INVITED PAPER

Robust and Scalable Graph-Based Semisupervised Learning

Graph-based semisupervised learning methods and new techniques for handling contaminated noisy labels, and gigantic data sizes for web applications, are reviewed in this paper.

By Wei Liu, Jun Wang, *Member IEEE*, and Shih-Fu Chang, *Fellow IEEE*

ABSTRACT | Graph-based semisupervised learning (GSSL) provides a promising paradigm for modeling the manifold structures that may exist in massive data sources in high-dimensional spaces. It has been shown effective in propagating a limited amount of initial labels to a large amount of unlabeled data, matching the needs of many emerging applications such as image annotation and information retrieval. In this paper, we provide reviews of several classical GSSL methods and a few promising methods in handling challenging issues often encountered in web-scale applications. First, to successfully incorporate the contaminated noisy labels associated with web data, label diagnosis and tuning techniques applied to GSSL are surveyed. Second, to support scalability to the gigantic scale (millions or billions of samples), recent solutions based on anchor graphs are reviewed. To help researchers pursue new ideas in this area, we also summarize a few popular data sets and software tools publicly available. Important open issues are discussed at the end to stimulate future research.

KEYWORDS | Anchor graphs; graph-based semisupervised learning (GSSL); image annotation; image classification; image search; label diagnosis; large scale; noisy labels

Manuscript received June 25, 2011; revised January 20, 2012; accepted April 20, 2012. Date of publication July 9, 2012; date of current version August 16, 2012.
W. Liu is with the Department of Electrical Engineering, Columbia University, New York, NY 10027 USA (e-mail: wliu@ee.columbia.edu).
J. Wang is with IBM T. J. Watson Research Center, Yorktown Heights, NY 10598 USA (e-mail: wangjun@us.ibm.com).
S.-F. Chang is with the Department of Electrical Engineering and the Department of Computer Science, Columbia University, New York, NY 10027 USA (e-mail: sfchang@ee.columbia.edu).

Digital Object Identifier: 10.1109/JPROC.2012.2197809

I. INTRODUCTION

In the era of data divulgence, there has been broad interest in leveraging a massive amount of data available in open sources such as the Web to help solve long standing problems like object recognition, topic detection, and multimedia information retrieval. One promising direction gaining a lot of attention aims to develop the best ways of combining labeled data (often of limited amount) and a huge pool of unlabeled data in forming abundant training resources for optimizing machine learning models. This learning paradigm is referred to as semisupervised learning (SSL) [63]. Of various SSL methods, graph-based approaches have attracted wide attention due to their good performance and ease of implementation. Graph-based semisupervised learning (GSSL) treats both labeled and unlabeled samples as vertices (nodes) in a graph and builds pairwise edges between these vertices which are weighed by the affinities (similarities) between the corresponding sample pairs. The small portion of vertices carrying seed labels are then harnessed via graph partition or information propagation to predict the labels for the unlabeled vertices. For instance, the graph mincuts approach formulated GSSL as a graph cut problem [3], [4]. Other GSSL methods such as graph transduction formulated GSSL as a regularized function estimation over the graph. These regularized methods optimize the tradeoff between fitting a label prediction function on the labeled samples and a regularization term which encourages smoothness of such a function over the graph. The weighted graph, producing the optimal label prediction function, essentially propagates the initial label information from the labeled samples to the vast amount of unlabeled ones. Many popular GSSL algorithms including graph cuts [3], [4], [25], [31], graph-based random walks [1], [45],

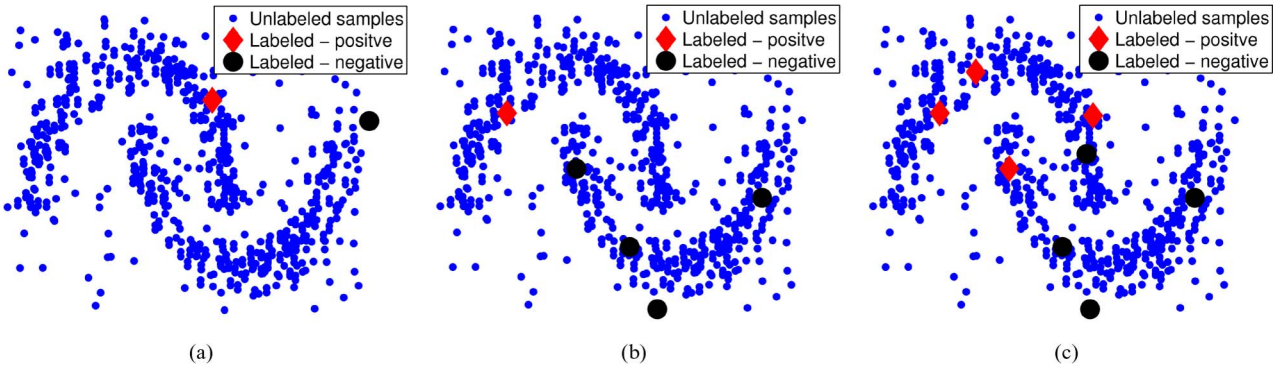

Fig. 1. *Examples illustrating the sensitivity of GSSL to adverse labeling conditions. Particularly challenging conditions are shown in (a) where the only negative label (denoted by a black circle) resides in an outlier region, in (b) where the numbers of labels over different classes are unbalanced, and in (c) where a significant portion in eight given labels is wrong (two out of eight).*

manifold regularization [2], [42], and graph transduction [61], [64] have been proposed. Comprehensive surveys of these methods can be found in [6] and [63].

Despite the promise and popularity of the above GSSL methods, it remains challenging to develop a general solution suitable for processing gigantic data sets such as those from the Web. The most important challenges center around two issues—noisy contaminated labels often seen in open sources and the extremely large data size at the level of hundreds of millions or more. We elaborate on details of these two issues in the following.

A. Contaminated Noisy Labels

Web data are often associated with rich metadata, some of which can be culled to provide useful labels describing the content of the data. One good example is the user provided tags for the images and videos on media sharing forums such as Flickr and Youtube. However, such free labels come at a cost of low accuracy. As discussed in [27], the tag accuracy of web images may be as low as 50% or even less. This problem has been well recognized, as discussed in the review papers [21], [15]. Many prior solutions for this problem usually started with an initial data pool (e.g., top results of a keyword search by image search engines) and subsequently utilized data mining techniques such as random walks [21] or probabilistic latent semantic analysis (pLSA) [15] to discover the hidden patterns and then filter out the noisy labels.

For GSSL, the challenging data conditions mentioned above and other issues discussed below have been shown to result in major performance degradation [52]. Fig. 1 displays that some examples showing noisy labels lead to poor graph transduction results for most of the existing GSSL algorithms. The first ill condition involves uninformative labels [Fig. 1(a)]. In this case, the only negative label (dark circle) is located in an outlier region where the low-density connectivity prevents diffusion of this label to the rest of the graph. As a result, conventional GSSL methods classify the majority of the unlabeled nodes in the graph as positive [52]. Such a condition is very common in content-based image retrieval (CBIR) or related media analysis tasks in which the visual query example does not necessarily fall in the salient region of the target semantic class.

Another difficult case involves imbalanced labeling in which the ratio of the given labels between classes does not match the underlying class proportion. For example, Fig. 1(b) depicts two half-circles with almost equal numbers of samples. However, the initial labels contain three negative labels and only one positive label, so SSL is strongly biased toward the negative class. This imbalanced labeling case also occurs frequently in real-world applications. For example, for image classification [13], [14], [43], typically there are much more negative samples than positive ones. Fig. 1(c) demonstrates another ill condition—mislabeling. It is one of the most frequently encountered cases in practical applications like web image search where tags, captions, and metadata associated with images are often not carefully assigned or verified due to subjective interpretation or content ambiguity.

B. Scalability

The second challenging issue associated with GSSL for web-scale applications is scalability—how to successfully handle millions or billions of data points. Unfortunately, most SSL methods scale poorly with the data size. They usually have a square time complexity $O(dn^2)$ (suppose that data live in $\mathbb{R}^d$) for neighborhood graph construction and a nearly linear time complexity $O(kn)$ (k is a constant) for label propagation over a graph, so the overall time complexity remains as $O(dn^2)$. Such a square time complexity is computationally prohibitive in large-scale applications, preventing the adoption of GSSL methods in practice.

A few solutions have been proposed recently. Delalleu *et al.* [11] proposed a nonparametric inductive function for label prediction based on a subset of samples and aggressive truncation in calculating the graph Laplacian. However, the truncation sacrifices the topology structure within the majority of input data and hence will likely lose useful information of the data set. Zhu and Lafferty [66] fitted a generative mixture model to the raw data and proposed harmonic mixtures to span the label prediction function, but the key step, i.e., constructing a large sparse graph, needed in estimating the harmonic mixtures remains open. Tsang and Kwok [49] scaled up the manifold regularization method first proposed in [2] by solving the dual optimization problem of manifold regularization subject to a sparsity constraint, but such optimization still requires heavy computation (taking $O(1/\epsilon^8)$ time where $\epsilon > 0$ is the approximation factor) to achieve a good approximate solution. Zhang *et al.* [60] applied the Nyström approximation to build a huge graph adjacency matrix, but there is no guarantee for the positive semidefiniteness of the resulting graph Laplacian, which is required to ensure convexity of the optimization problem and convergence of the solution. Fergus *et al.* [16] approximated the label prediction function by exploiting smooth eigenfunctions of 1-D graph Laplacians calculated from each dimension of data, whose derivation relies on several assumptions about the data, e.g., dimension independence and 1-D uniform distributions, which are not true for real-world data.

C. Scope of the Paper

The purpose of this paper is to review a few promising techniques in tackling the two important issues mentioned above which if adequately resolved will facilitate generalization of the popular GSSL methods to many emerging applications involving gigantic data sets. Besides reviews of related methods, we will provide a detailed description of the promising results presented in [53] for noisy label diagnosis and [36] for scalable GSSL using *anchor graphs*. We will also discuss the strategies of combining these two approaches in handling complicated real-world cases which involve both ill conditions.

The structure of this paper is organized as follows. Section II includes reviews of the basic formulation and notations used in GSSL. Reviews of the basic building blocks for GSSL, such as graph construction and label propagation, are shown in Section III. We summarize the label diagnosis and self-tuning (LDST) technique developed in [53] in Section IV, and give an extensive review of the scalable GSSL methods including anchor graphs [36] in Section V. Common data sets and software resources for experiments in this space are presented in Section VI. Finally, open issues are discussed in Section VII. Whenever possible, we focus on the intuitive understanding of the methods and insights learned from prior experiments, using exemplar results to illustrate how the techniques perform. Readers are also referred to the original papers for details of the discussed methods.

