Learning Semantic Scene Models by Object Classification and Trajectory Clustering

Tianzhu Zhang[1] Hanqing Lu[1] Stan Z. Li[1,2]
[1]National Laboratory of Pattern Recognition & [2]Center for Biometrics and Security Research,
Institute of Automation, Chinese Academy of Sciences 95 Zhongguancun Donglu, Beijing, China.
{tzzhang, luhq, szli}@nlpr.ia.ac.cn

Abstract

Activity analysis is a basic task in video surveillance and has become an active research area. However, due to the diversity of moving objects' category and their motion patterns, developing robust semantic scene models for activity analysis remains a challenging problem in traffic scenarios.

This paper proposes a novel framework to learn semantic scene models. In this framework, the detected moving objects are first classified as pedestrians or vehicles via a co-trained classifier which takes advantage of the multi-view information of objects. As a result, the framework can automatically learn motion patterns respectively for pedestrians and vehicles. Then, a graph is proposed to learn and cluster the motion patterns. To this end, trajectory is parameterized and the image is cut into multiple blocks which are taken as the nodes in the graph. Based on the parameters of trajectories, the primary motion patterns in each node (block) are extracted via Gaussian Mixture Model (GMM), and supplied to this graph. The graph-cut algorithm is finally employed to group the motion patterns together, and trajectories are clustered to learn semantic scene models. Experimental results and applications to real-world scenes show the validity of our proposed method.

1. Introduction

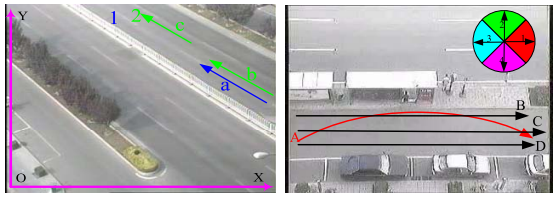
Extracting the motion patterns from videos to develop automatically traffic management systems meets the great needs in the world. Video-based activity analysis helps to convey the information along which path or direction the vehicles or pedestrians should move or walk. Such summarized information is fundamental on which the rules of abnormal behavior of moving objects are defined. With such a system, the initialization work of supplying the information of scene models, such as traffic paths, entry and exit points of traffic, can be saved. However, the motion of pedestrians and vehicles is complex and their motion patterns are different from each other. Thus automatically learning the semantic scene models from videos becomes a challenging problem in computer vision and pattern recognition.

Recently, many approaches on learning semantic scene models have been proposed by trajectory analysis [7, 3, 20, 13, 19]. One of most important work is to introduce object classification into trajectory analysis. Wang et al. [20] use scene context features (such as position, area in pixels, and velocity) to cluster trajectories into different types (vehicles vs. pedestrians). Experimental results show that this method is more efficient. But due to low resolution, shadow, different viewing angles, classifying objects with only these features is not enough in video surveillance.

During learning semantic scene models, lots of methods have been proposed to cluster trajectories, and these methods can be categorized into two classes: spatial distance based methods [5, 8, 13, 20] and spatial distribution based methods [19]. Spatial distance based methods take only the pairwise similarities between trajectories. The proposed trajectory similarities or distances include Euclidean distance [5], Hausdorff distance and its variations [8, 20], and Dynamic Time Warping (DTW) [9]. These approaches have several drawbacks: lack a probabilistic explanation for abnormality detection, require the cluster number in advance, have a high computation cost and may not well approximate the true similarity. For example, in Figure 1(a), trajectories a, b and c belong to two different clusters. Trajectories b and c are in the same cluster, but trajectories a and b will have much higher similarity than b and c using the similarity defined in [5, 8, 20]. Therefore, spatial distribution based methods are proposed by Wang et al. [19] to avoid these drawbacks. Wang et al. [19] use the distributions of observations (positions and moving directions of objects) on the trajectories for trajectory analysis, but do not take into account the integrity of each trajectory. Based on the definition in [19], in Figure 1(b), the observations (positions and moving directions of objects) on the trajectories A, B, C and D are similar, so they are clustered into the same activity. In practice, we think trajectories B, C and

978-1-4244-3991-1/09/$25.00 ©2009 IEEE

D are the same cluster, and trajectory A is abnormal and represents another cluster.



(a) (b)

Figure 1. (a) Some measured similarities between trajectories may not well approximate the true similarity in the physical world. (b) It is different between distributions of observations on trajectories and distributions of trajectories.

