https://doi.org/10.2352/EI.2024.36.10.IPAS-253


# Data and Label Graph Fusion for Semi-supervised Learning: Application to Image Categorization

***A. Baradaaji*[1], *F. Dornaika*[1,2,*], *I. Arganda-Carreras*[1,2,3,4]**
[1] ***University of the Basque Country UPV/EHU, San Sebastian, Spain***
[2] ***IKERBASQUE, Basque Foundation of Science, Bilbao, Spain***
[3] ***DIPC, San Sebastian, Spain***
[4] ***Biofisika Institute, Leioa, Spain***

## Abstract

*In this paper, a novel framework for semi-supervised learning based on graphs is introduced. We present an innovative approach for concurrently estimating label inference and performing a linear transformation. This specific linear transformation is directed towards achieving a discriminant subspace, which effectively reduces the dimensionality of the data. To enhance the semi-supervised learning process, our framework places a strong emphasis on leveraging the inherent data structure and incorporating the information provided by soft labels from the available unlabeled samples. The method we propose ultimately results in an improved discriminative linear transformation. The effectiveness of our approach is verified through a series of experiments conducted on real image datasets. These experiments not only confirm the efficacy of our proposed method but also demonstrate its superior performance when compared to semi-supervised methods that simultaneously incorporate integration and label inference.*

**Keywords:** *Graph-based semi-supervised learning, data graph, label graph, graph fusion, pattern recognition.*

## Introduction

Graph-based semi-supervised learning (GSSL) encounters several challenges. Firstly, its scalability to large datasets is not straightforward, resulting in significant computational overhead and time-intensive processes. Moreover, the choice of labeled samples can exert a substantial impact on the final performance. GSSL methods must also exhibit robustness in the face of potential mislabeling or outliers, which can adversely affect their performance. Lastly, the quantification of confidence and security in semi-supervised methods remains an unsolved issue.

In this paper, we introduce an efficient semi-supervised learning approach known as DLGF (Data and Label Graph Fusion for Semi-Supervised Learning). DLGF addresses these challenges by iteratively and concurrently estimating the soft label matrix for unlabeled data and the linear transform. It harnesses the power of graphs, which offer rich and meaningful information to enhance the capabilities of semi-supervised algorithms. This method integrates multiple graphs into a unified graph using an adaptive fusion process that amalgamates data and the current soft-label estimates. Furthermore, DLGF autonomously assigns appropriate weights to each graph matrix.

The paper contains the following contributions:

- A method is developed to estimate the soft label matrix and the linear transform simultaneously.
- An adaptive label graph is created using the current soft label estimates and the available labels, which is then merged with the data graph to perform two types of regularization: Regularization of the labels and regularization of the transformed data.

The rest of this paper is structured as follows: We begin by introducing the notations and reviewing related work. Subsequently, we present the semi-supervised model. Following that, we provide the results of our experiments conducted on five real image datasets, along with comparisons to other semi-supervised methods. Finally, we draw our conclusions.

## Related work and notations

In this paper, big bold letters represent matrices and small bold letters represent vectors. The training data matrix is denoted $\mathbf{X} \in \mathbb{R}^{D\times(l+u)}$, where $D$ is the dimension of the samples and $N = l+u$ denotes the number of samples, $l$ and $u$ denote the number of labeled and unlabeled samples, respectively. $Tr(.)$ denotes the trace of a matrix. Table 1 summarizes the main notations used in this paper.

In the field of graph-based semi-supervised learning, two main approaches have been explored: label propagation on graphs (e.g. [8, 10, 3]) and graph-based embedding methods (e.g., [2, 7, 13]) with structural constraints (such as sparsity) on the graphs. SLDA [14] and ISDA [11] use the LDA criterion to calculate linear transforms. In these methods, the between-class graph is fully connected from the perspective of LDA graph [9]. The edge connecting two samples of the same class $c$ has a weight of $1/N - 1/n_c$ ($n_c$ is the size of class $c$), while the weight of the edge connecting samples of different classes is $1/N$.

While the proposed method and the JSLDE method [1] require label and projected data smoothing, the current approach is more rigorous in its smoothing techniques. In [1], the graph enforcing the smoothness only depends on the data, which can result in poor quality if aberrant features or samples are present. Conversely, the proposed method uses a label graph to compute a similarity graph, and the smoothness of the labels or projected data is ensured through the automatic fusion of the data graph and the label graph. The proposed method incorporates self-supervision by using the label graph, taking advantage of the coarse labels.

