Pattern Recognition 46 (2013) 2711–2723

Contents lists available at SciVerse ScienceDirect

Pattern Recognition

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

M^{4}L: Maximum margin Multi-instance Multi-cluster Learning for scene modeling

Tianzhu Zhang [a,b,*], Si Liu [a,b], Changsheng Xu [a,b], Hanqing Lu [a,b]

[a] National Lab of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, PR China
[b] China-Singapore Institute of Digital Media, Singapore 119615, Singapore

ARTICLE INFO

Article history:
Received 14 October 2012
Received in revised form
10 January 2013
Accepted 15 February 2013
Available online 7 March 2013

Keywords:
Scene understanding
Maximum margin clustering
Multiple instance learning (MIL)
Gaussian Mixture Model (GMM)
Constrained Concave–Convex Procedure (CCCP)

ABSTRACT

Automatically learning and grouping key motion patterns in a traffic scene captured by a static camera is a fundamental and challenging task for intelligent video surveillance. To learn motion patterns, trajectory obtained by object tracking is parameterized, and scene image is spatially and evenly divided into multiple regular cell blocks which potentially contain several primary motion patterns. Then, for each block, Gaussian Mixture Model (GMM) is adopted to learn its motion patterns based on the parameters of trajectories. Grouping motion pattern can be done by clustering blocks indirectly, and each cluster of blocks corresponds to a certain motion pattern. For one particular block, each of its motion pattern (Gaussian component) can be viewed as an instance, and all motion patterns (Gaussian components) constitute a bag which can correspond to multiple semantic clusters. Therefore, blocks can be grouped as a Multi-instance Multi-cluster Learning (MIMCL) problem, and a novel Maximum Margin Multi-instance Multi-cluster Learning (M^{4}L) algorithm is proposed. To avoid processing a difficult optimization problem, M^{4}L is further relaxed and solved by making use of a combination of the Cutting Plane method and Constrained Concave–Convex Procedure (CCCP). Extensive experiments are conducted on multiple real world video sequences containing various patterns and the results validate the effectiveness of our proposed approach.

1. Introduction

Video scene understanding by unsupervised inference of motion patterns in static camera scenes is a very important task in visual surveillance. With the increasing of the surveillance systems, large amounts of video data are created every day, and it is difficult and time-consuming to label and organize these videos manually. Therefore, various methods have been made to understand the videos automatically [1–3]. By clustering the motion patterns, we can obtain information along which path or direction the vehicles or pedestrians should move or walk. Based on such information, it is very convenient to detect abnormal activities and meet the great needs for traffic management systems. However, the motion of pedestrians and vehicles is complex and their motion patterns are different from each other. Thus, automatically clustering the motion patterns for video understanding is a challenging problem in computer vision and pattern recognition.

The standard approach for analysis of video sequences involves four primary parts: (1) moving object detection; (2) object classification; (3) motion pattern learning and clustering and (4) activity analysis. There is a lot of progress made in each of the modules. For moving object detection, the Gaussian Mixture Model (GMM) [1] is frequently adopted to model background. For classification of objects into different categories (e.g. a vehicle, a person), scene context features (such as position, area in pixels, and velocity) [4] are used to cluster trajectories into different types (vehicles vs. pedestrians), and show effective performance by experimental results. However, due to low resolution, shadow, and different viewing angles, object classification only using these features is not enough in video surveillance. Therefore, a co-training based method [5] is proposed to train classifiers with multiple different kinds of features. For motion pattern learning and clustering, the most commonly used features are low level motion and appearance features [6–8]. Examples of such features include sparse or dense optical flows [9], spatiotemporal gradients [10], and object trajectories obtained after detection and tracking [11,5,3]. Based on different descriptions, various learning and clustering methods are adopted [12–14]. For activity analysis, existing approaches [1,4,5] use clustered motion patterns to recognize the abnormal activities of objects in video sequences.

* Corresponding author at: National Lab of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, PR China. Tel.: +86-10-82631857.
E-mail address: tzzhang10@gmail.com (T. Zhang).

0031-3203/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.patcog.2013.02.018

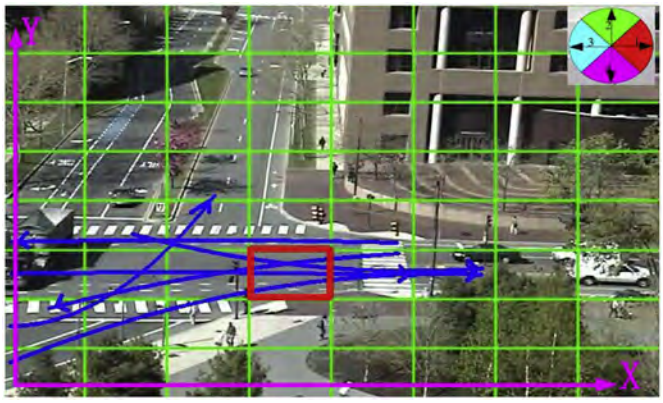

Fig. 1. Multiple primary motion patterns may exist in a certain block as shown in the red rectangle. Here, each trajectory represents a motion pattern. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this Article.)

In this work, we mainly focus on the third part about motion pattern learning and clustering, i.e., how to learn the motion pattern and design a suitable clustering algorithm. As shown in Fig. 1, object trajectory can be obtained by tracking algorithm [1], and each trajectory is a potential motion pattern in the scene image. Distribution of trajectories in a certain region can be viewed as a Gaussian distribution from the statistic point of view. The scene image can be divided into multiple cell blocks, and each block may have multiple distributions (multiple clusters of trajectories). Therefore, the Gaussian Mixture Model (GMM) is adopted to learn spatial distributions of trajectories in each block, and each Gaussian component is one of the underlying motion pattern. Then, how to cluster the motion patterns of all blocks in the scene image? Since each cluster of blocks is corresponding to a certain motion pattern, grouping motion patterns can be done by clustering blocks indirectly. For each block, each motion pattern (Gaussian component) can be viewed as an instance, and all motion patterns (Gaussian components or instances) constitute a bag which may contain multiple semantic clusters simultaneously. In this way, each block (bag) is associated with not only multiple instances but also multiple clusters. Therefore, we formulate the blocks grouping task as a Multi-instance Multi-cluster Learning (MIMCL) problem. There is little work directly dealing with this problem. Most of the existing clustering methods [15–18] are designed to solve traditional Single-instance Single-cluster Learning (SISCL) problems, and some methods [19,20] only consider a bag including one semantic cluster and deal with Multi-instance Single-cluster Learning (MISCL) problems. In many real-word cases, such as block clustering, a bag (block) may belong to more than one clusters. Therefore, it is unreasonable to adopt SISCL or MISCL formulation and assign a bag to only one cluster.

Considering the MIMCL formulation and the maximum margin clustering criterion [18,20,21], we propose a novel algorithm named M^{4}L, i.e. Maximum Margin Multi-instance Multi-cluster learning, to cluster motion patterns for video scene modeling. Briefly, M^{4}L assumes a linear model for each cluster, where the output of a bag on one cluster is set to be the maximum prediction scores of all the instances. Subsequently, the outputs on all possible clusters for all instances of a bag are adopted to define the margin. That is, for a certain cluster, the corresponding margin of a bag is defined by using the output of the most discriminative instance, and the margin of the bag with respect to the clustering system is set to be the minimum margin of the bag over all possible clusters. Obviously, each instance is adopted to determine the output on each possible cluster and the correlations between different clusters are also considered in the combination phase. Therefore, the connections between the instances and the clusters are explicitly exploited by M^{4}L. Compared with the existing approaches, the contributions of our work can be summarized as follows.

1. We formulate motion pattern clustering in video surveillance as a Multi-instance Multi-cluster Learning (MIMCL) problem, which provides a new perspective and is very suitable to group motion patterns.
2. To group motion patterns efficiently, we propose a novel clustering algorithm denoted as M^{4}L: Maximum Margin Multi-instance Multi-cluster Learning, which adopts the theory of support vector machine and aims at finding the maximum margin hyperplane to separate the data from different classes.
3. To solve the nonconvex M^{4}L algorithm, we make use of combination of Constrained Concave–Convex Procedure (CCCP) and the Cutting Plane method for efficient optimization solution.

The rest of the paper is organized as follows. In Section 2, we introduce some related work to this paper. The proposed approach is described in details in Section 3 including motion pattern learning and motion pattern clustering. Experimental results are reported and analyzed in Section 4. Finally, we conclude the paper with future work in Section 5.

2. Related work

The problem of scene modeling in visual surveillance is not new [4,1,2,5,22,14,23–25]. In general, the task is to lay out the structure of traffic scenes (e.g., roads, sidewalks, intersections), or learn motion patterns (e.g., pedestrian crossings, vehicles turning). The proposed work is an attempt to learn and cluster motion patterns from static camera videos without any user intervention. The most related work to our method is scene understanding in visual surveillance, and clustering relevant methods. We review the state-of-the-arts of these two topics, respectively.

2.1. Video scene understanding

In video surveillance, many methods attempt to learn motion patterns for video scene understanding. Stauffer and Grimson [1] use a real-time tracking algorithm in order to learn patterns of motion (or activity) from the obtained tracks. Due to the use of co-occurrence matrix from a finite vocabulary, these approaches are independent from the trajectory length. However, the vocabulary size is limited for effective clustering and time ordering is sometimes neglected. Hu et al. [2] generate trajectories using fuzzy k-means algorithms for detecting foreground pixels. Trajectories are then clustered hierarchically and each motion pattern is represented with a chain of Gaussian distributions. However, the number of clusters must be given manually and the data must be of equal length, which weakens the dynamic aspect. Wang et al. [4] propose a trajectory similarity measure to cluster the trajectories and then learn the scene model from trajectory clusters. Basharat et al. [26] learn patterns of motion as well as patterns of object motion and size. These approaches [4,26] are adapted to real-time applications and time-varying scenes because the number of clusters is not specified and they are updated over time. However, it is difficult to select a criterion for new cluster initialization that prevents the inclusion of outliers

and insures optimality. Zhang et al. [5] model pedestrians' and vehicles' trajectories as graph nodes and apply a graph-cut algorithm to group the motion patterns together. The drawback is the quality of the clusters which is dependent on the decision of how to split (merge) a set that is not generally reflected along the tree. Gryn et al. [27] introduce the direction map as a representation that captures the spatiotemporal distribution of motion direction across regions of interest in space and time. However, the direction map is able to capture only a single major orientation or motion modality at each spatial location of the scene. Yang et al. [8] propose a novel unsupervised approach for video scene understanding. They first quantize low-level features into words, then screen out useful words for motion pattern detection. Moreover, they use diffusion maps framework to detect motion patterns at different scales. Based on clustering of clips, they can also identify dominant motion patterns occurring in each clip cluster. Imran et al. [14] adopt a mixture model representation of salient patterns of optical flow, and present an algorithm for learning these patterns from dense optical flow in a hierarchical, unsupervised fashion. In addition, their representation avoids any quantization and loss of information in the feature space.