After clustering trajectories, semantic scene models are obtained for each cluster. Paths can be detected by modeling the spatial extents of trajectory clusters [20, 3, 7]. Entry and exit points are detected at the ends of paths based on the velocity distribution [20]. Makris and Ellis [3], Mckenna and Nait-Charif [15] detect these points from start/end points of trajectories by the Gaussian Mixture Models (GMM).

In this paper, we propose a novel method to learn semantic scene models by object classification and trajectory clustering. To avoid labeling a large training set by hand and improve classification performance, the co-training learning algorithm is adopted to train two classifiers, LDA-based classifier and AdaBoost classifier, on two independent features. To further cluster each class of trajectories, we make use of spatial distribution of trajectories. The spatial distribution is fitted by an underlying parameter model instead of the discrete observations of trajectories in traditional methods [19]. Experimental results illustrate the effectiveness and efficiency of the proposed method.

2. Our Approach

We take two steps to learn semantic scene models: (1) train classifiers under our co-training framework to classify object into vehicle or pedestrian, and (2) cluster each class of trajectories with their spatial distributions (trajectories' parameters and directions). Figure 2 illustrates the framework. Real-time background subtraction and tracking [16] is used to detect and track the moving objects. For each foreground object, a co-training classifier (Section 3) is adopted to classify it into vehicle or pedestrian. For each kind of trajectory, it is further clustered. Finally, semantic scene models are obtained for each cluster of trajectory and used in some applications.

For the first step, we adopt a semi-supervised method to learn two classifiers for object classification (Section 3), inspired by the idea of co-training learning. Two sets of features are predefined and they are relatively independent

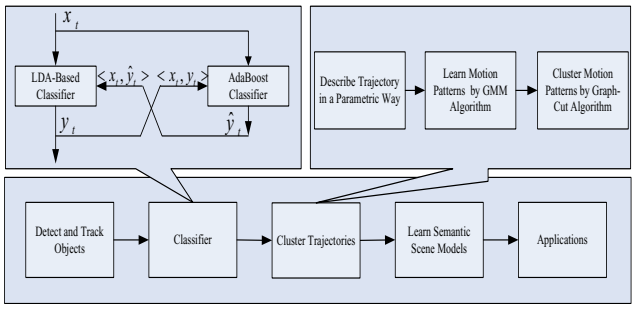


Figure 2. The proposed flowchart of learning semantic scene models.

of each other: (1) scene context features, such as position, area in pixels, velocity; and (2) appearance features based on Multi-block Local Binary Pattern (MB-LBP) [22]. Two labeled sets are then prepared based on them, each for training one of the classifiers. Each classifiers prediction on the unlabeled samples to enlarge the training set of the other. Experiments demonstrate that the co-training algorithm can generate an accurate classifier conveniently and effectively.

For the second step, a graph is adopted to cluster motion patterns of each category for trajectory clustering. Trajectories are described in a parametric way (Section 4.1) and scene image is cut into multiple blocks which are taken as the nodes of the graph. Each node is viewed as the distributions of trajectories (trajectories' parameters and moving directions of objects), not the observations (positions and moving directions of objects) on the trajectories [19]. In this way, spatial distance between trajectories is reflected by their parameters. As in Figure 1(b), trajectory A will be distinguished and viewed as a different cluster because its parameters are different. Trajectory distribution in a node can be viewed as a Gaussian distribution from statistic point of view. Each node may have multiple distributions, so the Gaussian Mixture Model (GMM) is adopted to describe spatial distributions of trajectories for each node (Section 4.2). Each of the Gaussian components is one of the underlying motion patterns. Then motion patterns are clustered by graph-cut algorithm (Section 4.3) to obtain its corresponding semantic region.

Based on the semantic regions and their corresponding motion patterns, trajectories are further clustered. For each cluster of trajectories, entry/exit points and primary trajectories are learnt by Mean-Shift based multiple data mode-seeking algorithm [21]. As a result, semantic scene models are learnt (Section 5) and used for some applications (Section 6.4). The proposed method is verified on extensive real video data and the results are encouraging (Section 6). The main contributions can be summarized as follows:

- Fusing different features and enlarging unlabeled samples under our co-training framework improve the classification performance.

- Trajectory is described in a parametric way which can save storage and reduce computation cost.
- It is robust to cluster the motion patterns based on not only their value in each node but also their neighboring correlations in the graph.

