2014 22nd International Conference on Pattern Recognition

Locality Constrained Encoding Graph Construction and Application to Outdoor Object Classification

Fadi Dornaika[1,2], Alireza Bosaghzadeh[1]
[1] University of the Basque Country (UPV/EHU)
[2] IKERBASQUE, Basque Foundation for Science
San Sebastian, Spain

Houssam Salmane and Yassine Ruichek
IRTES-SeT
University of Technology Belfort-Montbeliard
Belfort, France

***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 that is 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 the 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. The experiments show that the proposed method can outperform competing methods.**

I. 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). Thus, the detector itself will provide the class. The second category of approaches estimates the class after a detection phase (e.g., [1]). 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 [2].

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 [3]. In [4], 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 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 [5], 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 the 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 (LBP) as image descriptors. 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. One of the advantages of the LBP descriptor is its constant and low dimensionality whatever the number of considered pixels from which low-level features are extracted. Furthermore, the LBP descriptor has been considered as a good discriminant feature. The paper is organized as follows. Section II briefly

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

1051-4651/14 $31.00 © 2014 IEEE
DOI 10.1109/ICPR.2014.429

describes some graph construction methods. Section III briefly describes the LBP descriptor. Section IV presents our proposed graph construction method. Section V presents the application of the proposed graph to the problem of semi-supervised categorization of outdoor objects.

II. Graph construction

A. Traditional graph construction (K-Nearest Neighbor and ϵ-Neighborhood Methods)

The classic graph construction 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, the K-Nearest Neighbor and ϵ-Neighborhood Methods can be used in order to find the neighbors of a datum. In both methods there is a function that defines the distance (similarity) of one input with respect to the others. 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. The KNN graph remains the more common approach since it is more adaptive to scale and data density while an improper threshold value in the ϵ-neighborhood graph could result in disconnected components or subgraphs in the dataset or even isolated singleton vertices.

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, [6] 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 $t \rightarrow \infty$ the weights will become 0 and 1. 0 when there is no connection and 1 when two nodes are connected. The authors in [7] proposed the use of inverse of distance as weight, $sim(\mathbf{x}_i, \mathbf{x}_j) = \frac{1}{||\mathbf{x}_i - \mathbf{x}_j||}$.

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

It is worth to mention that the KNN graph can be used as a first stage (providing a binary adjacency graph) in more sophisticated graph construction methods such as the method presented in [8].

B. ℓ_1 graph construction

Instead of building a graph in two different processes of adjacency construction and weight calculation, the authors in [5] 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 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

$$\min ||\mathbf{b}||_1 \;\; s.t. \;\; \mathbf{y} = \mathbf{X}\,\mathbf{b} \quad (2)$$

where vector $\mathbf{b} \in \mathbb{R}^N$ is the coefficient vector. Due to the presence of noise, Eq. ((2)) will become

$$\min ||\mathbf{b}||_1 \;\; s.t. \;\; \|\mathbf{y} - \mathbf{X}\,\mathbf{b}\|_2 < \zeta \quad (3)$$

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 coefficients. The more similar a signal in the database to the sample, the bigger it's coefficient. In this way the neighbors and their weights are calculated simultaneously.

There is also a formulation that can account for sparse outliers in the signals. This is given by:

$$\min ||\mathbf{b}||_1 + ||\mathbf{e}||_1 \;\; s.t. \;\; \mathbf{y} = \mathbf{X}\mathbf{b} + \mathbf{e} \quad (4)$$

The above problem can be casted into the form given in (2) by solving for the augmented vector $\mathbf{b}' = (\mathbf{b}^T, \mathbf{e}^T)^T$:

$$\min \|\mathbf{b}'\|_1 \quad s.t. \quad [\mathbf{X}\ \mathbf{I}]\,\mathbf{b}' = \mathbf{y} \quad (5)$$