In this paper, we also attempt to learn motion patterns by trajectory analysis for video scene understanding. Different from many existing methods, our algorithm groups trajectories indirectly by clustering blocks first. Moreover, we formulate motion pattern clustering as a Multi-instance Multi-cluster Learning problem, which provides a new perspective for video scene understanding and is very suitable to group motion patterns.

2.2. Clustering relevant methods

Clustering [28,29] is one of the most fundamental research topics in both data mining and machine learning communities. It aims at dividing data into groups of similar objects, i.e. clusters. Many clustering methods have been proposed in the literature for last ten years, including k-means clustering [15], mixture models [15], spectral clustering [16,17], maximum margin clustering [18], and maximum margin multiple instance clustering (M^3IC) [20]. For video scene understanding, we propose a maximum margin multi-instance multi-cluster learning (M^4L) method, which is closely related to the learning frameworks of multi-instance learning [30], multi-label learning [31,32], maximum margin clustering [18], maximum margin multiple instance clustering (M^3IC) [20], maximum margin method for multi-instance multi-label learning (M^3MIML) [33] and traditional supervised learning.

Multi-instance learning [30], or multi-instance single-label learning (MISL), was proposed for investigation of drug activity prediction problem. The task of MISL is to learn a function from a set of MISL training examples, where a bag contains multiple instances and has a label, and instance has no label. After the seminal work of Dietterich et al. [30], numerous MISL learning algorithms have been proposed [34,35] and successfully applied to many applications especially in image categorization and retrieval [36,37]. More works on MISL can be found in [38].

Multi-label learning [31,32], or single-instance multi-label learning (SIML), originated from the investigation of text categorization problems. The task of SIML is to learn a function from a set of SIML training examples, where a bag is an instance, and has multiple labels. A number of SIML learning algorithms have been proposed by exploiting the relationships between different labels [39,40]. SIML techniques have been successfully applied to applications including text and image categorization [31,41]. More works on SIML can be found in [42].

Multi-instance multi-label learning (MIML) [43–45,33] is a newly proposed framework, where each example in the training set is associated with multiple instances as well as multiple labels. Many real-world problems involving ambiguous objects can be properly formalized under MIML. For instance, in image classification, an image generally contains several naturally partitioned patches, and each can be represented as an instance, while such an image can correspond to multiple semantic classes simultaneously, such as clouds, grassland and lions.

Xu et al. [18] proposed maximum margin clustering (MMC), which borrows the idea from the support vector machine theory and aims at finding the maximum margin hyperplane which can separate the data from different classes in an unsupervised way. Their experimental results showed that the MMC technique often obtains more accurate results than the conventional clustering methods. Therefore, considering the MISL, SIML, MIML, and maximum margin criterion, many clustering methods are proposed, such as, maximum margin multiple instance clustering (M^3IC) [20], maximum margin method for multi-instance multi-label learning (M^3MIML) [33].

According to the above definitions, it is clear that the traditional supervised learning (SISL) can be regarded as a degenerated version of either MISL or SIML. Furthermore, SISL, MISL and SIML are all degenerated versions of MIML. Our multi-instance multi-cluster learning (MIMCL) problem is different from multi-instance multi-label learning (MIML). The MIMCL is for clustering, and we do not know the label information. However, The MIML is for classification, and we know the label information used to learn model. Moreover, our maximum margin multi-instance multi-cluster Learning (M^4L) is different from maximum margin multiple instance clustering (M^3IC) [20] and maximum margin method for multi-instance multi-label learning (M^3MIML) [33]. That is because the M^3IC is a Multi-instance Single-cluster Learning (MISCL) method, and M^3MIML is a supervised learning method for classification.

3. Our scene modeling algorithm

Scene modeling in video surveillance is to learn and cluster motion patterns which are the moving paths and directions of objects, from observing the behavior of moving objects. The flowchart of our method is shown in Fig. 2. Given input video, after object detection and tracking, we adopt our scene modeling algorithm, and output semantic scene models. For video scene modeling, we take two steps. For the first step, trajectories obtained by object tracking are described in a parametric way (Section 3.1.1) and scene image is cut into multiple blocks. Each block is viewed as the distributions of trajectories (trajectories' parameters and moving directions of objects). Trajectory distribution in a block can be viewed as a Gaussian distribution from statistic point of view. Because each block may have multiple distributions, the Gaussian Mixture Model (GMM) is adopted to describe spatial distributions of

Fig. 2. The flowchart of our proposed scene modeling method.

trajectories for each block (Section 3.1.2). Each of the Gaussian components is one of the underlying motion patterns. For the second step, we cluster the blocks to group the motion patterns indirectly. We formulate the block grouping task as a Multi-instance Multi-cluster Learning (MIMCL) problem, where each block is associated with not only multiple instances (motion patterns) but also multiple clusters. Then, we propose a Maximum Margin Multi-instance Multi-cluster Learning method to cluster blocks to obtain their corresponding semantic regions.

Based on the semantic regions and their corresponding motion patterns, trajectories are further clustered. For each cluster of trajectories, primary trajectories are learnt by Mean-Shift algorithm [46]. As a result, video scene modeling are learnt. Next, we will introduce how to learn motion patterns for each block in Section 3.1, and show the detail about how to cluster the motion patterns in Section 3.2.

3.1. Motion pattern learning

3.1.1. Trajectory description

We implement real-time background subtraction and tracking as [1], so that moving objects can be reasonably separated from background and tracked over time. For each object, the corresponding trajectory is obtained as shown in Fig. 3. In the 2-D image coordinates, whose origin is on the bottom left corner, a trajectory can be described as $T=\{(x_1,y_1),(x_2,y_2),\ldots,(x_n,y_n)\}$. In general traffic scenes, trajectory of a vehicle is not complicated, for simplicity, we use quadratic curve ($y=a\times x^2+b\times x+c$) to describe the trajectory. For a tracked object, all points from start point to end point are collected to calculate the parameters (a, b, c) by least squares fit to the y values. Moving direction of object (v) is quantized into four directions as in Fig. 1. The parameters (a, b, c, v) are features of a trajectory.

Noisy points have a bad effect on the least squares algorithm to learn the parameters (a, b, c). Therefore, we have to preprocess each trajectory to delete these points. For a trajectory $T=\{(x_1,y_1),(x_2,y_2),\ldots,(x_n,y_n)\}$, distance between near points is defined as $d_i=\|x_{i+1}-x_i\|+\|y_{i+1}-y_i\|$, and change of distance Δd_i is $\|d_{i+1}-d_i\|$. The mean u and the standard deviation δ of $\{\Delta d_i, i=1,2,\ldots,n-2\}$ can be obtained. If $\|(\Delta d_i-u)/\delta\|>2.5$, the three points $(x_i,y_i),(x_{i+1},y_{i+1})$ and (x_{i+2},y_{i+2}) are considered to be unstable and deleted; otherwise, these points are saved and used to calculate the parameters. Instead of saving all points of a trajectory, the parametric way reduces storage space. In addition, it is convenient to extract motion patterns. Some results of trajectory description in a parametric way are showed in Fig. 3. Based on this description, blocks which one object has passed will be set to the same parameters (a, b, c, v) based on the object's trajectory. As a result, this description reflects the spatial distribution of trajectory. Finally, we can adopt the trajectory parameters to group the blocks with similar motion patterns.

3.1.2. Learning motion patterns by GMM algorithm

There are lots of motion patterns in traffic scenarios. These motion patterns can be obtained for each pixel in the scene image, but it is time and storage consuming. Since adjacent pixels in scene image have similar motion patterns, it is feasible to cut the scene image into $R\times C$ relatively small blocks as shown in Fig. 1 and learn these motion patterns based on each block.

Objects can be classified into vehicles or pedestrians, and there are two types of trajectories. One belongs to vehicles, and the other belongs to pedestrians. For each type of trajectory, the motion patterns of each block can be viewed as Gaussian distributions from statistic point of view. Because each block may contain many motion patterns, we adopt the multiple Gaussian models to represent them. There are four advantages to learn motion patterns by the GMM algorithm: (1) Multiple Gaussian models are enough to describe each block which may contain many various motion patterns. This is because the number of traffic rules is limited, which causes the number of motion patterns in each small block to be limited. (2) Outlier trajectories can be removed by updating the weight of Gaussian model, hence primary motion patterns can be learnt from long-term observations. (3) The weight of Gaussian model can be viewed as the importance of its corresponding motion pattern, hence the number of important activities will be known. (4) The computational cost is low.

