On-line and Unsupervised Learning for Codebook based Visual Recognition

Jie XU

School of Computer Science

Faculty of Engineering

University of New South Wales

A thesis submitted for the degree of

Doctor of Philosophy

2011

PLEASE TYPE

THE UNIVERSITY OF NEW SOUTH WALES
Thesis/Dissertation Sheet

Surname or Family name: **XU**

First name: **JIE** Other name/s:

Abbreviation for degree as given in the University calendar: **PhD**

School: **Computer Science and Engineering** Faculty: **Engineering**

Title: **On-line and Unsupervised Learning for Codebook based Visual Recognition**

Abstract 350 words maximum

In this thesis we develop unsupervised and on-line learning algorithms for codebook based visual recognition tasks. First, we study the Probabilistic Latent Semantic Analysis (PLSA), which is one instance of codebook based recognition models. It has been successfully applied to visual recognition tasks, such as image categorization, action recognition, etc. However it has been learned mainly in batch mode, and therefore it cannot handle the data that arrives sequentially. We propose a novel on-line learning algorithm for learning the parameters of the PLSA under that situation. Our contributions are two-fold: (i) an on-line learning algorithm that learns the parameters of the PLSA model from incoming data; (ii) a codebook adaptation algorithm that can capture the full characteristics of all features during the learning. Experimental results demonstrate that the proposed algorithm can handle sequentially arriving data that the batch PLSA learning cannot cope with.

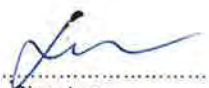
We then look at the Implicit Shape Model (ISM) for object detection. ISM is a codebook based model in which object information is retained in codebooks. Existing ISM based methods require manual labeling of training data. We propose an algorithm that can label the training data automatically. We also propose a method for identifying moving edges in video frames so that object hypotheses can be generated only from the moving edges. We compare the proposed algorithm with a background subtraction based moving object detection algorithm. The experimental results demonstrate that the proposed algorithm achieves comparable performance to the background subtraction based counterpart, and it even outperforms the counterpart in complex situations.

We then extend the aforementioned batch algorithm for on-line learning. We propose an on-line training data collection algorithm and also an on-line codebook based object detector. We evaluate the algorithm on three video datasets. The experimental results demonstrate that our algorithm outperforms the state-of-the-art on-line conservative learning algorithm.

Declaration relating to disposition of project thesis/dissertation

I hereby grant to the University of New South Wales or its agents the right to archive and to make available my thesis or dissertation in whole or in part in the University libraries in all forms of media, now or here after known, subject to the provisions of the Copyright Act 1968. I retain all property rights, such as patent rights. I also retain the right to use in future works (such as articles or books) all or part of this thesis or dissertation.

I also authorise University Microfilms to use the 350 word abstract of my thesis in Dissertation Abstracts International (this is applicable to doctoral theses only).



Signature

Witness

Jan 05, 2011

Date

The University recognises that there may be exceptional circumstances requiring restrictions on copying or conditions on use. Requests for restriction for a period of up to 2 years must be made in writing. Requests for a longer period of restriction may be considered in exceptional circumstances and require the approval of the Dean of Graduate Research.

FOR OFFICE USE ONLY Date of completion of requirements for Award:

THIS SHEET IS TO BE GLUED TO THE INSIDE FRONT COVER OF THE THESIS

Copyright Statement

Authenticity Statement

I certify that the Library deposit digital copy is a direct equivalent of the final officially approved version of my thesis. No emendation of content has occurred and if there are any minor variations in formatting, they are the result of the conversion to digital format.

Jie XU,

Jan 05, 2011

Originality Statement

I hereby declare that this submission is my own work and to the best of my knowledge it contains no materials previously published or written by another person, or substantial proportions of material which have been accepted for the award of any other degree or diploma at UNSW or any other educational institution, except where due acknowledgement is made in the thesis. Any contribution made to the research by others, with whom I have worked at UNSW or elsewhere, is explicitly acknowledged in the thesis. I also declare that the intellectual content of this thesis is the product of my own work, except to the extent that assistance from others in the project's design and conception or in style, presentation and linguistic expression is acknowledged.

Jie XU,

Jan 05, 2011

Acknowledgements

First of all, I would like to thank my thesis supervisors: Dr. Yang Wang, Dr. Getian Ye and Dr. Wei Wang. This thesis would not have been completed without their commitment and support.

Dr. Yang Wang has become my NICTA supervisor since 2009. I would like to express my sincere gratitude to Yang, for his support, guidance and simply everything in my Ph.D. study. Doing research under his supervision has been a very pleasant time of my life. His knowledge, patience, kindness, and enthusiasm have brought out the best from me. I have learnt a lot from him, especially on problem formulation, experiment design and technical writing. I cannot imagine this thesis being finished without him.

Dr. Getian Ye was my supervisor between 2006 and 2009. I would like to thank Getian, for his caring supervision during that period of time. As a supervisor, Getian endeavoured to teach me the fundamental research skills. Thanks to his effort, I managed to determine my research topic.

Dr. Wei Wang has been my CSE supervisor since my bachelor's study. I would like to thank Dr. Wei Wang, for his continuous encouragement and support. Wei is a very bright person, and I have learned a lot from him through our discussion. Furthermore, Wei is very knowledgeable in a wide range of areas. His passion for research will always been my source of inspiration.

I would like to thank Dr. Jian Zhang, for introducing me to the research discipline. Additionally, I am also grateful to Prof Shin'ichi Satoh for his support to my internship at National Institute of Informatics in Japan. I want to thank all the colleagues in NICTA: Bang

Matt Zhang, Gunawan Herman, Sakrapee (Paul) Paisitkriangkrai, Nobuyuki Morioka, Pranam Janney, Zhidong Li, Tuan Hue Thi and Jun Yang. Thank you for making our research lab an active research place.

Last but not least, my special thanks belong to my family. I would like to express my deepest gratitude to my parents, and also my wife Rosanne for their support and love all the way.

Abstract

In this thesis we develop unsupervised and on-line learning algorithms for codebook based visual recognition tasks. First, we study the Probabilistic Latent Semantic Analysis (PLSA), which is one instance of codebook based recognition models. It has been successfully applied to visual recognition tasks, such as image categorization, action recognition, etc. However it has been learned mainly in batch mode, and therefore it cannot handle the data that arrives sequentially. We propose a novel on-line learning algorithm for learning the parameters of the PLSA under that situation. Our contributions are two-fold: (i) an on-line learning algorithm that learns the parameters of the PLSA model from incoming data; (ii) a codebook adaptation algorithm that can capture the full characteristics of all features during the learning. Experimental results demonstrate that the proposed algorithm can handle sequentially arriving data that the batch PLSA learning cannot cope with.

We then look at the Implicit Shape Model (ISM) for object detection. ISM is a codebook based model in which object information is retained in codebooks. Existing ISM based methods require manual labeling of training data. We propose an algorithm that can label the training data automatically. We also propose a method for identifying moving edges in video frames so that object hypotheses can be generated only from the moving edges. We compare the proposed algorithm with a background subtraction based moving object detection algorithm. The experimental results demonstrate that the proposed algorithm achieves comparable performance to the background subtraction based counterpart, and it even outperforms the counterpart in complex situations.

We then extend the aforementioned batch algorithm for on-line learning. We propose an on-line training data collection algorithm and also an on-line codebook based object detector. We evaluate the algorithm on three video datasets. The experimental results demonstrate that our algorithm outperforms the state-of-the-art on-line conservative learning algorithm.

Contents

List of Figures

Chapter 1

Introduction

1.1 Background

This thesis deals with computer based visual recognition problems. Generally speaking, visual recognition is the process of assigning labels to images and videos, according to some rules. For human beings, visual recognition is usually effortless and robust in different environments, regardless of lighting conditions and viewpoint variations. The performance of computers on visual recognition tasks is still far behind that of human beings. As a result, visual recognition still remains an active area for computer vision research.

There have already been some computer-based visual recognition applications, and they are listed as follows:

- Intelligent Video Surveillance - Many buildings have been equipped with close-circuit TV cameras to ensure the security inside premises. Manual scrutinizing of videos is tedious and inefficient. Intelligent video surveillance systems, which detects abnormal activities in the videos, are employed to assist human operators for video scrutiny.

- Content based Image Search - The Internet has become a huge repository of images. This has led to the demand for intelligent web search engines for images. Traditional image search is based on keywords, where the similarity

of images are determined by their labels. A more natural way of image search is based on image content. Given a query image, content based image search engine can return visually similar images to the exemplar. One example is Google's Picasa Digital Image Organizer, which provides a built-in function that associates faces with identities from version 3.5 onwards. Using a face image as the query image, the database returns photos associated with that face.

- Face Detection - An face detection system is an application that detects human faces from given images or videos. It is perhaps the most popular computer vision application in our daily life. For example, Canon digital cameras usually come with a built-in face detect function to assist users to capture better photos.

- Computer Aided Diagnosis - Medical imaging techniques, such as X-radiation (X-RAY) and Magnetic Resonance Image (MRI), are playing a more and more important role in medical examinations. Medical images contain a great deal of information which should be analyzed comprehensively within a short time. To achieve fast and reliable analysis, Computer Aided Diagnosis (CAD) systems are employed to assist doctors in interpreting the medical images. They can help to identify possible diseases by highlighting conspicuous regions in the images.

Computer based visual recognition is a difficult problem, due to the following challenges:

- Illumination difference - Given the same scene, images captured under different illumination conditions can exhibit great difference. Computers require advanced recognition algorithms to accomplish visual recognition under different lighting conditions.

- Background clutter - In most image collections, objects are not segmented from the background. Cluttered background can add to the difficulty of recognizing objects inside images.

- Occlusions - Due to viewpoint variations, part of the same object can be covered by another object. With partially available appearance information, computers require sophisticated recognition algorithms to identify the same object.

Successful visual recognition approaches are expected to handle these aforementioned issues.

It is important to note that the visual recognition is a multi-facet problem. Given images and videos, we might be interested in the scene category to which an image belong; we might want to know if there is a certain event occurring in surveillance videos; we might even want to localize pedestrians in the videos. To confine the scope of our thesis, related terminologies are explained as follows:

- Class and Category - Both words are used interchangeably. In this thesis, they are referred to as sets of visually consistent data. For example, the "beach" image class includes images that capture beach scenes; the "running" action video category includes the videos that exhibit running activities.

- Categorization - It is the task of determining which category the data belongs to, from pre-defined categories. For example, action categorization determines the action category of a given video clip.

- Detection - It is the task of specifying the locations of target objects inside images or videos. For example, pedestrian detection entails the task of localizing pedestrians inside images or videos.

- Localization - the same as detection.

The focus of this thesis is image/video categorization, and object detection from videos.

1.2 Task Definition

1.2.1 Codebook based Visual Recognition

Codebook based recognition is one of the widely used recognition methods for visual recognition. The idea of codebook learning can be explained using the following text reading example. Given a text document, our understanding of it relies on the interpretation of its words (See Figure 1.1). The meaning of each word can be obtained from a dictionary. With the help of the dictionary, we can understand any given document based on our understanding of its words. In this example, the dictionary serves as a bridge that connects our high-level document understanding and low-level word interpretations.

Analogously, computers can accomplish the task of visual recognition in a similar way. In order to let computers understand images/videos, a visual dictionary can be constructed (See Figure 1.2 and 1.3). The constructed visual dictionary retains our knowledge, which can help computers understand the content of images/videos. This kind of visual dictionary is usually termed as a "codebook" in computer vision. Codebooks are constructed by applying some "coding" rules to features extracted from images/videos. The coding rules are essentially clustering algorithms, which range from the traditional k-means clustering Sivic *et al.* (2005) to randomized tree based methods Moosmann *et al.* (2006) recently. A codebook is comprised of the cluster centers. Using the codebook, imaging data can be converted into an intermediate representation, upon which sophisticated models can be designed for recognition.

Codebook based recognition methods have been demonstrated effective for visual recognition problems Sivic *et al.* (2005), Csurka *et al.* (2004), Moosmann *et al.* (2006). In this thesis, we will focus on three instances of codebook based recognition methods.

1.2.2 On-line and Unsupervised Learning

Using a codebook, we can obtain an intermediate representation for raw data. For recognition purpose, different statistical models can be learned upon the inter-

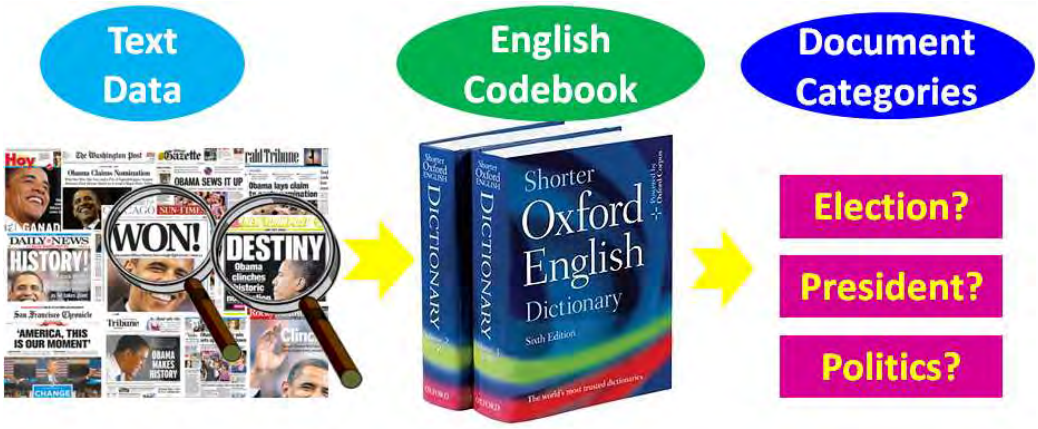


Figure 1.1: With the help of a dictionary, we can classify the given document on our understanding of the words in the document.

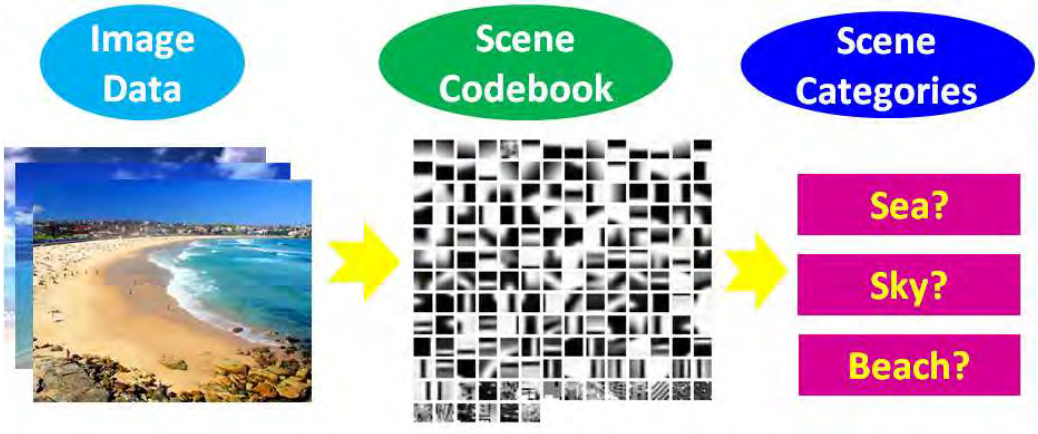


Figure 1.2: With the help of a codebook, computers can classify scene images into different categories.

mediate representation. The purpose of model learning is to produce a mapping which maps the representation into semantic classes.

Traditional model learning is often done in off-line mode. In off-line learning, the entire training set is given in advance and the model is learned in a batch mode. The pitfalls of off-line learning include the following issues : (i) all training data must be available so that they can be accessed at the same time; (ii) once a

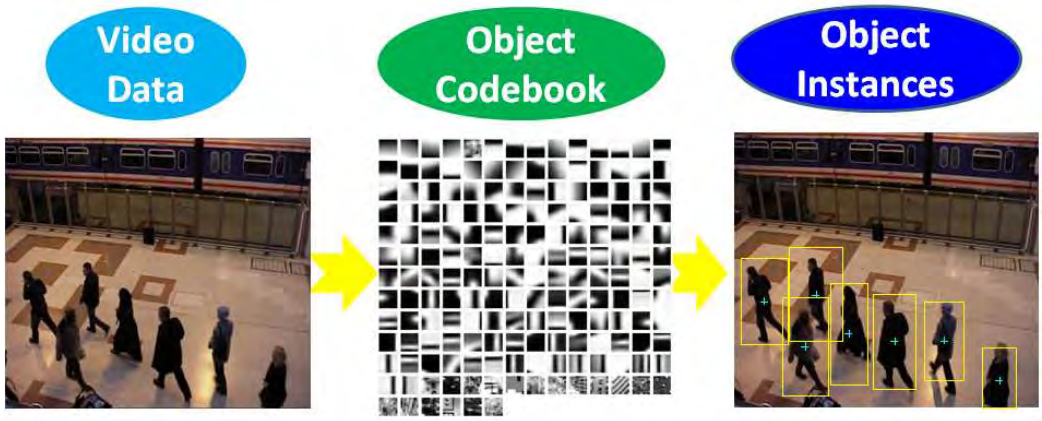


Figure 1.3: With the help of a codebook, computers can detect objects in videos.

model is learned, it cannot adapt to new data. Before the advance of the Internet, it is expensive to obtain a large amount of training data. However, a huge amount of images and videos can be easily obtained nowadays. There is a heavy memory consumption for such gigantic amount of data if the learning is conducted in off-line mode. Furthermore, training data might even arrive sequentially like daily news data. Under this circumstance, not all training data is available at the same time. To cope with these new challenges, on-line learning is called for. Unlike the off-line learning, on-line learning keeps adjusting the existing model using new data. Its memory requirement can be lowered since only part of training data is needed at each learning epoch. Moreover, it also enables the model adaptation to new data. As a result, developing on-line learning algorithms for recognition is our first focus in this thesis.

Traditional learning for visual recognition is supervised learning, in which manual labeling of training data is required. Supervised approaches can achieve high recognition accuracies when there is enough labeled training data. This is especially true for many object detection applications in computer vision. However, labeled data is hard to obtain for many problems, whereas unlabeled data can usually be obtained free of cost. As a result, investigating methods for unsupervised learning using a large amount of unlabeled data is becoming more and more important. Hence, developing unsupervised learning methods for visual recognition is another focus of this thesis.

1.3 Contributions

In this thesis, we develop on-line and unsupervised learning algorithms for codebook based visual recognition. Our contributions are introduced in the following sections.

1.3.1 On-line Learning for codebook based Visual Categorization

In this thesis, firstly, we develop an on-line learning algorithm for codebook based image/video categorization. The codebook based learning model we look at is the Probabilistic Latent Semantic Analysis (PLSA). The PLSA is a latent topic model which originates from the text mining community. It is originally proposed for document categorization. However it has been successfully applied to some visual recognition tasks, such as image and video classification. For visual recognition, the learning of the PLSA requires a codebook, which is usually formed by running the k-means clustering on extracted feature descriptors.

The PLSA models have been learned mainly in batch mode, which requires that all training data should be accessed at the same time. Such batch learning has an excessive memory requirement when dealing with large datasets. Furthermore, it cannot handle the data that arrives sequentially. To tackle these disadvantages of batch learning, we propose an on-line learning algorithm for the PLSA. Our proposed learning algorithm can learn the PLSA model from sequentially arriving data. Our contributions are two-fold: (i) an on-line learning algorithm that learns the parameters of the PLSA model from incoming data; (ii) a codebook adaptation algorithm that can capture the full characteristics of all features during the learning. Experimental results demonstrate that the proposed algorithm can handle sequentially arriving data.

1.3.2 Unsupervised Learning for codebook based Visual Detection

Our second contribution is an unsupervised learning algorithm for codebook based moving object detection from videos. The codebook based object model we study is the Implicit Shape Model Leibe *et al.* (2008). The Implicit Shape Model (ISM) is a part-based model for object detection. At the learning stage, local features are extracted from training data. A codebook of local features is then constructed by clustering all the extracted features. Semantic information is also kept in the codebook. At runtime, local features are extracted from images, and then used to match against the codewords. The codewords then cast votes for object hypotheses.

The ISM has been shown effective for object detection, however manual labeling of training data is required for the model. In this thesis, we propose an unsupervised learning method using Multiple Instance Learning to replace the manual labeling of learning. Moreover, we also propose an moving edge detection scheme to reduce the hypothesis searching at runtime. We evaluate the proposed algorithms on three video datasets, and experimental results demonstrate the efficacy of our proposed methods.

1.3.3 Unsupervised On-line Learning for codebook based Visual Detection

The last contribution of our thesis is an unsupervised on-line learning algorithm for codebook based moving object detection. The codebook based object model we look at is also the Implicit Shape Model Leibe *et al.* (2008). But we use another type of codebook, which possesses a tree-like structure.

This on-line algorithm is an extension to the aforementioned unsupervised object detection method which learns an ISM model in off-line mode. Once the learning accomplishes, the model is fixed and no adaptation to new data can be made. However it is important to adapt the object model to emerging objects, especially in video surveillance. Our contribution includes an on-line training data collection algorithm and also an on-line codebook based object

detector. We evaluate the proposed algorithm on three video datasets. The experimental results shows that the proposed algorithm outperforms the state-of-the-art unsupervised on-line detection algorithm.

In addition to the work presented in the thesis, I also work on some other research projects. The list of all the related publications during my PhD study can be found in the Appendix.

1.4 Thesis Structure

The structure of this thesis is summarized as follows: in Chapter 2 we introduce a general learning framework for visual recognition, which includes feature extraction, data representation and modeling learning. We also review the literature for visual recognition in Chapter 2. It is noted that Chapter 2 only covers a high level overview of the literature. Details of the related work is presented in each chapter.

In Chapter 3 we detail our proposed on-line PLSA algorithm for visual categorization. We compare our on-line PLSA with the conventional PLSA and also the QB-PLSA. Finally we review the performance improvements of our on-line PLSA on the task of scene and action categorization.

In Chapter 4 we present our unsupervised learning method for object detection. We discuss the advantage of the proposed algorithm over the existing state of the art. A series of experiments is conducted to demonstrate the performance of the proposed framework. We extend this algorithm in Chapter 5 to enable its on-line adaptation for object detection. Experiments are also conducted to show the performance of the on-line learning algorithm.

Finally in Chapter 6, we draw conclusions upon our contributions, and discuss about the future work.

Chapter 2

Literature Review

This thesis investigates on-line and unsupervised learning for visual recognition problems. Visual recognition is essentially a pattern recognition problem. In this chapter we first introduce a pattern recognition framework. We then review the literature for visual recognition, especially on the tasks of object, scene and action recognition.

2.1 A Recognition Framework

2.1.1 An Overview

Visual recognition is essentially a pattern recognition problem, whose goal is to label unseen imaging data based on observations from existing data. Generally, a recognition system usually comprises three major components, namely feature extraction, data modeling and model learning Duda *et al.* (2001).

As the first step, the feature extraction characterizes raw data with measurements that have high intra-class similarity and low inter-class similarity. The data modeling describes the relationship between observed and unseen data based on the extracted features. Different models contain different settings, whose parameters can be learned through the model learning. Details of each components are discussed in the following sections.

2.1.2 Feature Extraction

Feature extraction can be divided into two steps, namely feature detection and features description. Specifically, the feature detection identifies the distinctiveness of data, whereas the feature description describes the distinctiveness of data. The final output of feature extraction is a set of feature descriptors used for high-level processing. Different combination of feature detector and descriptors can impact recognition performance Mikolajczyk & Schmid (2005).

2.1.2.1 Feature Detectors

In this section we review the literature on low-level image feature detection.

Edges Edges are boundaries between two regions in images. More formally, edges are points in an image with discontinuities of image brightness. These points usually possess strong gradient magnitudes due to sharp changes in the brightness. As a result, edge detection algorithms usually rely on the computation of image gradients.

The approaches for edge detection can be divided into first-order and second-order methods. They differ in the way they compute edge responses. The first-order methods compute the edge responses using first-order derivatives such as gradient magnitude. It then searches for local maximas from the responses. Canny edge detection Canny (1987) can be considered as a first-order method. On the other hand, the second-order method searches for zero-crossing in second-order image derivatives. The **Marr-Hildreth** Brinks (2008) algorithm is one of those second-order methods for edge detection. Both methods require post-processing after the computation of edge responses.

Interest Points Interest points are distinctive points in images or videos, whose local neighborhood is rich in information. The term "corner" and "interest point" are used interchangeably in the literature. A corner point only means an intersection of edges. However an interest point can be a corner, and it can also be an isolated point with local intensity extrema.

There has been extensive research on interest point detection in the literature. The Moravec interest operator Moravec (1979) is one of the pioneer work on interest point detection. The response of each point is measured by the similarity between the patch centered at each point to patches nearby. The similarity is measured by the sum of squared differences (SSD) between the two patches. The response is anisotropic since the similarity is only calculated in the eight principle directions. As a result, the operator is rotationally invariant. To improve the Morave operator, Harris and Stephens propose the Harris operator Harris & Stephens (1988) which considers the differential of SSD with respect to directions directly. The Harris operator is simple and effective for interest point detection, and therefore it is still very popular in image processing. It has been extended for multi-scale interest point detection in Mikolajczyk & Schmid (2004). The multi-scale Harris operator can detect interest points invariant to translations, rotations and uniform rescaling in spatial domain.

To tackle the perspective distortions to images, the affine invariant interest point detector is proposed Mikolajczyk & Schmid (2004). Recently Lowe proposed the Scale Invariant Feature Transform (SIFT) detector Lowe (2004), which detects interest points invariant to image translation, scaling and robust to local geometric distortion.

Blobs Blobs are the image regions that have different appearance from their neighborhood; they can be brighter or darker than the surrounding regions.

Blob detectors can be categorized into the following two categories: (i) methods based on differential expressions; and (ii) methods based on local intensity extrema. Different operators are employed to the scale-space representation of images Lindeberg (1993); operators include the Laplacian operator and the Difference of Gaussians operators. On the other hand, intensity-based methods measure how stable a region is along the intensity dimension. The Maximally Stable Extremal Regions (MSER) detector Matas *et al.* (2004) is one of the intensity-based methods. It attempts to locate connected components in a thresholded image using a set of thresholds. The detected region is invariant to affine transformation of image intensities.

2.1.2.2 Feature Descriptors

Gradient based Descriptors Gradient based descriptors describe the gradient distribution of the region centered at the interest point. Scale Invariant Feature Transform (SIFT) descriptor is one of the popular descriptors for gradient description Lowe (2004). It computes the orientation histogram using magnitude and orientation values of the region centered at the interest point. The magnitude values are then weighted by a Gaussian function. The obtained histograms are then normalized to unit lengths to enhance the invariance to local affine distortions. The original SIFT descriptor yields a 128-dimensional descriptor. Various refinements have been proposed to improve this basic scheme. Ke *et al.* employ the Principal Component Analysis to normalize the gradient patch, which yields a 36-dimensional PCA-SIFT descriptor Ke & Sukthankar (2004). PCA-SIFT is faster for matching due to its low dimensional nature. However it is shown less distinctive than the original SIFT descriptor in Mikolajczyk & Schmid (2005). Mikolajczyk *et al.* propose another variant of SIFT called GLOH (Gradient Location and Orientation Histogram). The GLOH descriptor is shown to be more distinctive than the SIFT descriptor with the same number of dimensions Mikolajczyk & Schmid (2005), but it is computationally more expensive. To maintain the discriminative power of the SIFT descriptor and lower its computational cost, the SURF (Speeded Up Robust Features) descriptor is proposed in Bay *et al.* (2006). The computation of the SURF descriptor is claimed to be faster than the SIFT descriptor by its authors. They also claim that the SURF descriptor is more robust against different image distortion than the SIFT descriptor.

Shape based Descriptors Shape based descriptors are used for measuring similarity between shapes. The descriptor is one of shape based descriptors Belongie *et al.* (2002). This descriptor is computed at points along object contours. The key idea is to describe the distribution of relative position of points along the contour. The distribution is described by a local histogram of edge points in a radius-angle polar grid. It is employed for digit recognition, object recognition, and also trademark retrieval.

An issue of the shape context descriptor is that similar contours have very different histograms when contours are close to the bin boundaries. This can result in large distance between two similar shapes if L2-norm or $\chi 2$ distance is used. The Earth Moving Distance (EMD) Rubner *et al.* (1998) can be used to alleviate this problem. However it is computationally more expensive. Wang *et al.* propose an improved shape context descriptor to tackle this problem Wang *et al.* (2007). The idea is to overlap spans of adjacent angular bins so that the edge points in the overlapped regions are counted in both of the adjacent bins. As a result, two contours close to the original bin boundary will have similar histograms.

2.1.3 Data Modeling

The goal of data modeling is to relate observed data and unseen data. Depending on modeling schemes, the models can be divided into two categories, namely the generative models and the discriminative models.