II. PROBLEM FORMULATION AND NOTATIONS

We first give the notations of graph representation which will be used throughout this paper. Assume that we are given a data set $\mathcal{X}$, labeled samples $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_l, y_l)\}$, and unlabeled samples $\{\mathbf{x}_{l+1}, \ldots, \mathbf{x}_n\}$. Define the set of labeled inputs as $\mathcal{X}_l = \{\mathbf{x}_1, \ldots, \mathbf{x}_l\}$ with cardinality $|\mathcal{X}_l| = l$ and the set of unlabeled inputs as $\mathcal{X}_u = \{\mathbf{x}_{l+1}, \ldots, \mathbf{x}_n\}$ with cardinality $|\mathcal{X}_u| = u = n - l$, where typically $l \ll n$. The labeled set $\mathcal{X}_l$ is associated with labels $\mathcal{Y}_l = \{y_1, \ldots, y_l\}$, where $y_i \in \{1, \ldots, c\}$ $(i = 1, \ldots, l$ and c is the number of classes). The goal of SSL is to infer the missing labels $\mathcal{Y}_u = \{y_{l+1}, \ldots, y_n\}$ corresponding to the unlabeled set $\mathcal{X}_u$. A crucial component of GSSL is the construction of a weighted sparse graph $\mathcal{G}$ from the whole input data $\mathcal{X} = \mathcal{X}_l \cup \mathcal{X}_u$. After that, GSSL algorithms use $\mathcal{G}$ and the initial seed labels $\mathcal{Y}_l$ to infer $\hat{\mathcal{Y}}_u = \{\hat{y}_{l+1}, \ldots, \hat{y}_n\}$, which are expected to match the true labels $\mathcal{Y}_u$.

Assume that the undirected graph converted from the input data $\mathcal{X}$ is represented by $\mathcal{G} = (\mathcal{X}, E, W)$ in which the set of vertices is $\mathcal{X} = \{\mathbf{x}_i\}$ and the set of edges is $E = \{e_{ij}\}$. Each data point $\mathbf{x}_i$ is treated as a vertex and the weight of edge e_{ij} is W_{ij}. The edge weights $W = \{W_{ij}\}$ are collected to form a weight matrix $\mathbf{W} = (W_{ij})_{i,j}$. Similarly, the vertex degree matrix $\mathbf{D} = \mathrm{diag}([D_{11}, \ldots, D_{nn}])$ is defined by $D_{ii} = \sum_{j=1}^{n} W_{ij}$. For illustration, Fig. 2 gives an example of a graph as well as the corresponding graph quantities.

The graph Laplacian is defined as $= \mathbf{D} - \mathbf{W}$ and its normalized version is $\mathbf{L} = \mathbf{D}^{-1/2}\mathbf{D}^{-1/2}$. The graph

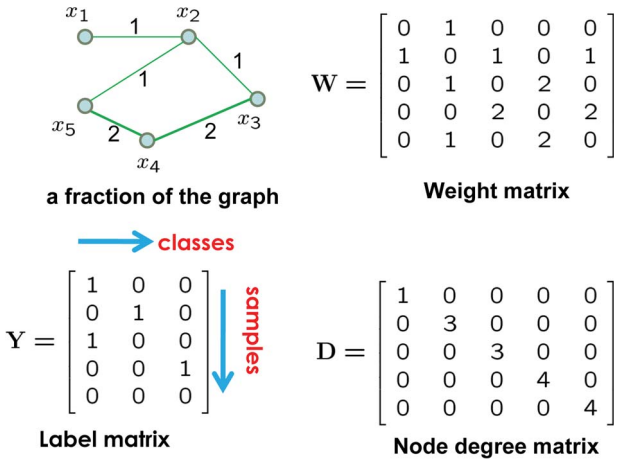

Fig. 2. *An example of a graph with only five vertices, the associated weight matrix* **W**, *the degree matrix* **D**, *and the label matrix* **Y**.

Laplacian can be viewed as a discrete operator on a function space $\{f : \mathcal{X} \mapsto \mathbb{R}^c\}$, which leads to a graph-based semi-inner product [10]

$$\langle f, \mathbf{L}f \rangle_{\mathcal{G}} = \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} W_{ij} \left\| \frac{f(\mathbf{x}_i)}{\sqrt{D_{ii}}} - \frac{f(\mathbf{x}_j)}{\sqrt{D_{jj}}} \right\|^2. \quad (1)$$

Finally, the label information is encoded into a label matrix $\mathbf{Y} \in \{1, 0\}^{n \times c}$, where $Y_{ij} = 1$ if sample $\mathbf{x}_i$ has a label $j \in \{1, \ldots, c\}$, i.e., $y_i = j$, and $Y_{ij} = 0$ otherwise. For single label problems (as opposed to multilabel ones), the constraints $\sum_{j=1}^{c} Y_{ij} = 1$ are imposed. Let $\mathbf{F} = f(\mathcal{X})$ save the output values of a label prediction function f applied to vertices in $\mathcal{X}$. Most GSSL methods utilize the graph quantity $\mathbf{W}$ and the initial discrete label matrix $\mathbf{Y}_l \in \{1, 0\}^{l \times c}$ to discover a continuous soft label matrix $\mathbf{F} \in \mathbb{R}^{n \times c}$ by minimizing a proper cost or penalty over the graph $\mathcal{G}$. The missing labels of the unlabeled vertices $\mathbf{x}_i$ $(i = l+1, \ldots, n)$ are then estimated as $\hat{y}_i = \arg\max_{j \in \{1, \ldots, c\}} F_{ij}$.

III. CLASSICAL GSSL METHODS

As mentioned earlier, graph-based approaches have emerged as popular solutions for many choices since they offer very promising performance improvements via simple and intuitive graph representation. Here, we provide a brief overview of the fundamental components of GSSL, including graph construction and two representative label propagation algorithms.

A. Graph Construction

To fulfill label prediction over a graph, the first step is to convert the input data set $\mathcal{X} = \mathcal{X}_l \cup \mathcal{X}_u$ to a graph $\mathcal{G} = (\mathcal{X}, E, W)$ consisting of n nodes of which each node stands for a sample $\mathbf{x}_i$. Furthermore, take E to be the set of undirected edges between node pairs. It is common to also associate a weighted and symmetric adjacency matrix $\mathbf{W} \in \mathbb{R}^{n \times n}$ with the edges E in $\mathcal{G}$. Each scalar W_{ij} represents the weight of the edge between nodes $\mathbf{x}_i$ and $\mathbf{x}_j$. Note that $\mathbf{W}$ has zeros on its diagonal.

The construction of $\mathcal{G}$ from $\mathcal{X}$ usually includes two steps. The first step is to compute a similarity score between every data pair using a similarity measure which creates a full adjacency matrix $\mathbf{K} \in \mathbb{R}^{n \times n}$. $K_{ij} = K(\mathbf{x}_i, \mathbf{x}_j)$ measures the sample similarity with a kernel function $K(\cdot)$. In the second step, the matrix $\mathbf{K}$ is sparsified and possibly reweighed to produce a sparse matrix $\mathbf{W}$. Sparsification is important since it leads to improved efficiency, better accuracy, and robustness to noise in the label propagation stage. Furthermore, a kernel function $K(\cdot)$ is often locally effective as a similarity measure because it includes unreliable edges between sample pairs that are relatively far apart.

Starting from the dense matrix $\mathbf{K}$, graph sparsification removes edges by finding a binary matrix $\mathbf{B} \in \{1, 0\}^{n \times n}$, where $B_{ij} = 1$ indicates that an edge is present between nodes $\mathbf{x}_i$ and $\mathbf{x}_j$, while $B_{ij} = 0$ indicates that the edge is absent (assume $B_{ii} = 0$ unless explicitly stated otherwise). There are two popular graph sparsification algorithms as summarized in [22]: neighborhood sparsification such as k-nearest neighbors (k-NN) and ϵ-neighborhood, and matching sparsification. The former often suffers from the issue that nodes in dense regions may have too many links. The latter explicitly addresses this issue by adding a constraint that every node has the same number of edge links, leading to much improved performances as reported in [22].

Once a graph has been sparsified and a binary matrix $\mathbf{B}$ has been found, several schemes can be further applied to adjust the weights in the matrix $\mathbf{K}$, generating the final weight matrix $\mathbf{W}$. Specifically, whenever $B_{ij} = 0$ the edge weight W_{ij} is also zero; however, $B_{ij} = 1$ implies $W_{ij} \geq 0$. Three possible schemes, including binary weighting, Gaussian kernel weighting [22], and locally linear reconstruction-based weighting [50], are commonly considered for specifying the nonzero elements in $\mathbf{W}$.

B. Label Propagation

Given a constructed graph $\mathcal{G} = (\mathcal{X}, E, W)$ whose geometric structure has been unveiled by the weight matrix $\mathbf{W}$, the label inference (or prediction) task is to propagate the seed labels $\mathcal{Y}_l$ on the labeled nodes $\mathcal{X}_l$ to all of the unlabeled ones $\mathcal{X}_u$ in $\mathcal{G}$, accomplishing the label predictions $\hat{\mathcal{Y}}_u$. Designing robust label propagation algorithms over graphs is a widely studied subject and the state of the arts are surveyed in [6] and [63].

Here we are particularly interested in a category of approaches which predict the soft labels $\mathbf{F} \in \mathbb{R}^{n \times c}$ through minimizing a cost function defined over the graph. The cost function typically involves a tradeoff between the smoothness of the predicted labels over the entire graph (i.e., consistency of label predictions on closely connected nodes), and the accuracy of the predicted labels in fitting the given hard labels on the labeled nodes $\mathcal{X}_l$. Many existing methods like Gaussian fields and harmonic functions (GFHF) method [64] and local and global consistency (LGC) method [61] fall in this category.