Our algorithm can be described as follows. For each type of trajectory, each block in the scene is modeled by a mixture of K Gaussian distributions for trajectory parameters. For a certain block, the series of trajectories $\{T_t=(a_t,b_t,c_t,v_t)\}_{t=1}^{N}$ which have passed the block are obtained. Here (a_t, b_t, c_t, v_t) are parameters of a trajectory T_t, and are used to learn the parameter distribution of the block. The probability that the block has a value of T_t at time t can be written as

$$P(T_t)=\sum_{i=1}^{K} w_{i,t}\times\eta(T_t,u_{i,t},\Sigma_{i,t}), \tag{1}$$

where $w_{i,t}$ is the weight parameter of the ith Gaussian component at time t, $\eta(T_t,u_{i,t},\Sigma_{i,t})$ is the ith Normal distribution of component with mean $u_{i,t}$ and covariance $\Sigma_{i,t}$. Here $\Sigma_{i,t}$ is assumed to be diagonal matrix. Although this is certainly not the case, the assumption allows us to avoid a costly matrix inversion at the expense of some accuracy.

$$u_{i,t}=(u_{i,t}^a,u_{i,t}^b,u_{i,t}^c)^T \tag{2}$$

$$\sigma_{i,t}=(\sigma_{i,t}^a,\sigma_{i,t}^b,\sigma_{i,t}^c)^T \tag{3}$$

$$\Sigma_{i,t}^{1/2}=\begin{pmatrix}\sigma_{i,t}^a & 0 & 0\\ 0 & \sigma_{i,t}^b & 0\\ 0 & 0 & \sigma_{i,t}^c\end{pmatrix} \tag{4}$$

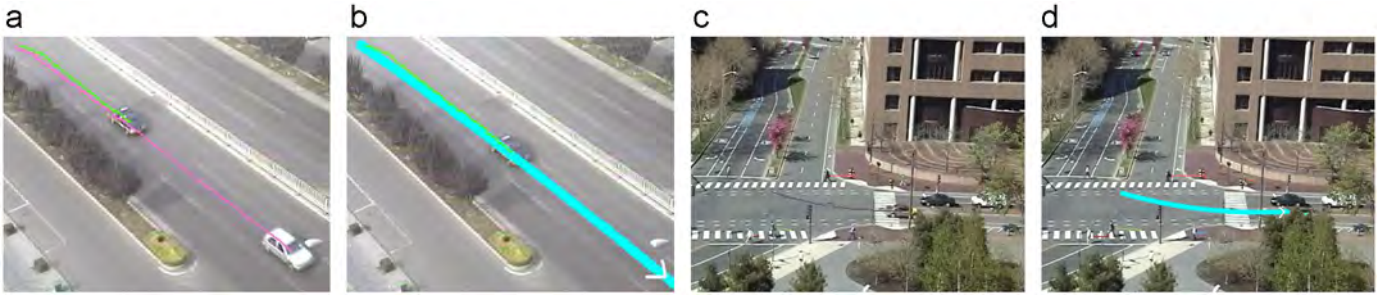

Fig. 3. Examples of trajectory fitting. The white arrows are the moving directions of objects. (b) and (d) are the trajectory fitting results corresponding to two trajectories (a) and (c), respectively.

The K distributions are ordered based on the fitness value $w_{i,t}$. Parameters u and σ for unmatched distributions remain the same. The first Gaussian component that matches the test trajectory will be updated by the following update equations:

$$w_{i,t} = (1-\alpha)w_{i,t-1} + \alpha(M_{i,t}) \tag{5}$$

$$u_{i,t} = (1-\rho)u_{i,t-1} + \rho T_t \tag{6}$$

$$\sigma_{i,t}^2 = (1-\rho)\sigma_{i,(t-1)}^2 + \rho(T_t-u_{i,t})^T(T_t-u_{i,t}) \tag{7}$$

$$\rho = \alpha\eta(T_t|u_{i,t},\sigma_{i,t}), \tag{8}$$

where $M_{i,t}$ is 1 for the model which matched and 0 for the remaining models, $1/\alpha$ defines the time constant which determines change. If none of the K distributions matches the trajectory, the component with the minimum weight is replaced by a distribution with the current value (a_t, b_t, c_t) as its mean, the v_t as its moving direction, an initially high variance, and a low weight parameter. In our experiments, the number of primary motion pattern of a small block is no more than 3 in the traffic scene, therefore, K is manually set to be 3 for simplicity. For our method, this parameter is not critical to decide the number of motion patterns in each block. This is because we have the weight $(w_{i,t})$ for each Gaussian component (motion pattern), which can help us decide the importance of each motion pattern and the number of motion patterns in each block. α is manually set to be 0.1, the initial high variance of (a, b, c) are (0.05, 0.2, 20), the low weight is 0.05. When we learn motion patterns for each block, we do not update the v_t based on the GMM algorithm for each motion pattern. We adopt a simple way to obtain the motion direction for each motion pattern by counting the four possible directions.

To improve the motion pattern learning, outlier trajectories must be removed. Usually these are noisy trajectories caused by tracking or classification errors, anomalous trajectories, e.g., a car drives out of the way, or some pedestrians roaming between different paths. In visual surveillance, these may be of particular interest, and as expected our algorithm can detect them. For a scene, the GMM algorithm is adopted to learn motion patterns for a long time. If a trajectory's parameters are similar with the motion patterns of blocks which the object has passed, and the motion patterns have a high weight, the trajectory is received to update the GMM model. Otherwise, the trajectory is viewed as a noisy trajectory and deleted.

After the motion pattern learning, each block has K Gaussian distributions, and each of the Gaussian components is one of the underlying motion patterns. For a scene, all motion patterns ($\mathbb{G}$) learnt by GMM algorithm are collected into $\mathbb{G} = \{\vec{g}_{ij}^{\hat{k}} | i = 1,2,\ldots,R, j = 1,2,\ldots,C, \hat{k} = 1,\ldots,K\}$, where $\vec{g}_{ij}^{\hat{k}} = (a_{ij}^{\hat{k}}, b_{ij}^{\hat{k}}, c_{ij}^{\hat{k}}, v_{ij}^{\hat{k}})^T$ is the $\hat{k}$ th motion pattern of the block (i, j). In a block, each Gaussian component $(\vec{g}_{ij}^{\hat{k}} = (a_{ij}^{\hat{k}}, b_{ij}^{\hat{k}}, c_{ij}^{\hat{k}}, v_{ij}^{\hat{k}})^T)$is viewed as an instance, and all Gaussian components are composed of a bag. Next, we introduce how to use the proposed M^{4}L algorithm to cluster the blocks to group motion patterns.

3.2. Motion pattern clustering

3.2.1. Problem description

Up to now, the problem is how to cluster motion patterns. In a scene image, some blocks may contain multiple different motion patterns, and our aim is to cluster the blocks which have a certain motion pattern. After motion learning as introduced in Section 3.1.2, each block generally contains several primary motion patterns, and each pattern can be viewed as an instance. All motion patterns (instances) of a block are constituted of a bag. For the bag, it may belong to multiple semantic clusters simultaneously. Then, the motion pattern clustering problem is transformed into grouping blocks, which is a standard MIMCL problem.

For mathematical description, suppose we are given a set of n bags, $\{B_i, i = 1,2,\ldots,n\}$, where $n = R \times C$ is the number of blocks, and $B_i = \{\vec{g}_{\tilde{i}\tilde{j}}^{\hat{k}} | \hat{k} = 1,\ldots,K\}$ contains the K motion patterns of the block $(\tilde{i},\tilde{j})$. The instances (motion patterns) in the bag B_i are denoted as $B_i = \{B_{i1}, B_{i2}, \ldots, B_{in_i}\}$, where n_i is the total number of instances in this bag and is equal to K. Our goal is to group these given bags (blocks) into k clusters, such that the semantic concepts in different clusters can be distinct from each other and each bag may be contained in multiple clusters.

3.2.2. Formulation

Given a sample $(B_i; \Omega_i)$, let Ω_i denote all possible clusters for B_i. For each cluster $p \in \{1,2,\ldots,k\}$, we define a weight vector w_p. M^{4}L assumes that the p-th cluster belongs to Ω_i if one instance B_{ij} in bag B_i has the maximum prediction with respect to w_p. That is

$$\Omega_i = \left\{ p | p = \arg\max_{p \in \{1,2,\ldots,k\}} (w_p^T B_{ij}), B_{ij} \in B_i \right\} \tag{9}$$

The Ω_i denotes that the bag B_i may belong to multiple clusters. We can define the bag margin of a bag B_i on the p-th cluster $(p \in \Omega_i)$ as

$$BM_p = \max_{j \in B_i} (w_p^T B_{ij} - w_{p^*}^T B_{ij}) \tag{10}$$

where, $p^* = \arg\max_{q \setminus p}(w_q^T B_{ij})$. Throughout this paper, "\" means ruling out. Therefore, this definition can also be written as: $p^* = \arg\max_{q \neq p}(w_q^T B_{ij})$. It is obvious that BM_p is determined by the most "discriminative" instance for cluster p. Then, the margin of the bag B_i with respect to the clustering system is set to be the minimum margin of $(B_i; \Omega_i)$ over all possible clusters

$$\min_{p \in \Omega_i} \max_{j \in B_i} (w_p^T B_{ij} - w_{p^*}^T B_{ij}) \tag{11}$$

Compared with the maximum margin multiple instance clustering (M^{3}IC) [20] method, we can see the definitions of bag margin are different. In [20], the authors adopt Eq. (10) as the bag margin, which just describes one possible cluster. However, for our problem, the bag margin is defined as Eq. (11), which represent the margin over all possible clusters.

In addition, as shown in [18,20], it is necessary to add a constraint to keep the cluster balance, which can avoid a trivially optimal solution by assigning all patterns to the same class. In this case, the resultant margin will be infinite. Moreover, it can also separate a single outlier or a very small group of samples from the rest of the data. Therefore, we consider the following equation to alleviate such trivial solutions.

$$-l \leq \sum_{i=1}^{n} \sum_{j \in B_i} \frac{1}{n_i} w_p^T B_{ij} - \sum_{i=1}^{n} \sum_{j \in B_i} \frac{1}{n_i} w_q^T B_{ij} \leq l \tag{12}$$

Considering these issues, M^{4}L can then be formulated as

$$\begin{aligned} \min_{w_1,\ldots,w_k,\xi_i \geq 0} \quad & \frac{1}{2}\sum_{p=1}^{k} \|w_p\|^2 + \frac{C}{n}\sum_{i=1}^{n} \xi_i \\ \text{s.t.} \quad & i = 1,\ldots,n, \\ & \Omega_i = \left\{ p | p = \arg\max_{p \in \{1,2,\ldots,k\}} (w_p^T B_{ij}), B_{ij} \in B_i \right\} \\ & \min_{p \in \Omega_i} \max_{j \in B_i} (w_p^T B_{ij} - w_{p^*}^T B_{ij}) \geq 1 - \xi_i \\ & \forall p,q \in \{1,2,\ldots,k\} \end{aligned}$$

$$-l \le \sum_{i=1}^{n}\sum_{j\in B_i}\frac{1}{n_i}w_p^T B_{ij} - \sum_{i=1}^{n}\sum_{j\in B_i}\frac{1}{n_i}w_q^T B_{ij} \le l \tag{13}$$

However, it is difficult to solve the optimization problem (13). This is because the convexity of $w_{p^*}^T B_{ij}$ is unknown. To tackle this problem, we replace $w_{p^*}^T B_{ij}$ with $mean_{q\setminus p} w_q^T B_{ij}$. The "mean" function calculates the average value of the input function with respect to the subscript variable. Therefore, Eq. (11) turns to

$$\min_{p\in\Omega_i}\left(\frac{k}{k-1}\max_{j\in B_i}\left(w_p^T B_{ij} - mean_q w_q^T B_{ij}\right)\right) \tag{14}$$

