{W}\right| = [\mathbf{w}_1 \ \mathbf{w}_2 \ \cdots \ \mathbf{w}_m]$$
$$\mathbf{S}_T = \sum_{i=1}^{N} (\mathbf{x}_i - \mu)(\mathbf{x}_i - \mu)^T \tag{2.28}$$

where $\mathbf{W} \in \mathbb{R}^{d \times m}$ is a matrix with orthonormal columns. The total scatter matrix $\mathbf{S}_T$ is the mean of all samples. $\mathbf{w}_i \ (i = 1, 2, \cdots, m)$ is the set of d-dimensional eigenvectors of $\mathbf{S}_T$ corresponding to the m largest eigenvalues. The new feature vectors $\mathbf{y}_i \in \mathbb{R}^m$ are defined by the following linear transformation that $\mathbf{y}_i = \mathbf{W}^T \mathbf{x}_i \ (i = 1, 2, \cdots, N)$.

The disadvantage of unsupervised algorithms for dimensionality reduction is not to uzilize label information when performing works of classification. Due to the training sample data are labeled, supervised dimensionality reduction will uzilize label information for developing efficient and better algorithms, and applying on categories of backgrounds, e.g., text classification, handprint classification and face recognition. LDA maps the data into low-dimensional subspace, which to maxmize the ratio of between-class to within-class distance. The object function of LDA could be solved by eigendecomposition on the scatter matrices to achieve the maximum discrimination.

The classical dimensionality reduction techniques are to discover the data structure lying on linear subspace of high-dimensional input sample data. In real-world, categories of datasets contain nonlinear structures. They are invisible to such as PCA or LDA classical dimensionality reduction methods. Graph-base Manifold Learning algorithms for dimensionality reduction came out.

Graph-based Manifold learning called Locality Preserving Projections (LPP) [40] that is a linear technique has been proposed for dimensionality reduction that preserves local relationships within the data set and uncovers its essential manifold structure. LPP algorithm finds a transformation matrix $\mathbf{W} \in \mathbb{R}^{d \times m}$ that maps N original samples to a set of low-dimensional data points ($\mathbf{x}_i \rightarrow \mathbf{y}_i = \mathbf{W}^T \mathbf{x}_i$). LPP method is of particular applicability in the special case where $\{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N\} \in \mathcal{M}$ and $\mathcal{M}$ is a nonlinear manifold embedded in $\mathbb{R}^m$. The linear transformation $\mathbf{W}$ can be obtained by minimizing an objective function as follows:

$$\min_{\mathbf{W}} \sum_{i,j=1}^{n} \left\|\mathbf{y}_i - \mathbf{y}_j\right\|^2 \mathbf{S}(i, j) \tag{2.29}$$

The weight matrix $\mathbf{S}$ (called heat kernel) is constructed through the nearest neighbor graph.

The minimization problem can be converted to solve a generalized eigenvalue problem as follows:

$$\mathbf{X}\mathbf{L}\mathbf{X}^T\mathbf{W} = \lambda \mathbf{X}\mathbf{D}\mathbf{X}^T\mathbf{W} \tag{2.30}$$

where the classic Laplacian matrix is given by $\mathbf{L} = \mathbf{D} - \mathbf{S}$ where $\mathbf{D}$ is a diagonal matrix whose elements are the row or column (since the weight matrix $\mathbf{S}$ is symmetric) sums

of $\mathbf{S}$ matrix. Besides, we could choose to utilize the normalized Laplacian matrix $\mathbf{L}$ is defined by $\mathbf{L} = \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{S}\mathbf{D}^{1/2}$ where $\mathbf{I}$ denotes the identity matrix. Let the column vectors $\{\mathbf{w}_0, \mathbf{w}_1, \cdots, \mathbf{w}_{m-1}\}$ be the solutions of Eq. 2.30, ordered according to their eigenvalues $\lambda_0 < \lambda_1 < \cdots < \lambda_{m-1}$. Thus, the embedding is as follows: $\mathbf{x}_i \rightarrow \mathbf{y}_i = \mathbf{W}^T\mathbf{x}_i$.

After a series of research in the recent years, Graph-based dimensionality reduction methods have been developed with tremendous achievements.The variants of locality preserving projection algorithms also have been introduced to resolve dimensionality reduction problems. Supervised Locality Preserving Projection [131] is a variant of Locality Preserving Projection, where the known class labels of the data points are used. Unlike LPP, Supervised Locality Preserving Projection projects high-dimensional data to the embedded low space taking class membership relations into account. This allows obtaining well-separated clusters in the embedded space. It is worthwhile to highlight the discriminant power of Supervised Locality Preserving Projection by using class information besides inheriting the properties of LPP. Therefore, Supervised Locality Preserving Projection demonstrates powerful recognition performance when applied to some pattern recognition tasks. W. Yang et al. [120] proposed a Graph-based Subspace Semi-supervised Learning Framework(SSLF) for dimensionality reduction. It includes with three Semi-supervised dimensionality reduction methods: Subspace Semi-supervised Locality Preserving Projection (SSLPP), Subspace Semi-supervised Marginal Fisher Analysis (SSMFA) and Subspace Semi-supervised Linear Discriminant Analysis (SSLDA). SSLF uzilize a mapping subspace via applying supervised learning method and embedding the unlabeled and labeled data into subspace. In the subspace, to ensure the homogeneous points are as close as together and heterogeneous points are apart from each other. In the end, the weight matrix is obtained by the projection points in subspace and the labels. A two-dimensional extension of locality preserving projections (2DLPP) is proposed by S. Chen et al. [14]. It overcomes the disadvantage of LPP that the singularity of matrix when the dimensionality of matrix is high. 2DLPP aims at computing based on two dimension image matrices, which is different from LPP that based on $1D$ vectors. The other two linear projection methods for dimensionality reduction that 2DPCA [119] and 2DLDA [61] preserve the Euclidean structure of image space, while 2DLPP finds an embedding that preserves local information and detects the intrinsic image manifold structure. As the one of authors of LPP, X. He proposed Orthogonal Locality Preserving Projection (OLPP) [9]. For overcoming the weakness of nonorthogonal LPP, which is hard to reconstruct the data, OLPP obtains orthogonal basis functions so that OLPP owns better ablility of locality preserving that LPP. Besides, discriminating power is related to the ablity of locality preserving. It results in OLPP is more discriminative than LPP. Kernel locality preserving projections [15] utilizes the nonlinear kernel mapping is used to map the data into an implicit feature space, which is successfully used in Support Vector Machine(SVM), and then a linear transformation is performed to preserve within-class geometric structures in implicit feature space. Thus, it can gain a nonlinear subspace that can approximate the intrinsic geometric structure of the face manifold.

2.6/ Graph-based Feature Selection

The revelant information and features effect the result of pattern recognition in the real-world. For solving this problem, Feature Selection is employed to select features and reduce dimensionality. It aims to extract subset of relevant features from raw samples. Fea-

ture Selection obtains better learning accuracy, low computing and model interpretablity. Fig. 2.2 descripts the basic processing framework of Feature Selection for classification task. Usually, the tranditional categorization of Feature Selection algorithms are supervised, semi-supervised and unsupervised learning methods via class labels in the classification problems. Furthermore, under the data feature perpective, the categorization of Feature Selection algorithms could be classified into flat features, structured features and streaming features. Mostly of Graph-based Manifold Learning algorithms are developed in Flat features models. Filter models, Wrapper models and Embedding models are three mainly directions of Flat features models.

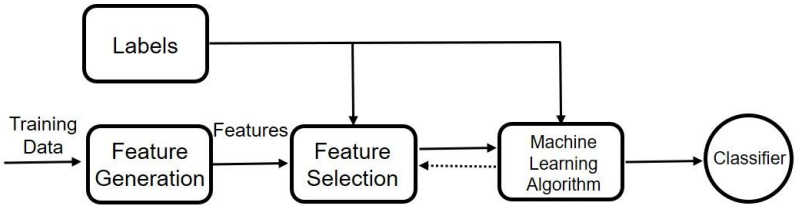


Figure 2.2: The basic processing framework of Feature Selection for classification task

Laplacian Score for Feature Selection [38] was the early stage and classical algorithm for jointly Graph-based Manifold Learning and Feature Selection method. Laplacian Score is a filter-based algorithm in Feature Selection, that focuses on independent of learning algorithm and could react in supervised and unsupervised setting. The basic idea of LSFS uzilizes locality preserving to evaluate the features. We regards L_r as r-th feature in each sample data, f_{ri} as i-th sample of r-th feature ($i = 1, \cdots, N$). The following function represents Laplacian Score of r-th feature:

$$\begin{aligned} \widetilde{\mathbf{f}}_r &= \mathbf{f}_r - \frac{\mathbf{f}_r^T \cdot \mathbf{D} \cdot \mathbf{1}}{\mathbf{1}^T \cdot \mathbf{D} \cdot \mathbf{1}} \cdot \mathbf{1} \\ L_r &= \frac{\widetilde{\mathbf{f}}_r^T \cdot \mathbf{L} \cdot \widetilde{\mathbf{f}}_r}{\widetilde{\mathbf{f}}_r^T \cdot \mathbf{D} \cdot \widetilde{\mathbf{f}}_r} \end{aligned} \tag{2.31}$$

where the matrix $\mathbf{L}$ is often called graph Laplacian that $\mathbf{L} = \mathbf{D} - \mathbf{S}$, $\mathbf{D} = \mathrm{diag}(\mathbf{S} \cdot \mathbf{1})$ and $\mathbf{1} = [1, \cdots, 1]^T$. $\mathbf{S}$ here is a weight matrix same as $\mathbf{S}$ in Eq. 2.29. Then the top k-th ranked features with high scores are selected as selected features for classification or clustering tasks.

Trace Ratio Criterion for Feature Selection [77] is another typical algorithm of Graph-based Manifold Learning on Feature Selection. It optimizes the subspace score and finds score is maximized. The Trace Ratio Criterion could be descripted as following:

$$\mathbf{y} = \mathbf{W}^T \mathbf{x} \tag{2.32}$$

where $\mathbf{W} \in \mathbb{R}^{d \times m} (m \ll d)$ means selection matrix. We regard $\mathbf{w}_i \in \mathbb{R}^{d \times 1}$ as $\mathbf{w}_i = \left[\underbrace{0, \cdots, 0}_{i-1}, 1, \underbrace{0, \cdots, 0}_{d-i}\right]^T$. So $\mathbf{W}$ could be rewritten as $\mathbf{W} = [\mathbf{w}_{K,1}, \mathbf{w}_{K,2}, \cdots, \mathbf{w}_{K,m}]$, where K is is a permutation of $\{1, 2, \cdots, d\}$. Denote data matrix by $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N] \in \mathbb{R}^{d \times N}$.

$\{\mathbf{F}_1, \mathbf{F}_2, \cdots, \mathbf{F}_d\}$ represents d features in each data $\mathbf{x}_i$. Feature subset ($[\mathbf{F}_{K,1}, \mathbf{F}_{K,2}, \cdots, \mathbf{F}_{K,m}]$) is denoted as $\Phi(K)$. Likewise, we regard $\mathbf{W}_K = [\mathbf{w}_{K,1}, \mathbf{w}_{K,2}, \cdots, \mathbf{w}_{K,m}]$, where $\mathbf{w}_i$ is defined as above.

If we suppose $\Phi(K)$ is selected, the sample data $\mathbf{x}$ is transformed into $\mathbf{y} = \mathbf{W}_K^T\mathbf{x}$. The score of feature subset $\Phi(K)$ is defined:

$$score\left(\Phi(K)\right) = \frac{\sum_{ij}\left\|\mathbf{y}_i - \mathbf{y}_j\right\|^2 (\mathbf{S}_b)_{ij}}{\sum_{ij}\left\|\mathbf{y}_i - \mathbf{y}_j\right\|^2 (\mathbf{S}_w)_{ij}} = \frac{\mathrm{tr}\left(\mathbf{W}_K^T\mathbf{X}\mathbf{L}_b\mathbf{X}^T\mathbf{W}_K\right)}{\mathrm{tr}\left(\mathbf{W}_K^T\mathbf{X}\mathbf{L}_w\mathbf{X}^T\mathbf{W}_K\right)} \tag{2.33}$$

where $(\mathbf{S}_w)_{ij}$ means within-class matrix and $(\mathbf{S}_b)_{ij}$ is between-class matrix. $\mathbf{L}_w$ and $\mathbf{L}_b$ are Laplacian matrices, that $\mathbf{L}_w = \mathbf{D}_w - \mathbf{S}_w$ and $(\mathbf{D}_w)_{ij} = \sum_j (\mathbf{S}_w)_{ij}$, $\mathbf{L}_b = \mathbf{D}_b - \mathbf{S}_b$ and $(\mathbf{D}_b)_{ij} = \sum_j (\mathbf{S}_b)_{ij}$.

The Trace Ratio Criterion for Feature Selection is to find the feature subset with maximum score via the optimization as following:

$$\Phi(K) = \arg\max_{\Phi(K)} \frac{\mathrm{tr}\left(\mathbf{W}_K^T\mathbf{X}\mathbf{L_b}\mathbf{X}^T\mathbf{W}_K\right)}{\mathrm{tr}\left(\mathbf{W}_K^T\mathbf{X}\mathbf{L_w}\mathbf{X}^T\mathbf{W}_K\right)} \tag{2.34}$$

[77] supports a graph-based Feature Selection Framework via Trace Radio Criterion as above shown. Besides, [77] provide an novel optimal solution for solving Eq. 2.34.

Joint $\ell_{2,1}$-norms Minimization and Graph-based for Feature Selection [76] uzilized regularization on least square regression for supervised and semi-supervised categorization tasks as following:

$$\min_{\mathbf{W}} J(\mathbf{W}) = \left\|\mathbf{X}^T\mathbf{W} - \mathbf{Y}\right\|_{2,1} + \gamma\|\mathbf{W}\|_{2,1} \tag{2.35}$$

where $\mathbf{W}$ is also called transformation matrix or selection matrix, but its dimension changes by $\mathbf{W} \in \mathbb{R}^{d\times C}$, class labels matrix $\mathbf{Y} = [\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_N]^T \in \mathbb{R}^{N\times C}$, samples data $\mathbf{X} \in \mathbb{R}^{d\times N}$.

Eq. 2.35 seems that solving this joint $\ell_{2,1}$-Norms problem is difficult as both of the terms are non-smooth. An efficient iterative algorithm to solve the optimization problem with proved convergence descripted in [76] detailedly. Once $\mathbf{W}$ is obained, it could be used for feature selection problem.

Categories of Unsupervised Learning feature selection methods are also proposed recently years [121] [10] [45] [86] [64]. Besides, Semi-supervised Learning feature selection methods include with [129] [36] [20] recently.

2.7/ The other State-of-the-art topics on Graphs

2.7.1/ Deep Learning on Graphs and Manifolds

In recent years, Deep Learning [56] is learnt complicated concepts from simple ones from multi-layer architechture, which Artificial Neural Networks (ANN) are one of the popular

techniques for realizating this deep multi-layer hierarchies. We also notice that the power of Deep Learning in serval fields, e.g. Speech, Image, Video signals and so on [54] [58]. These fields also exist underlying non-Eclidean structure. Deep Learning on Graphs and Manifolds are a field that to uzilize Deep Learning techniques to operate on Graph Theory and Manifold Learning [42] [19] [52] [6].

However, Deep Learning on Graphs and Manifolds topic still needs to solve several problems as follows. First and foremost, Deep Learning on Graphs and Manifolds is mainly to focus on solving theh static homogeneous graphs. Due to graph structures are usually fixed. Besides, edges and vertices of graph structure regard that they come from single source. But in some of real-world scenarios, these two assumptions are not existing usually. For example, in the social networks, someone could come into a network and quit the network at anytime. The other example is recommender system, different input forms (e.g. images, texts or videos) may mean same product which they just have different types. So Deep Learning on Graphs and Manifolds topic should consider to develop dynamic and heterogeneous graph stuctures. The second, a receptive field of vertices means vertices set. It includes the neighbors of central node and itself. However, the number of neighbors of central node are quite different, sometimes single and sometimes thousands. It needs to explore that to select a suitable receptive field of vertices.

With the success of Deep Learning, researchs have abtained categories of algorithms from recurrent networks, deep autoencoders and convolution networks to achieve the related algorithms about Deep Learning on Graphs and Manifolds. In the next, a brief review of Deep Learning on Graphs and Manifolds is introduced.

The pioneer work of Deep Learning on Graphs and Manifolds is proposed by Scarselli et al. [96] on the year of 2008, which it obtains Neurals Networks on Graphs. After a few of years sleeping, non-Euclidean Neural Networks field is developed via merging Machine Learning (Special for Deep Learning), Computer Vision and Natural Language Processing since [42]. Bruna et al. [42] uzilize Convolutional Neural Networks (CNN) on graphs in spectral domain. In [19], it proposes the CNNs in spectral graph that to design fast localized convolutional filters on graphs. The Graph Convolutional Networks (GCN) algorithm in [52] presents an approach for semi-supervised learning on graph-structured data. This approach provides a deep architecture in order to learn the soft label in a transductive setting. GCN brings a new round of fervor in Deep Learning on Graphs and Manifolds. Graph Auto-encoders [11], Graph Spatial-temporal Networks [97], Graph Generative Networks [124] and Graph Attention Networks [106] descript that to uzilize GCN as central algorithm to capture structural dependencies. Since above algorithms proposed, categories of representative extensions and improvements on spectral graph convolutional networks [62] [59] [34] [71] [80] [27]. The GCN algorithm is introduced as following.

Spectral convolutions on graph are defined as the multiplication of a signal $\mathbf{x} \in \mathbb{R}^N$ with a filter $\mathbf{g} = diag(\mathbf{w})$ where $\mathbf{w} \in \mathbb{R}^N$ is parameterized in Fourier domain that $\mathbf{g} \star \mathbf{x} = \mathbf{U}\,\mathbf{g}_w\,\mathbf{U}^T\,\mathbf{x}$. $\mathbf{U}^T\mathbf{x}$ and $\mathbf{g}_w$ could be regarded respectively as the graph Fourier transform of $\mathbf{x}$ and a function of $\mathbf{\Lambda}$ which is $\mathbf{g}(\mathbf{\Lambda})$, $\mathbf{\Lambda}$ being the diagonal matrix of the eigenvalues of the graph Laplacian $\mathbf{L}$. For solving the problem of expensive computing cost, [35] proposed Chebyshev polynomials to unfold $\mathbf{g}_w$: $\mathbf{g} \approx \sum_{k=0}^{K} \mathbf{w}_k\, T_k(\tilde{\mathbf{\Lambda}})$. Where T_k refer to Chebyshev polynomials. $T_k(x) = 2\,x\,T_{k-1}(x) - T_{k-2}(x)$, and $T_0(x) = 1$, $T_1(x) = x$. The largest eigenvalue of the Laplacian matrix $\mathbf{L}$ is denoted by $\lambda_{\max}$. The diagonal matrix $\tilde{\mathbf{\Lambda}}$

is given by $\tilde{\mathbf{\Lambda}} = \frac{2}{\lambda_{\max}}\mathbf{\Lambda} - \mathbf{I}$. $\mathbf{w}_k$ is Chebyshev coefficients.

So the spectral convolutions on graphs with a truncated expansion in terms of Chebyshev polynomials could be rewritten:

$$\mathbf{g} \star \mathbf{x} \approx \sum_{k=0}^{K} \mathbf{w}_k \cdot T_k(\tilde{\mathbf{L}})\,\mathbf{x} \tag{2.36}$$

where $\cdot$ can represent the scalar product, $\tilde{\mathbf{L}} = \frac{2}{\lambda_{\max}}\mathbf{L} - \mathbf{I}$ and $\mathbf{L} = \mathbf{I} - \mathbf{D}^{-\frac{1}{2}}\mathbf{S}\,\mathbf{D}^{-\frac{1}{2}}$. If we just expand 1_{st}-order polynomial in (2.36) to limit convolution operation, and according to further approximate in the linear formulation where K=1 and $\lambda_{\max} \approx 2$ in the (2.36), we get :

$$\begin{aligned}\mathbf{g}_w \star \mathbf{x} &\approx \mathbf{w}_0 \cdot \mathbf{x} + \mathbf{w}_1 \cdot (\frac{2}{\lambda_{\max}}\mathbf{L} - \mathbf{I})\,\mathbf{x} \\ &= \mathbf{w}_0 \cdot \mathbf{x} - \mathbf{w}_1 \cdot \mathbf{D}^{-\frac{1}{2}}\mathbf{S}\mathbf{D}^{-\frac{1}{2}}\mathbf{x}\end{aligned} \tag{2.37}$$

where $\mathbf{w}_0$ and $\mathbf{w}_1$ are free parameters. If we constrain the number of parameters to address over-fitting and to minimize the number of matrix multiplications, $\mathbf{w} = \mathbf{w}_0 = -\mathbf{w}_1$ could be used in (2.37). We get the first order graph convolution as: $\mathbf{g}_w \star \mathbf{x} = \mathbf{w} \cdot (\mathbf{I} + \mathbf{D}^{-\frac{1}{2}}\mathbf{S}\mathbf{D}^{-\frac{1}{2}})\,\mathbf{x}$.

We approximate $\lambda_{\max} \approx 2$ which means that the eigenvalues of $\mathbf{I} + \mathbf{D}^{-\frac{1}{2}}\mathbf{S}\mathbf{D}^{-\frac{1}{2}}$ are between 0 and 2. This operator will lead to numerical unstable. So the renormalization trick is introduced which replaces $\mathbf{I} + \mathbf{D}^{-\frac{1}{2}}\mathbf{S}\mathbf{D}^{-\frac{1}{2}}$ by $\hat{\mathbf{D}}^{-\frac{1}{2}}\hat{\mathbf{S}}\hat{\mathbf{D}}^{-\frac{1}{2}}$, where $\hat{\mathbf{S}} = \mathbf{S} + \mathbf{I}$ and $\hat{\mathbf{D}}_{ii} = \sum_j \hat{\mathbf{S}}_{ij}$. The Graph Convolutional Networks (GCN) algorithm in [52] presented a deep neural network approach for semi-supervised learning on graph-structured data. For semi-supervised label propagation, and with two layers, the GCN model has the following form:

$$\mathbf{E} = f(\mathbf{X}, \mathbf{A}) = \mathrm{softmax}\left(\hat{\mathbf{A}}\,\mathrm{ReLU}(\hat{\mathbf{A}}\,\mathbf{X}^T\,\mathbf{W}^{(0)})\mathbf{W}^{(1)}\right) \tag{2.38}$$

where $\mathbf{X}^T \in \mathbb{R}^{N\times d}$ denotes the input data with N samples and d dimensions, $\hat{\mathbf{A}}$ is a renormalized graph matrix $\hat{\mathbf{A}} = \hat{\mathbf{D}}^{-\frac{1}{2}}\hat{\mathbf{S}}\hat{\mathbf{D}}^{-\frac{1}{2}}$, $\mathbf{W}^{(0)} \in \mathbb{R}^{d\times H}$ is an input-to-hidden weight matrix for a hidden layer with H features, *ReLU ()* is the rectified linear activation function, and $\mathbf{W}^{(1)} \in \mathbb{R}^{H\times C}$ is a hidden-to-output weight matrix with C feature maps. C denotes the number of classes. $\mathrm{Softmax}$ denotes the softmax activation function. $\mathbf{E} \in \mathbb{R}^{N\times C}$ is the matrix of labels associated with all samples. The work in [52] estimates the model parameters $\mathbf{W}^{(0)}, \mathbf{W}^{(1)}, \ldots$ using deep learning tools in which the loss function is given by the cross-entropy error between the known labels and the output of the model.

Thanks to lots of researchers and companies make the code toolboxes open source, such as PyTorch has PyG library [25] for geometric deep learning extension and Graph Signal Processing(GSP) toolbox [101] [85] is based on Matlab and Python which implements categories of operation on graphs from simple ones like filtering to advanced ones like interpolation or graph learning, they help researchers to learn and implement Deep Learning on Graphs and Manifolds experiments faster.

2.7.2/ The Graph-based application on Learning to Hash

Nowadays, the tremendous data dimensionality and quantity increasing in the real world, especially the Internet and World Wide Web time coming. A series of Internet compa-

nies collect dramatic dataset increasing quickly, such as videos of Youtube, images of Instagram, messages of Wechat and blogs of Twitter. We have to face data searching accuracy, computing ability and search time cost.

Nearest Neighbor Searching [24] is one of the typical methods to focus on searching similar (even the same) samples from a specific dataset. The query sample defined by q and specific dataset defined by χ with N entries. The time complexity of Nearest Neighbor Searching is $O(N)$. Approximate Nearest Neighbor (ANN) Searching [123] came out to improve the time consuming when the number of data entries N is huge. Hashing theory was introduced into Nearest Neighbor Searching by Locality-Sensitive Hashing (LSH) [47] [31] since 1998. Locality-Sensitive Hashing obtains binary codes for high dimensional sample data points. Besides, it preserves the similarity of original data. We regard hashing algorithm as generating l-bits code, also meaning that own l hash functions. The original feature space could be partitioned into 1 code and 0 code via hash functions. We conclude that l hash functions bring l-bits code, then to partition feature space into 2^l parts, that is also regarded as hash buckets. In the end, the whole points fall into various hash buckets.

Learning to hash is a data dependent approach that different from such Locality-Sensitive Hashing algorithm as data independent. It focused on learning hash function from a particular data set. Learning to hash could make the result of nearest neighbor searching in hashing space is as close as possible to the result in original result. Besides, the search cost will also small that in original result. Motivated by Graph-based Manifold Learning algorithm (Laplacian Eigenmaps [2]) and Semantic Hashing [93], Spectral Hashing (SH) [116] brings Learning to Hash method proposed by Yair Weiss, Antonio Torralba and Rob Fergus since 2009. In SH, it performs to keep neighborhoods in space of input via hamming space, and desiring the hashing code to be uncorrelated and balanced. We suppose that the input sample data $\mathbf{x} \in \mathbb{R}^d$. We employ hash fuctions $H = \{h_1, h_2, \cdots, h_K\}$ for computing K-bit binary codes $\mathbf{y} = \{y_1, y_2, \cdots, y_K\}$ for $\mathbf{x}$ as $\mathbf{y} = \{y_1, y_2, \cdots, y_K\} = \{h_1(\mathbf{x}), h_2(\mathbf{x}), \cdots, h_K(\mathbf{x})\}$. So the $\mathbb{k}$-th bit could be computed for $y_{\mathbb{k}} = h_{\mathbb{k}}(\mathbf{x})$.

Define $\{\mathbf{y}_i\}_{i=1}^N$ as codewords list for N data points. $\mathbf{W} \in \mathbb{R}^{N \times N}$ means affinity matrix as the usual manifold learning algorithms, which $W(i,j) = \exp(-\left\|\mathbf{x}_i - \mathbf{x}_j\right\|^2 / \epsilon^2)$ (ϵ is a balance parameter). $\mathbf{L}$ is Laplacian matrix $\mathbf{L}$=$\mathbf{D}-\mathbf{W}$. The objective of spectral hashing is formulated as $h_k : \mathbb{R}^d \to \mathbb{B}^k$. We could project the input sample data $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^N \in \mathbb{R}^{d \times N}$ to binary codes as $\mathbf{Y} = \{h_1(\mathbf{X}), h_2(\mathbf{X}), \cdots, h_K(\mathbf{X})\} = H(\mathbf{X})$. The following is the objective function of SH:

$$\begin{aligned} & \min \sum_{ij} W_{ij} \left\|\mathbf{y}_i - \mathbf{y}_j\right\|^2 \\ & = \min trace\left(\mathbf{Y}^\top \mathbf{L} \mathbf{Y}\right) \\ & \text{subject to}: \quad \mathbf{Y}(i,j) \in \{-1, 1\}; \quad \mathbf{Y}^T \mathbf{Y} = \mathbf{I} \end{aligned} \tag{2.39}$$

where $\mathbf{Y}^T \mathbf{Y} = \mathbf{I}$ imposes orthogonality between hash bits to minimize the redundancy and $\mathbf{Y}^T \mathbf{1} = \mathbf{0}$ ensures that the hash bit reaches a balanced partitioning of the data. Eq. 2.40 is hard to optimize without removing $\mathbf{Y}(i,j) \in \{-1, 1\}$. After removing $\mathbf{Y}(i,j) \in \{-1, 1\}$, the solution is simply K eigenvectors of $\mathbf{D}-\mathbf{W}$ with minimal eigenvalue (after excluding the trivial eigenvector 1 which has eigenvalue 0).

With the help of Anchor Graph [68], anchor graph hashing (AGH) [69] uzilized Large Graph Construction algorithm for hashing searching. In Section 2.1, Eq. 2.5 descripts how to construct anchor graph. Finally, the anchor graph gives a data-to-data affinity matrix as $\mathbf{S} = \mathbf{Z}\mathbf{\Lambda}^{-1}\mathbf{Z}^{\top} \in \mathbb{R}^{N\times N}$. The definition of $\mathbf{Z}$ and $\mathbf{\Lambda}$ could be found near by Eq. 2.5. The final binary codes can be obtained through calculating the sign function over a spectral embedding as

$$\mathbf{Y} = \mathrm{sgn}(\mathbf{Z}\mathbf{\Lambda}^{\frac{1}{2}}\mathbf{V}\mathbf{\Sigma}^{\frac{1}{2}}) \tag{2.40}$$

where the matrix $\mathbf{V} = [\mathbf{v}_1, \cdots, \mathbf{v}_{\mathbb{k}}, \cdots, \mathbf{v}_K] \in \mathbb{R}^{k\times K}$ and $\mathbf{\Sigma} = diag(\sigma_1, \cdots, \sigma_{\mathbb{k}}, \cdots, \sigma_K) \in \mathbb{R}^{K\times K}$, that lower-case k is the number of anchors and $\{\mathbf{v}_{\mathbb{k}}, \sigma_{\mathbb{k}}\}$ are the eigenvector–eigenvalue pairs that Section 2.1 about Anchor Graph part and [69] show detail context.

According to the introduction of spectral hashing and anchor graph hashing, Learning to hash aims at learning a hash function ($\mathbf{y} = h(\mathbf{x})$) which to project the input points ($\mathbf{x}$) into compact code ($\mathbf{y}$). For a query q, to ensure the result of Nearest Neighbor search as close as the real result of nearest neighbor search. However, Learning to hash needs to face five problems [111] as followings. 1) what hash function $h(\mathbf{x})$ is adopted. 2) what similarity is provided in the input space. 3) what similarity in the coding space is used. 4) what loss function is chosen for the optimization objective. 5)what optimization technique is adopted. Most of Graph-based Learning to hash algorithms happen in the third problem that is similarity matrix problem, which is the most important link between Graph-based Manifold Learning and Learning to hash, and also the same problem need to solve in the two domains. Inductive Hashing on Manifolds (IHM) [98] [99] describes the inductive solution for out-of-sample via non-parametric manifold learning as the basis of hash function. Locally Linear Hashing (LLH) [48] focuses on preserving the locally linear manifold structures of high-dimensional data in a low-dimensional Hamming space.