Both LGC and GFHF define a cost function $\mathcal{Q}$ which integrates the combined contributions of two penalty terms: global smoothness $\mathcal{Q}_{\text{smooth}}$ and local fitting $\mathcal{Q}_{\text{fit}}$. The final label predictions $\mathbf{F}$ are obtained by minimizing $\mathcal{Q}$

$$\begin{aligned} \mathbf{F}^* &= \arg\min_{\mathbf{F} \in \mathbb{R}^{n \times c}} \mathcal{Q}(\mathbf{F}) \\ &= \arg\min_{\mathbf{F} \in \mathbb{R}^{n \times c}} \{\mathcal{Q}_{\text{smooth}}(\mathbf{F}) + \mathcal{Q}_{\text{fit}}(\mathbf{F})\}. \end{aligned} \quad (2)$$

A very natural instantiation of the above cost function was proposed by LGC [61] as follows:

$$\mathcal{Q}^{\mathrm{LGC}}(\mathbf{F}) = \frac{1}{2}\left(\|\mathbf{F}\|_{\mathcal{G}}^2 + \mu\|\mathbf{F}-\mathbf{Y}\|^2\right). \quad (3)$$

The first term $\|\mathbf{F}\|_{\mathcal{G}}^2$ represents label smoothness over $\mathcal{G}$, and the second term $\|\mathbf{F}-\mathbf{Y}\|^2$, in which $\mathbf{Y} \in \{1,0\}^{n\times c}$ is the hard label matrix defined in Section II, measures the empirical loss on the given labels. In LGC, the label smoothness term is defined as

$$\mathcal{Q}_{\mathrm{smooth}} = \frac{1}{2}\|\mathbf{F}\|_{\mathcal{G}}^2 = \frac{1}{2}\langle \mathbf{F}, \mathbf{L}\mathbf{F}\rangle_{\mathcal{G}} = \frac{1}{2}\mathrm{tr}(\mathbf{F}^\top \mathbf{L}\mathbf{F}). \quad (4)$$

The coefficient $\mu > 0$ in (3) balances the global smoothness term and the local fitting term. If we set $\mu = \infty$ and use a standard graph Laplacian for the smoothness term, (3) reduces to the GFHF formulation [64]. Its cost function only keeps the smoothness term as follows:

$$\mathcal{Q}^{\mathrm{GFHF}}(\mathbf{F}) = \frac{1}{2}\mathrm{tr}(\mathbf{F}^\top \mathbf{F}). \quad (5)$$

The GFHF solution $\mathbf{F}$ that minimizes the above cost satisfies two conditions

$$\nabla_{\mathbf{F}_u}\mathcal{Q}^{\mathrm{GFHF}} = {}_{uu}\mathbf{F}_u = 0$$
$$\mathbf{F}_l = \mathbf{Y}_l \quad (6)$$

where $\mathbf{F}_l = f(\mathcal{X}_l)$ and $\mathbf{F}_u = f(\mathcal{X}_u)$ are the output values of $f(\cdot)$ on the labeled and unlabeled data, $\mathbf{F} = [\mathbf{F}_l^\top, \mathbf{F}_u^\top]^\top$, and ${}_{uu}$ is the submatrix of corresponding to the unlabeled data set $\mathcal{X}_u$. The first equation above denotes the zero gradient of the cost function on the unlabeled data, and the second equation requires clamping the predicted label values $\mathbf{F}_l$ on the initial label values $\mathbf{Y}_l$.

Both LGC and GFHF fit in a univariate regularization framework, where the soft label matrix $\mathbf{F}$ is treated as the only variable in optimization. Since their cost functions are convex, the optimal solutions to (3) and (5) can both be easily obtained by solving linear systems.

IV. A ROBUST GSSL METHOD FOR HANDLING NOISY LABELS

In order to address various challenging issues of ill labels, Wang *et al.* extended the existing univariate GSSL formulations to a *bivariate* optimization problem over the predicted soft labels and the initial hard labels [52]. Then, a new GSSL method, called graph transduction via alternating minimization (GTAM), was designed to propagate the initial labels, meanwhile performing optimization over both label variables. Note that this bivariate formulation provides the flexibility to manipulate the initially ill labels and thus has the potential to resolve noisy labels. Below we introduce the formulation of bivariate graph transduction and its extension to label tuning called label diagnosis through self-turning (LDST) [53].

A. Bivariate Optimization

The univariate GSSL framework in (2) treats the initial hard labels $\mathbf{Y}$ as "golden" correct labels and propagates them to the unlabeled nodes, but as discussed above this often leads to serious performance degradation when the initial labels are no longer trustable. To handle this problem, a novel bivariate optimization framework is proposed to explicitly optimize both the predicted soft label matrix $\mathbf{F}$ and the initial hard label matrix $\mathbf{Y}$

$$(\mathbf{F}^*, \mathbf{Y}^*) = \arg\min_{\mathbf{F}\in\mathbb{R}^{n\times c}, \mathbf{Y}\in\mathbb{B}^{n\times c}} \mathcal{Q}(\mathbf{F}, \mathbf{Y}) \quad (7)$$

where $\mathbb{B}^{n\times c}$ is the set of all binary matrices $\mathbf{Y}$ in size $n \times c$ that satisfy $\sum_j Y_{ij} = 1$ and $Y_{ij} = 1$ for the labeled node $\mathbf{x}_i \in \mathcal{X}_l$ with $y_i = j$. Note that the solution space of the bivariate optimization framework includes label matrices $\mathbf{Y}$, which may initially contain contaminated noisy labels but will become refined after optimization.

The bivariate cost function used by GTAM [52] is

$$\mathcal{Q}(\mathbf{F}, \mathbf{Y}) = \frac{1}{2}\mathrm{tr}\Big(\mathbf{F}^\top \mathbf{L}\mathbf{F} + \mu(\mathbf{F}-\mathbf{Y})^\top(\mathbf{F}-\mathbf{Y})\Big). \quad (8)$$

Exactly solving both variables $\mathbf{Y}$ and $\mathbf{F}$ is intractable since (7) is a mixed integer programming problem over binary $\mathbf{Y}$ with constraints and continuous $\mathbf{F}$. To address this issue, GTAM first reduces the original bivariate problem to a univariate problem which only retains the variable $\mathbf{Y}$.

Given fixed $\mathbf{Y}$, the cost function in (8) is convex in terms of $\mathbf{F}$, allowing the optimal $\mathbf{F}^*$ to be discovered by simply zeroing the gradient of $\mathcal{Q}$ with respect to $\mathbf{F}$, that is

$$\nabla_{\mathbf{F}^*}\mathcal{Q} = 0 \Longrightarrow \mathbf{F}^* = (\mathbf{L}/\mu + \mathbf{I})^{-1}\mathbf{Y} = \mathbf{P}\mathbf{Y} \quad (9)$$

where $\mathbf{P} = (\mathbf{L}/\mu + \mathbf{I})^{-1}$. Next, replace $\mathbf{F}$ in (8) with its optimal version $\mathbf{F}^*$ in (9), yielding the univariate

optimization problem

$$\begin{aligned}\mathbf{Y}^* = \arg\min \frac{1}{2}\mathrm{tr}(\mathbf{Y}^\top \mathbf{A}\mathbf{Y}) \\ \text{s.t. } \mathbf{Y} \in \{1,0\}^{n\times c} \\ \sum_{j=1}^{c} Y_{ij} = 1 \qquad \forall i \in \{1,\ldots,n\} \\ Y_{ij} = 1 \qquad \forall \mathbf{x}_i \in \mathcal{X}_l \text{ with } y_i = j \end{aligned} \tag{10}$$

where $(1/2)\mathrm{tr}(\mathbf{Y}^\top \mathbf{A}\mathbf{Y}) = \mathcal{Q}(\mathbf{F}^*, \mathbf{Y}) = \mathcal{Q}(\mathbf{P}\mathbf{Y}, \mathbf{Y})$ and $\mathbf{A} = \mathbf{P}\mathbf{L}\mathbf{P} + \mu(\mathbf{P}-\mathbf{I})^2$. Note that there are three groups of constraints in the above optimization problem. The first group produces a binary integer programming problem, the second one makes single class assignment (i.e., each node can only be assigned to one class label), and the third one fixes the initial seed labels in $\mathbf{Y}$.

B. Label Normalization

An approximate yet fast solution to the NP-hard problem in (10) is to use gradient descent to greedily update the binary label variable $\mathbf{Y}$. However, this will raise another practical issue of biased classification since the class with more labels will dominate in each iteration, namely, most label refinements are associated with large classes. To handle the bias issue, a proven effective method is to use a normalized label variable $\tilde{\mathbf{Y}} = \mathbf{V}\mathbf{Y}$ to replace the original variable $\mathbf{Y}$ in the bivariate cost (8), that is

$$\mathcal{Q} = \frac{1}{2}\mathrm{tr}\Big(\mathbf{F}^\top \mathbf{L}\mathbf{F} + \mu(\mathbf{F}-\mathbf{V}\mathbf{Y})^\top(\mathbf{F}-\mathbf{V}\mathbf{Y})\Big). \tag{11}$$

The diagonal matrix $\mathbf{V} = \mathrm{diag}([v_1,\ldots,v_n])$ is introduced as the normalization term to balance the influence of labels from different classes and modulate the label importance based on node degrees. The values v_i $(i = 1,\ldots,n)$ are computed as the class-normalized node degrees for the labeled vertices [52], and thus depend on the current label variable $\mathbf{Y}$. It allows the labeled nodes of higher degrees to contribute more during the label propagation process. However, the total diffusion of each class is kept equal or proportional to the class prior. Therefore, the influence of different classes is balanced even if the initial class labels are unbalanced.

Finally, a locally optimal solution to (11) can be achieved through 1) updating $\mathbf{F}$ as $\mathbf{P}\mathbf{V}\mathbf{Y}$ given fixed $\mathbf{Y}$; and 2) updating $\mathbf{Y}$ via gradient-based greedy search over all $n * c$ binary elements in $\mathbf{Y}$ given fixed $\mathbf{F}$. Since this optimization procedure explicitly minimizes the cost function $\mathcal{Q}$ over two label variables $\mathbf{F}$ and $\mathbf{Y}$ alternatively, it is named as graph transduction via alternating minimization (GTAM) [52].

C. Label Tuning

The GTAM method has achieved much better performance due to its robustness to labels, but it still cannot handle incorrect labels since the initial labels are treated as ground truths, resulting in erroneous label propagation results. In order to handle problematic labels, a bidirectional mechanism for performing label tuning over the graph is proposed, leading to the following approach named label diagnosis through self-tuning (LDST) [53]. While preserving the optimal $\mathbf{F}^*$, LDST explores greedy search among the most beneficial gradient directions of $\mathcal{Q}$ on both labeled and unlabeled nodes.