For simplicity, we define two concatenated vectors to consider all clusters together, which are $\tilde{w}=[w_1^T, w_2^T, \ldots, w_p^T, \ldots, w_k^T]^T$ and $B_{ij(p)}=[0,0,\ldots,B_{ij}^T,\ldots,0]^T$, where 0 is a $1\times d$ zero vector, and d is the dimension of B_{ij}. In $B_{ij(p)}$, only the $(p-1)d+1$ to pd-th elements are nonzero and equal to B_{ij}. Then, we have $\tilde{w}^T B_{ij(p)} = w_p^T B_{ij}^T$. Combining with Eq. (14), the formulation (13) can be transformed to

$$\begin{aligned}
&\min_{\tilde{w},\xi_i\ge 0} \quad \frac{1}{2}\|\tilde{w}\|^2 + \frac{C}{n}\sum_{i=1}^{n}\xi_i\\
&\text{s.t.} \quad i=1,\ldots,n,\\
&\quad \Omega_i = \left\{p \,|\, p = \arg\max_{p\in\{1,2,\ldots,k\}} (\tilde{w}^T B_{ij(p)}), j\in B_i\right\}\\
&\quad \min_{p\in\Omega_i}\left(\frac{k}{k-1}\max_{j\in B_i}(\tilde{w}^T B_{ij(p)} - mean_q \tilde{w}^T B_{ij(q)})\right) \ge 1-\xi_i\\
&\quad \forall p,q\in\{1,2,\ldots,k\}\\
&\quad -l \le \sum_{i=1}^{n}\sum_{j\in B_i}\frac{1}{n_i}\tilde{w}^T\left(B_{ij(p)} - B_{ij(q)}\right) \le l
\end{aligned} \tag{15}$$

In the following subsections, we will introduce how to solve Eq. (15) by use of Cutting Plane Optimization and CCCP.

3.2.3. Cutting plane optimization

To simplify the notation, we define three functions: $g(\tilde{w},i,j,p) = \tilde{w}^T B_{ij(p)} - mean_q \tilde{w}^T B_{ij(q)}$, $h(\tilde{w},i,p) = k/(k-1)\max_{j\in B_i} g(\tilde{w},i,j,p)$ and $f(\tilde{w},i) = \min_{p\in\Omega_i} h(\tilde{w},i,p)$.

For Eq. (15), there are n slack variables ξ_i. To solve it efficiently, we first derive the 1-slack form of Eq. (15) as in [47]. More specifically, we introduce a single slack variable $\xi\ge 0$ and reformulate Eq. (15) into the following optimization problem:

$$\begin{aligned}
&\min_{\tilde{w},\xi\ge 0} \quad \frac{1}{2}\|\tilde{w}\|^2 + C\xi\\
&\text{s.t.} \quad i=1,\ldots,n \quad \forall c\in\{0,1\}^n\\
&\frac{1}{n}\sum_{i=1}^{n} c_i f(\tilde{w},i) \ge \frac{1}{n}\sum_{i=1}^{n} c_i - \xi \quad \forall p,q\in\{1,2,\ldots,k\}\\
&\quad -l \le \sum_{i=1}^{n}\sum_{j\in B_i}\frac{1}{n_i}\tilde{w}^T(B_{ij(p)} - B_{ij(q)}) \le l
\end{aligned} \tag{16}$$

It has been proved that the solution to Eq. (16) is identical to Eq. (15) with $\xi = (1/n)\sum_{i=1}^{n}\xi_i$ in paper [47]. Although the number of variables in Eq. (16) is greatly reduced, the number of constrains is increased from n to 2^n. Our proposed algorithm targets to find a small subset of constraints from the whole set of constraints in Eq. (16) that ensures a sufficiently accurate solution. Specifically, we employ an adaptation of the cutting plane algorithm [48] to solve the maximum margin clustering problem, where we construct a nested sequence of successively tighter relaxations of Eq. (16) as follows:

$$\begin{aligned}
&\min_{\tilde{w},\xi\ge 0} \quad \frac{1}{2}\|\tilde{w}\|^2 + C\xi\\
&\text{s.t.} \quad i=1,\ldots,n \quad \forall c\in\Psi\\
&\frac{1}{n}\sum_{i=1}^{n} c_i f(\tilde{w},i) \ge \frac{1}{n}\sum_{i=1}^{n} c_i - \xi\\
&\forall p,q\in\{1,2,\ldots,k\}\\
&-l \le \sum_{i=1}^{n}\sum_{j\in B_i}\frac{1}{n_i}\tilde{w}^T(B_{ij(p)} - B_{ij(q)}) \le l
\end{aligned} \tag{17}$$

Moreover, [47] proves theoretically that we can always find a polynomially sized subset of constraints, with which the solution of the relaxed problem fulfills all constraints from Eq. (16) up to a precision of ε, i.e.

$$\forall c\in\{0,1\}^n: \quad \frac{1}{n}\sum_{i=1}^{n} c_i f(\tilde{w},i) \ge \frac{1}{n}\sum_{i=1}^{n} c_i - (\xi+\varepsilon) \tag{18}$$

That is, the remaining exponential number of constraints are guaranteed to be violated by no more than ε, without the need for explicitly adding them to the optimization problem. Eq. (17) starts with an empty constraint subset Ψ and it computes the optimal solution subject to the constraints in Ψ. After getting the solution $\tilde{w}$, the most violated constraint can be computed as

$$c_i = \begin{cases} 1, & \text{if } f(\tilde{w},i)\le 1\\ 0, & \text{otherwise}\end{cases} \tag{19}$$

The algorithm then adds the most violated constraint into the subset Ψ. In this way, we construct a successive strengthening approximation of the original problem (16) by a cutting plane that cuts off the current optimal solution from the feasible set [48]. The algorithm stops when no constraint in (17) is violated by more than ε.

3.2.4. Optimization via the CCCP

In each iteration of the cutting plane algorithm, we need to solve problem (17) to obtain the optimal classifying hyperplane under the current working constraint set Ψ. Although the objective function in (17) is convex, the constraints are not, which makes Eq. (17) difficult to solve. However, the constraint can be viewed as the sum of a convex function and a concave function. Therefore, it is very suitable for our problem using the Constrained Concave–Convex Procedure (CCCP), which is just designed to solve the optimization problems with a concave convex objective function with concave convex constraints [49]. Given an initial point $\tilde{w}^{(0)}$, CCCP iteratively computes $\tilde{w}^{(t+1)}$ from $\tilde{w}^{(t)}$ by replacing $f(\tilde{w},i)$ with its first order Taylor expansions at $\tilde{w}^{(t)}$, and solving the resulting quadratic programming problem, until convergence. In the following, we will show how to utilize CCCP to solve Eq. (17).

To use CCCP, we should first calculate the gradient and the first-order Taylor expansion of $f(\tilde{w},i)$ at $\tilde{w}^{(t)}$. Since $f(\tilde{w},i)$ is a non-smooth functions w.r.t. $\tilde{w}$, we replace its gradient with its subgradient as follows:

$$\begin{aligned}
&\frac{\partial f(\tilde{w},i)}{\partial \tilde{w}}\Big|_{\tilde{w}=\tilde{w}^{(t)}}\\
&\quad = \frac{\partial f(\tilde{w},i)}{\partial h(\tilde{w},i,p)}\times\frac{\partial h(\tilde{w},i,p)}{\partial g(\tilde{w},i,j,p)}\times\frac{\partial g(\tilde{w},i,j,p)}{\partial \tilde{w}}\Big|_{\tilde{w}=\tilde{w}^{(t)}}\\
&\quad = \sum_{p\in\Omega_i^{(t)}}\left(a_{ip}^{(t)}\sum_{j\in B_i}\left(b_{ij}^{(t)}\times\frac{k}{k-1}\left(B_{ij(p)} - 1/k\sum_{p=1}^{k}B_{ij(p)}\right)\right)\right)
\end{aligned} \tag{20}$$

where

$$a_{ip}^{(t)} = \begin{cases} 1 & \text{if } p = \arg\min_{p\in\Omega_i^{(t)}} h(\tilde{w}^{(t)},i,p)\\ 0 & \text{otherwise}\end{cases}$$

and

$$b_{ij}^{(t)}=\begin{cases}1 & \text{if } j=\arg\max_{j\in B_i} g(\tilde{w}^{(t)},i,j,p)\\ 0 & \text{otherwise.}\end{cases}$$

Given an initial point w^0, the CCCP computes w^{t+1} from w^t by replacing $f(\tilde{w},i)$ in the constraint with its first-order Taylor expansion at w^t, i.e.

$$\begin{aligned}f(\tilde{w},i)&=f(\tilde{w}^{(t)},i)+(\tilde{w}-\tilde{w}^{(t)})^T\frac{\partial f(\tilde{w},i)}{\partial\tilde{w}}\Big|_{\tilde{w}=\tilde{w}^{(t)}}\\&=\tilde{w}^T\frac{\partial f(\tilde{w},i)}{\partial\tilde{w}}\Big|_{\tilde{w}=\tilde{w}^{(t)}}\\&\quad+\min_{p\in\Omega_i}\left(\frac{k}{k-1}\max_{j\in B_i}(\tilde{w}^TB_{ij(p)}-mean_q\tilde{w}^TB_{ij(q)})\right)-(\tilde{w}^{(t)})^T\\&\quad\times\sum_{p\in\Omega_i^{(t)}}\left(a_{ip}^{(t)}\sum_{j\in B_i}\left(b_{ij}^{(t)}\times\frac{k}{k-1}\left(B_{ij(p)}-1/k\sum_{p=1}^{k}B_{ij(p)}\right)\right)\right)\\&=\tilde{w}^T\frac{\partial f(\tilde{w},i)}{\partial\tilde{w}}\Big|_{\tilde{w}=\tilde{w}^{(t)}}\end{aligned}\tag{21}$$

where

$$\Omega_i^{(t)}=\left\{p\,|\,p=\arg\max_{p\in\{1,2,\ldots,k\}}((\tilde{w}^{(t)})^TB_{ij(p)})\quad\forall\, j\in B_i\right\}$$

By substituting the above first-order Taylor expansion (21) into Eq. (17), we obtain the following quadratic programming (QP) problem:

$$\begin{aligned}&\min_{\tilde{w},\xi\ge0}\quad\frac{1}{2}\|\tilde{w}\|^2+C\xi\\&\text{s.t.}\quad i=1,\ldots,n\quad\forall c\in\Psi\\&\frac{1}{n}\tilde{w}^T\sum_{i=1}^{n}c_i\frac{\partial f(\tilde{w},i)}{\partial\tilde{w}}\Big|_{\tilde{w}=\tilde{w}^{(t)}}\ge\frac{1}{n}\sum_{i=1}^{n}c_i-\xi\\&\quad\forall p,q\in\{1,2,\ldots,k\}\\&-l\le\sum_{i=1}^{n}\sum_{j\in B_i}\frac{1}{n_i}\tilde{w}^T(B_{ij(p)}-B_{ij(q)})\le l\end{aligned}\tag{22}$$

Following the CCCP, the obtained solution w^t from this problem is then used as w^{t+1} and the iteration continues until convergence. Eq. (22) can be solved by any state-of-the-art QP solvers, such as Mosek [50]. However, there are many constraints. For simplicity, we try to use the Wolfe dual of Eq. (22) to obtain the optimum solution.

