Adjacency Matrix Construction Using Sparse Coding for Label Propagation

Haixia Zheng, Horace H.S. Ip, and Liang Tao

Centre for Innovative Applications of Internet and Multimedia Technologies
(AIMtech Centre), Department of Computer Science,
City University of Hong Kong, Kowloon, Hong Kong
{hxzheng2,liangtao3}@student.cityu.edu.hk,
cship@cityu.edu.hk

Abstract. Graph-based semi-supervised learning algorithms have attracted increasing attentions recently due to their superior performance in dealing with abundant unlabeled data and limited labeled data via the label propagation. The principle issue of constructing a graph is how to accurately measure the similarity between two data examples. In this paper, we propose a novel approach to measure the similarities among data points by means of the local linear reconstruction of their corresponding sparse codes. Clearly, the sparse codes of data examples not only preserve their local manifold semantics but can significantly boost the discriminative power among different classes. Moreover, the sparse property helps to dramatically reduce the intensive computation and storage requirements. The experimental results over the well-known dataset Caltech-101 demonstrate that our proposed similarity measurement method delivers better performance of the label propagation.

1 Introduction

The issues faced by many practical applications in pattern recognition are that only a few labeled data are available and large amounts of data remain unlabeled. Since it is quite expensive and time consuming to label data whereas unlabeled data can be easier to obtain. How to efficiently combine abundant unlabeled data and limited labeled data is an important research field, which is the main aim of semi-supervised learning techniques [1].

In the past decade, the graph-based semi-supervised learning approaches have aroused considerable interests as they can generate elegant mathematical formulation through label propagation and are easy to be implemented . The graph-based semi-supervised learning approaches build the whole dataset as a graph where the vertices represent the data and the edges represent the pairwise relationships. Zhu et al. [2] utilized the harmonic property of Gaussian random field over the graph for semi-supervised learning. Belkin [3] learned a regression function that fits the labels at labeled data and at the same time maintains smoothness over the data manifold expressed by the graph. Zhou et al. [4] proposed to conduct semi-supervised learning with the local and global consistency.

A. Fusiello et al. (Eds.): ECCV 2012 Ws/Demos, Part III, LNCS 7585, pp. 315–323, 2012.

Delalleu et al. [5] proposed a nonparametric inductive function which makes label prediction based on a subset of samples and then truncates the graph Laplacian with the selected subset and its connection to the rest samples. Zhang et al. [6] applied the Nyström approximation to the huge graph adjacency (or affinity) matrix. Fergus et al. [7] specified the label prediction function using smooth eigenvectors of the graph Laplacian which are calculated by a numerical method. Liu et al. [8] constructed a tractable large graph via a small number of anchor points, and predicted the label for each data point as a locally weighted average of the labels on anchor points.

However, the performance of these graph-based semi-supervised learning approaches relies heavily on the adjacency matrix construction. Each entry in adjacency matrix denotes the weight of the corresponding edge, which specifies the similarity between the two data samples. A number of methods have been proposed to construct the adjacency matrix. The simplest method to measure the similarity of data points is to use the Euclidean distances between them. A straightforward extension is the K-Nearest Neighbor (KNN) [9], in which only the edges between a data point and its K-Nearest Neighbors have non-zero weights. Another widely used method is using Gaussian Kernel Similarity [4] to compute the edge weights of the graph. Roweis et al. [10] first proposed to reconstruct the sample from its neighboring points and utilize the local linear reconstruction coefficients as graph weights. Cheng et al. [11] measured the similarities among data points by decomposing each data point as a L_1 sparse linear combination of the rest of the data points. Hence, all these adjacency matrix construction methods are using the original high-dimension data points and are thus relatively computationally expensive.

In this paper, we compute the similarities among the data points based on their low dimensional sparse codes. The key idea is that their sparse codes of data points not only preserve their local manifold structures but can improve the discriminative power among different classes. Further, the sparse property also helps to reduce the intensive computation and storage requirements. Therefore, this approach is much easier to scale up for the large scale dataset. We evaluate the proposed approach on the popular dataset Caltech-101 for the image categorization task. The experimental results indicate that the proposed algorithm significantly improves label propagation performance.