Recently, several graph-based semi-supervised hashing methods [112] [113] [118] are proposed to take advantage of the information of both labeled and unlabeled data. The traditional Graph Cuts technique [5] is also used for Supervised Hashing Coding [28] [65]. A series of algorithms publish in the top conferences and journals which graph-based manifold learning technique on learning to hash. They show the novel, effective, and excellent experiment results in this domain.

2.8/ Conclusion

In this chapter, categories of Graph-based Manifold Learning techniques are introduced.

Specifically, Graph Construction and Large-Scale Graphs in Section 2.1, Unsupervised Graph-Based Manifold Learning in Section 2.2, Semi-supervised Graph-Based Manifold Learning in Section 2.3, Graph-based Label Propagation in Section 2.4, Graph-Based Dimensionality Reduction in Section 2.5, Graph-based Feature Selection in Section 2.6 and the other State-of-the-art topics on Graphs in Section 2.7 (e.g. Deep Learning on Graphs and Manifolds in Subsection 2.7.1, Graph-based application on Learning to Hash in Subsection 2.7.2).

Graph Construction is a crucial step in graph-based manifold learning which is the conversion of data into a weighted graph. Supervised, Semi-supervised and Unsupervised

Learning tasks can use graphs in order to estimate their model. Graph-based Label Propagation rely on the idea of building a graph whose nodes are data points (labeled and unlabeled) and edges represent similarities between points. Known labels are used to propagate information through the graph in order to label all nodes. Dimensionality reduction is to map high-dimensional data into lower dimensional subspace without losing discriminant information. Feature Selection approaches aim to select a small subset of features that minimize redundancy and maximize relevance to the target such as the class labels in classification. Deep Learning on Graphs and Manifolds is an umbrella term for emerging techniques attempting to generalize (structured) deep neural models to non-Euclidean domains that are graphs and manifolds. Learning to Hash is motivated by Graph-based Manifold Learning algorithm (Laplacian Eigenmaps algorithm [2]) and Semantic Hashing [93] for efficient nearest neighbor search in massive databases [116]..

In the following chapter, Chapter 3 discusses a novel nonlinear method called Flexible Discriminant graph-based Embedding with feature selection (FDEFS). Chapter 4 introduces a joint graph-based embedding and explicit feature weighting for getting a flexible and inductive nonlinear data representation on manifolds. Chapter 5 proposes an algorithm of graph convolution based semi-supervised Embedding (GCSE). Chapter 6 describes an Elastic graph-based embedding with deep architecture which deeply explores the structural information of the data. Finally, the conclusions of the work and some perspectives are given in the last Chapter 7.

II

Contributions

3

Learning a Semi-supervised Discriminant Graph-Based Embedding and Feature Selection via L21 constrained linear transform

In this chapter, we introduce a novel graph-based learning algorithm called flexible discriminant graph-based embedding feature selection (FDEFS), which is used for image categorization. We propose a novel graph-based embedding technology for reducing dimension of high dimensional data and for extracting the feature of data. Our proposed algorithm FDEFS includes Manifold Smoothness, Margin Discriminant Embedding and the Sparse Regression for feature selection, which could be used for semi-supervised and supervised learning setting. The $\ell_{2,1}$-norm regularization is being brought into for implicit feature weighting . We also utilize the sparse regression to select the feature on the original features and to provide an inductive projection model. The optimization shows efficient processing for the objective function. The experiments implement on various public datasets such as scene dataset, face dataset and object dataset demonstrate the success of proposed algorithm competing with the other compared algorithms.

3.1/ Review of Flexible Semi-Supervised Embedding

In [21], the authors introduced a flexible semi-supervised feature extraction method. It used a margin that is defined in two local neighborhoods. This is a sample-based margin that relies on intra-class and inter-class samples. Thus, for every labeled data sample $\mathbf{x}_i \in \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_l\}$ a margin is defined in the non-linear space.

Any labeled sample $\mathbf{x}_i (i \in C_k, k = 1, 2, \ldots, C)$ has l_k intra-class distances and $l - l_k$ inter-class distances. C_k is the set of indices of samples from the same class and $l_k = |C_k|$. Assuming that the non-linear subspace has one dimension, i.e., the whole dataset is projected onto an axis ($\mathbf{X}_l \Longrightarrow \mathbf{z}$). Two similarity matrices S^w and S^b associated with the labeled samples are introduced. These are defined as follows. $S^w_{\text{ij}} = 1/l_k$, if $i \in C_k$ and $j \in C_k$, $S^w_{\text{ij}} = 0$ otherwise. $S^b_{\text{ij}} = 1/(l - l_k)$, if $i \in C_k$ and $j \notin C_k$, $S^b_{\text{ij}} = 0$ otherwise. The average margin ξ for the whole labeled dataset is defined by:

$$\begin{aligned}\xi = \frac{1}{l}\sum_{i=1}^{l}\xi_i &= \frac{1}{l}\sum_{i=1}^{l}\Big(\sum_{j\notin C_k}\frac{\left\|z_i - z_j\right\|^2}{l-l_k} - \sum_{t\in C_k}\frac{\|z_i - z_t\|^2}{l_k}\Big)\\ &= \frac{1}{l}\sum_{i=1}^{l}\sum_{j=1}^{l}\Big(\left\|z_i - z_j\right\|^2\mathbf{S}_{ij}^{b} - \left\|z_i - z_j\right\|^2\mathbf{S}_{ij}^{w}\Big)\\ &= \frac{1}{l}(2\mathbf{z}^T\mathbf{D}_l\mathbf{z} - \mathbf{z}^T\mathbf{M}_l\mathbf{z}) \end{aligned} \tag{3.1}$$

where $\mathbf{D}_l = \mathbf{I} + \mathbf{D}^b \in \mathbb{R}^{l\times l}$ and $\mathbf{M}_l = 3\mathbf{I} + \mathbf{D}^b + \mathbf{S}^b + (\mathbf{S}^b)^T - 2\mathbf{S}^w \in \mathbb{R}^{l\times l}$. The diagonal matrix $\mathbf{D}^b$ with the entries being the column sums of $\mathbf{S}^b$. Besides, $\mathbf{z}^T\mathbf{S}^b\mathbf{z} = \mathbf{z}^T(\mathbf{S}^b)^T\mathbf{z}$ is used in Eq.(3.1). More details can be found in [21]. A non-linear projection is then obtained by maximizing the above margin. Thus, we have:

$$\mathbf{z} = \arg\min_{\mathbf{z}} \mathbf{z}^T\mathbf{M}_l\,\mathbf{z} \;\; s.t.\; \mathbf{z}^T\mathbf{D}_l\,\mathbf{z} = 1 \tag{3.2}$$

Since we have l labeled samples for training and u unlabeled samples for testing, the above criterion can be written as: $\mathbf{z} = \arg\min_{\mathbf{z}} \mathbf{z}^T\tilde{\mathbf{M}}_l\mathbf{z} \;\; s.t.\; \mathbf{z}^T\tilde{\mathbf{D}}_l\mathbf{z} = 1$, where $\tilde{\mathbf{M}}_l = (\mathbf{M}_l,\, 0\,;\, 0,\, 0) \in \mathbb{R}^{N\times N}$, $\tilde{\mathbf{D}}_l = (\mathbf{D}_l,\, 0\,;\, 0,\, 0) \in \mathbb{R}^{N\times N}$, the dimension of $\mathbf{z}$ is N.

Based on the analysis above, the work of [21] extended it to the multi-dimension embedding case using the following:

$$\begin{aligned}\min_{\mathbf{Z},\mathbf{W},\mathbf{b}} \; tr(\mathbf{Z}^T\mathbf{L}\mathbf{Z}) + \lambda\, tr(\mathbf{Z}^T\tilde{\mathbf{M}}_l\mathbf{Z}) + \mu(\,\|\mathbf{W}\|_F^2 + \gamma\left\|\mathbf{X}^T\mathbf{W} + \mathbf{1}\mathbf{b}^T - \mathbf{Z}\right\|_F^2)\\ s.t.\; \mathbf{Z}^T\tilde{\mathbf{D}}_l\mathbf{Z} = \mathbf{I}\end{aligned} \tag{3.3}$$

where λ, μ and γ are positive balance parameters, $\mathbf{Z} \in \mathbb{R}^{N\times N}$ denotes the non-linear embedding of all data samples and $\mathbf{I} \in \mathbb{R}^{N\times N}$ denotes the identity matrix. $\mathbf{L} = \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{S}\mathbf{D}^{1/2}$. The main strength of the model presented in Eq. (3.3) is the fact that the non-linear embedding and the linear model are simultaneously estimated.

3.2/ Overview of Proposed Approach

The FDEFS algorithm inherits the partly advantages of JELSR and [21], extends to semi-supervised and supervised learning setting. And provides an embedding framework with feature selection the feature selection of the original features. Moreover, FDEFS owns two methods to estimate the graph-based embedding data. The first is a non-linear projection of the available training data. The other method via the linear regressor model represented by the matrix **W** and the shift vector **b**, specially, this method could be used for yielding the possibility of the method to work with unseen samples. FDEFS maintains the margin-based discriminant embedding and manifold smoothness for increasing the image classification recognition rate.

3.2.1/ Model of the proposed method

The work of 3.3 in [21] proposed an inductive feature extraction method that retains the advantages of flexible manifold embedding and margin based discriminant embedding for the non-linear graph based embedding. With joint the advantages of JELSR and [21], we propose to minimize the following criterion under the constraint $\mathbf{Z}^T\tilde{\mathbf{D}}_l\mathbf{Z} = \mathbf{I}$: $\lambda\, tr(\mathbf{Z}^T\mathbf{M}_l\mathbf{Z})$

$$\begin{aligned} f(\mathbf{Z},\mathbf{W},\mathbf{b}) &= tr(\mathbf{Z}^T\mathbf{L}\mathbf{Z}) + \lambda\, tr(\mathbf{Z}^T\mathbf{M}_l\mathbf{Z}) + \mu\,(\|\mathbf{W}\|_{2,1} \\ &\quad +\gamma\left\|\mathbf{X}^T\mathbf{W} + \mathbf{1}\mathbf{b}^T - \mathbf{Z}\right\|_2^2) \\ &= tr(\mathbf{Z}^T\mathbf{L}_1\mathbf{Z}) + \mu\,(\|\mathbf{W}\|_{2,1} + \gamma\left\|\mathbf{X}^T\mathbf{W} + \mathbf{1}\mathbf{b}^T - \mathbf{Z}\right\|_2^2) \end{aligned} \tag{3.4}$$

where $\mathbf{L}_1 = \mathbf{L} + \lambda\,\widetilde{\mathbf{M}_l}$, and $\mathbf{1} = [1, 1, \cdots, 1]^T \in \mathbb{R}^{N\times 1}$. $\mathbf{L}$ is Laplacian Matrix that $\mathbf{L} = \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{S}\mathbf{D}^{1/2}$. $\mathbf{S}$ means the similarity matrix that is symmetric, $\mathbf{D}$ is a diagonal matrix whose elements are the row or column (since the matrix is symmetric) sums of $\mathbf{S}$ matrix, $\mathbf{I}$ denotes the identity matrix. $\widetilde{\mathbf{M}_l}$ from the concept of margin among classes has been already used for getting discriminant projections in the literature [21, 108]. Wang et al. [108] used a margin that is defined in two local neighborhoods. This is a sample-based margin that relies on a specific neighborhood size for intra-class and inter-class samples. In [21], the authors introduce a margin for the semi-supervised case.

$\mathbf{W} = [\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_m] \in \mathbb{R}^{d\times m}$ is the linear transform matrix, m is the dimensionality of embedding, and $\mathbf{Z} = [\mathbf{z}_1; \mathbf{z}_2; \ldots; \mathbf{z}_N] \in \mathbb{R}^{N\times m}$ denotes the data matrix of embedding (i.e., the non-linear projection of $\mathbf{X}$). The $\ell_{2,1}$ norm of $\mathbf{W}$ is given by $\|\mathbf{W}\|_{2,1} = \sum_{i=1}^{d}\|\hat{\mathbf{w}}_i\|_2$ where $\hat{\mathbf{w}}_i$ denotes the i_{th} row of $\mathbf{W}$. This norm promotes row sparsity of the linear transform $\mathbf{W}$. $\mathbf{b}$ is the bias vector. The regression model regresses the original samples $\mathbf{X}$ to the non-linear projection data $\mathbf{Z}$ by imposing that $\mathbf{Z} = \mathbf{X}^T\mathbf{W} + \mathbf{1}\mathbf{b}^T \in \mathbb{R}^{N\times d}$. In the above criterion, there are three balance parameters: μ, λ and γ.

In the above criterion, there are three different terms to be minimized. The first term ($tr(\mathbf{Z}^T\mathbf{L}\mathbf{Z})$) imposes locality preserving of the non-linear projection for manifold smoothness. Besides, the second term ($\lambda\, tr(\mathbf{Z}^T\mathbf{M}_l\mathbf{Z})$) is used for maximizing a margin via the label samples. The last term ($\|\mathbf{W}\|_{2,1} + \gamma\left\|\mathbf{X}^T\mathbf{W} + \mathbf{1}\mathbf{b}^T - \mathbf{Z}\right\|_2^2$) will get a linear regression model via $\ell_{2,1}$-norm regularization which the norm performs feature ranking and feature selection [46, 133].

3.2.2/ Solution to the proposed model

The processing of optimization steps are shown as following. At first, we try to vanish the derivatives of the objective function 3.4 with respect $\mathbf{W}$ and $\mathbf{b}$,

$$\begin{aligned} &\frac{\partial f}{\partial \mathbf{b}} = \mathbf{0} \Rightarrow \\ &\mathbf{b} = \frac{1}{N}(\mathbf{Z}^T\mathbf{1} - \mathbf{W}^T\mathbf{X}\mathbf{1}) \end{aligned} \tag{3.5}$$

$$\begin{aligned} &\frac{\partial f}{\partial \mathbf{W}} = \mathbf{0} \Rightarrow \\ &\mu\,[2\,\mathbf{U}\,\mathbf{W} + 2\,\gamma(\mathbf{X}\mathbf{X}^T\mathbf{W} + \mathbf{X}(\mathbf{1}\mathbf{b}^T - \mathbf{Z}))] = \mathbf{0} \end{aligned} \tag{3.6}$$

where $\mathbf{U}$ is a diagonal matrix whose diagonal elements are given by:

$$\mathbf{U}_{ii} = 1/(2 \cdot \|\hat{\mathbf{w}}_i\|_2) \quad (i = 1, 2, \ldots, d) \tag{3.7}$$

$\hat{\mathbf{w}}_i$ denotes the ith row of $\mathbf{W} \in \mathbb{R}^{d \times m}$. When $\mathbf{U}$ is fixed, the derivative $\frac{\partial f}{\partial \mathbf{W}}$ (3.6) can also be regarded as the derivative of the following objective function:

$$\mathcal{L}(\mathbf{Z}, \mathbf{W}, \mathbf{b}, \mathbf{U}) = tr(\mathbf{Z}^T \mathbf{L}_1 \mathbf{Z}) + \mu \left(tr(\mathbf{W}^T \mathbf{U} \mathbf{W}) + \gamma \left\| \mathbf{X}^T \mathbf{W} + \mathbf{1}\mathbf{b}^T - \mathbf{Z} \right\|_2^2\right) \tag{3.8}$$

Consequently, we will solve the following problem to approximate the solution to Eq.(3.4):

$$\begin{aligned} &\min_{\mathbf{Z},\mathbf{W},\mathbf{b},\mathbf{U}} \mathcal{L}(\mathbf{Z}, \mathbf{W}, \mathbf{b}, \mathbf{U}) = \\ &tr(\mathbf{Z}^T \mathbf{L}_1 \mathbf{Z}) + \mu \left(tr(\mathbf{W}^T \mathbf{U} \mathbf{W}) + \gamma \left\| \mathbf{X}^T \mathbf{W} + \mathbf{1}\mathbf{b}^T - \mathbf{Z} \right\|_2^2\right) \end{aligned} \tag{3.9}$$

In details, by inserting Eq.(3.5) into (3.6), we get:

$$\mathbf{U}\mathbf{W} + \gamma(\mathbf{X}\mathbf{X}^T\mathbf{W} + \mathbf{X}(\frac{1}{N}\mathbf{1}(\mathbf{1}^T\mathbf{Z} - \mathbf{1}^T\mathbf{X}^T\mathbf{W}) - \mathbf{Z})) = \mathbf{0} \tag{3.10}$$

By using $\mathbf{X}_c = \mathbf{X} \cdot \mathbf{H}_c$ with $\mathbf{H}_c = \mathbf{I} - \frac{1}{N}\mathbf{1} \cdot \mathbf{1}^T$ ($\mathbf{H}_c$ is the centring matrix), Eq. (3.10) can be written as:

$$\begin{aligned} &\mathbf{U}\,\mathbf{W} + \gamma(\mathbf{X}_c {\mathbf{X}_c}^T)\mathbf{W} = \gamma \mathbf{X}_c \mathbf{Z} \\ \Rightarrow\ &\mathbf{W} = (\mathbf{U} + \gamma \mathbf{X}_c {\mathbf{X}_c}^T)^{-1} \cdot \gamma \mathbf{X}_c \mathbf{Z} = \mathbf{A}_1 \mathbf{Z} \end{aligned} \tag{3.11}$$

where $\mathbf{A}_1 = \gamma(\mathbf{U} + \gamma \mathbf{X}_c {\mathbf{X}_c}^T)^{-1}\mathbf{X}_c$.

We have $\|\mathbf{W}\|_{2,1} = 2\, tr(\mathbf{W}^T\mathbf{U}\mathbf{W})$ when $\hat{\mathbf{w}}_i$ is not equal to zero. By plugging Eq.(3.5) and (3.11) into $\mathbf{X}^T\mathbf{W} + \mathbf{1}\mathbf{b}^T$, we get:

$$\begin{aligned} \mathbf{X}^T\mathbf{W} + \mathbf{1}\mathbf{b}^T &= \mathbf{X}^T\mathbf{A}_1\mathbf{Z} + \frac{1}{N}\mathbf{1}\mathbf{1}^T\mathbf{Z} - \frac{1}{N}\mathbf{1}\mathbf{1}^T\mathbf{Z}\mathbf{X}^T\mathbf{A}_1\mathbf{Z} \\ &= (\mathbf{I} - \frac{1}{N}\mathbf{1}\mathbf{1}^T)\mathbf{X}^T\mathbf{A}_1\mathbf{Z} + \frac{1}{N}\mathbf{1}\mathbf{1}^T\mathbf{Z} \\ &= \mathbf{H}_c\mathbf{X}^T\mathbf{A}_1\mathbf{Z} + \frac{1}{N}\mathbf{1}\mathbf{1}^T\mathbf{Z} \\ &= \mathbf{B}_1\mathbf{Z} \end{aligned} \tag{3.12}$$

where $\mathbf{B}_1 = \mathbf{H}_c\mathbf{X}^T\mathbf{A}_1 + \frac{1}{N}\mathbf{1}\mathbf{1}^T$.

By inserting Eq. (3.12) into Eq. (3.4), the latter becomes:

$$\begin{aligned} f(\mathbf{Z}) &= tr(\mathbf{Z}^T\mathbf{L}_1\mathbf{Z}) + 2\mu\, tr(\mathbf{Z}^T\mathbf{A}_1^T\mathbf{U}\mathbf{A}_1\mathbf{Z}) \\ &\quad + \mu\,\gamma\, tr((\mathbf{B}_1\mathbf{Z} - \mathbf{Z})^T(\mathbf{B}_1\mathbf{Z} - \mathbf{Z})) \\ &= tr(\mathbf{Z}^T(\mathbf{L}_1 + \mu\mathbf{A}_1^T\mathbf{U}\mathbf{A}_1 + \mu\,\gamma\,(\mathbf{B}_1 - \mathbf{I})^T(\mathbf{B}_1 - \mathbf{I}))\mathbf{Z}) \\ &= tr(\mathbf{Z}^T\hat{\mathbf{K}}\mathbf{Z}) \end{aligned} \tag{3.13}$$

where $\hat{\mathbf{K}} = \mathbf{L}_1 + \mu\mathbf{A}_1^T\mathbf{U}\mathbf{A}_1 + \mu\gamma(\mathbf{B}_1 - \mathbf{I})^T(\mathbf{B}_1 - \mathbf{I})$.

Thus, the non-linear embedding $\mathbf{Z}$ is estimated by minimizing the above criterion under the constraint $\mathbf{Z}^T\tilde{\mathbf{D}}_l\mathbf{Z} = \mathbf{I}$ that also used in the criterion (3.4):

$$f(\mathbf{Z}) = tr(\mathbf{Z}^T\hat{\mathbf{K}}\mathbf{Z}) \quad s.t.\ \mathbf{Z}^T\tilde{\mathbf{D}}_l\mathbf{Z} = \mathbf{I} \tag{3.14}$$

Therefore, $\mathbf{Z}$ can be solved by a generalized eigenvalue decomposition. Once $\mathbf{Z}$ is estimated, the linear model $\mathbf{W}$ and $\mathbf{b}$ will be given by (3.5) and (3.11).

In order to estimate $\mathbf{Z}$ one should solve the generalized eigen decomposition problem presented in (3.14). At this stage, this problem requires the matrix $\hat{\mathbf{K}} = \mathbf{L}_1 + \mu\mathbf{A}_1^T\mathbf{U}\mathbf{A}_1 + \mu\gamma(\mathbf{B}_1 - \mathbf{I})^T(\mathbf{B}_1 - \mathbf{I})$ to be known, which is not the case since the matrix $\mathbf{U}$ depends on the entries of $\mathbf{W}$ and this latter is unknown. Thus, we will adopt an iterative scheme in which the matrix $\mathbf{U}$ is initialized and then an estimate for $\mathbf{Z}$, $\mathbf{W}$, $\mathbf{b}$, and $\mathbf{U}$ will be computed.

At the beginning of the iterative process, by fixing the diagonal matrix $\mathbf{U}$, the remaining unknowns $\mathbf{Z}$, $\mathbf{W}$, and $\mathbf{b}$ can be estimated using closed form solutions given by Eqs. (3.14), (3.11), (3.5). Once $\mathbf{W}$ is known, the matrix $\mathbf{U}$ can be updated using Eq. (3.7). This process is repeated until convergence. The proof of convergence is given in the appendix.

The estimated $\mathbf{W}$ at convergence can be used for feature ranking and selection. The score of the rth feature is given by $score(r) = ||\mathbf{W}_{\mathbf{r}*}||_2, (r = 1, 2, \cdots, d)$, where $\mathbf{W}_{\mathbf{r}*}$ denotes the rth row of $\mathbf{W}$. These scores are used for ranking the original features as well as the rows of $\mathbf{W}$. Thus, we can select the most relevant features in original samples data matrix $\mathbf{X}$ and their corresponding rows in the estimated $\mathbf{W}$ matrix. An illustrative example (3.1) showing how the computed $\mathbf{W}$ can be used for feature ranking and selection. In our proposed algorithm, the final non-linear embedding matrix $\mathbf{Z}_s$ is given by $\mathbf{Z}_s = \mathbf{X}_s^T\mathbf{W}_s + \mathbf{b}^T$ where $\mathbf{X}_s$ and $\mathbf{W}_s$ are the selected matrices using the computed scores $||\mathbf{W}_{r*}||_{r=1}^d$.

The main steps that allow the simultaneous estimation of the non-linear embedding and its linear regression are given by **Algorithm 2.**

We emphasize the fact that the JELSR method does not exploit label information in order to compute the selection and the non-linear embedding. On the other hand, our proposed method exploits the label information in order to compute both the non-linear projection and the feature selection.

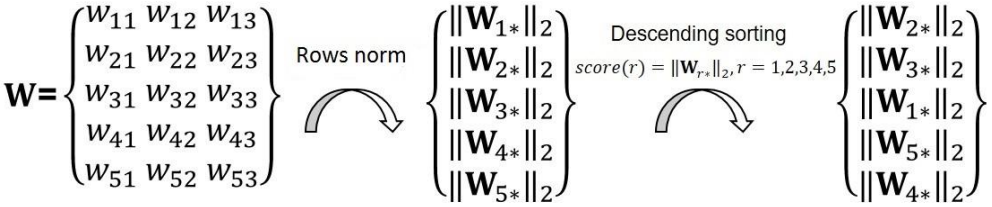


Figure 3.1: The figure illustrates a simple example in which the number of the original features is five, and the number of new features is three. Following the procedure described above the ranked features are (2, 3, 1, 5, 4).

3.2.3/ Convergence analysis

The proposed FDEFS algorithm uses an iterative scheme (third step in Algorithm 1) in order to minimize the objective function (3.4). In this section, we prove that the proposed iterative process will converge. First, we provide the following lemma [74]:

Lemma [1].

$$\frac{\|\mathbf{p}\|_2^2}{2 \times \|\mathbf{q}\|_2} - \|\mathbf{p}\|_2 \geq \frac{\|\mathbf{q}\|_2^2}{2 \times \|\mathbf{q}\|_2} - \|\mathbf{q}\|_2 \tag{3.15}$$

Algorithm 1 Flexible and Discriminant Non-linear Embedding with Feature Selection Algorithm (FDEFS)

Input: Data samples: $\mathbf{X}$; Initial matrix: $\mathbf{U}$=$\mathbf{I}_{\mathrm{d\times d}}$; Regularization parameters: μ, λ and γ; Number of selected feature (optional): s

Output: Non-linear embedding matrix $\mathbf{Z}_s$; linear transform $\mathbf{W}_s$ and $\mathbf{b}$;

1: Compute the sparse graph matrix $\mathbf{S}$ using Eq. (2.11);
2: Compute the normalized graph Laplacian matrix $\mathbf{L}$;
3: Iteratively update until convergence $\mathbf{Z}$, $\mathbf{W}$, $\mathbf{b}$ and $\mathbf{U}$ (in that order) using Eqs. (3.14), (3.11), (3.5), and (3.7), respectively. ;
4: Optionally, estimate the scores of all features using $score(r) = ||\mathbf{W}_{\mathbf{r}*}||_2, (r = 1, 2, \ldots, d)$. Sort these scores by descending order and select the s features that have the largest values. This ranking is used for selecting the original features in both the data matrix $\mathbf{X}$ and the estimated linear transform $\mathbf{W}$. Compute the final non-linear embedding matrix $\mathbf{Z}_s$ as $\mathbf{Z}_s = \mathbf{X}_s^T\mathbf{W}_s + \mathbf{1}\mathbf{b}^T$;

where $\mathbf{p}$ and $\mathbf{q}$ are two arbitrary nonzero vectors having the same dimension.

Proposition [1]. *The objective of the problem (3.4) in each iteration is non-increasing by using the optimization process in the third step of Algorithm I.*

Proof. We fix $\mathbf{U}$ as $\mathbf{U}^t$ in the tth iteration and compute the solution $\mathbf{Z}^{t+1}$, $\mathbf{W}^{t+1}$ and $\mathbf{b}^{t+1}$ at iteration $t+1$ using the Equations presented in step 3 in Algorithm 1. Since this step minimizes the objective function (3.9) for a fixed $\mathbf{U}$, the following inequality holds:

$$\mathcal{L}(\mathbf{Z}^{t+1}, \mathbf{W}^{t+1}, \mathbf{U}^t, \mathbf{b}^{t+1}) \leq \mathcal{L}(\mathbf{Z}^t, \mathbf{W}^t, \mathbf{U}^t, \mathbf{b}^t) \tag{3.16}$$

This yields:

$$\begin{aligned}\mathcal{L}(\mathbf{Z}^{t+1}, \mathbf{W}^{t+1}, \mathbf{U}^t, \mathbf{b}^{t+1}) + \mu\left\|\mathbf{W}^{t+1}\right\|_{2,1} - \mu\left\|\mathbf{W}^{t+1}\right\|_{2,1} \leq \\ \mathcal{L}(\mathbf{Z}^t, \mathbf{W}^t, \mathbf{U}^t, \mathbf{b}^t) + \mu\left\|\mathbf{W}^t\right\|_{2,1} - \mu\left\|\mathbf{W}^t\right\|_{2,1}\end{aligned} \tag{3.17}$$

We have: (i) $\frac{1}{2}\|\mathbf{W}\|_{2,1} = \frac{1}{2}\sum_{i=1}^{d}\|\hat{\mathbf{w}}_i\|_2$, (ii) $tr((\mathbf{W}^{t+1})^T\mathbf{U}^t\mathbf{W}^{t+1}) = \sum_{i=1}^{d}\frac{\|\widehat{\mathbf{w}}_i^{t+1}\|_2^2}{2\|\widehat{\mathbf{w}}_i^t\|_2}$ and (iii) $tr(\mathbf{W}^{t^T}\mathbf{U}^t\mathbf{W}^t) = \sum_{i=1}^{d}\frac{\|\widehat{\mathbf{w}}_i^t\|_2^2}{2\|\widehat{\mathbf{w}}_i^t\|_2}$ where $\hat{\mathbf{w}}_i$ denotes the ith row of $\mathbf{W}$.