Table 1: Definition of the main variables used in this article.

| Symbol | Description |
|---|---|
| $\mathbf{W}$ | Linear projection matrix $\in \mathbb{R}^{D\times d}$ |
| $\mathbf{H}_d$, $\mathbf{L}_d$ | Data graph and its Laplacian $\in \mathbb{R}^{N\times N}$ |
| $\mathbf{H}_b$, $\mathbf{L}_b$ | Between-class graph and its Laplacian $\in \mathbb{R}^{N\times N}$ |
| $\mathbf{H}_l$, $\mathbf{L}_l$ | Label graph and its Laplacian $\in \mathbb{R}^{N\times N}$ |
| $\mathbf{X}$ | Samples matrix $\in \mathbb{R}^{D\times N}$ |
| $\mathbf{L}_p$ | Laplacian for projected data smoothness $\in \mathbb{R}^{N\times N}$ |
| $\mathbf{L}_s$ | Laplacian for soft labels smoothness $\in \mathbb{R}^{N\times N}$ |
| $\mathbf{Y}$ | Original label matrix $\in \mathbb{B}^{N\times C}$ |
| $\mathbf{F}$ | Soft label matrices $\in \mathbb{R}^{N\times C}$ |
| $w_d$ , $w_l$ | Weights for soft labels graph |
| $v_d$ , $v_l$ | Weights for projected data graph |

## Proposed approach

Our proposed model is designed to capture the underlying data structure and its concealed clusters by employing two similarity graphs. Simultaneously, it learns a soft label matrix denoted as $\mathbf{F}$ and a linear projection matrix designated as $\mathbf{W}$ using both the available labels and the entire dataset. The first of these graphs, referred to as the "data graph", encodes the degree of similarity between the various features present in the data. In parallel, the "label graph" is constructed to represent the similarities among labels assigned to all samples. During the iterative optimization process, unlabeled samples are assigned predicted soft labels. Subsequently, these two distinct graphs are amalgamated, and an automatic weighting mechanism is applied to enhance the final solution. This involves fine-tuning the influence of both the data and the labels. The computation of soft labels and the linear transformation is executed with the objective of satisfying various critical properties, including achieving class differentiation and ensuring the smoothness of labels and projections.

### *Soft label matrix to between-class graph and label graph*

In this section, we establish a relation connecting a given estimate of the label matrix $\mathbf{F}$ with the between-class graph matrix $\mathbf{H}_b$ as well as to the label graph matrix $\mathbf{H}_l$.

**Between-class graph matrix $\mathbf{H}_b$** is the similarity matrix of the between class graph. In the supervised case and using the $K$ nearest neighbor adjacency graph, the entries of the graph matrix $\mathbf{H}_b$ can be specified as follows:

$$A_b(i,j) = \begin{cases} 1 & \text{if } [\mathbf{x}_i, \mathbf{x}_j \text{ are in different class}] \\ & \text{and} [\mathbf{x}_j \in N_b(\mathbf{x}_i) \text{ or } \mathbf{x}_i \in N_b(\mathbf{x}_j)] \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

where $N_b(\mathbf{x}_i)$ is a list of neighbors of $\mathbf{x}_i$ with different labels.

In our work, many data samples have no class information. A label matrix $\mathbf{F}$ is used, where each row $\mathbf{F}_{i*}$ represents the class membership of the data sample $\mathbf{x}_i$. The inner product $\mathbf{F}_{i*} \cdot \mathbf{F}_{j*}$ is 1 if the two samples $\mathbf{x}_i$ and $\mathbf{x}_j$ have the same label, and 0 if they do not. Thus, the between-class graph similarity matrix $\mathbf{H}_b$ can be derived from the soft label matrix by the relation $A_b(i,j) = 1 - \mathbf{F}_{i*} \cdot \mathbf{F}_{j*}$, which can also be written in matrix form as:

$$\mathbf{H}_b = \mathbf{E} - \mathbf{F}\mathbf{F}^T \quad (2)$$

where $\mathbf{E} \in \mathbb{R}^{N\times N}$ is a matrix whose entries are equal to one.

**Label graph matrix $\mathbf{H}_l$** is a symmetric similarity matrix representing the degree of similarity between labels. Integrating label and data information into a useful graph is the biggest challenge in making label information as important and influential as data information. The entries of this matrix $\mathbf{H}_l(i,j)$ are set to the Pearson correlation coefficient in the range [-1, +1], which encodes a normalized similarity value for a given pair of soft labels $\mathbf{F}_{i*}$ and $\mathbf{F}_{j*}$. The obtained matrix then becomes non-negative and sparse by keeping the $K$ highest values for each row.

