Automatic image annotation via compact graph based semi-supervised learning

Mingbo Zhao\textsuperscript{a,b}, Tommy W.S. Chow\textsuperscript{b}, Zhao Zhang\textsuperscript{a,*}, Bing Li\textsuperscript{c}

\textsuperscript{a}School of Computer Science and Technology, Soochow University, PR China
\textsuperscript{b}Department of Electronics Engineering, City University of Hong Kong, Hong Kong Special Administrative Region
\textsuperscript{c}School of Economics, Wuhan University of Technology, PR China

\section*{Abstract}

The insufficiency of labeled samples is major problem in automatic image annotation. However, unlabeled samples are readily available and abundant. Hence, semi-supervised learning methods, which utilize partly labeled samples and a large amount of unlabeled samples, have attracted increased attention in the field of image annotation. During the past decade, graph-based semi-supervised learning has been becoming one of the most important research areas in semi-supervised learning. In this paper, we propose a novel and effective graph based semi-supervised learning method for image annotation. The new method is derived by a compact graph that can well grasp the manifold structure. In addition, we theoretically prove that the proposed semi-supervised learning method can be analyzed under a regularized framework. It can also be easily extended to deal with out-of-sample data. Simulation results show that the proposed method can achieve better performance compared with other state-of-the-art graph based semi-supervised learning methods.

\section*{1. Introduction}

In the real world, there are ever-increasing vision image data generated from Internet surfing and daily social communication. These metadata can be labeled or unlabeled, and accordingly be utilized for image retrieval, summarization, and indexing. In order to handle these datasets for realizing the above tasks, automatic annotation is an elementary step, which can be formulated as a pattern classification problem and accomplished by learning-based techniques. Traditionally, the learning-based methods can be categorized into two categories: (1) supervised learning, which aims to predict the labels of new-coming data samples from the observed labeled set, i.e., to handle the classification problem or to preserve the discriminative information which is embedded in the training set. Typical supervised learning methods include Support Vector Machines (SVM) [36,37], and Linear Discriminant Analysis (LDA) and its variants [1–3,17,18]; and (2) the other one is unsupervised learning, the goal of which is to handle the observed data with no labels and to grasp the intrinsic structure of the dataset. Typical unsupervised learning methods include clustering [19–22] and manifold learning methods, such as ISOMAP [23], Locally Linear Embedding (LLE) [24], and Laplacian Eigenmap (LE) [25]. In this paper, we mainly focus on the classification problem, which is traditionally a supervised learning task.

In order to handle the pattern classification problem, such as image annotation, the conventional supervised learning methods, such as Linear Discriminant Analysis (LDA) [1–3] and Support Vector Machine (SVM) [36,37], cannot deliver satisfactory classification accuracy when the number of labeled samples is not sufficient. However, labeling a large number of samples is time-consuming and costly. On the other hand, unlabeled samples are abundant and can be easily obtained in the real world. Hence, semi-supervised learning methods (SSL), which incorporate partly labeled samples and a large amount of unlabeled samples into learning, have become more effective than only relying on supervised learning. Recently, based on clustering and manifold assumptions, i.e., nearby samples (or samples of the same cluster or data manifold) are likely to share the same label [4–6], graph based semi-supervised learning methods have received considerable research interest in the area of semi-supervised learning. Typical methods include Manifold Regularization (MR) [15], Semi-supervised Discriminant Analysis (SDA) [16], Gaussian Fields and Harmonic Functions (GFHF) [4], Learning with Local and Global Consistency (LLGC) [5], and General Graph based Semi-supervised
Learning method (GGSSL) [7,8]. All of these methods represent both labeled and unlabeled sets by a graph, and then utilize its graph Laplacian matrix to characterize the manifold structure [25,26].

In general, the abovementioned graph based semi-supervised learning can be divided into two categories: inductive learning methods and transductive learning methods. The inductive learning methods, such as MR [15] and SDA [16], try to induce a decision function that has a low classification error rate on the whole data space; while the transductive learning methods, also known as Label Propagation, aim to directly predict the label information from the labeled set to the unlabeled set along the graph, which is much easier to handle and less complicated than inductive learning methods. Two well-known transductive learning methods are GHF [4] and LLGC [5]. GHF has an elegant probabilistic explanation, and the output labels are the probabilistic values; however, it cannot detect outliers in data. In contrast, LLGC can detect outliers; however, its output labels are not probabilistic values. Both the problems in GHF and LLGC have been eliminated by GGSSL [7,8], in which it can either detect the outliers or develop a mechanism to calculate the probabilities of data samples.

It should be noted that one important step of graph based SSL is to construct a graph with weights for characterizing the data structure. The graph is usually used to find the neighbors by k-neighborhood or ε-neighborhood in the whole data [23–26], and then to define the weight matrix on the graph. There are commonly two ways to define the weight matrix: one is to apply the Gaussian function [4,5,7,8,15,16] and the other is to employ the local linear reconstruction strategy [13,14]. The Gaussian function is easily manipulated in many graph based SSL, but estimating the optimal variance in the Gaussian function is very difficult [13]. The locally linear reconstruction strategy has no such problem, as it is based on the assumption that each sample can be reconstructed by a linear combination of its neighborhoods. The weight matrix is then automatically calculated when the neighborhood size is fixed. However, as pointed out in [10], using the neighborhoods of a sample to reconstruct it may not achieve the minimum result, which may not well capture the manifold structure of the dataset.

In this paper, motivated by the framework of GGSSL [7,8], we present an effective semi-supervised learning method, namely, Compact Graph based Semi-supervised Learning (CGSSL), for image annotation, which is based on a newly proposed compact graph. The newly proposed graph finds the neighbors of each sample and calculates the graph weights in a way as [13,14]. However, since the minimum reconstruction error of a sample may not be obtained by its own neighborhood, it aims to reconstruct the sample by using the neighbors of its adjacent samples and then preserve the graph weights corresponding to the minimum error. In this way, a more compact graph can be constructed, which can well capture the manifold structure of the dataset. In addition, in order to establish the connection to the normalized graph, we further symmetrize and normalize this compact graph. With these processes, the proposed CGSSL can be theoretically analyzed from the perspective of a graph, through which we show that the proposed CGSSL can be derived from a smoothness regularized framework. The proposed CGSSL can also be easily extended to its inductive out-of-sample version for handling new-coming data by using the same smoothness criterion. Finally, extensive simulations on image annotation and content based image retrieval show the effectiveness of the proposed CGSSL.

The main contributions of this paper are as follows: (1) we propose a new compact local reconstruction graph with symmetrization and normalization for semi-supervised learning. The new graph construction strategy can find a more compact way to approximate a sample with its neighborhoods, which can better grasp the manifold structure embedded in the dataset. In addition, with symmetrization and normalization processes, the proposed CGSSL can be analyzed theoretically under a regularized framework; (2) the proposed CGSSL is a transductive learning method, and it can also be easily extended to its inductive out-of-sample version for handling new-coming data by using the same smoothness criterion; (3) we analyze the relationships between the proposed CGSSL and other state-of-the-art graph based semi-supervised learning in terms of objectives, parameters and out-of-sample extensions, which are helpful for better understanding graph-based semi-supervised learning methods. Moreover, extensive simulations based on image annotation and content based image retrieval have verified the effectiveness of the proposed method; and (4) we further analyze the proposed CGSSL and other state-of-the-art methods from the time-to-process point of view and give an explicit implementation choice. Simulation results regarding computational time show that the proposed CGSSL can be more suitable and practical for handling large-scale image annotation tasks.

The rest of this paper is organized as follows: In Section 2, we will provide some notations and present the proposed CGSSL; in Section 3, we will give detailed analysis and out-of-sample extensions for the proposed CGSSL; extensive simulations are conducted in Section 4, and final conclusions are drawn in Section 5.