3. Object Classification

3.1. LDA-Based Classifier

Scene context features reflect the properties of objects in the scene image, so they can be used to distinguish objects. For a certain scene, scene context features are robust and useful to classify object [20, 23]. In this paper, scene context features: x-, y- image coordinates, area in pixels, speed, direction of motion, aspect ratio, percentage occupancy, are used to find an optimal direction of projection to separate the positive and negative sample with fisher linear discriminant analysis (LDA). The projection function is defined as: $g = w^T r$, where $w = (S^1 + S^2)^{-1}(u^1 - u^2)$, u^1, u^2 are the means of the two classes (people and vehicle), and S^1, S^2 are the covariance matrices. The formulation of u^t and S^t, $t = 1, 2$, are as follows: $u^t = \frac{1}{n_t}\sum_{j}^{n_t} r_j$, $S^t = \frac{1}{n_t - 1}\sum_{j}^{n_t}(r_j - u^r)(r_j - u^r)^T$, where r_j is the j^{th} sample.

3.2. AdaBoost Classifier

The LDA-based classifier constructed by scene context features is relative to scene, Therefore, an appearance classifier based on Multi-block Local Binary Pattern (MB-LBP) [22] features is adopted to improve the performance of classification. MB-LBP is extended from the original LBP feature [14], which has been proven to be a powerful appearance descriptor with computational simplicity. Besides, this feature is also successfully applied in many low resolution image analysis tasks [6]. However, it is limited to calculate the information in a small region and has no ability to capture large-scale structures of objects. MB-LBP is developed on image patches divided into sub-blocks (rectangles) with different sizes. This treatment provides a mechanism for us to capture appearance structures with various scales and aspect ratios. Intrinsically, MB-LBP is to measure the intensity differences between sub-blocks in image patches. Calculation on blocks is robust to noises, lighting changing. At the same time, MB-LBP can be computed very efficiently by using integral images [18]. The feature set of MB-LBP feature is large and contains much redundant information. AdaBoost algorithm is used to select significant features and construct a binary classifier. The Gentle AdaBoost [4, 17] is adopted for the reason that it is simple to be implemented and numerically robust.

3.3. The Co-Training Framework

To train the classifiers, labeling a large training set by hand can be time-consuming and tedious. The difficulty is the high cost of acquiring a large set of labeled examples to train the two classifiers. Of course, gathering a large number of unlabeled examples in most applications has much lower cost, as it requires no human intervention. One typical method for training the image classifier is to adopt the co-training learning algorithm [1] from a combination of both labeled and unlabeled data. The basic idea is to train two classifiers on two independent "views "(features) of the same data, using a relatively small number of examples. Then the most confidently labeled examples from one classifier are added to the labeled set of the other classifier. Blum and Mitchell [1] prove that co-training can find a very accurate classification rule, starting from a few labeled examples if the two feature sets are statistically independent. However, Levin et al. [11] empirically prove that co-training is still possible even in the case the independence assumption does not hold.

In our algorithm two relatively independent features are used: scene context features and MB-LBP features as the object representation. Each feature is used to train a classifier. The classifiers are trained as follows: for a certain scene, some samples are labeled to train the two classifiers, then the classifiers are used to classify unlabeled examples to obtain their labels and add those newly labeled examples which are confident enough to update the training set for each other. This learning process can be repeated many times. As the LDA-based classifier is relative to scene, for a different scene, the AdaBoost classifier is used to label samples to train the LDA-based classifier, then they are training for each other. To ensure the correct classification, from one training data set to another, their appearances change slowly.

The main advantages of this scheme are: (1) It is a collaborative approach that uses the strength of different views of the object to help improve each other, hence a more robust classification. (2) Mass manual labeling is avoided. Experiments demonstrate that co-training can generate accurate classifiers. After training classifiers,we make final classification decision according to the output of the classifier with more confidence.

4. Trajectory Clustering

4.1. Trajectory Description

A trajectory can be obtained by tracking the centroid of an object, in the 2-D image coordinates as in Figure 1(a), whose origin is on the bottom left corner, it can be described as $T = \{(x_1, y_1), (x_2, y_2), \cdots, (x_n, y_n)\}$. In general traffic scenes, trajectory of a vehicle is not complicated, thus it is reasonable to 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 Figure 1(b). The parameters (a, b, c, v) are features of a trajectory.

4.2. 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 and learn these motion patterns based on each block.

Objects are classified into vehicles or pedestrians, and there are two kinds of trajectories. For each kind 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. (2) Outlier trajectories can be removed by updating the weight of Gaussian model, hence primary motion patterns are able to 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 computation cost is low. Experiments demonstrate that the GMM algorithm is efficient.