### *Latent space with local margins' maximization*

The desired linear transform can be achieved by maximizing the local margins in the projection space between heterogeneous samples. This is achieved by maximizing the following criterion:

$$\max_{\mathbf{W}} \tfrac{1}{2}\textstyle\sum_i \sum_j \|\mathbf{W}^T\mathbf{x}_i - \mathbf{W}^T\mathbf{x}_j\|^2 A_b(i,j) = \max_{\mathbf{W}} Tr\left(\mathbf{W}^T\mathbf{X}\mathbf{L}_b\mathbf{X}^T\mathbf{W}\right) \quad (3)$$

Where, $\mathbf{L}_b = \mathbf{D}_b - \mathbf{H}_b$ is the Laplacian matrix of the between-class graph, $\mathbf{D}_b$ is a diagonal matrix with elements are the column (or row) sums of the symmetric between-class graph matrix $\mathbf{H}_b$.

When equation (2) is substituted into equation (3), the latter becomes:

$$\begin{aligned} &\max_{\mathbf{W}} Tr\left(\mathbf{W}^T\mathbf{X}(\mathbf{D}_b - (\mathbf{E} - \mathbf{F}\mathbf{F}^T))\mathbf{X}^T\mathbf{W}\right) \Longleftrightarrow \\ &\min_{\mathbf{W}} Tr\left(\mathbf{W}^T\mathbf{X}(\mathbf{E} - \mathbf{F}\mathbf{F}^T)\mathbf{X}^T\mathbf{W}\right) \;\; s.t. \;\; \mathbf{W}^T\mathbf{X}\mathbf{D}_b\mathbf{X}^T\mathbf{W} = \mathbf{I}. \end{aligned} \quad (4)$$

The label matrix $\mathbf{F}$ can also be predicted in the semi-supervised context. The linear transform $\mathbf{W}$ and the label matrix $\mathbf{F}$ can be estimated simultaneously by minimizing the following criterion:

$$g(\mathbf{W},\mathbf{F}) = Tr\left(\mathbf{W}^T\mathbf{X}(\mathbf{E} - \mathbf{F}\mathbf{F}^T)\mathbf{X}^T\mathbf{W}\right) \;\; s.t. \;\; \mathbf{W}^T\mathbf{X}^T\mathbf{D}_b\mathbf{X}\mathbf{W} = \mathbf{I} \text{ and } \mathbf{F} \geq 0 \quad (5)$$

In the proposed criterion (5), the between-class similarity matrix is more general and based on margins associated with all data. Each sample $\mathbf{x}_i$ has two different encodings. The first encoding is the linear projection $\mathbf{W}^T\mathbf{x}_i$, commonly known as the low-dimensional representation. The second encoding is the soft label $\mathbf{F}_{i*}$, which provides a prediction of class membership.

### *Projected data and label smoothness over a fused graph*

To address the issue of high-dimensional data, it is desirable to exploit the structure of the data by imposing smoothing constraints on the associated graphs, leading to a better model (e.g., [4, 13, 15]). This approach aims to simultaneously estimate and recover both labels and linear transforms by combining two forms of smoothing: smoothing of the labels and smoothing of the embedding. The smoothing is applied to a fused graph, which is obtained by integrating the data graph and the label graph (updated during the optimization process), represented by $w_d\mathbf{H}_d + w_l\mathbf{H}_l$, where $w_d$ and $w_l$ are automatically determined weights. The data graph $\mathbf{H}_d$ is calculated once, while the computation of the label graph $\mathbf{H}_l$ is updated using the current estimated $\mathbf{F}$.

The term that enforces the label smoothness is:

$$\begin{aligned} &\tfrac{1}{2}\textstyle\sum_i\sum_j \|\mathbf{F}_{i*} - \mathbf{F}_{j*}\|^2 \left[w_d A_d(i,j) + w_l A_l(i,j)\right] \\ &\quad = w_d \left(\tfrac{1}{2}\textstyle\sum_i\sum_j \|\mathbf{F}_{i*} - \mathbf{F}_{j*}\|^2 A_d(i,j)\right) \\ &\quad + w_l \left(\tfrac{1}{2}\textstyle\sum_i\sum_j \|\mathbf{F}_{i*} - \mathbf{F}_{j*}\|^2 A_l(i,j)\right) \\ &\quad = w_d\, Tr\left(\mathbf{F}^T\mathbf{L}_d\,\mathbf{F}\right) + w_l\, Tr\left(\mathbf{F}^T\mathbf{L}_l\,\mathbf{F}\right). \end{aligned} \quad (6)$$

The projected data smoothness term is given by:

$$\begin{aligned}&\tfrac{1}{2}\textstyle\sum_i\sum_j \|\mathbf{W}^T\mathbf{x}_i-\mathbf{W}^T\mathbf{x}_j\|^2\,[v_d A_d(i,j)+v_l A_l(i,j)]\\&=v_d\,Tr\left(\mathbf{W}^T\mathbf{X}\mathbf{L}_d\mathbf{X}^T\mathbf{W}\right)+v_l\,Tr\left(\mathbf{W}^T\mathbf{X}\mathbf{L}_l\mathbf{X}^T\mathbf{W}\right).\end{aligned}\tag{7}$$

