Scalable Semi-Supervised Learning by Efficient Anchor Graph Regularization

Meng Wang, *Member, IEEE*, Weijie Fu, Shijie Hao,
Dacheng Tao, *Fellow, IEEE*, and Xindong Wu, *Fellow, IEEE*

Abstract—Many graph-based semi-supervised learning methods for large datasets have been proposed to cope with the rapidly increasing size of data, such as Anchor Graph Regularization (AGR). This model builds a regularization framework by exploring the underlying structure of the whole dataset with both datapoints and anchors. Nevertheless, AGR still has limitations in its two components: (1) in anchor graph construction, the estimation of the local weights between each datapoint and its neighboring anchors could be biased and relatively slow; and (2) in anchor graph regularization, the adjacency matrix that estimates the relationship between datapoints, is not sufficiently effective. In this paper, we develop an Efficient Anchor Graph Regularization (EAGR) by tackling these issues. First, we propose a fast local anchor embedding method, which reformulates the optimization of local weights and obtains an analytical solution. We show that this method better reconstructs datapoints with anchors and speeds up the optimizing process. Second, we propose a new adjacency matrix among anchors by considering the commonly linked datapoints, which leads to a more effective normalized graph Laplacian over anchors. We show that, with the novel local weight estimation and normalized graph Laplacian, EAGR is able to achieve better classification accuracy with much less computational costs. Experimental results on several publicly available datasets demonstrate the effectiveness of our approach.

Index Terms—Semi-supervised learning, anchor graph, local weight estimation

1 Introduction

In many real-world classification tasks, we are usually faced with datasets in which only a small portion of samples are labeled while the rest are unlabeled. A learning mechanism called semi-supervised learning (SSL), which is capable of fully leveraging unlabeled data and labeled data to achieve better classification, is therefore proposed to deal with this situation. In recent years, various semi-supervised learning methods [55] have been developed to adapt different kinds of data, including mixture models [6], co-training [4], semi-supervised support vector machines [18], and graph-based SSL [54]. This learning mechanism is broadly used in many real-world applications such as data mining [30], [34], [35], [36], [52], [53] and multimedia content analysis [14], [15], [24], [25], [49].

In this paper, we focus on the family of graph-based semi-supervised learning methods. These methods are built based on a cluster assumption [51]: nearby points are likely to have the same label. A typical model of these algorithms consists of two main parts: a fitting constraint and a smoothness constraint, both of which have clear geometric meanings. The former means that a good classification function should not change too much from the initial label assignment, while the latter means that this function should have similar semantic labels among nearby points. Based on the above formulation, these algorithms generally produce satisfying classification results in a manifold space [12], [13], [20], [40], [45], [50]. Meanwhile, graph-based SSL can be intuitively explained in a label propagation perspective [54], i.e., the label information from labeled vertices is gradually propagated through graph edges to all unlabeled vertices.

Most traditional graph-based learning methods, however, focus on classification accuracy while ignoring the underlying computational complexity, which is of great importance for the classification of a large dataset, especially given the recent explosive increase in Internet data. The complexity mainly arises from two aspects. The first is the kNN strategy for graph construction, and the second is the inverse calculation of the normalized Laplacian matrix in optimization. Both of them are time-consuming and have a large storage requirement.

To reduce the cubic-time complexity, recent studies seek to speed up the intensive computation of the graph Laplacian manipulation. Anchor graph regularization (AGR) [21], [22], a recently proposed graph-based learning model for large datasets, constructs a novel graph with datapoints and anchors. It reduces the computational cost via subtle matrix factorization by utilizing the intrinsic structure of data distribution. It is exactly in linear time with data size. The anchor graph model has been widely applied to many applications [8], [19], [41], [42], [43] and achieves satisfactory performance.

- *M. Wang, W. Fu, and S. Hao are with the School of Computer and Information Science, Hefei University of Technology, Hefei 230009, China. E-mail: {eric.wangmeng, fwj.edu, hfut.hsj}@gmail.com.*
- *D. Tao is with the Centre for Quantum Computation & Intelligent Systems, and the Faculty of Engineering & Information Technology, University of Technology, Ultimo, NSW 2007, Australia. E-mail: dacheng.tao@uts.edu.au.*
- *X. Wu is with the School of Computer Science and Information Engineering, Hefei University of Technology, Hefei 230009, China, and the Department of Computer Science, University of Vermont, Burlington, VT 05405, USA. E-mail: xwu@cs.uvm.edu.*

Manuscript received 6 Oct. 2015; revised 2 Feb. 2016; accepted 16 Feb. 2016. Date of publication 26 Feb. 2016; date of current version 1 June 2016. Recommended for acceptance by J. Bailey.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TKDE.2016.2535367

There are two key parts in AGR. The first part is anchor graph construction, in which a local weight matrix that measures the relationship between datapoints and anchors needs to be estimated. The second part is anchor graph regularization, in which an adjacency matrix that measures the strength of graph edges needs to be designed. For local weight estimation, a method called Local Anchor Embedding (LAE) is employed in [21] to replace the conventional kernel-defined weight computation. However, we will demonstrate the limitation of the optimization objective of LAE. Moreover, the LAE process is expensive as it involves a gradient descent solver. For adjacency matrix design, Liu et al. [21] introduce a method that constructs the adjacency matrix based on local weight matrix. But actually it has been shown in [21] that the regularization framework can be equivalent to a regularization on anchors with a "reduced" graph Laplacian. Therefore, in this work, we propose a novel approach named Efficient Anchor Graph Regularization (EAGR) with a novel local weight estimation method and a more effective normalized graph Laplacian over anchors. In comparison with AGR, EAGR obtains comparable or better accuracy in a shorter time in SSL based classification tasks.

The main contributions of our work are as follows.

(1) We introduce a novel graph-based SSL approach that is able to deal with large datasets. By improving the conventional AGR method, we show that the proposed EAGR is able to achieve better classification accuracy with even much less implementation time. The EAGR also empirically shows its advantages over many existing SSL methods developed for large datasets, such as the methods in [26], [47], [48].
(2) We point out the limitation of the conventional local anchor embedding method and propose a novel approach for local weight estimation. We reformulate the objective function of LAE by replacing the inequality constraint with an absolute operation and obtain an efficient and effective analytical solution. In addition, we incorporate the locality constraints into the objective function to further improve the performance.
(3) Instead of designing an adjacency matrix for all datapoints, we directly compute an adjacency matrix for anchors by exploring their commonly connected datapoints. We show that the derived normalized graph Laplacian over anchors is more effective than the "reduced" graph Laplacian in [21]. Graph-based learning is performed with the corresponding regularization on anchors.

The rest of this paper is organized as follows. In Section 2, we survey related work. In Section 3, we briefly review the AGR algorithm and conduct an in-depth analysis of its limitations. The proposed approach EAGR is described in Section 4. In Section 5, we conduct experiments on several publicly available datasets to validate our model. Finally, we conclude in Section 6.

2 Related Work

In this section, we focus on the related work of graph-based semi-supervised learning. Once we have constructed a graph for all datapoints, the labels for classification can be propagated from limited labeled data to remaining unlabeled data [17].

For many years, researchers have focused on improving the classification accuracy of graph-based SSL via designing more appropriate label propagation models with simple kNN graph. Zhu et al. [54] advocated the formulation of the learning problem based on Gaussian random field and gave intuitive interpretations for their model. Belkin et al. [2] proposed a classification function which is defined only on the sub-manifold rather than the whole ambient space. Zhou et al. [51] subsequently suggested the design of a classification model which is sufficiently smooth with respect to the intrinsic structure collectively revealed by the known labeled and unlabeled points. There are additionally many works that focus on graph construction to improve classification accuracy. For instance, Zelnik et al. [46] first stated that it would be helpful to consider a local scale in computing the affinity between each pair of points for the edge. Wang et al. [37] developed a graph-based SSL approach based on a linear neighborhood model which assumes that each datapoint can be linearly reconstructed from its neighborhood. Similarly, Tian et al. [33] proposed learning a nonnegative low-rank graph to capture global linear neighborhoods. Although these methods show promising performance in various applications, they are not sufficiently scalable in terms of computing and storage costs, which imposes limitations in handling large datasets.

With the rapid increase in data size, researchers have paid more attention to designing novel approaches to reduce the computational cost of graph-based learning. Wang et al. [38] proposed a multiple random divide-and-conquer approach to construct an approximate neighborhood graph and presented a neighborhood propagation scheme to further enhance the accuracy. Huang et al. [16] proposed a novel label propagation algorithm in which the label information is first propagated from labeled instances to unlabeled instances, and then labels spread among the unlabeled instances until a steady state is reached. These algorithms simplify either the graph construction or the label propagation, and so the computational cost is reduced to some extent. Additionally, hashing strategies [7], [32], [29] can also be applied to facilitate large-scale classification.

Recently, Liu et al. [21] proposed an anchor graph regularization approach. The graph is constructed with datapoints and anchors, which simultaneously reduces computational cost and storage cost. As a result, the anchor graph model has been applied to clustering [5], [44], hashing [23], manifold ranking [43], multi-graph learning [9], and tracking [41]. For example, Yang et al. [44] proposed a low-rank learning method to improve the clustering performance for large-scale manifold data by a two-step bipartite random walk through cluster nodes. Cai et al. [5] proposed an efficient computation of the spectral embedding of data with an anchor-based representation to improve spectral clustering. Liu et al. [23] proposed an anchor graph based hashing method to learn appropriate compact codes with a feasible computational cost. Xu et al. [43] designed a new adjacency matrix with the anchor graph to speed up manifold ranking for image retrieval. Wu et al. [41] presented a local landmark approximation (LLA) model, which iteratively solves its target function based on gradient projection. The model is

applied to visual tracking and achieves state-of-the-art performance. In spite of these, AGR still has some limitations, which are analyzed and addressed in the following sections.

3 Anchor Graph Regularization

In this section, we first review the anchor graph regularization model and then give a detailed analysis of its limitations.

3.1 Anchor Graph Regularization Formulation

Given a dataset $\mathcal{D} = \{(\mathbf{x}_1, \mathbf{y}_1), (\mathbf{x}_2, \mathbf{y}_2), \ldots, (\mathbf{x}_l, \mathbf{y}_l), \mathbf{x}_{l+1}, \ldots, \mathbf{x}_n\}$ with n samples in d dimensions, we can obtain a set of representative anchors $\mathbf{U} = \{\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_m\} \in \mathbb{R}^{d\times m}$. Typically, these anchors are selected by a clustering method such as K-means. These anchors clearly share the same feature space with the original datapoints. Let $f : \mathbf{x} \rightarrow \mathbf{R}$ be a real value function which assigns each point a label from c distinct classes. Then, once we obtain a local weight matrix $\mathbf{Z}$ that measures the potential relationships between datapoints and anchors, we can estimate $f(\mathbf{x})$ for each datapoint as a weighted average of the labels of the anchor set

$$f(\mathbf{x}_i) = \sum_{k=1}^{m} Z_{ik} f(\mathbf{u}_k), \tag{1}$$

with the constraints $\sum_{k=1}^{m} Z_{ik} = 1$ and $Z_{ik} \geq 0$. Note that the element Z_{ik} represents the local weight between datapoint $\mathbf{x}_i$ and anchor $\mathbf{u}_k$.