By using these three expressions in the functional $\mathcal{L}$ (given by (3.9)) and in the inequality (3.17), the latter becomes

$$\begin{aligned}&tr((\mathbf{Z}^{t+1})^T\mathbf{L}_1\mathbf{Z}^{t+1}) + \mu(\left\|\mathbf{W}^{t+1}\right\|_{2,1} + \gamma\left\|\mathbf{X}^T\mathbf{W}^{t+1} + \mathbf{1}(\mathbf{b}^{t+1})^T - \mathbf{Z}^{t+1}\right\|_2^2) + \\ &\mu\sum_{i=1}^{d}(\frac{\|\widehat{\mathbf{w}}_i^{t+1}\|_2^2}{2\|\widehat{\mathbf{w}}_i^t\|_2} - \|\widehat{\mathbf{w}}_i^{t+1}\|_2) \leq \\ &tr((\mathbf{Z}^t)^T\mathbf{L}_1\mathbf{Z}^t) + \mu(\left\|\mathbf{W}^t\right\|_{2,1} + \gamma\left\|\mathbf{X}^T\mathbf{W}^t + \mathbf{1}(\mathbf{b}^t)^T - \mathbf{Z}^t\right\|_2^2) \\ &+\mu\sum_{i=1}^{d}(\frac{\|\widehat{\mathbf{w}}_i^t\|_2^2}{2\|\widehat{\mathbf{w}}_i^t\|_2} - \|\widehat{\mathbf{w}}_i^t\|_2)\end{aligned} \tag{3.18}$$

From Lemma (1), we have:

$$\mu \sum_{i=1}^{d} \left(\frac{\left\|\widehat{\mathbf{w}}_i^{t+1}\right\|_2^2}{2\left\|\widehat{\mathbf{w}}_i^{t}\right\|_2} - \left\|\widehat{\mathbf{w}}_i^{t+1}\right\|_2\right) \geq \mu \sum_{i=1}^{d} \left(\frac{\left\|\widehat{\mathbf{w}}_i^{t}\right\|_2^2}{2\left\|\widehat{\mathbf{w}}_i^{t}\right\|_2} - \left\|\widehat{\mathbf{w}}_i^{t}\right\|_2\right)$$

Based on the above inequality, the inequality (3.18) yields:

$$\begin{aligned} &tr((\mathbf{Z}^{t+1})^T \mathbf{L}_1 \mathbf{Z}^{t+1}) + \mu\left(\left\|\mathbf{W}^{t+1}\right\|_{2,1} + \gamma \left\|\mathbf{X}^T\mathbf{W}^{t+1} + \mathbf{1}(\mathbf{b}^{t+1})^T - \mathbf{Z}^{t+1}\right\|_2^2\right) \leq \\ &tr((\mathbf{Z}^{t})^T \mathbf{L}_1 \mathbf{Z}^{t}) + \mu\left(\left\|\mathbf{W}^{t}\right\|_{2,1} + \gamma \left\|\mathbf{X}^T\mathbf{W}^{t} + \mathbf{1}(\mathbf{b}^{t})^T - \mathbf{Z}^{t}\right\|_2^2\right) \end{aligned} \tag{3.19}$$

Thus, we have

$$f(\mathbf{Z}^{t+1}, \mathbf{W}^{t+1}, \mathbf{b}^{t+1}) \leq f(\mathbf{Z}^{t}, \mathbf{W}^{t}, \mathbf{b}^{t}) \tag{3.20}$$

This indicates that the objective function (3.4) is monotonically decreasing in each iteration. Since the objective function has zero as a lower bound, the iterative process will converge. □

3.3/ Experiments and Results

The proposed method is evaluated with six different public image datasets that include three scene datasets, two face datasets and one object dataset. These are as follows: 8 Sports Event Categories Dataset [60], Scene 15 Dataset [55], MIT 67 Indoor Scene [88], Extended YALE Face Dataset B [30], ORL Face Dataset [95] and COIL-20 Object dataset [72]. The following describes the details of every dataset.

3.3.1/ Datasets

8 Sports Event Categories Dataset: The 8 sports event categories dataset is provided by Li and Fei-Fei [60]. The dataset contains 8 sports event categories: rowing (250 images), badminton (200 images), polo (182 images), bocce (137 images), snowboarding (190 images), croquet (236 images), sailing (190 images), and rock climbing (194 images). These images are high-resolution. We use 130 images in every category. The total number of images is 1040 images. We randomly select 50% and 70% of data as the training dataset and use the remaining 50% and 30% of data as the test dataset.

Scene 15 Dataset: Scene 15 dataset [55] contains 4485 gray images of 15 different scenes including both indoor scenes and outdoor scenes. The dataset does not provide separated training and test sets. We use 130 images in every category. The total number of images is 1950 images. We randomly select 50% and 70% of data as the training dataset and use the remaining 50% and 30% of data as the test dataset.

MIT 67 Indoor Scene Dataset: MIT Indoor 67 [88] is a challenging indoor scene dataset, which contains 67 scene categories and 15,620 color images. We use 50 images in every category giving 3350 images. We randomly select 40 images per category (80% of each category) for training. The remaining 10 images are used for testing.

Extended YALE Face Dataset B: The cropped version that contains 38 individuals [30] has been used in our experiments. The images of the cropped version contain illumination variation and facial expression variation. The images are in gray scale, and we have rescaled them to 32×32 pixels. We use a subset of this database containing 50 images for each person. We randomly select 20% and 40% of data as the training set and use the remaining 80% and 60% of data as the test set.

ORL Face Dataset: The ORL dataset [95] depicts the face images of 40 subjects. Each subject has ten different face images. For some subjects, the images were taken at different times, varying the lighting, facial expressions (open / closed eyes, smiling / not smiling) and facial details (glasses / no glasses). All the images were taken against a dark homogeneous background with the subjects in an upright, frontal position (with tolerance for some side movement). We randomly select 10% and 20% of data as the training dataset and use the remaining 90% and 80% of data as the test dataset.

COIL-20 Object Dataset: This dataset (Columbia Object Image Library) [72] consists of 1440 images of 20 objects. Each object has 72 images. Between each image and its following, the object rotated by an angle of 5 degrees. We use a subset of the database containing 70 images for each object. We randomly select 10% and 20% of data as the training dataset and use the remaining 90% and 80% of data as the test dataset.

3.3.2/ Experimental Setup

Local Binary Patterns descriptor [81] is used on the outdoor scene datasets, such as 8 Sports Event Categories Dataset, Scene 15 Dataset and MIT 67 Indoor Scene Dataset. The local LBP descriptor is the uniform one having 59 features. Thus, for an image with b non-overlapping blocks, the length of the image descriptor is $59\,b$. The raw image data is used on Extended YALE Face Dataset B, ORL Face Dataset and COIL-20 Object Dataset. All results are obtained with ten random splits of the data into a training set and a testing set. For the training sets, two different percentages are considered that are stated in Section 3.3.1.

We select several of state-of-the-art graph-based algorithms which including with Semi-Supervised Discriminant Analysis (SDA) [7], Semi-Supervised Discriminant Embedding (SDE) [12], Laplacian eigenmaps (LE) [2], Gaussian fields and harmonic functions (GFHF) [134], Robust multi-class graph transduction (RMGT) [67], Locally Linear Embedding (LLE) [92], Manifold Regularized Deep Learning Architecture Algorithm (MRDL) [127], JELSR [46], Kernel Flexible Model Embedding (KFME) [22], Flexible Semi-Supervised Embedding algorithm (FSSE) [21]. In the graph-based embedding algorithms(SDA, SDE, LE, LLE, MRDL, JELSR and FSSE), we will use the same processing steps that the embedding matrix is computed, then to utilize Nearest Neighbor Classifier (NN) to compare the performance of image classification accuracy rate. We use the normalized Laplacian matrix $\mathbf{L} = \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{S}\mathbf{D}^{1/2}$. It is noteworthy that we obtain two layers in the MRDL algorithm as [127] descripted. Besides, for comparing the performance of two different graphs that used in FSSE [21] and in our algorithm 2.11, we report the results of them. According to [46], JELSR algorithm is used for unsupervised feature selection by computing the scores of all features. It also can be used for graph-based embedding. In our proposed algorithm, there are three different parameters that λ, μ, and γ. Each parameter scans the following set $\{10^{-2}, 10^{-1}, 1, 10^{1}, 10^{2}, 10^{3}\}$. We report the best average recognition rate of all methods from the best parameter configuration.

3.3.3/ Method Comparison

We compare the proposed method with the competing ones on various categories datasets including with the 8 Sports Event Categories Dataset, Scene 15 Dataset, MIT 67 Indoor Scene Dataset, Extended YALE Face Dataset B, ORL Face Dataset and COIL-20 Object Dataset. Tables 3.1, 3.2, 3.3, 3.4, 3.5 and 3.6 summarize the obtained average recognition rate obtained by our proposed algorithms and 12 competing methods: SDE [12], SDA [7], LE [2], GFHF [134], RMGT [67], LLE [92], MRDL [127], KFME [22], JELSR [46], JELSR (KNN Graph), FSSE [21], FGSE (KNN Graph).

We can observe that the FDEFS method gave the best results. Furthermore, in general, the FDEFS method was superior to all competing methods. This observation holds true for all six datasets and for different percentages of training data, except for KFME method in COIL-20 Object Dataset corresponding to the 10% training samples data expriment.

We have visualized the distribution of the images after the non-linear projection of the FDEFS using t-SNE [70]. Figure 3.5.a illustrates the distribution of the Extended Yale images (1900 images) in the original feature space. Figure 3.5.b illustrates the distribution of the same images in the non-linear space given the embedded data matrix $\mathbf{Z}$. In this case, the two plots depict the same 1900 images in a 2D space. For the FDEFS algorithm, 50 % of images were labeled. From this figure, it can be seen that by using our proposed projection method the images of the same class distribute closely and the samples of different classes are pushed far away as much as possible.

Figures (3.2), (3.3) and (3.4) depict the average performances of the competing methods, (SDA, SDE, LLE, MRDL, JELSR, FSSE and our proposed algorithm (FDEFS)), as a function of the number of non-linear features. These figures correspond to 8 Sports dataset, Scene 15 dataset and Extended YALE Face Dataset B, respectively. The KFME algorithm does not depend on the feature dimension since it is a label embedding method. In fact, the used competing methods are either linear or nonlinear. For the nonlinear methods, the maximum feature dimension is the number of samples N. For the linear methods, the maximum feature dimension is the number of original features d. In the above figures, we presented the accuracy curves till dimension 200 in the projection space. This is motivated by the fact that highest performance are reached with a low number of features.

3.3.4/ Analysis of results

From the obtained tables and figures, we can see that the proposed methods achieve superior recognition performance on different types of images. Based on these experimental results, a number of interesting observations can be made. These are as follows:

- According to Tables 3.1, 3.2, 3.3, 3.4, 3.5 and 3.6 we can observe that our proposed methods Flexible Discriminant graph-based Embedding with feature selection (FDEFS) outperformed all compared state-of-the-art methods on six public image datasets (including scene datasets, object and face datasets). The proposed methods have also given the best recognition rate for different percentages for the training part. The only outperformed by the KFME method on the COIL-20 dataset corresponding to the 10% training samples data experiment is adopted.

The method was outperformed in one configuration out of 198 configurations (depicted in Tables 3.1-3.6).

- The performance of the two competing methods, JELSR and FGSE when used with a sparse graph (kernelized sparse nonnegative graph Similarity matrix) is better than their performance obtained with a KNN Graph. Nevertheless, this better performance is still outperformed by our methods.
- As it can be seen from the recognition accuracy curves in Figures 3.2, 3.3 and 3.4, by increasing the number of features in the projection subspace the recognition accuracy of the proposed methods will not necessarily increase. Consider that the proposed methods will provide good performance without using a lot of features, this is one of advantages of the proposed FDEFS method.
- The performance of all methods on the MIT 67 dataset is relatively poor. There are many reasons for that observation. First, the training images used in each class are very few. Second, the used image descriptor may not be suited to this dataset. We recall that our objective here is to compare different projection methods in the same context and not to look for the best image descriptor for this dataset.

3.3.5/ Effect of the number of selected original features

In this section, we study the effect of the number of selected original features s used for recomputing the non-linear embedding matrix $\mathbf{Z}_s = \mathbf{X}_s^T\mathbf{W}_s + \mathbf{b}^T$, where $\mathbf{W}_s$ and $\mathbf{X}_s$ are the selected matrix using the computed scores $\|\mathbf{W}_{r*}\|_{r=1}^{d}$. Considering that the non-linear embedding with feature selection is one of the merits in our method, we aim to study the performance of the proposed method as a function of s. Figure (3.6) and (3.7) depict the obtained recognition rate as a function of s (number of the selected original features) for the 8 Sports dataset and Scene 15 dataset, respectively. The test samples per class were 50%.

These figures show that the number of the selected original features, s, can be any value above 70 % of the original size in order to guarantee the superiority of the proposed method. We emphasize that any value above will give similar performances. We can also note that the feature selection in our proposed FDEFS method is implicit in the sense that it does not need any explicit setting of s (i.e., ranking and selecting s original features) since the non-linear embedding and its linear regression are obtained with a row sparsity of the linear transform. This was also shown empirically since our method is still superior to other competing methods even in the case where all original features were used.

3.3.6/ Parameter Sensitivity Analysis

In this section, we study the effect of the regularization parameters on the classification performance of the proposed methods: FDEFS. More precisely, the proposed methods have three parameters: λ, μ, and γ. For studying the influence of these parameters (λ, μ and γ) on the performance of the proposed methods, we quantified the performance when these parameters varied. For a given dataset, we conducted two different experiments. In the first experiment, we set μ and γ to a sub-optimal value, we then study the performance as a function of the third parameter λ. The latter varies within a certain range. Figures

Table 3.1: Best average recognition rate (%) obtained on the 8 Sports Event using 10 random splits with two different percentages for the training part.

Dataset	**8 Sports Event**	
Method	P=50%	P=70%
SDE [12]	51.98	55.96
SDA [7]	63.46	66.06
LE [2]	51.48	54.07
GFHF [134]	62.25	64.29
RMGT [67]	62.58	64.20
LLE [92]	54.92	59.10
MRDL [127]	58.48	60.51
KFME [22]	62.58	65.03
JELSR [46]	55.92	57.60
JELSR (KNN)	52.46	55.48
FSSE [21]	64.72	67.18
FSSE (KNN)	59.96	63.24
FDEFS	**66.98**	**70.00**

(3.8) and (3.9) depict the variation of the recognition rate of the proposed method as a function of λ for the 8 Sports and Extended Yale B datasets, respectively.

In the second experiment, we fix λ and vary μ and γ by a grid search within a certain range. Figures (3.10) and (3.11) depict the variation of the recognition rate of the proposed method as a function of μ and γ for the 8 Sports and Extended Yale B datasets, respectively. From Figures (3.8) and (3.9), it turns out that the parameter λ is not significantly affecting the final recognition rate in most of the experimental results. It means that μ and γ are more important parameters. From Figures (3.10) and (3.11), one can deduce the optimal domain for the balance parameters μ and γ. On the two datasets, the best domain for γ is the interval $[0, 50]$. On the other hand, the performance is less sensitive to the choice of μ. We can observe that the proposed method has actually two balance parameters since the third one can be fixed, which is exactly the same number of balance parameters used by existing graph-based semi-supervised embedding techniques.

3.3.7/ Convergence Analysis

In this section, we empirically study the convergence property of the proposed optimization algorithm. Figures 3.12 and 3.13 show the objective function value and the classification accuracy versus the number of iterations of the proposed method on the 8 sports and 15 scene datasets. As it can be seen a real convergence is reached with less than 10 iterations. we provided a proof of convergence of the proposed algorithm on subsection 3.2.3.

Table 3.2: Best average recognition rate (%) obtained on the Scene 15 dataset using 10 random splits with two different percentages for the training part.

Dataset	**Scene 15 Dataset**	
Method	P=50%	P=70%
SDE [12]	46.10	48.07
SDA [7]	61.52	63.73
LE [2]	41.47	43.68
GFHF [134]	61.58	63.57
RMGT [67]	61.59	63.49
LLE [92]	44.26	47.42
MRDL [127]	52.16	54.64
KFME [22]	60.89	63.74
JELSR [46]	51.83	58.59
JELSR (KNN)	41.37	44.24
FSSE [21]	64.78	68.17
FSSE (KNN)	50.96	55.62
FDEFS	**66.53**	**69.91**

Table 3.3: Best average recognition rate (%) obtained on the MIT 67 Indoor dataset using 10 random splits with two different percentages for the training part.

Dataset	**67 Indoor**
Method	P=80%
SDE [12]	11.91
SDA [7]	11.87
LE [2]	9.33
GFHF [134]	16.46
RMGT [67]	16.46
LLE [92]	11.15
MRDL [127]	11.42
KFME [22]	16.33
JELSR [46]	12.42
JELSR (KNN)	9.45
FSSE [21]	16.46
FSSE (KNN)	10.06
FDEFS	**17.58**

Table 3.4: Best average recognition rate (%) obtained on the Extended YALE-B using 10 random splits with two different percentages for the training part.

Dataset	**Extended YALE-B**	
Method	P=20%	P=40%
SDE [12]	85.92	92.76
SDA [7]	89.96	96.54
LE [2]	80.01	86.39
GFHF [134]	89.26	92.74
RMGT [67]	89.82	93.04
LLE [92]	91.47	95.75
MRDL [127]	92.20	96.32
KFME [22]	90.39	92.85
JELSR [46]	85.31	90.13
JELSR (KNN)	75.03	84.54
FSSE [21]	93.71	98.31
FSSE(KNN)	93.36	98.18
FDEFS	**95.23**	**98.75**

Table 3.5: Best average recognition rate (%) obtained on the ORL dataset using 10 random splits with two different percentages for the training part.

Dataset	**ORL**	
Method	P=10%	P=20%
SDE [12]	55.27	64.18
SDA [7]	60.92	75.94
LE [2]	66.78	75.34
GFHF [134]	68.33	79.66
RMGT [67]	68.72	80.00
LLE [92]	68.50	80.91
MRDL [127]	69.33	81.28
KFME [22]	67.82	79.22
JELSR [46]	68.31	77.81
JELSR (KNN)	49.69	63.84
FSSE [21]	70.08	81.91
FSSE(KNN)	50.42	74.44
FDEFS	**73.67**	**84.31**

Table 3.6: Best average recognition rate (%) obtained on the COIL-20 datasets using 10 random splits with two different percentages for the training part.

Dataset	**COIL-20**	
Method	P=10%	P=20%
SDE [12]	89.10	95.33
SDA [7]	95.33	98.07
LE [2]	90.39	96.38
GFHF [134]	96.01	98.14
RMGT [67]	96.06	98.18
LLE [92]	91.81	94.71
MRDL [127]	93.02	96.10
KFME [22]	**96.98**	98.56
JELSR [46]	93.80	96.88
JELSR (KNN)	75.03	84.54
FSSE [21]	93.60	97.83
FSSE(KNN)	86.19	93.61
FDEFS	96.04	**98.89**

3.4/ Conclusion

This chapter presented a novel graph-based discriminant embedding method for both supervised and semi-supervised settings. It can be used with various categories of image classification tasks for which labeled data can have a small size. The proposed algorithm combines the advantages of Margin Discriminant Embedding and Sparse Regression in order to get a flexible non-linear graph embedding method. A byproduct of the framework is feature selection which is a straightforward application of the sparse regression. All of these advantages are put into a common framework that tackles the problem of image recognition. The learned data representations are more informative and discriminative in comparison with other competing methods. The resulting problem was efficiently solved by an iterative optimization strategy, in which its convergence property is demonstrated from both theoretical and experimental perspectives. The conducted experiments show the superiority of the proposed method with respect to the competing methods.

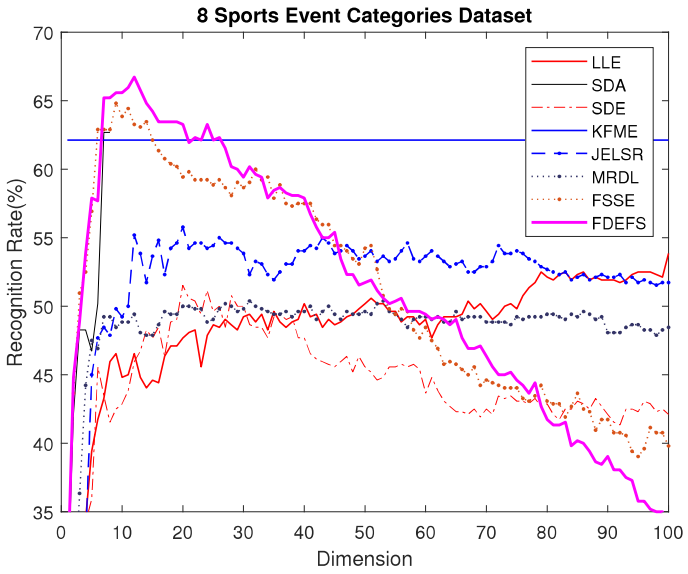


Figure 3.2: Recognition accuracy vs. feature dimension for the 8 Sports Event Categories Dataset. Test samples per class were 50%. The classifier used was the NN.

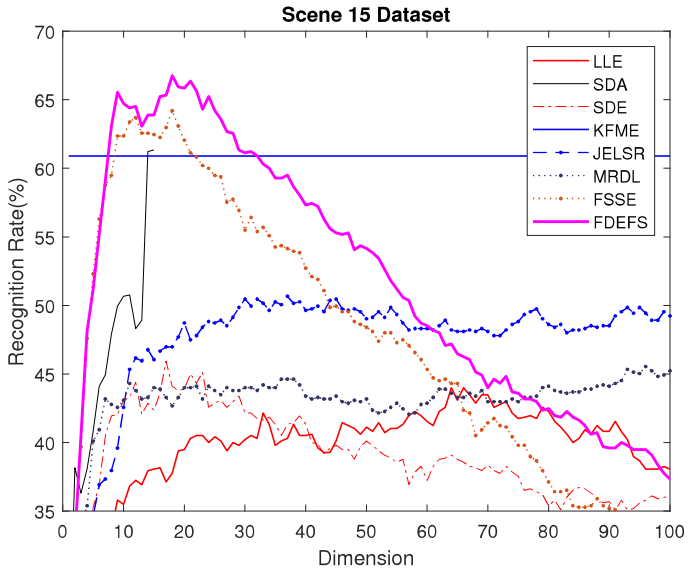


Figure 3.3: Recognition accuracy vs. feature dimension for the Scene 15 dataset. Test samples per class were 50%. The classifier used was the NN.

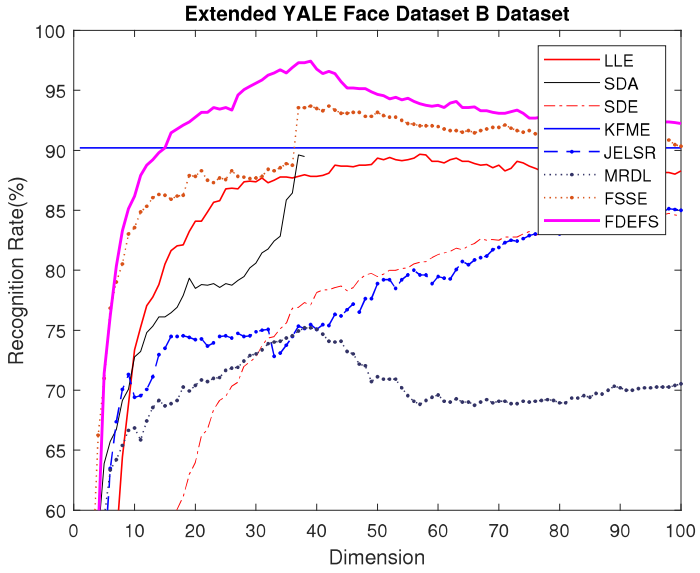
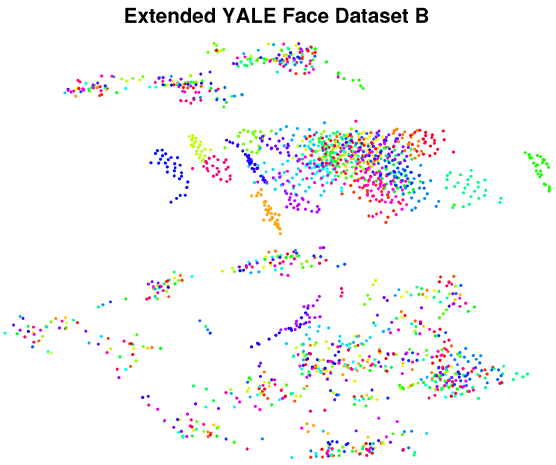
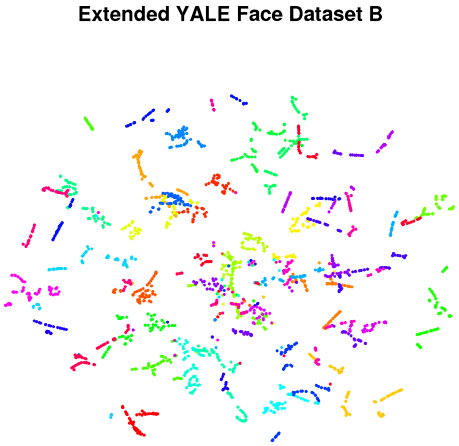


Figure 3.4: Recognition rate as a function of the number of selected original features in the Extended YALE Face Dataset B. Test samples per class were 20%. The classifier used was the NN.



(a) Original features.



(b) Projected features.

Figure 3.5: t-SNE visualization of (a) the original features and (b) non-linear projection obtained by using our FDEFS algorithm on Extended YALE Face Dataset B. In this case, the two plots depict the same 1900 images in a 2D space. For the FDEFS algorithm, 50 % of images were labeled.

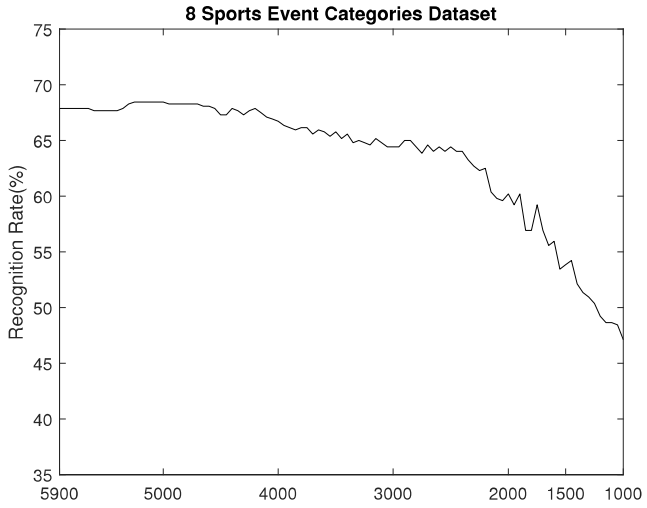


Figure 3.6: Recognition rate as a function of the number of selected original features for the 8 Sports Event Categories Dataset. Test samples per class were 50%.

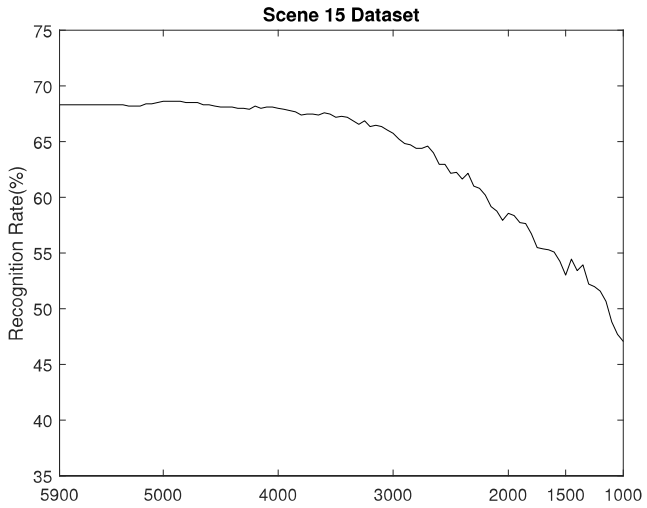


Figure 3.7: Recognition rate as a function of the number of selected original features for the Scene 15 dataset. Test samples per class were 50%. The classifier used was 1-NN.

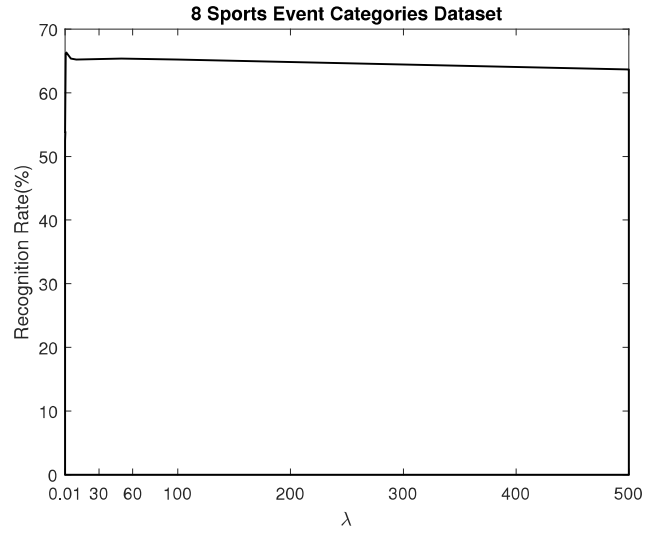


Figure 3.8: Recognition accuracy of the proposed method as a function of λ on the 8 Sports Event Categories Dataset. The test samples per class were 50%. The classifier used was 1-NN.

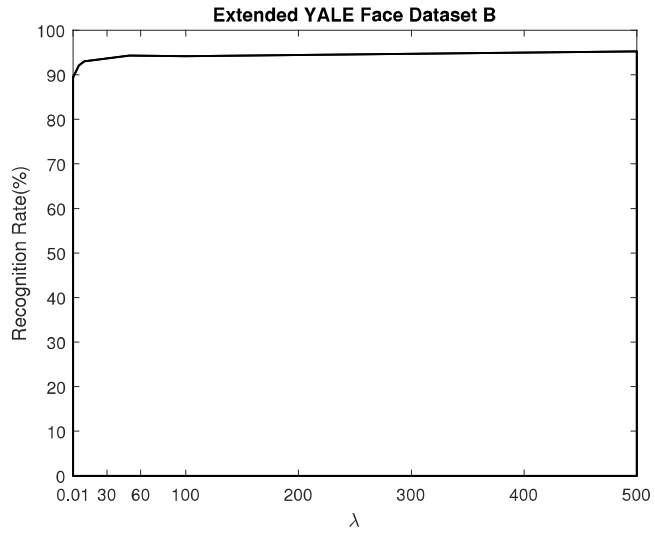


Figure 3.9: Recognition accuracy of the proposed method as a function of λ on Extended YALE Face Dataset B. The test samples per class were 20%. The classifier used was 1-NN.

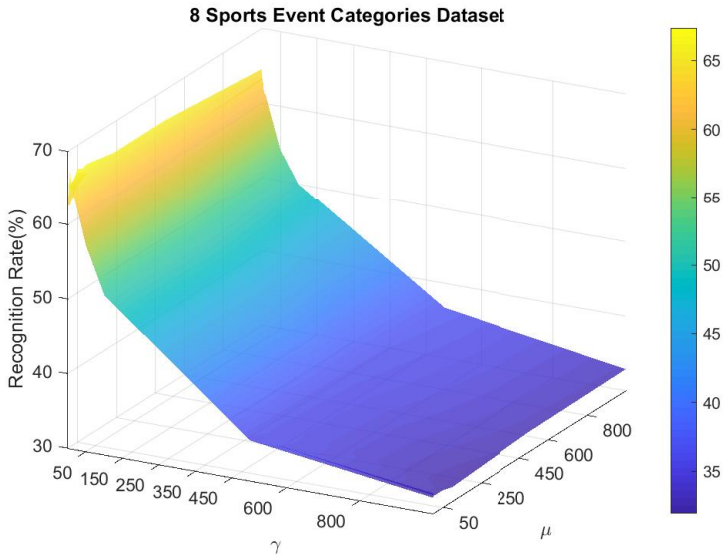


Figure 3.10: Recognition accuracy as a function of μ and γ for the proposed method on the 8 Sports Event Categories Dataset. Test samples per class were 50%. The classifier used was 1-NN.