Note that $\mathbf{Y}$ is constrained to a binary space. Therefore, the labeling operation changes the value from 0 to 1 for a certain element Y_{ij}, and the unlabeling operation (i.e., removing the incorrect label) does the reverse by setting Y_{ij} to 0 from 1. To decrease the cost function $\mathcal{Q}$, one can manipulate the label variable $\mathbf{Y}$ in both directions, labeling and unlabeling. Note that at each time the labeling operation is carried out on one unlabeled node with the minimum element in the gradient $\nabla_{\tilde{\mathbf{Y}}}\mathcal{Q}$, while the unlabeling operation is executed on one labeled node with the maximum element in $\nabla_{\tilde{\mathbf{Y}}}\mathcal{Q}$. The bidirectional gradient search, including both labeling and unlabeling operations, can achieve the steepest decrease on the cost $\mathcal{Q}$. It is described as follows:

$$\begin{aligned} Y_{i^+j^+} &= 1, \qquad \text{for } (i^+,j^+) = \arg\min_{\mathbf{x}_i\in\mathcal{X}_u, 1\le j\le c} \nabla_{\tilde{Y}_{ij}}\mathcal{Q} \\ Y_{i^-j^-} &= 0, \qquad \text{for } (i^-,j^-) = \arg\max_{\mathbf{x}_i\in\mathcal{X}_l, 1\le j\le c} \nabla_{\tilde{Y}_{ij}}\mathcal{Q} \end{aligned} \tag{12}$$

where (i^+,j^+) and (i^-,j^-) are the optimal entries in the label variable $\mathbf{Y}$ for labeling and unlabeling operations, respectively. In other words, through one iteration of bidirectional gradient search, one most reliable label is added and meanwhile one least confident label is removed.

Fig. 3 demonstrates the effect of the LDST approach on a synthetic data set with initially noisy labels. In the first t iterations, a number of unlabeling and labeling operations are executed so as to eliminate the contaminated labels and add new trustable labels. After LDST, GTAM is conducted to propagate the refined labels to the remaining unlabeled nodes in the graph, resulting in much more accurate label prediction results as shown in Fig. 3.

Table 1 shows the effectiveness of LDST on reranking image search results (some example images are displayed in Fig. 4) crawled from the web search engine Flickr. Due to the noisy tags associated with the searched web images, the precision of the initial text search for nine text queries is only about 0.67 in the top-100 reduced set on the average. To apply the graph-based LDST method, the top-1500 returned images from the text search results are included in the graph, in which the top-60 images are treated as the pseudopositive samples. The LDST method

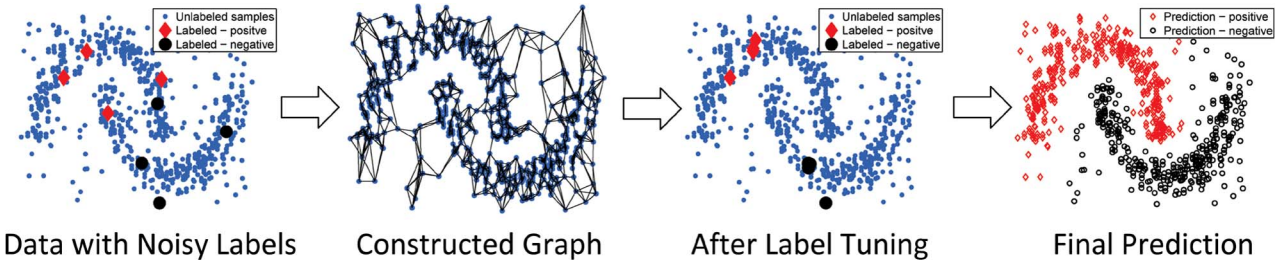

Fig. 3. *The demonstration of the label tuning method on the synthetic two-moon data set with initially noisy labels.*

Table 1 The Mean Precision of Top-100 Ranked Images Over Nine Queries by Different Methods

Method	Text	LGC	GTAM	VisualRank	LDST
Precision	0.6711	0.7989	0.8144	0.7900	**0.8756**

is then performed to diagnose and tune the labels before propagating them to the rest of samples in the graph. As shown in Table 1, the accuracy of top-100 images after label tuning and propagation was significantly improved to about 0.87, much higher than those results obtained by graph-based label propagation without tuning noisy labels (e.g., GTAM or VisualRank [23]).

Note that in the above experiment, the GSSL method is applied as a postprocessing reranking step after a reduced working set is obtained from web search. The size of the working set can be controlled so that the square computational complexity of graphs does not become a bottleneck. In the following section, we will address how to handle gigantic data scale when directly applying GSSL.

V. SCALABLE GSSL METHODS FOR HANDLING GIGANTIC DATA SETS

We address the other key challenge, scalability, in this section. We first review a few works in the recent literature which tried to address large-scale SSL. Specifically, we will summarize the ideas and results presented in the latest scalable GSSL works including our work based on *anchor graphs* [36].

Fig. 4. *Example images of text search results from Flickr.com. A total of nine text queries are searched: dog, tiger, panda, bird, flower, airplane, Forbidden City, Statue of Liberty, and Golden Gate Bridge.*

A. Overview

Almost all SSL methods can be categorized to two families: transductive and inductive. The former aims to infer labels of unlabeled data without developing an explicit classification model, thus lacking the capability of dealing with novel data. The latter takes advantage of both labeled and unlabeled data to train classification models (i.e., inductive models) which can be used to handle unseen data outside the training data set. Consequently, inductive SSL is referred to as *truly* SSL [42]. Several classical GSSL methods including GFHF [64], LGC [61], and GTAM [52] are purely transductive, and other methods such as graph mincuts [3], [3], [31], spectral graph partitioning [25], random walks [1], [45], and local learning [56] also belong to the transductive family since they focus on predicting information associated with the existing unlabeled data. The inductive family consists of transductive support vector machines (SVMs) [24], semisupervised SVMs [7], manifold regularization [2], [42], multikernel SSL [46], translated kernel logistic regression [41], etc. The recent advance in inductive SSL explores relevance ranking [57], structured output learning [39], and multilabel learning [58].

For the scalability issue, most purely transductive methods are not suitable solutions except the blockwise supervised inference method [59], which, however, made a restrictive assumption that the data graph has a block structure. Because of the capability of handling novel data, scalable inductive SSL is more desirable for real-world web-scale data, which usually anticipates dynamic novel data. The scalable GSSL works mentioned in Section I-B, including nonparametric function induction [11], harmonic mixtures [66], sparsified manifold regularization [49], prototype vector machines [60], and eigenfunction [16], are actually inductive. In what follows, we will discuss

Table 2 Summary of Scalable GSSL Methods That Use Efficient Inductive Models. $A(\cdot)$ Is an Affinity Function, $K(\cdot)$ Is a Kernel Function, and $p(\mathcal{C}_k|\mathbf{x})$ Is the Posterior Probability of Sample $\mathbf{x}$ Assigned to Cluster $\mathcal{C}_k$

Inductive Model	Anchors $\{\mathbf{u}_k\}$	Parameters $\{a_k\}$	Weights $\{Z_{ik}\}$
Nonparametric Function Induction [11]	exemplars	labels on anchors	$\frac{A(\mathbf{x}_i,\mathbf{u}_k)}{\sum_{k'=1}^{m} A(\mathbf{x}_i,\mathbf{u}_{k'})}$
Harmonic Mixtures [66]	GMM clustering centers	labels on anchors	$p(\mathcal{C}_k\|\mathbf{x}_i)$
Prototype Vector Machines [60]	K-means clustering centers	classifier coefficients	$K(\mathbf{x}_i,\mathbf{u}_k)$
Anchor Graph Regularization [36]	K-means clustering centers	labels on anchors	$\frac{A(\mathbf{x}_i,\mathbf{u}_k)}{\sum_{k'=1}^{m} A(\mathbf{x}_i,\mathbf{u}_{k'})}$

these works and concentrate on the GSSL method called anchor graph regularization [36] that is not only inductive but also scalable.

The idea of *anchors* is related to the concept of randomly subsampled "beacons" in large-scale network analysis [28], in which node distances are inferred based on the triangulation over the distances between nodes and beacons. In the speech recognition community, the idea of anchor models has also been employed to prune the number of speaker models needed by the applications in speaker detection and speaker indexing on a large database and achieve a promising tradeoff between detection accuracy and computational efficiency [44]. One important distinction between these prior works and the anchor graph model is that the anchor graph model aims at inferring label information of a large number of original data points rather than the minimal-distortion reconstruction of the entire data graph. In addition, the implementation of anchor graphs is very efficient in both storage and computation.

B. Efficient Inductive Models

The representative inductive GSSL method manifold regularization [2] exploits all n training samples to span its inductive model, so classifying a single data point needs a time complexity $O(dn)$ which is inadequate for handling large-scale data. One way to remove this barrier is to modify the traditional inductive model that entangles all training samples, to a more economic inductive model. Such a model utilizes much fewer training samples, so it is much more efficient for label prediction.

Consider a soft label prediction (namely classification) function f: $\mathbb{R}^d \mapsto \mathbb{R}$ defined on the input samples $\mathcal{X} = \{\mathbf{x}_i\}_{i=1}^n$. To accomplish scalable GSSL, a few works [11], [36], [60], [66] have adopted the following inductive model:

$$f(\mathbf{x}_i) = \sum_{k=1}^{m} Z_{ik} a_k, \qquad m \ll n \tag{13}$$

in which a_k's are the model parameters and Z_{ik}'s are the weights specific to each sample $\mathbf{x}_i$. Equation (13) implies that the label of each sample can be interpreted as the weighted combination of a smaller subset (size m) of variables a_k's defined on m anchor points (or landmarks) $\mathcal{U} = \{\mathbf{u}_k\}_{k=1}^m$. By doing so, the solution space $\mathbb{R}^n$ of conventional GSSL is reduced to a much lower space $\mathbb{R}^m$. Consequently, the presented inductive model (13) can substantially alleviate the computational burden required in training full-size inductive models. Table 2 summarizes and compares four efficient inductive models adopting the anchor-based strategy, most of which can reach a training time complexity $O(m^3 + m^2 n)$ and all of which can attain a test time complexity $O(dm)$.