Therefore, by using the above coding for every training sample $\mathbf{x}_i$ and calculating sparse vector $\mathbf{b}_i$, one can construct the $\mathbf{W}$ matrix using the set of computed vectors $(\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_N)$. A directed graph with an asymmetric weight matrix $\mathbf{W}$ can be constructed using two following formulas: Sparsity Induced Similarity (SIS) [9] or [5]. It was shown in [10] that both formulas have given similar performances.

$$\text{(SIS):}W_{ij} = \frac{max\{b_i(j), 0\}}{\sum_{j=1}^{N} max\{b_i(j), 0\}}, \quad \ell_1 \text{ graph} W_{ij} = |b_i(j)|$$

Alternatively the ℓ_1 can also be solved by the following least absolute shrinkage and selection operator (LASSO):

$$\mathbf{b} = \arg\min_{\mathbf{b}} ||\mathbf{y} - \mathbf{X}\,\mathbf{b}||_2^2 + \lambda\,||\mathbf{b}||_1 \quad (6)$$

C. 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

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

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

III. Local Binary Patterns

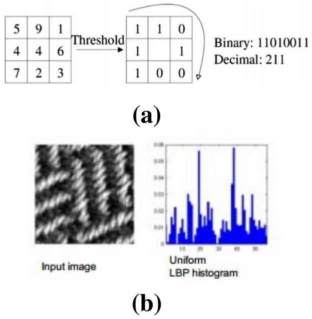

Fig. 1. **(a)** Example of basic LBP operator. **(b)** Example of LBP descriptor.

The original LBP operator labels the pixels of an image with decimal numbers, which are called LBPs or LBP codes that encode the local structure around each pixel [11]. It proceeds thus, as illustrated in Figure 1: Each pixel is compared with its eight neighbors in a neighborhood by subtracting the center pixel value; the resulting strictly negative values are encoded with 0, and the others with 1. For each given pixel, a binary number is obtained by concatenating all these binary values in a clockwise direction, which starts from the one of its top-left neighbor. The corresponding decimal value of the generated binary number is then used for labeling the given pixel. The histogram of LBP labels (the frequency of occurrence of each code) calculated over a region or an image can be used as a texture descriptor. The size of the histogram is 2^P since the operator $LBP(P, R)$ produces 2^P different output values, corresponding to 2^P different binary patterns formed by P pixels in the neighborhood. LBP methodology has been developed recently with large number of variations to improve performance in different applications. These variations focus on different aspects of the original LBP operator.

IV. Proposed LBP graph construction

Given N images (depicting different objects) $\mathbf{I}_1 \in \mathbb{R}^{d_1}, \mathbf{I}_2 \in \mathbb{R}^{d_2}, \ldots, \mathbf{I}_N \in \mathbb{R}^{d_N}$ and their LBP descriptors $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N \in \mathbb{R}^D$, we aim to build the graph G that models the pairwise similarities among images. We stress the fact that the images can have different sizes. However, their descriptors have the same size. In other words, $D = 2^P$ if all binary patterns are used in the histograms and $D = P\,(P-1)+3$ if only uniform patterns are used in the histogram estimation where P is the number of neighboring points used by the LBP operator.

We assume that each sample or image descriptor is represented by a sparse combination of the rest of the samples in the database through some coefficients. The obtained coefficients show the contribution of all samples in reconstructing a given one and demonstrate the similarity between them. These coefficients are then used to build the graph. Therefore, this assumption is the one adopted by ℓ_1 graph. However, we will show that our proposed method has similar or even better performance than that of ℓ_1 graph, yet with the clear advantage of being more efficient.

Our proposed method mainly relies on the Weighted Regularized Least Square minimization (WRLS) [12] that is based on ℓ_2 minimization. We stress the fact that, in [12], the WRLS was used in order to build a Collaborative Neighborhood Classifier (CNR) as an alternative to the recent ℓ_1 based Sparse Representation Classifier (SRC) [13], [14], [15]. In our method, there are two main differences with [12]. First, we address graph construction through the use of self-representation coefficients. Second, for estimating the graph weights, we introduce a Two Phase estimation scheme where the second WRLS phase uses only samples having high similarity and adaptively chosen without any predefined parameter. These two phases (described below) enforce the locality constraints in constructing the graph.