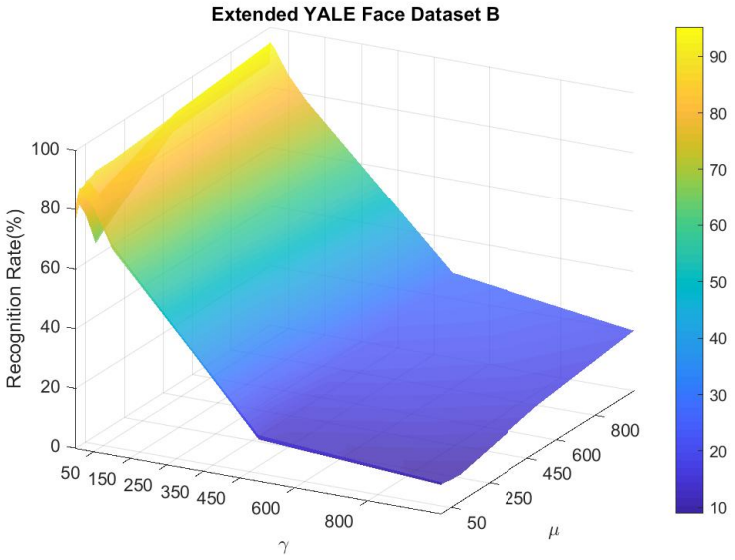


Figure 3.11: Recognition accuracy of the proposed method as a function of μ and γ on the Extended YALE Face Dataset B. Test samples per class were 20%. The classifier used was 1-NN.

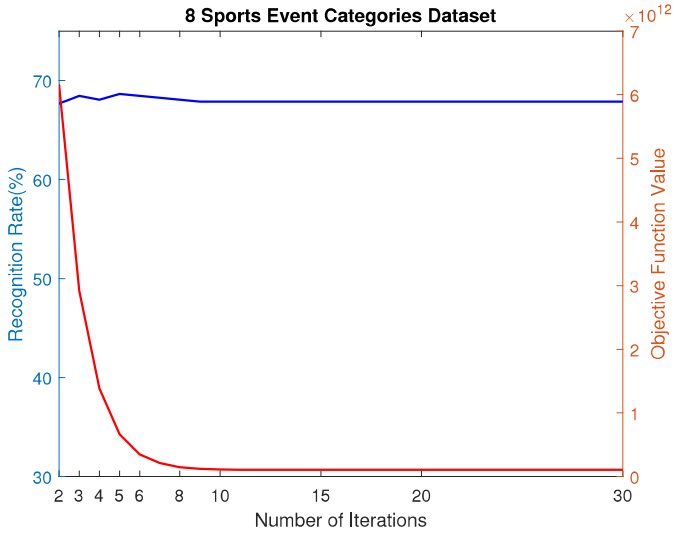


Figure 3.12: The convergence of the proposed FDEFS algorithm on the 8 Sports Event Categories Dataset. Test samples per class were 50%. The classifier used was the NN. The red curve depicts the objective function as a function of the iteration number. The blue curve depicts the recognition rate.

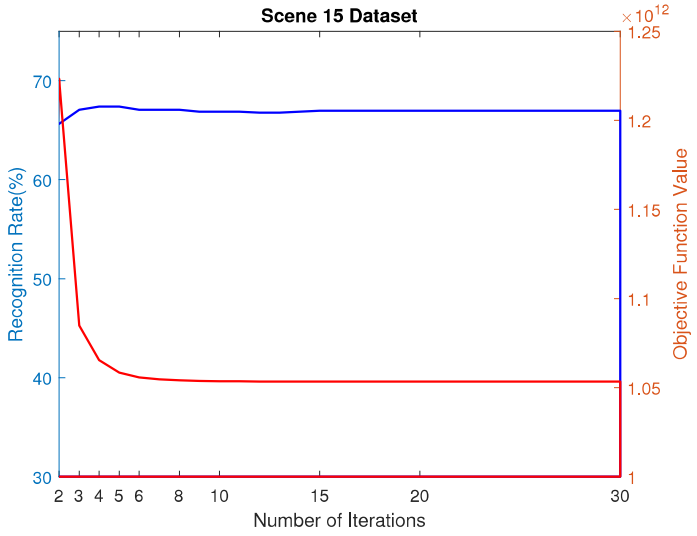


Figure 3.13: The convergence results of the proposed FDEFS algorithm on the Scene 15 Dataset. Test samples per class were 50%. The classifier used was the NN. The red curve depicts the objective function as a function of the iteration number. The blue curve depicts the recognition rate.

4

Joint Graph Based Embedding and Feature Weighting for Image Classification

In this chapter, we propose a joint graph-based embedding and feature weighting for getting a flexible and inductive nonlinear data representation on manifolds. Moreover, the chapter introduces a kernel variant of the model in order to get an inductive nonlinear embedding that is close to a real nonlinear subspace for a good approximation of the embedded data. The proposed criterion explicitly estimates the feature weights together with the projected data and the linear transformation such that data smoothness and large margins are achieved in the projection space. Experiments on image classification via shown on six public scene and face datasets, in a semi-supervised setting, show that our proposed methods can have a performance that is better than that of many state-of-the-art methods including linear and nonlinear methods.

4.1/ Overview of Proposed Approach

In this section, we will introduce our proposed novel algorithm and its kernel variant allowing the estimation of an inductive nonlinear embedding.

4.1.1/ Basic model: Graph based Semi-supervised Embedding

The work described in [21] proposed on inductive feature extration method. It includes the advantages of flexible manifold embedding and margin based discriminant embedding, meanwhile, to simultaneously estimate the nonlinear projection and the linear regression model by minimizing the following criterion:

Let us introduce the first objective function. This is given by the work described in [21]. This work proposed an inductive feature extraction method that retains the advantages of flexible manifold embedding and margin based discriminant embedding. The authors propose to simultaneously estimate the nonlinear projection and the linear regression model by minimizing the following criterion:

$$
\begin{aligned}
e(\mathbf{Z},\mathbf{W},\mathbf{b}) &= tr(\mathbf{Z}^T\mathbf{L}\,\mathbf{Z}) + \lambda\, tr(\mathbf{Z}^T\widetilde{\mathbf{M}}_l\mathbf{Z}) + \mu\left(\|\mathbf{W}\|^2 + \gamma\left\|\mathbf{X}^T\mathbf{W} + \mathbf{1}\mathbf{b}^T - \mathbf{Z}\right\|^2\right) \\
&= tr(\mathbf{Z}^T\mathbf{L}_1\mathbf{Z}) + \mu\left(\|\mathbf{W}\|^2 + \gamma\left\|\mathbf{X}^T\mathbf{W} + \mathbf{1}\,\mathbf{b}^T - \mathbf{Z}\right\|^2\right) \qquad (4.1)
\end{aligned}
$$

where $\mathbf{L}_1 = \mathbf{L} + \lambda\,\tilde{\mathbf{M}}_l$ and $\mathbf{1} \in \mathbb{R}^N$ is a column vector of $1s$. μ, γ, and λ are positive balance parameters. In the above objective function, the matrix $\mathbf{Z}$ represents the non-linear embedding, $\mathbf{W}$ and $\mathbf{b}$ are the linear transform that embed the data $\mathbf{X}$ such that $\mathbf{Z} \approx \mathbf{X}^T\mathbf{W} + \mathbf{1}\mathbf{b}^T$.

The objective function includes with four different terms previous to combine with $\mathbf{L}_1 = \mathbf{L} + \lambda\,\tilde{\mathbf{M}}_l$. The first term in the Eq.(4.1) is the graph smoothness criterion that imposes locality preserving on the unknown **Z**. The second term maximizes the margin associated with all labeled samples (as it is shown in [109]). The third one means regularization terms. The last term simultaneously forces the nonlinear **Z** projection to be as close as possible to a linear one and provides the linear regression model. Eq.(4.1) includes a non-linear approximation .Unfortunately, this method misses the importance of the original features. The next we introduce an unknown weight vector, denoted by τ, in which every element should encode the relevance of a feature.

4.1.2/ Proposed model: Joint Graph Based Embedding and Feature Weighting

The objective function in Eq.(4.1) includes a non-linear embedding and its linear approximation. However, the importance of the original features is not taken into account. In data mining, it is well known that feature selection or weighting can avoid, on one hand, over-fitting problems while improving classification performance, and on the other hand, can provide efficient and more cost-effective learning models [33].

The next we introduce an unknown weight vector, denoted by τ, in which every element should encode the relevance of a feature. Since we have d original features, the size of the unknown weight vector will be d. This vector is given by $\tau = (\tau_1, \tau_2, \ldots \tau_d)^T$ and subject to $\sum_{i=1}^{d} \tau_i = 1$. The diagonal matrix $\mathbf{T} = diag(\tau) \in \mathbb{R}^{d\times d}$ is given by

$$
\mathbf{T} = diag(\tau) = \begin{pmatrix} \tau_1 & 0 & \ldots & 0 \\ 0 & \tau_2 & \ldots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \tau_d \end{pmatrix}
$$

This method could take into consideration the relevance of every original feature in the data matrix **X** and improve their influence on the final estimation of both the non-linear embedding and the regression model. How to estimate the feature weights τ, the nonlinear projection **Z** and the linear regression model given by **W** and **b**, are the mainly considered work in the next processing. To this end, we minimize the following objective function:

$$e(\mathbf{Z},\ \mathbf{W},\ \mathbf{b},\ \tau) = tr(\mathbf{Z}^T\mathbf{L}_1\mathbf{Z}) + \mu\left(\|\mathbf{W}\|^2 + \gamma\left\|\mathbf{X}^T\mathbf{T}\mathbf{W} + \mathbf{1}\,\mathbf{b}^T - \mathbf{Z}\right\|^2\right)$$
$$s.t.\ \mathbf{Z}^T\tilde{\mathbf{D}}_l\mathbf{Z} = \mathbf{I},\ \sum_{i=1}^{d}\tau_i = 1,\ \text{and } \tau_i \geq 0 \tag{4.2}$$

The above objective function Eq.(4.2) not only provides a non-linear graph based embedding with discriminant information, but also takes into account the relevance of the original features in the criterion itself.

To the best of our knowledge, the problem defined in Eq.(4.2) does not have a closed-form solution. Thus, we will use an alternating optimization. We iterate two steps. In the first step, the vector of weights τ is fixed, thus allows to get a closed form solution for $\mathbf{Z}$, $\mathbf{W}$ and $\mathbf{b}$. In the second step, these variables are fixed and the vector of weights τ is estimated.

STEP 1. Fix τ, estimate $\mathbf{Z}$, $\mathbf{W}$, and $\mathbf{b}$.

Let $\mathbf{T}\mathbf{X} = \hat{\mathbf{X}}$. To obtain the optimal solution, we vanish the derivatives of the objective function Eq.(4.2) with respect to $\mathbf{W}$ and $\mathbf{b}$. This yields $\frac{\partial e}{\partial \mathbf{W}} = \mathbf{0}$ and $\frac{\partial e}{\partial \mathbf{b}} = \mathbf{0}$.

After some algebraic manipulations, we have:

$$\mathbf{b} = \frac{1}{N}(\mathbf{Z}^T\mathbf{1} - \mathbf{W}^T\hat{\mathbf{X}}\,\mathbf{1}) \tag{4.3}$$

$$\mathbf{W} = \gamma(\gamma\mathbf{X}_c\mathbf{X}_c^T + \mathbf{I})^{-1}\mathbf{X}_c\mathbf{Z} = \mathbf{A}\mathbf{Z} \tag{4.4}$$

where $\mathbf{A} = \gamma(\gamma\mathbf{X}_c\mathbf{X}_c^T + \mathbf{I})\mathbf{X}_c$. $\mathbf{X}_c$ is the centered data matrix, i.e., $\mathbf{X}_c = \hat{\mathbf{X}}\mathbf{H}_c$ with $\mathbf{H}_c$ being the centering matrix $\mathbf{H}_c = \mathbf{I} - \frac{1}{\mathbf{N}}\mathbf{1}\mathbf{1}^T$. We use the above expression for $\mathbf{W}$ and $\mathbf{b}$ in the regression function:

$$\begin{aligned}\hat{\mathbf{X}}^T\mathbf{W} + \mathbf{1}\mathbf{b}^T &= \hat{\mathbf{X}}^T\mathbf{A}\mathbf{Z} + \frac{1}{N}\mathbf{1}\mathbf{1}^T\mathbf{Z} - \frac{1}{N}\mathbf{1}\mathbf{1}^T\hat{\mathbf{X}}^T\mathbf{A}\mathbf{Z}\\ &= (\mathbf{I} - \frac{1}{N}\mathbf{1}\mathbf{1}^T)\hat{\mathbf{X}}^T\mathbf{A}\mathbf{Z} + \frac{1}{N}\mathbf{1}\mathbf{1}^T\mathbf{Z}\\ &= \mathbf{H}_c\hat{\mathbf{X}}^T\mathbf{A}\mathbf{Z} + \frac{1}{N}\mathbf{1}\mathbf{1}^T\mathbf{Z}\\ &= \mathbf{B}\mathbf{Z}\end{aligned} \tag{4.5}$$

where $\mathbf{B} = \mathbf{H}_c\hat{\mathbf{X}}^T\mathbf{A}\mathbf{Z} + \frac{1}{N}\mathbf{1}\mathbf{1}^T$.

$$\begin{aligned}e &= tr(\mathbf{Z}^T\mathbf{L}_1\mathbf{Z}) + \mu\cdot tr(\mathbf{Z}^T\mathbf{A}^T\mathbf{A}\mathbf{Z}) + \mu\gamma\cdot tr((\mathbf{B}\mathbf{Z} - \mathbf{Z})^T(\mathbf{B}\mathbf{Z} - \mathbf{Z}))\\ &= tr(\mathbf{Z}^T(\mathbf{L}_1 + \mu\mathbf{A}^T\mathbf{A} + \mu\,\gamma(\mathbf{B} - \mathbf{I})^T(\mathbf{B} - \mathbf{I}))\mathbf{Z})\\ &= tr(\mathbf{Z}^T(\mathbf{L}_1 + \mathbf{E})\mathbf{Z})\end{aligned} \tag{4.6}$$

where $\mathbf{E} = \mu\mathbf{A}^T\mathbf{A} + \mu\gamma(\mathbf{B} - \mathbf{I})^T(\mathbf{B} - \mathbf{I})$.

Thus, the non-linear embedding $\mathbf{Z}$ is estimated by minimizing the above criterion under the constraint used in the criterion happened in [21]:

$$\mathbf{Z} = \arg\min_{\mathbf{Z}} tr(\mathbf{Z}^T(\mathbf{L}_1 + \mathbf{E})\mathbf{Z})\ s.t.\ \mathbf{Z}^T\tilde{\mathbf{D}}_l\mathbf{Z} = \mathbf{I} \tag{4.7}$$

where $\tilde{\mathbf{D}}_l$ is the augmented diagonal matrix associated with [21]. Thus, $\mathbf{Z}$ can be solved by generalized eigenvalue decomposition. Once $\mathbf{Z}$ is estimated the corresponding regression $\mathbf{W}$, $\mathbf{b}$ are estimated by Eq.(4.4), and (4.3), respectively.

STEP 2. Fix $\mathbf{Z}$, $\mathbf{W}$, and $\mathbf{b}$, estimate $\boldsymbol{\tau}$. In this step, by fixing $\mathbf{Z}$, $\mathbf{W}$, and $\mathbf{b}$, the feature weight vector can be updated. We can note that only the third term in Eq.(4.2) depends on $\boldsymbol{\tau}$. Let $\mathbf{1}\mathbf{b}^T - \mathbf{Z} = \mathbf{M}$, and $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N] = [\mathbf{f}_1; \mathbf{f}_2; \cdots; \mathbf{f}_d]$ and $\mathbf{W} = (\mathbf{w}_1; \mathbf{w}_2; \ldots; \mathbf{w}_d)$. Thus, $\mathbf{f}_i$ is a row vector representing the ith row of the data matrix $\mathbf{X}$, and $\mathbf{w}_i$ is a row vector denoting the ith row of the matrix $\mathbf{W}$.

The third term in Eq.(4.2) can be written as:

$$
\begin{aligned}
h &= \left\| \mathbf{X}^T \mathrm{diag}(\tau_1, \tau_2 \cdots \tau_d)\mathbf{W} + \mathbf{M} \right\|_2^2 \\
&= \left\| \sum_{i=1}^{d} \mathbf{f}_i^T \tau_i \mathbf{w}_i + \mathbf{M} \right\|_2^2 \\
&= tr\left[\left(\sum_{i=1}^{d} \mathbf{f}_i^T \tau_i \mathbf{w}_i + \mathbf{M} \right)^T \left(\sum_{i=1}^{d} \mathbf{f}_i^T \tau_i \mathbf{w}_i + \mathbf{M} \right) \right] \\
&= tr[\sum_{i=1}^{d} \tau_i^2 \mathbf{A}_i^T \mathbf{A}_i + \sum_{i=1}^{d} \sum_{j \neq i}^{d} \mathbf{A}_i^T \mathbf{A}_j \tau_i \tau_j + 2 \sum_{i=1}^{d} \mathbf{M}^T \mathbf{A}_i^T \tau_i + \mathbf{M}^T \mathbf{M}] \\
&= \sum_{i=1}^{d} \tau_i^2 g_{ii} + \sum_{i=1}^{d} \sum_{j \neq i}^{d} g_{ij} \tau_i \tau_j + 2 \sum_{i=1}^{d} p_i \tau_i + tr(\mathbf{M}^T \mathbf{M})
\end{aligned} \tag{4.8}
$$

where $\mathbf{A}_i = \mathbf{f}_i^T \mathbf{w}_i$, $g_{ii} = tr(\mathbf{A}_i^T \mathbf{A}_i)$, $g_{ij} = tr(\mathbf{A}_i^T \mathbf{A}_j)$ and $p_i = tr(\mathbf{M}^T \mathbf{A}_i)$. Since the weights sum to one, we can construct the following Lagrangian:

$$
\begin{aligned}
\mathcal{L} &= h - \eta \left(\sum_{i=1}^{d} \tau_i - 1 \right) \\
&= \sum_{i=1}^{d} \tau_i^2 g_{ii} + \sum_{i=1}^{d} \sum_{j \neq i}^{d} g_{ij} \tau_i \tau_j + 2 \sum_{i=1}^{d} p_i \tau_i + tr(\mathbf{M}^T \mathbf{M}) - \eta \left(\sum_{i=1}^{d} \tau_i - 1 \right)
\end{aligned} \tag{4.9}
$$

By vanishing the derivative of the Lagrangian (given in Eq.(4.9)) with respect to τ_i, we have:

$$
\frac{\partial \mathcal{L}}{\partial \tau_i} = 2 g_{ii} \tau_i + 2 \sum_{j \neq i}^{d} g_{ij} \tau_j + 2 p_i - \eta = 0, \qquad i = 1, \ldots, d \tag{4.10}
$$

The d linear equations given in Eq. (4.10) can be written in the following matrix form:

$$
2\,\mathbf{G}\boldsymbol{\tau} + 2\mathbf{p} - \eta \mathbf{1} = 0 \tag{4.11}
$$

Thus, the solution for $\boldsymbol{\tau}$ is given by:

$$
\boldsymbol{\tau} = \mathbf{G}^{-1} \left(\frac{1}{2} \eta \mathbf{1} - \mathbf{p} \right) \tag{4.12}
$$

where $\mathbf{G} = (g_{ij}) \in \mathbb{R}^{d \times d}$, $\mathbf{p} = (p_i) \in \mathbb{R}^d$.

The constraint $\sum_{i=1}^{d} \tau_i = 1$ can be written in a matrix form as follows:

$$\mathbf{1}^T \boldsymbol{\tau} = 1 \tag{4.13}$$

By substituting Eq.(4.12) into Eq.(4.13), we can get the expression of η as:

$$\mathbf{1}^T \mathbf{G}^{-1}(\frac{1}{2}\eta \cdot \mathbf{1} - \mathbf{p}) = 1 \tag{4.14}$$

$$\Rightarrow \eta = \frac{2(1 + \mathbf{1}^T \mathbf{G}^{-1}\mathbf{p})}{\mathbf{1}^T \mathbf{G}^{-1}\mathbf{1}} \tag{4.15}$$

Finally, we can get the formula that updates τ by substituting Eq.(4.15) into Eq.(4.12) as follows:

$$\boldsymbol{\tau} = \mathbf{G}^{-1}\left(\frac{1 + \mathbf{1}^T \mathbf{G}^{-1}\mathbf{p}}{\mathbf{1}^T \mathbf{G}^{-1}\mathbf{1}}\mathbf{1} - \mathbf{p}\right) \tag{4.16}$$

Steps 1 and 2 are repeated until a stable solution is reached. Given an unseen sample $\mathbf{x}_{test}$, its embedding (a column vector) is given by $\mathbf{z}_{test} = \mathbf{W}^T diag(\boldsymbol{\tau})\mathbf{x}_{test} + \mathbf{b}$.

The main steps are given in Algorithm 2.

Algorithm 2 Joint graph-based Embedding with Feature Weighting (JEFW)

Input: Data samples: $\mathbf{X}$; Initial feature weights: $\boldsymbol{\tau} = (1, 1, \ldots, 1)^T \in \mathbb{R}^d$; Regularization parameters: μ, λ and γ.

Output: Non-linear embedding matrix $\mathbf{Z}$; linear transform $\mathbf{W}$ and $\mathbf{b}$; feature weights $\boldsymbol{\tau}$.

1: Compute the sparse graph matrix $\mathbf{S}$ using Eq. (2.11), note that $\mathbf{S}$ is symmetric. The normalized graph Laplacian matrix $\mathbf{L} = \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{S}\mathbf{D}^{1/2}$;
2: Fix feature weight vector $\boldsymbol{\tau}$, update $\mathbf{Z}$, $\mathbf{W}$, $\mathbf{b}$ using Eqs. (4.7), (4.4), and (4.3), respectively;
3: Fix $\mathbf{Z}$, $\mathbf{W}$, and $\mathbf{b}$, update $\boldsymbol{\tau}$ using Eq. (4.16);
4: Repeat steps 2 and 3 until the solution for $\mathbf{Z}$, $\mathbf{W}$, $\mathbf{b}$, and $\boldsymbol{\tau}$ becomes stable.

From the summarization of Algorithm 2, there are three main steps. The first step concerns the graph estimation using criterion (2.11). The computational complexity of this process is $O(N^2)$ where N is the number of samples. For step 2, the main computational cost is the eigenvalue decomposition of a $N \times N$ matrix (Eq.(4.7)). Thus, its computational complexity is $O(N^3)$. For step 3, the main computational cost is the inversion of a $N \times N$ matrix (Eq. (4.16)). Thus, its computational complexity is $O(N^3)$. Let T denote the number of iteration of Algorithm 2, it follows that the total computational complexity of the proposed JEFW method is $O(N^2) + TO(N^3)$.

4.1.3/ Convergence analysis

We can prove that the proposed Algorithm 2 makes the value of the objective function in Eq.(4.2) monotonically decrease. At iteration t, the solution is denoted by $\mathbf{Z}_t, \mathbf{W}_t, \mathbf{b}_t$, and $\boldsymbol{\tau}_t$. At iteration $t+1$, step 2 in Algorithm 2 estimates $\mathbf{Z}_{t+1}, \mathbf{W}_{t+1}, \mathbf{b}_{t+1}$, by minimizing the functional $e(\mathbf{Z}, \mathbf{W}, \mathbf{b}, \boldsymbol{\tau}_t)$. Thus, we have:

$$e(\mathbf{Z}_{t+1}, \mathbf{W}_{t+1}, \mathbf{b}_{t+1}, \boldsymbol{\tau}_t) \leq e(\mathbf{Z}_t, \mathbf{W}_t, \mathbf{b}_t, \boldsymbol{\tau}_t)$$

Step 3 estimates τ_{t+1} by minimizing $e(\mathbf{Z}_{t+1},\ \mathbf{W}_{t+1},\ \mathbf{b}_{t+1},\ \tau_t)$. Thus, we have:

$$e(\mathbf{Z}_{t+1},\ \mathbf{W}_{t+1},\ \mathbf{b}_{t+1},\ \tau_{t+1}) \leq e(\mathbf{Z}_{t+1},\ \mathbf{W}_{t+1},\ \mathbf{b}_{t+1},\ \tau_t)$$

From the above two inequalities, we will have:

$$e(\mathbf{Z}_{t+1},\ \mathbf{W}_{t+1},\ \mathbf{b}_{t+1},\ \tau_{t+1}) \leq e(\mathbf{Z}_t,\ \mathbf{W}_t,\ \mathbf{b}_t,\ \tau_t)$$

The above shows that the objective function in Eq.(4.2) monotonically decreases. We found empirically that this algorithm usually needs about five iterations in order to reach convergence.

4.1.4/ Kernel Variant

Consider that the nonlinearity of data cannot be close to a linear subspace [130], [103]. In such cases, the flexibility introduced by the linear regression term may not lead to good represenation in the nonlinear subspace. To tackle this limitation, we propose a kernel varaint of the above model that aims at a flexibility in which the regression itself is nonlinear. Thus, the role of the kernel trick is to seek an inductive nonlinear embedding that is close to a real nonlinear subspace. We were inspired by the kernel method [3] associated with the linear LapRLS algorithm. In this method, the optimized criterion is given by:

$$\min \sum_{i=1}^{l} F(\mathbf{d}(\mathbf{x}_i), \mathbf{y}_i) + \lambda_A \|\mathbf{d}\|_\kappa^2 + \lambda_I \|\mathbf{d}\|_\varsigma \tag{4.17}$$

where $\|\mathbf{d}\|_\kappa^2$ and $\|\mathbf{d}\|_\varsigma$ are the RKHS-norm and the graph-based smoothness of the model $\mathbf{d}$, respectively. $F(.,.)$ is a loss function. The model $\mathbf{d}$ could expand over all the N data sample in the form of:

$$\mathbf{d}(\mathbf{x}) = \sum_{j=1}^{N} \mathbf{v}_j^T \tau_j K(\mathbf{x}, \mathbf{x}_j) \tag{4.18}$$

where $\tau \in \mathbb{R}^N$ has a similar role to that included in Eq.(4.16). However, in the kernel variant the features correspond to the kernel function between a data sample and a set of N reference samples. $\mathbf{d}$ is a row vector having N elements, $\mathbf{K} \in \mathbb{R}^{N \times N}$ is a kernel Gram matrix, and the matrix $\mathbf{V} \in \mathbb{R}^{N \times C}$ is the unknown matrix of multipliers ($\mathbf{V} \in [\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_N]^T$). The entries of the Gram matrix are given by $\mathbf{K}(\mathbf{x}_i, \mathbf{x}_j)$ that represents a dot product in feature space. This kernel function can be Gaussian or polynomial. In our experiment, we choose the Gaussian kernel to test the performance of our algorithm.

The mapping of all data samples (based on the one given in Eq.(4.18)) can be written in a matrix form as:

$$\mathbf{Z} = \mathbf{K} \cdot diag(\tau) \cdot \mathbf{V} \tag{4.19}$$

where τ= [τ_1,τ_2,...,τ_N] in kernel method. Our proposed kernel method defines the row vector $\mathbf{z}_i$ as the nonlinear embedding of the sample $\mathbf{x}_i$. The dimension of $\mathbf{z}_i$ is N. We should make sure that the non-linear embedding of $\mathbf{x}_i$ is close to its kernel embedding. In

other words, Eq. (4.19) should be as satisfied as possible. The unknown of the problem are given by the nonlinear embedding $\mathbf{Z}$ and the matrix of multipliers $\mathbf{V}$.

We introduce the kernelized version by minimizing the following criterion:

$$e(\mathbf{Z}, \mathbf{V}) = trace(\mathbf{Z}^T \mathbf{L}_1 \mathbf{Z}) + \mu \, [trace(\mathbf{V}^T \hat{\mathbf{K}} \mathbf{V}) + \gamma \, trace((\hat{\mathbf{K}}\mathbf{V} - \mathbf{Z})^T (\hat{\mathbf{K}}\mathbf{V} - \mathbf{Z}))] \tag{4.20}$$

where $\hat{\mathbf{K}} = \mathbf{K} \cdot diag(\tau)$.

For fixed weights τ, We vanish the derivatives of the objective function e with respect to $\mathbf{V}$:

$$2\,\hat{\mathbf{K}}\,\mathbf{V} + 2\,\gamma\,\hat{\mathbf{K}}(\hat{\mathbf{K}}\,\mathbf{V} - \mathbf{Z}) = \mathbf{0} \tag{4.21}$$

This can written in the following form:

$$\mathbf{V} = \gamma \, (\mathbf{I} + \gamma \, \hat{\mathbf{K}})^{-1} \mathbf{Z} = \mathbf{A}_1 \mathbf{Z} \tag{4.22}$$

where $\mathbf{A}_1 = \gamma \, (\mathbf{I} + \gamma \, \hat{\mathbf{K}})^{-1}$. By plugging the above expression in Eq.(4.20), this becomes:

$$\begin{aligned} e(\mathbf{Z}) &= tr(\mathbf{Z}^T \mathbf{L}_1 \mathbf{Z}) + \mu \; tr(\mathbf{Z}^T \mathbf{A}_1^T \mathbf{A}_1 \mathbf{Z}) + \mu\gamma \cdot tr(\mathbf{Z}^T \mathbf{B}_1^T \mathbf{B}_1 \mathbf{Z}) \\ &= tr(\mathbf{Z}^T (\mathbf{L}_1 + \mu \, \mathbf{A}_1^T \hat{\mathbf{K}} \mathbf{A}_1 + \mu \, \gamma \, \mathbf{B}_1^T \mathbf{B}_1) \mathbf{Z}) \end{aligned} \tag{4.23}$$

where $\mathbf{B}_1 = \hat{\mathbf{K}}\mathbf{A}_1 + \mathbf{I}$, thus, the non-linear embedding $\mathbf{Z}$ with kernelized is estimated by minimizing the above criterion under the constraint used in the criterion Eq.(4.7):

$$\begin{aligned} \mathbf{Z} = \arg\min_{\mathbf{Z}} \; & tr(\mathbf{Z}^T (\mathbf{L}_1 + \mu \mathbf{A}_1^T \hat{\mathbf{K}} \mathbf{A}_1 + \mu\gamma \mathbf{B}_1^T \mathbf{B}_1) \mathbf{Z}) \\ s.t. \;\; & \mathbf{Z}^T \tilde{\mathbf{D}}_l \mathbf{Z} = \mathbf{I} \end{aligned} \tag{4.24}$$

where $\mathbf{Z}$ can be solved by generalized eigenvalue decomposition.