Kernel-defined weights are usually sensitive to hyperparamters and lack a meaningful interpretation. To measure the local weights more robustly, one can adopt a LLE[31] like objective function with a clear geometric meaning:

$$\begin{aligned} &\operatorname{argmin}_{\mathbf{z}_i} \|\mathbf{x}_i - \mathbf{U}_{\langle i\rangle}\mathbf{z}_{\langle i\rangle}\|^2 \\ &\text{s.t.} \mathbf{1}^{\mathrm{T}}\mathbf{z}_{\langle i\rangle} = 1, \mathbf{z}_{\langle i\rangle} \succeq 0, \end{aligned} \tag{2}$$

where $\langle i\rangle$ is a index set of s closest anchors of $\mathbf{x}_i$. A standard quadratic problem (QP) solver can be used to solve Eq. (2). To achieve a faster optimization, Liu et al. suggested a new algorithm named local anchor embedding, which employs Nesterovs method [28] to accelerate the gradient descent. More details can be found in [21].

Based on the local weights $\mathbf{Z} = \{\mathbf{z}_1, \ldots, \mathbf{z}_n\}^{\mathrm{T}} \in \mathbb{R}^{n\times m}$, the adjacency matrix between datapoints can be designed as

$$\mathbf{W} = \mathbf{Z}\Lambda^{-1}\mathbf{Z}^{\mathrm{T}}, \tag{3}$$

where the diagonal matrix Λ is defined as $\Lambda_{kk} = \sum_{i=1}^{n} Z_{ik}$, $k = 1, \ldots, m$. From Eq. (3), we can see that if two points are correlative $(W_{ij} > 0)$, they share at least one common anchor, and otherwise $W_{ij} = 0$. It is likely that datapoints sharing common anchors have similar semantic concepts.

Let $\mathbf{Y}_l = [\mathbf{y}_1{}^{\mathrm{T}}, \mathbf{y}_2{}^{\mathrm{T}}, \ldots, \mathbf{y}_l{}^{\mathrm{T}}]^{\mathrm{T}} \in \mathbb{R}^{l\times c}$ denote a class indicator matrix on labeled samples, with $Y_{ij} = 1$ if $\mathbf{x}_i$ belongs to class j and $Y_{ij} = 0$ otherwise. Let $\mathbf{A} = [\mathbf{a}_1{}^{\mathrm{T}}, \mathbf{a}_2{}^{\mathrm{T}}, \ldots, \mathbf{a}_m{}^{\mathrm{T}}]^{\mathrm{T}} \in \mathbb{R}^{m\times c}$ denote the prediction label matrix on the anchor set. Anchor graph regularization can be naturally formulated to deal with the standard multi-class SSL problem as follows:

$$\begin{aligned} Q(\mathbf{A}) &= \sum_{i=1}^{l} \|\mathbf{z}_i^{\mathrm{T}}\mathbf{A} - \mathbf{y}_i\|^2 + \frac{\gamma}{2}\sum_{i,j=1}^{n} W_{ij}\|\mathbf{z}_i^{\mathrm{T}}\mathbf{A} - \mathbf{z}_j^{\mathrm{T}}\mathbf{A}\|^2 \\ &= \|\mathbf{Z}_l\mathbf{A} - \mathbf{Y}_l\|_{\mathrm{F}}^2 + \gamma \mathrm{tr}(\mathbf{A}^{\mathrm{T}}\mathbf{Z}^{\mathrm{T}}(\mathbf{I} - \mathbf{W})\mathbf{Z}\mathbf{A}) \\ &= \|\mathbf{Z}_l\mathbf{A} - \mathbf{Y}_l\|_{\mathrm{F}}^2 + \gamma \mathrm{tr}(\mathbf{A}^{\mathrm{T}}\tilde{\mathbf{L}}\mathbf{A}), \end{aligned} \tag{4}$$

where $\tilde{\mathbf{L}} = \mathbf{Z}^{\mathrm{T}}(\mathbf{I} - \mathbf{W})\mathbf{Z} \in \mathbb{R}^{m\times m}$ is the reduced Laplacian, $\mathbf{Z}_l \in \mathbb{R}^{l\times m}$ is the sub-matrix according to the labeled part of local weight matrix $\mathbf{Z}$, and $\gamma > 0$ is a trade-off parameter.

From the above equation, we can see that, although AGR is performed with a regularization on all datapoints, it is equivalent to a regularization on anchors with a graph Laplacian $\tilde{\mathbf{L}}$. This is understandable, as the labels of other datapoints are actually inferred from anchors.

Then, the optimal $\mathbf{A}$ can be computed as follows:

$$\mathbf{A} = (\mathbf{Z}_l^{\mathrm{T}}\mathbf{Z}_l + \gamma\tilde{\mathbf{L}})^{-1}\mathbf{Z}_l^{\mathrm{T}}\mathbf{Y}_l. \tag{5}$$

Finally, we can employ the solved labels associated with anchors to predict the hard label for unlabeled samples as

$$\widehat{y}_i = \operatorname{argmax}_{j\in\{1,\ldots,c\}} \frac{\mathbf{Z}_{i\cdot} \times \mathbf{A}_{\cdot j}}{\lambda_j}, i = l+1, \ldots, n, \tag{6}$$

where $\mathbf{Z}_{i\cdot} \in \mathbb{R}^{1\times m}$ denotes the ith row of $\mathbf{Z}$, $\mathbf{A}_{\cdot j} \in \mathbb{R}^{m\times 1}$ is the jth column of $\mathbf{A}$, and the normalization factor $\lambda_j = \mathbf{1}^{\mathrm{T}}\mathbf{Z}\mathbf{A}_{\cdot j}$, suggested as a useful normalization strategy in [54], balances skewed class distributions.

3.2 Analysis of Anchor Graph Regularization

The AGR model has been widely used in many applications for its capability in dealing with relatively large datasets. However, the approach has limitations in the local weight estimation and adjacency matrix design, which are analyzed below.

The first limitation comes from the LAE method for local weight estimation. We demonstrate this fact by a toy example. Fig. 1 illustrates a toy example in a 3D space, in which the LAE method attempts to reconstruct the datapoint $\mathbf{x}_1$ by anchors $\mathbf{u}_1$, $\mathbf{u}_2$, $\mathbf{u}_3$ and minimizes $\|\mathbf{x}_1' - \mathbf{x}_1\|_2$, where $\mathbf{x}_1'$ is a reconstructed datapoint. Usually, to follow the shift-invariant and nonnegative weight requirements, we introduce two constraints $\mathbf{1}^{\mathrm{T}}\mathbf{z}_{\langle i\rangle} = 1$ and $\mathbf{z}_{\langle i\rangle} \succeq 0$ into the geometry reconstruction problem, as in Eq. (2). We put $\mathbf{x}_1$ at the origin of the coordinate according to the shift-invariant constraint. Under these constraints, all feasible reconstructed datapoints are limited in the region enclosed by these anchors, such as simplex $\boldsymbol{\alpha}$, as shown in Fig. 1a. It means that for $\mathbf{x}_1$, the value of $\|\mathbf{x}_1' - \mathbf{x}_1\|_2$ is at least the distance from $\mathbf{x}_1$ to $\boldsymbol{\alpha}$, i.e., the length of the blue line segment. Therefore, the best reconstructed point $\mathbf{x}_1^*$, following Eq. (2), is the crossover point as shown in this figure. This point $\mathbf{x}_1^*$ is on the boundary of the closed region $\boldsymbol{\alpha}$ and is linearly reconstructed by merely $\mathbf{u}_1$ and $\mathbf{u}_2$, e.g., $0.5 \times \mathbf{u}_1 + 0.5 \times \mathbf{u}_2$. This is to say, the local weight between $\mathbf{x}_1$ and $\mathbf{u}_3$ is zero. In addition, if we set up a plane along the nearest boundary and make it perpendicular to $\boldsymbol{\alpha}$, namely $\boldsymbol{\beta}$, and then change the positions of these anchors like Fig. 1b, we can see that this zero weight will not change as long as $\mathbf{u}_3$ and $\mathbf{x}_1$ are at different sides of the plane $\boldsymbol{\beta}$, even if $\mathbf{u}_3$ is closer to $\mathbf{x}_1$ than $\mathbf{u}_1$ and $\mathbf{u}_2$, because the best reconstructed

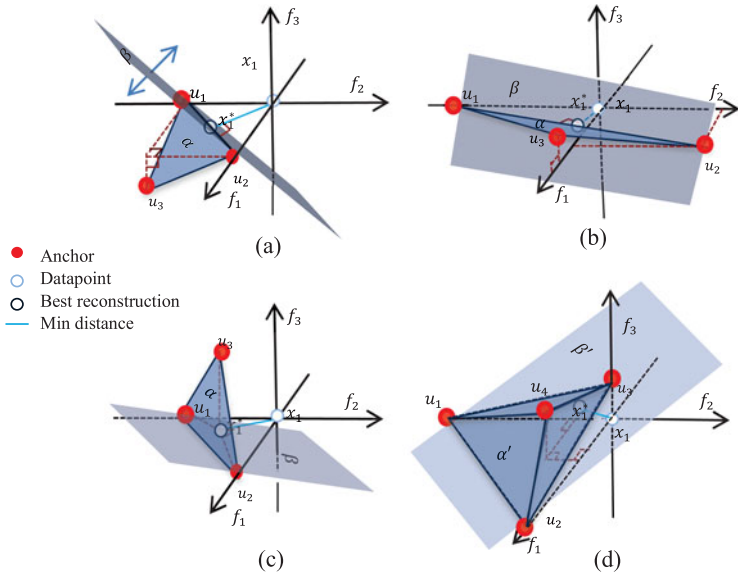


Fig. 1. 3-D toy example for datapoint reconstruction with anchors.

point $\mathbf{x}_1^*$ is still on the boundary. As shown in Fig. 1c, only when $\mathbf{u}_3$ and $\mathbf{x}_1$ are at the same side of $\boldsymbol{\beta}$, $\mathbf{x}_1^*$ will move inside the simplex $\boldsymbol{\alpha}$, and the previous zero weight can change to a positive value. We have discussed the situation above where $s = 3$. However, this geometric interpretation can also be easily extended to the case where $s > 3$. From Fig. 1d, we can see the difference is that the region enclosed by anchors now becomes a closed space, i.e., $\boldsymbol{\alpha}'$, and $\boldsymbol{\beta}'$ now lies along the boundary simplex of $\boldsymbol{\alpha}'$ which is closest to $\mathbf{x}_1$. Similarly, from this figure, we can see that the local weight between $\mathbf{x}_1$ and the anchor, which is on the opposite space of $\boldsymbol{\beta}'$, namely $\mathbf{u}_4$, is always zero. In addition to the above problem, computational cost can also be a disadvantage of LAE, despite several strategies have been applied in [21] to speed up the process.