A. Weighted Regularized Least Square minimization

Assume we have a datum $\mathbf{y}$ and want to represent it by a linear combination of samples (or subset) of the dataset $\mathbf{X}$ as $\mathbf{y} = \mathbf{X}\,\mathbf{b}$, where $\mathbf{b}$ is a N dimensional vector containing the weights of all samples in the database in representing $\mathbf{y}$ and $\mathbf{X} = [\mathbf{x}_1\mathbf{x}_2 ... \mathbf{x}_N]$ ($D \times N$ matrix) is the data matrix. We assume that each sample $\mathbf{x}_i$ is normalized using its ℓ_2 norm. The general formula to solve the above equation will be

$$\min ||\mathbf{b}||_r \quad s.t. \quad \|\mathbf{y} - \mathbf{X}\mathbf{b}\|_2 < \epsilon \tag{7}$$

which tries to represent the datum $\mathbf{y}$ by the smallest $||\mathbf{b}||_r$ which represents the ℓ_r-norm of $\mathbf{b}$ (i.e., $||\mathbf{b}||_r = (\sum_j |b(j)|^r)^{1/r}$.

In our work, we use the ℓ_2 norm of residual error with sparsification. Our sparsification consists in two main modules. The first one uses a weighted regularization in which each unknown coefficient has an independent weight derived from the similarity between the test sample and the corresponding sample in the dataset. The second module uses a two phase WRLS where the second stage only uses samples having large coefficients and adaptively chosen without any predefined parameter. Unlike the ℓ_1 minimization, the ℓ_2 norm, Eq. (7) will have a closed form solution and it can be calculated in a more efficient way.

The unknown vector $\mathbf{b}$ can be calculated by minimizing the following criterion (WRLS):

$$\mathbf{b} = \arg\min_{\mathbf{b}} \frac{1}{2} \left(\quad \|\mathbf{y} - \mathbf{X}\mathbf{b}\|_2^2 + \sigma \sum_{j=1}^{N} p_j^2\, b_j^2 \right) \tag{8}$$

where p_j is a positive weight associated with example $\mathbf{x}_j$ (or equivalently b_j). The solution to Eq. (8) will reconstruct the input signal $\mathbf{y}$ by a combination of the space spanned by $\mathbf{X}$. In Eq. (8), the criterion has two terms: the reconstruction error and the weighted regularized term. Thus, σ is a small positive scalar that balances the two terms effect.

By having different values for p_i one can control the coefficients getting bigger or smaller. The bigger the p_j is the smaller the b_j would be and vice versa. By using simple linear algebra calculations, the solution to Eq. (8) has a closed form solution that is given by:

$$\mathbf{b} = \left(\mathbf{X}^T\mathbf{X} + \sigma\,\mathbf{P}\right)^{-1} \mathbf{X}^T\,\mathbf{y} \tag{9}$$

$\mathbf{P}$ is a diagonal matrix with elements $\mathbf{P}_{jj} = p_j$. In our work, we use the following formula:

$$p_j = 1 - \exp(-\|\mathbf{y} - \mathbf{x}_j\|^2) \tag{10}$$

B. Graph construction using Two Phase WRLS (TPWRLS)

We have seen above that the weighted regularization term imposes a kind of sparsification on the coefficients b_i due to the use of the weights. In this section, we will propose an additional sparsification module that acts by invoking two passes of the WRLS minimization process. The first pass considers the whole data as a dictionary for WRLS minimization. Once the vector of coefficients is estimated, a subset of examples will be selected from the original dictionary and used as the new dictionary for a second pass of WRLS. The selection will rely on the magnitude of the coefficients.