2.1.3.1 Generative Models

Generative models are a class of models used for modeling the joint probability distribution of the observed variable y and unobserved variable x, i.e., $p(x, y)$. A generative model can be used to generate samples for any variables in the model. The parameters of the generative models are often estimated using maximum data likelihood method. Examples of the generative models include the Probabilistic Latent Semantic Analysis (PLSA) Hofmann (1999), the Latent Dirichlet Allocation (LDA) Blei *et al.* (2003), and also the Hidden Markov Model (HMM) Juang & Rabiner (1991).

2.1.3.2 Discriminative Models

In contrast to the generative models, the discriminative models are used to model the conditional probability distribution of $p(y|x)$. Unlike the generative models, discriminative models cannot generate samples for variables since there is no description of its underlying distribution. Examples of discriminative models

include the Support Vector Machine (SVM) Steinwart & Christmann (2008), the Boosting Freund & Schapire (1995), and also the Conditional Random Field Lafferty *et al.* (2001).

2.1.4 Model Learning

Once a model is determined, different style of machine learning algorithms can be employed to learn the model parameters.

2.1.4.1 Off-line and On-line Learning

Machine learning algorithms can be categorized based on how the learning occurs over time. Off-line learning, which is also called batch learning, requires that all input data are available at the time of learning. On the other hand, on-line learning does not have such requirement. The input data to the on-line learning can arrive continuously and sequentially. As a result, the model can adapt to incoming data during the learning life.

2.1.4.2 Supervised Learning

Machine learning algorithms can be categorized based on the levels of supervision. The supervised learning is the predominant kind of machine learning method. The input to the supervised learning consists of input data, and also the true interpretation of the data (the label of the data). The goal of supervised learning is to deduce a function which maps the input data to the interpretation defined by a supervisor.

The task of supervised learning is to predict the function outputs given any valid input data after learning from examples. Depending on the nature of the output data, supervised learning can be further classified as the *regressions* where outputs are continuous values, or *classifications* where outputs are discrete values.

2.1.4.3 Unsupervised Learning

Unlike the supervised learning, unsupervised learning works on the data without labels. The task of unsupervised learning is to determine the underlying structure of input data, and how they are organized. One kind of unsupervised learning is clustering, whose goal is to identify similar patterns inside the input data.

2.1.4.4 Semi-supervised Learning

Semi-supervised learning is a class of machine learning techniques whose input contains both labeled and unlabeled data for training. The input usually comprises a small amount of labeled data, and also a large amount of unlabeled data. Semi-supervised learning bridges the gap between supervised learning (with fully labeled data) and unsupervised learning (without any labeled data). It is of great practical significance to the problems where a fully labeled input data set is very expensive to obtain. It is shown to improve learning performance by jointly using unlabeled and labeled data Balcan *et al.* (2005).

2.2 Visual Recognition

In this section we review the literature on visual recognition. Visual recognition is often referred to as the task of categorizing images/videos into different categories. Historically, the work on visual recognition has been concentrated on objects, scenes and actions, due to respective practical importance. As such, related work is reviewed on these categories.

2.2.1 Object Recognition

Object recognition implies identifying objects inside given images. Specifically it involves detecting the existences of objects and localizing possible objects inside given images. Object recognition has been an active computer vision research area for decades, and it still remains a challenging area. An effective solution must be able to cope with several factors including background clutters and occlusions.

Numerous techniques have been proposed in the literature, which can be divided into global methods and local methods.

2.2.1.1 Global Methods

Global methods are characterized by the global representations of objects. On the basis of the global representation, object localization is achieved by scanning sliding windows on different locations. The global methods usually cast the object recognition problem into a binary classification problem. During training, an object classifier is constructed by learning the global statistics of robust image features from training images. At runtime, the learned classifier is applied over different locations and scales to predict the presence of objects in sub-windows. Different image features have been combined with this sliding window approach. For example, Dalai *et al.* combine the Histogram of Oriented Gradients (HoG) features with the linear SVM for pedestrian detection in images Dalal & Triggs (2005). Viola and Jones combine the Haar Wavelet with the

To speed up the hypotheses search, Lampert *et al.* propose an Efficient Subwindow Search (ESS) algorithm to perform object localization Lampert *et al.* (2008). The intuition of the ESS is that it is unnecessary to evaluate the quality function for all subwindows if only the best few subwindows are required; hence the search should be targeted directly to the subwindows that have the highest score by ignoring the rest of the searching space. The ESS is based on the branch-and-bound framework, which hierarchically splits the searching space into disjoint subsets while keeping the upper bounds of quality functions for subsets. As a result, large parts of the searching space can be discarded early if their upper bounds are lower than a guaranteed score. Experimental results on a number of datasets demonstrate its improvements on speed for object detection Lampert *et al.* (2008).

Despite of its achievements, a major drawback of the ESS is that its time complexity varies widely from $O(n^2)$ to $O(n^4)$ for a $n \times n$ image. When an object is not in the image, the optimal subwindow score is low and the ESS may takes up to $O(n^4)$ number of iterations to converge. To address this problem, An *et al.* propose two subwindow searching methods (I-ESS and A-ESS) based

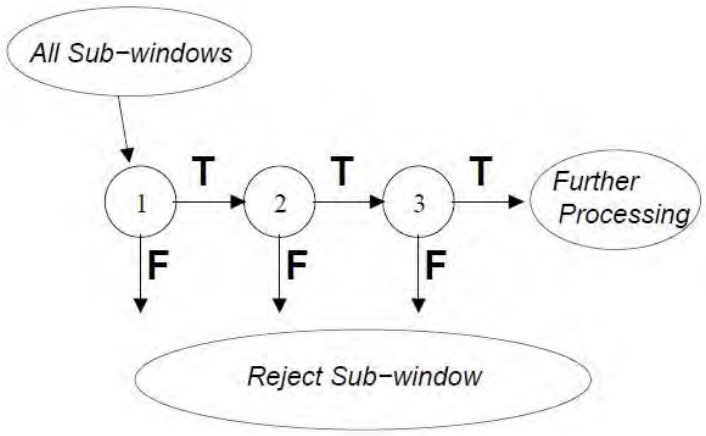


Figure 2.1: The cascade classification structure Viola & Jones (2002). A series of classifiers are applied to every subwindow. The classifier at each stage attempts to reject some number of subwindows. After several stages of processing, the number of subwindows would be radically reduced.

on the linear Kadane's Algorithm for 1D maximum subarray search in An *et al.* (2009). Experimental results show that I-ESS and A-ESS perform significantly faster than the ESS by reducing the worst case time complexity to $O(n^3)$ and $O(n^2)$ respectively.

The aforementioned branch-and-bound based algorithms have significantly improved localization speed. However for multi-class localization problems, they may rely on the linear scan of all the object models, which can be costly when there is an excessive number of object models. To improve the efficiency of large-scale object recognition, Yeh *et al.* propose an efficient method for concurrent object localization and recognition based on branch-and-bound Yeh *et al.* (2009). A data-dependent region hypothesis sampling scheme is proposed to efficiently select promising candidate regions. Experimental results show its superior performance in accuracy and speed compared to the ESS.

Different from the branch-and-bound framework, Viola and Jones propose a cascaded classification for efficient object localization in Viola & Jones (2002). Their method takes the advantages of the fact that an overwhelming majority

of the subwindows are negative. As show in Figure 2.1, a series of classifiers are applied to every subwindow. The classifier at each stage attempts to reject some number of subwindows. After several stages of processing, the number of subwindows would be greatly reduced. Essentially, the cascade attempts to reject as many negative subwindows as possible. A positive subwindow will trigger all the classifiers in the cascade. The training of the cascade classification is similar to that of a decision tree. The classifier at each stage depends on the training data passing through all the previous stages; hence deeper classifiers in the cascade face more difficult training samples. Experimental results on face detection demonstrate that a cascade of classifiers achieves similar performance compared to the monolithic classifiers, but the cascaded classifier is 10 times faster than the monolithic classifier.

The cascade structure has been shown to be effective in speeding up the object detection process. However, the fundamental problem of the cascade structure is that, the information obtained from each cascade stage will not be passed to next stage, and therefore historical information about the selected subwindows will not be used in subsequent stages. To address this problem, Bourdev and Brandt propose the soft cascade structure Bourdev & Brandt (2005), which is a generalization of the cascade structure. Their idea is to generalize each stage to be a scalar-valued decision function which is proportional to how well a given subwindow passes previous stages. As a result, the information obtained at early stages can be retained for later stages. Similarly, Dundar and Bi propose a new design of the cascade structure so that the classifiers at all the stages can be optimized for the final objective function Dundar & Bi (2007). As depicted in Figure 2.2, their cascade structure can optimize all the classifiers jointly using information from different stages.

Though the sliding window approach can be sped up using the cascade structures, its recognition can be affected by background clutters and occlusions. To alleviate the effect of background clutters and occlusions, local part based methods are proposed.

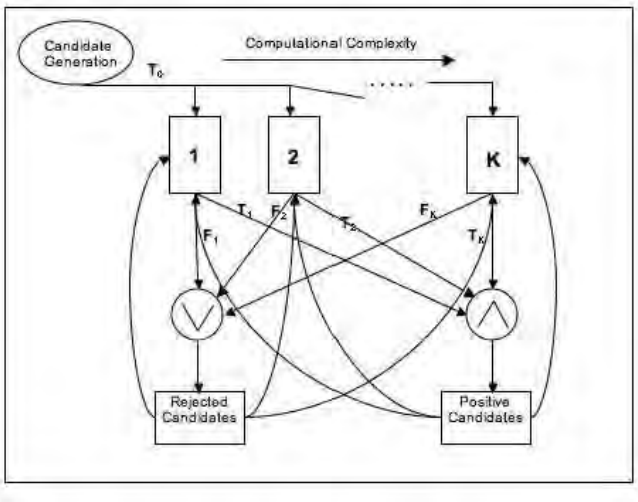


Figure 2.2: The joint cascade classification structure Dundar & Bi (2007). All the classifiers are jointly optimized.

2.2.1.2 Local Methods

Unlike global methods, local methods represent an object as a collection of parts. For example, parts of a human body, such as head, torso and legs, can be modeled separately. Part detectors can be constructed by learning feature statistics for different parts. The responses of part detectors are then integrated to form object hypotheses.

Numerous part based models have been proposed in the literature. Schneiderman and Kanade present an object detector that employs multiple classifiers for face detection in Schneiderman & Kanade (2004). In their approach, each classifier is based on the statistics of localized parts. Each part captures various combinations of locality in space, frequency and orientation. Similarly, Fischler and Elschlager introduce a pictural structure model for object recognition in Fischler & Elschlager (1973). Intuitively, it models an object as a collection of parts with connections between certain pairs of parts. Their pictural structure is quite general in the sense that different schemes can be employed to model the part connections. A natural way to model the part connections is an undirected graph based representation. On the basis of the undirected graph representa-

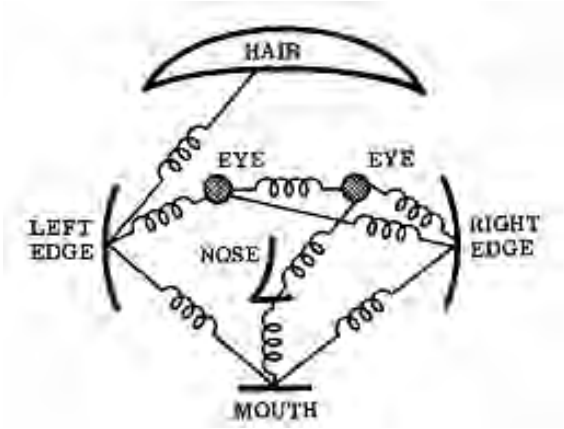


Figure 2.3: The pictural structure Fischler & Elschlager (1973) for a human face, where the springs stand for the connections between different parts.

tion, Fischler and Elschlager define the object localization problem as a energy minimization problem. However energy minimization problems are hard to solve efficiently, and it is often desirable to find more than a single (minimum energy) solution to object localization. To address these limitations, Felzenszwalb and Huttenlocher propose an efficient algorithm for solving the pictorial structure energy minimization problem in Felzenszwalb & Huttenlocher (2005), based on Viterbi algorithm Rabiner & Juang (1993). They employ statistical sampling techniques for locating multiple hypotheses for object localization. Experiments are conducted on the task of face and human detection in images.

Both the part based models in Schneiderman & Kanade (2004) and Fischler & Elschlager (1973) exploit the pairwise relationship between parts. Besides the pairwise relationship, the hierarchical relationship between parts is also useful for object modeling. Zhu and Yuille describe a hierarchical compositional model for deformable object detection in Zhu & Yuille (2006). Objects are represented by a hierarchical tree where the tree root corresponds to the complete object and the subtrees correspond to simpler features. Their hierarchical tree structure for the horse class can be found in Figure 2.4. This approach is tested on cat and horse detection in images. It is shown to perform well even in the presence of background clutter and occlusions. Despite of its good performance, the hierarchical model is inherently limited to only a moderate number of object classes.

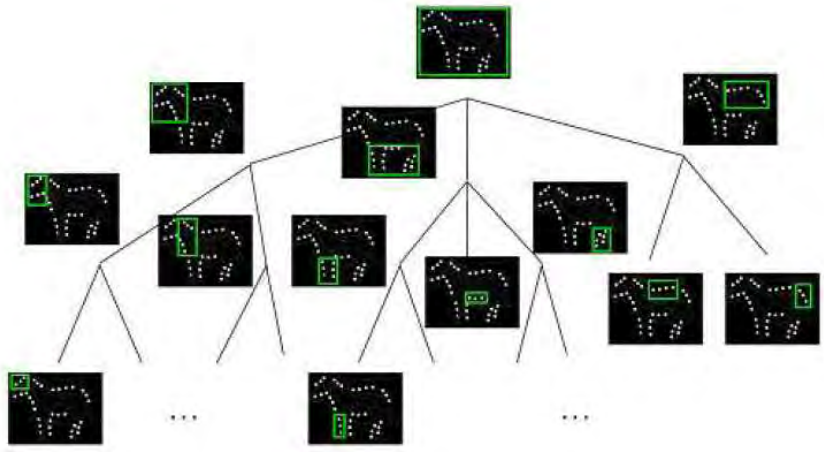
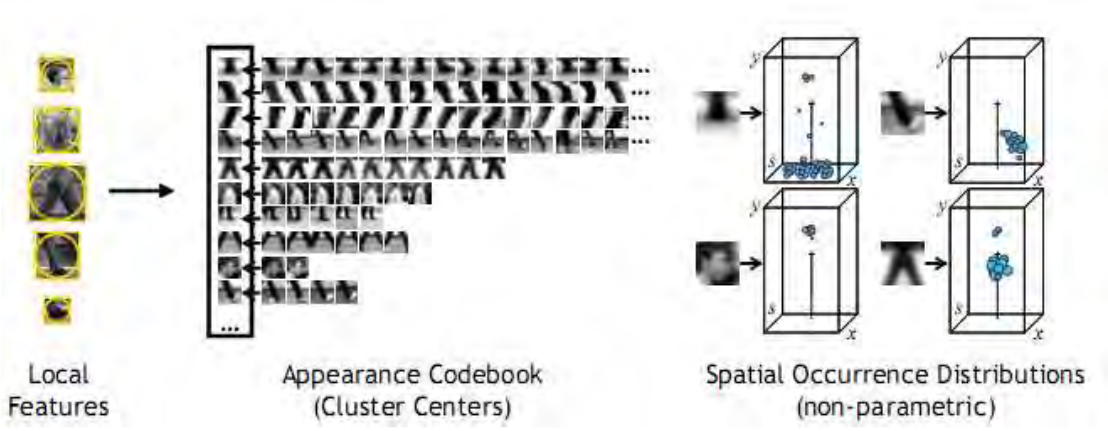


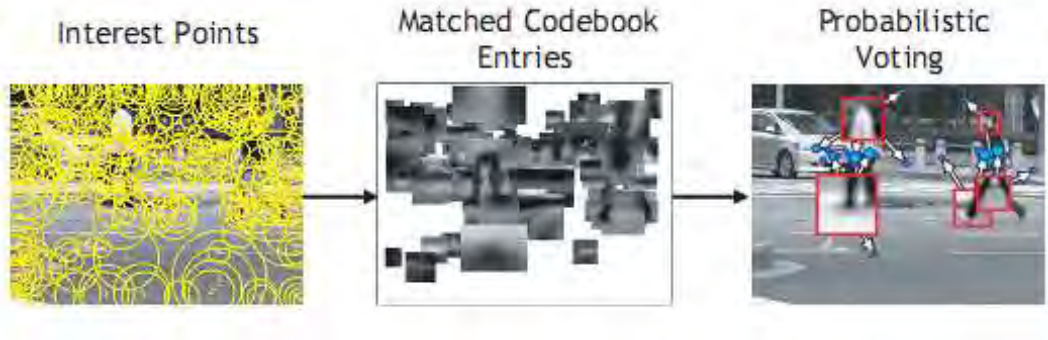
Figure 2.4: The hierarchical tree structure of a horse Zhu & Yuille (2006).

In order to pursuit a more general structural object model, Fidler *et al.* present a generic hierarchical object structure model in Fidler *et al.* (2006). By exploiting the statistical properties of the highly structured visual world, their model can overcome the computational complexity that encompasses a large number of different object categories.

The aforementioned models design specific parts for different objects, however there is another type of part based methods that possesses more generalization. They usually require a codebook of local features, which can be formed by applying clustering algorithms on extracted training features. A codebook consists of a set of codewords, each of which corresponds to a cluster center. Additionally, the spatial occurrence distribution with respect to the object center for each codeword is also recorded. The Generalized Hough Transform Ballard (1981) is employed in recognition. Each local feature from testing images are matched against the codewords, and valid matches cast probabilistic votes for object center hypotheses. These codebook based learning and recognition procedures are depicted in Figure 2.5. Leibe *et al.* present the Implicit Shape Model(ISM) for pedestrian detection in Leibe *et al.* (2005). The Implicit Shape Model, which is an instantiation of the codebook learning, captures object shape information based on salient image fragments. Under a similar framework, Opelt *et al.* develop



(a) The codebook learning procedure



(b) The codebook recognition procedure

Figure 2.5: The codebook learning and recognition procedure Leibe *et al.* (2005).

the Boundary-Fragment-Model (BFM) that detects objects using discriminative boundary fragments in Opelt *et al.* (2006).

Traditional codebook based methods cast weighted votes for possible object center hypotheses. These weights are usually learned in a generative framework, in the sense that only positive training samples are provided. Maji and Malik present a discriminative framework for the weight learning in Maji & Malik (2009), where both positive and negative training samples are required. They show that weights can be learned in a max-margin framework that directly optimizes the classification performance. Similarly, the codebook is generated using the k-means clustering. Using the codebook, the discriminative training takes into

account of both the codeword appearance and its spatial occurrence distribution with respect to the object center. Combined with a SVM based verification step, this max-margin codebook based method achieves significant improvement over the original codebook based method on car and horse detection. Though their method is based on discriminative training, it still relies on a generative codebook formed by the k-means clustering. Gall and Lempitsky propose a more discriminative approach for codebook learning in Gall & Lempitsky (2009). Their method employs a discriminative codebook, which is a random forest that maps image patches to weighted votes on object center hypotheses. Given both positive and negative training images, a class-specific random forest is learned for each class. For recognition, each class-specific random forest is applied to patches in testing images to cast votes for object center hypotheses. Similarly, Okada describes a discriminative method for codebook learning in Okada (2009). The Extremely Randomized Trees Geurts *et al.* (2006) are used as a discriminative codebook in his method. Both tree based codebook learning methods have been shown to outperform the traditional codebook based methods on some object detection tasks.

2.2.1.3 Hybrid Methods

Global methods focus on global appearance, but they can be affected by occlusion and background clutter. On the other hand, local part based methods perform well against occlusions, but they lack global structures. Global and part based methods are complements to each other. Hybrid methods are proposed to take advantages of both methods. For example, Felzenszwalb *et al.* presents a deformable part model for object detection in Felzenszwalb *et al.* (2008). The idea is to maintain both global and part models for recognition. Their model consists of a coarse global template covering the entire object and a set of higher resolution part templates. The templates are represented using the histogram of gradient (HOG) features Dalai *et al.* (2005). For recognition, a HOG feature pyramid is created by computing the HOG features of each level of a standard image pyramid. Object detection is based on the sliding window scanning, where the global template is used for matching at coarse scale while the part templates are used

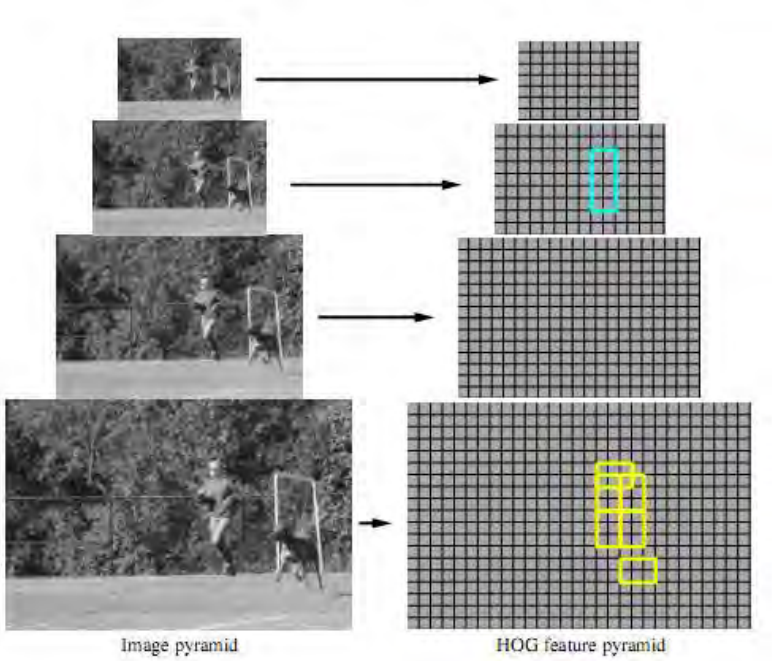


Figure 2.6: The HOG pyramid and the object hypotheses defined by the global template (near the top of pyramid) and the part-templates (near the bottom of the pyramid) Felzenszwalb *et al.* (2008).

for matching at finer scales. The hypotheses generation for pedestrian detection is shown in Figure 2.6. Their proposed method is demonstrated to perform well on rigid and highly deformable objects due to its hybrid nature.

Though the deformable part-model Felzenszwalb *et al.* (2008) is getting increasingly popular, its sliding window scanning based object detection is still computational expensive. To speed up object detection, a cascade object detection framework is proposed for the deformable part model in Felzenszwalb *et al.* (2010). The idea is to prune away partial hypotheses using thresholds on their scores. The notion of probably approximately admissible (PAA) thresholds are introduced to pick up thresholds that lead to low errors of hypotheses pruning. Experimental results show that the cascade framework can speed up object detection by more than an order of magnitudes without sacrificing the performance.

2.2.1.4 Motion based Methods

Motion based methods refer to the motion segmentation methods that detect moving objects in videos. Generally speaking, they assume that a compact region with different motion from the background is most likely to be a moving object. Motion based methods usually work by clustering moving pixels into layers with consistent motions. Khan and Shah propose a maximum a posterior (MAP) probability framework that uses multiple cues, such as spatial location, color and motion for segmentation in Khan & Shah (2001). In their framework weights are assigned to color and motion terms, which are adjusted per pixel. Wang and Ji presents a Dynamic Conditional Random Field (DCRF) model to integrate spatial and temporal constraints for object segmentation in Wang & Ji (2005). Their segmentation method employs both intensity and motion cues, and it combines dynamic information and spatial interaction of observed data. Similarly, Han *et al.* employ the Markov Random Field (MRF) model which integrates the color attributes and the spatial relationship between neighboring pixels for motion segmentation Han *et al.* (2006).

The aforementioned approaches rely on a persistent or slowly changing background, and cannot handle the background motions. Bugeau and Pérez propose a method to segment moving objects in dynamic scenes in Bugeau & Pérez (2007). In their method, grid points are grouped into clusters using the Mean Shift algorithm D. Comaniciu (2002). Object segmentations are achieved by associating objects to different clusters under a MAP-MRF framework. Experiment results show that their proposed method can handle camera motion and different background motions. On the other hand, existing object segmentation methods also rely on sufficient object motion, and they cannot work well with sparse and insufficient motions. Liu and Gleicher present an algorithm to learn object color and locality cues from the sparse motion information in Liu & Gleicher (2009). Their method works by estimating moving sub-objects using motion cues first. Color Gaussian Mixture Models are learned from the sub-objects as appearance models. The locations of these sub-objects are propagated to neighboring frames as locality cues. Finally object segmentation is achieved by combining the learned color and locality cues with motions cues in a MRF framework. Experiments on

videos with a variety of objects and camera motions demonstrate the effectiveness of their algorithm.

2.2.1.5 Context based Methods

The aforementioned approaches take advantages of appearance and motion cues for object recognition. However they ignore the contextual cues, which plays an important role in the biological vision. Psychophysics studies have shown that the analysis of contextual relationships between visual concepts play an important role in human vision Biederman *et al.* (1982). As a result, detection of a concept of interest can be facilitated by the presence of other concepts which might not even be of interest. This has led to efforts to account for context based recognition.

Various types of contextual cues have been investigated to improve object recognition tasks. Most are local context that considers information from the vicinities of objects, such as pixels, regions and objects. Pixel context captures the low-level feature interactions between spatially adjacent objects Shotton *et al.* (2008). Region context captures the interactions between regions surround objects Galleguillos *et al.* (2008). Finally object context describes the information between objects Parikh *et al.* (2008). Quite often, labels are assigned to different type of neighborhoods, and their label agreements are maximized based on the contextual information. In addition to using single context cue, Galleguillos *et al*. develop a multiple kernel learning algorithm to combine multiple context cues. Pixel, region and object context cues are integrated to form a unified similarity metric which is optimized for nearest neighbor classification Galleguillos *et al.* (2010). Similarly, Choi *et al*. present a context model that exploits the hierarchical context information from over 100 object categories in Choi *et al.* (2010). The empirical study of context in object detection Divvala *et al.* (2009) shows those frameworks that incorporate contextual information can be more effective for object detection tasks.

2.2.2 Scene Recognition

Scene recognition is the task of classifying scene images into semantic categories. It is not an easy task due to several factors, such as appearance variations, viewpoint variations, and illumination changes. Depending on the image representations, the work in the literature can be divided into two categories. The first category relies on low-level image representations, while the second one relies on intermediate-level image representations.

2.2.2.1 Low-level Image Modeling

This approach assumes that scene images can be discriminated based on the color/texture properties. For instance, a beach scene usually presents an important amount of blue (sea) and white (sky) color, whereas the presence of highly textured areas (trees) denotes a forest scene. Specifically, low-level image features (e.g. color/texture) are used to describe images. Different classifiers are employed to map low-level representations to high-level semantic labels. Different representations are reviewed in the following sections.

Global representations Global representations use low-level features computed from the whole image. Based on the global representation, Vailaya *et al.* demonstrate that low-level features can successfully discriminate different scene images in Vailaya *et al.* (1998). They investigate color histogram, color coherence vector, DCT coefficient, edge direction histogram, and edge coherence vector for global representations. A weighted k-NN classifier is employed for the classification. Experimental results show that edge-direction based features have the most discrimination power for scene recognition. It achieves an accuracy of 93.9% in an database containing 2,716 images.

Sub-region based representations Sub-region based representation uses low-level features extracted from sub-regions of images. Each image is firstly split into multiple sub-regions, from which features are extracted. Each sub-region is classified by classifiers, and the whole image is categorized based on the individual classification results for each sub-region.

Such representation stems from the work of Szummer and Picard in Szummer & Picard (1998), whose goal is to classify indoor and outdoor images. Each image is partitioned into 16 sub-regions, and then Ohta-space color histograms and MSAR texture features are extracted from each sub-region. The k-NN classifier is employed to classify each sub-regions using a histogram intersection measure, which computes the amount of overlap between corresponding bins in two given N-dimensional histograms X and Y:

$$dist(X,Y) = \sum_{i=1}^{N}(X(i) - min(X(i), Y(i))). \tag{2.1}$$

Finally image classification is achieved by the majority voting of the sub-region classification results. A performance of 90.3% is achieved for indoor/outdoor image classificaiton. A similar approach can be found in the work of Serrano *et al.* Serrano *et al.* (2004), where color and texture features are also extracted from sub-regions. However each sub-region is separately classified by the SVM. In comparison with the k-NN classifier, the SVM generates numerical confidence measure for each sub-region which can be combined numerically for final classification.