2. The proposed semi-supervised learning method

Let \( X = \{x_i, x_u, x_o\} \subset R^{d\times(l+u)} \) be the data matrix, where \( d \) is the number of data features, and the first \( l \) and the remaining \( u \) samples in \( X \) represent the labeled set \( x_l \) and unlabeled set \( x_u \), respectively. Each sample in \( X_l \) is associated with a class label \( c_i \), \( i \in \{1, 2, \ldots, c\} \), where \( c \) is the number of classes. The goal of graph based semi-supervised learning methods is to propagate the label information of the labeled set to the unlabeled set according to the distribution associated with both the labeled and unlabeled set [4,5], and through which the predicted labels of the unlabeled set, called soft labels, can be obtained.

2.1. Review of graph construction

In label propagation, a similarity matrix must be defined for evaluating the similarities between any two samples. The similarity matrix can be approximated by a neighborhood graph associated with weights on the edges. Officially, let \( G = (V, E) \) denote this graph, where \( V \) is the vertex set of \( G \) representing the training samples, and \( E \) is the edge set of \( G \) associated with a weight matrix \( W \) containing the local information between two nearby samples. There are many ways to define the weight matrix. A typical way is to use the Gaussian function [4,5,7,8,15,16]:

\[
    w_{ij} = \exp \left( -\frac{|x_i - x_j|^2}{2\sigma^2} \right) \quad x_i \in N_k(x_j) \quad \text{or} \quad x_j \in N_k(x_i),
\]

where \( N_k(x_i) \) is the k neighborhood set of \( x_i \), and \( \sigma \) is the Gaussian function variance. However, \( \sigma \) is hard to be determined, and even a small variation of \( \sigma \) can alter the results dramatically [13]. Wang et al. have proposed another strategy to construct \( G \) by using the neighborhood information of samples [13,14]. This strategy assumes that each sample can be reconstructed by a linear combination of its neighborhoods [24], i.e., \( x_i \approx \sum_{j \in N_k(x_i)} w_{ij} x_j \). It then calculates the weight matrix by solving a standard quadratic programming (QP) problem as:

\[
    \min_{F} \left\| x_i - \sum_{j \in N_k(x_i)} w_{ij} x_j \right\|^2_F \quad \text{s.t.} \quad w_{ij} \geq 0, \quad \sum_{j \in N_k(x_i)} w_{ij} = 1.
\]

The above strategy is empirically better than the Gaussian function, as the weight matrix can be automatically calculated in a closed
form once the neighborhood size is fixed. However, as pointed out in [10], using the neighborhoods of a sample to reconstruct it may not achieve a compact result. We use Fig. 1 to elaborate this fact, in which we generalize eight samples in \( R^2 \). Then, we in this next subsection, propose a more effective strategy for graph construction.

Taking \( x_1 \) as an example and letting \( k = 5 \), we first reconstruct \( x_1 \) by using its neighborhoods, i.e., \{\( x_1, x_2, x_3, x_4, x_5 \)\}. Following Eq. (2), we have \( x_1 = 0.1694x_5 + 0.2491x_6 + 0.4109x_8 + 0.1706x_2 \). In this case, the reconstruction error is \( ||x_1 - \bar{x}_1||_2^2 = 0.26852 \). Note that \( x_1 \) is also in the neighborhoods of \( x_6 \), i.e., \{\( x_6, x_1, x_4, x_5, x_8 \)\}. Hence, if we use them, i.e., \{\( x_6, x_4, x_5, x_8 \)\}, to reconstruct \( x_1 \) as \( \bar{x}_1 = 0.1694x_6 + 0.2491x_4 + 0.4109x_8 + 0.1706x_5 \), the error can be \( 0.04631 < 0.26852 \). This indicates that using the neighborhoods of \( x_6 \) to reconstruct \( x_1 \) is better than that using the neighborhoods of its own, as the former reconstruction error is much smaller.

### 2.2. The proposed graph construction

The above analysis motivates us to propose an improved local reconstruction strategy for graph construction. Since the minimum reconstruction error of a sample may not be obtained by its own neighborhood, we need to search in its adjacent samples and find the minimum error. This can be a more compact way to reconstruct each sample. In practice, we first generate a vector \( \bar{e} = [e_1, e_2, \ldots, e_5] \in \mathbb{R}^{1 \times 5} \) to preserve the minimum errors of samples. We then search each sample \( x_i \) and its neighborhood set \( N_j \) : \{\( x_j, x_{j+1}, x_{j+2} \)\} (including itself). For each \( x_j, i = 1 \) to \( n \), we use other samples in \( N_j \) to reconstruct it. If the reconstruction error \( \bar{e}_i \) is smaller than \( e_i \), replace \( e_i \) with \( \bar{e}_i \) and preserve the reconstruction weights of \( x_j \) into \( W \). The basic steps of the proposed strategy can be shown in Table 1. Fig. 2 shows the improvement of the proposed graph construction, in which we generalize a two-cycle dataset.

### 2.3. Symmetrization and normalization of graph weight

Note that the improved local reconstruction weight \( W \) obtained by Table 1 does not satisfy the symmetric property, i.e., \( W \neq W^T \), which means that it does not have a close relationship to the normalized graph theorem. Although this drawback can be overcome by the processes, i.e., \( W \rightarrow (W + W^T)/2 \) and \( D^{-1/2}WD^{-1/2} \) [7], the weight matrix still holds inhomogeneous degrees. In our work, to handle this problem, we design the proposed weight matrix as follows:

\[
\bar{W} = W \Delta^{-1}W^T. \tag{3}
\]

where \( \Delta \in \mathbb{R}^{(l+u) \times (l+u)} \) is a diagonal matrix with each element \( \Delta_{ij} \) being the sum of the \( j \)th row in \( W \), i.e., \( \Delta_{ij} = \sum_{i=1}^{l+u} W_{ij} \). It can be easily verified that \( W \) is symmetric, and the sum of each row or column of \( W \) is equal to 1; hence, \( W \) is also normalized. As pointed in [7], the normalization can strengthen the weights in the low-density region and weaken the weights in the high density region, which is advantageous for handling the case in which the density of datasets varies dramatically. Following Eq. (3), it can be noted that by neglecting the diagonal matrix \( \Delta \), \( WW^T \) is an inner product with each element \( \langle WW^T \rangle \), measuring the similarity between each pair-wise local reconstructed vector \( w_i \) and \( w_j \). Given that \( x_i \) and \( x_j \) are close to each other and in the same manifold, their corresponding \( w_i \) and \( w_j \) tend to be similar, and \( \langle WW^T \rangle \) will be a large value; otherwise, \( \langle WW^T \rangle \) will be equal to 0, if \( x_i \) and \( x_j \) do not share any neighborhoods. The weight matrix defined in Eq. (3) can also be easily extended to out-of-sample data by using the same smoothness criterion. We will discuss this issue in Section 3.2.