Once the first pass is achieved, we have an additional information provided by the estimated coefficients b_j. Indeed, $|b_j|$ provides a similarity measurement between the sample $\mathbf{y}$ and the example $\mathbf{x}_j$. Thus, our intuition is to keep the most similar examples and remove the remaining ones by exploiting the calculated vector $\mathbf{b}$. To do so, we compute an average similarity as $A = \frac{1}{N}\sum_{j=1}^{N} |b_j|$.

Let $\mathbf{X}_s$ be the data matrix formed by the selected examples (the ones whose $|b_j|$ is greater than the average similarity). In practice, we can use any threshold that belongs to $[A, b_{max}[$. Then, the vector $\mathbf{b}'$ associated with the selected examples will be solved using a formula similar to Eq. (9):

$$\mathbf{b}' = \left(\mathbf{X}_s^T\mathbf{X}_s + \sigma\,\mathbf{P}'\right)^{-1}\; \mathbf{X}_s^T\,\mathbf{y} \tag{11}$$

where the diagonal weights p'_j are given by:

$$p'_j = \frac{1}{|b_j|} \tag{12}$$

In order to avoid a very high disparity among the obtained p'_j weights, we normalized them using a unit variance normalization scheme. The above weights can reinforce the sparsification of the new estimated coefficients. Furthermore, the size of the data matrix $\mathbf{X}_s$ in the second phase is smaller than that of the whole data matrix $\mathbf{X}$ and, hence, the coefficients obtained in the second phase are sparser in comparison to those obtained in the first phase with the whole database. In the TPWRLS, the original N-vector $\mathbf{b}$ is set as follows. A non-selected example $\mathbf{x}_j$ will have $b_j = 0$ and a selected one will have the corresponding coefficient in the vector $\mathbf{b}'$ estimated by Eq. (11).

The detailed procedure for the TPWRLS graph construction is listed in Algorithm 1. This algorithm estimates the i^{th} row of the affinity matrix by coding the sample $\mathbf{x}_i$ w.r.t. to the set $\mathbf{X} - \{\mathbf{x}_i\}$. Note that the constructed graph is a directed graph, i.e., the weight matrix $\mathbf{W}$ is asymmetric.

```
Data: A given training sample set $\mathbf{X}$
Result: A weight matrix $\mathbf{W}$
Set the diagonal elements of $\mathbf{W}$ to zeros ;
for $i = 1, \ldots, N$ do
    Pick the sample $\mathbf{x}_i$ and form the data matrix $\mathbf{X}'$ = $\mathbf{X}$ - $\{\mathbf{x}_i\}$ ;
    Compute the $(N-1) \times (N-1)$ diagonal matrix $\mathbf{P}$ using Eq. (10) ;
    Calculate the $(N-1) \times 1$ vector $\mathbf{b}$ as
    $\mathbf{b} = (\mathbf{X}'^T\,\mathbf{X}' + \sigma\,\mathbf{P})^{-1}\mathbf{X}'^T\,\mathbf{x}_i$ ;
    Compute the average similarity for $\mathbf{x}_i$ as
    $\frac{1}{N-1}\sum_{j=1}^{N-1} |b_j|$ ;
    Set the selected samples $\mathbf{X}_s$ ($|b_j|$ are above the average similarity);
    Form the new diagonal weight matrix $\mathbf{P}'$ using Eq. (12) ;
    Calculate the vector $\mathbf{b}'$ as
    $\mathbf{b}' = \left(\mathbf{X}_s^T\mathbf{X}_s + \sigma\,\mathbf{P}_s\right)^{-1}\;\mathbf{X}_s^T\,\mathbf{x}_i$ ;
    Set the sparse vector $\mathbf{b}$ from $\mathbf{b}'$ ;
    for $j = 1, \ldots, N$ do
        if $i < j$ then
            Set $W_{ij} = |b_j|$
        else
            Set $W_{ij} = |b_{j-1}|$
        end
    end
end
```

Algorithm 1: TPWRLS graph construction.