First, we define four variables as follows: $\|c_m\|_1=(1/n)\sum_{i=1}^{n}c_{mi}, m=1,\ldots,|\Psi|$; $z_m=(1/n)\sum_{i=1}^{n}c_{mi}\partial f(\tilde{w},i)/\partial\tilde{w}|_{\tilde{w}=\tilde{w}^{(t)}}$, $m=1,\ldots,|\Psi|$; $b_0=\sum_{i=1}^{n}\sum_{j\in B_i}(1/n_i)(B_{ij(p)}-B_{ij(q)}), p,q\in\{1,\ldots,k\}$ and $x_0=[z_m-b_0b_0]$.

The Wolfe dual of Eq. (22) is

$$\begin{aligned}\max_{\lambda\ge0,\alpha\ge0,\beta\ge0}\quad&-\frac{1}{2}\left(\sum_{m=1}^{|\Psi|}\lambda_mz_m-\sum_{p=1}^{k}\sum_{q=1}^{k}\alpha_{pq}b_0+\sum_{p=1}^{k}\sum_{q=1}^{k}\beta_{pq}b_0\right)^2\\&+\sum_{m=1}^{|\Psi|}\lambda_m\|c_m\|_1-\sum_{p=1}^{k}\sum_{q=1}^{k}(\alpha_{pq}+\beta_{pq})l\\\text{s.t.}\quad&\sum_{m=1}^{|\Psi|}\lambda_m\le C\end{aligned}\tag{23}$$

To solve Eq. (23), we use the matlab quadratic programming function "quadprog". That is

$$\begin{aligned}&\min_x\quad Q(x)=\frac{1}{2}x^THx+f^Tx\\&\text{s.t.}\quad Aqp^Tx\le bqp\quad x\ge0,\end{aligned}\tag{24}$$

where,

$$H=x_0^Tx_0, bqp=C,$$

$$f=(\underbrace{-\|c_1\|_1,\ldots,-\|c_{|\Psi|}\|_1}_{|\Psi|},\underbrace{l,\ldots,l}_{2k^2})^T,$$

$$Aqp=(\underbrace{1,\ldots,1}_{|\Psi|},\underbrace{0,\ldots,0}_{2k^2})^T,$$

$$x=(\lambda_1,\ldots,\lambda_{|\Psi|},\alpha_{11},\ldots,\alpha_{pq}\ldots,\alpha_{kk},\beta_{11},\ldots,\beta_{kk})^T.$$

After obtaining the solution for Eq. (24), the solution of Eq. (22) can be obtained as follows:

$$\begin{aligned}\tilde{w}&\leftarrow\sum_{m=1}^{|\Psi|}\lambda_m\frac{1}{n}\sum_{i=1}^{n}c_{mi}\frac{\partial f(\tilde{w},i)}{\partial\tilde{w}}\Big|_{\tilde{w}=\tilde{w}^{(t)}}\\&\quad-\sum_{p=1}^{k}\sum_{q=1}^{k}(\alpha_{pq}-\beta_{pq})b_0\\\xi&\leftarrow\frac{Q(x)-\frac{1}{2}\tilde{w}^T\tilde{w}}{C}\end{aligned}\tag{25}$$

Based on the learned $\tilde{w}$, we can use Eq. (9) to obtain the clusters Ω_i for the bag B_i. The whole process of our solution is summarized in Algorithm 1.

Algorithm 1. The proposed M^{4}L algorithm.

Input:
Data: bags $\{B_1,\ldots,B_n\}$
Parameters: cluster number k, balance parameter l, regularization constant C, CCCP solution precision ε_1, cutting plane solution precision ε
Cutting Plane Iterations:
Obtain $\{B_{ij(p)}\}$ based on data.
Initialize $\tilde{w}^t$, $\Psi=\emptyset$, $t=0$, and $\forall 1\le i\le n$, $c_i=0$.
Solve Eq. (22) and add constraints into Ψ by (19).
while Eq. (18) is true with all the selected constraint c **do**
  $t=t+1$
  CCCP Iterations:
  Initialize $\tilde{w}^{t_s}=\tilde{w}^{t-1}$, $s=0$, $\Delta Q=10^{-2}$, $Q^{-1}=10^{-2}$
  while $\Delta Q/Q^{s-1}>\varepsilon_1$ **do**
    Get the solution $\tilde{w}^{t_{s+1}}$ of problem (22) under Ψ by

$$\tilde{w}^{(t_{s+1})}\leftarrow\sum_{m=1}^{|\Psi|}\lambda_m\frac{1}{n}\sum_{i=1}^{n}c_{mi}\frac{\partial f(\tilde{w},i)}{\partial\tilde{w}}\Big|_{\tilde{w}=\tilde{w}^{(t_s)}}$$

    solving (24)
$$-\sum_{p=1}^{k}\sum_{q=1}^{k}(\alpha_{pq}-\beta_{pq})b_0$$
$$\xi^{(t_{s+1})}\leftarrow\frac{Q^{s+1}-\frac{1}{2}(\tilde{w}^{(t_s+1)})^T\tilde{w}^{(t_s+1)}}{C}$$
    $s=s+1$
    $\Delta Q=Q^{s-1}-Q^s$
  end while
  $\tilde{w}^t=\tilde{w}^{t_s}$
  Compute the most violated bags, i.e., c_i^t, by

$$c_i^t=\begin{cases}1, & \text{if } f(\tilde{w}^t,i)\le1\\0, & \text{otherwise}\end{cases}$$

  and update the constraint set Ψ by $\Psi=\Psi\cup c^t$
end while
Output:
The cluster assignment results for each bag Ω_i.

4. Experimental results

In this section, we present a set of experiments to validate the effectiveness and the efficiency of the proposed method.

4.1. Datasets

We use three real world traffic datasets: the CASIA dataset [5], the NGSIM dataset [51] and the MIT dataset [7] as shown in Fig. 4. The CASIA dataset contains two typical scenes including straight road (S_1) and crossroad (S_2) in traffic scenes. The image size is 240×320. The NGSIM traffic scene dataset is almost 19 minutes long with image size of 420×600. The MIT dataset contains multiple video sequences about 92 minutes long from far-field traffic scenes with image size of 480×720. There are myriads of activities and interactions in the video data. All the three datasets contain multiple motion patterns of vehicles and also include illumination changes, occlusions, and different environmental effects.

4.2. Implementation details

All the experiments are conducted with MATLAB R2008b on a 2.66 GHz Intel Core(TM)2 Duo PC running Windows Vista with 2.0 GB main memory. On the CASIA dataset [5], NGSIM dataset [51], and MIT dataset [7], the scene images are experimentally cut into 16×16 blocks, 24×32 blocks, 24×36 blocks, respectively.

For M^{4}L, we set $\varepsilon = 0.01$, $\varepsilon_1 = 0.01$, and the class imbalance parameter l is set by grid search from the grid [0.01,0.1,1 : 1 : 5,10] and the parameter C is determined by searching from the exponential grid $2^{[-3:1:3]}$. $\tilde{w}^0$ is randomly initialized. To avoid the local minimal problem, for each experiment, we run the M^{4}L algorithm many times independently and report the final result with the minimal Q. In our experiments, we run 15 times. Like most of the existing clustering algorithms, it is difficult to decide the number of clusters. For our problem, we can obtain the probable number of clusters by use of the weight of each Gaussian component. Instead, in this work, we set the number of clusters manually based on the primary motion patterns in traffic scenes.

A block may include noisy motion pattern in real applications. To solve this problem, we can set a threshold th experimentally and confirm $\Omega_i = \{p | p = \arg\max_{p \in \{1,2,\ldots,k\}}(w_p^T B_{ij}), w_p^T B_{ij} > th, j \in B_i\}$. In the experiments, we use the weight of Gaussian component

Fig. 4. Three datasets: (a) the CASIA dataset [5] (two scenes: S1 and S2), (b) the MIT dataset [7], and (c) the NGSIM dataset [51].

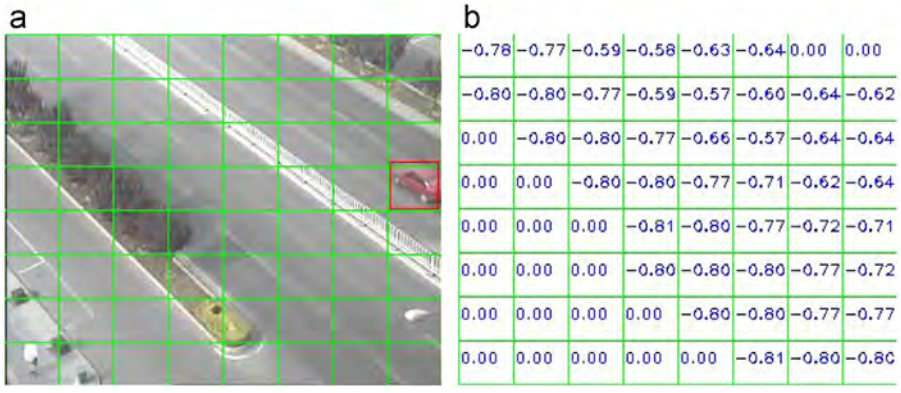

Fig. 5. (a) The scene image is cut into 8×8 blocks. (b) The values of b of the motion pattern with the highest weight in each block. Because the road is straight line, the parameter a is 0. For display, we omit the values of c.