2 Adjacency Matrix Construction Using Sparse Coding

In this section, we will present the adjacency matrix construction algorithm. Let $X = \{x_1, x_2, \cdots, x_n\}$ denote the whole dataset. The goal is to construct the adjacency matrix $W = [W_{ij}]_{n \times n}$, where W_{ij} denotes the similarity between the data x_i and x_j. It can be achieved by the following two steps:

2.1 Calculate Their Sparse Codes of All the Data

Sparse coding provides a class of algorithms for finding succinct representation of data. Given a large number of input data, sparse coding algorithms can

automatically compute a small number of representative patterns which are called basis set and the original data space can be represented by appropriate combination of basis set. Therefore, the high dimensionality of the original data can be reduced to the low dimensions of its sparse coding representation. Another important property of sparse coding method is that it usually produces a sparse representation of the data. Such a representation encodes much of the data using few active components, which makes the encoding easy to interpret and at the same time saves the store space and computational time.

For the dataset $X = \{x_1, x_2, \cdots, x_n\} \subset \mathbb{R}^m$, the optimization equation to compute sparse codes is:

$$\min_{A,B} \sum_{i=1}^{n} (\|x_i - B\alpha_i\|^2 + \lambda\phi(\alpha_i)), \tag{1}$$

where $B = [b_1, \cdots, b_p] \in \mathbb{R}^{m\times p}$ is the basis set, also called dictionary; $A = [\alpha_1, \cdots, \alpha_n] \in \mathbb{R}^{p\times n}$ is the sparse codes set for X, and most entries in A are zeros; λ is regularization parameter and sets to be 0.15 in our experiments; $\phi(\alpha_i)$ is regularization function.

Usually, $\phi(\alpha_i)$is set to be $\mathbb{L}_1$ penalty function $\|\alpha_i\|_1$, because $\mathbb{L}_1$ penalty yields a sparse solution for α_i and can be robust to irrelevant features [12,13]. But this does not consider the inherent domain knowledge embedded in the data itself.

In our algorithm, we set $\phi(\alpha_i)$ to be locality constraint regularization $\alpha_i^T E_i \alpha_i$, where E_i is a $p \times p$ diagonal matrix with its (j, j)-element equal to $\exp(\|x_i - b_j\|_2/\delta)$ and δ is used to adjust the weight decay speed for the locality constraint. It should be noted that locality is more essential than sparsity, as locality must lead to sparsity but not necessary vice versa [14,15].

This optimization problem for Eq. (1) is not jointly convex for basis set B and sparse codes A simultaneously. But the sparse coding problem is convex for basis set B when sparse codes A is fixed, and is also convex for sparse codes A when basis set B is fixed. Similar to [12,13], we can minimize the objective function with respect to basis set B and sparse codes A alternatively.

When sparse codes A is fixed, the Eq. (2) with respect to dictionary B can be solved using incremental codebook optimization method [15]:

$$\begin{array}{ll} \min_B & \sum_{i=1}^{n} (\|x_i - B\alpha_i\|^2 + \lambda\alpha_i^T E_i \alpha_i) \\ \text{s.t.} & \sum_{i=1}^{p} \|b_i\| \leq 1 \end{array} \tag{2}$$

To handle the scale issue associated with the dictionary, we add extra normalization constraints $\|b_i\| \leq 1$ $(i = 1, \cdots, p)$ into the above optimization.

When basis set B is fixed, the formula can be reduced to:

$$\min_A \sum_{i=1}^{n} (\|x_i - B\alpha_i\|^2 + \lambda\alpha_i^T E_i \alpha_i) \tag{3}$$

We can optimize every α_i alternatively and do not consider all the sparse codes $A = [\alpha_1, \cdots, \alpha_n]$ simultaneously. That is, when we focus on α_i, the other sparse

codes are fixed. Then we can get the analytical closed-form solution of Eq. (3) with respect to α_i as follows:

$$\begin{aligned} \alpha_i^* &= \arg\min_{\alpha_i} \sum_{i=1}^{n} (\|x_i - B\alpha_i\|^2 + \lambda \alpha_i^T E_i \alpha_i) \\ &= (B^T B + \lambda E_i) \backslash (B^T x_i) \end{aligned} \tag{4}$$

2.2 Construct Adjacency Matrix Based on Their Sparse Codes

The weight W_{ij} is conventionally calculated using Gaussian Kernel Similarity as

$$W_{ij} = \begin{cases} \exp\left(-\|\alpha_i - \alpha_j\|^2 / 2\sigma^2\right), & i \neq j \\ 0, & i = j \end{cases} \tag{5}$$

However, the major shortcoming using Gaussian Kernel Similarity method is that even a small perturbation of variable σ will make its performance dramatically different, and there is no reliable way to determine the optimal value of parameters especially when the amount of labeled data is rather small [8,16].