V. Performance evaluation

We evaluate the proposed graph construction method for solving the outdoor object categorization using semi-supervised learning. These objects can be captured by either a surveillance camera or an onboard camera. We assume that the detection of the image regions of these objects is carried out by algorithms like the ones described in [16], [17] for surveillance camera scenarios and by the detection and tracking algorithms described in [9], [18] for onboard cameras. Unlike supervised schemes (e.g., [19]) where observations are classified individually, the semi-supervised framework offer the possibility to infer the category of one or more detected objects at the same time (obstacles in a single or several video frames). In this section, we will present the graph-based label propagation framework used for evaluation. Then, we will present the quantitative evaluation using different graph-construction schemes applied to the outdoor object categorization.

A. Graph-based Label Propagation

The graph-based label propagation method imposes that samples with high similarity should share similar labels. Using this idea, one can propagate the labels of known samples to unknown ones and infer their labels. This idea can be written as

$$\min_{\mathbf{y}_i, i=1,\ldots,N} \sum_{i,j} ||\mathbf{y}_i - \mathbf{y}_j||^2 W_{ij} \tag{13}$$

where $\mathbf{y}_i$ is the posterior probability of samples belonging to C different classes, namely, $y_i(c) = p(c|\mathbf{x}_i); c = 1, 2, \ldots, C$, and $\mathbf{W}$ is the similarity matrix associated with the graph. For a labeled sample $\mathbf{x}_i$, $y_i(c) = 1$ if $\mathbf{x}_i$ belongs to the c^{th} class; $y_i(c) = 0$, otherwise. Consider the data matrix $\mathbf{X} = [\mathbf{X}_l \; \mathbf{X}_u]$ (a $D \times (p+q)$ matrix) containing labeled and unlabeled data, and the label matrix $\mathbf{Y} = [\mathbf{Y}_l \; \mathbf{Y}_u]$ (a $C \times (p+q)$ matrix).

Given the data matrix as well as the known labels, $\mathbf{Y}_l$, the goal is to derive the labels of unlabeled samples, $\mathbf{Y}_u$. It can be shown that the matrix of unknown labels is given by:

$$\mathbf{Y}_u = -\mathbf{Y}_l \, \mathbf{L}_{lu} \, \mathbf{L}_{uu}^{-1} \tag{14}$$

where $\mathbf{L}_{lu}$ and $\mathbf{L}_{uu}$ are submatrices of the Laplacian matrix $\mathbf{L}$:

$$\mathbf{L} = \begin{pmatrix} \mathbf{L}_{ll} & \mathbf{L}_{lu} \\ \mathbf{L}_{ul} & \mathbf{L}_{uu} \end{pmatrix}$$

$\mathbf{L}$ is given by $\mathbf{L} = (\mathbf{D}_{row} - \mathbf{W}) + (\mathbf{D}_{col} - \mathbf{W}^T)$ where $\mathbf{D}_{row}$ and $\mathbf{D}_{col}$ are diagonal matrices whose elements are the row and column sums of $\mathbf{W}$ matrix, respectively.

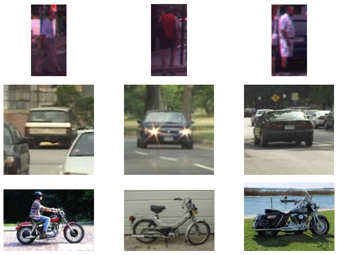

Fig. 2. Three classes of outdoor objects: Pedestrians, Cars/Vans, Motorbikes/bikes. Three examples in each class are shown.

B. Experimental results

We performed two groups of experiments. In the first group, the evaluation was carried using a whole dataset where the samples of each class were acquired using a given protocol. In the second group of experiments, we used labeled samples from a given dataset and unlabeled samples from another dataset.