### Proposed Semi-Supervised Model

Finally, a non-negative label matrix $\mathbf{F}$ is forced to contain orthogonal columns. The orthogonality of $\mathbf{F}$ is enforced by minimizing the Frobenius norm $\|\mathbf{F}^T\mathbf{F}-\mathbf{I}\|^2 = Tr\left((\mathbf{F}^T\mathbf{F}-\mathbf{I})^T(\mathbf{F}^T\mathbf{F}-\mathbf{I})\right)$. By including the smoothing constraints equations (6) and (7) and the orthogonality of $\mathbf{F}$, the problem (5) can be rewritten as follows:

$$\begin{aligned}g(\mathbf{W},\mathbf{F})&=Tr\left(\mathbf{W}^T\mathbf{X}(\mathbf{E}-\mathbf{F}\mathbf{F}^T)\mathbf{X}^T\mathbf{W}\right)\\&+\beta\,\left(w_d\,Tr(\mathbf{F}^T\mathbf{L}_d\,\mathbf{F})+w_l\,Tr(\mathbf{F}^T\mathbf{L}_l\,\mathbf{F})\right)\\&+\gamma\left(v_d\,Tr(\mathbf{W}^T\mathbf{X}\mathbf{L}_d\mathbf{X}^T\mathbf{W})+v_l\,Tr(\mathbf{W}^T\mathbf{X}\mathbf{L}_l\mathbf{X}^T\mathbf{W})\right)\\&+\alpha\,Tr\left((\mathbf{F}^T\mathbf{F}-\mathbf{I})^T(\mathbf{F}^T\mathbf{F}-\mathbf{I})\right)\,s.t.\;\mathbf{W}^T\mathbf{X}\mathbf{D}_b\mathbf{X}^T\mathbf{W}=\mathbf{I}\text{ and }\mathbf{F}\geq 0\end{aligned}\tag{8}$$

where $\beta$ and $\gamma$ are two balance parameters. The weights $w_d$, $w_l$, $v_d$ and $v_l$ are used to merge the data graph and the label graph. They are set to automatic values. $\alpha$ is a large positive number.

### Auto Weighted Graph Fusion

Since we have two graphs to be fused for two types of smoothness, we have four weights $w_d$, $w_l$, $v_d$, and $v_l$. Thus, $w_d$ and $w_l$ are the weights that fuse the two graphs into a single graph for soft label smoothness $\mathbf{H}_s = w_d\,\mathbf{H}_d + w_l\,\mathbf{H}_l$. Where, $v_d$ and $v_l$ are used to fuse the two graphs for projected data smoothness $\mathbf{H}_p = v_d\,\mathbf{H}_d + v_l\,\mathbf{H}_l$. During optimization, the label similarity matrix $\mathbf{H}_l$ and the four weights change, making the two graph matrices $\mathbf{H}_s$ and $\mathbf{H}_p$ hybrid and adaptive, which are used to derive the final desired solution.

Similar to methods that use automatic weighting (e.g., [5, 12]), we use weights corresponding to the second, third, fourth, and fifth terms in the objective function of equation (8), respectively. These adaptive weights are set to:

$$w_d=\frac{1}{2\sqrt{Tr(\mathbf{F}^T\mathbf{L}_d\,\mathbf{F})}}\quad,\quad w_l=\frac{1}{2\sqrt{Tr(\mathbf{F}^T\mathbf{L}_l\,\mathbf{F})}}\tag{9}$$

$$v_d=\frac{1}{2\sqrt{Tr(\mathbf{W}^T\mathbf{X}\mathbf{L}_d\mathbf{X}^T\mathbf{W})}}\;,\;v_l=\frac{1}{2\sqrt{Tr(\mathbf{W}^T\mathbf{X}\mathbf{L}_l\mathbf{X}^T\mathbf{W})}}\tag{10}$$

As it will be seen in the iterative optimization scheme, these weights are not constant, since they depend on $\mathbf{F}$, $\mathbf{W}$, and $\mathbf{L}_l$. We emphasize that due to the use of the automatic weights, the minimization of the following objective function is equivalent to the problem defined in (8):

$$\begin{aligned}e(\mathbf{W},\mathbf{F})&=Tr\left(\mathbf{W}^T\mathbf{X}(\mathbf{E}-\mathbf{F}\mathbf{F}^T)\mathbf{X}^T\mathbf{W}\right)\\&+\beta\left(\sqrt{Tr(\mathbf{F}^T\mathbf{L}_d\,\mathbf{F})}+\sqrt{Tr(\mathbf{F}^T\mathbf{L}_l\,\mathbf{F})}\right)\\&+\gamma\left(\sqrt{Tr(\mathbf{W}^T\mathbf{X}\mathbf{L}_d\mathbf{X}^T\mathbf{W})}+\sqrt{Tr(\mathbf{W}^T\mathbf{X}\mathbf{L}_l\mathbf{X}^T\mathbf{W})}\right)\\&+\alpha\,Tr\left((\mathbf{F}^T\mathbf{F}-\mathbf{I})^T(\mathbf{F}^T\mathbf{F}-\mathbf{I})\right)\;s.t.\;\mathbf{W}^T\mathbf{X}\mathbf{D}_b\mathbf{X}^T\mathbf{W}=\mathbf{I}\text{ and }\mathbf{F}\geq 0\end{aligned}\tag{11}$$