Despite of the good performance of low-level image representation, such representation ignores higher-level image information which could be very valuable in determining scene types. For example, semantic concepts such as sky, water, and snow, can be very helpful to the semantic understanding of scene images. As a result, researchers direct their attention on exploiting the semantic concepts.

2.2.2.2 Semantic Image Modeling

Low-level image features have been shown to be useful in discriminating images from different scenes. However Serrano *et al.* show that semantic concepts such as the presence of sky, sea, etc. can be used to improve scene classification performance obtained by using low-level features alone Serrano *et al.* (2004). In comparison with the low-level features, semantic concepts provide higher-level semantic understanding of images.

Related work in the literature can be grouped into the following four types. The first type uses semantic objects to represent the semantic information, in the

sense that each image is described by the occurrences of the semantic objects. Object detection is an indispensable step in this method. The second type employs more general intermediate representations to avoid object detection. Specifically, each image is described using a distribution of visual codewords, which is produced during training time. The third type describes different scenes using semantic properties, which encode the spatial structure of the scene. The last type describes each scene using scene prototypes which are defined by several parts under mutual geometric constraints. Each part corresponds to a region of interest. Each type of approach is reviewed in the following sections.

Semantic Objects In this method, scene information is collected from meaningful semantic concept objects, which are discovered by object detection algorithms. Different strategies have been proposed for detecting the semantic objects.

Fan *et al.* employ concept-sensitive salient objects to annotate images at the content level Fan *et al.* (2005). The concept-sensitive salient objects are detected using a set of detection functions, which are learned from the labeled image regions by using the SVM classifiers. For scene recognition, the finite mixture models are used to model the class distributions of relevant semantic objects. Specifically, a mixture model $P(X, C_j|\kappa, \omega_{c_j}, \theta_{c_j})$ can be learned for each semantic category C_j, where $\theta = \{\kappa, \omega_{c_j}, \theta_{c_j}, j = 1, ..., N_c\}$ is the set of the mixture parameters for N_c semantic scene categories. Given a test image I and the detected concept-sensitive salient objects $\{S_1, S_2, ..., S_n\}$, we can compute the posterior probability:

$$p(C_j|X, I, \theta) = \frac{p(X, C_j|\kappa, \omega_{c_j}, \theta_{c_j}, p(C_j))}{\sum_{i=1}^{N_c} p(X, C_i|\kappa, \omega_{c_i}, \theta_{c_i}, p(C_i))}, \tag{2.2}$$

where X corresponds to the visual features used for representing the salient objects, and $P(C_j)$ corresponds to the prior probability of scene category C_j in the database. The test image I is assigned one scene category based on the maximum posterior probability.

Although it is impossible to design detectors for all the objects in a scene, Luo *et al.* argue that the use of a small number of semantic object detectors can still achieve decent performance, if complemented with low-level features Luo

et al. (2005). They propose a framework in Luo *et al.* (2005), where semantic objects and low-level features are combined to form a rich description of scenes. The proposed framework integrates low-level and intermediate-level semantic information. The integration is achieved by a Bayesian Network (BN), which is a directed acyclic graph that encodes the dependence relationship between different nodes (variables). Using the BN, the joint probability distribution of variables can be efficiently computed for scene classification. Their method is quantitatively evaluated using low-level features (Ohta color space histograms and MSAR texture features), together with semantic features (sky and grass objects). Experimental results show that recognition performance can be improved with incorporated semantic features.

A Bayesian framework is also proposed in Aksoy *et al.* (2003), where semantic objects are prototype regions obtained by automatic image segmentations. A visual grammar is proposed to model the interactions between the prototype regions based on their spatial relationships. The visual grammar can help to distinguish two scenes with similar regions but different spatial arrangements. Experimental results show that the visual grammar bridges the gap between semantic objects and user semantics, by creating higher level representations that cannot be described by regions. Another image segmentation based approach is proposed in Fredembach *et al.* (2004), where eigenregions that encompass area, location and shape properties of image regions are used as semantic objects. Eigenregions are demonstrated to improve image classification performance.

The latest development of this approach focuses on multi-scale image processing. Tanaka *et al.* propose a multi-level resolution semantic concept representation for scene recognition in Tanaka *et al.* (2010). In their approach, an image is segmented into different level of resolutions, and a semantic model is constructed in each level to form the final global representation. The sizes of local regions are dynamically adjusted for different scenes. Experimental results demonstrate that their method can achieve significant improvement compared to single-level semantic concept modeling.

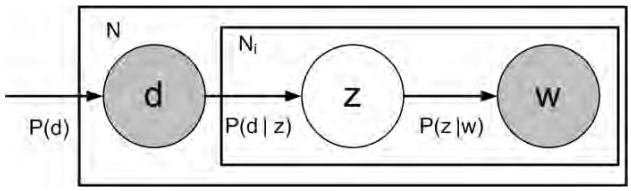
Semantic Concepts In comparison with the methods using semantic objects, this type of methods provides a more general representation. They describe each

image based on the Bag-of-Words representation. The Bag-of-Words model Sivic *et al.* (2005) originates from the text mining community, where each document is represented by a frequency vector of words. The counterpart for the "word" in the model is the "codeword" in images. To form the codewords for images, clustering algorithms are employed to cluster extracted local feature into clusters, whose centers are considered as the codewords. The collection of codewords are usually referred to as the codebook. On the basis of the codebook, each local feature is quantized to the most similar codeword using similarity measure.

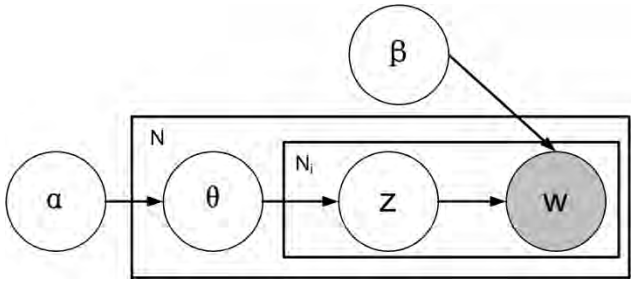
Using the Bag-of-Words representation, some Bayesian models from text mining area have been successfully applied to scene recognition. For example, the Probabilistic Latent Semantic Analysis (PLSA) Hofmann (2001), and the Latent Dirichlet Analysis (LDA) M. Blei *et al.* (2003) have been used for modeling scene categories. Both the PLSA and the LDA employ latent topic variables for the intermediate representations of scene images. Given a set of images $D = \{d_1, d_2, ..., d_N\}$, a codebook $W = \{w_1, w_2, ..., w_V\}$ formed by running the k-means clustering algorithm on the extracted features from D, the PLSA associates d_i and w_j via a hidden topic z_k; the joint probability of $p(d_i, z_k, w_j)$ is assumed to have the following form:

$$p(d_i, z_k, w_j) = p(w_j|z_k)p(z_k|d_i)p(d_i), \tag{2.3}$$

where $p(w_j|z_k)$ represents the probability of the codeword w_j occurring in the hidden topic z_k, $p(z_k|d_i)$ is the probability of the topic z_k occurring in d_i, and $p(d_i)$ can be considered as the prior probability of d_i. The graphical representation of the PLSA is shown in Figure 2.7(a), where N is the number of images and N_i is the number of visual feature descriptors in d_i. Similar to the PLSA, the LDA also treats each image as a mixture of various topics, where each topic is characterized by a distribution over visual words. In contrast to the PLSA, the LDA models the topic distributions $P(z|d)$ as latent random variables with Dirichlet priors. The Dirichlet prior allows the LDA to assign probabilities to unseen data with fewer parameters, and thus overfitting can be reduced. It is noted that, the PLSA model is equivalent to the LDA model under a uniform Dirichlet prior distribution Girolami & Kabán (2003). As shown in Figure 2.7(b), the LDA has two hyper parameters α and β, where α is the parameter of the Dirichlet prior for the topic



(a) The PLSA model



(b) The LDA model

Figure 2.7: The PLSA Hofmann (2001) and LDA model M. Blei *et al.* (2003). Filled circles indicate observed random variables and the unfilled are unobserved. N is the number of images and N_i is the number of local features in d_i.

distribution θ, and β is the parameter for the multinomial word distribution $P(w|z,\beta)$. Given the hyper parameters α and β, the joint distribution on topics $\mathbf{z}$ and words $\mathbf{w}$ in d_i can be written as:

$$p(\theta_i, \mathbf{z}, \mathbf{w}|\alpha, \beta) = p(\theta_i|\alpha) \prod_{j=1}^{N_i} p(z_j|\theta_i) p(w_j|z_j, \beta), \tag{2.4}$$

where N_i is the number of visual feature descriptors in d_i, $p(\theta_i|\alpha)$ is a Dirichlet distribution characterized by α, and $p(w_j|z_j,\beta)$ is a multinomial distribution conditioned on z_j, characterized by β.

Quelhas *et al.* provide an approach that combines the PLSA and supervised classification for scene recognition in Quelhas *et al.* (2005). In their approach, each image is represented by an occurrence vector of codewords based on the Bag-of-Words representation. The occurrence vector is then fed into the PLSA to generate a more compact representation, which is used for scene recognition with the SVM. Experimental results demonstrate that the PLSA based representation is not only discriminative for accurate classification, but also significantly

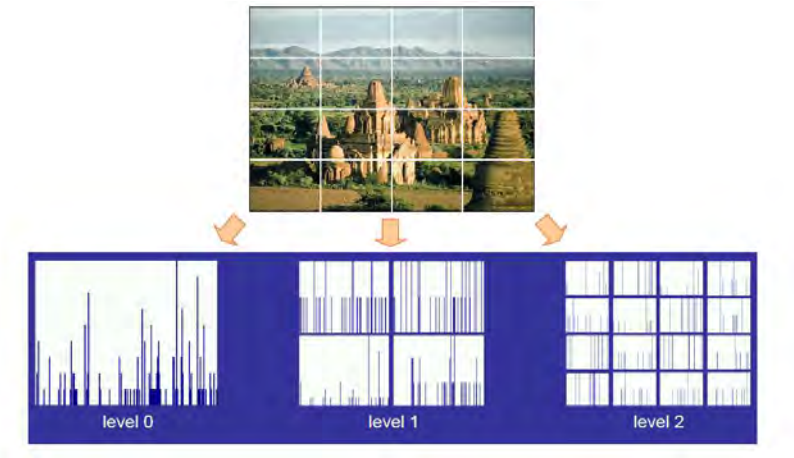


Figure 2.8: The Spatial Pyramid representation Lazebnik *et al.* (2006).

more robust when less training data is available. Similarly, Bosch *et al.* combines the PLSA based representation and a k-NN classifier for scene recognition in Bosch *et al.* (2006). The difference between these methods are as follows: (i) the number of scenes : 3 in Quelhas *et al.* (2005), while 13 in Bosch *et al.* (2006); (ii) the sampling of features : in Quelhas *et al.* (2005) feature descriptors are computed on sparse interest points, whereas in Bosch *et al.* (2006) descriptors are computed from dense regular grids. Importantly, dense sampling is demonstrated to outperform sparse sampling on natural image classification in Bosch *et al.* (2006). Fei-Fei and Perona Fei-Fei & Perona (2005) propose two variations of the LDA for unsupervised scene recognition. In their framework, scene categories are represented by intermediate-level "themes", which are learned from given local regions automatically. Experiments are conducted on a dataset containing 13 scene categories. Although the Bag-of-Words based methods have been shown to be effective, they neglect the contextual information such as the spatial structural information between local features, which could be useful for scene recognition. One remedy is to augment the basic Bag-of-Words framework with pairwise relationship between neighboring features. Sivic *et al.* extend the Bag-of-Words representation to include "doublets" which encode the spatially co-occurrence relationship between local features in Sivic *et al.* (2005). The extension is demonstrated to give a cleaner image segmentation, however no classification

results are reported. Lazebnik *et al.* propose a simple and computational efficient approach called "Spatial Pyramid" which is based on aggregating statistics of local features over fixed subregions in Lazebnik *et al.* (2006). As illustrated in Figure 2.8, their method works by dividing an image into increasingly fine sub-regions, inside which histograms of local features are computed. In this way, the spatial information is incorporated into the orderless Bag-of-Words representation. Finally scene recognition is achieved by the SVM using the pyramid matching kernel:

$$\kappa^L(X,Y) = \frac{1}{2^L} I(H_X^{(0)}, H_Y^{(0)}) + \sum_{l=1}^{L} \frac{1}{2^{L-l+1}} I(H_X^{(l)}, H_Y^{(l)}), \tag{2.5}$$

where X and Y are two sets of d-dimensional vectors, L is the maximum level of grid resolution, $H_X^{(l)}$ and $H_Y^{(l)}$ are the histograms of X and Y at level l, and $I(H_X^{(l)}, H_Y^{(l)})$ measures the histogram intersection Swain & Ballard (1991) between $H_X^{(l)}$ and $H_Y^{(l)}$. This method achieves high accuracy (81.4%) on a dataset of 15 natural scene categories.

Similar to the spatial pyramid, Lu and Ip propose the spatial mismatch kernel (SMK) for use with the SVM to categorize images Lu & Ip (2009). Their proposed kernel can capture the spatial structures of images. In their method, images are divided into equivalent blocks on a regular grid. Such blocks are used to generate 2D sequences. By decomposing each 2D sequence into two parallel 1D sequences (i.e. the row-wise and column-wise ones), the spatial mismatch kernel can measure the similarity of the 2D sequences. Experiments on the natural image databases demonstrate that the proposed method outperforms other methods using the Bag-of-Words framework and the SVM.

Semantic Properties The third type of semantic modeling exploits the semantic properties of scenes. It concentrates on holistic descriptions of scene structures. As a result, neither segmentation nor feature extraction is required. One example is the "spatial envelop" proposed by Oliva and Torralba in Oliva & Torralba (2001). A spatial envelop is a very low dimensional representation of the scene, and it consists of a set of perceptual dimensions, including naturalness, openness, roughness, expansion, and ruggedness. These perceptual dimensions

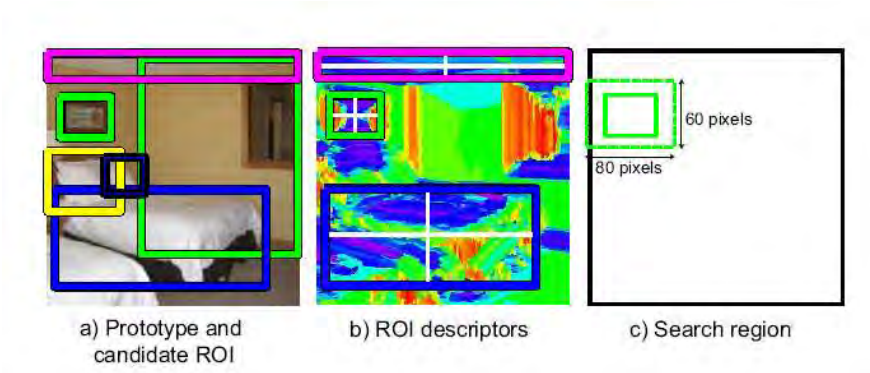


Figure 2.9: Example of a scene prototype Quattoni & Torralba (2009). (a) Scene prototypes with candidate ROI. (b) Illustration of the visual words and the regions used to compute histograms. (c) Search window to detect the ROI in a new image.

represent the dominant spatial structure of a scene, and they can be reliably estimated using spectral and coarsely localized information. The spatial envelop provides a meaningful representation of complex environmental scenes, and it also provides an efficient way for context modeling.

Semantic Prototypes Finally the last type of semantic modeling describes each scene using prototypes. Quattoni and Torralba propose the scene prototypes for indoor scene recognition in Quattoni & Torralba (2009). Each scene prototype is defined by parts under mutual geometric constraints, where each part corresponds to a region of interest (ROI). Inside each prototype, each part is allowed to move on a small window and their displacements are independent of each other. Each ROI is represented by a spatial pyramid of visual words. Figure 2.9 shows an example of the scene prototypes. The idea of using scene prototypes to model indoor scenes is due to the fact that some indoor scenes can be well characterized using global spatial properties, while others are better characterized by the objects they contain. The scene prototype can exploit both local and global discriminative information. Experiments show that better performance of indoor scene recognition can be achieved by combining local and global information.

2.2.3 Action Recognition

Human action recognition from video sequences has a wide range of applications, such as human computer interaction, surveillance and security, and sports analysis. The term "action" and "activity" have been used interchangeably in the literature. Generally speaking, the word "action" represents simple motion patterns which last for a short duration of time, while the word "activity" refers to the complex sequences of motion patterns which usually have longer temporal durations. However, there is no strict difference between them, and therefore we use the word "action" to refer to both motion patterns in this thesis. Numerous techniques have been proposed for action recognition in the literature, and they are reviewed in the following sections.

2.2.3.1 Motion based Methods

Motion is used as a strong cue for action recognition, as retinal image motion plays an important role in our visual motion perception experience Johansson (1975).

Initially motion characteristics are studied for action recognition. For example, Cutler and Davis employ time-frequency analysis to detect and characterize periodic motions for simple event detection Cutler & Davis (2000). Little and Boyd study the periodic structure of optical flows for gait recognition Little & Boyd (1998).

In addition to the motion characteristics, the spatial-temporal behaviors of the motion in video sequences are also studied. Davis and Bobick propose the "temporal templates" to capture both the shape cue of human movements in the space-time domain Davis & Bobick (1997). The temporal template is essentially a static vector-image where the vector value at each point keeps the motion information at the corresponding spatial location in the sequence of movements. Shechtman and Irani propose a space-time behavior based measure to detect the similarity between different actions Shechtman & Irani (2005). Their proposed similarity measure extends the notion of 2-dimensional image correlation to the 3-dimensional space-time volume, and it enables the correlation between dynamic behavior and actions. Similarly, human actions are interpreted as 3-dimensional

shapes induced by the silhouettes in the space-time volume in Gorelick *et al.* (2007).

2.2.3.2 Temporal Model based Methods

The motion based methods are shown to perform well on simple action recognition. However sophisticated models are needed in order to cope with complex actions.

The Hidden Markov Model (HMM) Juang & Rabiner (1991) is employed by early researchers to describe complex actions. The HMM models can deal with time-sequential data, and they exhibit time-scale invariability as well as learning capability. Yamato *et al.* use a HMM model to recognize human actions from videos Yamato *et al.* (1992). As a precessing step, each set of sequential images is converted into a symbol sequence by vector quantization of the extracted image features. A HMM is fitted to each category of human actions. Action recognition is achieved by determining the best matched HMM model. Similarly, Feng and Perona Feng & Perona (2002) construct the HMM models from vector-quantized movelets, which capture the motion from the main body parts. In addition to the traditional HMM model, several extensions are proposed for the analysis of human behaviors. For example, a layered HMM is proposed to represent office activities in Oliver *et al.* (2004). In comparison with the single-layer structure, multiple-layer structure enables multiple-layer analysis.

Although the HMM models appear to be robust against temporal segmentation of observations, they suffer from excessive parameters when applied to reasoning about long and complex human activities with insufficient training data. As a result, complex Bayesian models with fewer parameters are proposed to model human activities. Recently, Xiang and Gong Oliver *et al.* (2004) employ dynamic Bayesian networks to model complex activities of multiple objects in cluttered scenes. Specifically, an activity of multiple objects is represented using scene events, and their behaviors are analyzed using the correlations among different events. Oliver *et al.* propose a dynamic Bayesian network that integrates temporal, contextual, and ordering constraints with low-level visual detection results to recognize complex and long-term activities Oliver *et al.* (2004).

Recently the practice that describes temporal dependencies using causality has attracted the attention of researchers. It is natural to use causality to describe the dependencies. Some attempts have been made to develop causality models for video analysis. Prabhakar *et al.* present a data-driven approach to analyze causality in Prabhakar *et al.* (2010). There are two stages in their approach. The first stage encodes a sequence as a multivariate point-process over visual events, which are the occurrences of visual words. The second stage groups co-occurring codewords into independent causal sets by analyzing causal relationship between point-processes. The causal sets usually correspond to salient groupings of events. Experiment results demonstrate a significant improvement in performance across videos for social games and quasi-periodic events by their method. Similarly, Ni *et al.* propose to encode group-activities using three types of causalities, namely self-causality, pair-causality, and group-causality in Ni *et al.* (2009). They characterize the local interactions within, between, and among trajectories of different humans respectively. Experiment results show the promising results on group activity recognition using local causalities.

2.2.3.3 Interest Point based Methods

Interest point based action recognition is getting increasing popular due to its simplicity and effectiveness. This method represents video sequences using local interest points. Given a video sequence, spatio-temporal interest points are detected and robust feature descriptors are used to describe the interest points. The feature descriptors are usually vector-quantized into a set of codewords to form the video representation. The obtained representations are then fed into classifiers for recognition.

Numerous spatio-temporal features are proposed and combined with different classifiers for recognition. Laptev and Lindeberg extend the Harris interest points in the spatial domain to spatio-temporal domain Laptev & Lindeberg (2003). Their proposed detector can locate local structures that possess large variations in both space and time. Combined with this space-time features, Schuldt *et al.* train a SVM to classify human actions. Dollár *et al.* propose a spatio-temporal interest point detector based on a set of linear filters Dollar *et al.* (2005a), and

use these features with k-NN classifier for human action recognition. Ke *et al.* combine the spatio-temporal volumetric features with a cascade of classifiers for event detection Ke *et al.* (2005).

Given the spatio-temporal features, k-means clustering algorithm is usually employed to produce a set of codewords. Typically, the k-means algorithm groups features based on their similarity with a predefined cluster number. Liu *et al.* propose an approach that is able to automatically discover the optimal number of codewords by the Maximization of Mutual Information (MMI). Specifically, the MMI clustering further compresses the codewords generated by the k-means clustering to produce fewer but more meaningful codewords. Experimental results show that the codebook produced by the MMI clustering achieves much better performance than the k-means based codebook for action recognition Liu & Shah (2008).

All the aforementioned methods use histogram based representations, which discard the temporal ordering information, This ordering information, however, could contain important information about the action itself. For example, both disciplines of hurdle race and long jump consist of motion features for "running" and "jumping". To discriminate these two disciplines, the temporal ordering information is very important. To exploit such ordering, Nowozin *et al.* propose a sequential representation which retains this temporal order in Nowozin *et al.* (2007). For each interest point found, their method keeps the spatio-temporal coordinates as well as the descriptor. The descriptors are clustered to produce a codebook of prototypes. Using the codebook, each video is represented as a set of words of the form $(x; y; t; w)$, where $(x; y; t)$ is the spatio-temporal coordinates and w is the codebook index.

In addition to exploiting the temporal ordering information, Bregonzio *et al.* propose a method that exploits the global spatio-temporal distribution of interest points Bregonzio *et al.* (2009). Their idea is to extract holistic features from clouds of interest points accumulated over multiple temporal scales followed by an automatic spatio-temporal feature selection. A feature selection method measures the relevance of each feature according to how much its value varies within each action class and across different classes. Specifically, a feature is considered informative if its value has little variations for actions of the same

class but significant variations for actions of different classes. Experiment results demonstrate that the explicit modeling of global spatial and temporal distribution of interest points is highly discriminative and more robust for recognizing actions under occlusion.

2.2.3.4 Topic Model based Methods

Driven by the success of latent topic models on scene recognition, researchers have also applied topic models to the task of action recognition. Following the Bag-of-Words paradigm, spatio-temporal features are extracted from video sequences, and quantized into a set of codewords. Similar to the interest point based method, each sequence is then represented as a histogram of codeword occurrences. The obtained histograms are fed into the topic models for action recognition. For example, Bissacco *et al.* apply the Latent Dirichlet Allocation (LDA) to detect humans and estimate poses from single images in Bissacco *et al.* (2007). The LDA is used to model the distinctive intermediate themes for human poses. The codewords in their approach are generated from the histogram of oriented gradient features. Niebles *et al.* demonstrate the unsupervised learning of human action categories using the PLSA and the LDA in Niebles *et al.* (2008). The features used in their method are the spatio-temporal interest points detected in video sequences. Although the PLSA and the LDA can discover different topics correspond to different actions, they fail to take into consideration of the contextual relationship between features. To model such information, Wong *et al.* propose an extension to the PLSA for capturing both semantic (the content of interest points) and structural (geometric information between interest points) information for action recognition and localization Wong *et al.* (2007). Recently Wang and Mori propose two new topic models for human action recognition in Wang & Mori (2009). Different from the PLSA and the LDA, the latent topics in their models correspond to class labels, and therefore some of the latent variables become observable in their models. In comparison with the PLSA and the LDA, the training of their model is much easier due to the decoupling of model parameters. Furthermore, their models alleviate the problem of setting appropriate

number of latent topics. Action recognition results demonstrate that their models achieves comparable or significantly better results than the existing models.

Chapter 3

On-line Learning for Codebook based Visual Recognition

3.1 Introduction

As our first technical chapter, we look at the codebook based visual recognition task. Visual recognition is the task of assigning images/videos into different semantic categories. Codebook based visual recognition is based on the Bag-of-Words representation, which represents a document as a bag of words. In text mining area, statistical models have been developed based on the Bag-of-Words representation, and they have achieved notable success on document classification. These models have been successfully applied to visual recognition by computer vision researchers. We investigate one of these models, and propose an extension for its on-line learning.

The model we investigate is the Probabilistic Latent Semantic Analysis (PLSA) Hofmann (2001). The PLSA, which is a hierarchical graphical model, associates documents and words via latent "topic" variables. In the PLSA, each document is represented as a mixture model of "topics", and words in the document are considered as samples from the mixture components. Using these mixture components, a document can be reduced to a probability distribution of a fixed set of topics, which can be considered as a "reduced" description for the document.

Similarly, the Latent Dirichlet Allocation (LDA) M. Blei *et al.* (2003) also views each document as a mixture of topics. However the LDA assumes that each topic variable has a Dirichlet prior distribution, which could result in more reasonable mixtures of topics for a document. In comparison with the LDA, the PLSA is a simpler, and it is equivalent to the LDA under a uniform Dirichlet prior distribution Girolami & Kabán (2003). As a result, we choose the PLSA as the subject of our investigation.

The PLSA has been successfully employed on tasks of object categorization Sivic *et al.* (2005), scene recognition Bosch *et al.* (2008), and also action recognition Carlos Niebles *et al.* (2008), Savarese *et al.* (2008). However, the PLSA has the following disadvantages. Firstly, the conventional learning for the PLSA is batch (off-line) learning, and hence it cannot handle data that arrives sequentially or data that has excessive memory requirements. Secondly, the PLSA relies on a codebook of words for the Bag-of-Words representation. For visual recognition, the k-means clustering algorithm is usually employed to produce a codebook of visual words Sivic *et al.* (2005), Carlos Niebles *et al.* (2008), Bosch *et al.* (2008). An optimal codebook can be constructed by taking into account of all training features. However, it would be infeasible to apply the same algorithm on large datasets with excessive features. In practice, only a subset of the features is chosen for the clustering, and it results in a suboptimal codebook. In order to address the aforementioned problems, we propose an on-line learning algorithm for learning the PLSA from sequentially arriving data. Our contributions are two-fold: (i) an on-line learning algorithm that learns parameters of the PLSA model from sequentially incoming data; (ii) a codebook adaptation algorithm that can capture the full characteristics of all features during the on-line learning.

We apply the proposed algorithms to two visual recognition tasks, namely, scene and action recognition. The task of scene recognition is the categorization of scene images. Most work in the literature can be classified into one of the two following approaches. The first approach considers low-level feature based image representations Vailaya *et al.* (2001), Wu & Rehg (2008), Gupta *et al.* (2009), whereas the second one relies on intermediate semantic representations Vogel & Schiele (2004), Fei-Fei & Perona (2005), Bosch *et al.* (2008), Quattoni & Torralb (2009), Li *et al.* (2009). Among the work from the second category, some models

learn semantic concepts in a supervised manner Vogel & Schiele (2004), Quattoni & Torralb (2009), while the others learn without supervision Fei-Fei & Perona (2005), Bosch *et al.* (2008), Li *et al.* (2009). The PLSA based scene recognition belongs to the second category, and it has been shown by Bosch *et al.*, Bosch *et al.* (2008) to outperform previous models Vogel & Schiele (2004), Fei-Fei & Perona (2005). Action recognition is an important task in video surveillance. Numerous techniques have been proposed for action recognition in the literature. Early researchers study the motion characteristics for action recognition Cutler & Davis (2000), Little & Boyd (1998). The spatial-temporal motion behavior is also studied Davis & Bobick (1997), Shechtman & Irani (2005). These motion based methods are shown to perform well on simple action recognition. However temporal models are needed in order to cope with complex actions. Different models are proposed, such as the Hidden Markov Model (HMM) Oliver *et al.* (2004), the dynamic Bayesian networks Oliver *et al.* (2004), and the causality models Ni *et al.* (2009), Prabhakar *et al.* (2010). In addition to the temporal models, action recognition can also be realized using the interest point based methods. In these methods, salient features are extracted from video sequences, which are then clustered into codewords. By quantizing features into codewords, each sequence is represented as a histogram of codeword occurrences which is fed into classifiers for recognition. Numerous spatio-temporal features are proposed and combined with different classifiers for recognition Laptev & Lindeberg (2003), Ke *et al.* (2005), Bregonzio *et al.* (2009). Similar to these methods, topic model based methods also use the histogram representation for videos. However the histograms are fed into topic models, such as the LDA, the PLSA, etc. The PLSA has been employed for action recognition using spatial-temporal interest points Dollar *et al.* (2005b), and it has been shown to perform well on this task Carlos Niebles *et al.* (2008). The experimental results of our proposed algorithm show that the proposed algorithm can handle sequentially arriving datasets that the batch PLSA learning cannot cope with. They also show that the performance of our proposed method is comparable with that of the batch PLSA learning on visual recognition.