Similarly to the iterative process introduced in Section 4.1.2, the solution for $\mathbf{Z}$, $\mathbf{V}$, τ proceeds by repeating the following two steps. In the first step, fix τ and estimate $\mathbf{Z}$, $\mathbf{V}$ using Eqs. (4.24) and (4.22), respectively. In the second step, update τ using Eq. (4.16). In this expression, the matrices $\mathbf{X}^T$, $\mathbf{W}$, and $\mathbf{M}$ are replaced by $\mathbf{K}$, $\mathbf{V}$, and $\mathbf{Z}$, respectively.

Given an unseen sample $\mathbf{x}_{test}$ its embedding (a column vector) is given by

$$\mathbf{z}_{test} = \mathbf{V}^T [\tau_1 \, K(\mathbf{x}_{test}, \mathbf{x}_1), ..., \tau_N \, K(\mathbf{x}_{test}, \mathbf{x}_N)]^T \tag{4.25}$$

4.2/ Experiments and Results

Our method will be evaluated on six different public datasets including four scene datasets and one face dataset. These are as follows: 8 Sports Event Categories Dataset [60], Scene 15 Dataset [55], MIT 67 Indoor Scene Dataset [88], Caltech 101 Object Dataset [23], Extended YALE Face Dataset B [30] and ORL Face Dataset [95], which have already been introducted in the previous chapters.

4.2.1/ Experimental setup

For the Extended YALE Face Dataset B and ORL dataset, we use image raw brightnesses. For the datasets depict outdoor scenes, we use the block-based Local Binary Patterns [81] as the image descriptor that is a well-known image descriptor that is fast and invariant to monotonic illumination changes. The local LBP descriptor is the uniform one having 59 features. Thus, for an image with b non-overlapping blocks, the length of the image descriptor is $59\,b$.

We compare our proposed methods with some well-known art-of-state algorithms including Supervised-ISOMAP (S-ISOMAP) [29], Semi-Supervised Discriminant Embedding (SDE) [12], Semi-Supervised Discriminant Analysis (SDA) [7], Kernel Flexible Model Embedding (KFME) [22], Locally Linear Embedding (LLE) [92], Laplacian Eigenmaps (LE) [2], Gaussian Fields and Harmonic Functions (GFHF) [134], Robust multi-class graph transduction (RMGT) [67], Joint Embedding Learning and Sparse Regression (JELSR) [46], Manifold Regularized Deep Learning Architecture Algorithm (MRDL) [127] and Flexible Graph based Semi-supervised Embedding(FGSE) [21]. For the MRDL method, we used two layers.

For the FGSE and JELSR method, we report the performance with two graphs that one is KNN graph as it is used in the original FGSE [21] and JELSR [46], the other one is the kernel sparse graph. We report the best average recognition rate of all methods from the best parameter configuration. The proposed method has three balance parameters: λ, μ, and γ. Each parameter scans the following set $\{10^{-2}, 10^{-1}, 1, 10^{1}, 10^{2}, 10^{3}\}$. All results are obtained with ten random splits of the data into a training set and a testing set. For the training sets, two different percentages are considered that are stated in last section. For a fair comparison, we use the same graph that is obtained by the kernel sparse algorithm described in [127]. We use the normalized Laplacian matrix $\mathbf{L} = \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{S}\mathbf{D}^{1/2}$.

4.2.2/ Method comparison

We compare the proposed method with the competing ones on the six image datasets in this section. Tables 4.1, 4.2, 4.3, 4.5, 4.4 and 4.6 summarize the average performances obtained by the different methods on the six image datasets. These results correspond to an average over 10 random splits on the training data and testing data. For each dataset, two train percentages were considered except for Caltech 101 Object and MIT 67 Indoor Scene dataset were 80% for training data. We can observe that the JEFW method and its kernel variant gave the best results. Furthermore, in general, the JEFW method was superior to all competing methods. This observation holds true for all six datasets and for different percentages of training data.

Figures 4.1, 4.2, 4.3 and 4.4 depict the average performance of the competing methods, (KFME, LLE, GFHF, JELSR, FGSE, FGSE with sparse graph and our proposed algorithms(JEFW and KJEFW)), as a function of the number of non-linear features. These figures correspond to 8 Sports dataset, Scene 15 dataset, Extended YALE Face Dataset B, Caltech 101 Object and MIT 67 Indoor Scene dataset respectively. The KFME algorithm does not depend on the feature dimension since it is a label embedding method. For the nonlinear methods, the maximum feature dimension is the number of samples N. For the linear methods, the maximum feature dimension is the number of original features d. In the above figures, we presented the accuracy curves till dimension 200 in the pro-

jection space. This is motivated by the fact that highest performance are reached with a low number of features.

4.2.3/ Analysis of results

From the obtained tables and figures, we can see that the proposed algorithm achieves superior recognition performance on different applications. Based on these experimental results, a number of interesting observations can be made. These are as follows:

- Our proposed non-linear embedding method provided better performances than those of state-of-art algorithms on the different datasets. This holds true whether the competing method was a graph-based embedding method or graph-based label propagation method.
- By comparing the performances of JELSR and JELSR (KNN), FGSE and FGSE (KNN), we can conclude that the use of the kernelized sparse nonnegative graph matrix $\mathbf{S}$ is able to produce an embedding that can be much better than the embedding obtained by the KNN graph.
- As it can be seen from the recognition accuracy curves, by increasing the number of features in the projection subspace (obtained by the proposed method), the recognition accuracy does not necessarily increase. Thus, in practice, the proposed method will provide good performance without using a lot of features. According to the figures depicting the performances on 8 Sports Event, Scene 15 datasets, Caltech 101 Object and MIT 67 Indoor Scene dataset, the proposed method already provided an optimal performance with just quit low features.
- In general, the kernel variant has given better performance than the non-kernel method.
- The performance of the proposed methods JEFW and KJEFW is better than that of the FGSE method. This can be explained by the fact that the proposed methods (JEFW and KJEFW) explicitly quantify the relevance of the original features via the use of a weight vector that is estimated by the method. Since the weighted version of the features in the data are used in deriving the nonlinear embedding and its linear regression, these ones will lead to better features in the projection space. This explains that a better discrimination can be achieved with the explicit weighting of the original features.
- The performance of all methods on the MIT 67 dataset is relatively poor. There are many reasons for that observation. First, the training images used in each class are very few. Second, the used image descriptor may not be suited to this dataset. We recall that our objective here is to compare different projection methods in the same context and not to look for the best image descriptor for this dataset.

4.2.4/ Stability with respect to balance parameters

In this section, we study the effects of the balance parameters on the classification performance. More precisely, the proposed method has three balance parameters which are

Table 4.1: Best average recognition rate (%) obtained on the 8 Sports Event using 10 random splits with two different percentages for the training part.

Dataset	**8 Sports Event**	
Method	P=50%	P=70%
GFHF [134]	62.25	64.29
RMGT [67]	62.58	64.20
KFME [22]	62.58	65.03
SDE [12]	51.98	55.96
SDA [7]	63.46	66.06
LLE [92]	54.92	59.10
LE [2]	51.48	54.07
S-ISOMAP [29]	51.88	54.68
MRDL [127]	58.48	60.51
JELSR [46] (KNN Graph)	52.46	55.48
JELSR (Sparse Graph)	55.92	57.60
FGSE [21] (KNN Graph)	59.96	63.24
FGSE (Sparse Graph)	64.72	67.18
JEFW	65.81	68.56
KJEFW	**67.48**	**69.71**

λ, μ, and γ. For studying the influence of the balance parameters (λ, μ and γ) on the proposed algorithm, we quantified its performance when these parameters varied. For a given dataset, we conducted two types of experiments. In the first type of experiments, we fixed μ and γ to a rough sub-optimal value, we then study the performance as a function of the third parameter λ. The latter varies within a certain range. Figures 4.5 and 4.6 depict the variation of the recognition rate of the proposed method as a function of λ for the 8 Sports and Scene 15 dataset, respectively.

In the second type of experiments, λ is fixed. We report the recognition rate by varying μ and γ using a given grid search. Figures 4.7 and 4.8 depict the variation of the recognition rate of the proposed method as a function of μ and γ for the 8 Sports and Scene 15 dataset, respectively. From Figures 4.7 and 4.8, it turns out that the parameter λ is not significantly affecting the final recognition rate in most of the experimental results. It means that μ and γ are more important parameters. From Figures 4.7 and 4.8, one can deduce the optimal domain for the balance parameters μ and γ. This is not surprising since the residual error term, which is responsible for feature section and regression model, is controlled by the product $\mu\gamma$. Nevertheless for the used two image datasets, and for the two proposed methods, the optimal performance was obtained when μ and γ were around 10.

Table 4.2: Best average recognition rate (%) obtained on the Scene 15 dataset using 10 random splits with two different percentages for the training part.

Dataset	**Scene 15 Dataset**	
Method	P=50%	P=70%
GFHF [134]	61.58	63.57
RMGT [67]	61.59	63.49
KFME [22]	60.89	63.74
SDE [12]	46.10	48.07
SDA [7]	61.52	63.73
LLE [92]	44.26	47.42
LE [2]	41.47	43.68
S-ISOMAP [29]	42.74	45.28
MRDL [127]	52.16	54.64
JELSR [46] (KNN Graph)	41.37	44.24
JELSR (Sparse Graph)	51.83	58.59
FGSE [21] (KNN Graph)	50.96	55.62
FGSE (Sparse Graph)	64.78	68.17
JEFW	65.74	69.93
KJEFW	**68.24**	**70.91**

Table 4.3: Best average recognition rate (%) obtained on the Caltech 101 using 10 random splits.

Dataset	**Caltech 101**
Method	P=80%
GFHF [134]	31.22
RMGT [67]	31.19
KFME [22]	31.04
SDE [12]	26.70
SDA [7]	30.56
LLE [92]	24.52
LE [2]	22.28
S-ISOMAP [29]	22.51
MRDL [127]	26.60
JELSR [46] (KNN Graph)	22.95
JELSR (Sparse Graph)	25.61
FGSE [21] (KNN Graph)	25.26
FGSE (Sparse Graph)	34.27
JEFW	35.54
KJEFW	**37.36**

Table 4.4: Best average recognition rate (%) obtained on the ORL dataset using 10 random splits.

Dataset	**ORL**	
Method	P=50%	P=70%
GFHF [134]	90.60	93.92
RMGT [67]	90.55	94.17
KFME [22]	90.65	94.12
SDE [12]	91.15	93.67
SDA [7]	90.60	94.75
LLE [92]	93.35	97.08
LE [2]	84.00	87.25
S-ISOMAP [29]	88.35	93.92
MRDL [127]	93.50	97.42
JELSR [46] (KNN Graph)	83.54	89.86
JELSR (Sparse Graph)	91.40	95.42
FGSE [21] (KNN Graph)	94.75	98.00
FGSE (Sparse Graph)	96.30	98.67
JEFW	96.10	98.58
KJEFW	**96.90**	**99.42**

Table 4.5: Best average recognition rate (%) obtained on the MIT 67 Indoor using 10 random splits.

Dataset	**MIT 67**
Method	P=80%
GFHF [134]	16.46
RMGT [67]	16.46
KFME [22]	16.33
SDE [12]	11.91
SDA [7]	11.87
LLE [92]	11.15
LE [2]	9.33
S-ISOMAP [29]	11.21
MRDL [127]	11.42
JELSR [46] (KNN Graph)	9.45
JELSR (Sparse Graph)	12.42
FGSE [21] (KNN Graph)	10.06
FGSE (Sparse Graph)	16.46
JEFW	18.85
KJEFW	**19.67**

Table 4.6: Best average recognition rate (%) obtained on the Extended Yale B dataset using 10 random splits.

Dataset	**Extended Yale B**	
Method	P=20%	P=40%
GFHF [134]	89.26	92.74
RMGT [67]	89.82	93.04
KFME [22]	90.39	92.85
SDE [12]	85.92	92.76
SDA [7]	89.96	96.54
LLE [92]	91.47	95.75
LE [2]	80.01	86.39
S-ISOMAP [29]	85.12	89.93
MRDL [127]	92.20	96.32
JELSR [46] (KNN Graph)	75.03	84.54
JELSR (Sparse Graph)	85.31	90.13
FGSE [21] (KNN Graph)	93.36	98.18
FGSE (Sparse Graph)	93.71	98.31
JEFW	94.26	98.46
KJEFW	**94.85**	**99.49**

4.3/ Conclusion

This chapter presented two joint graph-based embedding and feature weighting methods for recognition and classification tasks. The two methods can be applied either in a semi-supervised setting or in a supervised setting. More precisely, we proposed a flexible graph-based semi-supervised embedding as well as its kernel variant. The proposed schemes retained the merits of Flexible Manifold Embedding and the discrimant graph-based nonlinear embedding. The proposed methods simultaneously estimate a discriminant nonlinear embedding as well as its inductive regression model while, at the same time, estimating the weights of the original features. The first proposed criterion aims at getting a smooth and discriminant nonlinear subspace that is very close to a linear subspace. The second proposed criterion (kernel variant) aims at getting a discriminant nonlinear subspace that is very close to a kernel based mapping. The proposed methods were evaluated on six public image databases. We provided a comparison with several competing linear and nonlinear methods performing label propagation or graph-based embedding. Our proposed methods outperformed the competing methods. This indicates that the embedding provided by the proposed methods was more discriminative than that provided by the competing graph-based embedding techniques.

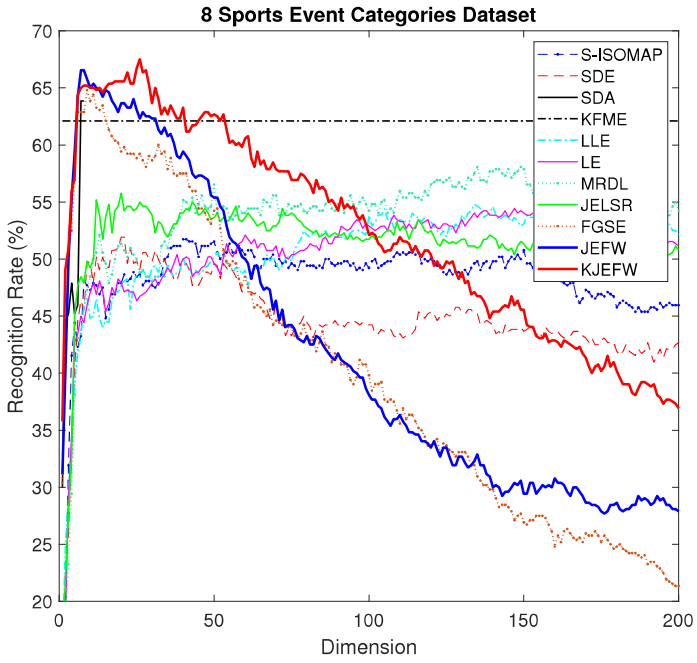


Figure 4.1: Recognition accuracy as a function of the feature dimension in the embedded space for the 8 Sports Event Categories Dataset. The training samples were set to 50 % of the whole dataset. The classifier used was the nearest neighbor classifier.

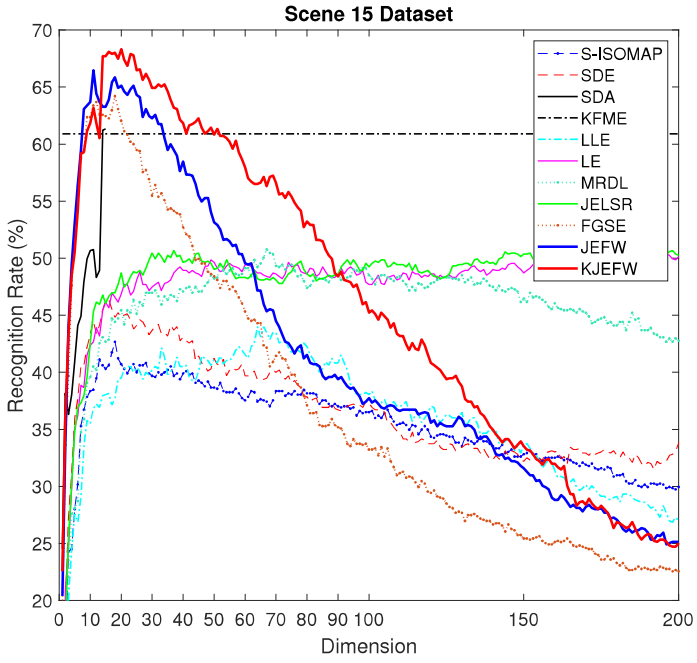


Figure 4.2: Recognition accuracy as a function of the feature dimension for the Scene 15 Dataset. The training samples were set to 50 % of the whole dataset. The classifier used was the nearest neighbor classifier.

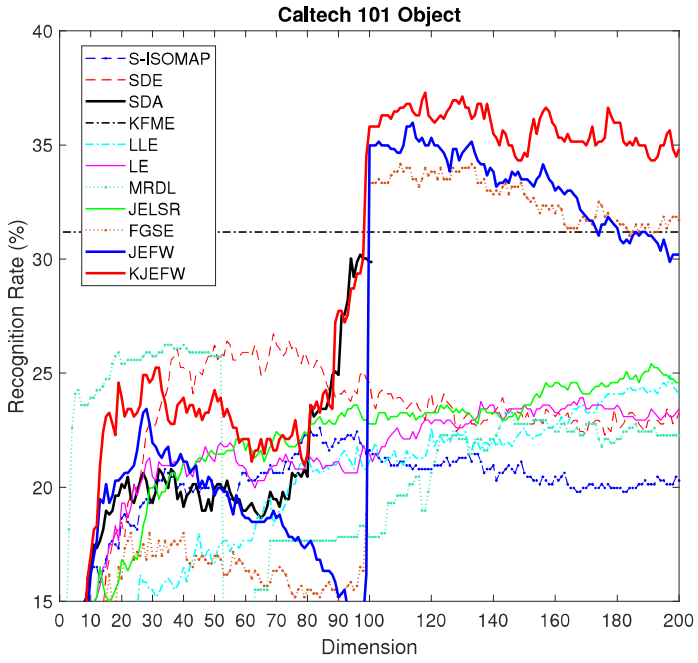


Figure 4.3: Recognition accuracy as a function of the feature dimension in the embedded space for Caltech 101 Object. The training samples were set to 80 % of the whole dataset. The classifier used was the nearest neighbor classifier.

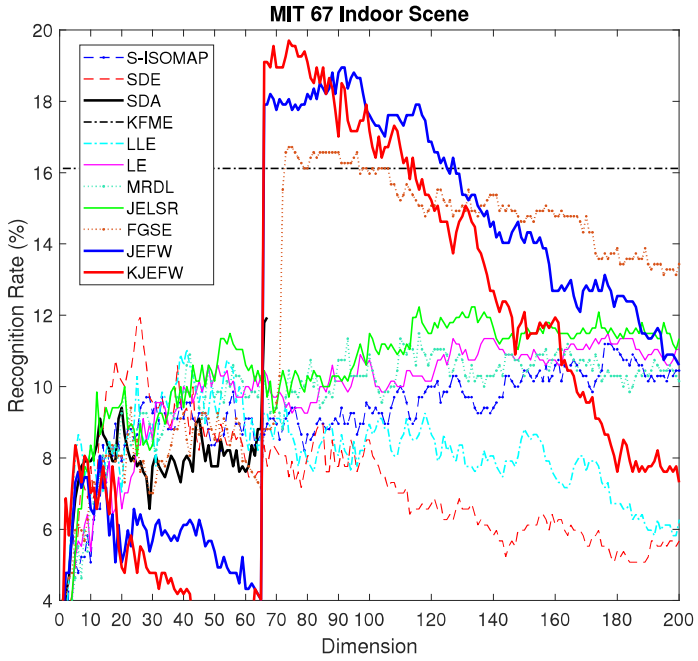
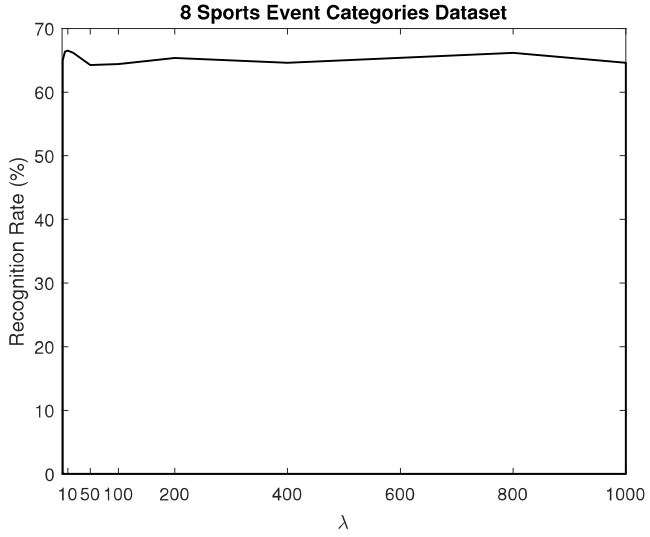
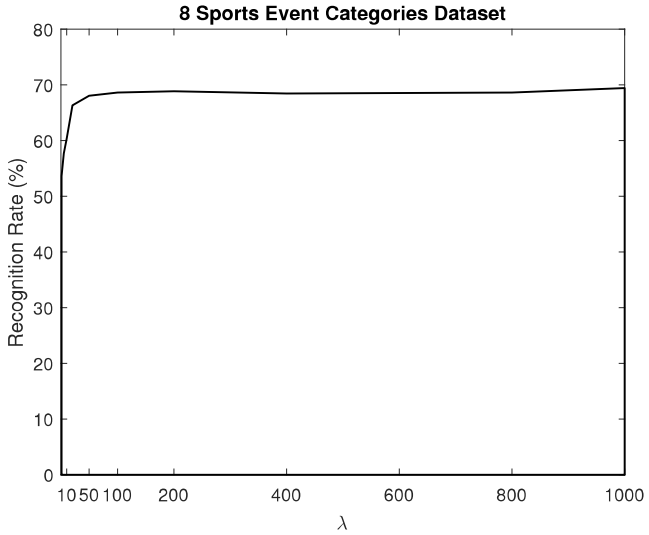


Figure 4.4: Recognition accuracy as a function of feature dimension for MIT 67 Indoor Scene. The training samples were set to 80 % of the whole dataset. The classifier used was the nearest neighbor classifier.

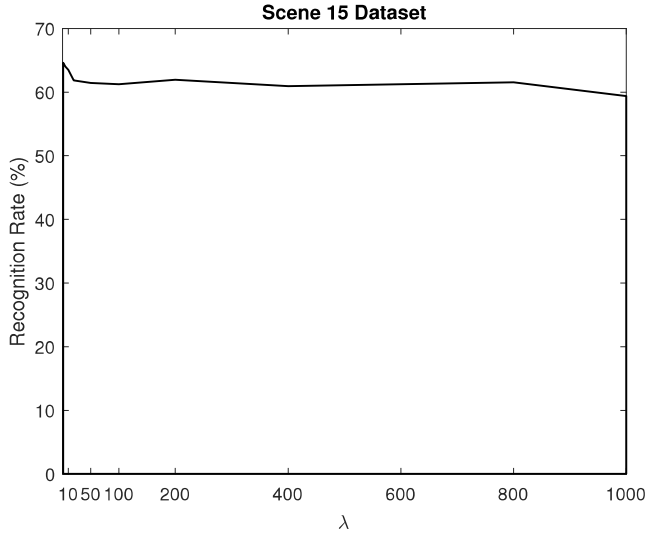


(a)

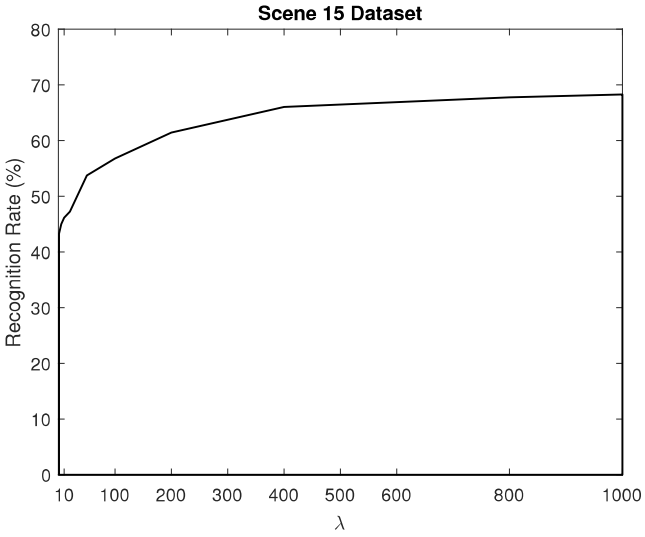


(b)

Figure 4.5: Recognition accuracy as a function of λ for the proposed method JEFW (a) and the kernel variant KJEFW (b) on the 8 Sports Event Categories Dataset. The training samples were set to 50 % of the whole dataset. The classifier used was the nearest neighbor classifier.

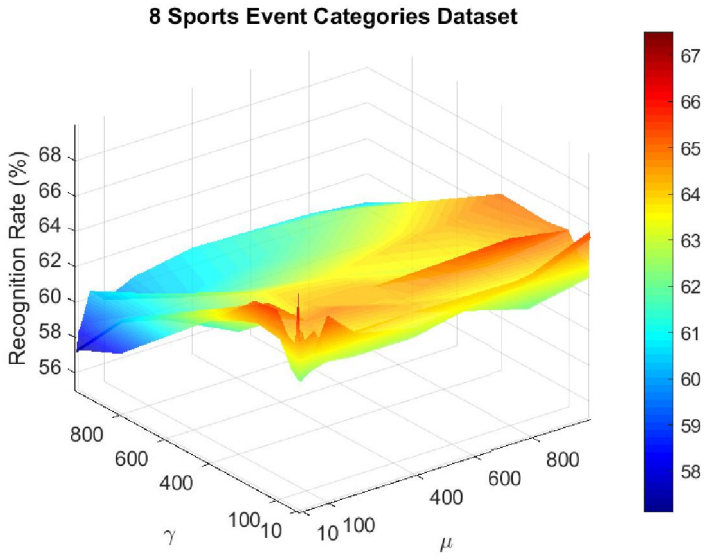


(a)

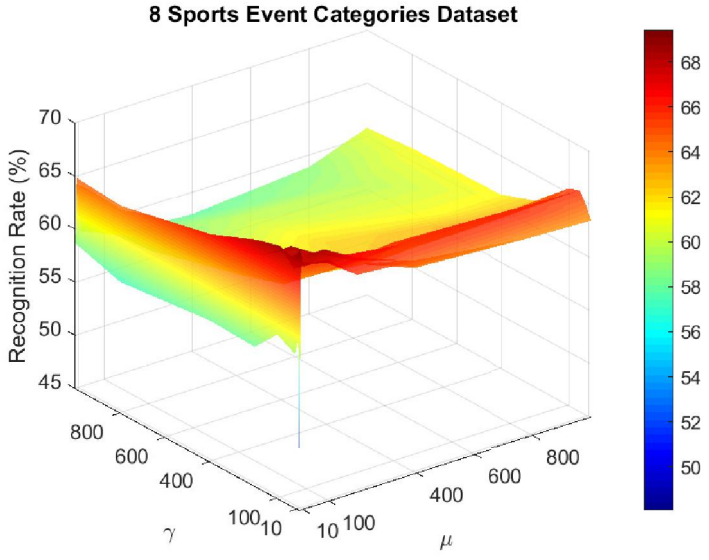


(b)

Figure 4.6: Recognition accuracy as a function of λ for the proposed method JEFW (a) and the kernelized variant KJEFW (b) on the Scene 15 Dataset. The training samples were set to 50 % of the whole dataset. The classifier used was the nearest neighbor classifier.

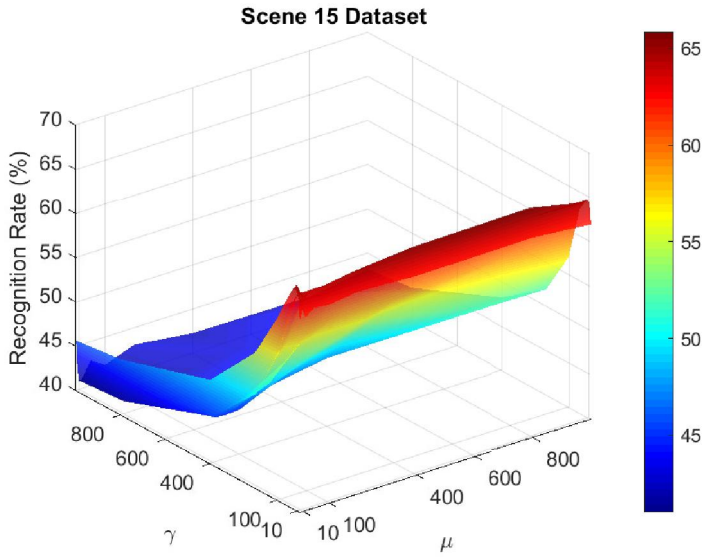


(a)

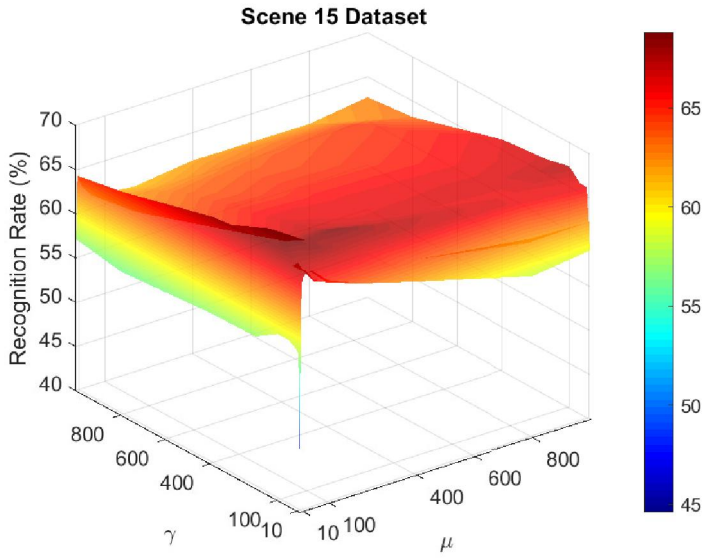


(b)

Figure 4.7: Recognition accuracy as a function of μ and γ for the proposed method JEFW (a) and the kernel variant KJEFW (b) on the 8 Sports Event Categories Dataset. The training samples were set to 50 % of the whole dataset. The classifier used was the nearest neighbor classifier.



(a)



(b)

Figure 4.8: Recognition accuracy as a function of μ and γ for the proposed method JEFW (a) and the kernel variant KJEFW (b) on the Scene 15 Dataset. The training samples were set to 50 % of the whole dataset. The classifier used was the nearest neighbor classifier.