Our algorithm can be described as follows. 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 $\{T_t = (a_t, b_t, c_t, v_t)\}_{t=1}^{N}$ are obtained. Here (a_t, b_t, c_t, v_t) are parameters of a trajectory T_t. They are used to learn the parameters distribution of blocks which the objects have passed. The probability that a certain 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 i^{th} Gaussian component at time t, $\eta(T_t, u_{i,t}, \Sigma_{i,t})$ is the i^{th} 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.

$$u_{i,t} = (u^a_{i,t}, u^b_{i,t}, u^c_{i,t})^T \tag{2}$$

$$\sigma_{i,t} = (\sigma^a_{i,t}, \sigma^b_{i,t}, \sigma^c_{i,t})^T \tag{3}$$

$$\Sigma^{\frac{1}{2}}_{i,t} = \begin{pmatrix} \sigma^a_{i,t} & 0 & 0 \\ 0 & \sigma^b_{i,t} & 0 \\ 0 & 0 & \sigma^c_{i,t} \end{pmatrix} \tag{4}$$

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,

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

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

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

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

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, K is manually set to be 3, 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.

4.3. Clustering Motion Patterns by Graph-Cut Algorithm

Up to now, a straightforward way to group the patterns is to use the commonly used clustering algorithm. However, the algorithms, for example, the K-means algorithm, do not consider the spatial relations between local blocks. Actually, for two neighboring blocks, it is possible that they contain similar motion patterns. Thus it is necessary to consider such spatial relations when grouping the motion patterns together, which yields the following graph-based algorithm.

Naturally, we can take each local block as a node, and two neighboring blocks in horizontal and vertical directions can be connected together. In terms of Markov Random Filed (MRF), we consider to minimize the following energy function:

$$E(L) = \sum_{p \in S} D_p(L_p) + \sum_{(p,q) \in \mathcal{N}} V_{p,q}(L_p, L_q), \tag{9}$$

where S is the block lattice, $\mathcal{N}$ is the pair-wise neighborhood system, the label set is $\{0, 1\}$. $D_p(L_p)$ denotes the cost of the block p to be labeled as L_p. $V_{p,q}(L_p, L_q)$ encourages spatially neighboring blocks to have similar labels.

Now a core task is to calculate the terms of $D_p(L_p)$ and $V_{p,q}(L_p, L_q)$ in Eq. (9). Note that we have not any prior information about the patterns. To assign values to $D_p(L_p)$, we first extract the primary motion patterns in the scene.

Since all motion patterns $(\mathbb{G})$ learnt by GMM algorithm are collected into $\mathbb{G} = \{\vec{g}^k_{ij} | i =$

$1, 2, \cdots, R, j = 1, 2, \cdots, C, k = 1 \cdots K.\}$, Where $\vec{g}_{ij}^k = (a_{ij}^k, b_{ij}^k, c_{ij}^k, v_{ij}^k)^T$ is the k^{th} motion pattern of the block (i, j), then we can consider to take the components from $\mathbb{G}$ as the primary motion patterns. These patterns will be used as references to calculate the energy term.

As for a scene, there are only a few primary motion patterns. The weight of a Gaussian model reflects the importance of a motion pattern. Therefore, a Gaussian model $\vec{g}_{ij}^k$ is selected as a primary motion pattern if its weight $w_{ij}^k > Th$. Th is a threshold which is manually set to be 0.85 in our experiments throughout our evaluation. In this way, all of the primary motion patterns ($\mathbb{G}_m$) are extracted. For a motion pattern in $\mathbb{G}_m$, it is considered as a reference, all of the primary motion patterns ($\mathbb{G}_m$) are viewed as two clusters, denoting them as $\vec{g}_r$, the other is $\vec{g}_a$. $\vec{g}_a$ is the average of motion patterns which belong to $\{\vec{g}_l | \, \|\vec{g}_r - \vec{g}_l\| > \lambda \times d_{mean}, v_r \cdot v_l = 1\}$, where λ is a tuned coefficient and is manually set to be 0.85, d_{mean} is the average distance between the reference and the motion patterns having similar velocity with the reference in $\mathbb{G}_m$. Distance between motion pattern $\vec{g}_1 = (a_1, b_1, c_1, v_1)$ and $\vec{g}_2 = (a_2, b_2, c_2, v_2)$ is defined as follows: if $v_1 \cdot v_2 = 1$, $\|\vec{g}_1 - \vec{g}_2\|^2 = (a_1 - a_2)^2 + (b_1 - b_2)^2 + (c_1 - c_2)^2$, otherwise, $\|\vec{g}_1 - \vec{g}_2\|^2 = Max$. Where Max is a large constant number and is set to be 99 in our experiments. Now we can calculate $D_p(L_p)$ as follows:

$$D_p(L_p) = \begin{cases} \frac{d_1}{d_1+d_2} & \text{if } L_p = 1 \\ \frac{d_2}{d_1+d_2} & \text{if } L_p = 0 \end{cases} \tag{10}$$