The above model can be written in a matrix form

$$\mathbf{f} = \mathbf{Z}\mathbf{a}, \qquad \mathbf{Z} \in \mathbb{R}^{n\times m}, \qquad m \ll n$$

where $\mathbf{f} = [f(\mathbf{x}_1), \ldots, f(\mathbf{x}_n)]^\top$ and $\mathbf{a} = [a_1, \ldots, a_m]^\top$. As investigated in [36], a good design principle for the weight matrix $\mathbf{Z}$ is $\sum_{k=1}^m Z_{ik} = 1$ and $Z_{ik} \geq 0$, which maintains the unified range of values for predicted soft labels via (13). Besides, the *manifold assumption* [2] implies that close-by data points tend to have similar labels and distant data points are less likely to take similar labels. Thus, a practical step as used in [36] sets $Z_{ik} = 0$ when anchor $\mathbf{u}_k$ is far away from $\mathbf{x}_i$. Then, Z_{ik} is defined based on a kernel function $K_h(\cdot)$ with a bandwidth h

$$Z_{ik} = \frac{K_h(\mathbf{x}_i, \mathbf{u}_k)}{\sum_{k' \in \langle i \rangle} K_h(\mathbf{x}_i, \mathbf{u}_{k'})} \qquad \forall k \in \langle i \rangle \tag{14}$$

where the notation $\langle i \rangle \subset \{1, \ldots, m\}$ is a small integer set saving only the indexes of s nearest anchors of $\mathbf{x}_i$; $Z_{ik} = 0$ for $k \notin \langle i \rangle$. A popular choice of the kernel is the Gaussian kernel $K_h(\mathbf{x}_i, \mathbf{u}_k) = \exp(-\|\mathbf{x}_i - \mathbf{u}_k\|^2/2h^2)$. Other domain-specific kernels may also be used. As a result of keeping a small anchor neighborhood, one can obtain a highly sparse $\mathbf{Z}$ and the involved complexities can also be greatly reduced [$O(sn)$ space and $O(dmn)$ time]. For the inductive purpose, we need to know the weights $Z_{\cdot k}$'s for novel samples. Obeying the same design principle as the matrix $\mathbf{Z}$, we define a nonlinear data-to-anchor mapping $\mathbf{z}(\mathbf{x})$: $\mathbb{R}^d \mapsto \mathbb{R}^m$ as follows:

$$\mathbf{z}(\mathbf{x}) = \frac{[\delta_1 K_h(\mathbf{x}, \mathbf{u}_1), \ldots, \delta_m K_h(\mathbf{x}, \mathbf{u}_m)]^\top}{\sum_{k=1}^m \delta_k K_h(\mathbf{x}, \mathbf{u}_k)} \tag{15}$$

Table 3 Summary of the Properties of Efficient Large Graphs. d Is the Data Dimension and n Is the Data Size

Large Graph	Time Complexity	Sparse $\mathbf{W}$	Nonnegative $\mathbf{W}$	Positive Semidefinite $\mathbf{\Delta}$
Approximate k-NN Graph [8]	$\Theta(dn^t)$ $(1 < t < 2)$	Yes	Yes	Yes
Nyström Graph [60]	$\Theta(dmn)$ $(m \ll n)$	No	No	No
Anchor Graph [36]	$\Theta(dmn)$ $(m \ll n)$	Yes	Yes	Yes

where $\delta_k \in \{1, 0\}$ and $\delta_k = 1$ if and only if anchor $\mathbf{u}_k$ is one of s nearest anchors of sample $\mathbf{x}$ in $\mathcal{U}$. Under this definition $\mathbf{z}(\mathbf{x}_i) \equiv [Z_{i1}, \ldots, Z_{im}]^\top$, the inductive model in (13) can be used to predict the label as

$$f(\mathbf{x}) = \mathbf{z}^\top(\mathbf{x})\mathbf{a} \tag{16}$$

for any sample $\mathbf{x}$ (training or novel test sample) in a universal manner. A kernel-free $\mathbf{z}(\mathbf{x})$ can be learned from the perspective of geometric reconstruction [36] despite longer computation time.

As for the design of anchors, Zhang *et al.* [60] and Liu *et al.* [36] have demonstrated that K-means clustering centers have a stronger representation power to cover the vast data set $\mathcal{X}$ and achieve better SSL performance than randomly sampled exemplars. Although the K-means clustering step incurs additional computation, its time complexity $O(dmnT)$ (T is the iteration number) is relatively manageable compared with the square complexity needed by traditional GSSL. Additionally, some fast implementations [26] for approximate K-means clustering may be used to further mitigate the computational overhead.

C. Efficient Large Graphs

Recall that in most GSSL methods an undirected weighted graph $\mathcal{G} = (\mathcal{X}, E, W)$ is built on n data points in $\mathcal{X} \subset \mathbb{R}^d$. One commonly used graph is the k-NN graph whose time complexity $O(dn^2)$ is infeasible for large-scale applications. Hence, designing efficient large graphs constitutes a major bottleneck of large-scale GSSL.

In the recent literature, construction of large graphs over gigantic data sets has attracted a lot of attention. Chen *et al.* [8] used divide and conquer algorithms to compute an approximate k-NN graph in nearly linear time $\Theta(dn^t)$ $(1 < t < 2)$, but the space complexity could still be large because a KD-tree indexing structure must be saved in memory. Zhang *et al.* [60] applied the Nyström matrix approximation to yield a low rank

$$\mathbf{W} = \mathbf{K}_{nm}\mathbf{K}_{mm}^{-1}\mathbf{K}_{nm}^\top \tag{17}$$

where matrices $\mathbf{K}_{nm}$ and $\mathbf{K}_{mm}$ denote the cross kernel between $\mathcal{X}$ and $\mathcal{U}$, and $\mathcal{U}$ and $\mathcal{U}$, respectively. Although the Nyström graph enjoys a strictly linear time complexity $\Theta(dmn)$, it may yield improper dense graphs that could limit the performance of GSSL. The anchor graph proposed in [36] used a Markov random walk model across data points and anchors to derive a low-rank graph adjacency matrix $\mathbf{W}$ by means of the data-to-anchor mapping matrix $\mathbf{Z}$

$$\mathbf{W} = \mathbf{Z}^{-1}\mathbf{Z}^\top \tag{18}$$

in which the diagonal matrix $= \mathrm{diag}([\Lambda_{11}, \ldots, \Lambda_{mm}]) \in \mathbb{R}^{m\times m}$ is defined by $\Lambda_{kk} = \sum_{i=1}^{n} Z_{ik}$. Because $\mathbf{Z}$ is highly sparse, such an adjacency matrix $\mathbf{W}$ is also empirically sparse (more explanations in [36]).

A very important property stemming from the anchor graph's design strategy is $\mathbf{W} \geq 0$. This nonnegative property is sufficient to guarantee the resulting graph Laplacian $= \mathbf{D} - \mathbf{W}$ to be positive semidefinite [10], and thus ensures global optimum of GSSL. Table 3 summarizes the key properties of efficient large graphs in recent literature.

The central idea of anchor graphs is to introduce a small set of anchor points and convert intensive data-to-data affinity computation to drastically reduced data-to-anchor affinity computation. This is a quite intuitive approach, as shown in Fig. 5. Data pair $\mathbf{x}_i$ and $\mathbf{x}_j$ become connected in the anchor graph if their nearest anchors

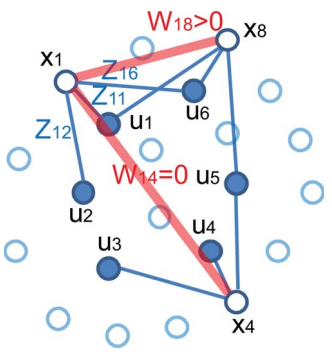

Fig. 5. *The basic idea of anchor graphs. $\{\mathbf{x}_i\}$ are data points. $\{\mathbf{u}_k\}$ are anchor points. Z_{ij}'s are the local supports of anchors for each data point $\mathbf{x}_i$. The affinities between data points are W_{ij}, which can be considered as overlap between the local supports of two data points $\mathbf{x}_i$ and $\mathbf{x}_j$.*

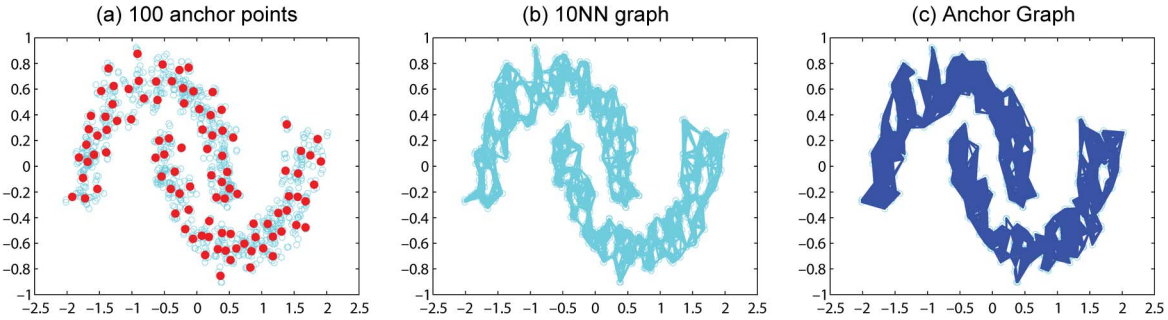

Fig. 6. *The two-moon problem of 1200 2-D points. (a) One hundred anchor points by K-means clustering* ($\mathbf{m} = \mathbf{100}$). *(b) The 10NN graph built on original points. (c) The anchor graph* ($\mathbf{s} = \mathbf{2}$) *leads to a very similar graph.*

overlap. Fig. 6 shows that the anchor graph built on the two-moon toy data is very close to the k-NN graph in topology yet with a much smaller computational cost.

In summary, the graph adjacency matrix $\mathbf{W}$ given by anchor graphs is nonnegative, sparse, and low rank (its rank is at most m). Hence, the anchor graph does not compute $\mathbf{W}$ explicitly, but instead keeps its low-rank form. The space cost of an anchor graph is $\Theta(sn)$ for storing $\mathbf{Z}$, and the time cost is $\Theta(dmnT + dmn)$ in which $\Theta(dmnT)$ originates from K-means clustering. Since $m \ll n$, the time complexity for constructing an anchor graph is linear in the data size n.

