Expert Systems with Applications 41 (2014) 7744–7753

Contents lists available at ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

Graph-based semi-supervised learning with Local Binary Patterns for holistic object categorization

F. Dornaika [a,b,*], A. Bosaghzadeh [a], H. Salmane [c], Y. Ruichek [c]

[a] University of the Basque Country UPV/EHU, San Sebastian, Spain
[b] IKERBASQUE, Basque Foundation for Science, Bilbao, Spain
[c] IRTES-SET, UTBM, 90010 Belfort, France

ARTICLE INFO

Article history:
Available online 22 June 2014

Keywords:
Graph-based semi-supervised learning
Graph-based label propagation
Local Binary Patterns
Holistic object classification
Outdoor scenes
Indoor scenes

ABSTRACT

In this paper, we develop a new efficient graph construction algorithm that is useful for many learning tasks. Unlike the main stream for graph construction, our proposed data self-representativeness approach simultaneously estimates the graph structure and its edge weights through sample coding. Compared with the recent ℓ_1 graph based on sparse coding, our proposed objective function has an analytical solution (based on self-representativeness of data) and thus is more efficient. This paper has two main contributions. Firstly, we introduce a principled Two Phase Weighted Regularized Least Square graph construction method. Secondly, the obtained data graph is used, in a semi-supervised context, in order to categorize detected objects in outdoor and indoor scenes using Local Binary Patterns as image descriptors. In many previous works, LBP descriptors (histograms) were used as feature vectors for object detection and recognition. However, our work exploits them in order to construct adaptive graphs using a self-representativeness coding. The experiments show that the proposed method can outperform competing methods.

1. Introduction

Advanced Driver Assistance Systems and outdoor video surveillance very often need to categorize detected objects/obstacles. In these scenarios, the considered classes usually determine different responses or levels of assessment related to the situation. Class information can be integrated within the global navigation architecture, for example, in obstacle avoidance, mapping or tracking modules. In assistance systems for commercial cars, classes can be used to trigger the corresponding alarms or actions.

Based on visual data, two main categories of approaches were developed. The first category of approaches uses a specific class trained detector (e.g., pedestrian detection Geronimo, Lopez, Sappa, & Graf, 2010). Thus, the detector itself will provide the class. The second category of approaches estimates the class after a detection phase (e.g., Jung, Lee, Yoon, Hwang, & Kim, 2006). The first category of approaches can be appealing if the application focuses on one single class. However, it becomes tedious and difficult to use whenever many classes should be simultaneously handled. The second category of approaches can be deployed regardless of the number of classes the system should recognize (Zhang, Liu, Liang, Huang, & Tian, 2013).

One interesting framework that belongs to the classification is the graph-based semi-supervised classification. This framework allows the classification of several instances at once given a set of labeled observations. This framework lends itself nicely to the domains of video surveillance and Advanced Driver Assistance. Indeed, in the related applications a given image can contain several detected candidates (image regions). In a video sequence, the detected candidates can be so numerous since they are detected in space and time. Therefore, the semi-supervised framework can label all detected objects.

Graph-based methods operate on a graph where a node corresponds to a data instance and a pair of nodes are connected by a weighted edge encoding the similarity between these two nodes. At present, the most popular graph construction manner is based on the K nearest neighbor and ϵ-ball neighborhood criteria. Once a neighborhood graph is constructed, the edge weights are assigned by Gaussian Kernels or coefficients provided by local reconstruction algorithms (Roweis & Saul, 2000). In Jebara, Wang, and Chang (2009), the authors propose a graph construction via b-Matching. The goal is to produce a binary adjacency matrix with the constraint that the resulting graph is undirected (symmetric weight matrix) and the constraint that each node will have the

* Corresponding author at: University of the Basque Country UPV/EHU, Manuel Lardizabal 1, San Sebastian 20018, Spain. Tel.: +34 943018034.
E-mail address: fdornaika@gmail.com (F. Dornaika).

http://dx.doi.org/10.1016/j.eswa.2014.06.025
0957-4174/

same degree[1] given by the parameter b. The solution was obtained by loopy belief propagation. It was shown that the label propagation algorithm that uses the resulting adjacency graph, has slightly better performance than that based on the KNN graph. However, the b-matching graph construction needs tuning the parameter b. Furthermore, since the output of b-matching is a binary weight matrix, an additional stage is needed for edge re-weighting.

In Yan and Wang (2009), the authors argue that the graph adjacency structure and the graph weights are interrelated and should not be separated. Thus, it is desired to develop a procedure that can simultaneously complete these two tasks within one step. To this end, every sample image is coded as a sparse linear combination of the rest of the training samples. This is carried out by implementing the ℓ_1 minimization process that finds the desired sparse representation of that sample. The obtained sparse coefficients will reflect the relation among samples, and hence will provide the graph adjacency structure as well as the weights of its edges.

This paper has two main contributions. Firstly, we introduce a Two Phase Weighted Regularized Least Square (TPWRLS) graph construction. Secondly, the obtained data graph is used, in a semi-supervised context, in order to categorize detected objects in driving/urban scenes using Local Binary Patterns as image descriptors. In many previous works, LBP descriptors (histograms) were used as feature vectors for object detection and recognition. However, our work exploits them in order to construct adaptive graphs using a self-representativeness coding.