For the first group of experiments, we used images depicting three classes (Pedestrian, cars/vans, and motorbikes) (See Fig. 2). The car and motorbike images were retrieved from PASCAL VOC2011 Example Images[2]. The pedestrian images are retrieved from CVC-01 Classification Dataset[3]. We gathered 450 images and used the whole set of images in a graph-based label propagation. Their LBP descriptors were retrieved using the uniform patterns. The compared graph construction methods (the affinity matrix used in Eq. (13)) are: KNN graph, LLE graph, ℓ_1 Graph [5], ℓ_1 (LASSO) Graph, WRLS, and the proposed TPWRLS. For ℓ_1 graph, the criterion (5) was optimized using the Matlab package provided by [20]. The LASSO optimization is carried out using the SLEP package[4].

In WRLS technique only the most similar samples for each row in the affinity matrix are used. Table I illustrates the correct labeling percentage obtained with different graph construction methods. These are average results that correspond to ten runs of the recognition algorithm with random partitions for labeled and unlabeled samples. For KNN and LLE graphs, this table depicts the best results obtained over a range of values for the neighborhood size. We can observe that (i) by increasing the number of labeled samples the increase in the success rate of all methods will not be very significant, (ii) most of the time the best results were obtained with the proposed TPWRLS method, and (iii) by increasing the radius of the LBP operator the recognition rate of all methods increases.

TABLE I. AVERAGE RECOGNITION RATE FOR DIFFERENT GRAPH CONSTRUCTION METHODS AND FOR DIFFERENT LABEL NUMBERS. THE GRAPHS WERE BUILT USING THE UNIFORM PATTERN LBP HISTOGRAMS WITH $P = 8$ AND $R = 1$ (UPPER TABLE), $R = 2$ (MIDDLE TABLE) AND $R = 3$ (LOWER TABLE).

$R = 1$	5	10	15	20	25	30	45
KNN	83.13	84.81	87.98	88.44	88.83	89.53	91.62
LLE	**83.31**	86.12	87.70	88.31	88.99	90.08	92.48
ℓ_1	82.53	90.48	91.75	92.33	93.36	94.14	96.54
$\ell_1(LASSO)$	78.78	81.62	82.59	82.46	82.93	83.28	84.22
$WRLS$	78.30	82.19	84.02	84.46	85.39	86.92	89.17
TPWRLS	83.22	**87.95**	**90.37**	**91.41**	**93.01**	**93.64**	**95.17**
$R = 2$							
KNN	86.78	91.05	92.64	93.15	94.16	94.64	96.38
LLE	**89.40**	**92.12**	93.90	94.44	94.96	95.64	96.57
ℓ_1	81.56	90.07	92.99	93.90	94.67	95.31	96.76
$\ell_1(LASSO)$	79.63	79.98	82.27	82.59	83.52	83.22	83.94
WRLS	84.14	86.48	90.20	90.92	92.21	92.17	93.78
TPWRLS	88.21	91.95	**95.09**	**96.21**	**96.48**	**97.06**	**97.65**
$R = 3$							
KNN	86.67	92.24	94.79	95.23	95.44	95.56	95.84
LLE	**91.68**	**93.14**	**95.06**	**95.38**	96.13	96.33	97.27
ℓ_1	78.53	91.81	93.21	94.69	96.08	96.47	97.27
$\ell_1(LASSO)$	84.09	89.86	89.58	90.44	91.12	91.44	92.51
WRLS	80.32	85.38	89.31	90.36	92.37	92.64	94.6
TPWRLS	87.49	92.62	94.35	94.9	**96.61**	**96.89**	**97.49**

TABLE II. CONFUSION MATRIX FOR ONE RUN. THE NUMBER OF LABELED SAMPLES WAS KEPT FIXED TO 10 IMAGES PER CLASS. THE UPPER TABLE CORRESPONDS TO THE KNN GRAPH AND THE LOWER TABLE TO THE PROPOSED TPWRLS GRAPH.