### 2.4. Label propagation process

We will then predict the labels of unlabeled samples based on a General Graph based Semi-supervised Learning process (GGSSL) [7,8]. Let \( Y = [y_1, y_2, \ldots, y_{l+u}] \in \mathbb{R}^{l+u \times (l+u)} \) be the initial labels of all samples, for the labeled sample \( x_j, y_{ji} = 1 \) if \( x_j \) belongs to the \( i \)th class; otherwise, \( y_{ji} = 0 \). For the unlabeled sample \( x_j, y_{ji} = 1 \) if

<table>
<thead>
<tr>
<th>Table 1</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>An effective algorithm for calculating reconstructed weight.</strong></td>
</tr>
<tr>
<td><strong>Input:</strong></td>
</tr>
<tr>
<td>Data matrix ( X \in \mathbb{R}^{(l+u) \times (l+u)} ), neighborhood number ( k ).</td>
</tr>
<tr>
<td><strong>Output:</strong></td>
</tr>
<tr>
<td>Weight matrix ( W = [x_1, x_2, \ldots, x_{l+u}] \in \mathbb{R}^{1 \times (l+u)} ).</td>
</tr>
<tr>
<td><strong>Algorithm:</strong></td>
</tr>
<tr>
<td>1. Generate an error vector ( e = [e_1, e_2, \ldots, e_{l+u}] \in \mathbb{R}^{1 \times (l+u)} ) with each element ( e_i = +\infty ) and initialize ( W ) as a zero matrix.</td>
</tr>
<tr>
<td>2. for each sample ( x_j ) ( j = 1 ) to ( t = u ), do</td>
</tr>
<tr>
<td>3. Identify the ( k ) neighborhood set as: ( N_j : x_j, x_{j+1}, x_{j+2} ).</td>
</tr>
<tr>
<td>4. for each sample ( x_j ) ( i = 1 ) to ( k ), do</td>
</tr>
<tr>
<td>5. Reconstruct ( x_j \approx \bar{x}<em>j = \sum</em>{i=1}^{l+u} W_{ij} \bar{x}_i ) according to Eq. (2).</td>
</tr>
<tr>
<td>6. if the reconstructed error ( \bar{e}_j =</td>
</tr>
<tr>
<td>7. ( \bar{e}_j =</td>
</tr>
<tr>
<td>8. end for</td>
</tr>
<tr>
<td>9. end for</td>
</tr>
<tr>
<td>10. output weight matrix ( W ).</td>
</tr>
</tbody>
</table>

![Fig. 1](image-url) Two ways of reconstructing \( x_1 \): seven data points with coordinates as: \( x_1(3,3), x_2(4,1), x_3(5,4), x_4(4,5), x_5(4,2), x_6(2,4), x_7(3,1), x_8(1,2) \). (a) Reconstruct \( x_1 \) using the neighbors of \( x_1 \) and (b) reconstruct \( x_1 \) using the neighbors of \( x_6 \).
is set to belonging to the outliers. The basic steps for the proposed are row vectors satisfying $0$ for the labeled data is the Frobenius norm of $\mathbf{F} + \mathbf{f}$ to balance $\lambda$ in the simulations. In this paper, $\mathbf{F}$ can be seen as the posterior probability of $x_i$ and the label information received represents the probabilization parameter $\mathbf{a}$ from its neighbors during the iteration. According to [7], the regularization parameter $g_j$ for the labeled data $x_j$ is set to $a_j$; for the unlabeled sample $x_j$, it is set to $x_j$ in the simulations. In this paper, we simply set $g_j = 0$ for the labeled sample, which means we constrain $\mathbf{F} = \mathbf{Y}$, and set the value of $g_j$ for the unlabeled sample by using a fivefold cross validation. By the iterative process in Eq. (4), we have:

$$F(t + 1) = F(t) \overline{\mathbf{W}} + \mathbf{Y} f_j. \quad (4)$$

where $I_i \in \mathbb{R}^{(k+1) \times (k+u)}$ is a diagonal matrix with each element being $x_j, I_j = I - x_j$, and $a_j (0 \leq a_j \leq 1)$ is a parameter for $x_j$ to balance the initial label information of $x_j$ and the label information received from its neighbors during the iteration. According to [7], the regularization parameter $g_j$ for the labeled data $x_j$ is set to $g_j$; for the unlabeled sample $x_j$, it is set to $x_j$ in the simulations. In this paper, we simply set $g_j = 0$ for the labeled sample, which means we constrain $\mathbf{F} = \mathbf{Y}$, and set the value of $g_j$ for the unlabeled sample by using a fivefold cross validation. By the iterative process in Eq. (4), we have:

$$F(t) = F(0)(\overline{\mathbf{W}})^t + \mathbf{Y} \sum_{k=0}^{t-1} (\overline{\mathbf{W}})^k. \quad (5)$$

Based on the properties of matrix, i.e., $\lim_{t \to \infty} (\overline{\mathbf{W}})^t = 0$ and $\lim_{t \to \infty} \sum_{k=0}^{t-1} (\overline{\mathbf{W}})^k = (I - \overline{\mathbf{W}}^{-1})$, the iterative process of Eq. (5) converges to:

$$\lim_{t \to \infty} F(t) = \mathbf{Y} (I - \overline{\mathbf{W}})^{-1}. \quad (6)$$

It can be easily proven that the sum of each column of $R$ is equal to 1 (see Appendix). This indicates that the elements in $R$ are probability values, and $f_y$ can be seen as the posterior probability of $x_j$ belonging to the $i$th class; when $i = c + 1, f_y$ represents the probability of $x_j$ belonging to the outliers. The basic steps for the proposed label propagation process can be seen in Table 2. In addition, since our proposed SSL method is based on a compact local reconstruction graph with normalization, we refer to our SSL method as Compact Graph based Semi-Supervised Learning (CGSSL).

Here, a simple example of the proposed label propagation process with outlier detection can be seen in Fig. 3. However, in many real-world applications, there may exist noisy labeled samples due to the carelessness of the annotator, and our method can also automatically erase such noisy labeled samples by setting $x_j > 0$. A case in point is given in Fig. 4, where we generalize a two-moon dataset with incorrectly labeled samples.

With the proposed CGSSL, we can directly use it for automatic image annotation. Specifically, given an image dataset, users only need to annotate a small part of data samples as the labeled set, while keep the remaining data samples as the unlabeled set. Then, by taking both the labeled and unlabeled set as inputs, the proposed CGSSL can automatically estimate the labels of the unlabeled set based on a label propagation process and output them to the users. In other words, the class labels of the unlabeled set can be automatically annotated by the proposed CGSSL, hence realizing automatic image annotation.

3. Analysis and extensions of the compact semi-supervised learning method

3.1. Regularization interpretation

In this subsection, we will analyze the proposed CGSSL from the perspective of a graph, through which we show that the proposed CGSSL can be derived in a regularized framework. Specifically, let $\text{Tr} (\cdot)$ be the trace operator and $\| \cdot \|_F$ is the Frobenius norm of matrix, i.e., $\| \mathbf{M} \|_F^2 = \text{Tr}(\mathbf{M}^T \mathbf{M})$ [38]; then, we have the following theorem:

<table>
<thead>
<tr>
<th>Table 2</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Compact graph based semi-supervised learning.</strong></td>
</tr>
<tr>
<td><strong>Input:</strong> Data matrix $X \in \mathbb{R}^{k \times (k+u)}$, label matrix $Y \in \mathbb{R}^{c(k+1) \times (k+u)}$, the number of nearest neighbor $k$ and other relative parameters.</td>
</tr>
<tr>
<td><strong>Output:</strong> The predicted label matrix $\mathbf{F} \in \mathbb{R}^{c(k+1) \times (k+u)}$.</td>
</tr>
<tr>
<td><strong>Algorithm:</strong></td>
</tr>
<tr>
<td>1. Construct the neighborhood graph and calculate the weight matrix $W$ as Table 1.</td>
</tr>
<tr>
<td>2. Symmetrize and normalize $W$ as $\overline{W} = W^{-1/2} W^1$ in Eq. (3), where $I$ is the diagonal matrix satisfying $I_k = \sum_{i=1}^{k} W_{ii}$.</td>
</tr>
<tr>
<td>3. Calculate the predicted label matrix $F-y(I-\overline{W})^{-1}$ in Eq. (6) and output $F$.</td>
</tr>
</tbody>
</table>
The predicted result of NCSSL in Eq. (6), i.e., \( F = \frac{1}{2} (I - P_L)^{-1} \), is equivalent to the optimal solution that minimizes the following objective function:

\[
J(F) = \frac{1}{2} \text{Tr}(F \Delta F^T) + \text{Tr}(F - Y)(U(F - Y))^T.
\]

where \( \Delta = I - W \) is the normalized graph Laplacian matrix, and \( U \) is a diagonal matrix satisfying \( U_{ij} = \mu_j \). By setting the derivation of \( J(F) \) w.r.t. \( F \) to zero, we have:

\[
\frac{\partial J(F)}{\partial F} = 2 \Delta F + 2U(F - Y) = 0.
\]