Similar to [10,16], we specify the similarity between two data points through geometric reconstruction. With the consideration that the sparse codes of data points not only preserve their local manifold structures but can boost the discriminative power among different classes, and that the sparse property can also help to reduce the computation cost, we assign the weights of adjacency matrix through local linear reconstruction coefficients of their sparse codes. Therefore, the objective is

$$\begin{aligned} &\min_W \ \sum_{i=1}^{n} \|\alpha_i - \sum_{j:\alpha_j \in N(\alpha_i)} W_{ij} \alpha_j\|^2, \\ &\text{s.t.} \quad W_{ij} \geq 0 \text{ and } \sum_{j:\alpha_j \in N(\alpha_i)} W_{ij} = 1 \end{aligned} \tag{6}$$

in which $N(\alpha_i)$ represents the neighborhood of α_i, and W_{ij} is the contribution of α_j to α_i. Obviously, the more similar α_j to α_i, the larger W_{ij} will be. Especially, when $\alpha_i = \alpha_k \in N(\alpha_i)$, then $W_{ik} = 1$ and $W_{ij} = 0\,(j \neq k, \alpha_j \in N(\alpha_i))$ is the optimal solution. Therefore, W_{ij} can be used to measure how similar α_j to α_i.

The reconstruction weights W_{ij} in Eq. (6) can be solved using the standard quadratic programming (QP) [16]. More importantly, the optimized regression weights in Eq. (6) are much sparser. After all the reconstruction weights W_{ij} are solved, we can obtain a sparse adjacency matrix $W = [W_{ij}]_{n \times n}$. In order to ensure adjacency matrix W is symmetric, we then set $W = (W + W^T)/2$.

3 The Framework of Label Propagation

Given a data set $\mathcal{X} = \{x_1, \cdots, x_l, x_{l+1}, \cdots, x_n\} \subset \mathbb{R}^m$ and a label set $\mathcal{L} = \{1, \cdots, c\}$, the first l data $x_i (i \leq l)$ are labeled as $y_i \in \mathcal{L}$ and the remaining data $x_i\,(l+1 \leq i \leq n)$ are unlabeled. The goal is to predict the label of the unlabeled data through label propagation. Let F denote the set of $n \times c$ matrices with

nonnegative entries. Matrix $F = [F_1^T, \cdots, F_n^T]^T \in \mathcal{F}$ corresponds to a classification on the dataset $\mathcal{X}$ by labeling each point x_i as the label $y_i = \arg\max_{j \le c} F_{ij}$. We can understand F as a vectorial function $F : \mathcal{X} \mapsto \mathbb{R}^c$ which assigns a vector F_i to each data x_i. Define a $n \times c$ matrix $Y \in \mathcal{F}$ with $Y_{ij} = 1$ if x_i is labeled as $y_i = j$ and $Y_{ij} = 0$ otherwise. Clearly, Y is consistent with the initial labels according to decision rule [4]. The algorithm can be summarized as follows:

(1) Compute the sparse codes $A = [\alpha_1, \cdots, \alpha_n]$ for all the data $\mathcal{X} = \{x_1, \cdots, x_l, x_{l+1}, \cdots, x_n\}$ through Eq. (4).
(2) Construct the adjacency matrix $W = [W_{ij}]_{n \times n}$ by solving the Eq. (6), and set $W = (W + W^T)/2$ to ensure W is symmetric.
(3) Iterate $F(t+1) = \gamma W F(t) + (1-\gamma)Y$ until convergence, where γ is a parameter and set to be 0.75 in our evaluations.
(4) Let F^* denote the limit the sequence $\{F(t)\}$. Label each data x_i as a label $y_i = \arg\max_{j \le c} F_{ij}$.

According to [16], the above algorithm will converge to

$$F^* = (1-\gamma)(\mathbf{I} - \gamma W)^{-1} Y, \tag{7}$$

in which $\mathbf{I}$ is the identity matrix of order n.

4 The Regularization Framework

We can derive our algorithm from a regularization framework. The cost function associated with F can be defined as follows:

$$Q(F) = \frac{1}{2}\left(\sum_{i=1}^{n} \sum_{j:\alpha_j N(\alpha_i)} W_{ij} \|F_i - F_j\|^2 + \mu \sum_{i=1}^{n} \|F_i - Y_i\|^2\right) \tag{8}$$

where $\mu > 0$ is the regularization parameter. The first term is the smoothness constraint, which means that a good classifying function should not change too much between nearby points. The second term of the formula is the fitting constraint, which means a good classifying function should not change too much from the initial label assignment [4]. Note that the fitting constraint term contains labeled and unlabeled data. Differentiate $Q(F)$ with respect to F, we have

$$\frac{\partial Q(F)}{\partial F} = (\mathbf{I} - W)F + \mu(F - Y) \tag{9}$$

Then we can easily get the optimal solution of Eq. (8) by setting Eq. (9) to zero:

$$F = (1-\gamma)(\mathbf{I} - \gamma W)^{-1} Y \tag{10}$$

where $\gamma = 1/(1+\mu)$. It can be observed that Eq. (10) is equal to the Eq. (7) —the closed form expression of the above iteration algorithm.