The second limitation is the adjacency matrix design. After describing the local connection between datapoints and their neighboring anchors, we consider building the adjacency relationship in the whole data space for graph regularization. For instance, we obtain a part of local weights as listed in Fig. 2a. To view these values graphically, we also use a toy example in Fig. 2b to show the relationship between these points, where the length of the edge represents the Euclidean distance between the datapoints and anchors. Now, if we calculate the adjacency weight according to Eq. (3) with these local weights, we have $W_{12} = \sum_{k=1}^{m} \frac{Z_{1k}Z_{2k}}{\Lambda_{kk}} = \frac{0.3\times0.4}{\Lambda_{11}} + \frac{0.4\times0.3}{\Lambda_{22}} + \frac{0.3\times0.3}{\Lambda_{33}}$ and $W_{34} = \sum_{k=1}^{m} \frac{Z_{3k}Z_{4k}}{\Lambda_{kk}} = \frac{0.1\times0.1}{\Lambda_{11}} + \frac{0.1\times0.1}{\Lambda_{22}} + \frac{0.8\times0.8}{\Lambda_{33}}$. Further, if we suppose that Λ_{kk} is nearly the same in homogeneous regions, such as in Fig. 2b, we can obtain $W_{12} = \frac{0.33}{\Lambda_{kk}}$ and $W_{34} = \frac{0.66}{\Lambda_{kk}}$. In this context, the adjacency weights of $\mathbf{x}_1\&\mathbf{x}_2$ and $\mathbf{x}_3\&\mathbf{x}_4$ can be numerically quite different, although the Euclidean distances between them are nearly the same. Therefore, this issue is likely to introduce mistakes in the remaining steps of graph-based SSL tasks.

4 Efficient Anchor Graph Regularization

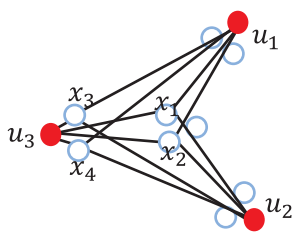
To address the above issues in AGR, we accordingly propose two improvements. First, we introduce a fast local anchor embedding method in anchor graph construction, which reformulates the local weight estimation problem to better measure $\mathbf{Z}$ and speeds up optimization. Second, we directly design a normalized graph Laplacian over anchors and show that it is more effective than the reduced graph Laplacian in AGR. More details are given in the following.

4.1 Fast Local Anchor Embedding (FLAE)

Apart from LAE described above, there exist other similar methods for local weight estimation like LLE [31], LLC [39], and LLA [41]. However, these methods either do not enforce weights to be non-negative or impose the non-negative constraint into objective function via inequality. In the former cases, non-negative similarity measures cannot be guaranteed. In the latter cases, limitation still exists when datapoints are outside of the convex envelope of anchors, according to the analysis in Section 3.2.

$\mathbf{Z}$	$\mathbf{u}_1$	$\mathbf{u}_2$	$\mathbf{u}_3$
$\mathbf{x}_1$	0.4	0.3	0.3
$\mathbf{x}_2$	0.3	0.4	0.3
$\mathbf{x}_3$	0.1	0.1	0.8
$\mathbf{x}_4$	0.1	0.1	0.8

(a) a part of local weight matrix



(b) local edges for these points

Fig. 2. Example of the adjacency modeling.

Therefore, here we aim to design a better local weight matrix $\mathbf{Z}$ to tackle these problems. We call our new weight estimation method Fast Local Anchor Embedding as we will demonstrate that the optimization problem has an analytical solution and can be implemented fast. It is worth mentioning here that, although we use a close name, the formulation of FLAE and its solution are actually quite different from the conventional LAE method. In fact, we have made two changes.

Change 1. Instead of the inequality constraint, we use absolute constraint in geometric reconstruction. Since the non-negative property in similar measurement is important to guarantee the global optimum of graph-based SSL [21], inequality constraint has been employed in most local weight estimation methods, e.g., LAE and LLA. However, as illustrated in Section 3.2, it would introduce additional mistakes when the datapoint is outside of convex envelope of anchors. Therefore, we integrate the non-negative property from another perspective: absolute operator. To this end, we set $\mathbf{z}_{\langle i\rangle} = |\mathbf{c}_{\langle i\rangle}|$ and the constraint $\mathbf{1}^{\mathrm{T}}|\mathbf{c}_{\langle i\rangle}| = 1$ is imposed to follow the shift-invariant requirement in our geometric reconstruction problem. Then, we can obtain the local weight vector $\mathbf{z}_{\langle i\rangle}$ for each datapoint $\mathbf{x}_i$, corresponding to the following problem:

$$\underset{\mathbf{c}_i}{\operatorname{argmin}} \|\mathbf{x}_i - \mathbf{U}_{\langle i\rangle}\mathbf{c}_{\langle i\rangle}\|^2 \tag{7}$$

$$\text{s.t.} \mathbf{1}^{\mathrm{T}}|\mathbf{c}_{\langle i\rangle}| = 1.$$

Compared with LAE, Eq. (7) can be a more direct model to handle the non-negative property in similarity measure. However, since there lacks a straightforward solution of the optimization problem, here we obtain a solution via shrinking the domain of the above problem.

Specifically, we first drop the absolute constraint in our model, which reduces the problem to a simple coding problem as:

$$\underset{\hat{\mathbf{c}}_{\langle i\rangle}}{\operatorname{argmin}} \|\mathbf{x}_i - \mathbf{U}_{\langle i\rangle}\hat{\mathbf{c}}_{\langle i\rangle}\|^2 \tag{8}$$

$$\text{s.t.} \mathbf{1}^{\mathrm{T}}\hat{\mathbf{c}}_{\langle i\rangle} = 1.$$

And the solution can be derived analytically by:

$$\tilde{\mathbf{c}}_{\langle i\rangle} = {\mathbf{C}_i}^{-1}\mathbf{1}, \tag{9}$$

$$\hat{\mathbf{c}}_{\langle i\rangle} = \tilde{\mathbf{c}}_{\langle i\rangle}/\mathbf{1}^{\mathrm{T}}\tilde{\mathbf{c}}_{\langle i\rangle}, \tag{10}$$

where $\mathbf{C}_i = (\mathbf{U}_{\langle i\rangle}^{\mathrm{T}} - \mathbf{1}{\mathbf{x}_i}^{\mathrm{T}})(\mathbf{U}_{\langle i\rangle}^{\mathrm{T}} - \mathbf{1}{\mathbf{x}_i}^{\mathrm{T}})^{\mathrm{T}} \in \mathbb{R}^{s\times s}$ is a data covariance matrix.

Then, we compute our local weight vector $\mathbf{z}_{\langle i\rangle}$ after obtaining the code $\hat{\mathbf{c}}_{\langle i\rangle}$ as follows:

$$\rho = \mathbf{1}^{\mathrm{T}}|\hat{\mathbf{c}}_{\langle i\rangle}| \tag{11}$$

$$\mathbf{c}_{\langle i\rangle} = \hat{\mathbf{c}}_{\langle i\rangle}/\rho, \tag{12}$$

$$\mathbf{z}_{\langle i\rangle} = |\mathbf{c}_{\langle i\rangle}|. \tag{13}$$

As we can see, the above solution $\mathbf{c}_{\langle i\rangle}$ satisfies the constraint $\mathbf{1}^{\mathrm{T}}|\mathbf{c}_{\langle i\rangle}| = 1$, it means this $\mathbf{c}_{\langle i\rangle}$ is a feasible solution of Eq. (7). Meanwhile, we can have the following conclusion.

Proposition 1. *The minimum of Eq. (8) will not greater than the minimum of Eq. (2).*

We leave the proof of the proposition to appendix.

Then, our task is to demonstrate that, with the above solution, the value of the objective function in Eq. (7) is not greater than the value of Eq. (8) with its optimal solution $\hat{\mathbf{c}}_{\langle i\rangle}$. If this conclusion can be drawn, it means that our method can lead to a smaller reconstruction error than LAE, and the effectiveness of our approach can be validated. The details are as follows.

Recall that, by solving Eq. (8), we obtain the optimal solution $\hat{\mathbf{c}}_{\langle i\rangle}$. Clearly, there are two possible cases regarding the obtained codes $\hat{c}_{ij} \in \hat{\mathbf{c}}_{\langle i\rangle}$: (1) $\forall\, \hat{c}_{ij} \geq 0$; and (2) $\exists\, \hat{c}_{ij} < 0$. Then, we follow Eqs. (11), (12) to yield codes $\mathbf{c}_{\langle i\rangle}$.

For the first case, we obtain $\mathbf{c}_{\langle i\rangle} = \hat{\mathbf{c}}_{\langle i\rangle}$. It means that the value of the objective function in Eq. (7) with our feasible solution is the same with the value of Eq. (8) with its optimal solution.

We then mainly focus on the second case. Since the obtained code $\hat{\mathbf{c}}_{\langle i\rangle}$ satisfies the constraint $\mathbf{1}^{\mathrm{T}}\hat{\mathbf{c}}_{\langle i\rangle} = 1$, we first substitute it into the objective function in Eq. (8) to yield the minimum of Eq. (8) as

$$\|\mathbf{x}_{\langle i\rangle}\mathbf{1}^{\mathrm{T}}\hat{\mathbf{c}}_{\langle i\rangle} - \mathbf{U}_{\langle i\rangle}\hat{\mathbf{c}}_{\langle i\rangle}\|^2 = \|\tilde{\mathbf{U}}_{\langle i\rangle}\hat{\mathbf{c}}_{\langle i\rangle}\|^2, \tag{14}$$

where $\tilde{\mathbf{U}}_{\langle i\rangle} = [\mathbf{U}_{\langle i\rangle}(:,1) - \mathbf{x}_i, \ldots, \mathbf{U}_{\langle i\rangle}(:,s) - \mathbf{x}_i]$ can be viewed as the new coordinates of s closest anchors of $\mathbf{x}_i$ in the datapoint-centered coordinate system.

Then, according to Eqs. (11), (12), we scale the code $\hat{\mathbf{c}}_{\langle i\rangle}$ in Eq. (14) by a constant ρ and obtain the value of the objective function in Eq. (7) with our feasible solution $\mathbf{c}_{\langle i\rangle}$,

$$\|\tilde{\mathbf{U}}_{\langle i\rangle}\mathbf{c}_{\langle i\rangle}\|^2 = \|\frac{1}{\rho}\tilde{\mathbf{U}}_{\langle i\rangle}\hat{\mathbf{c}}_{\langle i\rangle}\|^2 = \frac{1}{\rho^2}\|\tilde{\mathbf{U}}_{\langle i\rangle}\hat{\mathbf{c}}_{\langle i\rangle}\|^2. \tag{15}$$

Since $\exists\, \hat{c}_{ij} < 0$, we have $\rho > 1$. Therefore, we obtain the inequality relation that the value of Eq. (15) is smaller than Eq. (14).

Up to now, we have demonstrated that the value of the objective function in Eq. (7) with our feasible solution is not greater than the value of Eq. (8) with its optimal solution, which means we can obtain a smaller reconstruction error than LAE. Moreover, our analytical solution based on Eqs. (9)-(13) is much faster than the iterative solution obtained by LAE.

Change 2. We further incorporate the locality constraint into local weight estimation. To enhance coding efficiency, the locality constraint functions in other similar methods [21], [41] are replaced by using s closest anchors (landmarks), like approximated LLC [39] does. However, this manipulation is insufficiently suitable, because it ignores the real distance between local points (an negative example is shown in Fig. 1b). We therefore suggest keeping the locality constraint while using s closest anchors in local weight estimation simultaneously.