The rest of this chapter is organized as follows: Section 3.2 reviews the batch learning algorithm for the PLSA. Our proposed on-line PLSA is explained in

Section 3.3. The experimental results are presented in Section 3.4. Finally our conclusions are summarized in Section 3.5.

3.2 Batch PLSA

The PLSA was originally proposed for text modeling, in which words are considered as the elementary building blocks for documents. Rather than modeling documents as sets of words, the PLSA models each document as a mixture of latent topics, and each topic is characterized by a distribution over words. In the context of visual recognition, visual features are considered as words. It is important to note that the PLSA actually allows us to explicitly represent each document using intermediate semantic concepts (latent topics). Since the number of latent topics is usually much smaller than that of visual words, such modeling would result in a compact and low-dimensional representation for documents.

Denote $z_k \in Z = \{z_1, z_2, ..., z_K\}$ as a latent topic variable that associates words and documents, and X as the training co-occurrence matrix consists of co-occurrence counting of (d_i, w_j), collected from N visual documents $D = \{d_1, d_2, ..., d_N\}$, based on a codebook of V visual words $W = \{w_1, w_2, ..., w_V\}$. The joint probability of the visual document d_i, the hidden topic z_k and the visual word w_j is assumed to have the following form Hofmann (2001):

$$p(d_i, z_k, w_j) = p(w_j|z_k)p(z_k|d_i)p(d_i), \tag{3.1}$$

where $p(w_j|z_k)$ represents the probability of the word w_j occurring in the hidden topic z_k, $p(z_k|d_i)$ is the probability of the topic z_k occurring in the document d_i, and $p(d_i)$ can be considered as the prior probability of d_i. The joint probability of observation pair (d_i, w_j) can be generated by marginalizing over all the topic variables z_k:

$$p(d_i, w_j) = p(d_i) \sum_k p(z_k|d_i)p(w_j|z_k). \tag{3.2}$$

Let $n(d_i, w_j)$ be the number of occurrences of the word w_j in the document d_i, and all of them constitute the co-occurrence matrix X. The prior probability $p(d_i)$ can be computed as $p(d_i) \propto \sum_j n(d_i, w_j)$. The parameters of the PLSA then contain

multinomial distributions over K latent variables, and such that $\sum_{j=1}^{V} p(w_j|z_k) = 1$ and $\sum_{k=1}^{K} p(z_k|d_i) = 1$. We represent these $KV + KN$ probabilities using $\theta = \{\{p(w_j|z_k)\}_{j,k}, \{p(z_k|d_i)\}_{k,i}\}$. The log likelihood of X given θ can be written as

$$L(X|\theta) = \sum_{i=1}^{N}\sum_{j=1}^{V} n(d_i, w_j) \log p(d_i, w_j). \tag{3.3}$$

A maximum likelihood estimation of θ is obtained by maximizing the log likelihood using an Expectation Maximization (EM) algorithm Dempster *et al.* (1977). Starting from an initial estimate of θ, the expectation (E) step of the algorithm computes the following expectation function

$$\begin{aligned} Q(\hat{\theta}|\theta) &= E_z[\log P(X, Z|\hat{\theta})|X, \theta] \\ &\propto \sum_{i=1}^{N}\sum_{j=1}^{V} n(d_i, w_j) \sum_{k=1}^{K} p(z_k|d_i, w_j) \log[p(z_k|d_i)p(w_i|z_k)]. \end{aligned}$$

The maximization (M) step then calculates the new estimate $\hat{\theta}$ by maximizing the expectation function $Q(\hat{\theta}|\theta)$ computed at E-step. The EM algorithm alternates between both steps until the log likelihood converges. It is shown in Hofmann (2001) that, the E step is equivalent to computing the posterior probability $p(z_k|d_i, w_j)$, given the estimated $p(w_j|z_k), p(z_k|d_i)$:

$$p(z_k|d_i, w_j) = \frac{p(w_j|z_k)p(z_k|d_i)}{\sum_{l=1}^{K} p(w_j|z_l)p(z_l|d_i)}. \tag{3.4}$$

Using the computed $p(z_k|d_i, w_j)$ from the E-step, the M step corresponds to the following updates:

$$p(w_j|z_k)^{new} = \frac{\sum_{i=1}^{N} n(d_i, w_j)p(z_k|d_i, w_j)}{\sum_{m=1}^{V}\sum_{i=1}^{N} n(d_i, w_m)p(z_k|d_i, w_m)}, \tag{3.5}$$

$$p(z_k|d_i)^{new} = \frac{\sum_{j=1}^{V} n(d_i, w_j)p(z_k|d_i, w_j)}{\sum_{l=1}^{K}\sum_{j=1}^{V} n(d_i, w_j)p(z_l|d_i, w_j)}. \tag{3.6}$$

In (3.6), the denominator can also be written as $n(d_i)$, i.e., the total number of words occurring in d_i.

During the inference(testing) stage, given a testing image/video d_{test}, the topic-based intermediate representation $p(z_k|d_{test})$ are computed by using the

"fold-in" heuristic described in Hofmann (2001). The heuristic employs the EM algorithm in the same way as in the learning, while fixing the values of coefficients $p(w_j|z_k)$ obtained from the training stage.

3.3 On-line PLSA

3.3.1 Notations and Conventions

We are to present our algorithm for the PLSA under the setting of on-line learning, where training data is assumed to arrive at successive time slices. Denote $D^{(t)} = \{d_1^{(t)}, d_2^{(t)}, ..., d_{N_t}^{(t)}\}$ as the data stream arriving at time t. and $F^{(t)} = \{f_1^{(t)}, f_2^{(t)}, ..., f_{N_{f_t}}^{(t)}\}$ as the feature descriptors from $D^{(t)}$. Let $W^{(t)} = \{w_1, w_2, ..., w_{V_t}\}$ be the codebook of visual words at different time slice t, $X^{(t)}$ be the co-occurrence matrix at time t.

3.3.2 On-line Codebook Adaptation for the PLSA

The goal of our algorithm is to learn the parameters of the PLSA model from data streams. To learn a PLSA model for visual recognition, a codebook of visual words is required. The k-means clustering algorithm is usually employed to produce the codebook using all the features in a training set. Niebles *et al.* show that a bigger sized codebook often results in a higher recognition accuracy Carlos Niebles *et al.* (2008). The size of the codebook is usually proportional to the number of features. However in practice, due to memory limitations, only a limited number of features are considered for the codebook construction when dealing with a large dataset. The produced codebook is not necessarily optimal as it only considers a subset of features. In order to construct a codebook that captures the full characteristics of all features, we present here an codebook adaptation algorithm for the codebook construction.

Given a feature vector $f_n^{(t)}$, and the corresponding codeword w_k, denote δ as the difference vector between $f_n^{(t)}$ and w_k, i.e., $\delta = f_n^{(t)} - w_k$. We simply assume that each dimension of δ follows an i.i.d. zero mean Gaussian distribution, and hence the squared Euclidean distance between $f_n^{(t)}$ and w_k can be approximately

Algorithm 1 On-line Codebook Adaptation

INPUTS:

$F^{(t)} = \{f_1^{(t)}, f_2^{(t)}, ..., f_{N_{f_t}}^{(t)}\}$ - feature vectors in the data stream at time (t).

$W^{(t-1)} = \{w'_1, w'_2, ..., w'_{V_{t-1}}\}$ - the codebook from time $(t-1)$.

$\Sigma^{(t-1)} = \{(\mu'_1, \sigma'_1), (\mu'_2, \sigma'_2), ..., (\mu'_{V_{t-1}}, \sigma'_{V_{t-1}})\}$ -distance statistics for $W^{(t-1)}$.

$\bar{\Sigma} = \{(\bar{\mu}, \bar{\sigma}^2)\}$ - the mean distance statistics computed from the initial set.

β - the learning factor.

OUTPUTS:

$W^{(t)} = \{w_1, w_2, ..., w_{V_t}\}$ - updated codebook for time (t).

$\Sigma^{(t)} = \{(\mu_1, \sigma_1), (\mu_2, \sigma_2)..., (\mu_{V_t}, \sigma_{V_t})\}$ - the distance statistics for $W^{(t)}$.

FOR each time slice $t \geq 1$

$V_t \leftarrow V_{t-1}$.

FOR each feature $f_n^{(t)}$

Choose a codeword w_k, where $k = \arg\min_i(\frac{\|f_n^{(t)} - w_i\|_2^2}{\sigma_i^2})$.

$d \leftarrow \|f_n^{(t)} - w_k\|_2^2$.

IF $d \leq \mu_k + 2.5\sigma_k$

$\mu_k \leftarrow \mu_k + \beta(d - \mu_k)$.

$\sigma_k^2 \leftarrow \sigma_k^2 + \beta(d - \mu_k)(d - \mu_k)$.

$w_k \leftarrow w_k + \beta(f_n^{(t)} - w_k.\ )$

ELSE

$V_t \leftarrow V_t + 1, \quad w_{V_t} \leftarrow f_n^{(t)}, \quad \mu_{V_t} \leftarrow \bar{\mu}, \quad \sigma_{V_t}^2 \leftarrow \bar{\sigma}^2$.

END IF

END FOR

END FOR

modeled as a Gaussian distribution Papoulis & Pillai (2002). The statistics μ_k and σ_k are the mean and standard deviation for the squared distances between w_k and all the features assigned to w_k.

The codebook adaptation algorithm is presented in Algorithm 1. As shown

in Algorithm 1, the inputs include the current feature set $F^{(t)}$, the previous codebook $W^{(t-1)}$, and also the distance statistics $\Sigma^{(t-1)}$ for $W^{(t-1)}$. The $\Sigma^{(t-1)}$ consists of corresponding means and variances for the squared distances between the features and corresponding codewords. During the adaptation process, each feature chooses one codeword to update. For each feature vector $f_n^{(t)}$, we select the codeword w_k that has the minimum of $\frac{\|f_n^{(t)}-w_k\|_2^2}{\sigma_k^2}$. Let d be the squared distance between $f_n^{(t)}$ and w_k. If the difference between d and μ_k is within 2.5 σ_k, both w_k and (μ_k, σ_k) will be updated using $f_n^{(t)}$. The updating factor β is set to be a flat rate (e.g., $\beta = 0.05$). If no match is found between $f_n^{(t)}$ and all the codewords, a new codeword is then initiated using $f_n^{(t)}$. The related distance statistics are initiated using $\bar{\mu}$, and $\bar{\sigma}^2$, which are the corresponding mean values computed from the initial training set.

3.3.3 On-line EM for PLSA Learning

The learning of the PLSA model in Section 3.2 employs the batch EM for parameter estimation. However, the batch EM is not applicable to data that arrives sequentially. To enable the learning of the PLSA under this situation, we propose an on-line learning algorithm for the PLSA. The idea of the proposed algorithm is to update the model parameter θ using $D^{(t)}$ received at each time slice. For a better understanding of the algorithm, we begin the on-line learning using the notations for batch learning in Section 3.2.

To enable the on-line learning for the PLSA, we firstly define two statistics $\langle n(d, w_j)\rangle_{z_k}$ and $\langle n(d, w)\rangle_{z_k}$ as follows:

$$\langle n(d, w_j)\rangle_{z_k} = \sum_{i=1}^{N} n(d_i, w_j) p(z_k|d_i, w_j), \;\; j \in \{1, 2, ..., V\}, \;\; k \in \{1, 2, ..., K\}. \tag{3.7}$$

$$\langle n(d, w)\rangle_{z_k} = \sum_{j=1}^{V}\sum_{i=1}^{N} n(d_i, w_j) p(z_k|d_i, w_j), \;\; k \in \{1, 2, ..., K\}. \tag{3.8}$$

Based on the above definitions, (3.5) in the M-step can be re-formulated as:

$$p(w_j|z_k) = \frac{\sum_{i=1}^{N} n(d_i, w_j) p(z_k|d_i, w_j)}{\sum_{m=1}^{V}\sum_{i=1}^{N} n(d_i, w_m) p(z_k|d_i, w_m)} = \frac{\langle n(d, w_j)\rangle_{z_k}}{\langle n(d, w)\rangle_{z_k}}. \tag{3.9}$$

The main idea of our proposed on-line EM is to replace the values of $\langle n(d,w_j)\rangle_{z_k}$ and $\langle n(d,w)\rangle_{z_k}$ with weighted mean values. We define the weighted mean for $\langle n(d,w)\rangle_{z_k}$ at time t as follows:

$$\langle n(d,w)\rangle_{z_k}^{(t)} = \eta(t)\sum_{\tau=1}^{t}((\prod_{s=\tau+1}^{t}\lambda(s))\sum_{i=1}^{N_\tau}\sum_{j=1}^{V}n(d_i^{(\tau)},w_j)p^{(\tau)}(z_k|d_i^{(\tau)},w_j)) \tag{3.10}$$

where

$$\eta(t) = (\sum_{\tau=1}^{t}\prod_{s=\tau+1}^{t}\lambda(s))^{-1}, \tag{3.11}$$

and $p^{(t)}(z_k|d_i^{(t)},w_j)$ represents the posterior probability computed using visual document stream $D^{(t)}$ and W. In (3.11), the parameter $\lambda(s)$ $(0 \le \lambda(s) \le 1,\ s = 1,2,3,...)$ is a time-dependent decaying factor that determines the contribution of the data stream at each time slice.

The $\langle n(d,w)\rangle_{z_k}^{(t)}$ in (3.10) can be expanded as follows:

$$\begin{aligned}\langle n(d,w)\rangle_{z_k}^{(t)} &= \eta(t)\lambda(t)\sum_{\tau=1}^{t}(\prod_{s=\tau+1}^{t-1}\lambda(s))\sum_{i=1}^{N_\tau}\sum_{j=1}^{V}n(d_i^{(\tau)},w_j)p^{(\tau)}(z_k=c|d_i^{(\tau)},w_j)\\ &= \eta(t)\lambda(t)\sum_{\tau=1}^{t-1}(\prod_{s=\tau+1}^{t-1}\lambda(s))\sum_{i=1}^{N_\tau}\sum_{j=1}^{V}n(d_i^{(\tau)},w_j)p^{(\tau)}(z_k=c|d_i^{(\tau)},w_j)\\ &+ \eta(t)\lambda(t)(\prod_{s=t+1}^{t-1}\lambda(s))\sum_{i=1}^{N_t}\sum_{j=1}^{V}n(d_i^{(t)},w_j)p^{(t)}(z_k=c|d_i^{(t)},w_j).\end{aligned} \tag{3.12}$$

We have

$$\begin{aligned}&\eta(t)\lambda(t)(\prod_{s=t+1}^{t-1}\lambda(s))\sum_{i=1}^{N_t}\sum_{j=1}^{V}n(d_i^{(t)},w_j)p^{(t)}(z_k=c|d_i^{(t)},w_j)\\ &= \eta(t)(\prod_{s=t+1}^{t}\lambda(s))\sum_{i=1}^{N_t}\sum_{j=1}^{V}n(d_i^{(t)},w_j)p^{(t)}(z_k=c|d_i^{(t)},w_j)\\ &= \eta(t)\sum_{i=1}^{N_t}\sum_{j=1}^{V}n(d_i^{(t)},w_j)p^{(t)}(z_k=c|d_i^{(t)},w_j)\end{aligned} \tag{3.13}$$

due to the fact that:

$$\prod_{s=t+1}^{t}\lambda(s) = 1. \tag{3.14}$$

Combining (3.12) and (3.13), we can rewrite $\langle n(d,w)\rangle_{z_k}^{(t)}$ as:

$$\langle n(d,w)\rangle_{z_k}^{(t)} = \eta(t)\lambda(t)\sum_{\tau=1}^{t-1}(\prod_{s=\tau+1}^{t-1}\lambda(s))\sum_{i=1}^{N_\tau}\sum_{j=1}^{V} n(d_i^{(\tau)},w_j)p^{(\tau)}(z_k=c|d_i^{(\tau)},w_j) + \eta(t)\sum_{i=1}^{N_t}\sum_{j=1}^{V} n(d_i^{(t)},w_j)p^{(t)}(z_k=c|d_i^{(t)},w_j). \tag{3.15}$$

Considering the fact that

$$\langle n(d,w)\rangle_{z_k}^{(t-1)} = \eta(t-1)\sum_{\tau=1}^{t-1}(\prod_{s=\tau+1}^{t-1}\lambda(s))\sum_{i=1}^{N_\tau}\sum_{j=1}^{V} n(d_i^{(\tau)},w_j)p^{(\tau)}(z_k=c|d_i^{(\tau)},w_j) \tag{3.16}$$

based on (3.10), we simplify (3.15) as

$$\begin{aligned}\langle n(d,w)\rangle_{z_k}^{(t)} &= \frac{\eta(t)}{\eta(t-1)}\lambda(t)\langle n(d,w)\rangle_{z_k}^{(t-1)} + \eta(t)\sum_{i=1}^{N_t}\sum_{j=1}^{V} n(d_i^{(t)},w_j)p^{(t)}(z_k=c|d_i^{(t)},w_j)\\ &= \langle n(d,w_j)\rangle_{z_k}^{(t-1)} + \eta(t)\{\sum_{i=1}^{N_t}\sum_{j=1}^{V} n(d_i^{(t)},w_j)p^{(t)}(z_k=c|d_i^{(t)},w_j)\\ &+ (\frac{\lambda(t)}{\eta(t-1)} - \frac{1}{\eta(t)})\langle n(d,w)\rangle_{z_k}^{(t-1)}\}.\end{aligned} \tag{3.17}$$

Based on (3.11), we have

$$\begin{aligned}\eta(t) &= (\sum_{\tau=1}^{t}\prod_{s=\tau+1}^{t}\lambda(s))^{-1} = (1+\lambda(t)\sum_{\tau=1}^{t-1}\prod_{s=\tau+1}^{t-1}\lambda(s))^{-1}\\ &= (1+\lambda(t)(\eta(t-1))^{-1})^{-1}\\ &= \frac{\eta(t-1)}{\lambda(t)+\eta(t-1)},\end{aligned} \tag{3.18}$$

which can be rewritten as:

$$\frac{1}{\eta(t)} = \frac{\lambda(t)}{\eta(t-1)} + 1. \tag{3.19}$$

Substituting (3.19) into (3.17), we can get

$$\langle n(d,w)\rangle_{z_k}^{(t)} = (1-\eta(t))\langle n(d,w)\rangle_{z_k}^{(t-1)} + \eta(t)[\sum_{i=1}^{N_t}\sum_{j=1}^{V} n(d_i^{(t)},w_j)p^{(t)}(z_k|d_i^{(t)},w_j)]. \tag{3.20}$$

Similarly, $\langle n(d,w_j)\rangle_{z_k}^{(t)}$ can also be simplified as:

$$\langle n(d,w_j)\rangle_{z_k}^{(t)} = (1-\eta(t))\langle n(d,w_j)\rangle_{z_k}^{(t-1)} + \eta(t)[\sum_{i=1}^{N_t} n(d_i^{(t)},w_j)p^{(t)}(z_k|d_i^{(t)},w_j)]. \tag{3.21}$$

Using (3.20), and (3.21), the on-line estimation for $p(w_j|z_k)$ at time t can be written as follows:

$$p^{(t)}(w_j|z_k) = \frac{\langle n(d,w_j)\rangle_{z_k}^{(t)}}{\langle n(d,w)\rangle_{z_k}^{(t)}}. \tag{3.22}$$

We can see from (3.20) and (3.21) that the values of $\langle n(d,w)\rangle_{z_k}^{(t)}$ and $\langle n(d,w_j)\rangle_{z_k}^{(t)}$ are obtained by combining previous values $\langle n(d,w)\rangle_{z_k}^{(t-1)}$ and $n(d,w_j)\rangle_{z_k}^{(t-1)}$ with current data $D^{(t)}$ using different weights respectively. The weights are controlled by the normalization coefficient $\eta(t)$ $(1/t \leq \eta(t) \leq 1)$ which is used as a learning rate. Sato *et al.* point out in Sato (2000) that the on-line EM is a stochastic approximation. To guarantee the estimate to converge to a local maximum of the likelihood function of the data, $\eta(t)$ must satisfy the following conditions:

$$\lim_{t\to\infty}\eta(t) = 0, \sum_{t=1}^{\infty}\eta(t) = \infty, \sum_{t=1}^{\infty}\eta^2(t) < \infty. \tag{3.23}$$

We employ the $(1/t)$-schedule as it is designed to match these conditions. Under this schedule, $\eta(t)$ is set to be $1/t$, which means each stream is weighted equally, i.e., $\lambda(s) = 1$. To combine this on-line algorithm with the codebook adaptation in Section 3.3.2, $\eta(t)$ is set to 1 for any newly created codeword w_j, since there is no past experience for the new codeword to learn from.

The workflow of the on-line learning algorithm with codebook adaptation is depicted in Figure 3.1, and its detailed algorithm can be found in Algorithm 2. The initial parameter estimates $\langle n(d,w_j)\rangle_{z_k}^{(0)}$, $\langle n(d,w)\rangle_{z_k}^{(0)}$, are estimated from an initial training set $D^{(0)}$ using standard EM of PLSA. During on-line learning, our algorithm performs codebook adaptation using $F^{(t)}$ from data stream $D^{(t)}$ at each time slice t, and then conducts the on-line learning based on the updated codebook $W^{(t)}$. As a result, the proposed algorithm can automatically and adaptively learn from data streams.

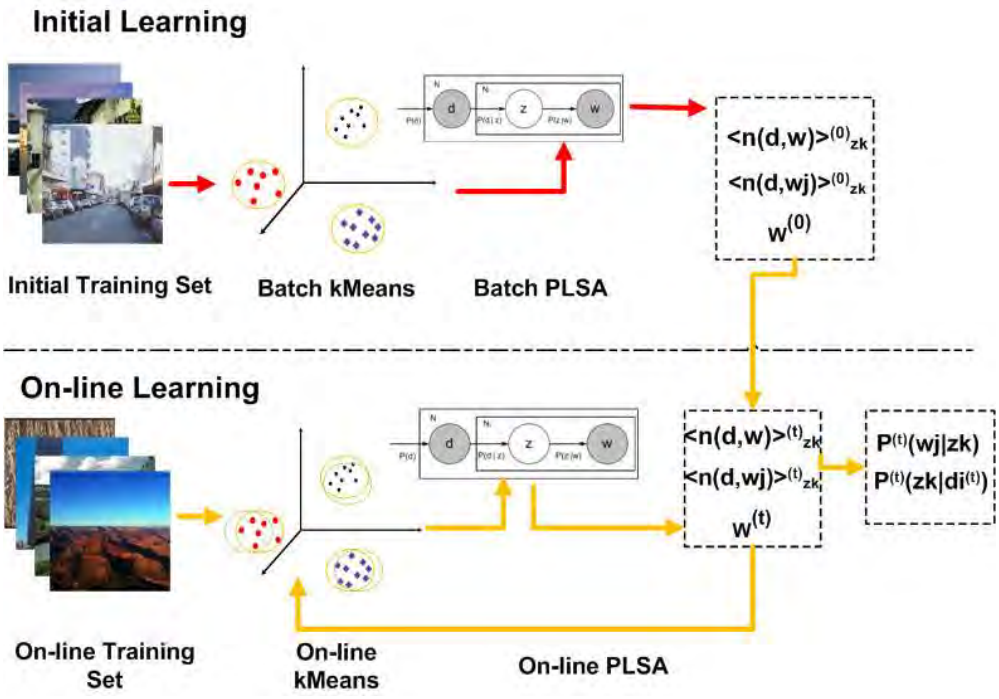


Figure 3.1: The workflow of the proposed algorithm. The initial training employs the batch PLSA, whereas the on-line training applies the proposed on-line learning.

3.3.4 Comparison with the QB-PLSA

Previous work on the on-line learning of the PLSA and other latent models can be found in text mining area Chou & Chen (2008), AlSumait *et al.* (2008), Chien & Wu (2008). However, all these algorithms are proposed for text mining only, and there is no visual codebook and its adaptation involved. The most related work to our algorithm is the QB-PLSA proposed by Chien *et al.* in Chou & Chen (2008). In this section we compare the proposed algorithm with the QB-PLSA.

The QB-PLSA essentially models the priors of the PLSA parameters using Dirichlet distributions. In the QB-PLSA, the on-line learning is achieved by using the Quasi Bayesian estimation Hamilton (1991). Following the notations in Section 3.1 and 3.3, we denote $D_t = \{D^{(1)}, D^{(2)}, ..., D^{(t)}\}$ as the training data

Algorithm 2 On-line EM Algorithm for PLSA Learning

INITIAL STAGE

Estimate $\langle n(d,w)\rangle_{z_k}^{(0)}$, $\langle n(d,w_j)\rangle_{z_k}^{(0)}$ from the initial training set.

Construct the initial codebook $W^{(0)}$ using $F^{(0)}$ from the initial training set.

ON-LINE LEARNING STAGE

FOR each image stream at time slice t

Codebook Adaptation:

Compute $W^{(t)}$ using $F^{(t)}$ and $W^{(t-1)}$ based on Algorithm 1.

Compute $X^{(t)} = \{n(d_i^{(t)}, w_j)\}$ with $W^{(t)}$ and $F^{(t)}$.

On-line EM Parameter Estimation:

Initialize $p^{(t)}(w_j|z_k)$, $p^{(t)}(z_k|d_i^{(t)})$ based on $W^{(t)}$.

FOR each EM iteration n

E-step:

Compute $p^{(t)}(z_k|d_i^{(t)}, w_j)$ using $p^{(t)}(w_j|z_k)$ and $p^{(t)}(z_k|d_i^{(t)})$ with (3.4).

M-step:

Compute $\langle n(d,w)\rangle_{z_k}^{(t)}$ using $X^{(t)}$, $\langle n(d,w)\rangle_{z_k}^{(t-1)}$, & $p^{(t)}(z_k|d_i^{(t)}, w_j)$ with (3.20).

Compute $\langle n(d,w_j)\rangle_{z_k}^{(t)}$ using $X^{(t)}$, $\langle n(d,w_j)\rangle_{z_k}^{(t-1)}$, & $p^{(t)}(z_k|d_i^{(t)}, w_j)$ with (3.21).

Compute $p^{(t)}(w_j|z_k)$ using $\langle n(d,w)\rangle_{z_k}^{(t)}$ and $\langle n(d,w_j)\rangle_{z_k}^{(t)}$ with (3.22).

Compute $p^{(t)}(z_k|d_i^{(t)})$ using $X^{(t)}$ and $p^{(t)}(z_k|d_i^{(t)}, w_j)$ with (3.6).

Check convergence:

compute $L^{(n)}(X^{(t)}|\theta)$ using $p^{(t)}(w_j|z_k)$ and $p^{(t)}(z_k|d_i^{(t)})$ with (3.3).

IF new likelihood $L^{(n)}$ satisfies convergence condition $|\frac{L^{(n)}-L^{(n-1)}}{L^{(n-1)}}| < \epsilon$

Terminate EM iteration.