## Optimization

To minimize the objective function given in (8), we use an iterative optimization algorithm.

First, the four weights $w_d$, $w_l$, $v_d$, and $v_l$ are initialized with a constant value (e.g., 0.5). The soft label matrix associated with the unlabeled samples $\mathbf{F}_{\mathscr{U}}$ is initialized randomly or with the constant value $1/C$, where $C$ is the number of classes. The iterative process solves $\mathbf{W}$ and $\mathbf{F}$ by calling the following phases:

**Phase 1**. Update $\mathbf{H}_b$ using equation (2) and updated $\mathbf{H}_l$ using the current estimate of $\mathbf{F}$. Then calculate the corresponding Laplacian matrices $\mathbf{L}_b$ and $\mathbf{L}_l$.

**Phase 2**. $\mathbf{F}$ is fixed, estimate the linear transform matrix $\mathbf{W}$. Here, the objective function (8) becomes:

$$\begin{aligned}g(\mathbf{W})&=Tr\left(\mathbf{W}^T\mathbf{X}(\mathbf{E}-\mathbf{F}\mathbf{F}^T)\mathbf{X}^T\mathbf{W}\right)\\&+\gamma\left(v_d\,Tr(\mathbf{W}^T\mathbf{X}\mathbf{L}_d\mathbf{X}^T\mathbf{W})+v_l\,Tr(\mathbf{W}^T\mathbf{X}\mathbf{L}_l\mathbf{X}^T\mathbf{W})\right)\end{aligned}\tag{12}$$

Let the matrix $\mathbf{Q} = \mathbf{X}(\mathbf{E}-\mathbf{F}\mathbf{F}^T)\mathbf{X}^T+\gamma\mathbf{X}(v_d\,\mathbf{L}_d+v_l\,\mathbf{L}_l)\mathbf{X}^T$. Then, the solution for $\mathbf{W}$ is nothing but the eigenvectors of $(\mathbf{X}\mathbf{D}_b\mathbf{X}^T)^{-1}\mathbf{Q}$ corresponding to the smallest eigenvalues.

**Phase 3**. $\mathbf{W}$ is fixed. The label matrix $\mathbf{F}$ is determined as follows. In this case, the objective function (8) will be:

$$\begin{aligned}g(\mathbf{F})&=Tr\left(\mathbf{W}^T\mathbf{X}(-\mathbf{F}\mathbf{F}^T)\mathbf{X}^T\mathbf{W}\right)+\beta\,Tr(\mathbf{F}^T\mathbf{L}_s\mathbf{F})\\&+\alpha\,Tr\left((\mathbf{F}^T\mathbf{F}-\mathbf{I})^T(\mathbf{F}^T\mathbf{F}-\mathbf{I})\right)s.t.\;\mathbf{F}\geq 0\end{aligned}\tag{13}$$

where $\mathbf{L}_s = w_d\,\mathbf{L}_d + w_l\,\mathbf{L}_l$. The above function can be minimized by using one step of a special gradient descent scheme. With respect to the matrix $\mathbf{F}$, the gradient matrix of the function $g$ is given by:

$$\begin{aligned}\tfrac{\partial g}{\partial\mathbf{F}}&=2\,[-\mathbf{X}^T\mathbf{W}\mathbf{W}^T\mathbf{X}\mathbf{F}+\beta\,\mathbf{L}_s\mathbf{F}+2\,\alpha\,\mathbf{F}\,(\mathbf{F}^T\mathbf{F}-\mathbf{I})]\\&=2\,[-\mathbf{Z}\mathbf{F}+\beta\,\mathbf{L}_s\mathbf{F}+2\,\alpha\,\mathbf{F}\,(\mathbf{F}^T\mathbf{F}-\mathbf{I})].\end{aligned}\tag{14}$$

where, $\mathbf{Z}=\mathbf{X}^T\mathbf{W}\mathbf{W}^T\mathbf{X}$. The matrices $\mathbf{Z}$ and $\mathbf{L}_s$ can be set to the differences between two non-negative matrices: $\mathbf{Z}=\mathbf{Z}^+-\mathbf{Z}^-$, and $\mathbf{L}_s = \mathbf{L}^+-\mathbf{L}^-$ with $\mathbf{Z}^+=\frac{1}{2}(|\mathbf{Z}|+\mathbf{Z})$ and $\mathbf{Z}^-=\frac{1}{2}(|\mathbf{Z}|-\mathbf{Z})$. $\mathbf{L}^+$ and $\mathbf{L}^-$ have similar expressions.