, where $d_1 = \min_k \left\|\vec{g}_r - \vec{g}_{ij}^k\right\|$, $d_2 = \min_k \left\|\vec{g}_a - \vec{g}_{ij}^k\right\|$. d_1 and d_2 represent the similarities between the motion patterns of block (i, j) and $\vec{g}_r$, $\vec{g}_a$ respectively. Furthermore, $V_{p,q}(L_p, L_q)$ is evaluated as

$$V_{p,q}(L_p, L_q) = \begin{cases} 0 & \text{if } L_p{=}L_q \\ d_0 & \text{otherwise} \end{cases} \tag{11}$$

, where d_0 is a constant to punish a label jumping, it is calculated as the standard deviation of $\{d_{ij} | \, i = 1, 2, \cdots, R; \, j = 1, 2, \cdots, C\}$, and d_{ij} is $\min_k(\left\|\vec{g}_r - \vec{g}_{ij}^k\right\|)$, which expresses the minimum distance between motion patterns of block (i, j) and the reference motion pattern.

Finally, for each motion pattern in $\mathbb{G}_m$, it is viewed as a reference, and the graph-cut algorithm [2, 10] is applied to optimize the objective function Eq.(9) to obtain its corresponding semantic region. In this way, all of the semantic regions are obtained. Trajectories which fit the same semantic regions are viewed as a cluster.

5. Learning Semantic Scene Models

For each cluster of trajectories, to learn semantic scene models, Mean-Shift algorithm is adopted to search the primary trajectory and entry/exit points. The measure as [20] is taken to extend the distribution of the trajectories. Table 1 is our algorithm summary. Two classifiers based on independent features are trained by our co-training framework to classify objects into vehicles or pedestrians. For each class of trajectories, it is parameterized and motion patterns learned by the GMM algorithm are clustered by the graph-cut algorithm. Finally, trajectories are clustered and semantic scene models are learned.

1. Classify object. (Section 3)
   (a) Train the LDA-based classifier and the AdaBoost classifier with our co-training framework.
   (b) Classify each foreground object into vehicle or pedestrian by the trained classifiers.
2. Cluster trajectory. (Section 4)
   For each kind of object (vehicle vs. pedestrian):
   (a) Parameterize its trajectories.(Section 4.1)
   (b) Learn the motion patterns ($\mathbb{G}$) based on the parameters for each block by the GMM algorithm.(Section 4.2)
   (c) Cluster the motion patterns ($\mathbb{G}_m$) by the graph-cut algorithm to group the trajectories.(Section 4.3)
   • Obtain corresponding semantic region for each motion pattern (g_r) in $\mathbb{G}_m$ by optimizing the objective function Eq.(9).
   • Cluster Trajectories based on these semantic regions.
3. Learn semantic scene model for each cluster of trajectories. (Section 5)

Table 1. *Algorithm summary.*

6. Experimental Results

6.1. Results of Object Classification

We implement real-time background subtraction and tracking as [16], so that moving objects can be reasonably separated from background and training samples can be obtained. We compare classifiers trained with labeled samples and our co-training framework in five scenes. The AdaBoost classifier is trained with $20,213$ positive samples (pedestrians) and $41,934$ negative samples (vehicles) labeled manually. These samples are obtained by normalizing blobs to 20×20 pixels and collected per 10 frames to reduce the correlation. The LDA-based classifier is trained with $12,000$ positive samples and $35,000$ negative samples for each scene using scene context features. Our classifiers are initialized with $2,720$ positive samples (pedestrians) and $6,716$ negative samples (vehicles) labeled manually in a data set, then they are trained with our co-training framework. For a different training set, the AdaBoost classifier

is used to classify unlabeled examples to obtain their labels and add those newly labeled examples which are confident enough to update the training set for LDA-based classifier training, then the two classifiers are training for each other. In this way, the training set of our classifiers becomes large and their classification correct rate is rising. In applications, the output of the classifier with more confidence is used to give the final classification decision. The objects in the test set are all not included in the training set. Appearances of objects vary significantly due to shadow, object merging as in Figure 7(f), so it is difficult for the AdaBoost classifier to have a good performance. Positions of objects effect mostly the classification of the LDA-based classifier. Comparing to the AdaBoost classifier and the LDA-based classifier, our classifiers do not require a very large set of labeled training data and achieve more considerable performance in diverse scenes. Table 2 shows the classification results.