KNN	Car (140)	Pedestrian (140)	Moto (140)
Car (pred.)	139	1	13
Pedestrian (pred.)	0	129	1
Motor (pred.)	1	10	126

TPWRLS	Car (140)	Pedestrian (140)	Moto (140)
Car (pred.)	133	0	10
Pedestrian (pred.)	3	140	2
Moto (pred.)	4	0	128

Table II depicts the confusion matrix for a given run and for a fixed number of labeled samples (10 images per class). The upper part corresponds to the KNN graph and the lower part to the proposed TPWRLS graph. For both graph methods, we can observe that the motorbikes class was the most difficult class to be recognized. This can be explained by the high

[2] http://pascallin.ecs.soton.ac.uk/challenges/VOC/voc2012/examples/index

[3] http://www.cvc.uab.es/adas

[4] $http://www.public.asu.edu/\tilde{j}ye02/Software/SLEP$

TABLE III. CPU TIMES (IN SECONDS) ASSOCIATED WITH THE GRAPH CONSTRUCTION METHODS (450 IMAGES).

	KNN	LLE	ℓ_1	ℓ_1 (LASSO)	WRLS	TPWRLS
CPU time	0.04	2.1	16.8	0.67	0.45	2.1

variability of texture and color of motorbikes. The F1-measures for the three classes (car, pedestrians, and motor-bikes) are 0.948, 0.955, and 0.909, respectively when KNN graphs were used. These F1-measures become 0.940, 0.982, and 0.941 when the TPRWLS graph is used. This means that the use of the proposed graph method has improved the F1-measures. Table III illustrates the CPU time (seconds) associated with the graph construction (450 images) for KNN, LLE, ℓ_1, ℓ_1 (LASSO), WRLS and the proposed method, respectively. The label propagation took 20ms for 420 images for all methods. We used MATLAB running on an Intel Core i7 CPU at 2.93 Ghz and 8 GB of RAM. As can be seen the fastest approach was the KNN method. The best accurate results were obtained with the proposed method with 2.1 seconds. The WRLS technique needed about one fourth of that time but has not the same accuracy.

In the second group of experiments, we used samples acquired and detected by a developed video surveillance system whose goal is to detect dangerous situations in level crossing environments [21]. The detected moving objects are then used as the unlabeled samples. In this case, we use three classes (Pedestrian, cars/vans, and road). The labeled samples for the pedestrian and car classes are set to the data used in the first group of experiments (50 car images, 50 pedestrian images, 15 road images). For the unlabeled (test) samples, we used 50 car images, 150 pedestrian images, and 14 road images. The obtained recognition rates for KNN, LLE, ℓ_1 and the proposed TPWRLS graphs were 84.5%, 84.6%, 85.1% and 89.25%, respectively.

As can be seen the performance of all graph methods decreased a bit compared with the first group of experiments. Nevertheless, it can be considered as good since the unlabeled samples are associated to a completely different scenario.

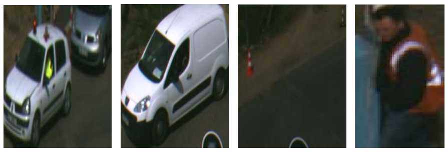

Fig. 3. Test outdoor objects captured by a video surveillance system.

VI. CONCLUSION

In this paper, we have proposed a locality constrained graph construction method that is based on Two Phase Weighted Regularized Least Square coding. The proposed data graph is used, in a semi-supervised context, in order to categorize detected objects in driving/urban scenes. It has been shown that our proposed graph construction method can give better performance than many state of the art graph construction methods including the most recent ℓ_1 graph.

ACKNOWLEDGMENT

This work was supported by the Spanish Government under the project TIN2010-18856 and the Franche Comté Regional Council (France) under project 2013C-06356.

REFERENCES

[1] H. Jung, Y. Lee, P. Yoon, I. Hwang, and J. Kim, "Sensor fusion based obstacle detection/classification for active pedestrian protection system," in *International Symposium on Visual Computing*, 2006.