Let $\mathbf{P}^+=\mathbf{Z}^-\mathbf{F}+\beta\,\mathbf{L}^+\mathbf{F}+2\,\alpha\,\mathbf{F}\mathbf{F}^T\mathbf{F}$ and $\mathbf{P}^-=\mathbf{Z}^+\mathbf{F}+\beta\,\mathbf{L}^-\mathbf{F}+2\alpha\mathbf{F}$. Therefore, the gradient of $g$ with respect to the element $F_{ij}$ is given by:

$$\tfrac{\partial g}{\partial F_{ij}}=[\tfrac{\partial g}{\partial\mathbf{F}}]_{ij}=[2\,(\mathbf{P}^+-\mathbf{P}^-)]_{ij}=2\,(P^+_{ij}-P^-_{ij}).\tag{15}$$

The update rule that allows the minimization of Eq. (13) becomes:

$$F^{t+1}_{ij}\longleftarrow F^t_{ij}-\lambda_{ij}\,\tfrac{\partial g}{\partial F_{ij}}.\tag{16}$$

To ensure a non-negative result, let learning rate (or gradient step) $\lambda_{ij}=\frac{F^t_{ij}}{2P^+_{ij}}$. Then updating rule of (16), becomes:

$$F^{t+1}_{ij}\longleftarrow F^t_{ij}\,\tfrac{P^-_{ij}}{P^+_{ij}}\qquad j=1,...,C.\;\;i=l+1,....,N.\tag{17}$$

After updating, the matrix $\mathbf{F}_{\mathscr{U}}$ is normalized so that its $\ell_1$ norm is equal to one, i.e., $|\mathbf{F}_{\mathscr{U}}|_1=1$. The resulting soft label matrix is $\mathbf{F}=\begin{pmatrix}\mathbf{F}_{\mathscr{L}}\\ \mathbf{F}_{\mathscr{U}}\end{pmatrix}$, where $\mathbf{F}_{\mathscr{L}}$ is equal to $\mathbf{Y}_{\mathscr{L}}$.

**Phase 4**. Fix both $\mathbf{W}$ and $\mathbf{F}$. Then we estimate the four weights using the equations (9), and (10).

Algorithm 1 shows the main steps of the proposed semi-supervised learning algorithm. Once the model is learned, we can map any training or test data sample to the projection subspace by doing the following:

$$\mathbf{y}=\mathbf{W}^T\mathbf{x}.\tag{18}$$

**Algorithm 1** Data and Label Graph Fusion For SSL (DLGF)

**Input:**
- Data matrix $\mathbf{X} \in \mathbb{R}^{D\times N}$, and its label matrix $\mathbf{Y} \in \mathbb{B}^{N\times C}$.
- Parameters $\alpha$, $\beta$, and $\gamma$.

**Output:**
The label matrix $\mathbf{F}$ and the linear transform matrix $\mathbf{W}$.

**Process:**
1- Initialize $\mathbf{F} \in \mathbb{R}^{N\times C}$ as explained in the optimization section.
2- Estimate $\mathbf{H}_d$ and its Laplacian $\mathbf{L}_d$. Use the KNN graph.
3- Initialize the weights $w_d$, $w_l$, $v_d$, and $v_l$ to 1/2.
**Repeat**

- Estimate $\mathbf{H}_b$, then $\mathbf{H}_l$ and its Laplacian $\mathbf{L}_l$.
- Estimate $\mathbf{W}$ using the smallest eigenvalues of Eq. (12).
- Create $\mathbf{Z} = \mathbf{X}^T\mathbf{W}\mathbf{W}^T\mathbf{X}$ and calculate $\mathbf{F}_{\mathscr{U}}$ using Eq. (17).
- Normalize $\mathbf{F}_{\mathscr{U}}$ such that $\|\mathbf{F}_{\mathscr{U}}\|_1 = 1$.
- Using the Eqs. (9), and (10), we update $w_d$, $w_l$, $v_d$, and $v_l$

**Until** *Convergence*
4- Discrete the estimated $\mathbf{F}$, update $\mathbf{H}_b$ and $\mathbf{D}_b$. Then recalculate $\mathbf{W}$.

# Performance Study

## Experimental setup

We use the following five image datasets: **Extended Yale**[1] (1774 images, 28 classes), **PIE**[2] (1926 images, 68 classes), **UMIST**[3] (575 images, 20 classes), **MNIST**[4]: (60,000 images, 10 classes, Resnet50 features), **Caltech101**[5] (9000 images, 101 classes).