We obtain local weight vector $\mathbf{z}_{\langle i\rangle} = |\mathbf{c}_{\langle i\rangle}|$ according to the following objective:

$$\underset{\mathbf{c}_i}{\operatorname{argmin}} \|\mathbf{x}_i - \mathbf{U}_{\langle i\rangle}\mathbf{c}_{\langle i\rangle}\|^2 + \lambda\|\mathbf{d}_{\langle i\rangle} \odot \mathbf{c}_{\langle i\rangle}\|^2 \quad (16)$$

$$\text{s.t.} \mathbf{1}^{\mathrm{T}}|\mathbf{c}_{\langle i\rangle}| = 1,$$

where $\odot$ denotes the element-wise multiplication. $\mathbf{d}_{\langle i\rangle}$ is the locality adaptor that allows a different freedom for each anchor proportional to its distance to the datapoint $\mathbf{x}_i$. Specifically,

$$\mathbf{d}_{\langle i\rangle}(k) = \exp\left(\frac{\|\mathbf{x}_i - \mathbf{U}_{\langle i\rangle}(:,k)\|^2}{\sigma}\right), \quad k = 1, \ldots, s \quad (17)$$

where the parameter σ is used to adjust the weight decay speed for the locality adaptor.

Similarly, we can obtain a feasible solution as before. The only change compared to the pervious processes is that we replace Eq. (9) with

$$\tilde{\mathbf{c}}_{\langle i\rangle} = (\mathbf{C}_i + \lambda \operatorname{diag}(\mathbf{d}_{\langle i\rangle}))^{-1}\mathbf{1}. \quad (18)$$

To summarize, we demonstrate the steps of FLAE in Algorithm 1.

Algorithm 1. Fast Local Anchor Embedding (FLAE)

Input: datapoints $\{\mathbf{x}_i\}_{i=1}^n \in \mathbb{R}^{d\times 1}$, anchor set obtained from K-means $\mathbf{U} \in \mathbb{R}^{d\times m}$, parameters s and λ.

for $i = 1$ **to** n **do**

1. For $\mathbf{x}_i$, find s nearest neighbors in $\mathbf{U}$ and record the index set $\langle i\rangle$.
2. Measure the locality adaptor $\mathbf{d}_{\langle i\rangle}$ for datapoint $\mathbf{x}_i$ with its nearest neighbors $\mathbf{U}_{\langle i\rangle}$ via Eq. (17).
3. Compute the code $\mathbf{c}_{\langle i\rangle}$ via Eq. (18), and Eqs. (10)-(12).
4. Obtain the final solution $\mathbf{z}_{\langle i\rangle}$ via Eq. (13).
5. $Z_{i,\langle i\rangle} = \mathbf{z}_{\langle i\rangle}^{\mathrm{T}}$.

end for

Output: FLAE matrix $\mathbf{Z}$.

4.2 Normalized Graph Laplacian over Anchors

We consider a graph as a stable one if it satisfies the cluster assumption, which means nearby datapoints are likely to have the same labels. Obviously, it is important for building a well performed anchor-graph-based SSL classification model. In Section 3.2, we have observed the limitation of the anchor graph built based on the conventional adjacency between datapoints. As demonstrated in Section 3.1, in anchor graph regularization, the regularization on all datapoints is actually equivalent to regularization on anchors with a reduced graph Laplacian, which is built over only anchors. Therefore, here we directly design an adjacency matrix among anchors and then derive a normalized graph Laplacian over anchors. For clarity, we use the subscripts i, j, k and s, t, r to denote the indices of datapoints and anchors respectively in this section. The details are as follows.

Recall that the label for each datapoint is estimated as a weighted average of the labels of the anchor set via Eq. (1), which only involves the local weight vector of the datapoint and the labels of its nearest anchors. Given local weights, the label vectors of the nearest anchors are crucial to the final label prediction for the datapoint. If two nearby datapoints share a lot of nearest anchors, their labels are likely to be similar. However, nearby datapoints are not guaranteed to have identical nearest anchors quite often. Fig. 3 illustrates a toy example, where the nearest anchors of nearby pairwise datapoints x_1 and x_2 are the same while those of x_1 and x_3 are not. We prefer the Laplacian matrix has the following characteristics: 1) the elements corresponding to the nearby pairwise anchors should be negative, which means these anchors tend to have similar labels; 2) the elements corresponding to dissimilar pairwise anchors should be zero, which means their labels are irrelevant [27]. To this end, we design the adjacency matrix between anchors as

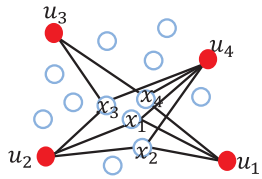


Fig. 3. Toy example of the nearest anchors of datapoints.

$$\mathbf{W} = \mathbf{Z}^{\mathrm{T}}\mathbf{Z}, \quad (19)$$

where $W_{st} = \sum_{k=1}^n Z_{ks}Z_{kt}$. Accordingly, its normalized Laplacian matrix is

$$\bar{\mathbf{L}} = \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2}, \quad (20)$$

where the diagonal matrix $\mathbf{D}$ is defined as $D_{ss} = \sum_{t=1}^m W_{st}$. Note that our adjacency matrix actually explores all datapoints as the transitional points rather than anchors alone. Thus, our model keeps the computational efficiency of the anchor graph regularization and its effectiveness in regularization.

As for AGR, it constructs an $n \times n$ adjacency matrix as $\mathbf{W} = \mathbf{Z}\Lambda^{-1}\mathbf{Z}^{\mathrm{T}}$. Based on this, AGR employs the following reduced Laplacian matrix $\tilde{\mathbf{L}}$ over anchors for its regularization function

$$\begin{aligned}\tilde{\mathbf{L}} &= \mathbf{Z}^{\mathrm{T}}(\mathbf{I} - \mathbf{Z}\Lambda^{-1}\mathbf{Z}^{\mathrm{T}})\mathbf{Z}, \\ &= \mathbf{Z}^{\mathrm{T}}\mathbf{Z} - \mathbf{Z}^{\mathrm{T}}(\mathbf{Z}\Lambda^{-1}\mathbf{Z}^{\mathrm{T}})\mathbf{Z}.\end{aligned} \quad (21)$$

Now we compare the two $m \times m$ Laplacian matrices over anchors, according to the previously mentioned characteristics. For convenience, we take a non-diagonal element L_{st} for example, which is computed as

$$L_{st} = \begin{cases} -\alpha\mathbf{Z}_{\cdot s}^{\mathrm{T}}\mathbf{Z}_{\cdot t} & \text{for EAGR,} \\ \mathbf{Z}_{\cdot s}^{\mathrm{T}}\mathbf{Z}_{\cdot t} - \mathbf{Z}_{\cdot s}^{\mathrm{T}}(\mathbf{Z}\Lambda^{-1}\mathbf{Z}^{\mathrm{T}})\mathbf{Z}_{\cdot t} & \text{for AGR ,} \end{cases} \quad (22)$$

where $\alpha = (D_{ss}D_{tt})^{-\frac{1}{2}}$, $\mathbf{Z}_{\cdot s} \in \mathbb{R}^{n\times 1}$ denotes the sth col of $\mathbf{Z}$.

We discuss the sign of L_{st} in the following two cases according to the relations of the pairwise anchors $\mathbf{u}_s\&\mathbf{u}_t$.

(1) The first case is that $\mathbf{u}_s\&\mathbf{u}_t$ share at least one common datapoint $\mathbf{x}_i$, namely, they are nearby. For EAGR, we clearly have $L_{st} = -\alpha\mathbf{Z}_{\cdot s}^{\mathrm{T}}\mathbf{Z}_{\cdot t} \leq -\alpha Z_{is}Z_{it} < 0$.

However, for AGR, the sign of the element L_{st} depends on the values of $\mathbf{Z}_{\cdot s}^{\mathrm{T}}\mathbf{Z}_{\cdot t}$ and $\mathbf{Z}_{\cdot s}^{\mathrm{T}}(\mathbf{Z}\Lambda^{-1}\mathbf{Z}^{\mathrm{T}})\mathbf{Z}_{\cdot t}$. Therefore, the element L_{st} can be positive, zero, or negative. If $L_{st} > 0$, these nearby anchors tend to have different labels. If $L_{st} = 0$, these nearby anchors tend to be irrelevant. When $L_{st} < 0$, we can obtain the expected result, that is, nearby anchors tend to have similar labels. In short, the labels of nearby pairwise anchors in EAGR are enforced to be similar, while this is uncertain for AGR.

(2) The second case is that $\mathbf{u}_s\&\mathbf{u}_t$ does not share any common datapoint, namely, they are dissimilar. For EAGR, we have $L_{st} = -\alpha\mathbf{Z}_{\cdot s}^{\mathrm{T}}\mathbf{Z}_{\cdot t} = 0$. However, for AGR, although $\mathbf{Z}_{\cdot s}^{\mathrm{T}}\mathbf{Z}_{\cdot t}$ is zero, $\mathbf{Z}_{\cdot s}^{\mathrm{T}}(\mathbf{Z}\Lambda^{-1}\mathbf{Z}^{\mathrm{T}})\mathbf{Z}_{\cdot t}$ may still be positive, which makes L_{st} negative. For example, anchors $\mathbf{u}_s\&\mathbf{u}_t$ connect with two different datapoints $\mathbf{x}_i\&\mathbf{x}_j$ respectively, and these datapoints share another anchor $\mathbf{u}_r$ simultaneously. Then, we obtain $L_{st} < 0$, since $L_{st} = 0 - \mathbf{Z}_{\cdot s}^{\mathrm{T}}(\mathbf{Z}\Lambda^{-1}\mathbf{Z}^{\mathrm{T}})\mathbf{Z}_{\cdot t} < 0 - Z_{is} \quad (\mathbf{Z}\Lambda^{-1}\mathbf{Z}^{\mathrm{T}})_{ij}Z_{jt}$, where $(\mathbf{Z}\Lambda^{-1}\mathbf{Z}^{\mathrm{T}})_{ij} \geq Z_{ir}\Lambda_{rr}^{-1}Z_{jr} > 0$ and $Z_{is} > 0$, $Z_{jt} > 0$. Therefore, unexpected negative Laplacian weights could exist between dissimilar anchors in AGR model while our EAGR is free from this situation.

Later in Section 5.2, based on real-world datasets, we will show several examples by comparing the number of non-zero elements at the non-diagonal positions of the above $m \times m$ Laplacian matrices. In conclusion, based on the proposed $\mathbf{W}$, we can better describe the adjacency between anchors and make the smoothness constraint more effective for the stable anchor graph construction. As the common linked datapoints are used as transitional points in building $\mathbf{W}$, we note that our adjacency matrix $\mathbf{W}$ is totally different from the simple kNN graph, which only depends on the sparse anchors themselves.