5

Inductive Semi-supervised Learning with Graph Convolution Based Regression

This brief chapter introduces a framework for supervised and semi-supervised learning by estimating a non-linear embedding that incorporates Spectral Graph Convolutions structure. The proposed algorithm exploits data-driven graphs in two ways. First, it integrates data smoothness over graphs. Second, its regression loss function jointly uses the data and their graph in the sense that the regressor model sees convolved data samples. The resulting framework can solve the problem of over-fitting on local neighborhood structures for image data having varied natures like outdoor scenes, faces and man-made objects. The proposed Graph Convolution based Semi-supervised Embedding (GCSE) not only provides a new perspective to non-linear embedding research but also induces the standpoint on Spectral Graph Convolutions methods. A series of experiments are conducted on four image datasets in order to compare the proposed method with some state-of-art semi-supervised methods. This evaluation demonstrates the effectiveness of the proposed embedding method.

5.1/ Related work

5.1.1/ Spectral Convolution Networks: Theoretical background

Spectral convolutions on graph are defined as the multiplication of a signal $\mathbf{x} \in \mathbb{R}^N$ with a filter $\mathbf{g} = diag(\mathbf{w})$ where $\mathbf{w} \in \mathbb{R}^N$ is parameterized in Fourier domain:

$$\mathbf{g} \star \mathbf{x} = \mathbf{U}\, \mathbf{g}_w\, \mathbf{U}^T \mathbf{x} \tag{5.1}$$

where $\mathbf{U}^T\mathbf{x}$ and $\mathbf{g}_w$ could be regarded respectively as the graph Fourier transform of $\mathbf{x}$ and a function of $\mathbf{\Lambda}$ which is $\mathbf{g}(\mathbf{\Lambda})$, $\mathbf{\Lambda}$ being the diagonal matrix of the eigenvalues of the graph Laplacian $\mathbf{L}$. For solving the problem of expensive computing cost, [35] proposed Chebyshev polynomials to unfold $\mathbf{g}_w$:

$$\mathbf{g} \approx \sum_{k=0}^{K} \mathbf{w}_k\, T_k(\tilde{\mathbf{\Lambda}}) \tag{5.2}$$

where T_k refer to Chebyshev polynomials. $T_k(x) = 2\,x\,T_{k-1}(x) - T_{k-2}(x)$, and $T_0(x) = 1$, $T_1(x) = x$. The largest eigenvalue of the Laplacian matrix $\mathbf{L}$ is denoted by $\lambda_{\max}$. The

diagonal matrix $\tilde{\mathbf{\Lambda}}$ is given by $\tilde{\mathbf{\Lambda}} = \frac{2}{\lambda_{\max}}\mathbf{\Lambda} - \mathbf{I}$. $\mathbf{w}_k$ is Chebyshev coefficients. So the spectral convolutions on graphs with a truncated expansion in terms of Chebyshev polynomials could be rewritten:

$$\mathbf{g} \star \mathbf{x} \approx \sum_{k=0}^{K} \mathbf{w}_k \cdot T_k(\tilde{\mathbf{L}})\,\mathbf{x} \tag{5.3}$$

where $\cdot$ can represent the scalar product, $\tilde{\mathbf{L}} = \frac{2}{\lambda_{\max}}\mathbf{L} - \mathbf{I}$ and $\mathbf{L} = \mathbf{I} - \mathbf{D}^{-\frac{1}{2}}\mathbf{S}\,\mathbf{D}^{-\frac{1}{2}}$. If we just expand 1_{st}-order polynomial in (5.3) to limit convolution operation, and according to further approximate in the linear formulation where K=1 and $\lambda_{\max} \approx 2$ in the (5.3), we get :

$$\begin{aligned} \mathbf{g}_w \star \mathbf{x} &\approx \mathbf{w}_0 \cdot \mathbf{x} + \mathbf{w}_1 \cdot (\frac{2}{\lambda_{\max}}\mathbf{L} - \mathbf{I})\,\mathbf{x} \\ &= \mathbf{w}_0 \cdot \mathbf{x} - \mathbf{w}_1 \cdot \mathbf{D}^{-\frac{1}{2}}\mathbf{S}\mathbf{D}^{-\frac{1}{2}}\mathbf{x} \end{aligned} \tag{5.4}$$

where $\mathbf{w}_0$ and $\mathbf{w}_1$ are free parameters. If we constrain the number of parameters to address over-fitting and to minimize the number of matrix multiplications, $\mathbf{w} = \mathbf{w}_0 = -\mathbf{w}_1$ could be used in (5.4). We get the first order graph convolution as:

$$\mathbf{g}_w \star \mathbf{x} = \mathbf{w} \cdot (\mathbf{I} + \mathbf{D}^{-\frac{1}{2}}\mathbf{S}\mathbf{D}^{-\frac{1}{2}})\,\mathbf{x}$$

We approximate $\lambda_{\max} \approx 2$ which means that the eigenvalues of $\mathbf{I} + \mathbf{D}^{-\frac{1}{2}}\mathbf{S}\mathbf{D}^{-\frac{1}{2}}$ are between 0 and 2. This operator will lead to numerical unstable. So the renormalization trick is introduced which replaces $\mathbf{I} + \mathbf{D}^{-\frac{1}{2}}\mathbf{S}\mathbf{D}^{-\frac{1}{2}}$ by $\hat{\mathbf{D}}^{-\frac{1}{2}}\hat{\mathbf{S}}\hat{\mathbf{D}}^{-\frac{1}{2}}$, where $\hat{\mathbf{S}} = \mathbf{S} + \mathbf{I}$ and $\hat{\mathbf{D}}_{ii} = \sum_j \hat{\mathbf{S}}_{ij}$.

5.1.2/ Graph Convolutional Networks (GCN) for transductive semi-supervized learning

The Graph Convolutional Networks (GCN) algorithm in [52] presented a deep neural network approach for semi-supervised learning on graph-structured data. For semi-supervised label propagation, and with two layers, the GCN model has the following form:

$$\mathbf{E} = f(\mathbf{X}, \mathbf{A}) = \mathrm{softmax}\left(\hat{\mathbf{A}}\,\mathrm{ReLU}(\hat{\mathbf{A}}\,\mathbf{X}^T\,\mathbf{W}^{(0)})\mathbf{W}^{(1)}\right) \tag{5.5}$$

where $\mathbf{X}^T \in \mathbb{R}^{N\times d}$ denotes the input data with N samples and d dimensions, $\hat{\mathbf{A}}$ is a renormalized graph matrix $\hat{\mathbf{A}} = \hat{\mathbf{D}}^{-\frac{1}{2}}\hat{\mathbf{S}}\hat{\mathbf{D}}^{-\frac{1}{2}}$, $\mathbf{W}^{(0)} \in \mathbb{R}^{d\times H}$ is an input-to-hidden weight matrix for a hidden layer with H features, *ReLU ()* is the rectified linear activation function, and $\mathbf{W}^{(1)} \in \mathbb{R}^{H\times C}$ is a hidden-to-output weight matrix with C feature maps. C denotes the number of classes. $\mathrm{Softmax}$ denotes the softmax activation function. $\mathbf{E} \in \mathbb{R}^{N\times C}$ is the matrix of labels associated with all samples. The work in [52] estimates the model parameters $\mathbf{W}^{(0)}, \mathbf{W}^{(1)}, \ldots$ using deep learning tools in which the loss function is given by the cross-entropy error between the known labels and the output of the model.

5.2/ Overview of Proposed Approach

The GCN concept which is briefly introduced in the previous section motivates us to jointly use the data and their associated graph in order to derive a non-linear embedding

of the data and not only their labels as it is the case in [52]. The basic idea is to replace data samples by their convolution with a certain graph. To this end, we may use a first order approximation of spectral graph convolutions to replace the original samples data in regression model. Our goal is to estimate an inductive non-linear embedding in which the regressor is based on graph convolution.

In our proposed scheme, the sought non-linear projection should be as close as possible to a GCN model. Thus, we should impose that $\mathbf{Z} \approx \hat{\mathbf{D}}^{-\frac{1}{2}}\hat{\mathbf{S}}\,\hat{\mathbf{D}}^{-\frac{1}{2}}\mathbf{X}^T\,\mathbf{W}$. Thus, the proposed linear regression model is given by the unknown matrix $\mathbf{W} \in \mathbb{R}^{N\times N}$ which can be seen as filter parameters or transform matrix. The non-linear projection $\mathbf{Z}$ is the convolved signal matrix. We also could add a bias term $\mathbf{b}$ to the regression model. Thus, we have:

$$\mathbf{Z} \approx \hat{\mathbf{D}}^{-\frac{1}{2}}\hat{\mathbf{S}}\hat{\mathbf{D}}^{-\frac{1}{2}}\mathbf{X}^T\,\mathbf{W} + \mathbf{1}\,\mathbf{b}^T \tag{5.6}$$

Our proposed model not only inherits the latent advantages of spectral graph convolutions in reducing the problem of over-fitting on local neighborhood structures for graphs with wide node degree distributions, (e.g., scene and face images), but also provides the advantages of margin-based discriminant embedding and manifold smoothness. So the objective function focuses on the joint estimation of the non-linear projection data $\mathbf{Z}$, the linear transform matrix $\mathbf{W}$ and the shift vector $\mathbf{b}$. Thus, these unknowns are estimated by minimizing the following criterion:

$$\begin{aligned} h(\mathbf{Z},\mathbf{W},\mathbf{b}) &= tr(\mathbf{Z}^T(\mathbf{L}+\lambda\,\tilde{\mathbf{M}}_l)\mathbf{Z}) + \mu\,(\|\mathbf{W}\|_2^2 + \gamma\,\left\|\hat{\mathbf{D}}^{-\frac{1}{2}}\hat{\mathbf{S}}\hat{\mathbf{D}}^{-\frac{1}{2}}\mathbf{X}^T\,\mathbf{W} + \mathbf{1}\mathbf{b}^T - \mathbf{Z}\right\|_2^2) \\ &s.t.\ \mathbf{Z}^T\tilde{\mathbf{D}}_l\mathbf{Z} = \mathbf{I} \end{aligned} \tag{5.7}$$

In the sequel, $\mathbf{L}+\lambda\,\tilde{\mathbf{M}}_l$ will be denoted by $\mathbf{V}$. λ, μ and γ are regularization parameters. The above criterion simultaneously provides graph smoothness in locality preserving (imposed by the term $tr(\mathbf{Z}^T\,\mathbf{L}\,\mathbf{Z})$, margin maximization of labeled data samples (imposed by the term $tr(\mathbf{Z}^T\tilde{\mathbf{M}}_l\,\mathbf{Z})$, and spectral graph convolution based regression. The above citerion imposes ℓ_2 regularization on the transform matrix $\mathbf{W}$.

In the sequel, we show how the optimal solution for function (5.7) can be derived. We vanish the derivatives of the objective function (5.7) with respect to $\mathbf{W}$ and $\mathbf{b}$. This yields:

$$\mathbf{b} = \frac{1}{N}(\mathbf{Z}^T\mathbf{1} - \mathbf{W}^T\hat{\mathbf{X}}^T\mathbf{1}) \tag{5.8}$$

$$\mathbf{W} = \gamma(\gamma\hat{\mathbf{X}}^T\mathbf{H}_c\mathbf{H}_c^T\hat{\mathbf{X}} + \mathbf{I})^{-1}\hat{\mathbf{X}}^T\mathbf{H}_c\mathbf{Z} \tag{5.9}$$

where $\hat{\mathbf{X}} = \hat{\mathbf{D}}^{-\frac{1}{2}}\hat{\mathbf{S}}\hat{\mathbf{D}}^{-\frac{1}{2}}\mathbf{X}^T$ and $\mathbf{H}_c = \mathbf{I} - \frac{1}{\mathbf{N}}\mathbf{1}\mathbf{1}^T$. Besides, let $\mathbf{P}$ denotes the matrix $\gamma(\gamma\hat{\mathbf{X}}^T\mathbf{H}_c\mathbf{H}_c^T\hat{\mathbf{X}} + \mathbf{I})^{-1}\hat{\mathbf{X}}^T\mathbf{H}_c$, thus Eq. (5.9) can be rewritten as $\mathbf{W} = \mathbf{P}\,\mathbf{Z}$. So the spectral graph regression could be deduced by:

$$\begin{aligned} \hat{\mathbf{X}}\mathbf{W} + \mathbf{1}\mathbf{b}^T &= \hat{\mathbf{X}}\,\mathbf{P}\,\mathbf{Z} + \frac{1}{N}\mathbf{1}\mathbf{1}^T\mathbf{Z} - \frac{1}{N}\mathbf{1}\mathbf{1}^T\hat{\mathbf{X}}\,\mathbf{P}\,\mathbf{Z} \\ &= (\mathbf{I} - \frac{1}{N}\mathbf{1}\mathbf{1}^T)\,\hat{\mathbf{X}}\,\mathbf{P}\,\mathbf{Z} + \frac{1}{N}\mathbf{1}\mathbf{1}^T\mathbf{Z} \\ &= \mathbf{H}_c\,\hat{\mathbf{X}}\,\mathbf{P}\,\mathbf{Z} + \frac{1}{N}\mathbf{1}\mathbf{1}^T\mathbf{Z} \end{aligned} \tag{5.10}$$

Algorithm 3 Proposed GCSE algorithm

Input: Data matrix: $\mathbf{X}$; graph matrix $\mathbf{S}$; parameters: μ, λ and γ.
Output: Non-linear embedding matrix $\mathbf{Z}$; linear transform $\mathbf{W}$ and bias term $\mathbf{b}$;

1: Renormalize the matrix $\hat{\mathbf{S}} = \mathbf{I} + \mathbf{S}$ using $\hat{\mathbf{D}}^{-\frac{1}{2}}\hat{\mathbf{S}}\hat{\mathbf{D}}^{-\frac{1}{2}}$;
2: Compute the spectral graph convolutions matrix $\hat{\mathbf{X}} = \hat{\mathbf{D}}^{-\frac{1}{2}}\hat{\mathbf{S}}\hat{\mathbf{D}}^{-\frac{1}{2}}\mathbf{X}^T$;
3: Estimate $\mathbf{Z}$, $\mathbf{W}$ and $\mathbf{b}$ using Eqs. (5.11), (5.9) and (5.8) respectively;

Let $\mathbf{Q} = (\mathbf{I} - \frac{1}{N}\mathbf{1}\mathbf{1}^T)\hat{\mathbf{X}}\,\mathbf{P} + \frac{1}{N}\mathbf{1}\mathbf{1}^T$, thus Eq. (5.10) can be written in a more compact form $\hat{\mathbf{X}}\mathbf{W} + \mathbf{1}\mathbf{b}^T = \mathbf{Q}\,\mathbf{Z}$. By plugging Eqs. (5.8), (5.9) and (5.10), into (5.7), this one becomes:

$$\begin{aligned} h &= tr(\mathbf{Z}^T\mathbf{V}\,\mathbf{Z}) + \mu\, tr(\mathbf{Z}^T\mathbf{P}^T\mathbf{P}\,\mathbf{Z}) + \mu\,\gamma\, tr((\mathbf{Q}\,\mathbf{Z} - \mathbf{Z})^T(\mathbf{Q}\,\mathbf{Z} - \mathbf{Z})) \\ &= tr(\mathbf{Z}^T(\mathbf{V} + \mu\mathbf{P}^T\mathbf{P} + \mu\gamma(\mathbf{Q} - \mathbf{I})^T(\mathbf{Q} - \mathbf{I}))\mathbf{Z}) \end{aligned}$$

The constrained optimization problem becomes:

$$\begin{aligned} \mathbf{Z} \;&=\; \arg\min_{\mathbf{Z}} tr(\mathbf{Z}^T(\mathbf{V} + \mu\,\mathbf{P}^T\mathbf{P} + \mu\,\gamma\,(\mathbf{Q} - \mathbf{I})^T(\mathbf{Q} - \mathbf{I}))\mathbf{Z}) \\ &s.t.\;\; \mathbf{Z}^T\tilde{\mathbf{D}}_l\mathbf{Z} = \mathbf{I} \end{aligned} \tag{5.11}$$

The optimization problem in (5.11) is a generalized eigen-decomposition problem that can be solved efficiently. The solution for the $\mathbf{Z}$ matrix is given by the eigenvectors corresponding to the smallest eigenvalues.

Once the non-linear projection $\mathbf{Z}$ is estimated, the associated regression model ($\mathbf{W}$ and $\mathbf{b}$) are obtained using Eqs. (5.9) and (5.8), respectively. Algorithm 3 summarizes the main steps of the proposed graph convolution based semi-supervised embedding (GCSE). A graphical illustration of the proposed method is given in Figure 5.1. The inductive property of the proposed method allows to project an unseen test data sample $\mathbf{x}_u \in \mathbb{R}^d$. This is given by $\mathbf{z}_u = \mathbf{W}^T\hat{\mathbf{x}}_u + \mathbf{b}$. The convolved version $\hat{\mathbf{x}}_u$ is computed as follows. First, we compute the edge weights between $\mathbf{x}_u$ and the original data samples $\mathbf{X}$. This allows expanding the original graph matrix, $\mathbf{S}$, by one row and one column. This expanded graph matrix is then renormalized in the same way described above. $\hat{\mathbf{x}}_u$ is obtained by $\hat{\mathbf{x}}_u^T = \mathbf{a}_{N+1}\,\mathbf{X'}^T$ where $\mathbf{a}_{N+1}$ denotes the last row of the normalized graph matrix and $\mathbf{X'}^T = [\mathbf{X}^T; \mathbf{x}_u{}^T]$.

5.2.1/ Differences with deep learning models

A convolutional neural network (CNN) [54, 58] is a special neural network that has been widely applied to a variety of pattern recognition problems, such as computer vision, speech recognition, etc. The architecture of CNNs consists of a cascade of layers where each layer can be a convolutional layer, a sub-sampling layer, or a fully connected layer. The CNNs can provide an end-to-end solution to the image analysis and classification.

On the other hand, the Graph Convolutional Networks are a family of algorithms for semi-supervised learning that use the collection of data samples together with their associated graph. It motivates the convolution via a localized first-order approximation of spectral graph convolutions. It is worth noting that although our model is inspired by the concept of Graph Convolution presented in [52], our work has several differences with the latter. First, our work addresses flexible non-linear data embedding while the work in [52]

addresses label estimation using several layers of Graph Convolutions. Second, our work provides an inductive model that is able to embed unseen data samples, whereas the work in GCN is a transductive model. Third, our model is obtained using a closed-form solution, whereas the solution in GCN is based on iterative schemes. Fourth, the main application domain in GCN is the semi-supervised document classification for which binary graphs are already defined and fixed. In our work, we analyze image datasets for which similarity graphs are more challenging to estimate.

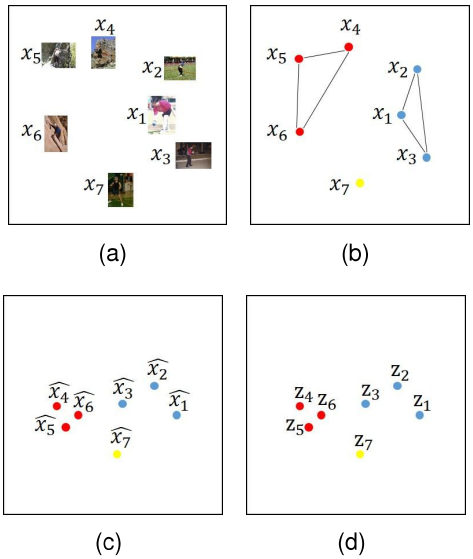


Figure 5.1: Different steps of the proposed method. (a) Original image data. (b) Constructed pairwise similarity graph. (c) Blended data samples using $\hat{\mathbf{D}}^{-\frac{1}{2}}\hat{\mathbf{S}}\hat{\mathbf{D}}^{-\frac{1}{2}}\mathbf{X}^T$. (d) The obtained nonlinear projections.

5.3/ Experiments and Results

This section presents the performance of the proposed embedding on different kinds of image data which include scene, face and object datasets. We use the semi-supervised setting since it is more challenging than the supervised one. The datasets used are 8 Sports Event Categories dataset [60], Scene 15 dataset [55], Extended YALE Face dataset [30] and COIL-20 object [72] dataset, which have already been introducted in the previous chapters. In the 8 Sports Event Categories and Scene 15 datasets, we use block-based Local Binary Patterns (LBP) [81] as image descriptor. For these two datasets, the number of blocks is set to 10×10 and the LBP descriptor is the uniform one having 59 features. For the face datasets, due to the small size of the face images, we use image raw brightness as image descriptor.

5.3.1/ Experimental setup

We compare our proposed GCSE method with several state-of-the-art algorithms: Flexible Graph-based Semi-supervised Embedding algorithm (FGSE) [21], Manifold Regularized Deep Learning Architecture Algorithm (MRDL) [127], Kernel Flexible Model Em-

bedding (KFME) [22], Joint Embedding Learning and Sparse Regression (JELSR) [46], Supervised Laplacian Eigenmaps (SLE) [89], Semi-Supervised Discriminant Analysis (SDA) [7], Semi-Supervised Discriminant Embedding (SDE) [126], LLE [92]. All the above methods provide data embedding except the KFME method which is a label propagation method. Once the embedding is obtained, the data are classified in the obtained space using the Nearest Neighbor Classifier (NN). The MRDL method is used with two layers in our tests. The sparse graph $\mathbf{S}$ is computed using the method proposed in [127] which proposes a kernel sparse algorithm for the similarity matrix estimation. We use the normalized graph Laplacian matrix $\mathbf{L} = \mathbf{I} - \mathbf{D}^{-\frac{1}{2}}\mathbf{S}\mathbf{D}^{-\frac{1}{2}}$. For the JELSR and FGSE methods, we use two types of graphs: the classic KNN graph and the kernel sparse graph of [127] to compare the performance of the different graphs. The proposed method has three balance parameters: λ, μ and γ. We set each parameter to a subset of values belonging to $\{10^{-3}, 10^{-2}, 10^{-1}, 1, 10^{1}, 10^{2}, 10^{3}, 10^{6}, 10^{9}\}$. We then report the recognition accuracy (using the best average recognition rate) of all methods from the best parameter configuration. All results are obtained with ten random splits of the data into a labeled set and an unlabeled set. For the labeled sets, two different percentages are considered for every dataset. According to [46], JELSR algorithm is used for unsupervised feature selection by computing the scores of all features. It also can be used for graph-based embedding.

5.3.2/ Method comparison

The performance of the different competing methods is summarized in Tables 5.1, 5.2, 5.3 and 5.4. The results correspond to four different datasets and different labeling percentages. These results correspond to an average over 10 random splits. In these tables, the two methods JELSR and FGSE were used with two types of graph: KNN graphs and sparse graphs.

Figures 5.2 depicts the average performance of the competing methods (KFME, LLE, JELSR, MRDL, FGSE and our proposed algorithm (GCSE)) as a function of the number of the features in the projection space. This figure corresponds to the 8 Sports and Scene 15 datasets. The KFME algorithm does not depend on the feature dimension since it is a label propagation method. The maximum dimension of SDA method is given by $C - 1$, where C is the number of classes.

From the obtained results depicted in the previous tables and figures, we can draw the following conclusions:

- Our proposed algorithm GCSE has achieved very good performance in all four datasets: 8 Sports Event Categories Dataset, Scene 15 Dataset, Extended YALE Face Dataset and COIL-20 Object Dataset. For the first two datasets, the superiority of the proposed method is obvious. This can be explained by the fact the use of Graph Convolution principle in the regression function can tackle the class high variability. The proposed method was slightly outperformed by KFME on the COIL-20 dataset corresponding to the case of 20% of labeled data.

- By inspecting the results obtained by the FGSE (sparse), FGSE (KNN), JELSR (sparse), JELSR (KNN), we can observe that the kernel sparse graph used in graph smoothness has significantly improved the performance of the same frameworks that use the classic KNN graph.

Table 5.1: The best average recognition rate in (%) on the 8 Sports Event Categories dataset (10 random splits).

Method / 8 Sports	P = 50%	P = 70%
LLE [92]	54.92	59.10
SLE [89]	51.40	50.90
SDA [7]	63.46	66.06
SDE [126]	51.98	55.96
MRDL [127]	51.77	52.85
KFME [22]	62.58	65.03
JELSR (KNN) [46]	52.46	55.48
JELSR (sparse)	55.92	57.60
FGSE (KNN) [21]	60.96	63.24
FGSE (sparse)	64.72	67.18
GCSE	**67.04**	**69.97**

- From Figure 5.2, we can observe that by increasing the feature dimension the performance cannot be improved. In fact, the highest rate is always reached with very few features indicating that the proposed method has achieved a very good dimensionality reduction.

Table 5.2: Average recognition rate in (%) on the Scene 15 dataset (10 random splits).

Method / Scene 15	P = 50%	P = 70%
LLE [92]	44.26	47.42
SLE [89]	50.48	50.65
SDA [7]	61.52	63.73
SDE [126]	46.10	48.07
MRDL [127]	46.59	47.91
KFME [22]	60.89	63.74
JELSR (KNN) [46]	41.37	44.24
JELSR (sparse)	51.83	58.59
FGSE (KNN) [21]	50.96	55.62
FGSE (sparse)	64.78	68.17
GCSE	**67.26**	**70.36**

Table 5.3: Average recognition rate in (%) on the Extended Yale dataset (10 random splits).

Method / Extended YALE	P = 20%	P = 40%
LLE [92]	91.47	95.75
SLE [89]	83.20	93.39
SDA [7]	89.96	96.54
SDE [126]	85.92	92.76
MRDL [127]	76.78	78.97
KFME [22]	90.39	92.85
JELSR (KNN) [46]	75.03	84.54
JELSR (sparse)	85.31	90.13
FGSE (KNN) [21]	93.36	98.18
FGSE (sparse)	93.71	98.31
GCSE	**94.74**	**98.53**

Table 5.4: Average recognition rate in (%) on the COIL-20 Object dataset (10 random splits).

Method / COIL20	P = 10%	P = 20%
LLE [92]	91.81	94.71
SLE [89]	82.03	88.56
SDA [7]	95.33	98.07
SDE [126]	89.10	95.33
MRDL [127]	88.00	88.86
KFME [22]	**96.98**	98.56
JELSR (KNN) [46]	85.48	93.01
JELSR (sparse)	93.80	96.88
FGSE (KNN) [21]	86.19	93.61
FGSE (sparse)	93.60	97.83
GCSE	96.60	**99.50**

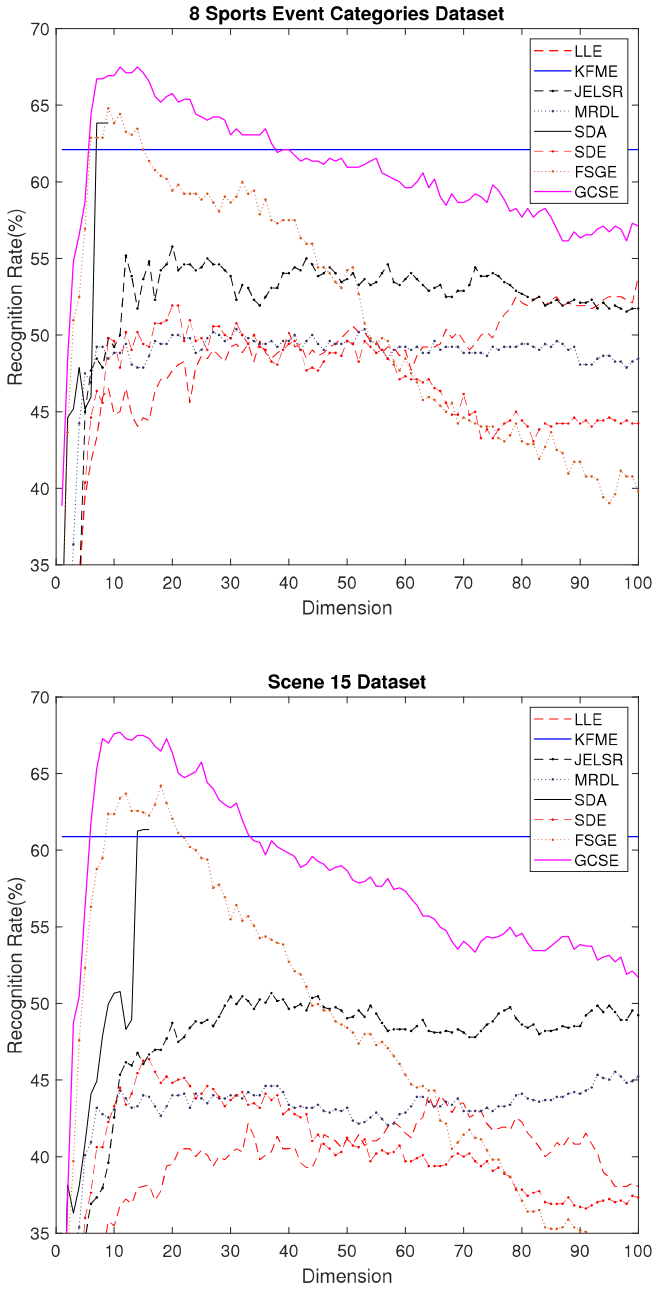


Figure 5.2: Recognition accuracy vs. feature dimension for 8 Sports Event Categories Dataset and Scene 15 Dataset. The unlabeled samples per class were 50%. The classifier used was 1-NN.

5.4/ Conclusion

We proposed a framework for discriminative non-linear Graph-based embedding with Spectral Graph Convolutions Structure in this chapter. This framework can solve the over-fitting on local neighborhood structures for graphs. The framework combines many criteria: manifold smoothness, Margin Discriminant Embedding and regression with Graph Spectral Convolutions. Experiments on scene, face and object image datasets have shown the superiority of this model with respect to many competing algorithms.

6

Semi-supervised Elastic Manifold Embedding with Deep Learning Architecture

Graph-based embedding aims to reduce the dimension of high dimensional data and to extract relevant features for learning tasks. In this letter, we propose an Elastic graph-based embedding with deep architecture which deeply explores the structural information of the data. We introduce a flexible deep learning that can overcome the limitations and weaknesses of single-layer learning models. The proposed deep architecture incorporates the geometrical manifold structure of the data. The resulting framework can be used for semi-supervised and supervised settings. Besides, the resulting optimization problems can be solved efficiently. We apply the algorithm on five public image datasets including scene, face and object datasets. These experiments demonstrate the effectiveness of the proposed embedding method, and also show that the proposed method compares favorably with many competing state-of-the-art graph-based methods.

6.1/ Short Review of Deep learning

Deep learning has fueled great strides in a variety of computer vision problems, such as object detection, motion tracking, action recognition, human pose estimation, and semantic segmentation. It allows computational models of multiple processing layers to learn and represent data with multiple levels of abstraction mimicking how the brain perceives and understands multi-modal information, thus implicitly capturing intricate structures of large-scale data. The ambition to create a system that simulates the human brain fueled the initial development of neural networks. Table 6.1 summarizes some important milestones in the history of neural networks and machine learning, leading up to the era of deep learning.

Table 6.1: Important milestones in the history of neural networks and machine learning, leading up to the era of deep learning.