D. Anchor Graph Regularization

Anchor graphs can greatly reduce the time complexity of GSSL. To illustrate this, let us consider a standard multiclass SSL setting where each labeled sample $\mathbf{x}_i$ $(i = 1 \ldots, l)$ carries a discrete label $y_i \in \{1, \ldots, c\}$ from c distinct classes. We denote by $\mathbf{Y} = [\mathbf{Y}_l^\top, \mathbf{Y}_u^\top]^\top \in \mathbb{R}^{n \times c}$ a class indicator matrix with $Y_{ij} = 1$ if $y_i = j$ and $Y_{ij} = 0$ otherwise. By utilizing the inductive label prediction model in (16), we only need to solve the soft labels associated with anchors, i.e., $\mathbf{A} = [\mathbf{a}_1, \ldots, \mathbf{a}_c] \in \mathbb{R}^{m \times c}$ in which each column vector accounts for a single class. We introduce the graph Laplacian regularization norm $\Omega_G(\mathbf{f}) = (1/2)\mathbf{f}^\top \mathbf{f}$. For each class j, we pursue a label prediction function $\mathbf{f}_j = \mathbf{Z}\mathbf{a}_j$. Then, the anchor graph regularization (AGR) framework [36] is formulated as follows:

$$\begin{aligned}\min_{\mathbf{A}=[\mathbf{a}_1,\ldots,\mathbf{a}_c]} & \mathcal{Q}^{\mathrm{AGR}}(\mathbf{A}) \\ &= \sum_{j=1}^{c} \Omega_G(\mathbf{Z}\mathbf{a}_j) + \frac{\mu}{2}\sum_{j=1}^{c} \|\mathbf{Z}_l\mathbf{a}_j - \mathbf{Y}_{lj}\|^2 \\ &= \frac{1}{2}\mathrm{tr}(\mathbf{A}^\top \mathbf{Z}^\top \mathbf{Z}\mathbf{A}) + \frac{\mu}{2}\|\mathbf{Z}_l\mathbf{A} - \mathbf{Y}_l\|_{\mathrm{F}}^2 \end{aligned} \tag{19}$$

where $\mathbf{Z}_l \in \mathbb{R}^{l \times m}$ is the submatrix corresponding to the labeled data set $\mathcal{X}_l$, $\mathbf{Y}_{lj}$ represents the jth column vector of the initial label matrix $\mathbf{Y}_l$, $\|.\|_{\mathrm{F}}$ stands for the *Frobenius* norm, and $\mu > 0$ is the tradeoff parameter.

Because of the low-rank structure of $\mathcal{Q}^{\mathrm{AGR}}$, a closed-form solution for the optimal $\mathbf{A}^* = [\mathbf{a}_1^*, \ldots, \mathbf{a}_c^*]$ can be obtained in $O(m^3 + m^2 n)$ time. Through applying the inductive model in (16), we are able to predict the hard label for any sample $\mathbf{x}$ (unlabeled training samples or novel test samples) as

$$\hat{y}(\mathbf{x}) = \arg\max_{j \in \{1,\ldots,c\}} \mathbf{z}^\top(\mathbf{x})\mathbf{a}_j^* \tag{20}$$

and in constant time $O(dm + sc)$.

To sum up, AGR consists of three steps: 1) seek anchors via K-means clustering [$O(dmnT)$ time]; 2) compute $\mathbf{Z}$ [$O(dmn)$ time]; and 3) run label propagation [$O(m^3 + m^2 n)$ time]. In each step, the space complexity is bounded by $O(d(m + n))$. Evaluations conducted on several real-world data sets up to 630 000 samples showed the significant accuracy improvement achieved by AGR. The experimental results on the extended MNIST data set are shown in Table 4 (more results in [36]). Note that although the running time of two AGR approaches shown in Table 4 is longer than the baseline methods 1NN and eigenfunction, they are far more efficient than traditional GSSL methods

Table 4 Classification Error Rates (in Percent) on Unlabeled Training Samples in Extended MNIST (630 000) With $l = 100$ Labeled Samples. AGR-1: Use Predefined $\mathbf{Z}$ [(14)]; AGR-2: Use Learned $\mathbf{Z}$. Both PVM and AGR-1 Use the Gaussian Kernel. All of PVM, AGR-1, and AGR-2 Use $m = 500$ K-Means Clustering Centers as Anchor Points. Both AGR-1 and AGR-2 Set $s = 3$

Method	Error Rate (%)	Running Time (second)
1NN	39.65±1.86	5.46
Eigenfunction [16]	36.94±2.67	44.08
PVM [60]	29.37±2.53	266.89
AGR-1 [36]	**24.71±1.92**	232.37
AGR-2 [36]	**19.75±1.83**	331.72

such as GFHF, LGC, and GTAM whose high time complexity $O(dn^2)$ prevents their practical implementations on large-scale data sets.

E. Adapting AGR to Web Scale

For massive data at the web scale that is at least in the level of billions, the linear time complexity $O(dmnT+m^2n)$ of AGR is inadequate, but we can still benefit from the inductive nature of the anchor graph model, and adapt AGR to cope with web-scale data collections.

To handle the gigantic data scale, we can leverage the "subsampling" idea. Concretely, we construct an anchor graph over a subsampled data subset containing about 1 million samples on which AGR is applied, resulting in the soft label matrix $\mathbf{A}$ associated with the anchors. Aided by the inductive label prediction model in (16) and (20), the anchors' label matrix $\mathbf{A}$ and the data-to-anchor mapping $\mathbf{z}(\cdot)$ defined in (15) can be readily integrated to predict the labels for any novel samples which were not included in the subset used for training the anchor graph model. In other words, we learn an inductive anchor graph model by using a training data set of manageable size (e.g., millions) and subsequently apply the learned model to predict the labels for unseen samples from a much larger pool (e.g., billions). Fig. 7 illustrates the process of adapting the anchor graph model to web-scale scenarios.

We demonstrate the effectiveness of AGR in handling web-scale data by designing a group of inductive classification experiments on the well-known 80 Million Tiny Images database [48]. To evaluate the performance, we draw on the fully labeled image subset CIFAR-10 [30], which is drawn from the 80 Million Tiny Images database and composed of ten object classes. In addition, we subsample 1 million images from the entire 80 million pool, of which we also manually label 3000 images not belonging to the CIFAR-10 classes as "others." The inductive experimental setting is: 1) a training set (1.049 M) includes 50 000 images from ten classes of the CIFAR-10 data set and 999 000 images sampled from the whole 80 million pool (containing 2000 images labeled as "others"); 2) a test set (11 000) includes 10 000 CIFAR-10 images from ten classes and 1000 "others" images; 3) during each SSL trial, the labeled samples in the training set contain 1000 or 2000 randomly chosen images from each of ten CIFAR-10 classes and the "others" class, and the rest of images are treated as the unlabeled samples; and 4) the classification accuracy is measured over the test set. Every image in our experiments is represented by a 384-dimensional GIST feature vector [40]. The reported computational speed is based on implementations on a workstation with a 2.53-GHz Intel Xeon CPU and 48-GB random access memory (RAM).

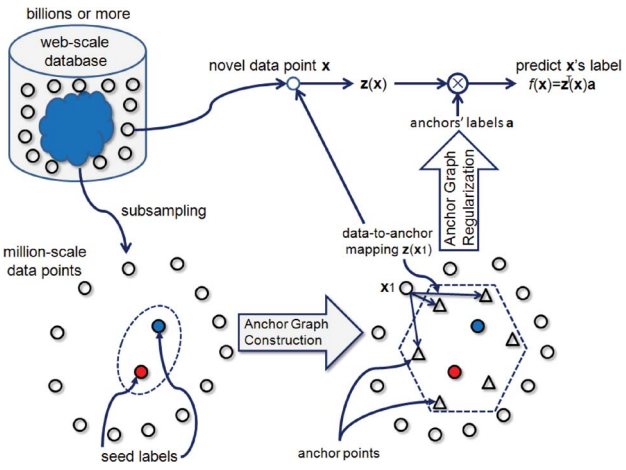

Fig. 7. *Adapting the anchor graph model to web scale through million-scale subsampling and applying the inductive model to novel samples.*

Fig. 8. *Example images from 80 Million Tiny Images database used in our web-scale experiments.*

Fig. 8 shows several example training and test images from ten CIFAR-10 classes and the "others" class. Tables 5 and 6 list the classification accuracy averaged over ten training trials for every compared method. Note that 1NN and linear SVM do not use any unlabeled samples. From Tables 5 and 6, we can see that the AGR approach achieves the highest classification accuracy, while achieving a very fast classification speed during the test stage and a training speed comparable with prior GSSL solutions (e.g., PVM). This web-scale experimental design corroborates that AGR can be well adapted to cope with web-scale data through training anchor graph models over million-scale data sets and then inductively applying them to predict labels for novel data from a larger pool which may contain billions of samples.

Table 5 Classification Accuracy on Web-Scale Data (80 Million Tiny Images plus CIFAR-10). A Subset Containing 1.049 Million Samples With About 1% Samples Labeled Was Used to Train AGR in a Semisupervised Setting. The AGR Model Was Then Applied to Classify Novel Samples Drawn From the Same Web-Scale Data Set. Both PVM and AGR ($s = m/10$) Use the Gaussian Kernel. The Test Time for Each Compared Method Is the Average Classification Time per Test Sample

Method	1NN	Linear SVM	Eigenfunction	PVM (1K anchors)	PVM (2K anchors)	AGR (1K anchors)	AGR (2K anchors)
Accuracy (%)	51.66	60.14	53.86	60.55	60.95	**62.39**	**64.23**
Train Time (sec)	0	8.00	149.83	213.88	517.82	206.60	477.61
Test Time (sec)	6.29e-4	2.66e-6	1.39e-4	5.79e-5	1.27e-4	6.20e-5	1.39e-4

VI. DATA SETS AND SOFTWARE RESOURCES

Many image data sets have been used in the literature to evaluate the performance of SSL. Two popular examples are images of digits in the USPS data set [19] (9298 samples only) and handwritten digits in the MNIST data set [32] (70 000 samples). However, these are relatively small and lack the desired level of complexity expected in web-scale multimedia content.

To simulate the noisy label conditions, one could search the web image search engines using text keywords and download the top several thousands of results. As mentioned in [27], the accuracy of the labels of such data sets could be as low as 50%. The sizes are limited, much less than the million scale discussed above.

In the following, we discuss a few data sets of much larger sizes and content diversity that may be suitable for the large-scale evaluation.

A. The 80 Million Tiny Images Data Set

The 80 Million Tiny Images data set [48] is a large collection of 32×32 tiny images, which has been frequently used as a benchmark data set for a variety of vision and multimedia problems including object recognition, object categorization, image annotation, image retrieval, and hashing. However, only a small portion of this data set is manually labeled and the associated meta information is fairly noisy. The CIFAR-10 data set [30] is a fully annotated 60 000 subset of this image database, which consists of ten customary object classes.

B. NUS-WIDE Data Set

This data set includes 269 648 images and associated tags crawled from the popular social forum Flickr [9]. The images are labeled over 81 concepts which can be used as ground truths for performance evaluation, although some images still do not have complete labels over the categories. The original tags can be used to simulate the noisy label problem discussed in this paper.