Let us introduce a set of variables, i.e., \( x_i = 1/(1 + \mu_i) \), \( i \in \{1, 2, \ldots, I + U\} \), or \( I_x = (I + U)^{-1} \); then, the optimal solution of Eqs. (7) and (8) can be derived as:

\[
F^* = \text{YU}(\bar{I} - \bar{U})^{-1} = \text{YU}(I - P - U)^{-1} = \text{YU}(I + U)^{-1}(I - \bar{W} + \bar{U})^{-1} = \text{YI}(I - \bar{W}L_s)^{-1}.
\]

The second equation holds as \( \bar{I} = I - \bar{W} \), and the fourth equation holds as \( I_f = I - I_x = U(I + U)^{-1} \). Hence, following Eq. (10), we can see that the optimal solution in Eq. (10) is equivalent to the predicted labels in Eq. (6).

### 3.2. Out-of-sample extension

In general, the proposed SSL is a transductive method, which is an off-line model process. However, it cannot deal with out-of-sample data, i.e., it cannot predict the fault or normal condition for a new-coming scene. In this subsection, we will apply the local approximation strategy to extend SSL to the out-of-sample data. Specifically, for new-coming test data \( z \), since we know nothing about its label information, we can treat it as an unlabeled sample and utilize the same smoothness constraint in Eq. (2) as a criterion for predicting the label of \( z \). In addition, the inclusion of \( z \) should not affect the prediction results of the original training data samples. Following the work in [22], we first give the smoothness criterion for the new-coming sample \( z \) as:

\[
J(f(z)) = \sum_{j \in X, \bar{X}, \bar{N}_z} \bar{w}(z, x_j) ||f(z) - f_j||^2.
\]

where \( N_z \) is the neighborhood set of \( z \), \( \bar{w}(z, x_j) \) is the similarity between \( z \) and \( x_j \), and \( f(z) \) is the predicted label of \( z \). Note that the nearest neighbors of \( z \), i.e., \( N_z \), are search in \( z \cup X \), and the weight matrix \( \bar{w}(z, x_j) \) satisfies the sum-to-one constraint, i.e., \( \sum_{j \in X, \bar{X}, \bar{N}_z} \bar{w}(z, x_j) = 1 \). Since \( J(f(z)) \) is convex in \( f(z) \), it is minimized by setting the derivation of \( J(f(z)) \) in Eq. (11) w.r.t. \( f(z) \) to zero. Then, the optimal \( f(z) \) can be given as follows:

\[
f(z) = \sum_{j \in X, \bar{X}, \bar{N}_z} \bar{w}(z, x_j)f_j.
\]
Here, we define the weight matrix following the same criterion as Eq. (3). Specifically, we first calculate $w(z, x_i)$ by the same strategy following Eq. (2), which is to minimize the following objective function:

$$J(f(z)) = \left\| f(z) - \sum_{j \in N(z)} w(z, x_j) f_j \right\|^2_F \quad \text{s.t.} \quad w(z, x_j) \geq 0, \quad \sum_{j \in N(z)} w(z, x_j) = 1. \quad (13)$$

We then extend $w(z)$ following the same form of Eq. (3), i.e.:

$$\hat{w}(z) = w(z) A^{-1} W^T, \quad (14)$$

Here, it can be easily verified that $\sum_{j \in N(z)} w(z, x_j) \hat{w}(z, x_j) = 1$; then, following Eq. (12), the sum of the column in $f(z)$ is equal to 1, which indicates that each element in $f(z)$ can be seen as the probabilistic value. The basic steps for calculating the predicted label of new-coming sample $z$ is given in Table 3.

Two toy examples for evaluating the effectiveness of the proposed out-of-sample extension can be seen in Figs. 5 and 6, where we generalize a two-moon dataset and two-Swiss-roll dataset. In Fig. 5, we select two samples per class as the labeled set and the remaining as the unlabeled set; whereas, in Fig. 6, we only choose one sample per class as the labeled set. In the two examples, we first adopt the proposed labeled propagation method to predict the labels of the unlabeled data samples. We then use the strategy in Table 3 to induce the labels of all samples in the region $\{(x, y)|x \in [-2, 2], y \in [-2, 2]\}$ for the two-moon dataset and $\{(x, y)|x \in [-8, 8], y \in [-8, 8]\}$ for the two-Swiss-roll dataset. The simulation results are also given in Figs. 5a and b and 6a and b, from which we can observe that the induced decision boundary and region are intuitively satisfying, as they are in accordance with the intrinsic structure of the training dataset.

3.3. Discussion and related work

In this subsection, we will briefly discuss the relationships between the proposed CGSSL and other state-of-the-art graph based semi-supervised learning methods (see Fig. 6).

In general, based on the manifold and cluster assumption [4-6]; (1) nearby points are likely to have the same label and (2) points on the same structure (such as a cluster or a sub-manifold) are likely to have the same label, the graph based semi-supervised learning first employs a graph $G$ to model the whole dataset, and then estimates a classification function $f$ on $G$, under the following constraints: (1) it should be close to the given label assignments and (2) it should be smooth on the whole graph $G$. These two constraints are often characterized in a regularization framework. For integrity, we formalize the graph based semi-supervised learning framework in this subsection, which is helpful to build the relationships between the proposed NGSSL and other methods. Specifically, similar to Eqs. (7) and (8), we first give the following regularization objective function to be optimized:

$$J(F) = \alpha C(F, Y) + \beta S(F). \quad (15)$$

where $C(\cdot)$ is a loss function to measure the inconsistency between the predicted and initial labels on $X_s$, and $S(\cdot)$ is a regularized term punishing the smoothness of $F$ on the whole graph and is approximated by the following discrete form:

$$S(F) = \text{Tr}(FSF^T), \quad (16)$$

where $S \in \mathbb{R}^{(1+u)(1+u)}$ is the smoothness matrix. In fact, many traditional graph-based semi-supervised methods are similar to each other, with different choices of loss functions and regularized terms. Table 4 gives some examples of those methods, such as GFHF, LLGC, CGSSL, LNP, Manifold Regularization (MR), Semi-supervised Kernel Density Estimation (SSKDE) [11,12], and the proposed CGSSL. Here, $L$ represents the combinatorial graph Laplacian [4], and $I$ represents the normalized graph Laplacian [5]. They are both derived from the weight matrix $W$. For MR, $K$ represents the induced kernel of $f$ in the reproducing kernel Hilbert space, and $\gamma_1$ and $\gamma_2$ are two parameters balancing the tradeoff between two terms. For LNP and the proposed CGSSL, LLRW represents the Local Linear Reconstruction Weight.
3.4. Analysis computational complexity

In this subsection, we will analyze the computational complexities of different methods. Note that for most graph based methods, such as MR, GFFH, LLGC, CGSSL, LNP and the proposed CGSSL, one needs to construct a k nearest neighborhood graph, and the computational complexity is $O((l + u)^2k)$. For GFFH, LLGC and CGSSL, following the work in [8], the computational complexity for label propagation is $O((l + u)kc)$, where $k$ and $c$ are the numbers of neighborhoods and class, respectively. Hence, the total computational complexity of GFFH, LLGC, and CGSSL is $O((l + u)^2k + (l + u)kc)$. For LNP and the proposed CGSSL, one needs to additionally calculate the reconstructed weight matrix by solving standard QP problems in Eq. (2), where the computational complexities are $O((l + u)^2l^3)$ and $O((l + u)^2k^3)$ for LNP and CGSSL, respectively. Note that the computational complexity of CGSSL in this step is $k$ times that of LNP. This is reasonable, as CGSSL is to calculate the reconstructed weight matrix by searching all neighborhoods of the dataset for the minimum reconstructed error. Thus, the total computational complexities of LNP and CGSSL are $O((l + u)^2k + (l + u)k^3 + (l + u)kc)$ and $O((l + u)^2k + (l + u)k^3 + (l + u)kc)$, respectively. For MR, one needs to calculate the inverse of the graph Laplacian matrix and the computational complexity is $O(D^2)$, where $D$ is the dimensionality of the dataset. Hence, the total computational complexity of MR is $O((l + u)^2k + D^2)$. For SSKDE, since one does not need to construct a $k$ nearest neighborhood graph, its total computational complexity is only from the kernel density estimation process, and the computational complexity is $O((l + u)^2c)$. The comparisons of different methods are shown in Table 5.