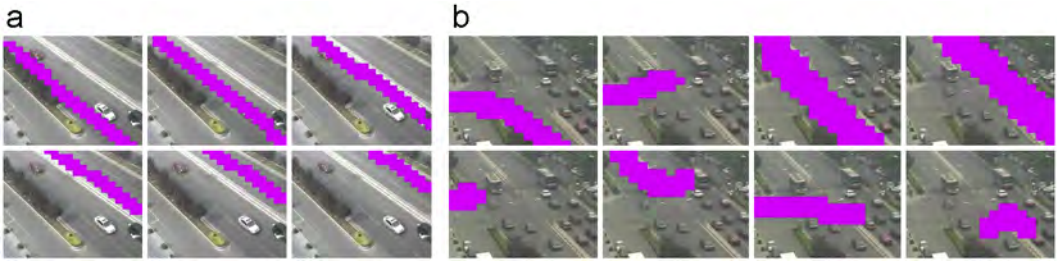

Fig. 6. Results of clustering blocks with similar motion patterns on CASIA dataset. (a) The six semantic regions in scene S1. (b) The eight semantic regions in scene S2.

to remove the noisy motion patterns. In addition, the data may be non-linearly separable, and it is better to map the data into a higher dimensional feature space via some mapping $\varphi(B_{ij(p)})$ and construct a separating hyperplane with maximum margin. This yields a non-linear decision boundary in the original input space, and Gaussian kernel can be adopted.

Since it is difficult to quantitatively evaluate the different methods by a criteria, most of existing work [4,8,5,14] attempt to demonstrate the effectiveness and efficiency of the methods by displaying the learned motion patterns in the video sequence. In this work, we adopt the trajectory clustering results to quantitatively compare with previous methods.

4.3. Motion pattern learning results

In Fig. 5, we show a simple example about the learned motion patterns on CASIA dataset. As shown in Fig. 3, vehicles are tracked from the entry point to the exit point, then the trajectories are fit to get motion pattern for each block which the vehicle has passed. As illustrated in Fig. 5, the scene image is cut into multi-blocks in Fig. 5(a). In the experiments, R and C are both set to be 8 for the 320×240 image resolution. Because the road is straight line, the parameter a is 0. For display, we omit the values of c, and just show the values of b for all blocks. The values of b of motion pattern with the highest weight in each block is displayed in Fig. 5(b). For blocks which vehicle has not passed, their features are 0. The GMM algorithm updates weight in an online way, which guarantees that the primary distribution for each block can be learnt. Based on Fig. 5(b), we can see that most of blocks have similar motion patterns. By clustering the blocks, we can group motion patterns and obtain the corresponding semantic regions. This results exhibit that our approach is effective to learn primary motion patterns for each block.

4.4. Motion pattern clustering results and analysis

For our proposed method, the convergence speed is very fast, and the computational time is less than 4 min on the three datasets, respectively. Based on the clustering results, we can confirm that our M^{4}L method is very efficient and effective to group motion patterns and is suitable to solve the MIMCL problem.

Results on CASIA dataset. Three trajectory clustering methods are compared in our experiments on this dataset. For clarity, they are denoted as I, II and III. I: As mentioned in paper [4], the modified Hausdorff distance is viewed as trajectory similarities and use spectral clustering [52]. II: Cluster trajectories based on their distributions [5]. III: our M^{4}L method takes two steps to obtain semantic regions. The first step is cutting the scene image into multiple blocks and learning motion patterns for each block by the GMM algorithm. For scene S1 and S2, the scene image 320×240 is cut into 16×16 blocks. The second step is clustering these motion patterns with M^{4}L method. Blocks having similar motion patterns are clustered to construct semantic region. On CASIA dataset, some results about block clustering are shown in Fig. 6(a) and (b). In scene S1 and S2, there are six and eight semantic regions of vehicles, respectively. Each of them represents a primary motion pattern.

Based on the semantic regions and their corresponding motion patterns, trajectories which fit the same semantic regions are considered as a cluster. To make a quantitative comparison, the statistical results of trajectory clustering in scene S1 are calculated. On this dataset, we give the details of average recall and average precision of the six clusters. The average recall and

Table 1
The comparisons among three methods about average precision and average recall in scene S1 on the CASIA dataset.

Method	Average precision (%)	Average recall (%)
I	91.6	94.5
II [5]	96.5	98.6
III (Ours)	99.1	99.3

Table 2
The Precision and Recall of two different methods on MIT dataset. TP is true positive, FN is false negative, FP is false positive.

Method	Cluster	TP	FN	FP	Recall (%)	Precision (%)
I	1	72	12	9	85.7	88.9
	2	152	39	29	79.6	84.0
	3	247	44	51	84.9	82.9
	4	464	137	153	77.2	75.2
	5	51	13	9	79.7	85.0
	6	62	19	10	76.5	86.1
	7	112	11	17	91.1	86.8
	8	77	18	13	81.1	85.6
	9	142	34	27	80.7	84.0
	10	25	8	17	75.8	59.5
II (Our)	1	72	9	7	88.9	91.1
	2	152	21	17	87.9	89.9
	3	247	29	33	89.5	88.2
	4	464	67	68	87.4	87.2
	5	51	10	8	83.6	86.4
	6	62	12	9	83.8	87.3
	7	112	8	16	93.3	87.5
	8	77	7	7	91.7	91.7
	9	142	20	17	87.7	89.3
	10	25	4	5	86.2	83.3

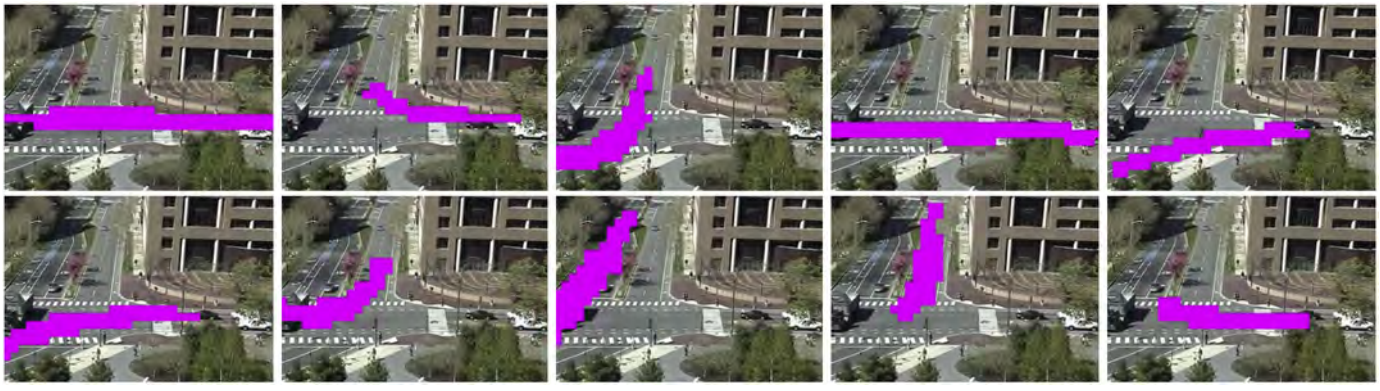

Fig. 7. Results of clustering blocks with similar motion patterns on MIT dataset including ten semantic regions.

Fig. 8. Results of clustering blocks with similar motion patterns on NGSIM dataset with twelve semantic regions.

precision are 99.3% and 99.1%, respectively, which are better than the result in [5] (98.6% and 96.5%). The detail is shown in Table 1. Based on the results, we can confirm that our M^{4}L method is effective to cluster motion patterns.

Results on MIT dataset. Two trajectory clustering methods are compared on this dataset. The first method is denoted as I, which adopts the modified Hausdorff distance as trajectory similarities and uses spectral clustering [52]. The second is our method. On this dataset, the scene image 480×720 is cut into 24×36 blocks. By use of our method, we obtain ten primary motion patterns as shown in Fig. 7. Based on the results and motion patterns, trajectories are grouped into ten clusters. To make a quantitative comparison, the statistical results are illustrated in Table 2. For the 1404 trajectories, there are ten clusters. The ground truth annotation of each cluster is labeled manually. There are 72, 152, 247, 464, 51, 62, 112, 77, 142 and 25 trajectories for cluster 1, 2, 3, 4, 5, 6, 7, 8, 9 and 10 respectively. Recall and Precision are used to measure the performance. These results show our method II performs the better. The average recall and precision are 88.0% and 88.2%, respectively, which are better than the results with [5] (79.8% and 81.7%).

Results on NGSIM dataset. We adopt the same way as on MIT dataset to compare our method with [4]. For our method, the scene image 420×600 is cut into 24×32 blocks, and the twelve primary motion patterns on this dataset are shown in Fig. 8. By use of the results of clustering blocks with similar motion patterns and their corresponding motion patterns, trajectories which belong to similar motion patterns are considered as a cluster. To make a quantitative comparison, the statistical results of trajectory clustering are illustrated in Table 3. For the 1365 trajectories, there are twelve clusters. The ground truth annotation of each cluster is labeled manually. There are 498, 294, 37, 58, 121, 66, 76, 83, 47, 12, 38 and 35 trajectories for cluster 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11 and 12 respectively. Recall and Precision are used to measure the performance. Compared with the method I, we can confirm that our method is efficient to cluster motion patterns, and obtains much better performances. On this dataset, the average recall and precision are 88.5% and 87.5%, respectively, which are better than the performance of [5] (78.7% and 80.2%).

As can be seen in Figs. 6–8, the proposed method is able to detect multiple semantic motion patterns in a completely unsupervised manner. In these crowded traffic scenes, multiple co-occurring patterns happen in the same region. We can see that our algorithm successfully groups blocks with multiple clusters corresponding to multiple semantical motion patterns.

Table 3
The Precision and Recall of two different methods on NGSIM dataset. TP is true positive, FN is false negative, FP is false positive.