4.3 Learning on Anchor Graph

Now we consider a standard multi-class SSL task. Given a set of labeled datapoints $\mathbf{x}_i \ (i = 1, \ldots, l)$ with the corresponding discrete label $y_i \in \{1, \ldots, c\}$, our goal is to predict the labels on the remaining unlabeled real datapoints associated with anchors. Let $\mathbf{Y}_l = [\mathbf{y}_1{}^{\mathrm{T}}, \mathbf{y}_2{}^{\mathrm{T}}, \ldots, \mathbf{y}_l{}^{\mathrm{T}}]^{\mathrm{T}} \in \mathbb{R}^{l\times c}$ denote a class indicator matrix on labeled datapoints with $Y_{ij} = 1$ if $\mathbf{x}_i$ belongs to class j and $Y_{ij} = 0$ otherwise. Suppose that, for each datapoint, we obtain a label prediction function $\boldsymbol{f}_i = \mathbf{z}_i^{\mathrm{T}}\mathbf{A}$ where $\mathbf{A} = [\mathbf{a}_1{}^{\mathrm{T}}, \mathbf{a}_2{}^{\mathrm{T}}, \ldots, \mathbf{a}_m{}^{\mathrm{T}}]^{\mathrm{T}} \in \mathbb{R}^{m\times c}$ denotes the prediction label matrix on the anchor set. Specifically, $\mathbf{a}_i \in \mathbb{R}^{1\times c}$ is the label vector of the anchor $\mathbf{u}_i$. Then, we combine anchors with datapoints to build a model called Efficient Anchor-Graph Regularization (EAGR) for SSL as follows:

$$\underset{\mathbf{A}}{\operatorname{argmin}} \sum_{i=1}^{l} \|\mathbf{z}_i^{\mathrm{T}}\mathbf{A} - \mathbf{y}_i\|^2 + \frac{\gamma}{2}\sum_{i,j=1}^{m} W_{ij}\|\frac{\mathbf{a}_i}{\sqrt{D_{ii}}} - \frac{\mathbf{a}_j}{\sqrt{D_{jj}}}\|^2. \quad (23)$$

We can also present this optimization problem in the following matrix form:

$$\underset{\mathbf{A}}{\operatorname{argmin}} \|\mathbf{Z}_l\mathbf{A} - \mathbf{Y}_l\|_{\mathrm{F}}^2 + \gamma\mathrm{tr}(\mathbf{A}^{\mathrm{T}}\bar{\mathbf{L}}\mathbf{A}), \quad (24)$$

where $\bar{\mathbf{L}} = \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2} \in \mathbb{R}^{m\times m}$ is the previously introduced normalized Laplacian matrix, $\|\cdot\|_{\mathrm{F}}$ stands for the Frobenius norm, and $\gamma > 0$ is the trade-off parameter.

With simple algebra, we can obtain a globally optimal solution for the anchors' label matrix as follows:

$$\mathbf{A} = (\mathbf{Z}_l^{\mathrm{T}}\mathbf{Z}_l + \gamma\bar{\mathbf{L}})^{-1}\mathbf{Z}_l^{\mathrm{T}}\mathbf{Y}_l. \quad (25)$$

This yields a closed-form solution for handling large scale SSL tasks. Lastly, like AGR, we utilize the local weight matrix with anchors' soft scores to predict the hard labels for unlabeled datapoints as Eq. (6).

To summarize, our EAGR consists of the following steps: (1) finding anchors by K-means; (2) estimating the local weight matrix $\mathbf{Z}$ by FLAE, as illustrated in Algorithm 1; (3) computing the normalized Laplacian matrix $\bar{\mathbf{L}}$ over anchors via Eq. (20); (4) carrying out the graph regularization via Eq. (25); and (5) predicting the hard labels on unlabeled datapoints via Eq. (6). As we can see, our EAGR keeps the simplicity of AGR and tends to be much faster, since it efficiently solves the local weight estimation with FLAE. To be clear, Tables 1 and 2 show the storage costs and time complexity of several semi-supervised learning algorithms, where n is the number of datapoints, m is the number of anchors, s is the number of the closest anchors to a datapoint, d is the dimension of features, and t is the number of iterations in the corresponding iterative optimization process.

TABLE 1
Comparison of Storage Costs of Three Graph-Based Methods

Approach	Storage
Learning with Local and Global Consistency (LLGC) [51]	$O(n^2)$
Anchor Graph Regularization (AGR) [21]	$O(mn)$
Efficient Anchor Graph Regularization (EAGR)	$O(mn)$

TABLE 2
Comparison of Computational Complexities of Three Graph-Based Methods

Approach	Find anchors	Design $\mathbf{Z}$	(reduced) graph Laplacian $\mathbf{L}$	Graph Regularization
LLGC	–	–	$O(dn^2)$	$O(n^3)$
AGR	$O(mndt)$	$O(mns + s^2dnt)$	$O(m^2n)$	$O(m^2n + m^3)$
EAGR	$O(mndt)$	$O(mns + s^2dn)$	$O(m^2n)$	$O(m^2n + m^3)$

TABLE 3
Details of the Five Datasets

	Alphadigits	Semeion	USPS	Letter	MNIST
# of instances	1,404	1,593	11,000	20,000	60,000
# of categories	36	10	10	26	10
# of features	320	256	256	16	784

5 Experiments

5.1 Data Settings

To evaluate the performance of our EAGR, we conduct experiments on five widely-used handwritten datasets: Binary Alphadigits[1] (Alphadigits for short), USPS[1], MNIST[2], Semeion,[2] and Letter Recognition[2] (Letter for short). Following the settings of [21], we divide them into three groups based on their sizes, that is, number of samples. It is worthwhile to note that the images in Letter have already been converted into 16 primitive numerical attributes (statistical moments and edge counts) to represent each sample. In other datasets, samples are still pixel images of different size. For these datasets, we directly use a high-dimensional vector of normalized grayscale values to represent each instance. The attributes of these datasets are listed in Table 3. All the experiments are implemented on a 2.4 GHz CPU, 32 GB RAM PC.

5.2 On the Two Improvements in EAGR

Before comparing our EAGR with other methods, we conduct two experiments to validate the effectiveness of the two improvements described in Sections 4.1 and 4.2 respectively. For a fair comparison, we define two intermediate versions between AGR and EAGR as:

(1) AGR+, which employs FLAE for estimating $\mathbf{Z}$ and use the reduced Laplacian matrix in Eq. (21).
(2) EAGR− , which employs LAE for estimating $\mathbf{Z}$ and use the normalized Laplacian matrix in Eq. (20).

The differences of the methods for comparison are also summarized in Table 4. For the involved parameters, we simply set anchor number to 500 and empirically set s to 3 to make the anchor graph sparse. We tune other parameters, i.e., λ and γ, to their optimal values. In this way, we can provide a fair comparison for these algorithms. We randomly select 10 labeled samples per class and leave the remaining ones unlabeled for SSL models in this section.

Table 5 shows the classification accuracies of the four different methods. Table 6 presents the average CPU time (in seconds) of two local estimation methods on different datasets. Meanwhile, Table 7 illustrates the number of nonzero elements at the non-diagonal positions in different Laplacian matrices as mentioned in Sectioin 4.2. From the results, we have two observations as follows.

First, by comparing AGR+ with AGR, we see that the former has comparable or better performances than the latter. In addition, Table 6 reveals that our proposed FLAE in AGR+ is much faster than LAE in AGR. These performances demonstrate the efficiency of our improvement on the local weight estimation.

1. available at http://cs.nyu.edu/~roweis/data.html
2. available at http://archive.ics.uci.edu/ml/

Second, by comparing EAGR- with AGR, we see that, although two methods use the same optimized $\mathbf{Z}$, the EAGR- has better classification performance than AGR on all five datasets. We can see from Table 7 that the normalized graph Laplacian of EAGR- has less non-zero elements than the reduced graph Laplacian of AGR. This means that, with a sparse normalized graph Laplacian, several incorrect links have been removed.

5.3 Comparison with Other Methods

Here we further compare the proposed EAGR approach with the following methods.

(1) FLAE-based label inference, which constructs the local matrix between unlabeled datapoints and the labeled ones with FLAE, and then predicts labels for unlabeled datapoints. This method actually introduces FLAE into the kernel regression approach [3], since each row vector $\mathbf{z}_i$ of $\mathbf{Z}$ is non-negative and normalized. The method is denoted as "FLAE-LI".
(2) The Eigenfunctions method introduced in [11], which solves the semi-supervised problem in a dimension-reduced feature space by only working with a small number of eigenvectors of the Laplacian. The method is denoted as "Eigenfucntion".
(3) Laplacian Support Vector Machines Trained in the Primal with preconditioned conjugate gradient, which is introduced in [26]. We first decompose a c-class classification problem into c one-versus-rest binary classification problems. Prediction is then made by assigning the sample to the class with the highest decision function value. The method is denoted as "LapSVMp".
(4) Learning with local and global consistency in [51], which is a typical graph-based learning method. It directly computes the affinity matrix with Gaussian kernel. In our experiments, we use the 6NN strategy in graph construction and use Gaussian kernel for weighting the edges. The method is denoted as "LLGC".
(5) Prototype Vector Machines with the Square Loss in [47], [48], which is a scale-up graph-based semi-supervised learning for multi-label classification tasks using a set of sparse prototypes derived from the data and the Gaussian kernel is used to define the graph affinity. The method is denoted as "PVM".
(6) Anchor Graph Regularization, which employs clustering centers as anchors with LAE, is proposed in [21]. It is the prime counterpart in our experiments, and we aim to improve its performance. The method is denoted as "AGR".

Since the last two methods and EAGR are based on either the anchors or the prototypes, we group them into "landmark-based learning" methods and perform K-means to obtain these cluster centers as the landmarks. Aiming at a fair comparison, the radius parameters of the Gaussian kernel in the methods (2-4) are set by five-fold cross-validation. The trade-off parameters in regularization of the above methods are empirically tuned to their optimal values.

5.3.1 Small Size Datasets

We first conduct experiments on two small datasets: Alphadigits and Semeion. For the three landmark-based methods,

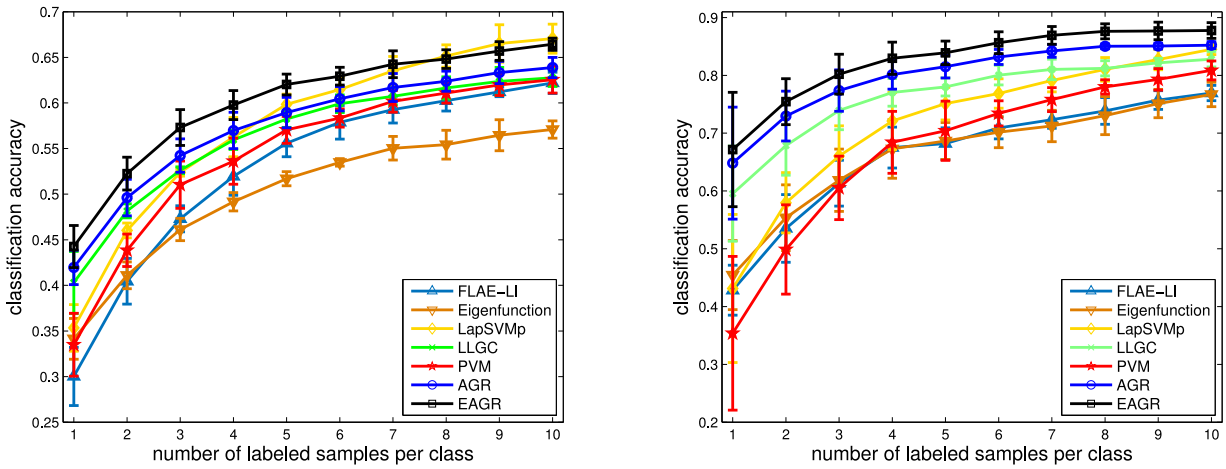
TABLE 4
Comparison of AGR, EAGR, AGR+, and EAGR-

	AGR	AGR+	EAGR-	EAGR
local weight matrix $\mathbf{Z}$	LAE	FLAE	LAE	FLAE
(reduced) graph Laplacian	$\mathbf{Z}^T\mathbf{Z} - \mathbf{Z}^T(\mathbf{Z}\mathbf{\Lambda}^{-1}\mathbf{Z}^T)\mathbf{Z}$	$\mathbf{Z}^T\mathbf{Z} - \mathbf{Z}^T(\mathbf{Z}\mathbf{\Lambda}^{-1}\mathbf{Z}^T)\mathbf{Z}$	$\mathbf{I} - \mathbf{D}^{-1/2}\mathbf{Z}^T\mathbf{Z}\mathbf{D}^{-1/2}$	$\mathbf{I} - \mathbf{D}^{-1/2}\mathbf{Z}^T\mathbf{Z}\mathbf{D}^{-1/2}$

For AGR and AGR+, we demonstrate the reduced graph Laplacian over anchors.