From Table 5, we have the following observations: (1) the computational complexity of MR is closely related to the number of dimensionality, while that of SSKDE is related the number of datasets and (2) for other graph based methods, we can see that the computational complexities of LNP and the proposed CGSSL are larger than those of GFFH, LLGC, and CGSSL. This can be reasonable, as LNP and CGSSL need to additionally calculate the reconstructed weight matrix by solving standard QP problems. However, since GFFH, LLGC and CGSSL use the Gaussian function to calculate the weight matrix, it should be necessary to spend some computational time for parameter selection in order to find the best Gaussian covariance.

4. Simulation

In this section, we evaluate the proposed methods with two synthetic datasets, several UCI and real-world datasets. For the
synthetic datasets, we evaluate the proposed method using one-Swiss-roll and two-cycle datasets. For other datasets, we focus on solving the image annotation problems based on seven UCI datasets and five real-world datasets, which are all benchmark datasets. Furthermore, we compare the proposed GCSSL with state-of-the-art SSL methods. In the comparative study, we randomly split each dataset into a labeled and unlabeled set.

4.1. Toy examples for sensitivity analysis

We first use two toy examples to compare the proposed CGSSL with the Euclidean distance-based method and GGSSL, where we set $k$ as 8. In the first toy example, we compare the annotated performance between the proposed CGSSL and Euclidean distance on one-Swiss-roll dataset, where the density of this dataset is dense in the core, while it is sparse in the surrounding area (see Fig. 7a). In this dataset, we only annotate one sample in the core of one-Swiss-roll, as the labeled set and the remaining samples as unlabeled samples. The simulation results of the annotation can be seen in Fig. 7b and c. For CGSSL, the marker size of each unlabeled data point is proportional to its predicted label; whereas, for Euclidean distance, the reciprocal of the distance between a given data point and the annotated point is used to evaluate the annotated results. From Fig. 7b and c, we can observe that the distance-based method fails to preserve the one-Swiss-roll structure. The reason for this is that the distance-based method only considers pair-wise distance, but ignores the whole data distribution. In contrast, the proposed CGSSL can well annotate the label of data points even if they are far from the annotated point, which indicates that the proposed CGSSL can better handle datasets with a complicated manifold structure.

In the second toy example, we compare the annotated performance of the proposed CGSSL and GGSSL in terms of robustness to the parameters on the two-cycle dataset with two classes, and each follows a cycle distribution. In each class, two samples are annotated as the labeled set and the remaining as unlabeled samples. The simulation results of annotation for GGSSL and CGSSL can be seen in Figs. 8 and 9. From Fig. 8, we can observe that the performance of GGSSL is quite sensitive to the Gaussian covariance $\sigma$ in the Gaussian function. For example, when we set $\sigma = \bar{\sigma}/10$, where $\bar{\sigma}$ is the half of the average distance of all pair-wise neighborhood samples, the annotated results for GGSSL can be seen in Figs. 8 and 9. From Fig. 8, we can observe that the performance of GGSSL is quite sensitive to the Gaussian covariance $\sigma$ in the Gaussian function. For example, when we set $\sigma = \bar{\sigma}/10$, where $\bar{\sigma}$ is the half of the average distance of all pair-wise neighborhood samples, the annotated results for GGSSL is fairly good. However, when we increase $\sigma$ to $\sigma = \bar{\sigma}$ or $\sigma = \bar{\sigma}/0.1$, the annotated performance will become quite poor, which indicates that

![Fig. 7. The label propagation process for annotation: one-Swiss-roll dataset (a) initial dataset with varied-density distribution (one labeled sample); (b) Euclidean Distance; and (c) CGSSL.](image)

![Fig. 8. The label propagation process with sensitivity analysis: two-cycle dataset (two labels per class) (a)–(c) the performances of GGSSL with different Gaussian variance, (b) $\sigma = \bar{\sigma}/0.1$, (c) $\sigma = \bar{\sigma}$ and (d) $\sigma = \bar{\sigma}/10$, where $\bar{\sigma}$ is half of the average distance of all pair-wise neighborhood samples.](image)
even a small variation of \( \sigma \) can make the results dramatically different. However, the proposed CGSSL does not have such a parameter. In addition, from Fig. 9, we can observe that the proposed CGSSL works well in a wide range of \( \mu_s \) from 10 to \( 10^5 \), which indicates that the proposed CGSSL is more robust to parameters. Clearly, a method that is less sensitive to the parameters is more suitable for real-world annotation tasks.

### 4.2. Image annotation

For solving the image annotation problem, we use six UCI and six real-world datasets to evaluate the performance of the methods. The UCI datasets are available at [http://archive.ics.uci.edu/ml](http://archive.ics.uci.edu/ml) and the real-world datasets include COIL100 [27], ETH80 [28], USPS [29], MNIST [30], and Extended Yale-B dataset [31]. The COIL100 dataset consists of images of 100 objects viewed from nine poses and 64 illumination conditions. In our simulation, the images are cropped and resized to 32 \( \times \) 32 pixels. The MNIST dataset contains 9298 gray-scale handwritten digit images scanned from envelopes by the U.S. Postal Service, with an image size of 28 \( \times \) 28 and transfer them to 256 grayscale levels. The USPS dataset contains 9298 gray-scale handwritten digit images scanned from envelopes by the U.S. Postal Service, with an image size of 16 \( \times \) 16 in 256 grayscale levels. The ETH80 dataset contains 80 objects, with each object represented by 41 views of images. The original size of each image is 128 \( \times \) 128, with 24 bit color levels per pixel. Similarly, we down-sample the size of images to 32 \( \times \) 32 and transfer each image to 256 gray levels.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Type</th>
<th>#Samples</th>
<th>#Features</th>
<th>#Class</th>
<th>Balance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bupa</td>
<td>UCI</td>
<td>345</td>
<td>6</td>
<td>2</td>
<td>0.725</td>
</tr>
<tr>
<td>Ionosphere</td>
<td>UCI</td>
<td>351</td>
<td>34</td>
<td>2</td>
<td>0.56</td>
</tr>
<tr>
<td>Heart-Statlog</td>
<td>UCI</td>
<td>270</td>
<td>13</td>
<td>2</td>
<td>0.8</td>
</tr>
<tr>
<td>Iris</td>
<td>UCI</td>
<td>150</td>
<td>4</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>Wine</td>
<td>UCI</td>
<td>178</td>
<td>13</td>
<td>1</td>
<td>0.6761</td>
</tr>
<tr>
<td>Segmentation</td>
<td>UCI</td>
<td>2100</td>
<td>19</td>
<td>7</td>
<td>1</td>
</tr>
<tr>
<td>Digit</td>
<td>UCI</td>
<td>5620</td>
<td>64</td>
<td>10</td>
<td>0.9508</td>
</tr>
<tr>
<td>COIL100 [27]</td>
<td>Object</td>
<td>7200</td>
<td>1024</td>
<td>100</td>
<td>1</td>
</tr>
<tr>
<td>ETH80 [28]</td>
<td>Object</td>
<td>3280</td>
<td>1024</td>
<td>80</td>
<td>1</td>
</tr>
<tr>
<td>USPS [29]</td>
<td>Hand-written digit</td>
<td>9298</td>
<td>256</td>
<td>10</td>
<td>0.4559</td>
</tr>
<tr>
<td>MNIST [30]</td>
<td>Hand-written digit</td>
<td>10,000</td>
<td>784</td>
<td>10</td>
<td>1</td>
</tr>
<tr>
<td>Extended Yale-B  [31]</td>
<td>Face</td>
<td>16,123</td>
<td>1024</td>
<td>38</td>
<td>1</td>
</tr>
</tbody>
</table>