Method	Cluster	TP	FN	FP	Recall (%)	Precision (%)
I	1	498	101	93	83.1	84.3
	2	294	79	81	78.8	78.4
	3	37	11	9	77.1	80.4
	4	58	13	15	81.7	79.5
	5	121	21	25	85.2	82.9
	6	66	18	21	78.6	75.9
	7	76	12	19	86.4	80.0
	8	83	9	7	90.2	92.2
	9	47	14	15	77.1	75.8
	10	12	6	5	66.7	70.6
	11	38	12	7	76.0	84.4
	12	35	10	9	77.8	79.6
II (Our)	1	498	17	9	96.7	98.2
	2	294	11	16	96.4	94.8
	3	37	8	5	82.2	88.1
	4	58	6	9	90.6	86.6
	5	121	13	18	90.3	87.1
	6	66	7	9	90.4	88.0
	7	76	9	6	89.4	92.7
	8	83	5	4	94.3	95.4
	9	47	15	13	75.8	78.3
	10	12	1	4	92.3	75.0
	11	38	8	6	82.6	86.4
	12	35	8	9	81.4	79.6

4.5. Semantic region results and analysis

Based on the clustering blocks and their corresponding motion patterns, trajectories which fit the same regions are considered as a cluster. For each cluster of trajectories, the methods in [5] are used to learn the corresponding semantic scene region. The path region is obtained by thresholding the density distribution. The learned semantic scene regions on three datasets are shown in Figs. 9–11. The red lines are the primary trajectories, which represent the primary motion patterns. The red arrows are the moving directions of objects, and the regions denoted with color are the learned paths of vehicle.

The results in Figs. 9–11 show the primary motion patterns happening on the three datasets. Let us take the results in Fig. 11 as an example. The scene is a crossroad, and contains multiple complex motion patterns. The learned motion patterns are time continuous and their distributions in the scene image are consistent with the actual paths. Therefore, they are very meaningful and represent the activities of objects at a higher semantic level. The similar results on this dataset are shown in [8,14], which adopted multi-scale analysis

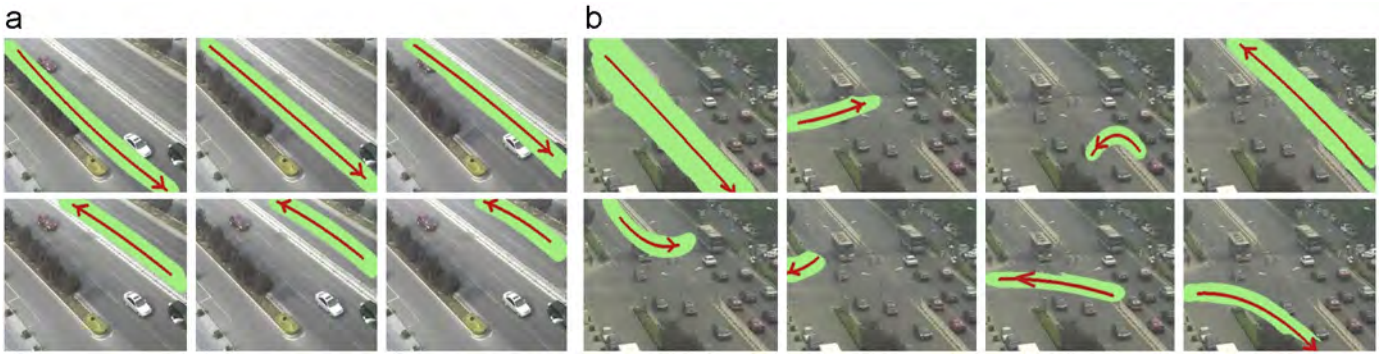

Fig. 9. The fine results of semantic scene regions in scene S1 and S2. The red arrows are the moving directions of objects, and the red curves are the primary trajectories. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this Article.)

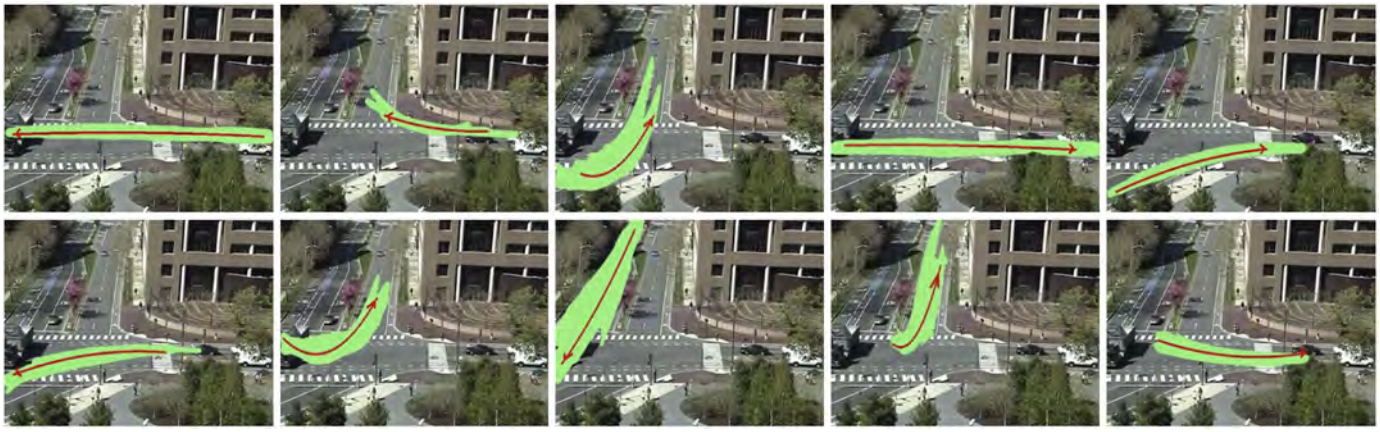

Fig. 10. The fine results of semantic scene regions on MIT dataset. The red arrows are the moving directions of objects, and the red curves are the primary trajectories. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this Article.)

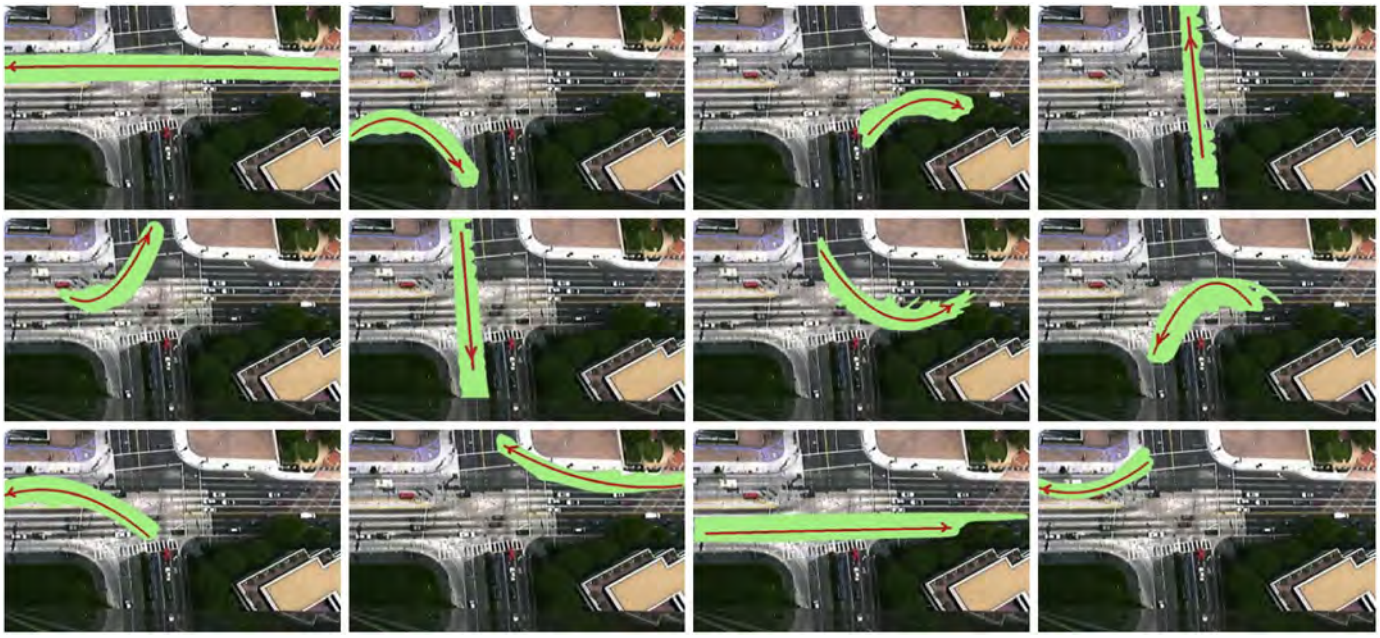

Fig. 11. The fine results of semantic scene regions on NGSIM dataset. The red arrows are the moving directions of objects, and the red curves are the primary trajectories. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this Article.)

and statistical representation of motion patterns, respectively. Because we use the trajectories as the description, it helps to make the learned motion pattern more continuous. Compared with [8,14], we can confirm that our method obtains comparable results, and for some motion patterns, such as the horizontal motion patterns that vehicles move through west-east road unhindered, our results are

much finer and more meaningful. Based on the learned information, we can have a variety of surveillance applications, for example, detect abnormal activities, improve object detection and infer the motion of objects in the scenes.

5. Conclusions

We have formulated motion pattern grouping as a Multi-instance Multi-cluster Learning (MIMCL) problem for video scene modeling. To solve this problem, a novel unsupervised clustering approach called M^4L is proposed. An efficient optimization solution is adopted by use of combination of Constrained Concave–Convex Procedure (CCCP) and the Cutting Plane method. Our M^4L is generic for MIMCL problem, which can also be used for image clustering problem (image with multiple semantical concepts). In the future, we will study how to guarantee the convergence of M^4L as the proofs in [47]. We will also investigate how to automatically decide the number of cluster and design an effective evaluation method.

Conflict of interest statement

None declared.

Acknowledgments

This work was supported by 973 Program (Project No. 2012CB316304).