Milestone/contribution	Contributor year
MCP model	McCulloch & Pitts, 1943
Backpropagation	Werbos, 1974
Boltzmann Machine	Ackley, Hinton & Sejnowski, 1985
Recurrent Neural Network	Jordan, 1986
Autoencoders	Rumelhart, Hinton et al., 1986
LeNet [57]	LeCun, 1990
LSTM [44]	Hochreiter & Schmidhuber, 1997
Deep Belief Network [43]	Hinton, 2006
Deep Boltzmann Machine [94]	Salakhutdinov & Hinton, 2009
AlexNet [54]	Krizhevsky, Sutskever, & Hinton, 2012
ResNet [37]	Kaiming He, Xiangyu Zhang et al., 2016
GCN [52]	Thomas N. Kipf, & Max Welling, 2017

6.2/ Overview of Proposed Approach

In this section, we will introduce the Elastic Manifold Embedding with deep architecture. The main idea is to utilize deep architecture learning to extract useful high-level features from low-level ones. Part of works have already be introduced on Chapter 3. In Subsection 3.1, the authors introduced a flexible semi-supervised feature extraction method [21].

Our learning proceeds layer by layer. Each layer has two main steps: (i) Estimating a data graph over the current data representation, and (ii) Computing the Elastic embedding (i.e., the nonlinear embedded data). We proceed as follows:

STEP 1. Kernelized and sparse graph.

In this step, we construct a kernelized and sparse graph over the current layer data. Let $\mathbf{S}$ denote the current graph. We were inspired by the graph construction method that is presented in [127]. This method provides a kernelized sparse graph by minimizing the following objective function:

$$\min_{\mathbf{S}\geq 0} \left\|\phi(\mathbf{X}) - \phi(\mathbf{X})\,\mathbf{S}\right\|_2^2 + \alpha \left\|\mathbf{S}\right\|_{\frac{1}{2}} + \beta \left\|\mathbf{S}\right\|_2^2 \tag{6.1}$$

where α and β are two positive regularization parameters. $\phi(\mathbf{X})$ is the kernel representation of the current input data $\mathbf{X}$.

The basic idea is to compute a non-negative graph affinity matrix using data self representation in kernel space with two regularization terms. The first regularization term, $\|\mathbf{S}\|_{\frac{1}{2}} = \sum_i^N \sum_j^N \sqrt{S(i,j)}$, is responsible for the graph sparsity. The second term, $\|\mathbf{S}\|_2^2 = \sum_i \sum_j S^2(i,j)$, is responsible for the stability of the estimated graph. Details on how to solve the above optimization can be found in [127].

STEP 2. Elastic manifold learning.

This step estimates the nonlinear embedded data $\mathbf{Z}$ from the current input data representation $\mathbf{X}$ and its associated graph $\mathbf{S}$. In order to compute the non-linear embedding matrix $\mathbf{Z}$ one should optimize the objective function in Eq. (3.3). In this elastic manifold

learning paradigm, the non-linear embedded data and the linear model are simultaneously estimated.

We vanish the derivatives of the objective function Eq. (3.3) with respect to $\mathbf{W}$ and $\mathbf{b}$. This yields $\frac{\partial e}{\partial \mathbf{W}} = \mathbf{0}$ and $\frac{\partial e}{\partial \mathbf{b}} = \mathbf{0}$.

After some algebraic manipulations, we have:

$$\mathbf{b} = \frac{1}{N}(\mathbf{Z}^T\mathbf{1} - \mathbf{W}^T\mathbf{X}\,\mathbf{1}) \tag{6.2}$$

$$\mathbf{W} = \gamma(\gamma\mathbf{X}_c\mathbf{X}_c^T + \mathbf{I})^{-1}\mathbf{X}_c\mathbf{Z} = \mathbf{A}\mathbf{Z} \tag{6.3}$$

where $\mathbf{A} = \gamma(\gamma\mathbf{X}_c\mathbf{X}_c^T + \mathbf{I})^{-1}\mathbf{X}_c$. $\mathbf{X}_c$ is the centered data matrix, i.e., $\mathbf{X}_c = \mathbf{X}\mathbf{H}_c$ with $\mathbf{H}_c$ being the centering matrix $\mathbf{H}_c = \mathbf{I} - \frac{1}{\mathbf{N}}\mathbf{1}\mathbf{1}^T$. We use the above expression for $\mathbf{W}$ and $\mathbf{b}$ in the regression function:

$$\begin{aligned}\mathbf{X}^T\mathbf{W} + \mathbf{1}\mathbf{b}^T &= \mathbf{X}^T\mathbf{A}\mathbf{Z} + \frac{1}{N}\mathbf{1}\mathbf{1}^T\mathbf{Z} - \frac{1}{N}\mathbf{1}\mathbf{1}^T\mathbf{X}^T\mathbf{A}\mathbf{Z}\\ &= (\mathbf{I} - \frac{1}{N}\mathbf{1}\mathbf{1}^T)\mathbf{X}^T\mathbf{A}\mathbf{Z} + \frac{1}{N}\mathbf{1}\mathbf{1}^T\mathbf{Z}\\ &= \mathbf{H}_c\mathbf{X}^T\mathbf{A}\mathbf{Z} + \frac{1}{N}\mathbf{1}\mathbf{1}^T\mathbf{Z}\\ &= \mathbf{B}\mathbf{Z}\end{aligned} \tag{6.4}$$

where $\mathbf{B} = \mathbf{H}_c\mathbf{X}^T\mathbf{A} + \frac{1}{N}\mathbf{1}\mathbf{1}^T$.

Finally, Eq. (3.3). can be written as

$$\begin{aligned}&tr(\mathbf{Z}^T\mathbf{L}_1\mathbf{Z}) + \mu \cdot tr(\mathbf{Z}^T\mathbf{A}^T\mathbf{A}\mathbf{Z}) + \mu\gamma \cdot tr((\mathbf{B}\mathbf{Z} - \mathbf{Z})^T(\mathbf{B}\mathbf{Z} - \mathbf{Z}))\\ &= tr(\mathbf{Z}^T(\mathbf{L}_1 + \mu\mathbf{A}^T\mathbf{A} + \mu\,\gamma(\mathbf{B} - \mathbf{I})^T(\mathbf{B} - \mathbf{I}))\mathbf{Z})\\ &= tr(\mathbf{Z}^T(\mathbf{L}_1 + \mathbf{E})\mathbf{Z})\end{aligned} \tag{6.5}$$

where $\mathbf{E} = \mu\mathbf{A}^T\mathbf{A} + \mu\,\gamma(\mathbf{B} - \mathbf{I})^T(\mathbf{B} - \mathbf{I})$ and $\mathbf{L}_1 = \mathbf{L} + \lambda\,\widetilde{\mathbf{M}_l}$.

Thus, the non-linear embedding $\mathbf{Z}$ is estimated by minimizing the above criterion under the constraint used in the criterion Eq. (3.3):

$$\mathbf{Z} = \arg\min_{\mathbf{Z}} tr(\mathbf{Z}^T(\mathbf{L}_1 + \mathbf{E})\mathbf{Z})\ s.t.\ \mathbf{Z}^T\tilde{\mathbf{D}}_l\mathbf{Z} = \mathbf{I} \tag{6.6}$$

where $\tilde{\mathbf{D}}_l$ is the augmented diagonal matrix associated with Eq. (3.3).

Thus, $\mathbf{Z}$ can be solved by generalized eigenvalue decomposition. Once $\mathbf{Z}$ is estimated, the corresponding regression model ($\mathbf{W}$, $\mathbf{b}$) is estimated by Eq. (6.3) and (6.2), respectively.

Algorithm 4 shows the main steps of the proposed Semi-supervised Elastic Manifold Embedding with Deep Architecture.

We emphasize that the proposed framework also provides a deep linear model represented by the cascade of all linear models $\{(\mathbf{W}_1, \mathbf{b}_1), ..., (\mathbf{W}_L, \mathbf{b}_L)\}$.

Algorithm 4 Semi-supervised Elastic Manifold Embedding with Deep Architecture

Input:

- Data samples: $\mathbf{X} \in \mathbb{R}^{\mathbf{d}\times\mathbf{N}}$;
- Number of layers, L
- Regularization parameters: μ, λ and γ.

Output:

- Non-linear embedding matrix $\mathbf{Z} \in \mathbb{R}^{m\times N}$ (m is the embedding dimensionality, $m \leq N$);
- Linear model $\{(\mathbf{W}_1, \mathbf{b}_1), ..., (\mathbf{W}_L, \mathbf{b}_L)\}$

Process:

- Initialize the current layer number, $cl = 1$
- Initialize the current layer input data $\mathbf{X}_{cl} = \mathbf{X}$;

1: **while** $cl \leq L$ **do**
2: Based on the current layer data, $\mathbf{X}_{cl}$, compute the kernelized sparse graph matrix $\mathbf{S}$ using Eq. (6.1). Compute the normalized graph Laplacian matrix $\mathbf{L} = \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{S}\mathbf{D}^{1/2}$;
3: Based on the current Laplacian matrix, compute the non-linear embedded data $\mathbf{Z} \in \mathbb{R}^{N\times N}$ using Eq. (6.6). Estimate $\mathbf{W}_{cl}$ and $\mathbf{b}_{cl}$ using Eq. (6.3), and (6.2), respectively;
4: $cl \leftarrow cl + 1$;
5: Set the next layer input data by using $\mathbf{X}_{cl} = \mathbf{Z}^T(1 : N, 1 : m) \in \mathbb{R}^{m\times N}$($m$ is the next layer embedding dimensionality, $m \leq N$).

Table 6.2: Best average recognition rate (%) obtained on the 8 Sports Event dataset using 10 random splits with two different percentages for the training part.

Dataset	**8 Sports Event**	
Method	P=50%	P=70%
LLE [92]	54.92	59.10
SDA [7]	63.46	66.06
LE [2]	51.48	54.07
GFHF [134]	62.25	64.29
RMGT [67]	62.58	64.20
SDE [12]	51.98	55.96
MRDL [127]	58.48	60.51
KFME [22]	62.58	65.03
MLAN [75]	56.30	59.84
JELSR [46]	55.92	57.60
JELSR (KNN) [46]	52.46	55.48
FSSE (KNN) [21]	59.96	63.24
Proposed (one layer)	64.72	67.18
Proposed (two layers)	65.96	68.53
Proposed (three layers)	**66.46**	**69.33**

Table 6.3: Best average recognition rate (%) obtained on the Scene 15 dataset using 10 random splits with two different percentages for the training part.

Dataset	**Scene 15**	
Method	P=50%	P=70%
LLE [92]	44.26	47.42
SDA [7]	61.52	63.73
LE [2]	41.47	43.68
GFHF [134]	61.58	63.57
RMGT [67]	61.59	63.49
SDE [12]	46.10	48.07
MRDL [127]	52.16	54.64
KFME [22]	60.89	63.74
MLAN [75]	40.71	42.53
JELSR [46]	51.83	58.59
JELSR (KNN) [46]	41.37	44.24
FSSE (KNN) [21]	50.96	55.62
Proposed (one layer)	64.78	68.17
Proposed (two layers)	65.93	68.41
Proposed (three layers)	**66.43**	**69.01**

Table 6.4: Best average recognition rate (%) obtained on the COIL-20 dataset using 10 random splits with two different percentages for the training part.

Dataset	**COIL-20**	
Method	P=10%	P=20%
LLE [92]	91.81	94.71
SDA [7]	95.33	98.07
LE [2]	90.39	96.38
GFHF [134]	96.01	98.14
RMGT [67]	96.06	98.18
SDE [12]	89.10	95.33
MRDL [127]	93.02	96.10
KFME [22]	**96.98**	**98.56**
MLAN [75]	92.26	95.21
JELSR [46]	93.80	96.88
JELSR (KNN) [46]	75.03	84.54
FSSE (KNN) [21]	86.19	93.61
Proposed (one layer)	93.60	97.83
Proposed (two layers)	96.33	98.38
Proposed (three layers)	96.32	98.50

Table 6.5: Best average recognition rate (%) obtained on the Extended Yale B dataset using 10 random splits with two different percentages for the training part.

Dataset	**Extended Yale B**	
Method	P=20%	P=40%
LLE [92]	91.47	95.75
SDA [7]	89.96	96.54
LE [2]	80.01	86.39
GFHF [134]	89.26	92.74
RMGT [67]	89.82	93.04
SDE [12]	85.92	92.76
MRDL [127]	92.20	96.32
KFME [22]	90.39	92.85
MLAN [75]	76.27	85.96
JELSR [46]	85.31	90.13
JELSR (KNN) [46]	75.03	84.54
FSSE (KNN) [21]	93.36	98.18
Proposed (one layer)	93.71	98.31
Proposed (two layers)	94.71	98.93
Proposed (three layers)	**95.04**	**99.09**

Table 6.6: Best average recognition rate (%) obtained on the PF01 dataset using 10 random splits with two different percentages for the training part.

Dataset	**PF01 dataset**	
Method	P=30%	P=50%
LLE [92]	67.91	78.53
SDA [7]	78.29	89.12
LE [2]	45.17	52.72
GFHF [134]	52.66	61.92
RMGT [67]	53.15	62.50
SDE [12]	66.04	72.93
MRDL [127]	67.80	76.63
KFME [22]	56.67	66.01
MLAN [75]	43.31	55.30
JELSR [46]	59.33	68.86
JELSR (KNN) [46]	52.16	63.31
FSSE (KNN) [21]	77.67	89.77
Proposed (one layer)	80.39	90.56
Proposed (two layers)	82.66	**92.54**
Proposed (three layers)	**83.05**	92.50

6.3/ Experiments and Results

6.3.1/ Datasets

Our method will be evaluated on six different public datasets including four scene datasets and one face dataset. These are as follows: 8 Sports Event Categories Dataset [60], Scene 15 Dataset [55], Extended YALE Face Dataset B [30], COIL-20 Object dataset [72] and Postech Faces 01' Face Dataset (PF01) [51], which all datasets except PF01 have already been introducted in the previous chapters. For PF01, it contains the true-color face images of 103 people, representing 17 various images per person. All of the people in the database are Asians. We randomly select 30% and 50% of data as the training set.

6.3.2/ Experimental setup

For the datasets depicting outdoor scenes, we use the block-based Local Binary Patterns as the image descriptor. The utilization of the LBP descriptor was motivated by its efficiency. Indeed, the LBP descriptor is a well-known image descriptor that is fast and invariant to monotonic illumination changes. The local LBP descriptor is the uniform one having 59 features. Thus, for an image with b non-overlapping blocks, the length of the image descriptor is $59\,b$. In our implementation, we used 100 blocks. For the COIL-20 Object, Postech Faces 01' Face and Extended YALE Face Dataset, we use image raw brightnesses.

We compare the proposed framework with several state-of-the-art algorithms: Locally Linear Embedding (LLE) [92], Semi-Supervised Discriminant Analysis (SDA) [7], Laplacian Eigenmaps (LE) [2], GFHF [134], RMGT [67], Semi-Supervised Discriminant Embedding (SDE) [125], Manifold Regularized Deep Learning Architecture Algorithm (MRDL) [127], Kernel Flexible Model Embedding (KFME) [22], Multi-View Learning With Adaptive Neighbors (MLAN) [75], Joint Embedding Learning and Sparse Regression (JELSR) [46], Flexible Semi-Supervised Embedding (FSSE) [21]. Note that MLAN [75] was proposed for multiview cases (i.e., each image has several types of descriptors). It is used for unsupervised clustering and semi-supervised learning. In our experiments, we only use one single view when MLAN is used. All the above methods provide data embedding except the GFHF, RMGT, KFME and MLAN methods which are label propagation methods. Once the embedding is computed, data are classified in the obtained space using the Nearest Neighbor (NN) Classifier. In MRDL algorithm, two layers are used in our tests. According to [46], JELSR algorithm is used for unsupervised feature selection by computing the scores of all features. It also can be used for graph-based embedding. For the JELSR method, we use two types of graphs: the classic KNN graph and the kernel sparse graph of [127] to compare the performance of the different graphs. Our proposed method has three balance parameters: λ, μ and γ. We set each parameter to a subset of values belonging to $\{1, 10^1, 10^2, 10^3\}$. In addition to the regularization parameter, the intermediate layers need to fix the dimension m. This belonged to $\{C, 2C, 3C, 4C\}$ where C is the number of classes. We then report the best recognition accuracy (best average recognition rate) of all methods from the best parameter configuration. All results are obtained with ten random splits of the data into a train set and a test set. For train sets, two different percentages are considered for every dataset.

6.3.3/ Methods comparison

In this section, we compare our proposed method with the state-of-the-art methods on the five public datasets. Tables 6.2, 6.3, 6.4, 6.5 and 6.6 summarizes the obtained average recognition rate obtained by our proposed algorithm and the competing methods: LLE, SDA, LE, GFHF, RMGT, SDE, MRDL, KFME, MLAN, JELSR, JELSR (KNN), FSSE (KNN). The results are associated with the five image datasets.

Figures 6.1 depicts the performance of the competing methods LLE, SDA, SDE, KFME, JELSR and MRDL as well as that of our proposed method as a function of the number of non-linear features, for the two image datasets: 8 Sports Event Categories Dataset and Extended YALE Face Dataset. The framework has been tested with one, two, and three layers. In the figure, one curve is associated with each case. We emphasize that the KFME algorithm does not depend on the feature dimension since it is a label embedding method. In fact, the used competing methods are either linear or nonlinear. For the nonlinear methods, the maximum feature dimension is the number of samples N. For the linear methods, the maximum feature dimension is the number of original features d. In the figure, we presented the accuracy curves till dimension 100 in the projection space. This is motivated by the fact that highest performances are reached with a low number of features.

6.3.4/ Effect of the Model Depth

In these experiments, we investigate the influence of model depth (number of layers) on classification performance. We report results on the 8 Sports Event Categories Dataset and Extended YALE Face Dataset. Figures 6.2 illustrates the recognition rate versus the number of layers.

6.3.5/ Analysis of results

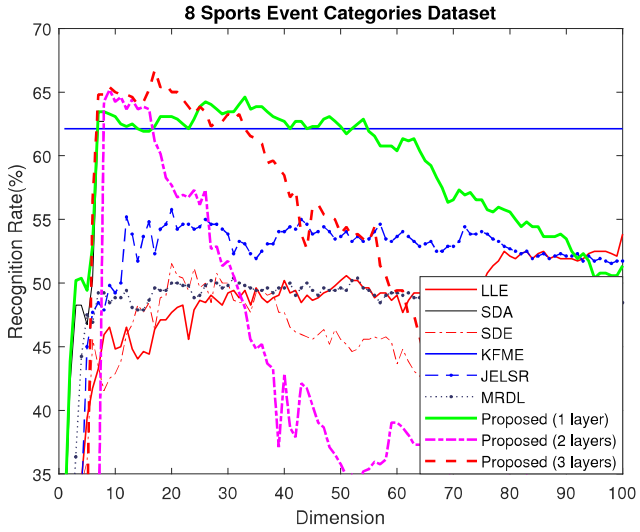
From the obtained tables and figures, we can see that the proposed method has achieved superior recognition performance on different types of images. Based on these experimental results, a number of interesting observations can be made. These are as follows:

- According to Tables 6.2, 6.3, 6.4, 6.5 and 6.6, we can observe that our proposed method outperformed all compared state-of-the-art methods for different percentages for the training part on four public image datasets, except for the COIL-20 Object Dataset. The proposed method was slightly outperformed by KFME on the COIL-20 dataset corresponding to the 10% and 20% train data experiment.
- From Figures 6.1, we can observe that by increasing the feature dimension, the rate cannot be improved. In fact, the highest rate is always reached with very few features indicating that the proposed method has achieved a very good dimensionality reduction. Unlike many non-linear embedding methods whose performance deteriorates by increasing the number of features, ours do not have such disadvantage.
- From Figures 6.2, we can observe that with the increase of layers of the deep architecture, the recognition accuracy will increase with the best results being obtained with four or five layers. We also observe that for models deeper than eight layers on

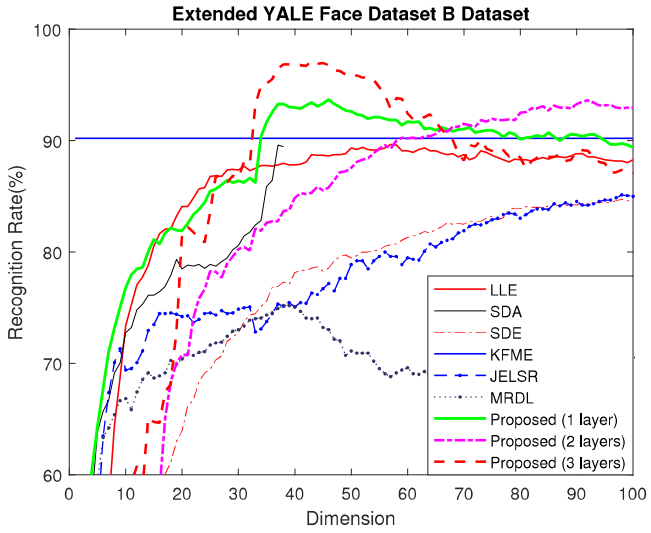
the 8 Sports Event Categories and Extended YALE Face Dataset, the performance cannot be increased.

6.4/ Conclusion

We proposed a framework for Elastic graph-based embedding with deep learning architecture which explores the structural information of the data in this chapter. This framework can solve the over-fitting on local neighborhood structures in graphs. The framework integrates many criteria like Manifold Smoothness, Elastic Manifold Embedding and Deep Learning architecture. Experiments on the scene, face and object image datasets have shown the superiority of the proposed method with respect to many competing algorithms.

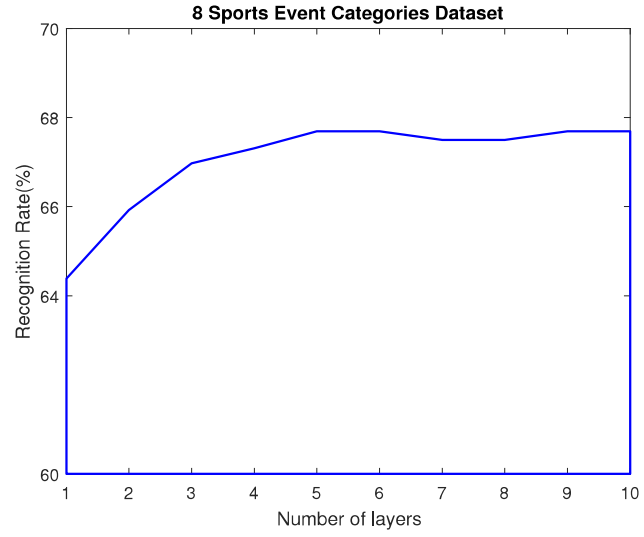


(a)

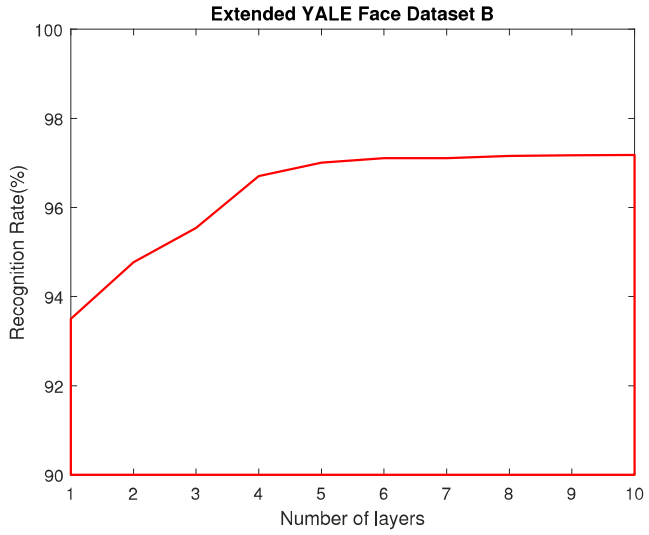


(b)

Figure 6.1: Recognition accuracy as a function of the feature dimension in the embedded space for 8 Sports Event Categories Dataset and Extended YALE Face Dataset. The training samples were set to 50 % and 20 % of the two datasets, respectively. The classifier used was the nearest neighbor classifier.



(a)



(b)

Figure 6.2: Influence of model depth (number of layers) on classification performance for 8 Sports Event Categories Dataset and Extended YALE Face Dataset. The training samples were set to 50 % and 20 % of the two datasets, respectively. The classifier used was the nearest neighbor classifier.

III

Conclusion and Perspectives

7

Conclusion and Perspectives

7.1/ Conclusion

In this thesis, we focused on Graph-based Manifold Learning techniques and its applications to image recognition and classification.

First, we introduced and reviewed some state-of-the-art methods in Graph-based Manifold Learning in Chapters 1 and 2. This included topics about Graph Construction and Large-Scale Graphs in Section 2.1, Unsupervised Graph-Based Manifold Learning in Section 2.2, Semi-supervised Graph-Based Manifold Learning in Section 2.3, Graph-based Label Propagation in Section 2.4, Graph-Based Dimensionality Reduction in Section 2.5, Graph-based Feature Selection in Section 2.6 and the other State-of-the-art topics on Graphs in Section 2.7 (e.g., Deep Learning on Graphs and Manifolds in Subsection 2.7.1, Graph-based application on Learning to Hash in Subsection 2.7.2).

A series of related algorithms are introduced in Chapters 3, 4, 5 and 6. These algorithms focus on nonlinear and inductive algorithms (i.e., the proposed methods are able to estimate the embedding of unseen samples). They are suitable for working in supervised and semi-supervised learning settings. While the proposed techniques are generic pattern recognition tools, our experimental evaluation was mainly conducted on image datasets. The specific context is as follows:

In Chapter 3, a novel nonlinear method called Flexible Discriminant graph-based Embedding with feature selection (FDEFS) is proposed. We seek a non-linear ans a linear representation of the data that can be suitable for generic learning tasks such as classification and clustering. Besides, a byproduct of the proposed embedding framework is the feature selection of the original features, where the estimated linear transformation matrix can be used for feature ranking and selection.

In Chapter 4, we investigate strategies and related algorithms to develop a joint graph-based embedding and an explicit feature weighting for getting a flexible and inductive nonlinear data representation on manifolds. The proposed criterion explicitly estimates the feature weights together with the projected data and the linear transformation such that data smoothness and large margins are achieved in the projection space. Moreover, this chapter introduces a kernel variant of the model in order to get an inductive nonlinear embedding that is close to a real nonlinear subspace for a good approximation of the embedded data.

In Chapter 5, we propose the graph convolution based semi-supervised Embedding (GCSE). It provides a new perspective to non-linear data embedding research, and

makes a link to signal processing on graph methods. The proposed method utilizes and exploits graphs in two ways. First, it deploys data smoothness over graphs. Second, its regression model is built on the joint use of the data and their graph in the sense that the regression model works with convolved data. The convolved data are obtained by feature propagation. The resulting scheme can solve and address the problem of over-fitting on local neighborhoods for image data of various types like faces, outdoor scenes, and man-made objects.

In Chapter 6, a flexible deep learning that can overcome the limitations and weaknesses of single-layer learning models is introduced. We call this strategy an Elastic graph-based embedding with deep architecture which deeply explores the structural information of the data. The resulting framework can be used for semi-supervised and supervised settings. Besides, the resulting optimization problems can be solved efficiently.

7.2/ Perspectives

Based on the methods proposed in this thesis, we can highlight several future tracks.

- The thesis work proposes several methods that are able to simultaneously derive linear and non-linear models for the semi-supervised and supervised learning setting. However, while these models are flexible and can be used for out-of-sample cases, most of the propsoed models are still shallow ones. Thus, it would be very appealing to replace the shallow non-linear and linear models by deep neural networks in the future work.
- The work mainly focuses on Graph-based Manifold Learning for Semi-supervised or Supervised applications. When we do not use the label information in the training stage, the proposed methods can be easily extended to the unsupervised case.
- The thesis work addresses single-view data where each sample or image is represented by one type of descriptors. Thus, it would be very interesting to extend the developed methods to the case of multi-view embedding or any other learning task. Indeed, image data can be represented by multiple views. Very often data can be collected from multiple sources or represented by several types of descriptors. For instance, images and videos can be described by various types of descriptors, e.g. HoG [17], LBP [82], GIST [83], Gabor [18], and any type of deep features that can be obtained trough transfer learning. These descriptors can capture different aspects of data and can be complementary to each other.
- In our work, we deploy the concept of feature propagation (usually used in Graph Convolution Networks) in order to get an enhanced linear regression that allows the embedding of unseen samples. Future work would investigate more sophisticated regression models by using advanced tricks like the use of iterative feature propagation or the joint performing of the graph estimation and the feature propagation. In general, future work can also target sophisticated GCNs for semi-supervised learning. Indeed, deep learning on graphs and manifolds is considered as a hot research topic.
- In our work, the ranking of the original features is obtained as a by product of the proposed learning models. Future work would assess this selection more thor-

oughly. In particular, performance will be quantified in several learning tasks like clustering and classification on the original selected features.

- Motivated by the graph-based label propagation methods (e.g., Gaussian Fields and Harmonic Function (GFHF) [134] and Flexible Manifold embedding (FME)), the label information can be retrieved by adopting a given criterion that involves the graph. Future work would modify the proposed frameworks such that they can deal directly with label distribution and their explicit mapping instead of data embedding.

- In this work, we consider that the graph matrix that describes the inter-connections of different samples is given as input. Future work would consider more sophisticated learning models in which both the graph and the embedding parameters are simultaneously estimated. By doing so, more optimal solutions are expected.

- Although the proposed methods are scalable to large-scale training datasets thanks to their inductive property, it would be of great interest to upgrade their learning models so that they can easily construct a graph over very large datasets. Indeed, the typical graph construction algorithms usually can not be used for large scale datasets since the size of the graph matrix is equal to the square of the training dataset size. Improving the scalability of the proposed approaches (i.e., we use the entire training set with labeled and unlabeled samples for model learning) could be carried out by adopting a bipartite graph via the use of Anchors. The use of anchors allows to have small graphs that can be efficiently constructed.

The plans of research career should can consider near-term prospect ($\leq$ 3 years), medium-term prospect ($<$ 10 years) and long-term prospect ($\geq$ 10 years).

- Near-term prospect stage. In fact, the related activities can address some topics described in Sections 2.1, 2.3, 2.2, 2.4, 2.5 and 2.6 of Chapter 2. These are related to various fields in Graph-based Manifold Learning. However, those topics have already experienced a long time development and research since LLE [92] and ISOMAP [104] methods published. More exactly, Semi-supervised learning using Gaussian fields and harmonic functions (ICML, 2003) [134] is the pioneering method in Graph-Based Semi-supervised Manifold Learning and Graph-based Label Propagation fields. LLE (Nature, 2000) [92], ISOMAP (Nature, 2000) [104], Laplacian Eigenmaps (NIPS, 2002) [2] and Locality Preserving Projection (NIPS, 2004) [40] are representative algorithms in the Unsupervised Graph-Based Manifold Learning field. Laplacian Score for Feature Selection [38] seems to be the pioneering work in Graph-based Feature Selection field. Even the topic of Graph-based Learning to Hash was developed since 2009 by Spectral Hashing [116]. It has to say I have already missed the bonus time of those fields above. These fields could be an opportunity to extend this thesis work in the a few year future.