### Table 6

Dataset information for each dataset ("Balance" is defined as the ratio between the number of samples in the smallest class and the number of samples in the largest class).
For MR, the Laplacian regularized Least Squares (LapRLS) is chosen in our simulations, which is a special case of MR. Its implementation code is downloaded from the website of http://manifold.cs.uchicago.edu/manifold_regularization/software.html. Here, the Gaussian variance in MR is set using the same strategy as Eq. (17), and the extrinsic and intrinsic regularization parameters are searched from \( \{10^{-6}, 10^{-3}, 10^{-1}, 1, 10, 10^2, 10^3\} \). In addition, for simplicity, we adopt the linear kernel in MR, i.e., \( f(x_j) = W^T x_j \), where \( W \) is the projection matrix and can provide a simple model to map the samples.

The average results over 20 random splits of different methods with varied numbers of labeled samples are shown in Fig. 11. From the simulation results, we have the following observations: (1) for all of the methods, the annotated accuracies can be higher when the number of labeled samples increases. For example, the accuracy for the proposed CGSSL will increase by about 15% when the percentage of labeled samples varies from 5% to 50% for most UCI datasets. This increase can even achieve 20% in the Extended Yale-B dataset when the number of labeled samples increases from 2 to 20 per class. This indicates that the labeled samples are actually helpful for classification. However, we also observe that, in most cases, the accuracy of different methods will not increase any more when a sufficient number of labeled samples can be given; (2) all semi-supervised learning methods outperform supervised learning methods, such as 1NN, by about 4–20%. For example, given few labeled samples, the proposed CGSSL can even achieve 20% improved accuracy to 1NN in most real-world datasets. This indicates that by incorporating unlabeled samples into learning, classification accuracy can be greatly improved and semi-supervised learning methods generally outperform conventional supervised methods, such as 1NN, if there are inefficient labeled samples; (3) among all semi-supervised learning methods, the proposed CGSSL can achieve almost the best performance in real-world datasets and some of the UCI datasets, i.e., CGSSL can provide higher accuracy than other semi-supervised methods due to the compact graph construction as analyzed in Section 2. For example, CGSSL can achieve 1–3% improvement over LNP and 5–6% improvement over GFHF, LLGC and SSKDE, respectively, in most UCI and real-world datasets. This superiority can even achieve 8% compared to GFHF and LLGC in the Bupa dataset. Another observation is that GGSSL can achieve competitive results to the proposed CGSSL in most real-world datasets. However, it should be noted that the result of GGSSL is obtained by best tuning the parameters, while the proposed CGSSL can easily construct the compact graph and obtain the results when the neighborhood size is fixed; (4) compared with inductive methods, such as MR, we can observe that MR can achieve competitive results to CGSSL in most UCI datasets and the Extended Yale-B dataset, but is not superior to other semi-supervised methods in real-world applications. The main reason for this is that real-world datasets (except for Extended Yale-B) have more complex manifold structures than UCI datasets; hence, the graph constructed by transductive learning methods, such as the proposed CGSSL, can well capture such manifold structure of datasets, and the labels can be correctly propagated in these cases. However, for the Extended Yale-B dataset, CGSSL and other transductive learning methods are not superior to MR, possibly because of the strong lighting variations of images. Hence, in this case, the labels may not be properly propagated, which will degrade the performance of CGSSL and other transductive learning methods; and (5) we also compare the performance between the proposed CGSSL and SSKDE, which is a semi-supervised extension to Kernel Density Estimation based on statistical models. From the simulations, we can see that the proposed CGSSL, GGSSL, and LLGC generally outperform SSKDE in most real-world datasets, as they based on normalized graph Laplacian, while SSKDE is based on graph Laplacian. For example, in COIL100 and ETH80 datasets, the proposed CGSSL can achieve approximately 5–6% improvement over SSKDE, which generally confirms its superiority compared to SSKDE.

It should also be pointed out that the proposed CGSSL cannot achieve better performance in some UCI and Extended Yale-B datasets. The reasons for this can be as follows: (1) for some UCI datasets, such as Iris, Wine and Segmentation datasets, we observe that these datasets are usually with low dimensionality and their data structure is very simple, which means that the manifold structure of these datasets may not be very clear and their patterns are easy quite to classify. Therefore, the superiority of CGSSL is not demonstrated, and the proposed CGSSL can only achieve competitive, or not much better, performance compared with other state-of-the-art graph based semi-supervised learning methods; and (2) for the Extended Yale-B face dataset, we observe that it contains the images of 38 human subjects, with each human subject having 64 different lighting variations. Because of strong lighting variations of face images, there may exist cases in which two pair-wise face images with the same lighting variation, but in different human subjects, can be closer; whereas, two pair-wise face images with different lighting variations, but actually in the same human subject, may be far away. This indicates that using the
conventional neighborhood graph cannot well capture the correct manifold structure of the dataset, as we expect the images within the same human subject to be closer. As a result, the labels from the labeled set cannot be properly propagated to the unlabeled set, causing the performances of CGSSL and other graph based learning methods to be degraded.

4.3. Comparison between transductive learning and inductive out-of-sample extension

In this subsection, we compare the performance between the transductive method in Table 2 and inductive out-of-sample extension in Table 3 for the proposed CGSSL. In this study, we conduct the simulations for different real-world datasets with varied numbers of labeled samples. Specifically, we randomly select 20% of samples from each dataset as a testing set and the remaining samples as a training set. We then split the training set into two sets, i.e., the labeled set and the unlabeled set. Next, we compare the transductive version of CGSSL with the inductive extension of CGSSL. Since the transductive CGSSL cannot deal with testing samples, for comparison, we regard testing samples as unlabeled and integrate them for training.

The average results over 20 random splits for different datasets with varied numbers of labeled samples are shown in Fig. 12. From the figure, we can see that the inductive out-of-sample extension method performs better than the transductive method in most cases. This is because the inductive method considers the entire dataset, including both labeled and unlabeled samples, which can provide more information for learning.

Fig. 11. The average accuracy over 20 random splits for different datasets: (a) Bupa; (b) Ionosphere; (c) Heart-Statlog; (d) Wine; (e) Segmentation; (f) Iris; (g) COIL100; (h) ETH80; (i) USPS; (j) MNIST; (k) Digit; and (l) Extended Yale-B.
the simulation results, we have the following observations: (1) similar to the transductive CGSSL, the accuracies obtained by inductive extension of CGSSL increase when the number of labeled samples increases. For example, the accuracy for the inductive extension of CGSSL will increase by about 20–25% when the number of labeled samples varies from 1 to 20 for ETH and Extended Yale-B datasets; (2) the accuracies obtained by transductive CGSSL are slightly better than those obtained by inductive extension of CGSSL. This is natural, as we have considered the testing samples as unlabeled samples when we perform transductive CGSSL. However, the results of the inductive extension of CGSSL are satisfied and believable, which demonstrate the effectiveness of inductive extension when handling the new-coming samples.