Scene	S1	S2	S3	S4	S5
LDA-Based classifier	87.1%	88.3%	91.3%	91.5%	82.5%
AdaBoost calssifier	91.1%	87.3%	89.8%	90.3%	80.5%
Our	98.2%	97.3%	96.6%	97.4%	96.8%

Table 2. Classification results of the three classifiers. Our classifier is best because it fuses multiple features and enlarges training set from unlabeled samples.

6.2. Results of Trajectory Clustering

After classifying different types of trajectories (vehicles vs. pedestrians) with object classifiers, each class of trajectories is further clustered. We have tested our algorithm of trajectory clustering in many scenes. Due to limited space, the results of clustering trajectories of vehicles in two typical scenes S1 and S2 (see Fig. 3), which include straight road and crossroad.

Before clustering, 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 each 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. Otherwise, the trajectory is viewed as a noisy trajectory and deleted. In this way, 364 and 417 trajectories are selected in scene S1, S2 respectively.

Trajectory Description. Each trajectory obtained by the visual tracker is formulated as a quadratic curve, and the parameters (a, b, c) are calculated by least squares. Noisy points have a bad effect on the least squares algorithm. Therefore, every trajectory is preprocessed to delete these points. For a trajectory $T = \{(x_1, y_1), (x_2, y_2), \cdots, (x_n, y_n)\}$, 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 $\{\Delta d_i, i = 1, 2, \cdots, n-2\}$ can be obtained. If $\left\|\frac{\Delta d_i - u}{\delta}\right\| > 2.5$, 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 in this way. For complex trajectories, they can be treated as the combination of multiple simple trajectories. Some results of trajectory description in a parametric way are showed in Figure 3.

(a) (b) (c) (d)

Figure 3. Examples of trajectory fitting. The white arrows are the moving directions of objects. (a) and (c): The trajectories in scene $S1$ and $S2$ respectively. (b) and (d): The results of trajectory fitting.

Trajectory Clustering. Three trajectory clustering methods are compared in our experiments. For clarity, they are described as I, II and III. I: As mentioned in paper [20], the modified Hausdorff distance is viewed as trajectory similarities and use spectral clustering [12]; II: Based on the parameters of trajectories, Euclidean distance is considered as trajectory similarities and use spectral clustering [12]. III: Cluster trajectories based on their distributions. The method I is more time-consuming than II and III. Table 3 gives a quantitative comparison in computation cost on a P4 3.0GHz CPU. Our method III 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. The corresponding computation time is showed in Table 3. The second step is clustering these motion patterns. For scene S1 and S2, the scene image 320×240 is cut into 16×16 blocks. Blocks having similar motion patterns are clustered to construct semantic region. Based on the weight of Gaussian model, 9 and 18 motion patterns are selected as the primary motion patterns for scene S1 and S2 respectively. The corresponding computation time with the graph-cut algorithm is 10 seconds and 19 seconds. Because some primary motion patterns are similar, it causes that their corresponding semantic regions may have the same blocks. After removing the same semantic regions, all of the primary semantic regions are obtained.

TN	200	400	600	800	1000
I (seconds)	1112	4138	11687	17578	28072
II (seconds)	2	5	12	23	38
III (seconds)	4.8	5.6	6.0	7.2	8.3

Table 3. The first row (TN) is the number of trajectory. The second and third row are the corresponding computation time of I and II, the unit is second. The fourth row is the corresponding time of learning motion patterns with the GMM algorithm.

Some results are showed in Fig. 4. 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 scene S1 are illustrated in table 4. For the 364 trajectories, there are six clusters. The ground truth annotation of each cluster is labeled manually. There are 57, 110, 41, 29, 86 and 41 trajectories for cluster 1, 2, 3, 4, 5 and 6 respectively. Recall and Precision are used to measure the performance. For scene S2, the results of the three methods are shown in Figure 5. These results show our method III based on spatial distribution of trajectories performs the best.