TABLE 5
Comparison on Classification Accuracy (Mean±std) of Different Anchor-Graph-Based Approaches

	AGR	AGR+	EAGR-	EAGR
Alphadigits	64.43 ± 1.06	65.21 ± 1.32	65.27 ± 0.76	65.92 ± 0.71
Semeion	85.16 ± 1.25	87.05 ± 1.10	85.31 ± 0.92	87.17 ± 1.33
USPS	86.53 ± 1.34	86.21 ± 1.46	87.62 ± 1.20	87.21 ± 1.33
Letter	57.35 ± 0.81	60.40 ± 0.97	58.28 ± 0.97	61.31 ± 1.04
MNIST	86.66 ± 1.14	86.51 ± 1.19	88.54 ± 1.06	88.45 ± 0.94



(a) Alphadigits (b) Semeion

Fig. 4. Classification accuracy versus the number of labeled samples on small-size datasets.

we empirically set the landmark number to 500 and $l = \{1, 2, \ldots, 10\}$ labeled samples per class. The average classification accuracies with standard deviations over 10 trials are illustrated in Fig. 4. As a general trend, it can be seen that, as the number of labeled data increases, the performances of all methods become better. In addition, we observe that the performances of the LapSVMp, LLGC, PVM, AGR, and EAGR methods stay at a higher level than Eigenfunction. The reason is that the former produce a good complete graph to model the data distribution, while the latter builds a backbone graph only. Specifically, the proposed EAGR method has the similar standard deviations with AGR, and it achieves the best accuracy among all landmark-based SSL methods in most cases, which demonstrates the effectiveness of the improved relationship modeling.

Besides classification accuracies, we record the implementation time costs of the above algorithms with 10 labeled samples per class in Table 8. It is noted that the time costs of K-means clustering in the three landmark-based SSL methods are listed separately in the table and the time excluding clustering is listed at the right side. As can be seen, excluding the time costs of K-means, our EAGR method is faster than AGR and PVM, which demonstrates the efficiency of our improvement among landmark-based methods. Although the total time costs of these landmark-

TABLE 6
Comparison on the Time Costs (Seconds) of Different Local Weight Estimation Approaches

	LAE	FLAE
Alphadigits	2.03	0.23
Semeion	2.34	0.28
USPS	15.81	1.80
Letter	27.35	2.80
MNIST	79.06	11.01

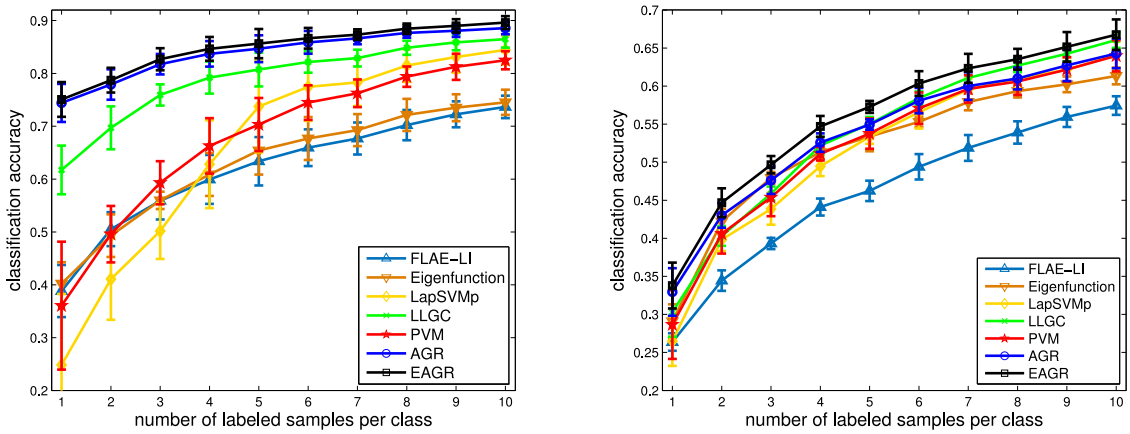
TABLE 7
Number of Non-Zero Elements at the Non-Diagonal Positions in Different Graph Laplacian Matrices

	Alphadigits	Semeion	USPS	Letter	MNIST
AGR	19,580	21,540	189,330	152,556	697,658
EAGR-	3,340	3,540	25,252	24,276	68,172

For AGR, we count the numbers in reduced graph Laplacian matrices over anchors. We can see that the normalized graph Laplacian of EAGR- is more sparse.

TABLE 8
Time Costs (Seconds) of the Compared Learning Algorithms on Small-Size Datasets

	Landmark-based Learning				Others			
	K-means	AGR	EAGR	PVM	FLAE-LI	Eigenfunction	LapSVMp	LLGC
Alphadigits	2.10	+2.17	+0.28	+0.32	0.14	0.95	7.25	0.73
Semeion	2.01	+2.48	+0.30	+0.34	0.13	0.73	1.13	0.83



(a) USPS (b) Letter

Fig. 5. Classification accuracy versus the number of labeled samples on medium-size datasets.

TABLE 9
Time Costs (Seconds) of the Compared Learning Algorithms on Medium-Size Datasets

	Landmark-Based Learning				Others			
	K-means	AGR	EAGR	PVM	FLAE-LI	Eigenfunction	LapSVMp	LLGC
USPS	59.00	+20.72	+4.45	+16.43	0.69	4.95	56.68	120.67
Letter	11.15	+35.42	+7.16	+25.04	1.14	1.09	139.39	432.73

based methods are larger than LLGC on these small datasets, we will see that LLGC will be quite slow on large datasets. Here we have simply used K-means to obtain cluster centers like [21], as clustering is not the focus of this work. In fact, many fast clustering algorithms can be explored to improve the speed of this process [1], [10].

5.3.2 Medium Size Datasets

For the USPS and Letter datasets, we empirically set the landmark number to 2,000 by taking both effectiveness and efficiency into consideration. Averaged over 10 trials, we calculate the classification accuracies with standard deviations for the referred methods. The results of USPS and Letter are shown in Fig. 5. Here we have the following three observations. First, although FLAE-LI produces reasonable results, it has lower accuracy than semi-supervised methods in most cases. This demonstrates the importance of graph construction which utilizes unlabeled samples in regularization. Second, although all the methods perform poorly when the number of labeled samples is small, the two anchor-graph-based methods are clearly superior to LLGC. A possible reason is that the fitting constraints of both AGR and EAGR are built on the labeled samples while the constraints of LLGC are built on the whole set. The fitting effects of AGR and EAGR, therefore, are more biased towards labeled information than that of LLGC. Third, when the number of labeled samples increases, the performances of all methods become better and the accuracy of EAGR improves more significantly than AGR. The reason is that, as the number of labeled samples increases, these classifiers model the characteristics of different classes better. However, this increase also makes the classifiers tend to be overfitted, which needs to be handled by an effective smoothness constraint. Owning to the better relationship modeling in the data space to meet this requirement, EAGR addresses the limitation of AGR as expected.

We also demonstrate the implementation time costs of the above algorithms with 10 labeled samples per class in Table 9. We can observe that, although LapSVMp is faster than LLGC, it is still computationally expensive on the larger data set, e.g., Letter, as its time complexity is quadratic with respect to n. The landmark-based learning algorithms, especially EAGR, need much less implementation costs than LLGC. In addition, the K-means clustering process accounts for the main portion of the EAGR's time cost. Therefore, if we can reduce the clustering time by employing other fast clustering algorithms [1], [10] or just selecting a part of the database samples to run K-means like [43], the advantage of our EAGR over the other two in terms of speed is expected to be greater.

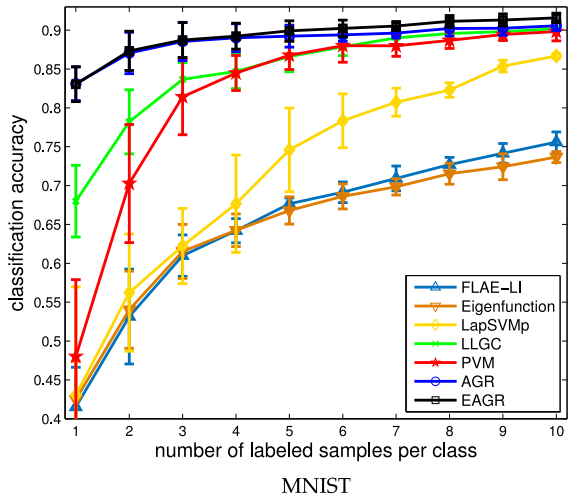


Fig. 6. Classification accuracy versus the number of labeled samples on the large-size dataset.

5.3.3 Large Size Dataset

For the MNIST dataset, we empirically set the landmark number to 2,000 in the three landmark-based methods. Fig. 6 shows the results of the algorithms on MNIST. We can observe that the two anchor-graph-based methods are clearly superior to LLGC when the number of labeled samples is small. Similar to the results in the previous experiments, we see that our method achieves better performance than AGR and PVM. Meanwhile, Table 10 also demonstrates that our EAGR is more advantageous than the AGR and PVM methods in terms of speed for the large dataset.

5.4 On the Parameters λ and γ

Finally, we test the sensitivity of the two parameters λ and γ involved in the proposed algorithm. For convenience, we simply set anchor number to 500 and empirically set $s = 3$. We first set γ to 1 and vary λ from 0.001 to 1,000. Fig. 7 shows the performance curve with respect to the variation of λ. From the figure, we observe that our method consistently outperforms AGR, and the performances stay at a stable level over a wide range of parameter variation. We then vary the value of γ from 0.001 to 1,000 for both EAGR and AGR ($\lambda = 10$). Fig. 8 demonstrates the performance variation. We can see that EAGR is superior to AGR when γ varies in a wide range and the curve of EAGR has a peak when $\gamma = 1$ in most cases. These observations demonstrate the robustness of the parameter selection in applying our method to different datasets.