We test the performance of our method with several semi-supervised approaches. All of these techniques, including ours, used the same configurations and conditions in Table 2 (i.e., databases, number of labeled images, dimensionality reduction, and parameter values) in the experiments.

Table 2: Setup and conditions used for the experiments

| | |
|---|---|
| **Parameters values** | $\{10^{-9}, 10^{-6}, 10^{-3}, 10^{0}, 10^{3}, 10^{6}, 10^{9}\}$ |
| **Feature reduction** | PCA retaining 95% of the data variance |
| **Dataset splits** | 50 % for training, and 50 % for testing. |
| $p$ **(# of labeled images per class)** | $\{1,2,3\}$ in Table 3, and $\{5,10,20,30\}$ in Table 4 |

We randomly splits the dataset into a training and a test part. In other words, we use ten random splits for evaluation.

## Experimental Results and Method Comparison

In this section, we present the results of our experiments designed to assess the effectiveness and performance of our proposed approach in comparison to five other semi-supervised methods. We conducted each method ten times using different random combinations of training and test samples and recorded the mean classification accuracy to ensure a fair comparison. The Table 3 displays the best average classification rates achieved on the test data. For classification in the low-dimensional space generated by the projection methods, we employed the Nearest Neighbor Classifier, with the highest accuracy highlighted in bold for emphasis.

[1] *www.vision.ucsd.edu/∼ leekc/ExtYaleDatabase/ExtYaleB.html*
[2] *www.ri.cmu.edu/projects/project_418.html*
[3] *https://www.sheffield.ac.uk/eee/research/iel/research/face*
[4] http://yann.lecun.com/exdb/mnist/
[5] *http://www.vision.caltech.edu/ImageDatasets/Caltech*101

## Analysis of results

The results of the experiments in Table 3 show that the proposed method (DLGF) outperforms the other competing methods. DLGF was found to be superior in all 16 cases (5 image datasets, each with different sets of labels). In comparison with the JSLDE method, the proposed DLGF method showed significant improvement, with differences in the rate of 11.7%, 10.3%, and 5.6% for $p = 1,2,3$, respectively. The results also demonstrate that the proposed method performs well even for large datasets (MNIST), as seen in Table 4.

## Visualization study of the graph weights

Figure 1 illustrates The values of the graph coefficients against the number of iterations on the MNIST dataset. (a) displays the weights for smoothing the soft labels, $w_d$ and $w_l$, while (b) shows the coefficients for smoothing the projected data, $v_d$ and $v_l$. These coefficients are dependent on the current solutions for the soft labels and linear transform, and their changes reflect the adaptive process where current solutions are used to update the estimation coefficients. As seen in the graphs for $w_d$ and $w_l$, $w_d$ decreases as optimization progresses after the third iteration, while $w_l$ increases. For the projected data smoothing constraints represented by $v_d$ and $v_l$, they generally decrease over both the data graph and the label graph, but the optimal graph may vary depending on the solution.

# Conclusion

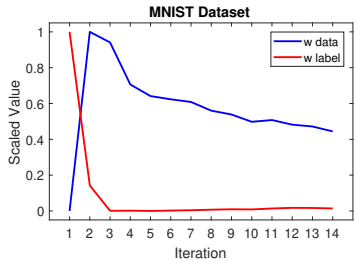
In this paper, we introduce a semi-supervised learning approach that involves the fusion of self-weighted data and label graphs. This method leverages unlabeled data to discern the data's underlying structure, predict soft labels, and employs an adaptive label graph to enhance these predictions. Through an iterative process, the data and label graphs are amalgamated to construct a high-performance graph, which serves as a critical criterion in the learning process. Importantly, this method concurrently estimates the fusion coefficients of the graphs, as well as the linear transform and soft labels. It stands out as an efficient and effective solution, particularly in scenarios where only a limited number of labeled images are available.

Table 3: Best average classification accuracy in (%) (test data).