  Deep Learning on Graphs and Manifolds is a novel field since the work by [42] (2015) that joint Deep Learning and Graph-based Manifold Learning. The work described in [52] proposed a spotlighting algorithm that is Graph Convolutional Networks (GCN) [52] since 2017. Due to its convincing performance and high interpretability, it has been a widely applied graph analysis method recently. Deep Learning on Graphs and Manifolds could be the major work in the near-term future.

- Medium-term prospect stage. Graph Theory and Manifold Learning are ones of the tools in Machine Learning field. The Medium-term prospect stage us t choose

fundamental, progressive and evolutionary field to utilize machine learning techniques, including Graph Theory, Manifold Learning, Deep Learning and other effective and novel tools, to develop Computer Vision. We realize that several advanced Computer Vision directions have been already launched along with the increasing computing ability of GPUs and CPUs. Which directions are suitable for my future research? I choose to focus on the Computer Vision related to Persons and Objects in the medium-term prospect stage. The human related tracks include Person (or Pedestrian and Vehicle) Re-identification, Crowd Counting, 3D Human (or Hand) Pose Estimation, Action Recognition (or Segmentation, Detection, Localization, Anticipation, Prediction, Forecasting). The object related tracks include Static Object (or Video-based Object) Recognition (or detection, segmentation and searching).

- Long-term prospect stage. In this stage, the goal is to build am academic team that to develop multiple research fields in Artificial Intelligence, and own industrial and engineering ability. Moreover, I will be committed to cultivating elite researchers in Information Technique field who will also follow my steps in striving for the scientific progress of my country in this stage.

7.3/ Publications during PhD Study

- **Ruifeng Zhu**, Fadi Dornaika* and Yassine Ruichek, Learning a discriminant graph-based embedding with feature selection for image categorization. Neural Networks(NN, Elsevier, IF: 7.197), 2019, 111: 35-46.

- **Ruifeng Zhu**, Fadi Dornaika* and Yassine Ruichek, Joint graph based embedding and feature weighting for image classification, Pattern Recognition(PR, Elsevier, IF: 5.785), 2019, 93: 458-469.

- **Ruifeng Zhu**, Fadi Dornaika* and Yassine Ruichek, Inductive Semi-supervised Learning with Graph Convolution Based Regression, Neurocomputing(Elsevier, IF: 4.072), Major Revise and Resubmit, 2019.

- **Ruifeng Zhu**, Fadi Dornaika* and Yassine Ruichek, Semi-supervised Elastic Manifold Embedding with Deep Learning Architecture, Pattern Recognition(PR, Elsevier, IF: 5.898), Accapted, 2019.

- **Ruifeng Zhu**, Fadi Dornaika* and Yassine Ruichek, Flexible and Discriminative Non-linear Embedding with Feature Selection for Image Classification, 2018 24th International Conference on Pattern Recognition (ICPR). IEEE, 2018: 3192-3197(Oral 10%).

- Fadi Dornaika*, Youssof El Traboulsi and **Ruifeng Zhu**, Robust and Flexible Graph-based Semi-supervised Embedding, 2018 24th International Conference on Pattern Recognition (ICPR). IEEE, 2018: 465-470(Poster).

- **Ruifeng Zhu**, Fadi Dornaika* and Yassine Ruichek,Discriminant Manifold Learning with Graph Convolution Based Regression for Image classification, 12th International workshop on Graph-Based Representation in Pattern Recognition (GbR), Tours, France, June 19-21, 2019(Accapted, Oral).

- **Ruifeng Zhu**, **Fadi Dornaika*** and Yassine Ruichek, A semi-supervised non-linear approach with graph-based Self-Adaptation Structure, 2020 25th International Conference on Pattern Recognition, Roma, Italy, Under Review.

Bibliography

[1] Peter N. Belhumeur, João P Hespanha, and David J. Kriegman. Eigenfaces vs. fisherfaces: Recognition using class specific linear projection. *IEEE Transactions on pattern analysis and machine intelligence*, 19(7):711–720, 1997.

[2] Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps and spectral techniques for embedding and clustering. In *Advances in Neural Information Processing Systems*, pages 585–591, 2002.

[3] Mikhail Belkin, Partha Niyogi, and Vikas Sindhwani. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *Journal of Machine Learning Research*, 7:2399–2434, 2006.

[4] Cheng Bin, Yang Jianchao, Yan Shuicheng, Fu Yun, and TS Huang. Learning with l1-graph for image analysis. *IEEE Transactions on Image Processing*, 19(4):858–866, 2010.

[5] Yuri Boykov, Olga Veksler, and Ramin Zabih. Fast approximate energy minimization via graph cuts. *IEEE Transactions on pattern analysis and machine intelligence*, 23(11):1222–1239, 2001.

[6] Michael M Bronstein, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst. Geometric deep learning: going beyond euclidean data. *IEEE Signal Processing Magazine*, 34(4):18–42, 2017.

[7] D. Cai, X. He, and J. Han. Semi-supervised discriminant analysis. In *2007 IEEE 11th International Conference on Computer Vision*, pages 1–7, Oct 2007.

[8] Deng Cai, Xiaofei He, Jiawei Han, et al. Isometric projection. In *AAAI*, pages 528–533, 2007.

[9] Deng Cai, Xiaofei He, Jiawei Han, and H-J Zhang. Orthogonal laplacianfaces for face recognition. *IEEE transactions on image processing*, 15(11):3608–3614, 2006.

[10] Deng Cai, Chiyuan Zhang, and Xiaofei He. Unsupervised feature selection for multi-cluster data. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 333–342, 2010.

[11] Shaosheng Cao, Wei Lu, and Qiongkai Xu. Deep neural networks for learning graph representations. In *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

[12] Hakan Cevikalp, Jakob J. Verbeek, Frédéric Jurie, and Alexander Kläser. Semi-supervised dimensionality reduction using pairwise equivalence constraints. In *International Conference on Computer Vision Theory and Applications*, volume 1, pages 489–496, 2008.

[13] Yan-Shuo Chang, Feiping Nie, Zhihui Li, Xiaojun Chang, and Heng Huang. Refined spectral clustering via embedded label propagation. *Neural computation*, 29(12):3381–3396, 2017.

[14] Sibao Chen, Haifeng Zhao, Min Kong, and Bin Luo. 2d-lpp: A two-dimensional extension of locality preserving projections. *Neurocomputing*, 70(4-6):912–921, 2007.

[15] Jian Cheng, Qingshan Liu, Hanqing Lu, and Yen-Wei Chen. Supervised kernel locality preserving projections for face recognition. *Neurocomputing*, 67:443–449, 2005.

[16] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[17] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, volume 1, pages 886–893. IEEE, 2005.

[18] John G Daugman. Complete discrete 2-d gabor transforms by neural networks for image analysis and compression. *IEEE Transactions on acoustics, speech, and signal processing*, 36(7):1169–1179, 1988.

[19] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in neural information processing systems*, pages 3844–3852, 2016.

[20] Gauthier Doquire and Michel Verleysen. A graph laplacian based approach to semi-supervised feature selection for regression problems. *Neurocomputing*, 121:5–13, 2013.

[21] Fadi Dornaika and Youssof El Traboulsi. Learning flexible graph-based semi-supervised embedding. *IEEE Transactions on Cybernetics*, 46(1):206–218, 2016.

[22] Youssof El Traboulsi, Fadi Dornaika, and Ammar Assoum. Kernel flexible manifold embedding for pattern classification. *Neurocomputing*, 167:517–527, 2015.

[23] Li Fei-Fei, Rob Fergus, and Pietro Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. *Computer vision and Image understanding*, 106(1):59–70, 2007.

[24] Charles D Feustel and Linda G Shapiro. The nearest neighbor problem in an abstract metric space. *Pattern Recognition Letters*, 1(2):125–128, 1982.

[25] Matthias Fey and Jan Eric Lenssen. Fast graph representation learning with pytorch geometric. https://github.com/rusty1s/pytorch_geometric, 2019.

[26] Ronald A Fisher. The use of multiple measurements in taxonomic problems. *Annals of eugenics*, 7(2):179–188, 1936.

[27] Hongyang Gao, Zhengyang Wang, and Shuiwang Ji. Large-scale learnable graph convolutional networks. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1416–1424. ACM, 2018.

[28] Tiezheng Ge, Kaiming He, and Jian Sun. Graph cuts for supervised binary coding. In *European Conference on Computer Vision*, pages 250–264. Springer, 2014.

[29] Xin Geng, De-Chuan Zhan, and Zhi-Hua Zhou. Supervised nonlinear dimensionality reduction for visualization and classification. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 35(6):1098–1107, 2005.

[30] A.S. Georghiades, P.N. Belhumeur, and D.J. Kriegman. From few to many: Illumination cone models for face recognition under variable lighting and pose. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(6):643–660, 2001.

[31] Aristides Gionis, Piotr Indyk, and Rajeev Motwani. Similarity search in high dimensions via hashing. In *Vldb*, pages 518–529, 1999.

[32] Chen Gong, Tongliang Liu, Dacheng Tao, Keren Fu, Enmei Tu, and Jie Yang. Deformed graph laplacian for semisupervised learning. *IEEE transactions on neural networks and learning systems*, 26(10):2261–2274, 2015.

[33] Donghai Guan, Weiwei Yuan, Young-Koo Lee, Kamran Najeebullah, and Mostofa Kamal Rasel. A review of ensemble learning based feature selection. *IETE Technical Review*, 31(3):190–198, 2014.

[34] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*, pages 1024–1034, 2017.

[35] David K. Hammond, Pierre Vandergheynst, and Rémi Gribonval. Wavelets on graphs via spectral graph theory. *Applied and Computational Harmonic Analysis*, 30(2):129–150, 2011.

[36] Yahong Han, Yi Yang, Yan Yan, Zhigang Ma, Nicu Sebe, and Xiaofang Zhou. Semisupervised feature selection via spline regression for video semantic recognition. *IEEE Transactions on Neural Networks and Learning Systems*, 26(2):252–264, 2014.

[37] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[38] Xiaofei He, Deng Cai, and Partha Niyogi. Laplacian score for feature selection. In *Advances in neural information processing systems*, pages 507–514, 2006.

[39] Xiaofei He, Deng Cai, Shuicheng Yan, and Hong-Jiang Zhang. Neighborhood preserving embedding. In *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*, volume 2, pages 1208–1213. IEEE, 2005.

[40] Xiaofei He and Partha Niyogi. Locality preserving projections. In *Advances in neural information processing systems*, pages 153–160, 2004.

[41] Xiaofei He, Shuicheng Yan, Yuxiao Hu, Partha Niyogi, and Hong-Jiang Zhang. Face recognition using laplacianfaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(3):328–340, 2005.

[42] Mikael Henaff, Joan Bruna, and Yann LeCun. Deep convolutional networks on graph-structured data. *arXiv preprint arXiv:1506.05163*, 2015.

[43] Geoffrey E Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006.

[44] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[45] Chenping Hou, Feiping Nie, Xuelong Li, Dongyun Yi, and Yi Wu. Joint embedding learning and sparse regression: A framework for unsupervised feature selection. *IEEE Transactions on Cybernetics*, 44(6):793–804, 2013.

[46] Chenping Hou, Feiping Nie, Xuelong Li, Dongyun Yi, and Yi Wu. Joint embedding learning and sparse regression: A framework for unsupervised feature selection. *IEEE Transactions on Cybernetics*, 44(6):793–804, 2014.

[47] Piotr Indyk and Rajeev Motwani. Approximate nearest neighbors: towards removing the curse of dimensionality. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 604–613, 1998.

[48] Go Irie, Zhenguo Li, Xiao-Ming Wu, and Shih-Fu Chang. Locally linear hashing for extracting non-linear manifolds. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2115–2122, 2014.

[49] Bingbing Jiang, Huanhuan Chen, Bo Yuan, and Xin Yao. Scalable graph-based semi-supervised learning through sparse bayesian model. *IEEE Transactions on Knowledge and Data Engineering*, 29(12):2758–2771, 2017.

[50] Qing-Yuan Jiang and Wu-Jun Li. Scalable graph hashing with feature transformation. In *IJCAI'15 Proceedings of the 24th International Conference on Artificial Intelligence*, pages 2248–2254, 2015.

[51] HC Kim, JW Sung, HM Je, SK Kim, BJ Jun, D Kim, and SY Bang. Asian face image database pf01. *Technical report*, 2001.

[52] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[53] Deguang Kong, Chris HQ Ding, Heng Huang, and Feiping Nie. An iterative locally linear embedding algorithm. *arXiv preprint arXiv:1206.6463*, 2012.

[54] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[55] Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 2169–2178. IEEE, 2006.

[56] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436, 2015.

[57] Yann LeCun, Bernhard E Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne E Hubbard, and Lawrence D Jackel. Handwritten digit recognition with a back-propagation network. In *Advances in neural information processing systems*, pages 396–404, 1990.

[58] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[59] Ron Levie, Federico Monti, Xavier Bresson, and Michael M Bronstein. Cayleynets: Graph convolutional neural networks with complex rational spectral filters. *IEEE Transactions on Signal Processing*, 67(1):97–109, 2018.

[60] Li-Jia Li and Li Fei-Fei. What, where and who? classifying events by scene and object recognition. In *IEEE Conference on Computer Vision*, pages 1–8. IEEE, 2007.

[61] Ming Li and Baozong Yuan. 2d-lda: A statistical linear discriminant analysis for image matrix. *Pattern Recognition Letters*, 26(5):527–532, 2005.

[62] Ruoyu Li, Sheng Wang, Feiyun Zhu, and Junzhou Huang. Adaptive graph convolutional neural networks. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[63] Xuelong Li, Di Hu, and Feiping Nie. Large graph hashing with spectral rotation. In *AAAI*, pages 2203–2209, 2017.

[64] Zechao Li and Jinhui Tang. Unsupervised feature selection via nonnegative spectral analysis and redundancy control. *IEEE Transactions on Image Processing*, 24(12):5343–5355, 2015.

[65] Guosheng Lin, Chunhua Shen, and Anton Van den Hengel. Supervised hashing using graph cuts and boosted decision trees. *IEEE transactions on pattern analysis and machine intelligence*, 37(11):2317–2331, 2015.

[66] Guangcan Liu, Zhouchen Lin, Shuicheng Yan, Ju Sun, Yong Yu, and Yi Ma. Robust recovery of subspace structures by low-rank representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(1):171–184, 2013.

[67] Wei Liu and Shih-Fu Chang. Robust multi-class transductive learning with graphs. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 381–388. IEEE, 2009.

[68] Wei Liu, Junfeng He, and Shih fu Chang. Large graph construction for scalable semi-supervised learning. In *Proceedings of the 27th International Conference on Machine Learning*, pages 679–686, 2010.

[69] Wei Liu, Jun Wang, Sanjiv Kumar, and Shih fu Chang. Hashing with graphs. In *Proceedings of the 28th International Conference on Machine Learning*, pages 1–8, 2011.

[70] L. V. D. Maaten and G. Hinton. Visualizing data using t-sne. *J. Mach. Learn. Res.*, 9(2605):2579–2605, 2008.

[71] Federico Monti, Davide Boscaini, Jonathan Masci, Emanuele Rodola, Jan Svoboda, and Michael M Bronstein. Geometric deep learning on graphs and manifolds using mixture model cnns. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5115–5124, 2017.

[72] Sameer A Nene, Shree K Nayar, Hiroshi Murase, et al. Columbia object image library(coil-20). Technical Report CUCS-005-96, Columbia University, 1996.

[73] Andrew Y. Ng, Michael I. Jordan, and Yair Weiss. On spectral clustering: Analysis and an algorithm. In *Advances in Neural Information Processing Systems 14*, pages 849–856, 2001.

[74] F. Nie, H. Huang, X. Cai, and C. Ding. Efficient and robust feature selection via joint l2,1-norms minimization. In *Adv. Neural Inf. Processing Systems*, volume 23, 2010.

[75] Feiping Nie, Guohao Cai, Jing Li, and Xuelong Li. Auto-weighted multi-view learning for image clustering and semi-supervised classification. *IEEE Transactions on Image Processing*, 27(3):1501–1511, 2018.

[76] Feiping Nie, Heng Huang, Xiao Cai, and Chris H Ding. Efficient and robust feature selection via joint $\ell_{2,1}$-norms minimization. In *Advances in neural information processing systems*, pages 1813–1821, 2010.

[77] Feiping Nie, Shiming Xiang, Yangqing Jia, Changshui Zhang, and Shuicheng Yan. Trace ratio criterion for feature selection. In *AAAI*, volume 2, pages 671–676, 2008.

[78] Feiping Nie, Shiming Xiang, Yun Liu, and Changshui Zhang. A general graph-based semi-supervised learning with novel class discovery. *Neural Computing and Applications*, 19(4):549–555, 2010.

[79] Feiping Nie, Dong Xu, Ivor Wai-Hung Tsang, and Changshui Zhang. Flexible manifold embedding: A framework for semi-supervised and unsupervised dimension reduction. *IEEE Transactions on Image Processing*, 19(7):1921–1932, 2010.

[80] Mathias Niepert, Mohamed Ahmed, and Konstantin Kutzkov. Learning convolutional neural networks for graphs. In *International conference on machine learning*, pages 2014–2023, 2016.

[81] Timo Ojala, Matti Pietikäinen, and David Harwood. A comparative study of texture measures with classification based on featured distributions. *Pattern Recognition*, 29(1):51–59, 1996.

[82] Timo Ojala, Matti Pietikainen, and Topi Maenpaa. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on pattern analysis and machine intelligence*, 24(7):971–987, 2002.

[83] Aude Oliva and Antonio Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *International journal of computer vision*, 42(3):145–175, 2001.

[84] Xi Peng, Zhiding Yu, Zhang Yi, and Huajin Tang. Constructing the l2-graph for robust subspace learning and subspace clustering. *IEEE Transactions on Systems, Man, and Cybernetics*, 47(4):1053–1066, 2017.

[85] Nathanaël Perraudin, Johan Paratte, David Shuman, Lionel Martin, Vassilis Kalofolias, Pierre Vandergheynst, and David K Hammond. GSPBOX: A toolbox for signal processing on graphs. https://epfl-lts2.github.io/gspbox-html, 2014.

[86] Mingjie Qian and Chengxiang Zhai. Robust unsupervised feature selection. In *Twenty-Third International Joint Conference on Artificial Intelligence*, 2013.

[87] Lishan Qiao, Limei Zhang, Songcan Chen, and Dinggang Shen. Data-driven graph construction and graph learning: A review. *Neurocomputing*, 312:336–351, 2018.

[88] Ariadna Quattoni and Antonio Torralba. Recognizing indoor scenes. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 413–420, 2009.

[89] Bogdan Raducanu and Fadi Dornaika. A supervised non-linear dimensionality reduction approach for manifold learning. *Pattern Recognition*, 45(6):2432–2444, 2012.

[90] Alberto Rossi, Matteo Tiezzi, Giovanna Maria Dimitri, Monica Bianchini, Marco Maggini, and Franco Scarselli. Inductive–transductive learning with graph neural networks. In *IAPR Workshop on Artificial Neural Networks in Pattern Recognition*, pages 201–212. Springer, 2018.

[91] Ryan Anthony Rossi, Rong Zhou, and Nesreen Ahmed. Deep inductive graph representation learning. *IEEE Transactions on Knowledge and Data Engineering*, 2018.

[92] Sam T. Roweis and Lawrence K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000.

[93] Ruslan Salakhutdinov and Geoffrey Hinton. Semantic hashing. *RBM*, 500(3):500, 2007.

[94] Ruslan Salakhutdinov and Geoffrey Hinton. Deep boltzmann machines. In *Artificial intelligence and statistics*, pages 448–455, 2009.

[95] Ferdinando S Samaria and Andy C Harter. Parameterisation of a stochastic model for human face identification. In *IEEE Workshop on Applications of Computer Vision*, pages 138–142. IEEE, 1994.

[96] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80, 2008.

[97] Youngjoo Seo, Michaël Defferrard, Pierre Vandergheynst, and Xavier Bresson. Structured sequence modeling with graph convolutional recurrent networks. In *International Conference on Neural Information Processing*, pages 362–373. Springer, 2018.

[98] Fumin Shen, Chunhua Shen, Qinfeng Shi, Anton Van Den Hengel, and Zhenmin Tang. Inductive hashing on manifolds. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1562–1569, 2013.

[99] Fumin Shen, Chunhua Shen, Qinfeng Shi, Anton Van den Hengel, Zhenmin Tang, and Heng Tao Shen. Hashing on nonlinear manifolds. *IEEE Transactions on Image Processing*, 24(6):1839–1851, 2015.

[100] Jianbo Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

[101] David I Shuman, Sunil K Narang, Pascal Frossard, Antonio Ortega, and Pierre Vandergheynst. The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. *IEEE signal processing magazine*, 30(3):83–98, 2013.

[102] Hong Tang, Tao Fang, and Peng-Fei Shi. Laplacian linear discriminant analysis. *Pattern Recognition*, 39(1):136–139, 2006.

[103] J.S. Taylor and N. Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, Cambridge, England, 2006.

[104] J B Tenenbaum, V De Silva, and J C Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.

[105] Matthew Turk and Alex Pentland. Face recognition using eigenfaces. In *Proceedings. 1991 IEEE computer society conference on computer vision and pattern recognition*, pages 586–587, 1991.

[106] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.

[107] De Wang, Feiping Nie, and Heng Huang. Large-scale adaptive semi-supervised learning via unified inductive and transductive model. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 482–491, 2014.

[108] Fei Wang, Xin Wang, Daoqiang Zhang, Changshui Zhang, and Tao Li. marginface: A novel face recognition method by average neighborhood margin maximization. *Pattern Recognition*, 42(11):2863–2875, 2009.

[109] Fei Wang, Xin Wang, Daoqiang Zhang, Changshui Zhang, and Tao Li. marginface: A novel face recognition method by average neighborhood margin maximization. *Pattern Recognition*, 42(11):2863–2875, 2009.

[110] Fei Wang and Changshui Zhang. Label propagation through linear neighborhoods. *IEEE Transactions on Knowledge and Data Engineering*, 20(1):55–67, 2007.

[111] Jingdong Wang, Ting Zhang, Nicu Sebe, Heng Tao Shen, et al. A survey on learning to hash. *IEEE transactions on pattern analysis and machine intelligence*, 40(4):769–790, 2017.

[112] Jun Wang, Sanjiv Kumar, and Shih-Fu Chang. Semi-supervised hashing for scalable image retrieval. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 3424–3431. IEEE, 2010.

[113] Jun Wang, Sanjiv Kumar, and Shih fu Chang. Sequential projection learning for hashing with compact codes. In *Proceedings of the 27th International Conference on Machine Learning*, pages 1127–1134, 2010.

[114] Jun Wang, Wei Liu, Sanjiv Kumar, and Shih-Fu Chang. Learning to hash for indexing big data—a survey. *Proceedings of the IEEE*, 104(1):34–57, 2016.

[115] Zhengxia Wang, Xiaofeng Zhu, Ehsan Adeli, Yingying Zhu, Chen Zu, Feiping Nie, Dinggang Shen, and Guorong Wu. Progressive graph-based transductive learning for multi-modal classification of brain disorder disease. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 291–299. Springer, 2016.

[116] Yair Weiss, Antonio Torralba, and Rob Fergus. Spectral hashing. In *Advances in neural information processing systems*, pages 1753–1760, 2009.

[117] John Wright, Allen Y Yang, Arvind Ganesh, S Shankar Sastry, and Yi Ma. Robust face recognition via sparse representation. *IEEE transactions on pattern analysis and machine intelligence*, 31(2):210–227, 2008.

[118] Chenxia Wu, Jianke Zhu, Deng Cai, Chun Chen, and Jiajun Bu. Semi-supervised nonlinear hashing using bootstrap sequential projection learning. *IEEE Transactions on Knowledge and Data Engineering*, 25(6):1380–1393, 2012.

[119] Jian Yang, David Zhang, Alejandro F Frangi, and Jing-yu Yang. Two-dimensional pca: a new approach to appearance-based face representation and recognition. *IEEE transactions on pattern analysis and machine intelligence*, 26(1):131–137, 2004.

[120] Wuyi Yang, Shuwu Zhang, and Wei Liang. A graph based subspace semi-supervised learning framework for dimensionality reduction. In *European Conference on Computer Vision*, pages 664–677. Springer, 2008.

[121] Yi Yang, Heng Tao Shen, Zhigang Ma, Zi Huang, and Xiaofang Zhou. L2, 1-norm regularized discriminative feature selection for unsupervised. In *Twenty-Second International Joint Conference on Artificial Intelligence*, 2011.

[122] Zhilin Yang, William W Cohen, and Ruslan Salakhutdinov. Revisiting semi-supervised learning with graph embeddings. *arXiv preprint arXiv:1603.08861*, 2016.

[123] Peter N Yianilos. Data structures and algorithms for nearest neighbor search in general metric spaces. In *Soda*, volume 93, pages 311–21, 1993.

[124] Jiaxuan You, Rex Ying, Xiang Ren, William L Hamilton, and Jure Leskovec. Graphrnn: A deep generative model for graphs. *arXiv preprint arXiv:1802.08773*, 2018.

[125] G. Yu, G. Zhang, C. Domeniconi, Z. Yu, and J. You. Semi-supervised classification based on random subspace dimensionality reduction. *Pattern Recognition*, 45:1119–1135, 2012.

[126] Guoxian Yu, Guoji Zhang, Carlotta Domeniconi, Zhiwen Yu, and Jane You. Semi-supervised classification based on random subspace dimensionality reduction. *Pattern Recognition*, 45(3):1119–1135, 2012.

[127] Yuan Yuan, Lichao Mou, and Xiaoqiang Lu. Scene recognition by manifold regularized deep learning architecture. *IEEE Transactions on Neural Networks and Learning Systems*, 26(10):2222–2233, 2015.

[128] Yan-Ming Zhang, Kaizhu Huang, Guang-Gang Geng, and Cheng-Lin Liu. Mtc: A fast and robust graph-based transductive learning method. *IEEE transactions on neural networks and learning systems*, 26(9):1979–1991, 2014.

[129] Zheng Zhao and Huan Liu. Spectral feature selection for supervised and unsupervised learning. In *Proceedings of the 24th international conference on Machine learning*, pages 1151–1157, 2007.

[130] W. Zheng, Z. Lin, and H. Wang. L1-norm kernel discriminant analysis via bayes error bound optimization for robust feature extraction. *IEEE Transactions on Neural Networks and Learning Systems*, 25(4):793–803, 2014.

[131] Zhonglong Zheng, Fan Yang, Wenan Tan, Jiong Jia, and Jie Yang. Gabor feature-based face recognition using supervised locality preserving projection. *Signal Processing*, 87(10):2473–2483, 2007.

[132] Dengyong Zhou, Olivier Bousquet, Thomas N Lal, Jason Weston, and Bernhard Schölkopf. Learning with local and global consistency. In *Advances in neural information processing systems*, pages 321–328, 2004.

[133] Xiaofeng Zhu, Xuelong Li, Shichao Zhang, Chunhua Ju, and Xindong Wu. Robust joint graph sparse coding for unsupervised spectral feature selection. *IEEE transactions on neural networks and learning systems*, 28(6):1263–1275, 2016.

[134] Xiaojin Zhu, Zoubin Ghahramani, and John D Lafferty. Semi-supervised learning using gaussian fields and harmonic functions. In *International Conference on Machine learning*, pages 912–919, 2003.

[135] Liansheng Zhuang, Haoyuan Gao, Zhouchen Lin, Yi Ma, Xin Zhang, and Nenghai Yu. Non-negative low rank and sparse graph for semi-supervised learning. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2328–2335, 2012.

[136] Liansheng Zhuang, Zihan Zhou, Shenghua Gao, Jingwen Yin, Zhouchen Lin, and Yi Ma. Label information guided graph construction for semi-supervised learning. *IEEE Transactions on Image Processing*, 26(9):4182–4192, 2017.

[137] Olga Zoidi, Anastasios Tefas, Nikos Nikolaidis, and Ioannis Pitas. Positive and negative label propagations. *IEEE Transactions on Circuits and Systems for Video Technology*, 28(2):342–355, 2016.

Abstract:

Graph-based Manifold Learning algorithms are regarded as a powerful technique for feature extraction and dimensionality reduction in Pattern Recogniton, Computer Vision and Machine Learning fields. These algorithms uzilize sample information contained in the item-item similarity and weighted matrix to reveal the intrinstic geometric structure of manifold. It exhibits the low dimensional structure in the high dimensional data. This motivates me to develop Graph-based Manifold Learning techniques on Pattern Recognition, specially, application to image categorization. The experimental datasets of thesis correspond to several categories of public image datasets such as face datasets, indoor and outdoor scene datasets, objects datasets and so on. Several approaches are proposed in this thesis: 1) A novel nonlinear method called Flexible Discriminant graph-based Embedding with feature selection (FDEFS) is proposed. We seek a non-linear and a linear representation of the data that can be suitable for generic learning tasks such as classification and clustering. Besides, a byproduct of the proposed embedding framework is the feature selection of the original features, where the estimated linear transformation matrix can be used for feature ranking and selection. 2) We investigate strategies and related algorithms to develop a joint graph-based embedding and an explicit feature weighting for getting a flexible and inductive nonlinear data representation on manifolds. The proposed criterion explicitly estimates the feature weights together with the projected data and the linear transformation such that data smoothness and large margins are achieved in the projection space. Moreover, this chapter introduces a kernel variant of the model in order to get an inductive nonlinear embedding that is close to a real nonlinear subspace for a good approximation of the embedded data. 3) We propose the graph convolution based semi-supervised Embedding (GCSE). It provides a new perspective to non-linear data embedding research, and makes a link to signal processing on graph methods. The proposed method utilizes and exploits graphs in two ways. First, it deploys data smoothness over graphs. Second, its regression model is built on the joint use of the data and their graph in the sense that the regression model works with convolved data. The convolved data are obtained by feature propagation. 4) A flexible deep learning that can overcome the limitations and weaknesses of single-layer learning models is introduced. We call this strategy an Elastic graph-based embedding with deep architecture which deeply explores the structural information of the data. The resulting framework can be used for semi-supervised and supervised settings. Besides, the resulting optimization problems can be solved efficiently.

Keywords: Machine Learning, Manifold Learning, Graph-based Embedding, Semi-supervised Learning, Feature Selection, Image Categorization, Pattern Recognition, Computer Vision