4.4. Confidence analysis of the parameters

In this subsection, we analyze the confidence intervals of the results in order to further show the robustness of the proposed method. Specifically, the proposed CGSSL does not have the parameter of Gaussian covariance (that is in GFHF, LLGC, and GGSSL); thus, in this revised version, we only need to provide the confidence intervals of $a_l$ and $a_u$ which are used in Eq. (4). The average annotated accuracies over 20 random splits of six different datasets with varied $a_l$ and $a_u$ are shown in Fig. 13, where the candidate set of $a_l$ is from $[0, 0.05, \ldots, 0.95]$, while the candidate of $a_u$ is from $[0.05, 0.1, \ldots, 0.95, 1]$. Here, we do not choose $a_l = 1$ and $a_u = 0$ as candidates, as $a_l = 1$ means that the labeled information
of the labeled set will not be considered in Eq. (4); whereas, $a_u = 0$ means that the labeled information of the unlabeled set will not be received from its neighborhoods. Both cases should be neglected when evaluating the confidence intervals of $a_l$ and $a_u$.

From simulation results in Fig. 13, we can draw the conclusion that the proposed CGSSL can work well in a wide interval of $a_l$ and $a_u$. In detail, for COIL20, USPS and Digit datasets, the proposed CGSSL can achieve robust annotated accuracies when $a_l$ and $a_u$ are in the intervals $[0.0; 0.9]$ and $[0.05, 0.95]$, respectively; whereas, for ETH80, MNIST and Extended Yale-B datasets, such confidence intervals of $a_l$ and $a_u$ are $[0.8, 0.85]$ and $[0.8, 0.85]$, respectively. In practice, since most of the datasets have wide confidence intervals of parameters $a_l$ and $a_u$, we can simply choose $a_l = 0$ and $a_u \leq 0.8$ for conducting the simulations. Clearly, a method that has wide intervals of parameters means that it is more robust to the parameters, and is more suitable and practical for real-world image annotation tasks.

4.5. Comparisons of computational time

In this subsection, we will compare the computational time of different methods when carrying out the simulation, which is shown in Table 7. Here, the configuration of the PC that we used is Intel (R) Core (TM) i7-4600U, CPU@3.4 GHz, and with 8 GB RAM.

From the simulation results in Table 7, we can see that the computational time is coincident to the analysis of computational
complexities of methods in Table 5 and has the following observations: (1) the proposed CGSSL needs more computational time for carrying out the simulations, except for some large-scale datasets such as Digit, COIL100, ETH80, USPS, MNIST, and Extended Yale-B datasets. In these datasets, SSKDE has the most computational time, as its computational complexity is closely related to the number of datasets; and (2) the LNP and the CGSSL indeed have more computational time than GFHF/LLGC/GGSSL, but not by a large factor. We can also see that in some large-scale datasets, such as COIL100, USPS, MNIST and Extended Yale-B datasets, the computational times of LNP and CGSSL are almost equal to those of GFHF/LLGC/GGSSL.

In addition, based on the analysis of Table 7, we will also give an explicit implementation choice of different methods from the time-to-process point of view: (1) for most UCI datasets, we can observe that it is better to utilize GFHF/LLGC/GGSSL for implementation, as their computational times are much smaller. Another reason is that UCI datasets are usually with small scale and simple structure. Hence, GFHF/LLGC/GGSSL can easily adjust the parameter of Gaussian covariance and achieve competitive performance compared with the proposed CGSSL; (2) for some real-world datasets, such as COIL100, USPS, MNIST and Extended Yale-B datasets, we observe that the computational times of CGSSL in these datasets are almost equal to those of GFHF/LLGC/GGSSL. Considering that CGSSL can achieve better performance than GFHF/LLGC/GGSSL, it is better to choose CGSSL for implementation. Another reason is that real-world datasets are usually with large scale and complex structure; thus, a method in which it is easy to adjust the parameters, such as the proposed CGSSL, can be more practical for handling the complex real-world datasets which are usually

Fig. 13. The average annotated accuracies over 20 random splits of COIL20, ETH80, MNIST, USPS, Extended Yale-B, and Digit Datasets with varied $a_l \in [0.05, 0.1, \ldots, 0.95]$ and $a_u \in [0.05, 0.1, \ldots, 0.95]$. 
Table 7
Computational time (s) of different methods.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>MR</th>
<th>SSKDE</th>
<th>GFHF</th>
<th>LLGC</th>
<th>GGSSL</th>
<th>LNP</th>
<th>CGSSL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ionosphere</td>
<td>1.0249e-3</td>
<td>2.4640e-4</td>
<td>9.9122e-4</td>
<td>9.9325e-4</td>
<td>9.9850e-4</td>
<td>1.1709e-3</td>
<td>2.4289e-3</td>
</tr>
<tr>
<td>Heart-Statlog</td>
<td>5.8540e-4</td>
<td>1.4580e-4</td>
<td>5.8752e-4</td>
<td>5.8951e-4</td>
<td>5.9153e-4</td>
<td>7.2576e-4</td>
<td>1.6934e-3</td>
</tr>
<tr>
<td>Iris</td>
<td>1.8006e-4</td>
<td>6.7500e-5</td>
<td>1.8360e-4</td>
<td>1.8964e-4</td>
<td>1.9636e-4</td>
<td>2.6040e-4</td>
<td>7.9800e-4</td>
</tr>
<tr>
<td>Wine</td>
<td>2.5567e-4</td>
<td>9.5052e-5</td>
<td>2.5774e-4</td>
<td>2.6046e-4</td>
<td>2.6884e-4</td>
<td>3.4888e-4</td>
<td>9.8681e-4</td>
</tr>
<tr>
<td>Digit</td>
<td>2.5294e-1</td>
<td>3.1584e-1</td>
<td>2.5312e-1</td>
<td>2.5327e-1</td>
<td>2.6090e-1</td>
<td>2.5600e-1</td>
<td>2.7614e-1</td>
</tr>
<tr>
<td>COIL100</td>
<td>1.4858e0</td>
<td>5.1840e0</td>
<td>4.2048e-1</td>
<td>4.2795e-1</td>
<td>4.3174e-1</td>
<td>4.3417e-1</td>
<td>4.4997e-1</td>
</tr>
<tr>
<td>ETH80</td>
<td>1.1598e0</td>
<td>8.6076e-1</td>
<td>8.8166e-2</td>
<td>8.8611e-2</td>
<td>8.9443e-2</td>
<td>8.8846e-2</td>
<td>1.0160e-1</td>
</tr>
<tr>
<td>USPS</td>
<td>7.0840e-1</td>
<td>6.6453e-1</td>
<td>6.9237e-1</td>
<td>7.0169e-1</td>
<td>7.0872e-1</td>
<td>7.1713e-1</td>
<td>7.3045e-1</td>
</tr>
<tr>
<td>MNIST</td>
<td>1.2819e0</td>
<td>1.0000e0</td>
<td>8.0080e-1</td>
<td>8.0546e-1</td>
<td>8.1255e-1</td>
<td>8.1592e-1</td>
<td>8.4176e-1</td>
</tr>
<tr>
<td>Extended Yale-B</td>
<td>3.1534e0</td>
<td>9.8781e0</td>
<td>2.0845e0</td>
<td>2.1064e0</td>
<td>2.1193e0</td>
<td>2.1228e0</td>
<td>2.1306e0</td>
</tr>
</tbody>
</table>
confronted in image annotation tasks, CBIR, and social network invoking. We have also included this point in the introduction.

4.6. Application for content based image retrieval

The goal of content-based image retrieval (CBIR) is to retrieve relevant images from the dataset by using semantic similarity between image content. In this subsection, we evaluate the CBIR performance of the proposed CGSSL, and compare them with those of the above mentioned semi-supervised graph based methods based on a large-scale image dataset, called the “Corel dataset” [32]. In this dataset, we choose a subset from the dataset for simulation, which contains 50 concepts, with each having 100 images [33]. Some samples of the Corel dataset are shown in Fig. 14.