END IF

END FOR

END FOR

received up to time t. According to the QB estimate, we have

$$\begin{aligned}\theta_{QB}^{(t)} &= \arg\max_{\theta} P(\theta|D_t) \\ &= \arg\max_{\theta} P(D^{(t)}|\theta)P(\theta|D_{t-1}) \\ &\cong \arg\max_{\theta} P(D^{(t)}|\theta)g(\theta|\varphi^{(t-1)}),\end{aligned} \tag{3.24}$$

where the posterior probability $P(\theta|D_{t-1})$ is approximated by the closest tractable prior probability distribution $g(\theta|\varphi^{(t-1)})$. The prior distribution $g(\theta|\varphi^{(t-1)})$ models the randomness of the parameter θ, with the hyper parameters $\varphi^{(t-1)}$ which evolve from historical data. It can be observed from (3.24) that the QB estimate provides a method for learning the parameters of the PLSA in an on-line manner. At time t, we can use the data $D^{(t)}$ to update the parameter θ and the prior distribution $g(\theta|\varphi^{(t-1)})$, and $D^{(t)}$ can be released after the update. The introduction of the prior distribution for θ plays an important role in the QB-PLSA. In Bayesian inference, *conjugate prior* is usually chosen for reproducible prior and posterior distribution pairs. Due to the multinomial nature of the PLSA parameters, the Dirichlet distribution is chosen to model the prior of θ. Given the assumption that all parameters are independent, the prior distribution for the θ can be expressed as:

$$g(\theta|\varphi^{(t-1)}) \propto \prod_{k=1}^{K}[\prod_{k=1}^{V} P(w_j|z_k)^{\alpha_{j,k}^{(t-1)}-1} \prod_{i=1}^{N_t} P(z_k|d_i)^{\beta_{k,i}^{(t-1)}-1}], \tag{3.25}$$

where $\varphi^{(t-1)} = \{\{\alpha_{j,k}^{(t-1)}\}_{j,k}, \{\beta_{k,i}^{(t-1)}\}_{k,i}\}$ are the hyper parameters of the Dirichlet distribution. At time t, given $\varphi^{(t-1)}$, $\theta^{(t)}$ can be estimated by maximizing the logarithm of posterior probability, which can be written as:

$$\begin{aligned} \theta_{QB}^{(t)} &\cong \arg\max_{\theta} P(D^{(t)}|\theta) g(\theta|\varphi^{(t-1)}) \\ &= \arg\max_{\theta} [\log P(D^{(t)}|\theta) + \log g(\theta|\varphi^{(t-1)})]. \end{aligned} \tag{3.26}$$

As a result, the extended expectation function can be expressed as follows:

$$\begin{aligned} R(\hat{\theta}^{(t)}|\theta^{(t)}) \propto & \sum_{i=1}^{N_t}\sum_{j=1}^{V} n(d_i, w_j) \sum_{k=1}^{K} p^{(t)}(z_k|d_i, w_j) \log[p^{(t)}(z_k|d_i) p^{(t)}(w_i|z_k)] \\ & + \sum_{j=1}^{V}\sum_{k=1}^{K} (\alpha_{j,k}^{(t-1)} - 1) \log p^{(t)}(w_j|z_k) \\ & + \sum_{i=1}^{N_t}\sum_{k=1}^{K} (\beta_{k,i}^{(t-1)} - 1) \log p^{(t)}(z_k|d_i). \end{aligned} \tag{3.27}$$

Following the EM algorithm, we calculate the extended expectation function $R(\hat{\theta}^{(t)}|\theta^{(t)})$ in the E-step, and we maximize it with respect to θ in the M-step. In the implementation, the E-step computes the posterior probability $p^{(t)}(z_k|d_i, w_j)$,

given the estimated $p^{(t)}(w_j|z_k), p^{(t)}(z_k|d_i)$, while the M-step corresponds to the following updates:

$$p^{(t)}(w_j|z_k)^{new} = \frac{\sum_{i=1}^{N_t} n(d_i, w_j) p^{(t)}(z_k|d_i, w_j) + (\alpha_{j,k}^{(t-1)} - 1)}{\sum_{m=1}^{V}[\sum_{i=1}^{N_t} n(d_i, w_m) p^{(t)}(z_k|d_i, w_m) + (\alpha_{j,m}^{(t-1)} - 1)]}, \tag{3.28}$$

$$p^{(t)}(z_k|d_i)^{new} = \frac{\sum_{j=1}^{V} n(d_i, w_j) p^{(t)}(z_k|d_i, w_j) + (\beta_{k,i}^{(t-1)} - 1)}{n(d_i) + \sum_{l=1}^{K}(\beta_{l,i}^{(t-1)} - 1)}. \tag{3.29}$$

The final estimates for $p^{(t)}(w_j|z_k)$ and $p^{(t)}(z_k|d_i)$ can be obtained once the EM converges to a local optimum. In addition to the parameter updates, the QB-PLSA is also geared with the hyper parameter updates. The exponential of the extended expectation function can be written as a new Dirichlet distribution:

$$\exp\{R(\hat{\theta}^{(t)}|\theta^{(t)})\} \propto \sum_{k=1}^{K}[\sum_{j=1}^{V} p^{(t)}(w_j|z_k)^{(\alpha_{j,k}^{(t-1)}-1)} \sum_{i=1}^{N_t} p^{(t)}(z_k|d_i)^{(\beta_{k,i}^{(t-1)}-1)}]. \tag{3.30}$$

The update equations for the hyper parameters at time t can then be produced as follows:

$$\alpha_{j,k}^{(t)} = \sum_{i=1}^{N_t} n(d_i, w_j) p^{(t)}(z_k|d_i, w_j) + \alpha_{j,k}^{(t-1)}, \tag{3.31}$$

$$\beta_{k,i}^{(t)} = \sum_{j=1}^{V} n(d_i, w_j) p^{(t)}(z_k|d_i, w_j) + \beta_{k,i}^{(t-1)}. \tag{3.32}$$

Both the proposed on-line PLSA and the QB-PLSA tackle the parameter updates under the EM framework. The motivation of the QB-PLSA is to address the parameter updates in a continuously changing word of documents Chien & Wu (2008). By the introduction of prior distribution and its hyper parameter updating scheme, the QB-PLSA can adapt the model in an on-line manner. Despite of the reported encouraging results, their method assumes some parametric models which might limit their capability of modeling complicated dataset. In contrast, our method is more flexible in the sense that it is data-driven and no parametric assumptions are made.

3.4 Experiments

We evaluate the performance of the proposed algorithm for both scene recognition and human action recognition. All the experiments are conducted on a PC with an Intel 2.09 GHz Core Duo CPU and 1GB of RAM. We use the Matlab implementation of PLSA code provided by Fergus *et al.* R. Fergus & Torralba (2005). We compare the on-line learning and batch learning in the experiments. It is noted that statistical significance testing is not conducted as no significant testing was reported in related papers.

3.4.1 The Picasa Scene Data

To demonstrate the advantages of the proposed algorithm, we conduct our first experiment using web images. We issue queries to the Google Picasa Web Album using the provided API Google (2009), and receive the images sequentially. All the collected images are resized to 320-by-256 pixels.

We repeat the experiments for 5 runs, and the average recognition performance is used for comparison. At each run, we firstly collect 238 images as the initial training set, and also 238 images as the testing set. We then collect different number streams of images for the on-line learning. We use 2, 3, and 4 streams. We collected 4743 images from 7 different scene categories in total at each run. Samples of the images can be found in Figure 3.2.

For feature extraction, we use the SIFT descriptor Lowe (2004) as the local feature descriptor. Feature points are densely sampled on regular grids from each image, and 80 SIFT descriptors are extracted per image. We adopt the scene recognition approach using the PLSA and a k-NN classifier proposed in Bosch *et al.* (2008) in the experiments. For the PLSA parameters, the codebook size is set to 1300, and the number of topics is fixed to 25. The parameter k for the k-NN classification is set to 17.

For the batch learning, we intend to learn a batch model using all the images from both the initial set and the on-line training set, which is 4505 in total. Let $|D|$, $|W|$, and $|Z|$ be the number of images, the codebook size, and the number of topics. The memory requirement for our Matlab based implementation is

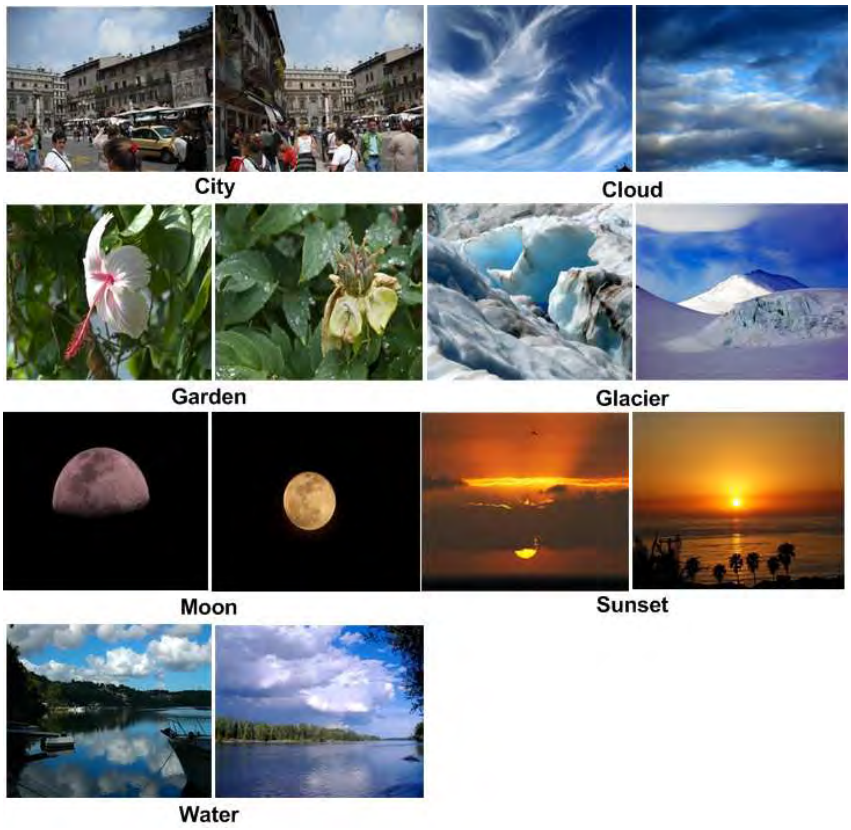
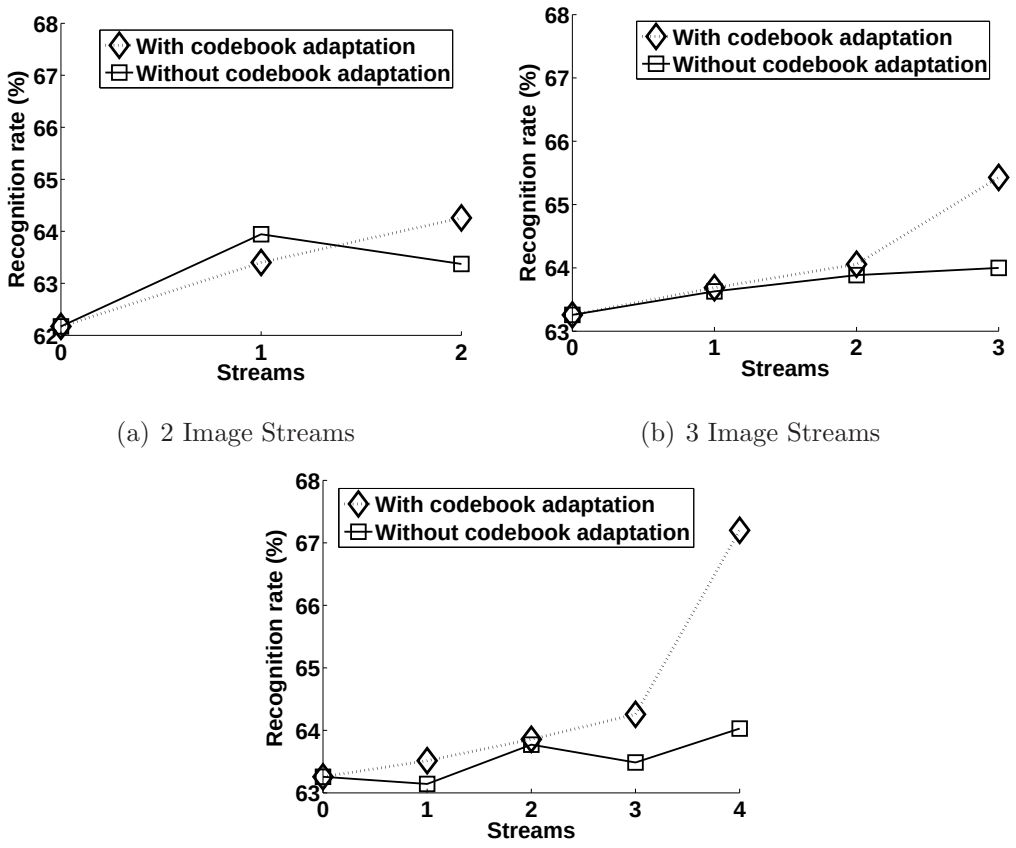


Figure 3.2: Sample images from the Picasa scene dataset.

$64 * |D||W||Z|$ bytes (In this experiment, $|D|$, $|W|$, and $|Z|$ are 4505, 1300, and 25 respectively). This memory requirement exceeds the Matlab memory capacity on our computer, and therefore the conventional batch PLSA learning cannot be performed on our system. Compared with the batch learning algorithm, the proposed algorithm has the advantages of lower memory consumption since only a subset of the training data is processed at each time.

At each run of the on-line learning, an initial model is learned using the batch learning algorithm on the initial training set. An initial codebook is also obtained by clustering all the features from the initial set. We then update the initial model using image streams. After each update, the topic vectors for the images from both the initial set and the testing set are re-computed using the updated model. The category of each testing image is determined by a k-NN search in



(a) 2 Image Streams

(b) 3 Image Streams

(c) 4 Image Streams

Figure 3.3: The performance of on-line learning algorithm based on the Google Picasa Web Album.

the initial set. To see the effect of the codebook adaptation, the experiments are conducted using the on-line learning algorithms with and without codebook adaptation respectively. Figure 3.3 reveals the overall trend of recognition rates throughout the on-line learning. It can be seen from Figure 3.3 that, the codebook adaptation helps to improve the recognition performance under different stream size settings. Table 3.1 summarizes the experimental results, and the highest accuracy of 67.20% is obtained by the algorithm with codebook adaptation.

Table 3.1: Summary of performance on the Picasa scene dataset. Results are reported as the average accuracies of 5 runs.

#streams	2	3	4
Without codebook adaptation	64.37%	64.00%	64.02%
With codebook adaptation	65.54%	65.42%	67.20%
Batch Learning	N/A		

3.4.2 The OT Scene Dataset

In our second experiment, we compare the proposed algorithm with the conventional batch PLSA, and also the QB-PLSA proposed in Chien & Wu (2008), using the OT scene dataset Oliva & Torralba (2001). The OT dataset contains 2688 images from 8 different scene categories, and they are 360 coasts, 328 forest, 260 highway, 308 inside of cities, 374 mountain, 410 open country, 292 streets, and 356 tall buildings. The average size of each image is 256-by-256 pixels. Sample images can be found in Figure 3.4.

For feature extraction, the SIFT descriptor Lowe (2004) is employed as the local feature descriptor. We again use the scene recognition approach using the PLSA and a k-NN classifier in this experiment, by following the settings detailed in Horster *et al.* (2008). The number of topics for the PLSA is set to 25, and the codebook size is set to 650. We optimize the k for the k-NN classifier in the experiments.

The experiments are repeated for 5 runs, and the average recognition performance is used for comparison. At each run, the dataset is randomly divided into 538 initial training images, 1612 on-line training images, and also 538 testing images. No labeling information is used in the random division. The on-line training set is further divided into 3 streams to test the on-line learning.

At each run of the batch learning, a PLSA model is learned using all the images from both the initial set and the on-line training set, and tested using the testing set. The average performance is used for comparison. Figure 3.5(a) depicts the average recognition rate for different k's of the k-NN algorithm on the testing set. It can be seen from the figure that, the best recognition rate

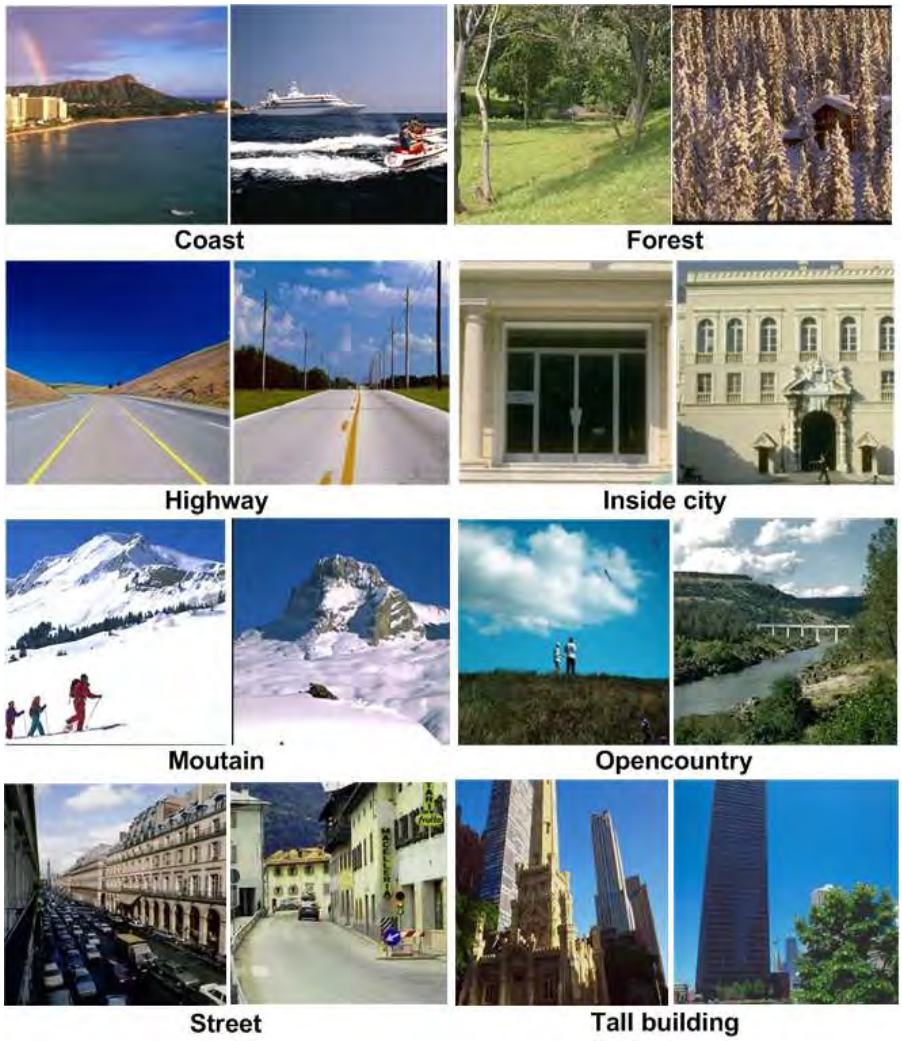


Figure 3.4: Sample images from the OT scene dataset.

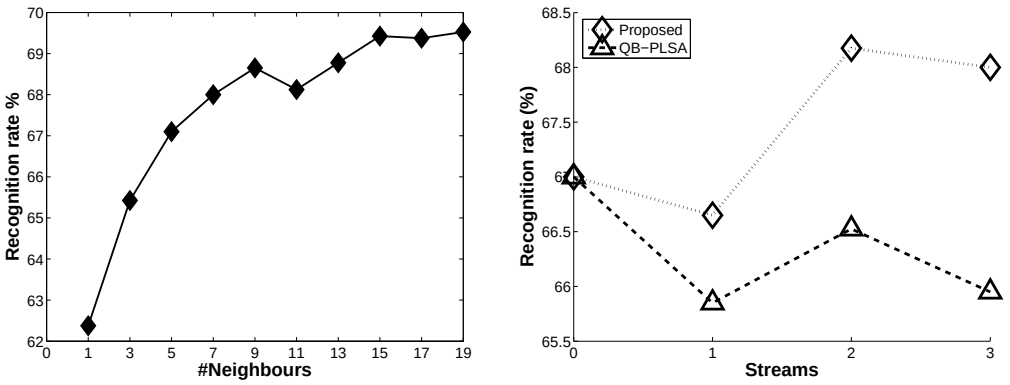
of 69.52% for the batch-learned PLSA is obtained when $k = 19$. This figure is consistent with the testing accuracy reported in Horster *et al.* (2008), based on the similar settings. As a result, $k = 19$ for the k-NN algorithm will serve in the following sections as a baseline for the evaluation of the proposed algorithm.

At each run of the on-line learning, we compare the proposed algorithm with the QB-PLSA. The QB-PLSA is proposed for text mining, and there is no codebook adaptation involved. To enable the QB-PLSA for scene recognition, the initial codebook generated from the initial training set is used as the visual code-

Table 3.2: Summary of performance on the OT scene dataset. Results are reported as the average accuracies of 5 runs.

Algorithms	Accuracies
QB-PLSA	65.95%
Proposed	68.00%
Batch Learning	69.53%

book for the QB-PLSA. In comparison with the QB-PLSA, the proposed algorithm employs the codebook adaptation to capture the feature characteristics in new data; it is more flexible as there is no assumption made on the distributions of the PLSA parameters. The average recognition performances for both on-line algorithms are demonstrated in Figure 3.5(b). It can be observed from Figure 3.5(b) that, the proposed algorithm achieves better recognition performance than the QB-PLSA throughout the entire learning.



(a) (b)

Figure 3.5: The recognition performance on the OT scene dataset. (a) The average recognition rates for different k's of the k-NN algorithm on the testing set, based on the batch-trained PLSA model. (b) The performance of both the proposed algorithm and the QB-PLSA.

Table 3.2 summarizes the recognition accuracies of all the learning algorithms, and it shows that the performance of the proposed algorithm is comparable with that of the batch-learned PLSA model.

Figure 3.6: Sample frames from the KTH action dataset.

3.4.3 The KTH Action Dataset

In our last experiment, we use the KTH human action dataset Schuldt *et al.* (2004) for evaluation. The KTH dataset is a single-view video dataset in human actions, which consists of 598 action videos. Six categories of human actions can be found in this dataset, and they are boxing, jogging, running, walking, hand waving and hand clapping. Each action is performed several times by 25 subjects in different environments with scale changes. There are only a single action in each video. Sample frames can be found in Figure 3.6.

For feature extraction, we choose the separable linear filter in Dollar *et al.* (2005b) for space-time interest point extraction in videos. The parameters of the separable linear filter are set to σ=2, and τ=2.5. Each extracted space-time patch is described as a concatenated vector of its brightness gradient. Then all the descriptors are projected to 100 dimensions using the PCA. The codebook size is set to 1000 for the KTH dataset.

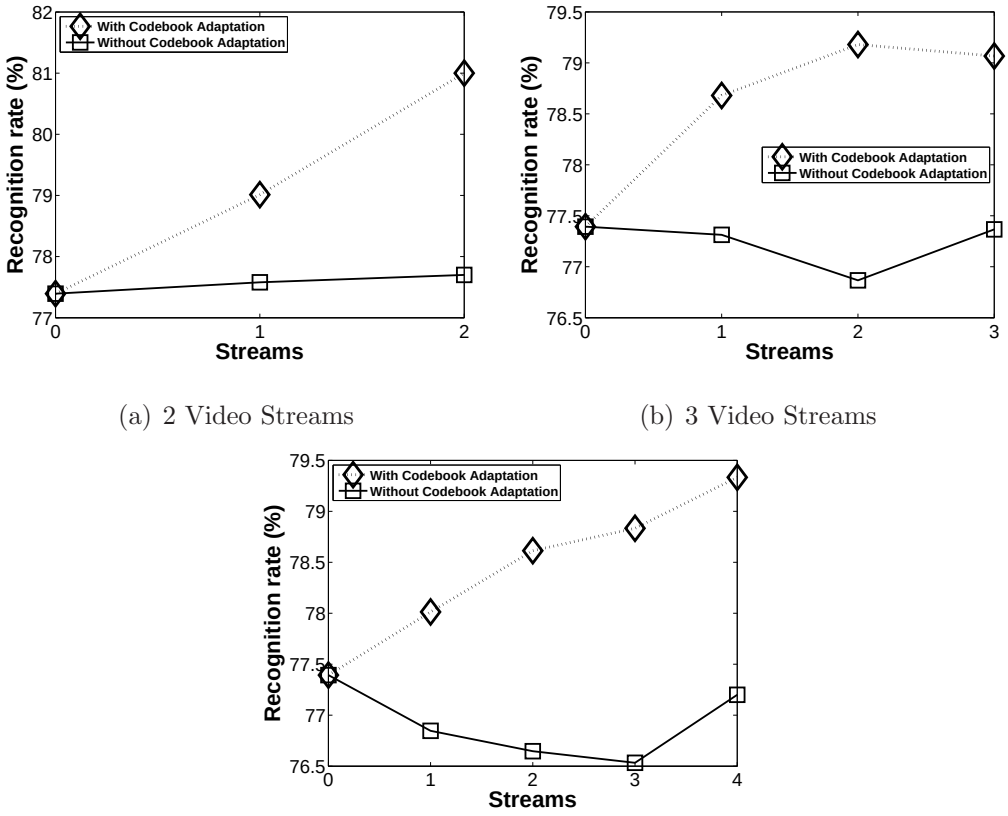
The experiments are conducted based on the leave-one-out testing paradigm (LOO). Since the KTH dataset contains a large number of features, only a sub-

set of features are chosen to construct the batch codebook Carlos Niebles *et al.* (2008), and we construct the codebook by using videos of only three subjects. After the codebook construction, a PLSA model is learned from the videos of 24 subjects using the batch learning algorithm, and is tested against the videos of the remaining subject. We repeat the LOO batch learning for 25 runs.

At each LOO run of the on-line learning, all the videos of one subject are used as the initial training set. The initial training set is used to construct the initial codebook, and also used to train the initial model, using the batch PLSA learning algorithm. Remaining training videos form the on-line learning set for adapting the model. The on-line training set is randomly split into different number of video streams without using any labeling information, and we use 2, 3, and 4 video streams in the experiments. After learning from each video stream, the adapted model is tested against the testing videos. We evaluate both on-line learning algorithms with and without codebook adaptation.

A confusion matrix is computed for each testing, and the recognition accuracies for the trained model are reported as the mean values of diagonal elements from the average confusion table of 25 runs. The performances of the proposed algorithm under different streams settings are depicted in Figure 3.7. Figure 3.7 also shows the codebook adaptation helps to improve the recognition performance for the on-line learning. Table 3.3 summarizes the performance of batch learning and on-line learning. It can be observed from the table that our batch-trained PLSA achieves the average accuracy of 80.17%, which is consistent with that reported in Carlos Niebles *et al.* (2008). The table reveals that the performance of the proposed algorithm is comparable with that of the conventional batch algorithm on this dataset.

Finally we compare the proposed algorithm with the QB-PLSA, with 7 video streams. This time we use the adapted codebook for the QB-PLSA at each stream.The accuracies of both algorithms on each of the video streams are depicted in Figure 3.8. The figure shows that the proposed algorithm also outperforms the QB-PLSA on the KTH dataset. The summary of final performance for the QB-PLSA, the proposed algorithm and the batch PLSA is summarized in Table 3.4.



(a) 2 Video Streams

(b) 3 Video Streams

(c) 4 Video Streams

Figure 3.7: The performance of on-line learning algorithm based on the KTH dataset.

Table 3.3: Summary of performance on the KTH action dataset. Results are reported as the averages of 25 runs.

#streams	2	3	4
Without codebook adaptation	77.70%	77.36%	77.20%
With codebook adaptation	81.00%	79.06%	79.33%
Batch Learning	80.17%		

3.5 Conclusions

In this chapter, we have proposed an on-line learning algorithm for PLSA based visual recognition. The proposed algorithm can automatically and adaptively

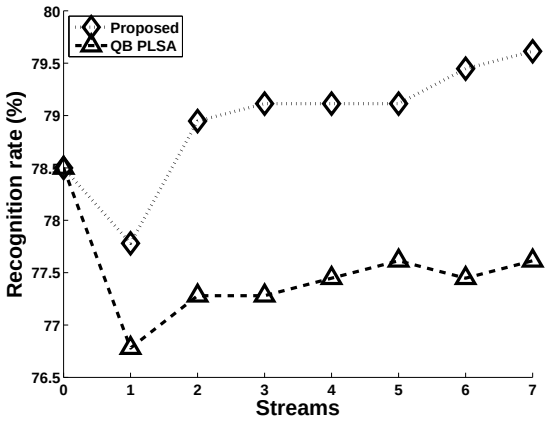


Figure 3.8: The performance of both the proposed algorithm and QB-PLSA on the KTH dataset.

Table 3.4: Summary of final performance on the KTH dataset. Results are reported as the average accuracies of 25 runs.

Algorithms	Accuracies
The QB-PLSA	77.61%
The Proposed Algorithm	79.61%
Batch Learning	80.17%

learn the parameters of the PLSA model during on-line learning. We evaluate the proposed algorithm using datasets for scene recognition and human action recognition. Experimental results demonstrate that the proposed algorithm can handle the data that the batch PLSA learning cannot deal with, and its performance is comparable with that of the batch PLSA learning on visual recognition.

Chapter 4

Unsupervised Learning for Codebook based Visual Detection

4.1 Introduction

In this chapter we study the codebook based object detection task. We focus on the special case of moving object detection. The detection of moving objects in videos, especially pedestrians, is an essential and important task in many vision applications, such as video compression, video surveillance, and content based video retrieval.