Method	Cluster	TP	FN	FP	Recall	Precision
I	1	57	0	19	100%	75.0%
	2	88	22	1	80.0%	98.9%
	3	40	1	3	97.6%	93.0%
	4	29	0	5	100%	85.3%
	5	81	5	2	94.2%	97.6%
	6	39	2	0	95.1%	100%
II	1	56	1	18	98.2%	75.7%
	2	84	26	1	76.4%	98.8%
	3	40	1	9	97.6%	81.6%
	4	27	2	10	93.1%	73.0%
	5	72	14	4	83.7%	94.7%
	6	38	3	5	92.7%	88.4%
III (Our)	1	57	0	0	100%	100%
	2	107	3	0	97.3%	100%
	3	41	0	3	100%	93.2%
	4	29	0	4	100%	87.9%
	5	81	5	0	94.2%	100%
	6	41	0	1	100%	97.6%

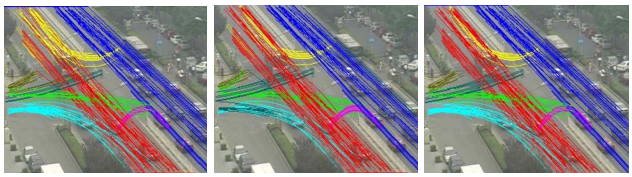
Table 4. The Precision and Recall of the three methods in scene S1. TP is true positive, FN is false negative, FP is false positive.

6.3. Results of Learning Semantic Scene Models

For each cluster of trajectories, the methods in Section 5 are used to learn semantic scene models. The path region is obtained by thresholding the density distribution. The learned semantic scene models of scene S1 and S2 are showed in Figure 6. The red lines are the primary trajectories, which represent the primary motion patterns. The start/end points of these lines are viewed as entry/exit points. The white arrows are the moving directions of objects.

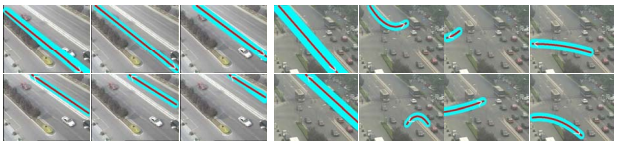
(a) (b)

Figure 4. Results of clustering motion patterns. (a) The six semantic regions in scene S1. (a) The eight semantic regions in scene S2.



(a) (b) (c)

Figure 5. (a), (b) and (c) are the results of trajectory clustering by I, II and III respectively.

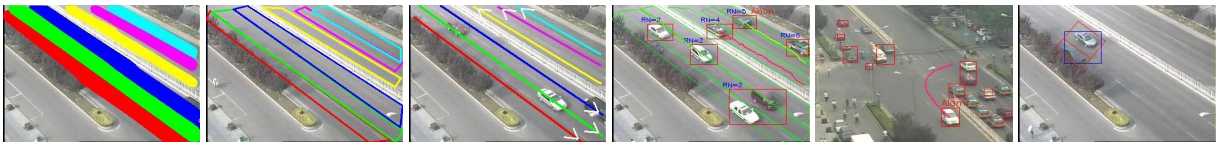


(a) (b)

Figure 6. (a) and (b) are results of semantic scene models in scene S1 and S2 respectively. The white arrows are the moving directions of objects, and the red curves are the primary trajectories.

6.4. Applications in Abnormal Detection and Object Segmentation

The learned semantic scene models can be directly used to real-time detection of abnormal behaviors in traffic scenarios. For scene S1 in Fig. 7(b), boundaries of the six semantic scene models for vehicles are described with a rectangle. The six paths are labeled from 1 to 6, for example, "RN=2" expresses the vehicle in the second path. When a vehicle moves from a path to another, the lane-merging activity happens. Fig. 7(d) shows a result. For scene S2, when an object enters the scene, it is classified into vehicle or pedestrian. For each vehicle/pedestrian class, we have already learnt the primary motion patterns for each block. For a trajectory, if its probability estimated by EQ.(1) is small, then we view it as an abnormal activity. An example is showed in Fig. 7(e). In addition, based on the semantic scene models, merged objects detected by the GMM algorithm can be segmented, a result is showed in Fig. 7(f).



(a) (b) (c) (d) (e) (f)

Figure 7. (a): The six semantic scene models of vehicles in scene S1. (b): The boundaries of these models in scene S1. (c): The primary trajectories of the six semantic scene models. (d): Lane-merging detection in scene S1. (e): Anomalous trajectory detection in scene S2. (f): The blue box is the result of GMM detector, the red box is the result of segmentation based on the semantic scene models in scene S1.