References

[1] C. Stauffer, E. Grimson, Learning patterns of activity using real-time tracking, IEEE Transactions on Pattern Analysis and Machine Intelligence 22 (8) (2000) 747–757.
[2] W. Hu, X. Xiao, Z. Fu, D. Xie, T. Tan, S. Maybank, A system for learning statistical motion patterns, IEEE Transactions on Pattern Analysis and Machine Intelligence 28 (9) (2006) 1450–1464.
[3] I. Saleemi, K. Shafique, M. Shah, Probabilistic modeling of scene dynamics for applications in visual surveillance, IEEE Transactions on Pattern Analysis and Machine Intelligence 31 (8) (2009) 1472–1485.
[4] X. Wang, K. Tieu, E. Grimson, Learning semantic scene models by trajectory analysis, in: In European conference on Computer Vision, 2006, pp. 110–123.
[5] T. Zhang, H. Lu, S. Li, Learning semantic scene models by object classification and trajectory clustering, in: In IEEE Conference on Computer Vision and Pattern Recognition, 2009, pp. 1940–1947.
[6] T. Xiang, S. Gong, Video behaviour profiling and abnormality detection without manual labelling, in: In IEEE International Conference on Computer Vision, 2005, pp. 1238–1245.
[7] X. Wang, X. Ma, E. Grimson, Unsupervised activity perception by hierarchical Bayesian models, in: IEEE Conference on Computer Vision and Pattern Recognition, 2007, pp. 1–8.
[8] Y. Yang, J. Liu, M. Shah, Video scene understanding using multi-scale analysis, in: IEEE International Conference on Computer Vision, 2009, pp. 1669–1676.
[9] T. Kanade, B. Lucas, An iterative image registration technique with an application to stereo vision, in: International Joint Conference on Artificial Intelligence, 1981, pp. 674–679.
[10] L. Kratz, K. Nishino, Anomaly detection in extremely crowded scenes using spatio-temporal motion pattern models, in: IEEE Conference on Computer Vision and Pattern Recognition, 2009, pp. 1446–1453.
[11] M. Brand, V. Kettnaker, Discovery and segmentation of activities in video, IEEE Transactions on Pattern Analysis and Machine Intelligence 22 (8) (2000) 844–851.
[12] I.N. Junejo, H. Foroosh, Trajectory rectification and path modeling for video surveillance, in: IEEE International Conference on Computer Vision, 2007, pp. 1–7.
[13] X. Wang, K.T. Ma, G.-W. Ng, E. Grimson, Trajectory analysis and semantic region modeling using a nonparametric Bayesian model, in: IEEE Conference on Computer Vision and Pattern Recognition, 2008, pp. 1–8.
[14] I. Saleemi, L. Hartung, M. Shah, Scene understanding by statistical modeling of motion patterns, in: IEEE Conference on Computer Vision and Pattern Recognition, 2010, pp. 2069–2076.
[15] R.O. Duda, P.E. Hart, D.G. Stork, Pattern Classification, John Wiley and Sons, Inc., 2001, pp. 1–16.
[16] J. Shi, J. Malik, Normalized cuts and image segmentation, IEEE Transactions on Pattern Analysis and Machine Intelligence 22 (8) (2000) 888–905.
[17] C. Ding, X. He, H. Zha, M. Gu, H.D. Simon, A min-max cut algorithm for graph partitioning and data mining, in: IEEE International Conference on Data Mining, 2001, pp. 107–114.
[18] L. Xu, J. Neufeld, B. Larson, D. Schuurmans, Maximum margin clustering, in: Neural Information Processing Systems, 2004, pp. 1537–1544.
[19] M.-L. Zhang, Z.-H. Zhou, Multi-instance clustering with applications to multi-instance prediction, Applied Intelligence 31 (1) (2009) 47–68.
[20] D. Zhang, F. Wang, L. Si, T. Li, M^3ic: maximum margin multiple instance clustering, in: IJCAI, 2009, pp. 1–12.
[21] F. Wang, B. Zhao, C. Zhang, Linear time maximum margin clustering, IEEE Transactions on Neural Networks 21 (2) (2010) 319–332.
[22] I. Saleemi, K. Shafique, M. Shah, Probabilistic modeling of scene dynamics for applications in visual surveillance, IEEE Transactions on Pattern Analysis and Machine Intelligence 31 (8) (2009) 1472–1485.
[23] D. Kuettel, M.D. Breitenstein, L.V. Gool, V. Ferrari, What's going on? Discovering spatio-temporal dependencies in dynamic scenes, in: IEEE Conference on Computer Vision and Pattern Recognition, 2010, pp. 1951–1958.
[24] Z. Gloria, R. Elisa, Earth mover's prototypes: a convex learning approach for discovering activity patterns in dynamic scenes, in: IEEE Conference on Computer Vision and Pattern Recognition, 2011, pp. 3225–3232.
[25] T. Hospedales, S. Gong, T. Xiang, A Markov clustering topic model for mining behaviour in video, in: IEEE International Conference on Computer Vision, 2009, pp. 1–8.
[26] A. Basharat, A. Gritai, M. Shah, Learning object motion patterns for anomaly detection and improved object detection, in: IEEE Conference on Computer Vision and Pattern Recognition, 2008, pp. 1–8.
[27] J.M. Gryn, R.P. Wildes, J.K. Tsotsos, Detecting motion patterns via direction maps with application to surveillance, Computer Vision and Image Understanding 113 (2) (2009) 291–307.
[28] A. Jain, R. Dubes, Algorithms for Clustering Data, Prentice-Hall, Englewood Cliffs, NJ, 1988, pp. 1–334.
[29] M. Filippone, F. Camastra, F. Masulli, S. Rovetta, A survey of kernel and spectral methods for clustering, Pattern Recognition 41 (2008) 176–190.
[30] T.G. Dietterich, R.H. Lathrop, T. Lozano-Perez, A. Pharmaceutical, Solving the multiple-instance problem with axis-parallel rectangles, Artificial Intelligence 89 (1997) 31–71.
[31] R.E. Schapire, Y. Singer, Boostexter: a boosting-based system for text categorization, in: Machine Learning, 2000, pp. 135–168.
[32] A.K. McCallum, Multi-label text classification with a mixture model trained by EM, in: AAAI 99 Workshop on Text Learning, 1999, pp. 1–10.
[33] M.-L. Zhang, Z.-H. Zhou, M3MIML: a maximum margin method for multi-instance multi-label learning, in: IEEE International Conference on Data Mining, 2008, pp. 688–697.
[34] S. Andrews, I. Tsochantaridis, T. Hofmann, Support vector machines for multiple-instance learning, in: In Neural Information Processing Systems 15, MIT Press, 2003, pp. 561–568.
[35] Q. Zhang, S.A. Goldman, Em-dd: an improved multiple-instance learning technique, in: Neural Information Processing Systems, 2001, pp. 1073–1080.
[36] Y. Chen, J. Bi, J.Z. Wang, Miles: multiple-instance learning via embedded instance selection, IEEE Transactions on Pattern Analysis and Machine Intelligence (2006) 1931–1947.
[37] Y. Chen, J.Z. Wang, D. Geman, Image categorization by learning and reasoning with regions, Journal of Machine Learning Research 5 (2004) 913–939.
[38] Z.-H. Zhou, Multi-Instance Learning: A Survey, Technical Report, 2004.
[39] H. Kazawa, T. Izumitani, H. Taira, E. Maeda, Maximal margin labeling for multi-topic text categorization, in: Neural Information Processing Systems 17, MIT Press, 2005, pp. 649–656.
[40] S. Zhu, X. Ji, W. Xu, Y. Gong, Multi-labelled classification using maximum entropy method, in: Proceedings of the SIGIR, ACM Press, 2005, pp. 274–281.
[41] M.R. Boutell, J. Luo, X. Shen, C.M. Brown, Learning multi-label scene classification, Pattern Recognition 37 (2004) 1757–1771.
[42] G. Tsoumakas, I. Katakis, Multi-label classification: an overview, International Journal of Data Warehousing and Mining 2007 (2007) 1–13.
[43] Z. Hua Zhou, M. Ling Zhang, Multi-instance multilabel learning with application to scene classification, Neural Information Processing Systems (2007) 1609–1616.
[44] Z. Jun Zha, X. Sheng Hua, T. Mei, J. Wang, G. Jun Qi, Z. Wang, Joint multi-label multi-instance learning for image classification, in: IEEE Conference on Computer Vision and Pattern Recognition, 2008, pp. 1–8.
[45] R. Jin, S. Wang, Z. Hua Zhou, Learning a distance metric from multi-instance multi-label data, in: IEEE Conference on Computer Vision and Pattern Recognition, 2009, pp. 896–902.
[46] Y. Cheng, Mean shift, mode seeking, and clustering, IEEE Transactions on Pattern Analysis and Machine Intelligence 17 (1995) 790–799.
[47] T. Joachims, Training linear SVMS in linear time, in: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2006, pp. 217–226.
[48] J.E. Kelley, The cutting-plane method for solving convex programs, Journal of the Society for Industrial Applied Mathematics 8 (1960) 703–712.
[49] A. Smola, S. Vishwanathan, T. Hofmann, Kernel methods for missing variables, in: Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics, 2005, pp. 325–332.
[50] 〈http://www.mosek.com/〉.

[51] Next generation simulation (NGSIM) dataset, in: Available at 〈http://www.ngsim.fhwa.dot.gov/〉.

[52] M. Meila, J. Shi, A random walks view of spectral segmentation, in: AISTATS, 2001, pp. 1–8.

Tianzhu Zhang is a postdoctoral fellow at Advanced Digital Sciences Center (ADSC), Singapore. He received his Bachelor's degree in communications and information technology from Beijing Institute of Technology, Beijing, in 2006. He obtained his Ph.D. in pattern recognition and intelligent systems from Institute of Automation, Chinese Academy of Sciences, Beijing, in 2011. His research interests are in computer vision and multimedia, including action recognition, object classification and object tracking.

Si Liu received the Ph.D. degree at the National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing, China in 2012. She is currently a Research Fellow with the Learning and Vision Group, National University of Singapore, Singapore. Her research interests include computer vision and multimedia.

Changsheng Xu is a Professor in National Lab of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences and Executive Director of China-Singapore Institute of Digital Media. His research interests include multimedia content analysis/indexing/retrieval, pattern recognition and computer vision. He has hold 30 granted/pending patents and published over 200 refereed research papers in these areas. He is an Associate Editor of ACM Transactions on Multimedia Computing, Communications and Applications and ACM/Springer Multimedia Systems Journal. He served as Program Chair of ACM Multimedia 2009. He has served as associate editor, guest editor, general chair, program chair, area/track chair, special session organizer, session chair and TPC member for over 20 IEEE and ACM prestigious multimedia journals, conferences and workshops.

Hanqing Lu received the Ph.D. degree in Huazhong University of Sciences and Technology, Wuhan, China in 1992. Currently he is Professor of Institute of Automation, Chinese Academy of Sciences. His research interests include image similarity measure, video analysis, object recognition and tracking. He published more than 200 papers in those areas.