[2] C. Zhang, J. Liu, C. Liang, Q. Huang, and Q. Tian, "Image classification using harr-like transformation of local features with coding residuals," *Signal Processing*, vol. 93, no. 8, pp. 2111–2118, August 2013.

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

[4] T. Jebara, J. Wang, and S. Chang, "Graph construction and b-matching for semi-supervised learning," in *ICML*, 2009.

[5] S. Yan and H. Wang, "Semi-supervised learning by sparse representation," in *SIAM International Conference on Data Mining*, 2009, pp. 792–801.

[6] M. Belkin and P. Niyogi, "Laplacian eigenmaps for dimensionality reduction and data representation," *Neural Comput.*, vol. 15, no. 6, pp. 1373–1396, Jun. 2003.

[7] C. Cortes and M. Mohri, "On transductive regression," in *NIPS*, 2006, pp. 305–312.

[8] F. Wang and C. Zhang, "Label propagation through linear neighborhoods," *IEEE Transactions on Knowledge and Data Engineering*, vol. 20, no. 1, pp. 55–67, 2008.

[9] H. Cheng, Z. Liu, and J. Yang, "Sparsity induced similarity measure for label propagation," in *IEEE International Conference on Computer Vision*, 2009.

[10] H. Wang, H. Huang, and C. Ding, "Image categorization using directed graphs," in *European Conference on Computer Vision*, 2010.

[11] D. Huang, C. Shan, M. Ardabilian, and Y. Wang, "Adaptive particle sampling and adaptive appearance for multiple video object tracking," *IEEE Trans. on Systems, Man, and Cybernetics-Part C: Applications and reviews*, vol. 41, no. 6, pp. 765–781, November 2011.

[12] J. Waqas, Z. Yi, and L. Zhang, "Collaborative neighbor representation based classification using l_2-minimization approach," *Pattern Recognition Letters*, vol. 34, no. 2, pp. 201–208, 2013.

[13] J. Wright, A. Yang, A. Ganesh, S. Sastry, and Y. Ma, "Robust face recognition via sparse representation," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 31, no. 2, pp. 210–227, 2009.

[14] X. Song, L. Jiao, S. Yang, X. Zhang, and F. Shang, "Sparse coding and classifier ensemble based multi-instance learning for image categorization," *Signal Processing*, vol. 93, no. 1, pp. 1–11, January 2013.

[15] H. Cheng, Z. Liu, L. Yang, and X. Chen, "Sparse representation and learning in visual recognition: Theory and applications," *Signal Processing*, vol. 93, no. 6, pp. 1408–1425, June 2013.

[16] K. Kim, T. H. Chalidabhongse, D. Harwood, and L. Davis, "Real-time foreground-background segmentation using codebook model," *Real-Time Imaging*, vol. 11, no. 3, pp. 172–185, 2005.

[17] W. Kim and C. Kim, "Background subtraction for dynamic texture scenes using fuzzy color histograms," *IEEE Signal Processing Letters*, vol. 13, no. 3, pp. 127–130, 2012.

[18] P. Pan and D. Schonfeld, "Visual tracking using high-order particle filtering," *IEEE Signal Processing Letters*, vol. 18, no. 1, pp. 51–54, 2011.

[19] Y. Han and G. Liu, "Efficient learning of sample-specific discriminative features for scene classification," *IEEE Signal Processing Letters*, vol. 18, no. 11, pp. 683–686, 2011.

[20] E. Candes and J. Romberg, *l1-magic: recovery of sparse signals via convex programming*, CALTECH, 2005, http://www.acm.caltech.edu/l1magic/.

[21] H. Salmane, Y. Ruichek, and L. Khoudour, "A novel evidence based model for detecting dangerous situations in level crossing environments," *Expert Systems with Applications*, vol. 41, pp. 795–810, 2014.