5 Experiments

First, we evaluate the performance of our algorithm over the widely-used dataset Caltech 101 [17]. The Caltech-101 dataset contains 9144 images in 101 classes (including airplanes, cameras, chairs, etc.) with significant variance in shape. The number of images per category varies from 31 to 800. Most images are medium resolution, i.e. about 300×300 pixels. Following the common experiment setup for Caltech-101 suggested by the original dataset [17] and also used by many other researchers [18], for each category we partitioned the whole dataset into 5, 10, 15, 20, 25 and 30 training images and the rest are test images. After that, we measured the performance using average accuracy over 102 classes (i.e. 101 classes and a 'background' class).

Our implementations utilizes a single descriptor type, the popular SIFT descriptor, as in [19,20]. The SIFT descriptors extracted from 16×16 pixel patches were densely sampled from each image in the whole dataset on a grid with step size of 8 pixels. All the image features are quantized into 2048 clusters. In our experiments the images were all preprocessed into gray scale, and resized to be no larger than 300×300 pixels with preserved aspect ratio.

Following the common benchmarking procedures, we repeat the experimental process by 10 times with different random selected training and test images to obtain reliable results. The average of per-category recognition rates were recorded for each run. Finally, we report our results by the average of all the recognition rates. We also compared our result with several existing approaches. Detailed comparison results are shown in Fig. 1.

In Fig. 1, it can be seen that our proposed approach outperforms other existing state-of-the-art image categorization algorithms at least 1.04% (15 training images per category). Our approach can achieve up to 2.34% of performance

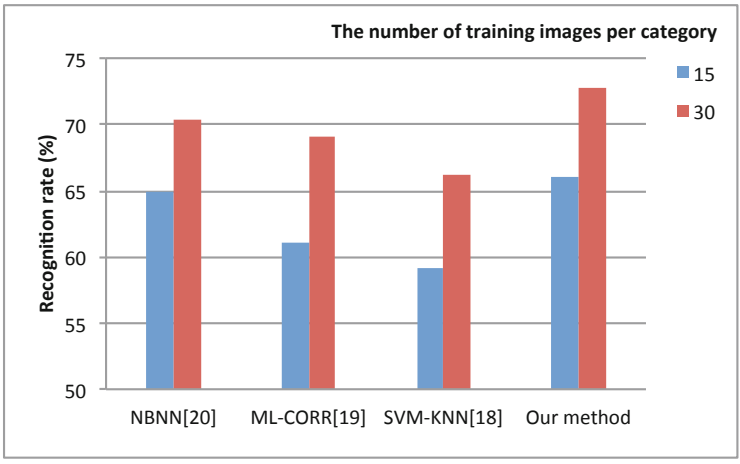

Fig. 1. Image categorization performance comparison on Caltech-101 dataset

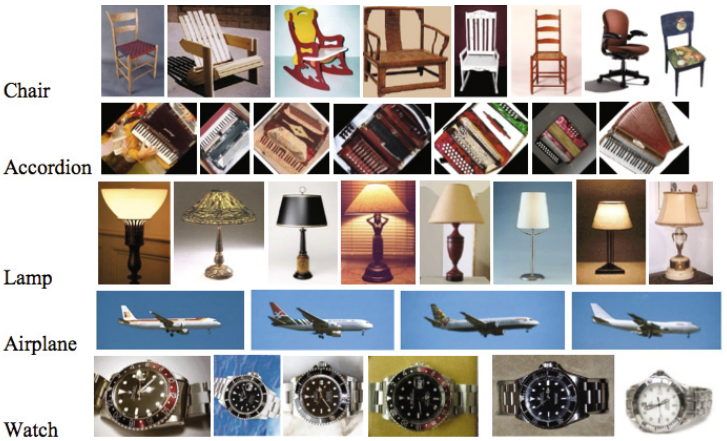

Fig. 2. Image classes with higher recognition rate

Fig. 3. Performance comparison between our method and LNP

improvement with 30 training images per category. At the same time, we can observe the recognition rate growing with the number of training images per category due to the benefit of growing supervisory information of training data. As illustrated in Fig. 2, with 15 training images per category, the recognition rate of higher than 75% is obtained in five classes.