C. ImageNet

ImageNet [12] is a database of web images organized according to the concepts (mostly nouns) defined in the WordNet ontology. At the time of writing, it contained more than 12 million images over more than 17 000 synsets. It is a very useful resource for image annotation research. Various SSL settings can be created by controlling the percentage of labels kept in the test set. However, like NUS-WIDE data set, image labels may not be complete, thus treating missing labels as negative will cause errors. In addition, the original contaminated labels are not available.

D. INRIA Web Image Data Set

The INRIA data set [29] contains a total of 71 478 web images acquired by a web search engine using different textual queries. In particular, 353 text queries cover a wide range of concepts, such as objects "cloud," "flag," and "car"; celebrity names "Jack Black," "Will Smith," and "Dustin Hoffman"; and locations "tennis course" and "golf course." Each image has the initial ranking score corresponding to the text query and the ground-truth label indicating whether it is relevant to the query. Hence, this is an excellent resource for evaluating the application of SSL techniques in the context of image retrieval and reranking such as those done in [20].

E. Software Tools

Several popular techniques implementing the GSSL methods reviewed in this paper are available, such as [62]

Table 6 Classification Accuracy on Web-Scale Data (80 Million Tiny Images plus CIFAR-10). A Subset Containing 1.049 Million Samples With About 2% Samples Labeled Was Used to Train AGR in a Semisupervised Setting. The AGR Model Was Then Applied to Classify Novel Samples Drawn From the Same Web-Scale Data Set. Both PVM and AGR ($s = m/10$) Use the Gaussian Kernel. The Test Time for Each Compared Method Is the Average Classification Time per Test Sample

Method	1NN	Linear SVM	Eigenfunction	PVM (1K anchors)	PVM (2K anchors)	AGR (1K anchors)	AGR (2K anchors)
Accuracy (%)	54.52	61.04	54.55	61.76	63.25	**63.84**	**65.94**
Train Time (sec)	0	16.68	154.04	214.74	520.17	207.48	480.12
Test Time (sec)	1.30e-3	2.75e-6	1.48e-4	6.02e-5	1.30e-4	6.33e-5	1.40e-4

for the classical GFHF method. The anchor graph method reviewed in Section V for large-scale GSSL can also be found at [34].

VII. OPEN ISSUES

Despite the exciting results introduced in this paper and other papers in the literature, there are many open issues remaining in the development of GSSL methods and the corresponding large-scale applications such as web-scale multimedia processing. In the following, we discuss a few topics for future work.

A. Integrated Solutions for Handling Gigantic Data Sets With Contaminated Labels

A natural extension of the methods reviewed in this paper is to integrate the scalable methods for graph construction and label propagation (such as the anchor graph) with the approaches for noisy label tuning. Currently, an integrated solution capable of solving both challenges simultaneously is still missing. One option is to use the anchor graph idea to derive the low-rank formulation of the objective function [see (10)] and the corresponding analytic solution based on gradient search [see (12)] for label tuning. Another option is to use the anchor graph to directly propagate the initial noisy labels to the entire large data set, following the process of mapping labels to the anchors and then reversely to the entire graph. In this case, the anchor graph serves as a filtering mechanism and the results of the propagation can be used to measure the "confidence" of the original labels and filter out least confident ones. Both approaches seem reasonable. Deeper investigation and comprehensive evaluation are needed.

B. Advanced Graph Construction Techniques

Most works on GSSL reported in the literature focused on label prediction, and efforts on graph construction have been relatively limited [22], [35]. Majority of the current graph construction methods use certain heuristics in an unsupervised setting. It will be desirable to combine the available label information associated with the given data to design proper graphs in a semisupervised fashion, similar to the way they are used to optimize the label predictions. Liu and Chang [35] tried to learn an adjacency matrix of a symmetry-favored k-NN graph using an SSL setting with some promising performance gains. However, much more work can be pursued in this direction.

C. Graphs of Heterogeneous Features and Relations

Most of the existing GSSL works consider a homogeneous type of nodes and relations. Like the experiments shown in this paper, all nodes correspond to a single type of content (such as images) and all links correspond to a single type of similarity metrics (such as the Gaussian kernel). However, such homogeneity is quite limited compared to the richness of content and relations. For example, in a social media network like Flickr many entities exist, e.g., users, objects, and tags, and many relations are available, such as friendship between users, similarities of content in different feature spaces, and relations between users and content (ownership, viewership, feedback, etc.). Development of new models to capture such heterogeneous data types and relations will greatly enhance the flexibility and representation power of the GSSL techniques. In addition, learning theories and algorithms that fully exploit the heterogeneous data and structures, beyond the initial interesting works in [5] and [33], will have significant impact in this field. Most of these works still consolidate diverse data and relations back to a single homogeneous type in the final inference stage. In [51], multifeature graphs were proposed to build multiple graphs each of which models a distinct feature. However, a joint process optimizing over multiple features is still lacking.

D. Active Learning and GSSL

Both active learning [47] and SSL methods aim to handle the challenging problem of scarce labels. However, well-known active learning methods are mostly performed in a supervised learning setting, such as SVM active learning in which the most informative unlabeled sample is selected to minimize the margin with respect to the current decision boundary. Integration of active learning with GSSL has not been thoroughly exploited, except the early work in [65] where greedy methods were used to minimize the estimated expected classification error (risk) based on a Gaussian random field formulation over graphs. However, scalability and robustness over large-scale data sets remain to be proved. Other active sampling criteria on graphs such as [18] also need to be reexamined.

E. Combining Approximate Nearest Neighbor Search and Graph-Based Approaches

In k-NN graphs, fast methods are needed in order to speed up the process of finding the nearest neighbors of each node. Applying approximate nearest neighbor (ANN) methods like hashing techniques can help achieve rapid construction of sparse graphs over gigantic data sets. Although one can simply apply the well-known hashing methods like [17] in large graph construction, most ANN methods are developed in an unsupervised fashion without taking into account existing label information. Recent works in learning-based hashing methods including unsupervised [55], semisupervised [54], and supervised [37] methods have shown promising improvements, but integration with GSSL has not been investigated and its performance remains to be verified. Along a related but slightly different direction, recent work in [38] extends the anchor graph method described in this paper to develop several graph-based hash code generation mechanisms which capture the data similarities along nonlinear manifolds and better approximate the semantic similarities. ■

REFERENCES

[1] A. Azran, "The rendezvous algorithm: Multiclass semi-supervised learning with Markov random walks," in *Proc. Int. Conf. Mach. Learn.*, 2007, pp. 49–56.

[2] M. Belkin, P. Niyogi, and V. Sindhwani, "Manifold regularization: A geometric framework for learning from labeled and unlabeled examples," *J. Mach. Learn. Res.*, vol. 7, pp. 2399–2434, Nov. 2006.

[3] A. Blum and S. Chawla, "Learning from labeled and unlabeled data using graph mincuts," in *Proc. Int. Conf. Mach. Learn.*, 2001, pp. 19–26.

[4] A. Blum, J. D. Lafferty, M. R. Rwebangira, and R. Reddy, "Semi-supervised learning using randomized mincuts," in *Proc. Int. Conf. Mach. Learn.*, 2004, pp. 13–20.

[5] J. Bu, S. Tan, C. Chen, C. Wang, H. Wu, L. Zhang, and X. He, "Music recommendation by unified hypergraph: Combining social media information and music content," in *Proc. ACM Int. Conf. Multimedia*, 2010, pp. 391–400.

[6] O. Chapelle, B. Schölkopf, and A. Zien, Eds., *Semi-Supervised Learning*. Cambridge, MA: MIT Press, 2006.

[7] O. Chapelle, V. Sindhwani, and S. S. Keerthi, "Optimization techniques for semi-supervised support vector machines," *J. Mach. Learn. Res.*, vol. 9, pp. 203–233, Feb. 2008.

[8] J. Chen, H.-R. Fang, and Y. Saad, "Fast approximate kNN graph construction for high dimensional data via recursive lanczos bisection," *J. Mach. Learn. Res.*, vol. 10, pp. 1989–2012, Sep. 2009.

[9] T.-S. Chua, J. Tang, R. Hong, H. Li, Z. Luo, and Y. Zheng, "NUS-wide: A real-world web image database from National University of Singapore," in *Proc. ACM Int. Conf. Image Video Retrieval*, 2009, DOI: 10.1145/1646396.1646452.

[10] F. Chung, "Spectral graph theory," *CBMS Regional Conference Series in Mathematics*. Providence, RI: AMS, 1997, No. 92.

[11] O. Delalleu, Y. Bengio, and N. L. Roux, "Efficient non-parametric function induction in semi-supervised learning," in *Proc. Int. Workshop Artif. Intell. Stat.*, 2005, pp. 96–103.

[12] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2009, pp. 248–255.

[13] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, *The PASCAL Visual Object Classes Challenge 2008 (VOC2008) Results*. [Online]. Available: http://www.pascal-network.org/challenges/VOC/voc2008/workshop/index.html

[14] L. Fei-Fei, R. Fergus, and P. Perona, "Learning generative visual models from few training examples: An incremental Bayesian approach tested on 101 object categories," *Comput. Vis. Image Understand.*, vol. 106, no. 1, pp. 59–70, Apr. 2007.

[15] R. Fergus, L. Fei-Fei, P. Perona, and A. Zisserman, "Learning object categories from internet image searches," *Proc. IEEE*, vol. 98, no. 8, pp. 1453–1466, Aug. 2010.

[16] R. Fergus, Y. Weiss, and A. Torralba, "Semi-supervised learning in gigantic image collections," *Advances in Neural Information Processing Systems*, vol. 22. Cambridge, MA: MIT Press, 2010, pp. 522–530.

[17] A. Gionis, P. Indyk, and R. Motwani, "Similarity search in high dimensions via hashing," in *Proc. Int. Conf. Very Large Data Bases*, 1999, pp. 518–529.

[18] A. Guillory and J. Bilmes, "Label selection on graphs," *Advances in Neural Information Processing Systems*, vol. 22. Cambridge, MA: MIT Press, 2010, pp. 691–699.

[19] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, 2nd ed. New York: Springer-Verlag, 2009.

[20] W. H. Hsu, L. S. Kennedy, and S.-F. Chang, "Video search reranking via information bottleneck principle," in *Proc. ACM Int. Conf. Multimedia*, 2006, pp. 35–44.

[21] W. H. Hsu, L. S. Kennedy, and S.-F. Chang, "Reranking methods for visual search," *IEEE MultiMedia*, vol. 14, no. 3, pp. 14–22, Jul.–Sep. 2007.

[22] T. Jebara, J. Wang, and S.-F. Chang, "Graph construction and b-matching for semi-supervised learning," in *Proc. Int. Conf. Mach. Learn.*, 2009, pp. 441–448.