Numerous approaches have been proposed in the literature for object detection. Traditionally global methods are used. They learn the global statistics of robust image features for objects. The examples for features include the histogram of oriented gradients (HoG) Dalal & Triggs (2005), the Haar Wavelets Viola & Jones (2001), and the edge templates Gavrila (2000). At runtime, the learned detector examines image features over locations and scales to predict the presence of objects in subwindows. Though global methods has been demonstrated effective in many cases, it can be affected by background clutters and occlusions. To cope with occlusions, part based methods Andriluka *et al.* (2008), Fergus *et al.* (2007) are proposed to model each individual object as a collection of different parts. Specifically, parts can be modeled either generatively Andriluka

et al. (2008), Bouchard *et al.* (2005), Leibe *et al.* (2008), or discriminatively Micilotta *et al.* (2005). At runtime, part responses are consolidated to form object hypotheses. Part based approaches have been demonstrated to possess considerable tolerance to partial occlusions. Global and part based methods complement each other. As such, hybrid methods are proposed to take advantages of both methods. For example, Felzenszwalb *et al.* present a deformable part model for object detection in Felzenszwalb *et al.* (2008), which consists of a coarse global template and finer part templates. In addition to appearance information, motion information is also useful in object detection. Motion based methods usually work by clustering moving pixels into layers with consistent motions, and different models are employed to aid the clustering, such as the Dynamic Conditional Random Field (DCRF) model Wang & Ji (2005), the Markov Random Field (MRF) model Han *et al.* (2006), etc. The aforementioned approaches take advantages of appearance and motion cues for object recognition. However they ignore contextual cues, which play an important role in biological vision. Psychophysics studies have shown that the analysis of contextual relationships between visual concepts play an important role in human vision Biederman *et al.* (1982). As a result, the detection of a concept of interest can be facilitated by the presence of other concepts which might not even be of interest. This has led to the effort to account for context based recognition Shotton *et al.* (2008), Galleguillos *et al.* (2008), Divvala *et al.* (2009), Choi *et al.* (2010).

The Implicit Shape Model (ISM) Leibe *et al.* (2008) is a part based model for object detection. It is a codebook based model as information collected from local parts is retained in codebooks. At the learning stage, local features are extracted from training images. A codebook of local features is then constructed by clustering the extracted features. At runtime, local features are extracted from images, and then they are used to match against the codewords in the codebook. The codebook instances would cast votes for valid matches. Object hypotheses are obtained by aggregating the votes. The ISM based object detection has been shown effective on various tasks. For example, Leibe *et al.* Leibe *et al.* (2005) employ an appearance based feature for pedestrian detection, and experimental results show that this technique is able to detect pedestrians under crowded situations. A shape based feature is used for pedestrian detection under a similar

framework in Rodriguez & Shah (2007). In this chapter we propose a unsupervised learning algorithm that extends the ISM for automatic moving object detection. Our contributions are two-fold: Firstly, existing ISM based methods require the manual labeling of training samples. We propose two novel automatic training sample selection algorithms to replace the manual labeling. Secondly, we propose a method for identifying moving edges in video frames so that object hypotheses can be generated from the moving edges. In comparison with existing ISM based object detection methods Rodriguez & Shah (2007), Leibe *et al.* (2008), our method reduces the searching space for hypotheses generation.

The rest of this chapter is organized as follows: related work is reviewed in Section 4.2, and the proposed method is described in Section 4.3. The experimental results are presented in Section 4.4, followed by the conclusions in Section 4.5.

4.2 Related Work

Several approaches have been proposed to tackle the manual labeling of training data. One idea is to generate a large amount of training data from a small labeled set. One example is the co-training method Balcan *et al.* (2005) proposed by Balcan *et al.*. Given a small hand labeled set, the co-training trains a *pair* of classifiers on two independent "views" (features) of the data. It then produces additional training data from the unlabeled data based on the concord between the two classifiers. Although it is required that the "views" should be statistically independent for the co-training to achieve the best result, it still achieves good results when the independence assumption does not hold Levin *et al.* (2003). The co-training has been employed to boost ensemble classifiers for the classification of moving blobs into vehicles and pedestrians in Javed *et al.* (2005). Alternatively, Wu *et al.* use a small labeled set to train a labeler. The labeler, which has high precision, segments and labels objects from unlabeled data automatically Wu & Nevatia (2007).

These aforementioned approaches require a hand labeled set for initialization. To overcome the limitation of hand labeling, the idea of automatic labeling is

proposed. For example, Nair *et al* design a simple and heuristic labeler based on the background subtraction Nair & Clark (2004). Specifically, foreground blobs with approximately correct aspect ratios and sizes are labeled as 'objects', whereas image regions contain no foreground are labeled as 'non-objects'. To make the background subtraction based automatic labeling more robust, Roth *et al.* use a PCA based subspace representation for appearance and shape to build a reconstructive model Roth *et al.* (2005). The reconstructive model is then used to verify the foreground blobs produced by the background subtraction.

In this chapter, we propose an automatic training sample selection scheme for ISM based object detection. Our algorithm is an unsupervised learning algorithm and it does not require any hand labeled sets. We compare our training sample labeling method with the background subtraction based method Nair & Clark (2004), which only considers foreground blobs with consistent aspect ratios. Such rigid requirement makes the latter conservative in data selection, and hence it might result in a biased training set. Similarly, our method is also based on the background subtraction. However our labeler is more flexible in the sense that there is no rigid requirements of aspect ratios. On the basis of the Multiple Instance Learning, our labeler can produce a training set with lower selection bias.

4.3 Our Work

In this section we describe our algorithm for unsupervised moving object detection. The proposed algorithm consists of training and testing components. For training, our automatic labeler produces a training set, which consists of foreground blobs from background subtraction. On the basis of the produced training set, a codebook of object silhouettes can be constructed for object detection. At runtime, moving edges are detected from video frames, and object are recognized from the moving edges. Details of each component are explained in the following sections.

4.3.1 Model Learning

4.3.1.1 Automatic Training Data Collection

We present the design of our automatic labeler in this section. A labeler is essentially an object detector, in the sense that it attempts to localize objects using subwindows. Generally speaking, we need to consider two issues in the design of a labeler, namely the errors and the bias. The errors can be further divided into alignment errors and also labeling errors. An alignment error occurs when the subwindow selected by a labeler contains an object with inaccurate size of positions, whereas a labeling error occurs when the selected subwindow contains no object. As for the bias, it determines the bias of the produced training set. A labeler should try to avoid introducing any bias into the training set, otherwise it may mislead object detection. For instance, if the labeler systematically fails to collect certain types of training samples, the corresponding detector would not be able to recognize the corresponding objects. We will demonstrate below how our design copes with these two issues.

We intend to learn a shape based object detector, and therefore a set of object silhouettes is required for training. The goal of our labeler is to produce training sets automatically from given videos. Our labeler is based on background subtraction. Given a training video, background subtraction produces foreground blobs for moving objects. However background subtraction results are not always perfect. They might contain background regions if the background subtraction is not robust to shadows and occlusions. Imperfect background subtraction can result in alignment errors and labeling errors.

To handle the errors, we introduce Multiple Instance Learning (MIL) into our labeler design. In MIL, training data comes in the form of "bags" which consist of instances. Instances can be labeled as either positive or negative, and the labels of instances determine their bag label. A positive bag contains at least one positive instance, whereas a negative bag contains no positive instances. For a given bag, the labels of its instances are usually unknown although we are aware of the bag label. The advantage of MIL lies in the fact that it can handle the ambiguities in the labels of instances. In our problem, each foreground blob can be modeled as a positive bag since we assume it contains at least one foreground

object. The instances in the bag correspond to the foreground blobs of individual objects.

Our automatic labeler is based on background subtraction. Background subtraction is a common approach to identifying moving objects. the idea is to compare each video frame with a background model. Background modeling is the essence of any background subtraction method. Median filtering is one of the commonly used techniques for background modeling Cucchiara *et al.* (2003). In this thesis, a buffer is kept to store previous L frames. Each pixel in the background model is defined as the median at the same location of all frames in the buffer. In this thesis we employ the median filtering to obtain a background model and L is set to 10. The background remains fixed at run time.

Given one foreground blob from background subtraction, we apply the following heuristic to segment object candidates from the foreground blob. As the first step, a smoothed histogram of pixel number over the x-axis is computed. We can locate the crests and troughs from the histogram. Assuming the tops of objects correspond to the crests, we can segment each individual object candidate using the crests and the troughs of the histogram. The size of each bounding box is proportional to the size of the corresponding foreground object.

Figure 4.1 depicts one example for bag formation. Given a video frame in Figure 4.1(a), a foreground blob is detected and it is shown in Figure 4.1(b). The smoothed histogram in Figure 4.1(c) is computed based on the detected foreground blob. Finally the formulated bag is shown in Figure 4.1(d), where the blue rectangle indicates a positive bag, and the red rectangles indicate the instances inside the bag. In addition to positive bags, we also create negative bags in a similar way. Negative bags do not contain any foreground objects and they are created from background regions. The instances of the negative bags correspond to the cropped background regions.

Let $\mathbb{B} = \{B_1^+, B_2^+, ..., B_n^+, B_1^-, ..., B_m^-\}$ be the set of n positive bags and m negative bags. Let $B_i^+ = \{x_{i1}^+, x_{i2}^+, ..., x_{iN_i}^+\}$ be the ith positive bag, and $B_j^- = \{x_{j1}^-, x_{j2}^-, ..., x_{jM_j}^-\}$ be the the jth negative bag. Denote X^+ and X^- be the sets of positive and negative instances respectively. Essentially, our proposed labeler attempts to select positive instances $\{x_{ij}^+\}$ for training an ISM. We propose two

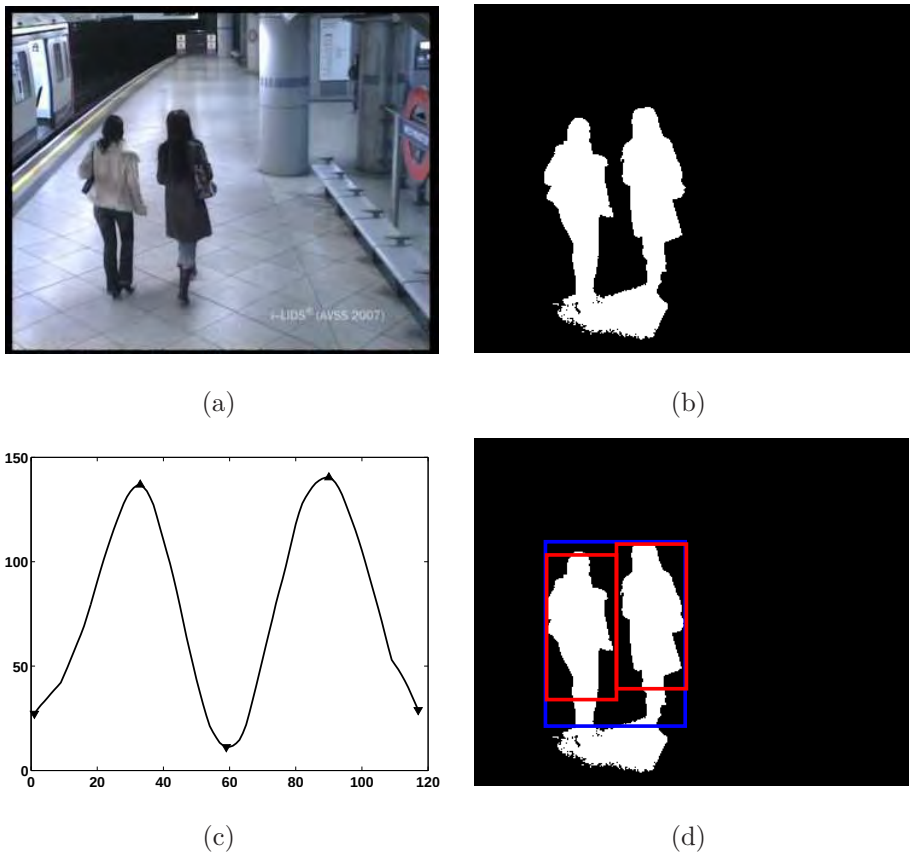


Figure 4.1: The formation of a positive bag: (a) a video frame; (b) a detected foreground blob; (c) the smoothed histogram computed on the basis of (b); (d) the blue rectangle indicates a bag, while the red rectangles indicate the instances inside the bag. The blue rectangles are generated as the smallest rectangle that covers the foreground blob, while the blue rectangles are generated using the proposed heuristic. To avoid shadows, we set the height of bounding boxes to be 80% of that of the corresponding foreground blobs.

methods for instance selections, namely the Noisy-OR model based selection and the kernel density based estimation. Details of each method are presented below.

The Noisy-OR Model based Estimation As the labels of instances are unknown, the goal of instance selection can be considered as selecting instances that have high confidence of appearing in positive bags. We favor instances which

have high probability appearing in positive bags and low probability in negative bags. Given an instance x_c, the probability can be estimated using the Noisy-OR model Maron & Lozano-Pérez (1998):

$$p(x_c|B_i^+) \propto \{1 - \prod_j [1 - p(x_c|B_{ij}^+)]\}, \tag{4.1}$$

$$p(x_c|B_j^-) \propto \prod_j [1 - p(x_c|B_{ij}^-)]. \tag{4.2}$$

Different estimates for $p(x_c|B_{ij}^+)$ and $p(x_c|B_{ij}^-)$ can be designed. As we intend to use object shape information for detection, we estimate both $p(x_c|B_{ij}^+)$ and $p(x_c|B_{ij}^-)$ using the following shape similarity:

$$p(x_c|B_{ij}^+) \propto \exp(-(D(x_c, x_{ij}^+))^2). \tag{4.3}$$

We compute $D(x_c, x_{jk}^+)$ using the distance transformation Breu *et al.* (1995). Given $p(x_c|B_i^+)$ and $p(x_c|B_i^-)$, the confidence $\text{Conf}(x_c)$ for x_c can be then measured as:

$$Conf(x_c) = \prod_{i=1}^{n} p(x_c|B_i^+) \prod_{j=1}^{m} p(x_c|B_j^-). \tag{4.4}$$

Instances with high confidence values will be selected.

The Kernel Density Based Estimation Though the Noisy-OR Model based estimation is easy to implement, it does not compute the distributions for positive and negative instances. Since both distributions can be very general, we propose to use the Kernel Density Estimator (KDE) Duda *et al.* (2001) to model the distributions.

According to the bag definition, each positive bag contains at least one positive instance while no negative bag contains positive instances. As a result, we have $x_{ik}^+ \in X^+ \cup X^-$ and $x_{jk}^- \in X^-$. The probability for a sample x_c appearing in a positive bag inside a positive bag B_{ij}^+ can be expressed in the following form:

$$p_X(x_c) = \lambda p_{X^+}(x_c) + (1 - \lambda) p_{X^-}(x_c), \tag{4.5}$$

where $p_{X^+}(x_c)$ and $p_{X^-}(x_c)$ measure the probability of x_c being a positive and a negative instance respectively, and $\lambda \in [0, 1]$ reflects the ratio between the number of positive and negative instances inside B_{ij}^+.

Though we are given the positive bags, the labels of the instances are unknown. As a result, $p_{X^+}(x_c)$ can not be evaluated directly. However we know that x_c is a positive instance, i.e., $x_c \in X^+$ if

$$p_{X^+}(x_c) > p_{X^-}(x_c). \tag{4.6}$$

The following inequality can then be obtained, using (4.5) and (4.6).

$$p_X(x_c) > p_{X^-}(x_c), \text{if } \mathrm{x_c} \in \mathrm{X}^+. \tag{4.7}$$

Both $p_X(x_c)$ and $p_{X^-}(x_c)$ can be estimated from the given positive and negative bags:

$$\begin{aligned} p_X(x_c) &\approx \frac{1}{n}\sum_{i=1}^{n}\sum_{k=1}^{N_i}\frac{1}{h}K(\frac{x_c - x_{ik}^+}{h}) \\ &\propto \frac{1}{n}\sum_{i=1}^{n}\sum_{k=1}^{N_i}\exp(-\frac{(D(x_c, x_{ik}^+))^2}{h^2}), \end{aligned} \tag{4.8}$$

$$\begin{aligned} p_{X^-}(x_c) &\approx \frac{1}{m}\sum_{j=1}^{m}\sum_{k=1}^{N_j}\frac{1}{h}K(\frac{x_c - x_{jk}^-}{h}) \\ &\propto \frac{1}{m}\sum_{j=1}^{m}\sum_{k=1}^{N_j}\exp(-\frac{(D(x_c, x_{jk}^+))^2}{h^2}), \end{aligned} \tag{4.9}$$

where $K(.)$ is the kernel density estimation, h is the size of the parzen window for the estimation, and $D(x_c, x_{jk}^+)$ is the similarity between x_c and x_{jk}^+, which can again be measured using the distance transform. Finally the confidence measure is defined as:

$$Conf(x_c) = \frac{p_X(x_c)}{p_{X^-}(x_c)}. \tag{4.10}$$

Again the instances with high confidence values will be selected.

Our automatic labeler algorithm is summarized in Algorithm 3. In comparison with the background subtraction based labeler Nair & Clark (2004), our labeler is more flexible in the sense that our instance selection is not a model based method. Our labeler can produce training sets with lower selection bias since there is no rigid requirements on the foreground aspect ratios. For example, the foreground blob in Figure 4.1(b) does not meet the aspect ratio requirement for a single

Algorithm 3 Automatic Labeler by Instance Selection

INPUTS:

$\mathbb{F} = \{F_1, F_2, ...F_R\}$ - The frames from a training video.

K - The number of instances to select from all the instances.

OUTPUTS:

A set of instances x_{ij} (object masks) for object detector training.

1: Form the set of positive and negative bags $\mathbb{B}$ using the proposed heuristic, from background subtraction results.

2: Compute the confidence measures for all the instances x_{ij}.

3: Select the top K instances x_{ij} (object silhouettes) based on the confidence measures.

pedestrian due to the imperfect detection, and therefore it is not considered by the background subtraction labeler. In contrast, our proposed labeler still takes into account of that kind of foreground blobs. By considering the diversity of foreground blobs, our produced training set may possess lower selection bias.

4.3.1.2 Codebook Learning

Given the training set produced by our automatic labeler, we can construct an Implicit Shape Model for object detection. An Implicit Shape Model is essentially a codebook of local features from the training set. The shape context descriptor Belongie *et al.* (2002) is chosen as our local feature descriptor as we want to capture object shape information. The number of radius bins and that of the angular bins are set to 5 and 12 respectively. As a result, the dimension of each shape context descriptor is 60.

The training samples are object masks. The shape context descriptors are attached to sampled points along the mask silhouettes. To construct a codebook of shapes, all the shape context descriptors are clustered into clusters using the k-means algorithm, where the χ^2 distance is chosen as the distance measure. For two K-bin histograms $g(k)$ and $h(k)$, their χ^2 distance is defined as:

$$\chi^2(g,h) = \frac{1}{2}\sum_{i=1}^{K}\frac{[g(i)-h(i)]^2}{g(i)+h(i)}. \tag{4.11}$$

The final cluster centroids correspond to the codewords in the codebook.

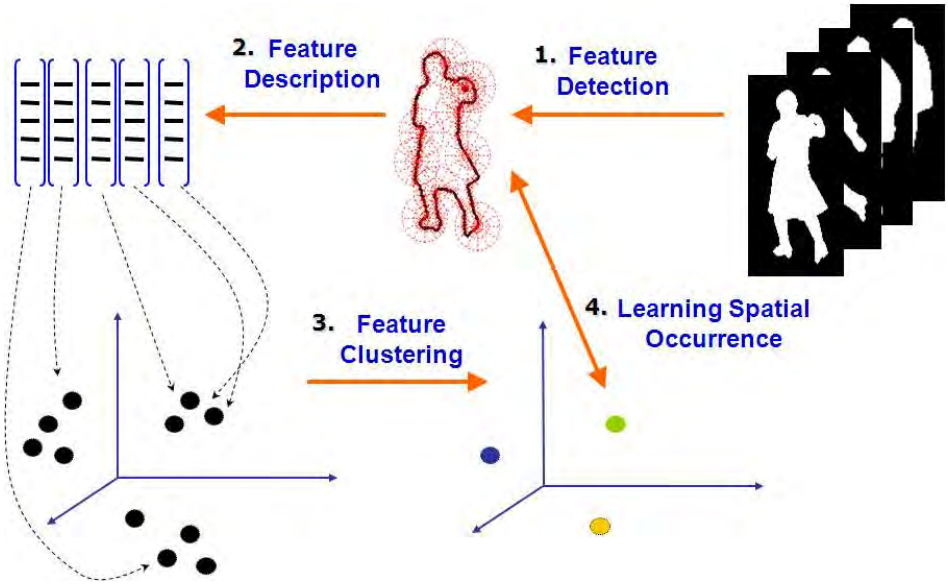


Figure 4.2: The procedure for learning a codebook of object shapes. Specifically, step 1 corresponds to the point sampling from the object silhouette, step 2 corresponds to the feature description using shape context descriptors. Step 3 and 4 correspond to the clustering using the k-means clustering.

Furthermore, the spatial occurrence distribution of each codeword with regard to object centers is also learned from the training data, for the purpose of hypotheses generation. All the collected shape context descriptors are iteratively compared with the codewords using the χ^2 distance. The pair of each descriptor and its most similar codeword is considered as a match. For each match, the relative position of the sample point to the corresponding mask center is stored. Additionally, a local patch of binary mask centered at the sampled point is also stored for each matched codeword. This local patch will be used later for segmentation purpose. The whole process of codebook learning is illustrated in Figure 4.2.

4.3.2 Object Detection

4.3.2.1 Moving Edge Detection

Rodriguez *et al.* apply an ISM for detecting pedestrians in videos Rodriguez & Shah (2007). To combine the ISM with motions, they apply the ISM on the silhouettes of foreground blobs extracted by the background subtraction. The background subtraction, however, might be unreliable and infeasible in busy urban areas. Instead of using background subtraction, we propose to apply the ISM on moving edges, which can be extracted without background subtraction. The method for moving edges are presented in the following sections.

Moving edges are obtained by comparing the edges identified in consecutive frames from testing videos. Specifically, we apply the Canny edge detector Canny (1986) to extract edges from each frame. Subsequently, we perform edge subtraction between edge maps from consecutive frames. For example, given the $(i-1)$th edge map, we only keep the edge pixels that do not exist in the same locations at the $(i-1)$th edge map.

Denote two consecutive edge maps as $E_i=\{(x_j, y_j)\}$ and $E_{i-1}=\{(x_k, y_k)\}$. The moving edge map E_m is described as $E_m = E_i \setminus E_{i-1}$. We also perform morphological operations to remove isolated pixels in E_m.

4.3.2.2 Hypotheses Generation

To generate the hypotheses for moving objects at runtime, we apply the ISM to the moving edges. We sample points from the detected moving edges, and then attach shape context descriptors to the sampled points. The obtained descriptors are used to match against the codewords, using the χ^2 distance as the dissimilarity measure. A match is said to be found if the distance is below a threshold. Once a match is found, the corresponding codeword would cast votes for hypotheses centers using the stored relative positional information, which is recorded during the model learning in Section 4.3.1.2. Consistent hypotheses configurations can be obtained by aggregating the votes using the generalized Hough Transform Leibe *et al.* (2008), followed by figure ground segmentation. Details for hypotheses generation are presented below.

Vote Aggregation Denote f_i as the shape context descriptor for the ith sampled point from location l_{f_i} at the current frame. A set of valid interpretation $\mathcal{I} = \{I_1, I_2, ..., I_n\}$ can be obtained by matching f_i with the codebook, and each interpretation I_j is weighted by $p(I_j|f_i, l_{f_i})$. As a result, the chance of observing object o at location x can be expressed as follows:

$$p(o, x|f_i, l_{f_i}) = \sum_j p(o, x|I_j, f_i, l_{f_i}) p(I_j|f_i, l_{f_i}). \tag{4.12}$$

Since f_i can be replaced by each valid interpretation I_j, $p(o, x|f_i, I_j, l_{f_i})$ can be simplified as $p(o, x|I_j, l_{f_i})$. Furthermore, the match between f_i and the codebook is independent of its location l_{f_i}, and therefore (4.12) can be simplified as:

$$\begin{aligned} p(o, x|f_i, l_{f_i}) &= \sum_j p(o, x|I_j, f_i, l_{f_i}) p(I_j|f_i, l_{f_i}) \\ &= \sum_j p(x|o, I_j, l_{f_i}) p(o|I_j, l_{f_i}) p(I_j|f_i) \end{aligned} \tag{4.13}$$

where $p(x|o, I_j, l_{f_i})$ represents the generalized Hough vote for an object, $p(o|I_j, l_{f_i})$ specifies the extent to which the shape interpretation matches the object silhouette, and $p(I_j|f_i)$ reflects the similarity between the f_i and I_j.

On the basis of the above derivation, the score of an object o detected at location x can be computed by marginalizing all the f_i that contributes to it, and consequently we have the following marginalization:

$$p(o, x) = \sum_i p(o, x|f_i, l_{f_i}) p(f_i, l_{f_i}), \tag{4.14}$$

where $p(f_i, l_{f_i})$ can be regarded as a function indicating whether or not a sample point is selected from location l_{f_i} of the moving edges.

After calculating the scores for all hypotheses, the search for local maximas in the voting space is conducted to find promising hypothesis locations. In order to avoid the quantization artifacts, Mean-Shift algorithm Comaniciu & Meer (2002) is employed for seeking local maximas in a continuous voting space.

Figure ground Segmentation Once we obtain the object hypotheses $h = (o, x)$, we can segment the figures out of the background under a probabilistic

framework. In other words, we estimate the per-pixel probability of being *figure* or *background*. Following the same notation from the previous section, the probability can be obtained by the following marginalization:

$$p(\mathbf{p} = figure|o, x) = \sum_{\mathbf{p}\in(l)} p(\mathbf{p} = figure|o, x, f_i, l_{f_i})p(f_i, l_{f_i}|o, x), \tag{4.15}$$

which can be expanded as

$$\begin{aligned} & p(\mathbf{p} = figure|o, x) \\ = & \sum_{\mathbf{p}\in(l)} \sum_{j} p(\mathbf{p} = figure|o, x, f_i, I_i, l_{f_i})p(f_i, l_{f_i}|o, x) \\ = & \sum_{\mathbf{p}\in(l)} \sum_{j} p(\mathbf{p} = figure|o, x, I_i, l_{f_i})p(f_i, l_{f_i}|o, x), \end{aligned} \tag{4.16}$$

where $p(\mathbf{p} = figure|o, x, I_j, l_{f_i})$ can be interpreted as the support from the local mask patch recorded in codeword I_i at the location l_{f_i} relative to the object center x. And $p(f_i, l_{f_i}|o, x)$ can be expressed as

$$p(f_i, l_{f_i}|o, x) = \frac{p(o, x, |f_i, l_{f_i})p(f_i|l_{f_i})}{p(o, x)}, \tag{4.17}$$

where $p(o, x, |f_i, l_{f_i})$ can be obtained from the vote aggregation.

Final segmentations can be computed based on the likelihood ratio between $figure$ and *background*. In our implementation, figure ground segmentation is achieved by backtracking the matching results. In other words, the recorded training image patches in the codebook are back-projected to the corresponding image locations. The bounding box for each pedestrian hypothesis is generated by aggregating all the foreground segmentations of the corresponding hypothesis.

4.4 Experiments

Experiment Setup We conduct the experiments using three datasets. They contain videos from both outdoor and indoor surveillance. We use these videos to evaluate individual algorithm components as well as the whole algorithm.

Three video datasets are used in our experiments, namely the PETS06 set, the Visor set, and the iLIDS set. The PETS06 set PETS2006 (2006) contains

pedestrian sequences recorded in a railway station which include different viewing angles and a certain degrees of occlusions. We use both the side-view and front-view sequences in the experiments. The Visor set Visor (2010) is developed by the Imagelab group of the University of Modena and Reggio Emilia. It contains a large set of multimedia data and annotations. Some outdoor surveillance videos are selected for evaluation. The iLIDS set iLIDS (2007) is collected from the Image library for intelligent detection systems (i-LIDS), which is a benchmark for video analytics systems. The videos we choose contain pedestrian footages recorded in a railway station by a front-mounted camera.

Evaluation Metric The output of our object detector is a set of bounding boxes for pedestrian hypotheses. Each bounding box is a minimum bounding box that confines the corresponding hypothesis. Ground truth bounding boxes are manually defined for each video. We choose the bounding boxes for evaluation because they can easily describe the precise locations of pedestrians. Furthermore, it is easier to obtain the ground truth.