In CBIR, low-level image representation is of great importance to characterize the feature information for each image. Current visual features include color, texture, shape, and different combinations of the previously mentioned methods. In this work, we combine a 48-dimension color histogram in HSV space [34] and a 48-dimension color texture moment (CTM) [35] to represent each image. For the color histogram, in order to avoid the computational burden, we only consider 16 bins to calculate the histograms for each of the three HSV channels. Specifically, the histogram of the hue channel is calculated as follows:

\[ h_{hi} = \frac{n_{hi}}{n_T}, \quad i = 1, 2, \ldots, 16 \]

(18)

where \( h_i \) is the \( i \)-th quantized level of the hue channel, \( n_{hi} \) is the total number of pixels with their hue values in that level, and \( n_T \) is the total number of pixels in the images. The histograms of the Saturation (\( h_s \)) and Value (\( h_v \)) channels can also be calculated as in Eq. (18). For the color texture moment (CTM), local Fourier transform (LFT) is adopted as a texture representation strategy, and eight characteristic maps are derived for describing different aspects of co-occurrence relations of images pixels in the three channels of the color space \( (SV \cos H, SV \sin H, \text{and } V) \). Then, CTM calculates the first and second moments of these maps to represent the natural color image pixel distribution.

We next design an automatic feedback strategy to model the retrieval process. For each query image submitted by the user, the system retrieves and ranks the images in the database. Here, the rank for each image in the database is based on the estimated label information after performing the proposed CGSSL or other state-of-the-art methods. The top images with the highest ranking score are then selected as the feedback images, and their feedback information can be used for re-ranking. It is worth noting that for the sake of feedback, users should mark them as either relevant or irrelevant. The above automatic feedback strategy is performed for four iterations for each query image.

We next discuss how to utilize the proposed CGSSL in a CBIR system. Given a query image submitted by the user, the CBIR system first treats it as a labeled set, while treating other images in the system database as an unlabeled set. Then, by taking both the labeled and unlabeled set as inputs, the proposed CGSSL can automatically annotate the labels of the unlabeled set, and the top images (maybe 10, 20, or more images) with the highest values of estimated labels are selected as the feedback images. Next, the users can annotate such feedback images as relevant or irrelevant to the query image. In other words, if a feedback image is judged as relevant, it will be added to the labeled set, making the number of the labeled set increase. The new-formed labeled set combined with the remaining unlabeled set can then be used as inputs for a new-round annotation. The process will be iteratively performed.

**Fig. 14.** Sample images of the Corel dataset.

**Fig. 15.** Average accuracy-scope curves on a test set of different methods for the second and fourth feedback iterations: Corel dataset (a) the second iteration and (b) the fourth iteration.
several times until the user’s requirements are satisfied. Hence, with the increase of the labeled set, the feedback annotated images are expected to be more and more relevant to the query image.

In a real image retrieval system, however, the query image is usually not in the image database. To simulate this environment, we divide the dataset into two non-overlapping subsets, i.e., the first one includes 4500 images used for training the system, and the second one includes 500 images used as query images. For the image retrieval results, we use an accuracy-scope curve [35] to evaluate the performance of different methods under a fixed iteration. The scope is the number of top-ranked images presented to the user, and the accuracy is the ratio between the numbers of relevant images to the given scope. The accuracy-scope curve describes the accuracies with varied scopes and, thus, gives an overall performance evaluation of the algorithms. We then fix the scope by 10, 20 and 50, and evaluate the performance of the different methods with varied iterations.

Fig. 15 shows the average accuracy-scope curves of different methods at iteration 2 and iteration 4. Fig. 16 shows the average accuracies with varied iterations under the fixed scope of 10, 20, and 50. From Figs. 15 and 16, we have the following observations:

1. (1) The proposed CGSSL can achieve better performance in most cases of dataset due to the reason that the new graph construction strategy in CGSSL can find a compact way to approximate a sample with its neighborhoods.
2. (2) Since the proposed CGSSL is based on the framework of GGSSL, it can either detect the outliers or develop a mechanism to calculate the probabilities of data samples. In addition, with symmetrization and normalization processes, the proposed CGSSL can be analyzed theoretically under a regularized framework.
3. (3) The proposed CGSSL can automatically erase noisy labeled samples that are generalized due to carelessness of the annotator, and are robust to the parameters. Clearly, a method that can handle noisy annotated samples and is less sensitive to the parameters is more suitable for real-world annotation tasks.
4. (4) The proposed CGSSL can be easily extended to its inductive out-of-sample version for handling new-coming data by using the same smoothness criterion.

While our method can achieve promising results for automatic image annotation and retrieval, our future work might focus on the following issues in order to improve the proposed method:

1. (1) The proposed CGSSL can achieve better performance in most cases of dataset due to the reason that the new graph construction strategy in CGSSL can find a compact way to approximate a sample with its neighborhoods.
2. (2) Since the proposed CGSSL is based on the framework of GGSSL, it can either detect the outliers or develop a mechanism to calculate the probabilities of data samples. In addition, with symmetrization and normalization processes, the proposed CGSSL can be analyzed theoretically under a regularized framework.
3. (3) The proposed CGSSL can automatically erase noisy labeled samples that are generalized due to carelessness of the annotator, and are robust to the parameters. Clearly, a method that can handle noisy annotated samples and is less sensitive to the parameters is more suitable for real-world annotation tasks.
4. (4) The proposed CGSSL can be easily extended to its inductive out-of-sample version for handling new-coming data by using the same smoothness criterion.

5. Conclusion

In this paper, we propose an effective label propagation procedure, which is based on a new local reconstruction graph with symmetrization and normalization for image annotation. Following the theoretical analysis and extensive studies in our work, we can draw the conclusions for the proposed CGSSL as:

1. (1) The proposed CGSSL can achieve better performance in most cases of dataset due to the reason that the new graph construction strategy in CGSSL can find a compact way to approximate a sample with its neighborhoods.
2. (2) Since the proposed CGSSL is based on the framework of GGSSL, it can either detect the outliers or develop a mechanism to calculate the probabilities of data samples. In addition, with symmetrization and normalization processes, the proposed CGSSL can be analyzed theoretically under a regularized framework.
3. (3) The proposed CGSSL can automatically erase noisy labeled samples that are generalized due to carelessness of the annotator, and are robust to the parameters. Clearly, a method that can handle noisy annotated samples and is less sensitive to the parameters is more suitable for real-world annotation tasks.
4. (4) The proposed CGSSL can be easily extended to its inductive out-of-sample version for handling new-coming data by using the same smoothness criterion.
Acknowledgement
This work was partly supported by the National Natural Science Foundation of China (Grant No. 61300209), partly supported by major program of National Natural Science Foundation of China (Grant No. 61033013) and also partly supported by the National Natural Science Foundation of China (Grant No. 61402310).

Appendix A

Denote \( e_k = [1, 1, \ldots, 1] \in \mathbb{R}^{1 \times 55} \) as the unit vector, to prove that the sum of each column of \( F \) is equal to 1, we need to prove \( e_{c-1}F = e_{c-1}W = e_{c-1}Y = e_{c-1}u \). Actually, we first have \( e_{c-1}W = e_{c-1}Y = e_{c-1}u \), then:

\[
\begin{aligned}
\{ e_{c-1}Y &= e_{c-1}u, \\
\quad e_{c-1}W &= e_{c-1}u, \\
- e_{c-1}Y_{j} + e_{c-1}W_{j} &= e_{c-1}Y_{j},
\}
\Rightarrow e_{c-1}Y_{j} = e_{c-1}Y_{j} - e_{c-1}W_{j} = e_{c-1}Y_{j} \quad (19)
\end{aligned}
\]

References