The whole proposed framework can be useful for at least two schemes: (i) inferring labels in large datasets having a tiny fraction of labeled samples, and (ii) online categorization of detected objects. The paper is organized as follows. Section 2 reviews some state of art methods for graph construction. Section 3 briefly describes the LBP descriptor. Section 4 presents our proposed graph construction method. Section 5 presents the application of the proposed graph to the problem of semi-supervised categorization of outdoor and indoor objects. Many experiments were conducted to evaluate and to compare the performances of the proposed methods with respect to competing methods.

2. Related work: graph construction methods

2.1. K-Nearest Neighbor method

This method is the most common method used for graph construction. It is decomposed into two separate and independent processes. First, the adjacency matrix is constructed (edges are set). Second, the weights of the edges are estimated.

For adjacency matrix construction, K-Nearest Neighbor can be used in order to find the neighbors of a datum. There is a function that defines the distance (similarity) of one input with respect to the others.

In the second phase, a weight should be assigned to each constructed edge. In general, this weight should quantify the similarity between two connected nodes. Let $sim(\mathbf{x}_i, \mathbf{x}_j)$ be the similarity score between neighbors $\mathbf{x}_i$ and $\mathbf{x}_j$, then the elements of the graph weight matrix $\mathbf{W}$ are given by Eq. (1). There are several choices for $sim(\mathbf{x}_i, \mathbf{x}_j)$. For instance, Belkin and Niyogi (2003) uses the heat kernel $sim(\mathbf{x}_i, \mathbf{x}_j) = e^{-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{t}}$ with different gaussian variance t values. In the extreme case where $t \to \infty$ the weights will become 0 and 1. 0 when there is no connection and 1 when two nodes are connected. The authors in Cortes and Mohri (2006) proposed the use of inverse of distance as weight, $sim(\mathbf{x}_i, \mathbf{x}_j) = \frac{1}{\|\mathbf{x}_i - \mathbf{x}_j\|}$.

[1] The degree of a node is equal to the sum of weights of all edges linked to that node.

$$W_{ij} = \begin{cases} sim(\mathbf{x}_i, \mathbf{x}_j) & \text{if } \mathbf{x}_i \text{ and } \mathbf{x}_j \text{ are neighbors} \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

2.2. ϵ-Ball graph method

In this method the neighbors of a given $\mathbf{x}_i$ are set to the data samples that belong to a sphere centered at $\mathbf{x}_i$ and having ϵ as radius. In ϵ-Neighborhood method the data which have the distance (similarity) less (more) than the threshold ϵ, will be selected as neighbors. The drawback of ϵ-Neighborhood method is that there might be some inputs without neighbors. Furthermore, the value of ϵ is user-defined, so different values should be tested in order to find the optimum one. KNN graph remains the more common approach since it is more adaptive to scale and data density while an improper threshold value in ϵ-Neighborhood graph could result in disconnected components or subgraphs in the dataset or even isolated singleton vertices.

2.3. LLE graph construction

Locally Linear Embedding (LLE) focuses on preserving the local structure of data. LLE formulates the manifold learning problem as a neighborhood-preserving embedding, which learns the global structure by exploiting the local linear reconstructions. It estimates the reconstruction coefficients by minimizing the reconstruction error of the set of all local neighborhoods in the dataset. It turned out that linear coding used by LLE can be used for computing the graph weight matrix.

Thus, LLE graph can be obtained by applying two stages: adjacency matrix computation followed by the linear reconstruction of samples from their neighbors. The adjacency matrix can be computed using the KNN or ϵ-Neighborhood method. The non-zero entries of the weight matrix $\mathbf{W}$ are estimated by reconstructing the sample from its neighboring points and minimizing the ℓ_2 reconstruction error defined as

$$\sum_{i=1}^{N} \|\mathbf{x}_i - \sum_j W_{ij}\mathbf{x}_j\|^2 \; s.t. \sum_{j=1}^{N} W_{ij} = 1. \quad (2)$$

where $W_{ij} = 0$ if $\mathbf{x}_i$ and $\mathbf{x}_j$ are not neighbors.

2.4. ℓ_1 Graph construction

Instead of building a graph in two different processes of adjacency construction and weight calculation, the authors in Yan and Wang (2009) tried to unify them in one single process. In their proposed method every sample is coded as a sparse linear combination of the rest of the training samples and the contributions of images in representing the sample are considered as weights.

Consider a D dimensional vector $\mathbf{y}$ as an input and a $D \times N$ database matrix $\mathbf{X}$, containing N samples. The goal is to represent input $\mathbf{y}$ as a sparse linear combination of database matrix $\mathbf{X}$. Mathematically, it can be written as

$$\min \|\mathbf{b}\|_1 \quad s.t. \quad \mathbf{y} = \mathbf{X}\mathbf{b} \quad (3)$$

where vector $\mathbf{b} \in \mathbb{R}^N$ is the coefficient vector. Due to the presence of noise, Eq. (3) will become

$$\min \|\mathbf{b}\|_1 \quad s.t. \quad \|\mathbf{y} - \mathbf{X}\mathbf{b}\|_2 < \zeta \quad (4)$$

which ζ represents a given tolerance error.

By solving the above minimization problem, the sparse vector $\mathbf{b}$ shows the contribution of each sample in reconstructing the input signal $\mathbf{y}$. As the vector $\mathbf{b}$ is sparse a lot of its elements are zero and few of them have non-zero values. Samples in the database which are far from the input signal will have very small or zero