To evaluate the object hypotheses, we follow the evaluation criteria employed in Leibe *et al.* (2008) that covers three categories, and they are relative distance, cover, and overlap. The relative distance measures the distance between the center of a bounding box and that of the ground truth. The cover and overlap measure how much area of the ground truth bounding box is covered by a detection hypothesis, and vice versa. A hypothesis is classified as a true positive if the relative distance ≤ 0.5 and both cover and overlap are above 50%. Examples of hypotheses can be found in the representative sample frames in the following sections.

4.4.1 Moving Edge Based Object Detection

As our first experiment, we evaluate the proposed moving edge based object detection approach using the PETS2006 benchmark dataset. The number of pedestrians in each sequence can be found in Table 4.1. The number of frames in each sequence ranges from 80 to 100. Each image frame is downsampled to 360 by 288 pixels. In this experiment, we manually create the training set which

Table 4.1: #Pedestrians in different sequences

Sequences	1	2	3
Side-view	566	189	286
Front-view	148	283	461

contains 25 binary masks of pedestrian images in front view and side view. It is noted that the training samples are independent of the testing images. Using this training set, we train an ISM using pedestrian silhouettes. At runtime, the ISM is applied to the moving edges extracted from the testing sequences.

Side-view Sequences For the side-view sequences, we compare the performance of our approach with the ISM trained by Leibe *et al.* Leibe *et al.* (2008) [1]. Both ISMs are trained using pedestrian silhouettes. The size of training sets for both methods are presented in Table 4.2. It can be seen from Table 4.2 that our training set is much smaller than theirs. At runtime, the ISM by Leibe *et al.* generates the pedestrian hypotheses based on all the extracted edges from each frame, however our hypotheses are generated on the moving edges only. Obviously our searching space is smaller as we only consider the moving edges.

Table 4.2: The comparison of training set

Methods	#Training Images	#Codewords
poposed	25	700
Leibe *et al.*	210	7475

The Precision Recall curves are presented in Figure 4.3, and sample detection outputs are shown in Figure 4.4. We can see that the proposed moving edge based object detection achieves better and comparable performance compared to the traditional ISM Leibe *et al.* (2008). Our advantage lies in a smaller searching space for hypotheses generation. As a result, our approach has lower

[1] Available from http://www.vision.ee.ethz.ch/bleibe/code/

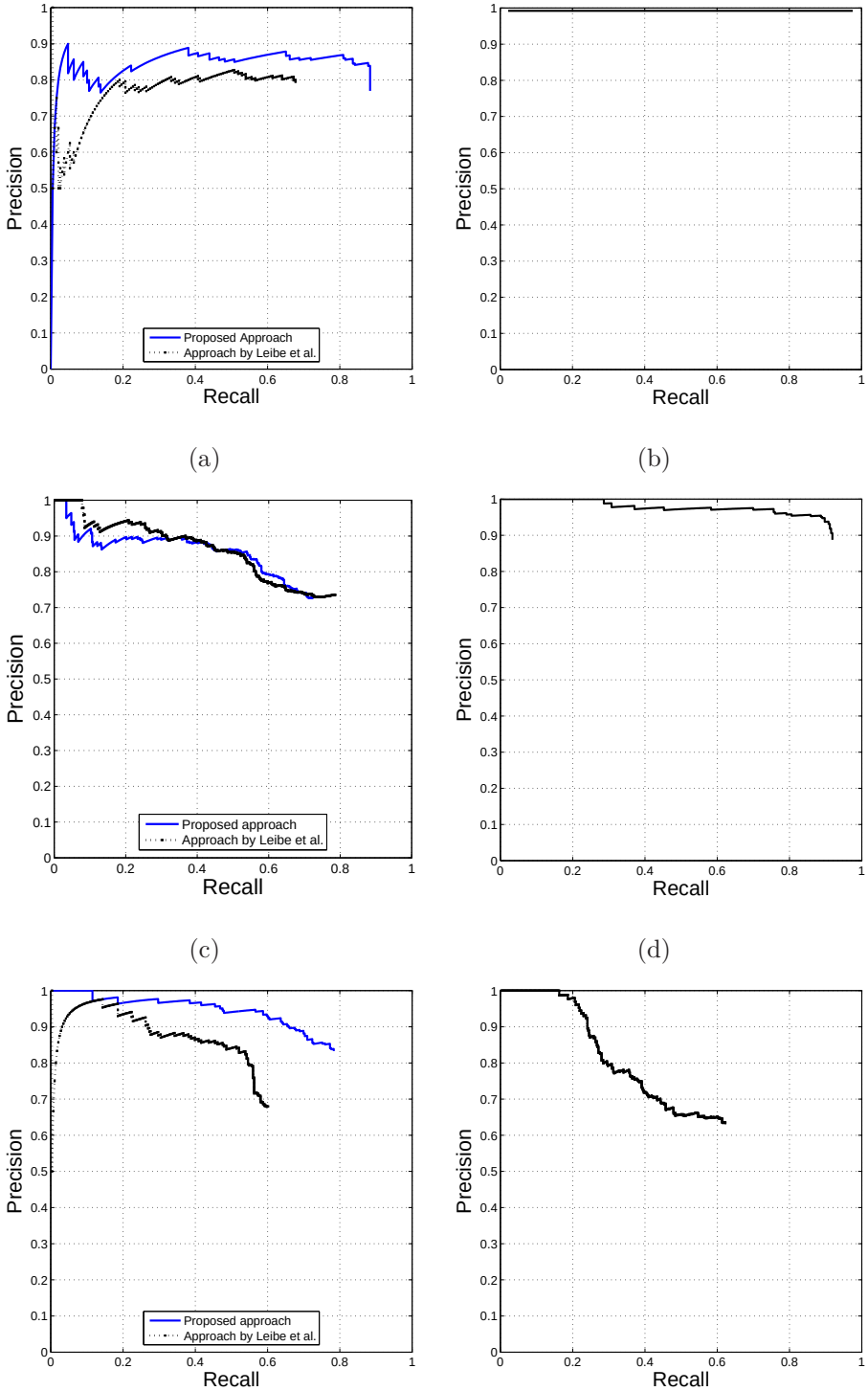


Figure 4.3: The RPC curves for both the side-view and front-view sequences. The curves on the left are from side-view sequences, whereas the curves on the right are from the front-view sequences.

computational cost, i.e., the averaged processing time per frame needed by the unoptimized Matlab implementation of our approach is 7.21 seconds, while the C++ implementation of Leibe *et al.* (2008) requires 7.27 seconds.

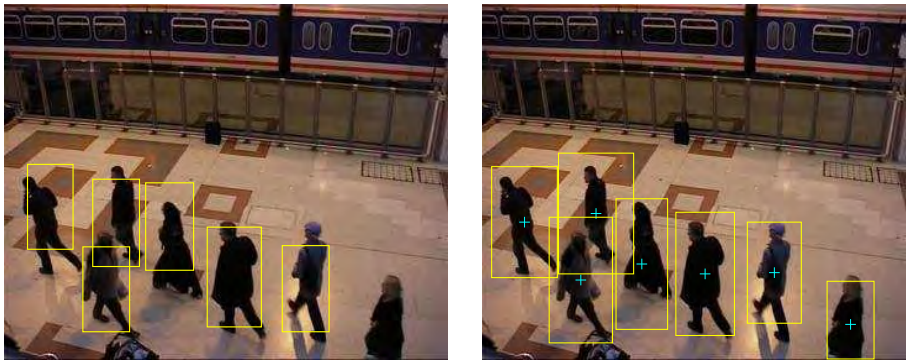
Front-view Sequences In this experiment, we do not use the approach in Leibe *et al.* (2008) for comparison as it is trained using side view images only. The related Precision Recall curves are depicted in Figure 4.3, and sample outputs are shown in Figure 4.5. We can see that the proposed approach is able to reliably detect moving pedestrians. It is noted that the precision is relatively lower for Sequence 3 due to the heavy occlusions between pedestrians.

4.4.2 Automatic Training Set Generation

In the previous experiment, we train object detectors using manual labeled sets. In our second experiment, we will train object detectors using training sets produced by the automatic labeler. The purpose of this experiment is to compare the two different instance selection schemes for automatic labeling, namely the Noisy-OR model based selection and the Kernel Density based selection.

Given a training sequence, training sets are produced using different selection schemes. At runtime, the learned object detectors detect objects from the moving edges. We use six sequences from the Visor Visor (2010) data set for evaluation, whose lengths range from 400 to 500 frames. For each frame, 50 frames are used for training, and the remaining frames are used for testing. Both selection schemes select 10 instances from the training frames.

The Precision Recall curves are shown in Figure 4.6, and sample outputs are demonstrated in Figure 4.7. It can be observed that the training sets produced by both schemes result in detectors with similar performance, but the detector trained using kernel density based selection scheme performs better on average. As a result, we will use the kernel density selection scheme for training set generation in the following experiments.



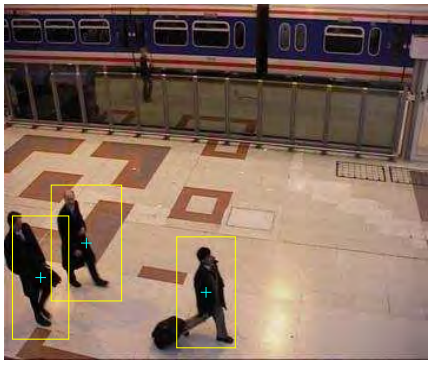
(a)

(b)

(c)

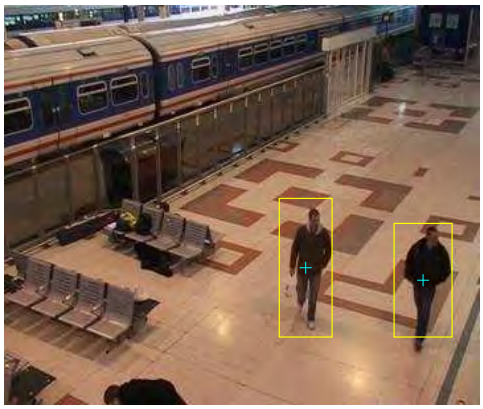
(d)

(e)



(f)

Figure 4.4: Sample outputs from both approaches on the side-view sequences. The outputs on the left are produced by the proposed approach, whereas outputs on the right are produced by the method of Leibe *et al.* .



(a) Front-view Sequence1

(b) Front-view Sequence 2

(c) Front-view Sequence 3

Figure 4.5: Detection results achieved by the proposed approach on the front-view sequences.

4.4.3 Unsupervised Object Detection

As the last experiment, we compare the proposed unsupervised object detection algorithm with the background subtraction based algorithm Nair & Clark (2004). Two data sets are used for evaluation in this experiments, namely the PETS06 set, and the iLIDS iLIDS (2007) set. Given a video, our automatic labeler will produce a training set for the object detector (ISM) using the first F frames, where F is a predefined value. The learned object detector will then conduct object detection on the remaining video frames. The object detection are also based on the extracted moving edges.

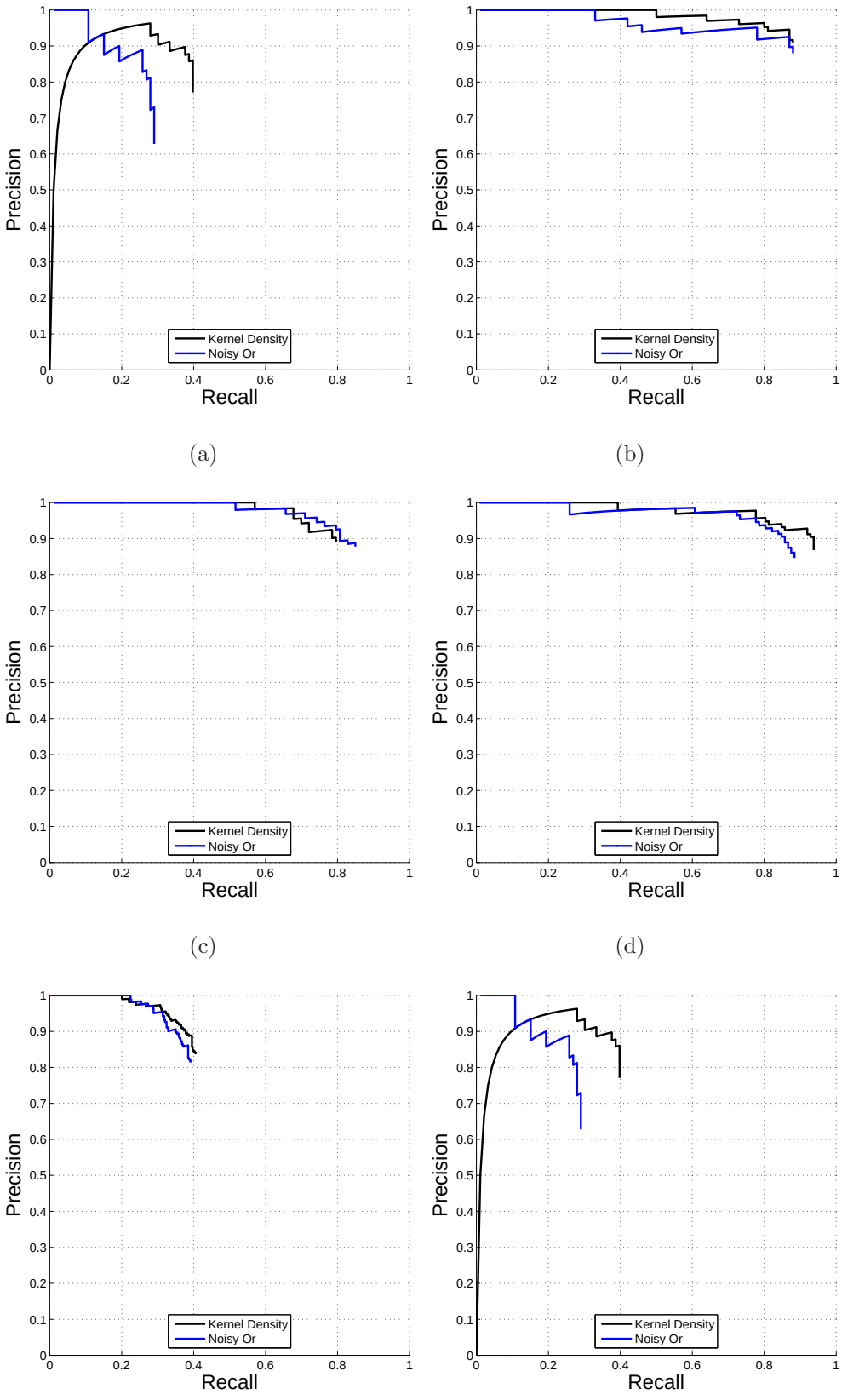
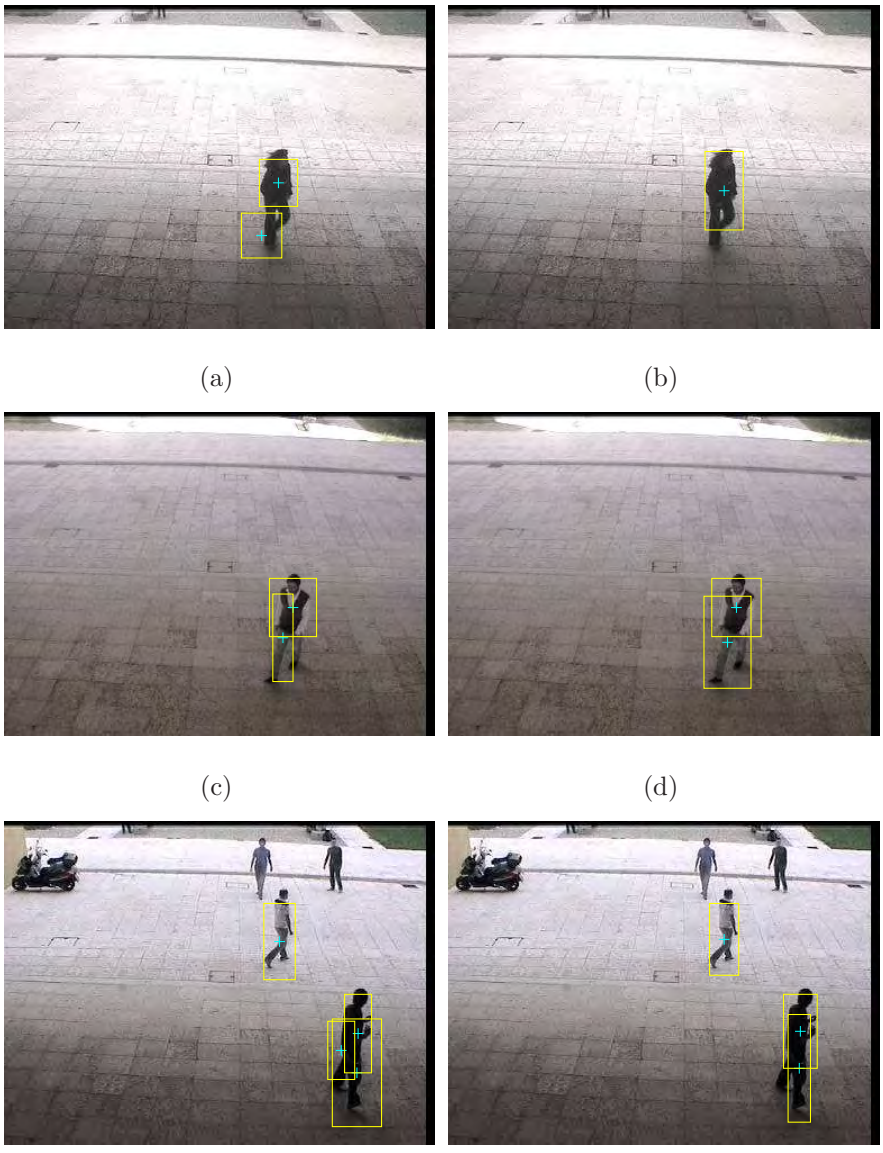


Figure 4.6: The performance comparison of two instance selection schemes on the Visor dataset.



(a) (b)

(c) (d)

(e) (f)

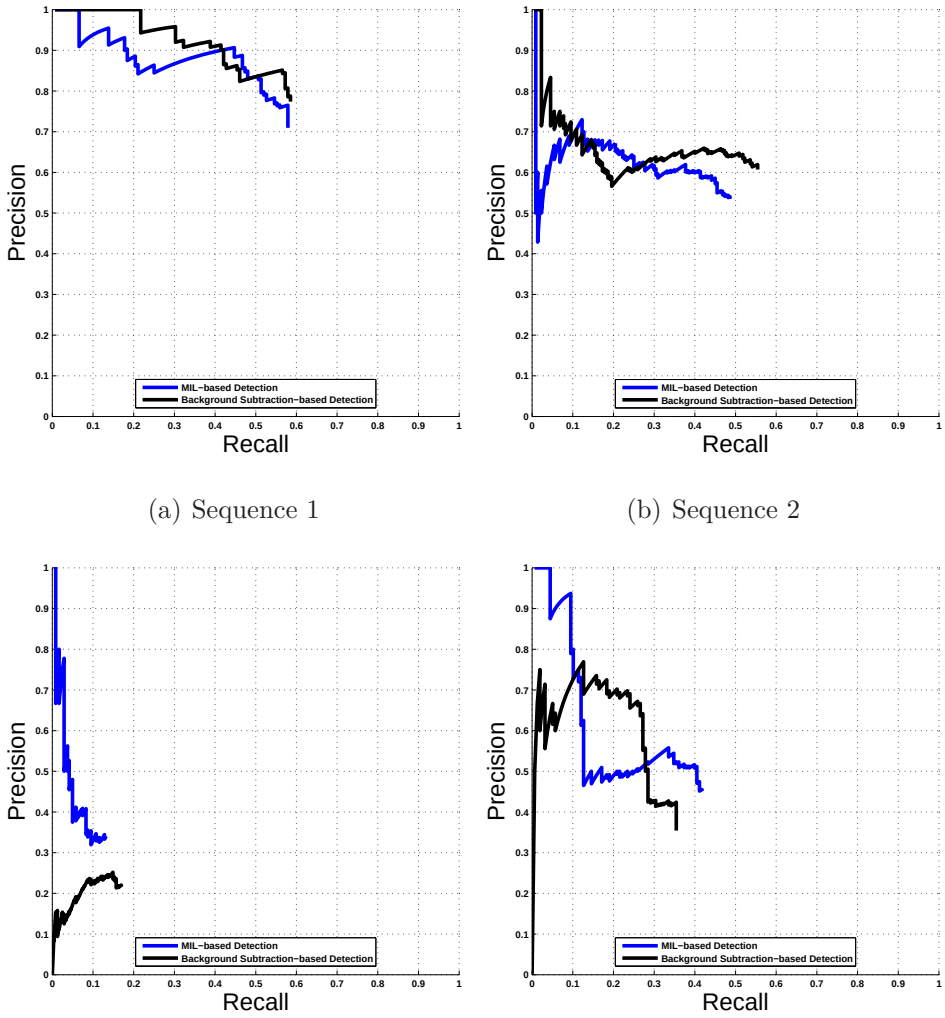
Figure 4.7: Sample outputs from the detectors trained by different instance selection schemes on the Visor dataset. The outputs on the left are produced by the Noisy-OR model based selection scheme, whereas the outputs on the right are produced by the Kernel density based selection scheme.

The PETS06 Set We use four side-view sequences from the PETS06 set, whose lengths range from 150 to 160 frames. The number of training frames F is set to 50. The number of instances K to select from the training frames by our proposed labeler is empirically set to 10.

The Precision Recall curves for the PETS06 sequences are depicted in Figure 4.8, and sample results are shown in Figure 4.9. It can be seen that the proposed algorithm achieves comparable or better performance compared to the background subtraction based algorithm on sequence 1, 2 and 4. This may due to the fact that there is no occlusions in the training frames from sequences 1,2 and 4. In other words, the distributions for the foreground blobs in these sequences may be unimodal distributions. As a result, the training sets produced by the background subtraction based labeler do not have high bias. On the other hand, the performance of our algorithm is superior than that of its counterpart on sequence 3. This should due to the heavy occlusions in sequence 3. In consequence, the background subtraction based labeler produces a biased training set, as it only considers foreground blobs that meet the aspect ratio of an individual object. In contrast, our framework considers all kinds of foreground blobs, and hence it works better under different situations.

The iLIDS set We extract three sequences from the PETS06 set, whose lengths range from 150 to 160 frames. The number of training frames F is also set to 50. The number of instances K to select from the training frames is empirically set to 10.

The Precision Recall curves and sample outputs are presented in Figure 4.10, and sample results are shown in Figure 4.11 respectively. Our proposed method performs better than the background subtraction based method on this dataset. The reason might due to the fact that there are occlusions in the training frames of this dataset. As a result, background subtraction based method produces a bias training set, whereas our method still work well with a lower bias.



(a) Sequence 1

(b) Sequence 2

(c) Sequence 3

(d) Sequence 4

Figure 4.8: Object detection performance comparison on the PETS side-view sequences.

4.5 Conclusions

In this chapter we have presented an unsupervised learning algorithm for moving object detection, which is an extension to the Implicit Shape Model(ISM). Our contributions are two-fold: (i) two Multiple Instance Learning based automatic labeling algorithms for training set generation. (ii) an moving edge detection

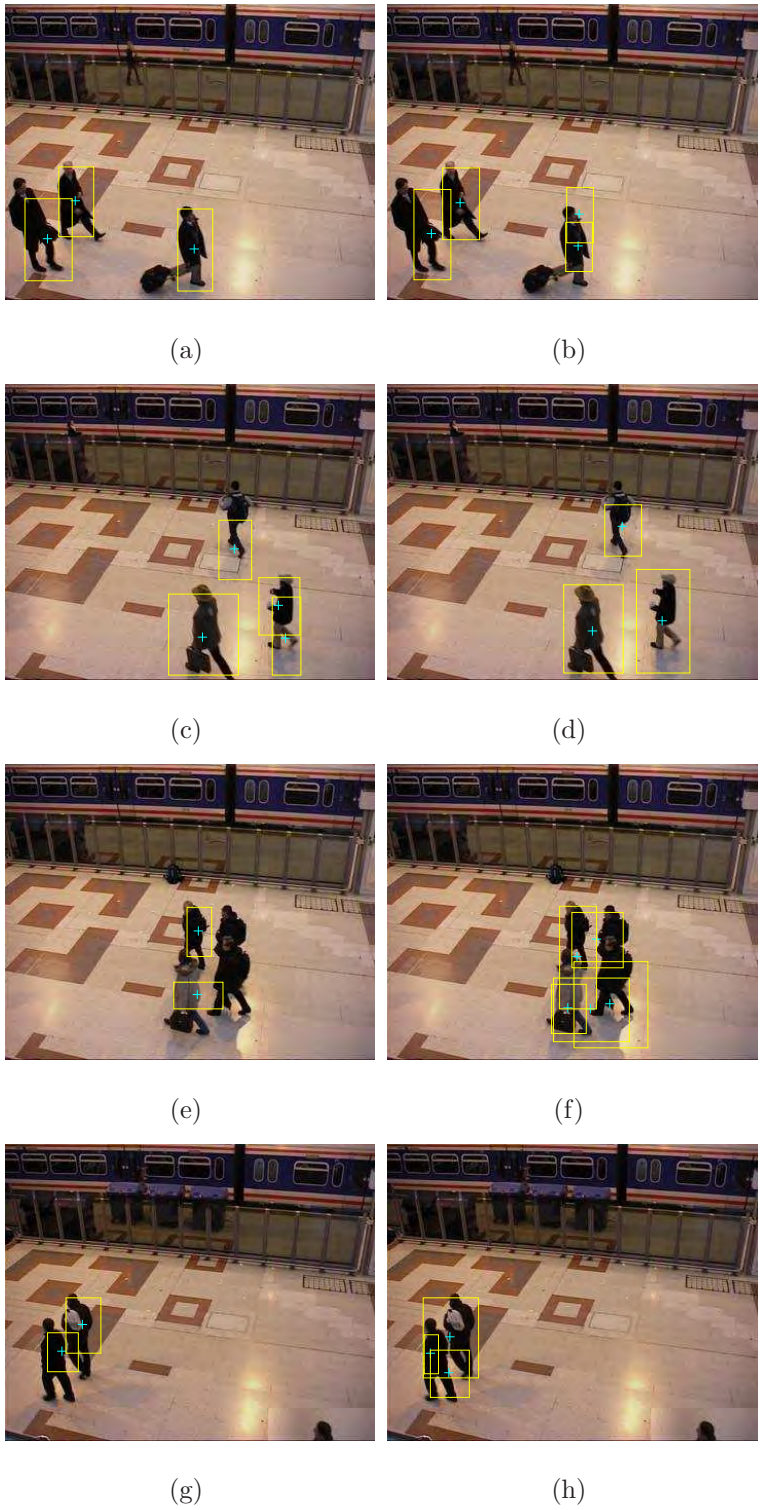
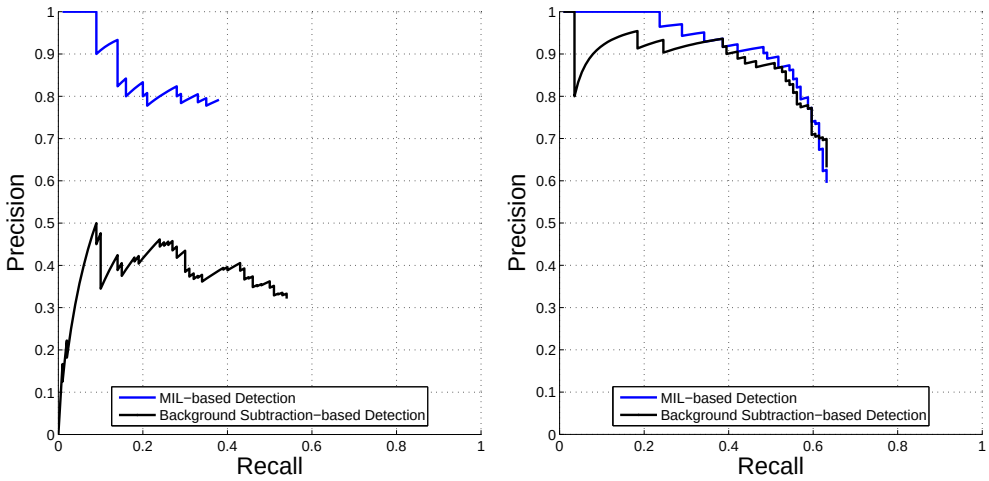
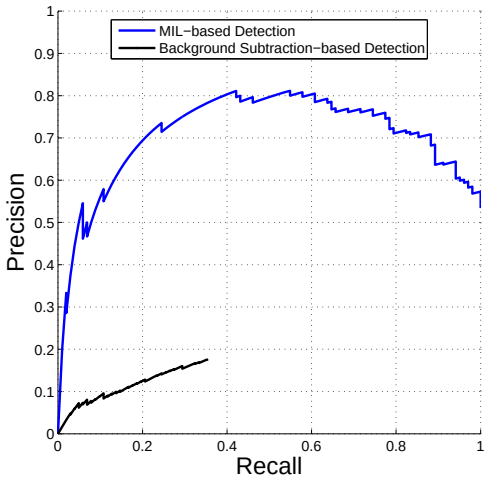


Figure 4.9: Sample outputs from both approaches on the PETS2006 dataset. The outputs on the left are produced by the proposed approach, whereas the outputs on the right are produced by the method in Nair & Clark (2004).