6 Conclusion

This work introduces a novel scalable graph-based semi-supervised learning algorithm named Efficient Anchor Graph Regularization (EAGR). It improves the AGR approach in the following two aspects. First, in anchor graph

TABLE 10
Time Costs (Seconds) of the Compared Learning Algorithms on Large-Size Dataset

	Landmark-Based Learning				Others			
	K-means	AGR	EAGR	PVM	FLAE-LI	Eigenfunction	LapSVMp	LLGC
MNIST	168.43	+130.07	+35.15	+87.74	5.59	27.40	3296.71	9521.15

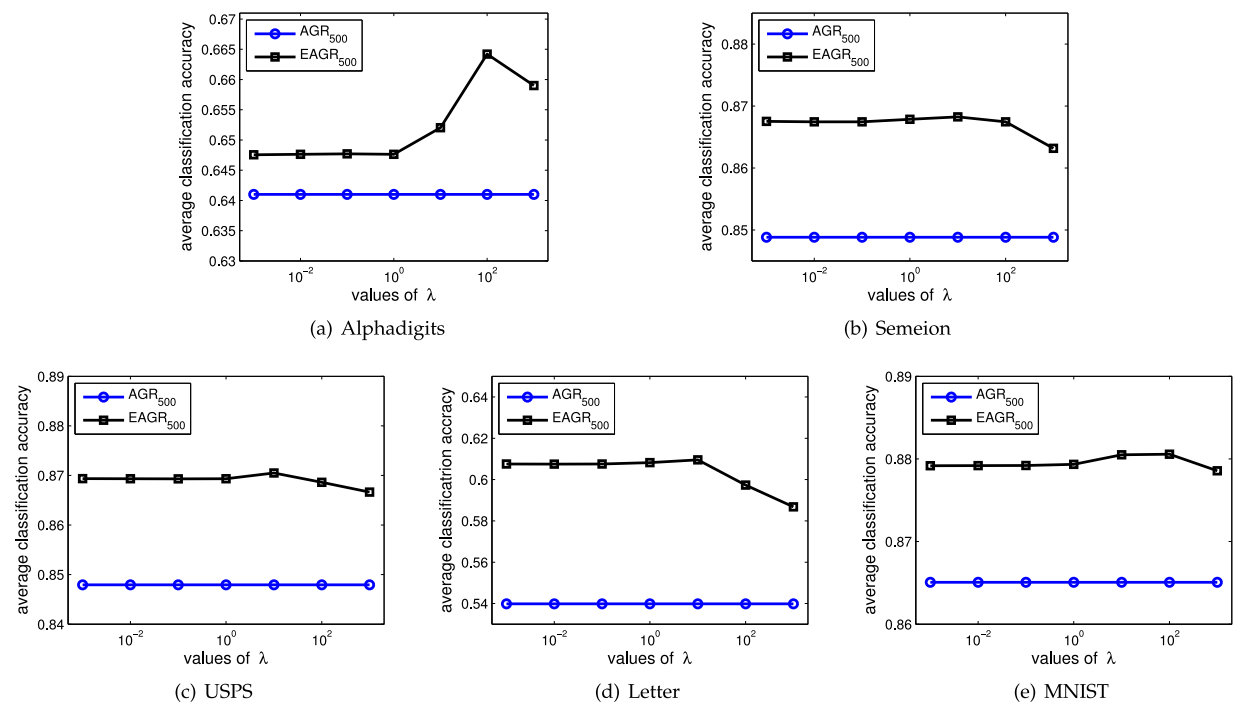


Fig. 7. Average performance curves of EAGR with respect to the variation of λ. Here, the number of labeled samples is set to 10 per class.

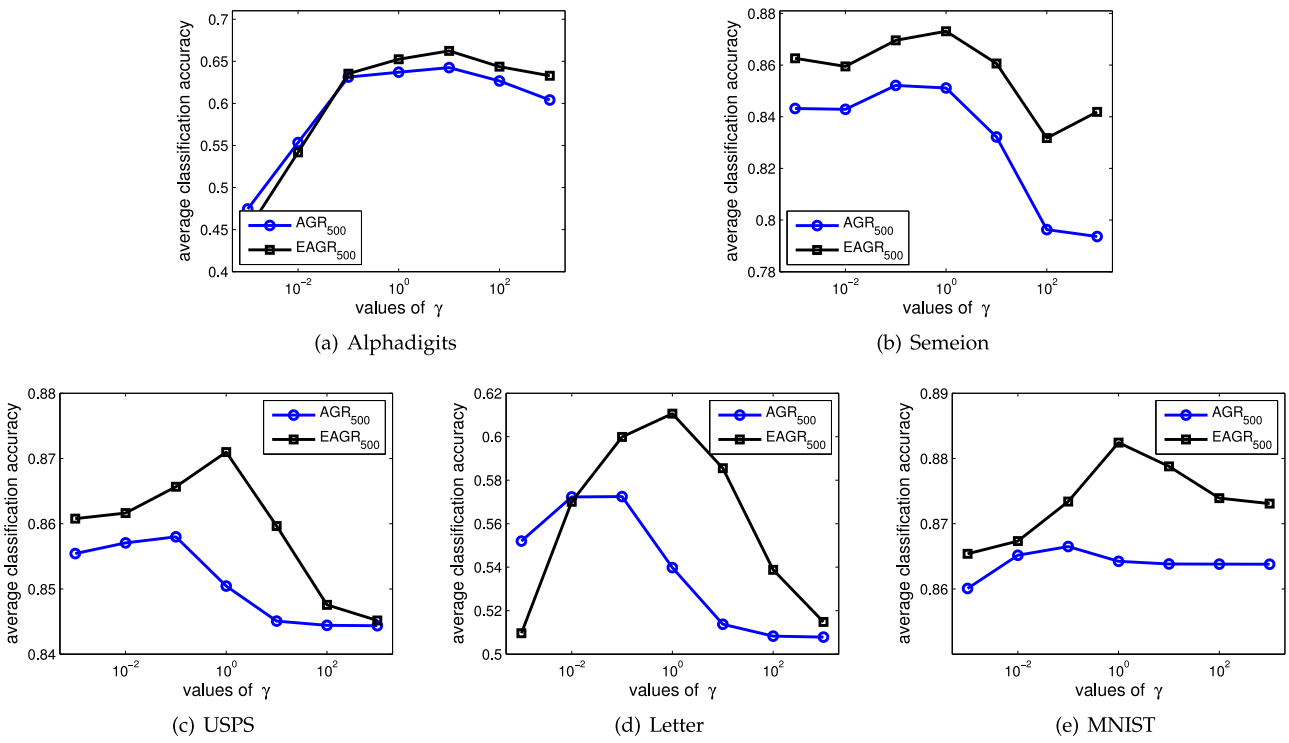


Fig. 8. Average performance curves of EAGR and AGR with respect to the variation of γ. Here, the number of labeled samples is set to 10 per class.

construction, it employs a novel fast local anchor embedding method to better measure the local weights between datapoints and neighboring anchors. Second, in anchor graph regularization, it employs a novel normalized graph Laplacian over anchors, which works better than the reduced graph Laplacian in AGR. For each improvement, we have provided an in-depth analysis on the limitation of the conventional method and the advantages of the new method. Experiments on publicly available datasets of various sizes have validated our EAGR in terms of classification accuracy and computational speed.

Appendix

Proof of Proposition 1

First, we present the following lemma.

Lemma 1. *Suppose f_1 is the minimization problem with objective function g in domain A, f_2 is the minimization problem with the same objective function g in domain B, and $B \subset A$, then for the optimal solution to f_1, e.g., x_A, and the optimal solution to f_2, e.g., x_B, we have $g(x_A) \leq g(x_B)$.*

The proof of the above lemma is clear. Since x_B is the optimal solution to f_2 in domain B, we have $x_B \in B$. Note $B \subset A$, so $x_B \in A$. Therefore, if $g(x_A) > g(x_B)$, x_A is not the optimal solution to f_1 in domain A.

Now we prove Proposition 1. Clearly, both Eq. (8) and Eq. (2) have the same objective function. In addition, if A denotes the domain of Eq. (8), i.e., $\mathbf{1}^{\mathrm{T}}\hat{\mathbf{c}}_{\langle i \rangle} = 1$ and B denotes the domain of Eq. (2), i.e., $\mathbf{1}^{\mathrm{T}}\mathbf{z}_{\langle i \rangle} = 1, \mathbf{z}_{\langle i \rangle} \succeq 0$, we have $B \subset A$. According to Lemma 1, the minimum of Eq. (8) is not greater than the minimum of Eq. (2), which completes the proof.

Acknowledgments

This work is partially supported by the National 973 Program of China under grants 2014CB347600 and 2013CB329604, the Program for Changjiang Scholars and Innovative Research Team in University (PCSIRT) of the Ministry of Education, China, under grant IRT13059, the National Nature Science Foundation of China under grants 61272393, 61322201 and 61432019, and the Australian Research Council Projects under grants DP-140102164 and FT-130101457.

References

[1] J. S. Beis and D. G. Lowe, "Shape indexing using approximate nearest-neighbour search in high-dimensional spaces," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, Jun. 1997, pp. 1000–1006.

[2] M. Belkin and P. Niyogi, "Semi-supervised learning on riemannian manifolds," *Mach. Learning*, vol. 56, nos. 1–3, pp. 209–239, 2004.

[3] C. M. Bishop, *Pattern Recognition and Machine Learning*. New York, NY, USA: Springer, 2006.

[4] A. Blum and T. Mitchell, "Combining labeled and unlabeled data with co-training," in *Proc. 11th Annu. Conf. Comput. Learning Theory*, 1998, pp. 92–100.

[5] D. Cai and X. Chen, "Large scale spectral clustering with landmark-based representation," *IEEE Trans. Cybern.*, vol. 45, no. 8, pp. 1669–1680, Aug. 2015.

[6] V. Castelli and T. M. Cover, "On the exponential value of labeled samples," *Pattern Recog. Lett.*, vol. 16, no. 1, pp. 105–111, 1995.

[7] J. Cheng, C. Leng, P. Li, M. Wang, and H. Lu, "Semi-supervised multi-graph hashing for scalable similarity search," *Comput. Vis. Image Understanding*, vol. 124, pp. 12–21, 2014.

[8] C. Deng, R. Ji, W. Liu, D. Tao, and X. Gao, "Visual reranking through weakly supervised multi-graph learning," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2013, pp. 2600–2607.

[9] C. Deng, R. Ji, D. Tao, X. Gao, and X. Li, "Weakly supervised multi-graph learning for robust image reranking," *IEEE Trans. Multimedia*, vol. 16, no. 3, pp. 785–795, Apr. 2014.

[10] C. Elkan, "Using the triangle inequality to accelerate k-means," in *Proc. 20th Int. Conf. Mach. Learning*, 2003, vol. 3, pp. 147–153.

[11] R. Fergus, Y. Weiss, and A. Torralba, "Semi-supervised learning in gigantic image collections," in *Proc. Conf. Adv. Neural Inform. Process. Syst.*, 2009, pp. 522–530.

[12] P. Foggia, G. Percannella, and M. Vento, "Graph matching and learning in pattern recognition in the last 10 years," *Int. J. Pattern Recog. Artif. Intell.*, vol. 28, no. 1, pp. 1–40, 2014.

[13] L. Gao, J. Song, F. Nie, Y. Yan, N. Sebe, and H. T. Shen, "Optimal graph learning with partial tags and multiple features for image and video annotation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, Jun. 2015, pp. 4371–4379.

[14] M. Guillaumin, J. Verbeek, and C. Schmid, "Multimodal semi-supervised learning for image classification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, Jun. 2010, pp. 902–909.

[15] S. Gupta, J. Kim, K. Grauman, and R. Mooney, "Watch, listen & learn: Co-training on captioned images and videos," in *Proc. Eur. Conf. Mach. Learning Knowl. Discovery Databases*, 2008, pp. 457–472.

[16] L. Huang, X. Liu, B. Ma, and B. Lang, "Online semi-supervised annotation via proxy-based local consistency propagation," *Neurocomputing*, vol. 149, pp. 1573–1586, 2015.

[17] M. S. T. Jaakkola and M. Szummer, "Partially labeled classification with Markov random walks," in *Proc. Conf. Adv. Neural Inform. Process. Syst.*, 2002, vol. 14, pp. 945–952.

[18] T. Joachims, "Transductive inference for text classification using support vector machines," in *Proc. 16th Int. Conf. Mach. Learning*, 1999, pp. 200–209.