7. Conclusions

In this paper, a novel framework is proposed to learn semantic scene models. First, classifier trained under co-training framework is adopted to classify trajectories into different types (vehicles vs. pedestrians). Second, each class of trajectories is further clustered by the spatial distributions of trajectories (trajectories' parameters and moving directions of objects). For each cluster of trajectories, semantic scene models are learnt and used in some applications. Experimental results validate the effectiveness and efficiency of our framework.

Acknowledgements

This work was supported by the following funding resources: National Natural Science Foundation Project #60833006, #60835002, #60518002. National Science and Technology Support Program Project #2006BAK08B06, National Hi-Tech (863) Program Project #2008AA01Z124.

References

[1] A. Blum and T. Mitchell. Combining labeled and unlabeled data with co-training. In *COLT: Proceedings of the Workshop on Computational Learning Theory, Morgan Kaufmann Publishers*, pages 92–100, 1998. 3

[2] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE Trans. On PAMI*, 23:1222–1239, November 2001. 5

[3] D.Makris and T.Ellis. Automatic learning of an activity-based semantic scene model. In *Proc. of IEEE Conference on Advanced Video and Signal Based Surveillance*, 2003. 1, 2

[4] J. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression: A statistical view of boosting. In *Annals of Statistics*, 2000. 3

[5] Z. Fu, W. Hu, and T. Tan. Similarity based vehicle trajectory clustering and anomaly detection. *ICIP 2005*, 2:II–602–5, Sept. 2005. 1

[6] A. Hadid, M. Pietikainen, and T. Ahonen. A discriminative feature space for detecting and recognizing faces. *IEEE Conference on CVPR*, 2:II–797–II–804 Vol.2, 2004. 3

[7] J.H.Fernyhough, A.G.Cohn, and D.C.Hogg. Generation of semantic region from image sequences. In *ECCV*, 1996. 1, 2

[8] I. Junejo, O. Javed, and M. Shah. Multi feature path modeling for video surveillance. *ICPR 2004*, 2:716–719 Vol.2, Aug. 2004. 1

[9] E. Keogh. Exact indexing of dynamic time warping. *In 28th International Conference on Very Large Data Bases*, pages 406–417, 2002. 1

[10] V. Kolmogorov and R. Zabih. What energy functions can be minimized via graph cuts? *IEEE Trans. On PAMI*, 26:147–159, May 2004. 5

[11] A. Levin, P. Viola, and Y. Freund. Unsupervised improvement of visual detectors using co-training. *IEEE Conference on ICCV*, pages 626–633 vol.1, Oct. 2003. 3

[12] M. Meila and J. Shi. A random walks view of spectral segmentation. In *In AI and Statistics (AISTATS*, 2001. 6

[13] I. N.Junejo and H. Foroosh. Trajectory rectification and path modeling for video surveillance. In *ICCV*. IEEE, 2007. 1

[14] T. Ojala, M. Pietikainen, and D. Harwood. A comparative study of texture measures with classification based on feature distributions. *Pattern Recognition*, pages 51–59, 1996. 3

[15] S.J.Mckenna and H.Nait-Charif. Learning spatial context from tracking using penalized likelihood. In *Proc. of ICPR*, 2004. 2

[16] C. Stauffer and W. E. L. Grimson. Learning patterns of activity using real-time tracking. *IEEE Trans. on PAMI*, 22(8):747–757, August 2000. 2, 5

[17] A. Torralba, K. Murphy, and W. Freeman. Sharing features: efficient boosting procedures for multiclass object detection. *IEEE Conference on CVPR*, 2:II–762–II–769 Vol.2, 2004. 3

[18] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. *IEEE Conference on CVPR*, 1:I–511–I–518 vol.1, 2001. 3

[19] X. Wang, K. T. Ma, G.-W. Ng, and E. Grimson. Trajectory analysis and semantic region modeling using a nonparametric bayesian model. In *CVPR*, 2008. 1, 2

[20] X. Wang, K. Tieu, and E. Grimson. Learning semantic scene models by trajectory analysis. In *ECCV*, volume III, pages 100–123, 2006. 1, 2, 3, 5, 6

[21] X. Yuan and S. Z. Li. Half quadratic analysis for mean shift: with extension to a sequential data mode-seeking method. In *IEEE International Conference on Computer Vision*, 2007. 2

[22] L. Zhang, S. Li, X. Yuan, and S. Xiang. Real-time object classification in video surveillance based on appearance learning. *VS2007*. 2, 3

[23] T. Zhang, S. Z. Li, S. Xiang, L. Zhang, and S. Liu. Co-training based segmentation of merged moving objects. In *VS2008*. 3