To further evaluate the effectiveness of our proposed adjacency matrix algorithm, we compare our algorithm with the Linear Neighborhood Propagation (LNP) [16], which computes the weights of affinity matrix based on local linear reconstruction of original high-dimension data points. As shown in Fig. 3, our algorithm outperforms Linear Neighborhood Propagation [16]. Importantly,

the adjacency matrix constructed by means of sparse codes will be much sparser than that based on original high-dimension data, thus the computation cost and storage space will be reduced.

6 Conclusion

We propose a novel affinity matrix construction approach for label propagation through the local linear reconstruction of their sparse codes. The proposed algorithm takes into consideration that sparse codes of data samples not only preserve their local manifold structures but can boost the discriminative power among different classes. Additionally, the sparse property leads to requires less computation cost and storage overhead. The experimental evaluations demonstrate that effectiveness of our proposed method is fully comparable with the LNP [16].

References

1. Zhu, X.: Semi-supervised learning literature survey. Technical report, Department of Computer Sciences, University of Wisconsin, Madison (2005)
2. Zhu, X., Ghahramani, Z., Lafferty, J.: Semi-supervised learning using gaussian fields and harmonic functions. In: ICML, pp. 912–919 (2003)
3. Belkin, M., Matveeva, I., Niyogi, P.: Regularization and Semi-supervised Learning on Large Graphs. In: Shawe-Taylor, J., Singer, Y. (eds.) COLT 2004. LNCS (LNAI), vol. 3120, pp. 624–638. Springer, Heidelberg (2004)
4. Zhou, D., Bousquet, O., Lal, T.N., Weston, J., Schölkopf, B.: Learning with local and global consistency. In: Advances in Neural Information Processing Systems 16, pp. 321–328. MIT Press (2004)
5. Delalleau, O., Bengio, Y., Roux, N.L.: Nonparametric function induction in semi-supervised learning. In: Proc. Artificial Intelligence and Statistics (2005)
6. Zhang, K., Kwok, J.T., Parvin, B.: Prototype vector machine for large scale semi-supervised learning. In: Proceedings of the 26th Annual International Conference on Machine Learning (2009)
7. Fergus, R., Weiss, Y., Torralba, A.: Semi-Supervised Learning in Gigantic Image Collections. In: Neural Information Processing Systems (2009)
8. Liu, W., He, J., Chang, S.F.: Large Graph Construction for Scalable Semi-Supervised Learning. In: International Conference on Machine Learning, pp. 679–686 (2010)
9. Belkin, M., Niyogi, P.: Laplacian Eigenmaps for Dimensionality Reduction and Data Representation. Neural Computation, 1373–1396 (2003)
10. Roweis, S.T., Saul, L.K.: Nonlinear dimensionality reduction by locally linear embedding. Science, 2323–2326 (2000)
11. Cheng, H., Liu, Z., Yang, J.: Sparsity induced similarity measure for label propagation. In: International Conference on Computer Vision, pp. 317–324 (2009)
12. Lee, H., Battle, A., Raina, R., Ng, A.Y.: Efficient sparse coding algorithms. In: Neural Information Processing Systems, pp. 801–808 (2006)
13. Mairal, J., Bach, F., Ponce, J., Sapiro, G.: Online dictionary learning for sparse coding. In: International Conference on Machine Learning, pp. 689–696 (2009)

14. Lu, Z., Peng, Y.: Latent semantic learning by efficient sparse coding with hypergraph regularization. In: Burgard, W., Roth, D. (eds.) AAAI (2011)
15. Wang, J., Yang, J., Yu, K., Lv, F., Huang, T.S., Gong, Y.: Locality-constrained Linear Coding for image classification. In: Computer Vision and Pattern Recognition, pp. 3360–3367 (2010)
16. Wang, F., Zhang, C.: Label propagation through linear neighborhoods. In: International Conference on Machine Learning, pp. 985–992 (2006)
17. Fei-Fei, L., Fergus, R., Perona, P.: Learning Generative Visual Models from Few Training Examples: An Incremental Bayesian Approach Tested on 101 Object Categories. In: Computer Vision and Pattern Recognition (2004)
18. Zhang, H., Berg, A.C., Maire, M., Malik, J.: SVM-KNN: Discriminative Nearest Neighbor Classification for Visual Category Recognition. In: Computer Vision and Pattern Recognition, vol. 2, pp. 2126–2136 (2006)
19. Jain, P., Kulis, B., Grauman, K.: Fast image search for learned metrics. In: Computer Vision and Pattern Recognition (2008)
20. Boiman, O., Shechtman, E., Irani, M.: In defense of Nearest-Neighbor based image classification. In: Computer Vision and Pattern Recognition (2008)