[19] S. Kim and S. Choi, "Multi-view anchor graph hashing," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, May 2013, pp. 3123–3127.

[20] P. Li, J. Bu, L. Zhang, and C. Chen, "Graph-based local concept coordinate factorization," *Knowl. Inform. Syst.*, vol. 43, no. 1, pp. 103–126, 2015.

[21] W. Liu, J. He, and S.-F. Chang, "Large graph construction for scalable semi-supervised learning," in *Proc. 27th Int. Conf. Mach. Learning*, 2010, pp. 679–686.

[22] W. Liu, J. Wang, and S.-F. Chang, "Robust and scalable graph-based semisupervised learning," *Proc. IEEE*, vol. 100, no. 9, pp. 2624–2638, Sep. 2012.

[23] W. Liu, J. Wang, S. Kumar, and S.-F. Chang, "Hashing with graphs," in *Proc. 28th Int. Conf. Mach. Learning*, 2011, pp. 1–8.

[24] X. Liu and B. Huet, "Concept detector refinement using social videos," in *Proc. Int. Workshop Very-Large-Scale Multimedia Corpus, Mining Retrieval*, 2010, pp. 19–24.

[25] F. D. Malliaros, V. Megalooikonomou, and C. Faloutsos, "Estimating robustness in large social graphs," *Knowl. Inform. Syst.*, vol. 45, no. 3, pp. 645–678, 2015.

[26] S. Melacci and M. Belkin, "Laplacian support vector machines trained in the primal," *J. Mach. Learning Res.*, vol. 12, pp. 1149–1184, 2011.

[27] B. Mohar, Y. Alavi, G. Chartrand, and O. Oellermann, "The laplacian spectrum of graphs," *Graph Theory, Combinatorics, Appl.*, vol. 2, pp. 871–898, 1991.

[28] Y. Nesterov, *Introductory Lectures on Convex Optimization*. New York, NY, USA: Springer, 2004, vol. 87.

[29] M. Norouzi, A. Punjani, and D. J. Fleet, "Fast exact search in hamming space with multi-index hashing," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 6, pp. 1107–1119, Jun. 2014.

[30] M. Okabe and S. Yamada, "Semisupervised query expansion with minimal feedback," *IEEE Trans. Knowl. Data Eng.*, vol. 19, no. 11, pp. 1585–1589, Nov. 2007.

[31] S. T. Roweis and L. K. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *Science*, vol. 290, no. 5500, pp. 2323–2326, 2000.

[32] J. Song, Y. Yang, Z. Huang, H. T. Shen, and J. Luo, "Effective multiple feature hashing for large-scale near-duplicate video retrieval," *IEEE Trans. Multimedia*, vol. 15, no. 8, pp. 1997–2008, Dec. 2013.

[33] Z. Tian and R. Kuang, "Global linear neighborhoods for efficient label propagation," in *Proc. SIAM Int. Conf. Data Mining*, 2012, pp. 863–872.

[34] I. Triguero, S. García, and F. Herrera, "Self-labeled techniques for semi-supervised learning: Taxonomy, software and empirical study," *Knowl. Inform. Syst.*, vol. 42, no. 2, pp. 245–284, 2015.

[35] R. R. Vatsavai, S. Shekhar, and T. E. Burk, "A semi-supervised learning method for remote sensing data mining," in *Proc. 17th IEEE Int. Conf. Tools Artif. Intell.*, Nov. 2005, pp. 207–211.

[36] D. Wang, F. Nie, and H. Huang, "Large-scale adaptive semi-supervised learning via unified inductive and transductive model," in *Proc. 20th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2014, pp. 482–491.

[37] F. Wang and C. Zhang, "Label propagation through linear neighborhoods," *IEEE Trans. Knowl. Data Eng.*, vol. 20, no. 1, pp. 55–67, Jan. 2008.

[38] J. Wang, J. Wang, G. Zeng, Z. Tu, R. Gan, and S. Li, "Scalable k-NN graph construction for visual descriptors," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, Jun. 2012, pp. 1106–1113.

[39] J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, and Y. Gong, "Locality-constrained linear coding for image classification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, Jun. 2010, pp. 3360–3367.

[40] M. Wang, X.-S. Hua, R. Hong, J. Tang, G.-J. Qi, and Y. Song, "Unified video annotation via multigraph learning," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 19, no. 5, pp. 733–746, May 2009.

[41] Y. Wu, M. Pei, M. Yang, J. Yuan, and Y. Jia, "Robust discriminative tracking via landmark-based label propagation," *IEEE Trans. Image Process.*, vol. 24, no. 5, pp. 1510–1523, May 2015.

[42] Y. Xiong, W. Liu, D. Zhao, and X. Tang, "Face recognition via archetype hull ranking," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2013, pp. 585–592.

[43] B. Xu, J. Bu, C. Chen, C. Wang, D. Cai, and X. He, "EMR: A scalable graph-based ranking model for content-based image retrieval," *IEEE Trans. Knowl. Data Eng.*, vol. 27, no. 1, pp. 102–114, Jan. 2015.

[44] Z. Yang and E. Oja, "Clustering by low-rank doubly stochastic matrix decomposition," in *Proc. 29th Int. Conf. Mach. Learning*, 2012, pp. 831–838.

[45] G. Yu, G. Zhang, Z. Zhang, Z. Yu, and L. Deng, "Semi-supervised classification based on subspace sparse representation," *Knowl. Inform. Syst.*, vol. 43, no. 1, pp. 81–101, 2015.

[46] L. Zelnik-Manor and P. Perona, "Self-tuning spectral clustering," in *Proc. Conf. Adv. Neural Inform. Process. Syst.*, 2004, pp. 1601–1608.

[47] K. Zhang, J. T. Kwok, and B. Parvin, "Prototype vector machine for large scale semi-supervised learning," in *Proc. 26th Annu. Int. Conf. Mach. Learning*, 2009, pp. 1233–1240.

[48] K. Zhang, L. Lan, J. T. Kwok, S. Vucetic, and B. Parvin, "Scaling up graph-based semi-supervised learning via prototype vector machines," *IEEE Trans. Neural Netw. Learning Syst.*, vol. 26, no. 3, pp. 444–457, Mar. 2015.

[49] L. Zhang, Y. Yang, M. Wang, R. Hong, L. Nie, and X. Li, "Detecting densely distributed graph patterns for fine-grained image categorization," *IEEE Trans. Image Process.*, vol. 25, no. 2, pp. 553–565, Feb. 2016.

[50] S. Zhao, H. Yao, Y. Yang, and Y. Zhang, "Affective image retrieval via multi-graph learning," in *Proc. 22nd ACM Int. Conf. Multimedia*, 2014, pp. 1025–1028.

[51] D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Schölkopf, "Learning with local and global consistency," in *Proc. Conf. Adv. Neural Inform. Process. Syst.*, 2004, vol. 16, no. 16, pp. 321–328.

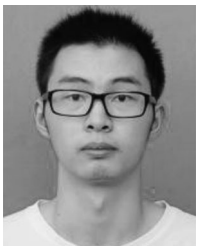
[52] Z.-H. Zhou and M. Li, "Semi-supervised regression with cotraining-style algorithms," *IEEE Trans. Knowl. Data Eng.*, vol. 19, no. 11, pp. 1479–1493, Nov. 2007.

[53] Z. Zhou and M. Li, "Tri-training: Exploiting unlabeled data using three classifiers," *IEEE Trans. Knowl. Data Eng.*, vol. 17, no. 11, pp. 1529–1541, Nov. 2005.

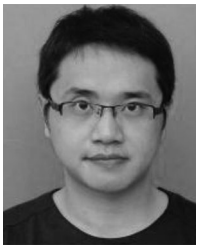
[54] X. Zhu, Z. Ghahramani, J. Lafferty et al.,"Semi-supervised learning using Gaussian fields and harmonic functions," in *Proc. 20th Int. Conf. Mach. Learning*, 2003, vol. 3, pp. 912–919.

[55] X. Zhu and A. B. Goldberg, "Introduction to semi-supervised learning," *Synthesis Lectures Artif. Intell. Mach. Learning*, vol. 3, no. 1, pp. 1–130, 2009.

Meng Wang received the BE and PhD degrees in the special class for the gifted young and the Department of Electronic Engineering and Information Science from the University of Science and Technology of China, Hefei, China, respectively. He is currently a professor at the Hefei University of Technology, Hefei. His current research interests include multimedia content analysis, search, mining, recommendation, and large-scale computing. He received the Best Paper Awards successively from the 17th and 18th ACM International Conference on Multimedia, the Best Paper Award from the 16th International Multimedia Modeling Conference, the Best Paper Award from the 4th International Conference on Internet Multimedia Computing and Service, and the Best Demo Award from the 20th ACM International Conference on Multimedia. He is a member of the IEEE.



Weijie Fu is currently working toward the PhD degree in the School of Computer and Information, Hefei University of Technology, Hefei, China. His current research interest includes machine learning and data mining.



Shijie Hao received the BE, MS, and PhD degrees from the School of Computer and Information, Hefei University of Technology (HFUT), Hefei, China. He is currently a lecturer at HFUT. His current research interests include machine learning and image processing.

Dacheng Tao is currently a professor of computer science with the Centre for Quantum Computation and Intelligent Systems, and the Faculty of Engineering and Information Technology , University of Technology Sydney, Ultimo, NSW, Australia, respectively. He mainly applies statistics and mathematics to data analytics problems and his research interests include computer vision, data science, image processing, machine learning, and video surveillance. His research results have expounded in one monograph and more than 200 publications at prestigious journals and prominent conferences, such as *IEEE T-PAMI, T-NNLS, T-IP, JMLR, IJCV*, NIPS, ICML, CVPR, ICCV, ECCV, AISTATS, ICDM; and ACM SIGKDD, with several Best Paper Awards, such as the Best theory/algorithm Paper Runner Up Award in IEEE ICDM07, the Best Student Paper Award in IEEE ICDM13, and the 2014 ICDM 10-year Highest-Impact Paper Award. He received the 2015 Australian Scopus-Eureka Prize, the 2015 ACS Gold Disruptor Award, and the 2015 UTS Vice-Chancellors Medal for Exceptional Research. He is a fellow of the IEEE, OSA, IAPR, and SPIE.

Xindong Wu received the bachelor's and master's degrees in computer science from the Hefei University of Technology, Hefei (HFUT), Hefei, China, and the PhD degree in artificial intelligence from the University of Edinburgh, Edinburgh, United Kingdom. He is currently a Yangtze River scholar in the School of Computer Science and Information Engineering, HFUT, and a professor of computer science at the University of Vermont, Burlington, VT, USA. His research interests include data mining, knowledge-based systems, and Web information exploration. He is the steering committee chair of the IEEE International Conference on Data Mining, the editor-in-chief of *Knowledge and Information Systems* (*KAIS*, by Springer), and a series editor of the Springer Book Series on Advanced Information and Knowledge Processing (AI&KP). He was the editor-in-chief of the *IEEE Transactions on Knowledge and Data Engineering* (*TKDE*, by the IEEE Computer Society) between 2005 and 2008. He served as the program committee chair for ICDM '03 (the 2003 IEEE International Conference on Data Mining), KDD-07 (the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining), and CIKM 2010 (the 19th ACM Conference on Information and Knowledge Management). He is a fellow of the IEEE.