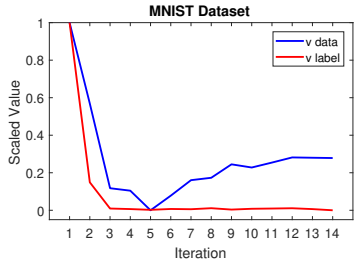
| **Dataset** | $p$ | SLDA[14] | TR-FSDA[6] | ISDA[11] | MVCGL[16] | JSLDE[1] | **DLGF** |
|---|---|---|---|---|---|---|---|
| PIE | 1 | 33.2 | 16.3 | 34.2 | 30.7 | 34.2 | **35.5** |
| | 2 | 46.5 | 24.1 | 51.8 | 45.4 | 52.7 | **54.1** |
| | 3 | 54.6 | 27.6 | 57.6 | 52.6 | 58.3 | **61.4** |
| UMIST | 1 | 37.6 | 38.3 | 38.1 | 42.2 | 44.2 | **55.9** |
| | 2 | 47.8 | 46.7 | 47.5 | 50.1 | 53.3 | **63.6** |
| | 3 | 58.0 | 53.3 | 57.6 | 56.7 | 63.0 | **68.6** |
| EXT-Yale | 1 | 47.9 | 41.8 | 47.9 | 42.9 | 46.4 | **50.8** |
| | 2 | 63.4 | 59.4 | 63.3 | 57.3 | 65.4 | **66.7** |
| | 3 | 67.1 | 63.4 | 67.2 | 64.7 | 68.8 | **70.1** |
| Caltech101 | 1 | 47.8 | 61.3 | 47.5 | 67.3 | 66.7 | **69.5** |
| | 2 | 65.8 | 67.7 | 65.9 | 72.2 | 75.3 | **78.2** |
| | 3 | 73.3 | 71.2 | 73.0 | 78.2 | 78.1 | **80.1** |

Table 4: Best average classification accuracy in (%) on the MNIST dataset (Unlabeled data).

| Data type | $p$ | MLAN | SLDA[14] | ISDA[11] | JSLDE[1] | **DLGF** |
|---|---|---|---|---|---|---|
| Unlabeled | 5 | 74.1 | 64.6 | 63.6 | 81.7 | **89.2** |
| | 10 | 81.8 | 79.6 | 78.4 | 86.0 | **90.1** |
| | 20 | 84.6 | 86.3 | 85.1 | 88.8 | **91.9** |
| | 30 | 86.3 | 90.5 | 89.9 | 91.6 | **93.5** |




(a) soft label smoothness.




(b) projected data smoothness.

Figure 1: Scaled value of the adaptive coefficients (AC) versus the number of iterations on the MNIST dataset.

## References


[1] F. Dornaika, A. Baradaaji, and Y. El Traboulsi. Semi-supervised classification via simultaneous label and discriminant embedding estimation. *Information Sciences*, 546:146–165, 2021.

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

[3] F. He, F. Nie, R. Wang, X. Li, and W. Jia. Fast semisupervised learning with bipartite graph for large-scale data. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1–13, 2019.

[4] C. Hou, F. Nie, X. Li, D. Yi, and Y. Wu. Joint embedding learning and sparse regression: A framework for unsupervised feature selection. *IEEE Transactions on Cybernetics*, 44(6):793–804, 2014.

[5] S. Huang, Z. Kang, I. W. Tsang, and Z. Xu. Auto-weighted multi-view clustering via kernelized graph learning. *Pattern Recognition*, 88:174–184, 2019.

[6] Y. Huang, D. Xu, and F. Nie. Semi-supervised dimension reduction using trace ratio criterion. *IEEE Transactions on neural networks and learning systems*, 23(3):519–526, 2012.

[7] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. *International Conference on Learning Representations*, 2017.

[8] F. Nie, J. Li, and X. Li. Parameter-free auto-weighted multiple graph learning: A framework for multiview clustering and semi-supervised classification. In *International Joint Conference on Artificial Intelligence, IJCAI*, 2016.

[9] F. Nie, S. Xiang, Y. Jia, C. Zhang, and S. Yan. Trace ratio criterion for feature selection. In *Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence*, 2008.

[10] H. Pan and Z. Kang. Robust graph learning for semi-supervised classification. In *International Conference on Intelligent Human-Machine Systems and Cybernetics*, 2018.

[11] S. Ren, X. Gu, P. Yuan, and H. Xu. An iterative paradigm of joint feature extraction and labeling for semi-supervised discriminant analysis. *Neurocomputing*, 273:466–480, 2018.

[12] S. Shi, F. Nie, R. Wang, and X. Li. Auto-weighted multi-view clustering via spectral embedding. *Neurocomputing*, 399:369–379, 2020.

[13] Y. E. Traboulsi and F. Dornaika. Flexible semi-supervised embedding based on adaptive loss regression: Application to image cate-

gorization. *Information Sciences*, 444:1–9, 2018.

[14] S. Wang, J. Lu, X. Gu, H. Du, and J. Yang. Semi-supervised linear discriminant analysis for dimension reduction and classification. *Pattern Recognition*, 57:179–189, 2016.

[15] R. Zhu, F. Dornaika, and Y. Ruichek. Joint graph based embedding and feature weighting for image classification. *Pattern Recognition*, 2019.

[16] N. Ziraki, F. Dornaika, and A. Bosaghzadeh. Multiple-view flexible semi-supervised classification through consistent graph construction and label propagation. *Neural Networks*, 146:174–180, 2022.