[23] Y. Jing and S. Baluja, "Visualrank: Applying pagerank to large-scale image search," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 11, pp. 1877–1890, Nov. 2008.

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

[25] T. Joachims, "Transductive learning via spectral graph partitioning," in *Proc. Int. Conf. Mach. Learn.*, 2003, pp. 290–297.

[26] T. Kanungo, D. M. Mount, N. S. Netanyahu, C. D. Piatko, R. Silverman, and A. Y. Wu, "An efficient k-means clustering algorithm: Analysis and implementation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 7, pp. 881–892, Jul. 2002.

[27] L. S. Kennedy, S.-F. Chang, and I. Kozintsev, "To search or to label?: Predicting the performance of search-based automatic image classifiers," in *Proc. ACM SIGMM Int. Workshop Multimedia Inf. Retrieval*, 2006, pp. 249–258.

[28] J. M. Kleinberg, A. Slivkins, and T. Wexler, "Triangulation and embedding using small sets of beacons," in *Proc. 45th Symp. Found. Comput. Sci.*, 2004, pp. 444–453.

[29] J. Krapac, M. Allan, J. Verbeek, and F. Juried, "Improving web image search results using query-relative classifiers," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2010, pp. 1094–1101.

[30] A. Krizhevsky, "Learning multiple layers of features from tiny images," Univ. Toronto, Toronto, ON, Canada, Tech. Rep., Apr. 2009.

[31] B. Kveton, M. Valko, A. Rahimi, and L. Huang, "Semi-supervised learning with max-margin graph cuts," in *Proc. Int. Conf. Artif. Intell. Stat.*, 2010, pp. 421–428.

[32] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[33] D. Liu, S. Yan, Y. Rui, and H.-J. Zhang, "Unified tag analysis with multi-edge graph," in *Proc. ACM Int. Conf. Multimedia*, 2010, pp. 25–34.

[34] W. Liu. [Online]. Available: http://www.ee.columbia.edu/dvmm/downloads/WeiGraphConstructCode2011/dlform.htm

[35] W. Liu and S.-F. Chang, "Robust multi-class transductive learning with graphs," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2009, pp. 381–388.

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

[37] W. Liu, J. Wang, R. Ji, Y.-G. Jiang, and S.-F. Chang, "Supervised hashing with kernels," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2012.

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

[39] H. Q. Min and V. Sindhwani, "Vector-valued manifold regularization," in *Proc. Int. Conf. Mach. Learn.*, 2011, pp. 57–64.

[40] A. Oliva and A. Torralba, "Modeling the shape of the scene: A holistic representation of the spatial envelope," *Int. J. Comput. Vis.*, vol. 42, no. 3, pp. 145–175, May 2001.

[41] H. A. Sanchez, J. M. Sotoca, and A. M. Uso, "Semi-supervised learning from a translation model between data distributions," in *Proc. Int. Joint Conf. Artif. Intell.*, 2011, pp. 1165–1170.

[42] V. Sindhwani, P. Niyogi, and M. Belkin, "Beyond the point cloud: From transductive to semi-supervised learning," in *Proc. Int. Conf. Mach. Learn.*, 2005, pp. 824–831.

[43] A. F. Smeaton, P. Over, and W. Kraaij, "Evaluation campaigns and TRECVid," in *Proc. ACM SIGMM Int. Workshop Multimedia Inf. Retrieval*, 2006, pp. 321–330.

[44] D. E. Sturim, D. A. Reynolds, E. Singer, and J. P. Campbel, "Speaker indexing in large audio databases using anchor models," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2001, pp. 429–432.

[45] M. Szummer and T. Jaakkola, "Partially labeled classification with Markov random walks," *Advances in Neural Information Processing Systems*, vol. 14. Cambridge, MA: MIT Press, 2002, pp. 945–952.

[46] X. Tian, G. Gasso, and S. Canu, "A multi-kernel framework for inductive semi-supervised learning," in *Proc. 19th Eur. Symp. Artif. Neural Netw.*, 2011, pp. 65–70.

[47] S. Tong and D. Koller, "Support vector machine active learning with applications to text classification," *J. Mach. Learn. Res.*, vol. 2, pp. 45–66, Nov. 2001.

[48] A. Torralba, R. Fergus, and W. T. Freeman, "80 million tiny images: A large dataset for non-parametric object and scene recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 11, pp. 1958–1970, Nov. 2008.

[49] I. W. Tsang and J. T. Kwok, "Large-scale sparsified manifold regularization," *Advances in Neural Information Processing Systems*, vol. 19. Cambridge, MA: MIT Press, 2007, pp. 1401–1408.

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

[51] J. Wang, "Semi-supervised learning for scalable and robust visual search," Ph.D. dissertation, Grad. Schl. Arts Sci., Columbia Univ., New York, NY, 2011.

[52] J. Wang, T. Jebara, and S.-F. Chang, "Graph transduction via alternating minimization," in *Proc. Int. Conf. Mach. Learn.*, 2008, pp. 1144–1151.

[53] J. Wang, Y.-G. Jiang, and S.-F. Chang, "Label diagnosis through self tuning for web image search," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2009, pp. 1390–1397.

[54] J. Wang, S. Kumar, and S.-F. Chang, "Semi-supervised hashing for large scale search," *IEEE Trans. Pattern Anal. Mach. Intell.*, 2012, DOI: 10.1109/TPAMI.2012.48.

[55] Y. Weiss, A. Torralba, and R. Fergus, "Spectral hashing," *Advances in Neural Information Processing Systems*, vol. 21.

Cambridge, MA: MIT Press, 2009, pp. 1753–1760.

[56] M. Wu and B. Schölkopf, "Transductive classification via local learning regularization," in *Proc. Int. Conf. Artif. Intell. Stat.*, 2007, pp. 628–635.

[57] Y. Yang, F. Nie, D. Xu, J. Luo, Y. Zhuang, and Y. Pan, "A multimedia retrieval framework based on semi-supervised ranking and relevance feedback," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 4, pp. 723–742, Apr. 2012.

[58] Y. Yang, F. Wu, F. Nie, H. T. Shen, Y. Zhuang, and A. G. Hauptman, "Web and personal image annotation by mining label correlation with relaxed visual graph embedding," *IEEE Trans. Image Process.*, vol. 21, no. 3, pp. 1339–1351, Mar. 2012.

[59] K. Yu, S. Yu, and V. Tresp, "Blockwise supervised inference on large graphs," in *Proc. ICML Workshop Learning With Partially Classified Training Data*, 2005, pp. 40–46.

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

[61] D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Schölkopf, "Learning with local and global consistency," *Advances in Neural Information Processing Systems*, vol. 16. Cambridge, MA: MIT Press, 2004, pp. 321–328.

[62] X. Zhu. [Online]. Available: http://pages.cs.wisc.edu/~jerryzhu/pub/harmonic_function.m

[63] X. Zhu, "Semi-supervised learning literature survey," Comput. Sci., Univ. Wisconsin-Madison, Madison, WI, Tech. Rep. 1530, 2008.

[64] X. Zhu, Z. Ghahramani, and J. D. Lafferty, "Semi-supervised learning using Gaussian fields and harmonic functions," in *Proc. Int. Conf. Mach. Learn.*, 2003, pp. 912–919.

[65] X. Zhu, J. Lafferty, and Z. Ghahramani, "Combining active learning and semi-supervised learning using Gaussian fields and harmonic functions," in *Proc. ICML Workshop Continuum From Labeled to Unlabeled Data in Machine Learning and Data Mining*, 2003, pp. 58–65.

[66] X. Zhu and J. D. Lafferty, "Harmonic mixtures: Combining mixture models and graph-based methods for inductive and scalable semi-supervised learning," in *Proc. Int. Conf. Mach. Learn.*, 2005, pp. 1052–1059.

ABOUT THE AUTHORS

Wei Liu received the B.S. degree in computer science from Zhejiang University, Hangzhou, China, in 2001 and the M.S. degree in pattern recognition from the Chinese Academy of Sciences, Beijing, China, in 2004. He is currently working toward the Ph.D. degree in the Department of Electrical Engineering, Columbia University, New York, NY.

He worked as an intern at Kodak Research Laboratories and IBM T. J. Watson Research Center in 2010 and 2011, respectively. His research interests include machine learning, computer vision, pattern recognition, and information retrieval.

Mr. Liu won the 2011 Facebook Fellowship and the IBM 2012 Josef Raviv Memorial Postdoctoral Fellowship.

Jun Wang (Member, IEEE) received the M.Phil. and Ph.D. degrees in electrical engineering from Columbia University, New York, NY, in 2010 and 2011, respectively.

Currently, he is a Research Staff Member in the Business Analytics and Mathematical Sciences Department, IBM T. J. Watson Research Center, Yorktown Heights, NY. He also worked as an intern at Google Research in 2009, and as a Research Assistant at Harvard Medical School, Harvard University, Cambridge, MA, in 2006. His research interests include machine learning, business analytics, information retrieval, and hybrid neural-computer vision systems.

Dr. Wang has been the recipient of several awards and scholarships, including the Jury thesis award from the Department of Electrical Engineering at Columbia University in 2011, the Google global intern scholarship in 2009, and a Chinese government scholarship for outstanding self-financed students abroad in 2009.

Shih-Fu Chang (Fellow, IEEE) received the B.S. degree in electrical engineering from the National Taiwan University, Taipei, Taiwan, in 1985 and the M.S. and Ph.D. degrees in electrical engineering and computer sciences from the University of California at Berkeley, Berkeley, in 1991 and 1993, respectively.

He is the Richard Dicker Professor in the Department of Electrical Engineering and the Department of Computer Science, and the Director of Digital Video and Multimedia Lab at Columbia University, New York, NY. He has made significant contributions to multimedia search, visual communication, media forensics, and international standards. He has worked in different advising/consulting capacities for industry research labs and international institutions.

Prof. Chang has been recognized with ACM SIGMM Technical Achievement Award, IEEE Kiyo Tomiyasu Award, Navy ONR Young Investigator Award, IBM Faculty Award, ACM Recognition of Service Award, and NSF CAREER Award. He and his students have received many Best Paper Awards, including the Most Cited Paper of the Decade Award from the *Journal of Visual Communication and Image Representation*. He is a Fellow of the American Association for the Advancement of Science. He served as Editor-in-Chief for the IEEE Signal Processing Magazine (2006–2008), and Chair of Columbia's Electrical Engineering Department (2007–2010).