(a) Sequence 1

(b) Sequence 2



(c) Sequence 3

Figure 4.10: Object detection performance comparison on the iLIDS sequences.

scheme for object detection. In our algorithm, the automatic labeler produces training sets for the ISM. A set of ISMs is then learned using the produced training sets. At runtime, moving edges are extracted from videos, and then object detection is achieved by applying the ISM to the moving edges. In addition to the experiments that evaluate different algorithm components, we also compare the proposed algorithm with a background subtraction based moving object

(a) (b)

(c) (d)

(e) (f)

Figure 4.11: Sample outputs from both approaches on the iLIDS dataset. The outputs on the left are produced by the proposed approach, whereas the outputs on the right are produced by the method in Nair & Clark (2004). The red rectangles indicate the region of interest for object detection.

detection algorithm. The experimental results demonstrate that the proposed algorithm achieves comparable performance compared to the background subtraction based counterpart, and it even outperforms the counterpart at complex situations.

Chapter 5

Unsupervised On-line Learning for Codebook based Visual Detection

5.1 Introduction

In this chapter we study again the codebook based object detection task. We also focus on the case of moving object detection. In comparison with the previous chapter, this chapter extends the previous chapter for on-line learning.

Moving object detection in videos is an important task in many computer vision applications. Numerous approaches have been proposed in the literature for object detection. Early researchers employ motion segmentation algorithms to detect moving objects Stauffer & Grimson (1999), Khan & Shah (2001), Wang & Ji (2005), Han *et al.* (2006). These algorithms assume that a compact region with different motion from the background is most likely to be a moving object. Moving pixels are clustered into layers with consistent motions. Without any high-level object information, motion segmentation based methods can be affected by background motions. To improve the performance, different supervised object modeling methods are proposed. The idea is to incorporate high-level

object information into low-level image models by human supervision. These supervised methods can be divided into global methods and local methods. Global methods construct global models for objects, and object detection is achieved by examining image features over locations and scales Dalai *et al.* (2005), Munder & Gavrila (2006). On the other hand, local methods model each individual object as a set of parts Bouchard *et al.* (2005), Andriluka *et al.* (2008), Gall & Lempitsky (2009), Okada (2009). Specifically, parts can be modeled either generatively Bouchard *et al.* (2005), Andriluka *et al.* (2008), Leibe *et al.* (2008), or discriminatively Felzenszwalb *et al.* (2008), Okada (2009). At runtime, part responses are consolidated to form the complete object hypothesis. Part based approaches have been demonstrated to possess considerable tolerance to partial occlusions. Among different part based object detectors, codebook based detector is a popular detector due to its simplicity and effectiveness.

All these supervised object models have been shown to perform well on object detection tasks. However they have the following two drawbacks: 1) Manual labeling. A detector usually benefits from a large training set as it provides more diversity, but manual labeling of large training sets is time-consuming and tedious. It is important to keep human effort to a practical level, so that the detection algorithms can scale well for problems with larger amount of data. For example, when training object detectors for a video surveillance network with different views from numerous cameras, it is highly undesirable to obtain training data by hand. 2) Off-line learning. All the learning is conducted in an off-line manner. As a result, the detector remains fixed after the training, and hence it cannot adapt to new data. However, in some detection tasks, on-line learning is more desirable because it enables the detector to adapt to new environments. For example, an adaptive object detector is favored in video surveillance networks, which assures the portability of detectors from one scene to another.

In this chapter, we propose an unsupervised on-line detection framework that overcomes the above drawbacks. Specifically, our contributions include an on-line labeler and an on-line object detector. The proposed labeler enables the on-line data collection. As for the on-line object detection, we propose a codebook based object detector which can adapt its codebook in an on-line manner. As shown in Figure 5.1, the training phase and the testing phase are not separated in this

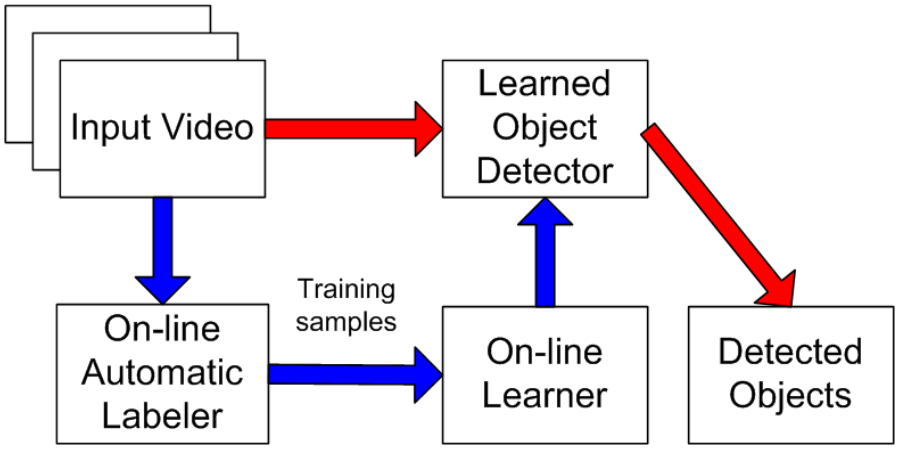


Figure 5.1: The flowchart of the proposed framework. Blue arrows indicate the training phase, while red arrows imply the testing phase. Both phases are not separated.

framework. As a result, incoming video sequences serve for the purpose of both training and testing.

The rest of this chapter is organized as follows: the related work to our framework is reviewed in Section 5.3, and the proposed work is described in Section 5.3, followed by the experimental results in Section 5.4. Finally the conclusions are summarized in Section 5.5.

5.2 Related Work

Motion plays an important role in moving object detection. Early researchers employ the background subtraction technique Stauffer & Grimson (1999), Elgammal *et al.* (2000) to capture moving objects. Foreground pixels are determined by the difference between the current scene and the background scene. Subsequently connected foreground pixels are grouped into foreground blobs. The key issue of this method is that it is completely low-level data driven, i.e., no high-level object information is used. As a result, both the foreground detection and the motion grouping are conducted in an ad-hoc manner that might not work well under complex situations. We call such methods as the no-supervision-and-low-performance methods. In comparison with the background subtraction, the strength of dif-

ferent object models lies in the incorporated high-level object information by human supervision. With the high-level object information, this method usually outperforms the background subtraction on object detection. This kind of supervised methods can be considered as the with-supervision-and-good-performance methods. By taking the advantages of the aforementioned methods, we propose a no-supervision-good-performance method to bridge the gap between the no-supervision-and-low-performance methods, and the with-supervision-and-good-performance methods. The key is the removal of human supervision for detector training. Several attempts have been made in the literature, and they can be categorized into two classes, namely semi-supervised and unsupervised learning methods. Both methods are reviewed in the following sections.

5.2.1 Semi-supervised Learning Methods

Semi-supervised learning methods are characterized by a small hand labeled set for initialization. A large training set can be obtained by different techniques using the small labeled set. One example of the semi-supervised learning is the co-training method, whose details can be found in Section 4.2.

5.2.2 Unsupervised Learning Methods

In the semi-supervised learning methods, human supervision (labeling) is still required. To overcome this limitation, unsupervised learning methods are proposed. Nair *et al.* employ a motion based object detector as the automatic labeler in Nair & Clark (2004). The idea is to employ the background subtraction to automatically label training examples for a pedestrian detector. Background subtraction, as mentioned above, can be affected by shadows, reflections, and illumination changes.

To improve the background subtraction labeler, Roth *et al.* use a PCA based reconstructive model Roth *et al.* (2005), to verify the motion detection results from background subtraction. The output of the background subtraction is used to build a reconstructive representation incrementally. To further improve the

robustness, multiple cues (shape and appearance) are used. By using the reconstructive model, false detections such as shadows and backgrounds would be excluded from the training set. Such learning is very conservative since only highly consistent motion blobs could be accepted by the model. Similarly, Ikizler-Cinbis *et al.* select training samples using the following logistic regression model in Ikizler-Cinbis *et al.* (2009):

$$p(y = \pm, x, w) = \frac{1}{1 + \exp(-y(w^T x))}, \tag{5.1}$$

where y is a class label, x is a feature vector concatenated with 1 for the bias term, and w is a weight vector. Training samples are selected if $p(y|x) > \tau_I (\tau_I \in [0, 1])$ is satisfied, and τ_I is set to as high as possible (0.96) to avoid introducing false positives for learning. The high threshold τ_I would result in a highly conservative labeler. Both selection methods are considered as conservative learning as they are both model based selections, and only the training samples that are highly consistent with the models are selected. Though the conservative learning guarantees the correctness of training samples, it has a bias to select samples that fit the model well.

Similar to the previous chapter, our labeler is also an unsupervised labeler, which is based on the Multiple Instance Learning. The difference is that the labeler can process on-line selection. In comparison with the conservative learning methods, our labeler selects training samples using density estimation rather than model fitting, and therefore the produced training set might has lower bias.

5.3 Our Work

In this section, we present the details of our unsupervised on-line learning algorithm for codebook based moving object detection.

5.3.1 Task Description

Given a video sequence, our learning task is to train an object detector in an unsupervised and on-line manner. The detector should output a bounding box for each visible object in the region of interest.

(a) (b)

Figure 5.2: The formation of a positive bag. (a) An image frame. (b) A detected foreground and its bag formulation. The blue rectangle corresponds to the foreground blob, while the red rectangles correspond to the instances inside the corresponding bag. The blue rectangles are generated as the smallest rectangles that covers the foreground blobs, while the blue rectangles are generated using the proposed heuristic.

5.3.2 The On-line Automatic Labeler

We intend to train a shape-based object detector, and therefore the output of the labeler is a set of object silhouettes. Following our design in the previous chapter, our on-line labeler is also based on background subtraction. Multiple Instance Learning (MIL) is introduced to handle possible alignment and labeling errors from background subtraction.

Assuming each foreground blob contains at least one foreground object, we model each blob as a positive bag inside which each instance correspond to an object candidate (See Figure 5.2). The object candidates are obtained by using the heuristic detailed in Section 4.3.1.1. The negative bags are formed in a similar way.

Let $\mathbb{B} = \{B_1^+, B_2^+, ..., B_n^+, B_1^-, ..., B_m^-\}$ be the set of n positive bags and m negative bags. Let $B_i^+ = \{x_{i1}^+, x_{i2}^+, ..., x_{iN_i}^+\}$ be the ith positive bag, and $B_j^- = \{x_{j1}^-, x_{j2}^-, ..., x_{jM_j}^-\}$ be the the jth negative bag. Denote X^+ and X^- be the sets of positive and negative instances respectively. Given $\mathbb{B}$, we employ the kernel density estimation to select instances. According to the MIL bag definition, each

positive bag contains at least one positive instance while no negative bag contains positive instance. As a result, we have $x_{ik}^{+} \in X^{+} \cup X^{-}$ and $x_{jk}^{-} \in X^{-}$. For the instance selection, we favor instances x_c that have high confidence values:

$$Conf(x_c) = \frac{p_X(x_c)}{p_{X^-}(x_c)}, \tag{5.2}$$

where $p_X(x_c)$ and $p_{X^-}(x_c)$ can be estimated from the given positive and negative bags:

$$\begin{aligned} p_X(x_c) &\approx \frac{1}{n}\sum_{i=1}^{n}\sum_{k=1}^{N_i}\frac{1}{h}K(\frac{x_c - x_{ik}^{+}}{h}) \\ &\propto \frac{1}{n}\sum_{i=1}^{n}\sum_{k=1}^{N_i}\exp(-\frac{(D(x_c, x_{ik}^{+}))^2}{h^2}), \end{aligned} \tag{5.3}$$

$$\begin{aligned} p_{X^-}(x_c) &\approx \frac{1}{m}\sum_{j=1}^{m}\sum_{k=1}^{M_j}\frac{1}{h}K(\frac{x_c - x_{jk}^{-}}{h}) \\ &\propto \frac{1}{m}\sum_{j=1}^{m}\sum_{k=1}^{M_j}\exp(-\frac{(D(x_c, x_{jk}^{+}))^2}{h^2}), \end{aligned} \tag{5.4}$$

where $K(.)$ is the kernel density estimation, h is the size of the parzen window for the estimation, and $D(x_c, x_{jk}^{+})$ is the similarity between x_c and x_{jk}^{+}. Details of the derivation of $p_X(x_c)$ and $p_{X^-}(x_c)$ can be found in Section 4.3.1.1.

The proposed kernel density estimation based instance selection scheme in 4.3.1.1 is a batch algorithm, which requires all data be available for selection. However the on-line detection setting does not meet this requirement. To enable on-line learning, one naive solution is separate a video sequence into segments, and then apply the algorithm to each segment. Despite of its feasibility, this solution ignores the relationship between different segments, and hence it is not optimal. The key to the on-line learning is the modeling of data streams. Mixture models have been employed to model the data streams for on-line learning Zhou *et al.* (2003), Heinz & Seeger (2006), Stauffer & Grimson (2002). On the basis of the mixture models, we propose an on-line algorithm to realize the on-line instance selection. It is assumed that the distribution of positive instances is dynamic, while the distribution of negative instances is stable. To model both distributions,

we keep a set of key instances for positive and negative data respectively. Due to the dynamic nature of the positive instance distribution, we only need to update the set of key positive instances. Denote $I_t^+ = \{x_1^+, x_2^+, ..., x_p^+\}$ as the key positive instance set at time t, and $I^- = \{x_1^-, x_2^-, ..., x_q^{(-)}\}$ be the key negative set. Both sets can be initialized using the batch instance selection algorithm. We also keep a set of weights for the positive key instances $W_t = \{w_1, w_2, ..., w_p\}$. Given the bag collection $\mathbb{B}_t = \{B_{t1}^+, B_{t2}^+, ..., B_{tn}^+, B_{t1}^-, ..., B_{tm}^-\}$ at time slice t, our on-line learning algorithm consists of two steps, namely the instance selection and the key instance update.

For the instance selection, we compute a confidence value for each $x_c \in B_i$ using I_t^+ and I^-. The confidence value is defined as:

$$Conf(x_c) = \frac{p_X(x_c)}{p_{X^-}(x_c)}, \tag{5.5}$$

where $p_X(x_c)$ and $p_{X^-}(x_c)$ can be estimated using the I_t^+ and I^- :

$$\begin{aligned} p_X(x_c) &\approx \frac{1}{p}\sum_{i=1}^{p} w_i \frac{1}{h} K(\frac{x_c - x_i^+}{h}) \\ &\propto \frac{1}{p}\sum_{i=1}^{p} w_i \exp(-\frac{(D(x_c, x_i^+))^2}{h^2}), \end{aligned} \tag{5.6}$$

$$\begin{aligned} p_{X^-}(x_c) &\approx \frac{1}{q}\sum_{j=1}^{q} \frac{1}{h} K(\frac{x_c - x_j^-}{h}) \\ &\propto \frac{1}{q}\sum_{j=1}^{q} \exp(-\frac{(D(x_c, x_j^-))^2}{h^2}). \end{aligned} \tag{5.7}$$

where $K(.)$ is the kernel density estimation, h is the size of the parzen window for the estimation, and $D(x_c, x_i^+)$ is the similarity between x_c and x_j^+, which can be measured using distance transform. On the basis of the confidence values, the top N_s instances are selected. We denote $I_t^s = \{x_1^s, x_2^s, ..., x_{N_s}^s\}$ as the set of selected instances.

In the key instance update step, every $x_i^s \in I_t^s$ is used to update the key instances, which are the model components in our method. Usually the model components in the mixture model are updated using a weighted mean. However, the weighted mean is not applicable here as we will use the silhouette information

to train object detectors. Computing the mean silhouette of two given silhouettes is feasible but there can be shape distortions in the "mean" silhouettes. As a result, we employ the k-medoid clustering to update the model components. In contrast to the k-means clustering, the k-medoids algorithm chooses the actual data as centers rather than their mean values. The details of component update, namely the key positive instance update, are presented as follows. For every $x_i^s \in I_t^s$, we select the key positive instances x_j^+ from I_t^+ that satisfies $dist(x_i^s, x_j^+) < \epsilon_{dist}$, where ϵ_{dist} is a predefined value. The k-medoids clustering is used to determine a center C_i from the selected x_j^+ and x_i^s. The center C_i is then added to I_t^+. If C_i already exists in I_t^+, its corresponding weight w_i is updated as $w_i + \alpha$, where α is the learning rate. On the other hand, if C_i is inserted as a new key instance to I_t^+, the the key instance with the least weight will be discarded. Eventually all the weights are normalized. The complete algorithm for on-line instance selection is presented in Algorithm 4.

5.3.3 The Object Detector

Given the training set produced by the automatic labeler, we can construct different kinds of object detector. The focus of this thesis is codebook based learning; we therefore choose a codebook based object detector, namely the Implicit Shape Model (ISM).

5.3.3.1 The Implicit Shape Model for Object Detection

The Implicit Shape Model Leibe *et al.* (2008) is a codebook based model since information collected from local parts is retained in the form of codebooks. At the learning stage, local features are extracted from training images. A codebook of local features is then constructed by clustering all the extracted features. At runtime, local features are extracted from the images, and then they are used to match against the codewords. The codewords would cast votes for valid matches. Object hypotheses are obtained by aggregating the votes. This voting method is also called the Generalized Hough Transform.

Algorithm 4 On-line Instance Selection

INPUTS:

$I_t^+ = \{x_1^+, x_2^+, ..., x_{N_k}^+\}$ - the key positive instances.

$W_t^+ = \{w_1, w_2, ..., w_{N_k}\}$ - the weights of the key positive instances.

$I^- = \{x_1^-, x_2^-, ..., x_{N_k}^-\}$ - the key negative instances.

$\mathbb{B}_t = \{B_1, B_2, ..., B_{N_t}\}$ - the bag collection at time slice t.

N_s - the number of instances to select from $\mathbb{B}_t$.

ϵ_{dist} - a threshold on distance.

α - the learning rate for on-line learning.

OUTPUTS:

$I_{t+1}^+ = \{x_1^+, x_2^+, ..., x_{N_k}^+\}$ - the updated key positive instances.

$W_{t+1}^+ = \{w_1, w_2, ..., w_{N_k}\}$ - the updated weights of the key positive instances.

$I_t^s = \{x_1^s, x_2^s, ..., x_{N_s}^s\}$ - the selected instances from $\mathbb{B}_t$.

1: **INSTANCE SELECTION**

2: Compute the confidence value (5.5) for each $x_{ij} \in B_i$ using I_t^+ and I^-.

3: Select the top N_s instances based on the confidence values, and keep them in I_t^s.

4: **KEY INSTANCE UPDATING**

5: **FOR EACH** $x_i^s \in I_t^s$

6: Select x_j^+ from I_t^+ that satisfies $dist(x_i^s, x_j^+) < \epsilon_{dist}$.

7: Determine a medoid C_i from { selected x_j^+, x_i^s }.

8: Insert C_i to I_t^+.

9: **IF** C_i exists in I_t^+

10: **THEN** set $w_i = w_i + \alpha$.

11: **ELSE** set $w_i = \alpha$, and discard the key instance with the least weight.

12: **END IF**

13: **END FOR**

14: Normalize the updated W_t^+ to obtain W_{t+1}^+.

The codebooks for the Generalized Hough Transform are usually generated using the unsupervised k-means clustering algorithm Leibe *et al.* (2008),Maji & Malik (2009). We call them generative codebooks Okada (2009) as only positive local features are used for training. Recently discriminative codebook generating methods are proposed Gall & Lempitsky (2009), Okada (2009). The generated codebooks are considered as discriminative codebooks as they are trained using both the positive and the negative features. The discrimination between positive and negative features enables the codebook to cast more reliable probabilistic votes. In Gall & Lempitsky (2009), a Random Hough Forest is constructed using positive and negative image patches, with an objective function that measures the class and offset uncertainty. Similarly, a set of Extremely Randomized Trees are constructed in Okada (2009), and the trees are grown using an objective function that combines the discrimination and the regression. The discriminative codebooks are shown to outperform the generative codebooks in the experiments. As a result, we choose the discriminative codebook in this chapter. Given a set of object silhouettes, we intend to construct a codebook of object silhouettes for object detection.

5.3.3.2 Randomized Trees as the Codebook

We choose the Extremely Randomized Tress Geurts *et al.* (2006) as our discriminative codebook. The randomized trees algorithm Geurts *et al.* (2006) constructs an ensemble of decision or regression trees. Initially, all data is stored in the root node. Starting from the root node, a number of splits are created by firstly choosing random values from a range of chosen attribute. Each random value serves as a threshold for separating the data. The binary split that achieves the best decision or regression performance is selected to split the root node into two child nodes. The same split process applies on each node recursively. The recursive split stops when there is not enough data to split, or all the data in the current node shares the same label.

The randomized trees are firstly proposed for classification, and Okada employs them as a codebook for the Hough voting in Okada (2009). For the voting, each primitive image feature passes through each randomized tree until it reaches

one of the leaf node. The leaf node contains measurements of discriminativeness of the image feature (whether it belongs to an object or not). The response of one image feature is an ensemble of the responses from all the trees. Using the responses, each feature can cast probabilistic votes for object hypotheses. The randomized tree construction algorithms in Geurts *et al.* (2006), Okada (2009) work on the whole training sets. It is not appropriate to use them under our settings, as we want to update the trees in an on-line fashion. Inspired by the on-line random forest algorithm in Saffari *et al.* (2009), we propose an on-line learning algorithm for constructing randomized trees here. It is noted that the randomized trees are different from the random forest as there is no bootstrapping involved in constructing the randomized trees Geurts *et al.* (2006).

We build each randomized tree as a decision tree, which contains decision nodes and leaf nodes. Unlike the leaf node, each decision node retains no object location but only a split condition $s = \{f_d, \theta_d\}$, where f_d and θ_d are a randomly chosen attribute from the image feature vector, and its threshold respectively. The split s is the best split chosen from a set of random splits $S = \{s_1, s_2, ..., s_N\}$ based on some quality measure. In this thesis, N is set to 10 and the information gain is chosen as the quality measure. Denote M as the set of image features in the current node. Let M_L and M_R be the images features in the left child node and right child node respectively, according to the split s. The information gain of split s is $IG_s(M) = \frac{|M_L|}{|M|}E(M_L) + \frac{|M_R|}{|M|}E(M_R) - E(M)$, where $E(M) = -\sum_{i=1}^{C} p_i \log(p_i)$ is the entropy for C classes.

When in off-line mode, all the data is available, and therefore a robust estimate can be made at each decision node. In the on-line mode, however, the data is gathered over time. As such, when to split depends on the following factors: i) whether there is enough data for a robust estimate of statistics; ii) whether the split is good enough in terms of the quality measure. Based on these factors, two hyper parameters are introduced for the on-line learning of a random tree: i) the minimum number of training data (i.e., shape context descriptors) γ to gather before making a split; ii) the minimum information gain δ for a node to split. As a result, a node can be split into two child nodes only if $|M| > \gamma$ and $\exists s \in S,\ IG_s(M) > \delta$. The values of γ and δ are set to 0.1 and 0.8 empirically.

The on-line learning algorithm for the construction of the randomized trees is presented in Algorithm 5. The input to algorithm is a training sample $\langle x, y \rangle$, which contains a feature descriptor x and its label $y \in \{1, 0\}$. Similar to the previous chapter we use the shape context descriptor as the feature descriptor. The positive features describe the sampled points from the selected instance B_{ij} from Algorithm 4, whereas the negative features describe the sample points from the background edges. Positive samples also retain the offsets to the centroids of an object, so that the constructed randomized tree can be used for probabilistic voting, which is detailed in Section 5.3.3.3. When updating a tree, a training sample firstly passes each randomized tree until it reaches the leaf node. After appending a new feature to the leaf node, we calculate whether it is necessary to split the current leaf. In the case of a split, the data retained in the old leaf node will be propagated to its child nodes, and the old leaf node becomes a decision node.

5.3.3.3 Object Detection using Randomized Trees

We begin the moving object detection with moving edge detection between adjacent frames. We apply Canny edge detection Canny (1986) to obtain the edge map for each frame. Moving edges are then extracted by comparing edges between adjacent frames. We then sample keypoints from the identified moving edges, and attach a shape context descriptor to each sampled keypoints. Let $F = \{f_1, f_2, ...f_n\}$ be the shape context descriptors obtained from the current frame, F will be then fed into the randomized trees $T = \{t_1, t_2, ..., t_n\}$ to cast probabilistic votes for an object o and its location x. The probabilistic vote $p(o, x|f_i, T)$ from feature f_i can be decomposed as follows:

$$p(o, x|f_i, T) = p(o|f_i, T)p(x|o, f_i, T). \tag{5.8}$$

The first term $p(o|f_i, T)$ is a probabilistic output from the ensemble of trees. Denote M_{f_i,t_j} as the set of training features belong to the leaf node to which f_i reaches in tree t_j. Let the number of training features in M_{f_i,t_j} be $N_{f_i,t_j} = |M_{f_i,t_j}|$, and that of the positive features be $N^p_{f_i,t_j} = |M^p_{f_i,t_j}|$. The purity of the leaf node can be defined as $\gamma_{f_i,t_j} = \frac{N^p_{f_i,t_j}}{N_{f_i,t_j}}$. We only consider the trees with leaf nodes whose

Algorithm 5 On-line Extremely Randomized Trees

INPUTS:

$\langle\ x,\ y\ \rangle$ - the feature descriptor x and its label $y \in \{1, 0\}$.

γ - the minimum number of training data to gather before making a split.

δ - the minimum information gain for a node to split.

T=$\{t_1, t_2, ..., t_n\}$ - a set of Extremely Randomized Trees.

OUTPUTS:

T'=$\{t'_1, t'_2, ..., t'_n\}$ - the updated Extremely Randomized Trees.

FOR EACH Extremely Randomized Tree t_i
  $l_j \leftarrow$ locateLeaf(x, t_i).
  $l_j \leftarrow$ appendData(l_j, $\langle x, y \rangle$).
  IF $|l_j| > \gamma$
    S $\leftarrow$ createSplts(l_j).
    IF $\exists s \in S,\ IG_s(l_j) > \delta$
      createLeftChild(l_j, s).
      createRightChild(l_j, s).
    END IF
  END IF
END FOREACH

purity is higher than a predefined threshold. We denote the number of such trees as $N^o_{f_i}$, and $p(o|f_i, T)$ can be defined as $p(o|f_i, T) = \frac{N^o_{f_i}}{N_T}$, where N_T is the number of randomized trees.

The second term $p(x|o, f_i, T)$ describes the distribution of possible object centroid locations in regard to f_i supposing f_i being part of the object. The distribution is estimated using a non-parametric density estimation using all the trees:

$$p(x|o, f_i, T) \propto \sum_{j=1}^{N_T} \{\gamma_{f_i,t_j} \sum_{k \in M^p_{f_i,t_j}} K(\frac{x - x^p_k(f_i)}{b(x^p_k)})\}, \tag{5.9}$$

where $K(.)$ is a window function, $b(.)$ is its bandwidth, and $x^p_k(f_i)$ corresponds

Algorithm 6 Unsupervised On-line Object Detection

INITIALIZATION

Initialize I^+ and I^- using the batch instance selection algorithm.

AUTOMATIC LABELING AND ON-LINE LEARNING

FOR EACH frame

Perform background subtraction, and group foreground pixels into blobs.

Apply Algorithm 4 to select foreground blobs and update I^+.

Attach descriptors to sample edge points from instances and background.

Use the descriptors to update the randomized trees based on Algorithm 5.

END FOR

ON-LINE MOVING OBJECT DETECTION

FOR EACH frame

Identify moving edges.

Attach descriptors to the sample edge points from the moving edges.

Use the randomized trees to cast probabilistic votes based on the descriptors.

END FOR

to the object centroid location relative to the feature f_i based on the positive training feature x_k^p.

The complete proposed unsupervised moving object detection algorithm is summarized in Algorithm 6. The proposed algorithm updates the randomized trees using collected training samples from every frame, and then the updated trees are used to cast probabilistic votes for object hypotheses on the moving edges.

5.4 Experiments

Experiment Setup We conduct the experiments using three datasets. They contain videos from both outdoor and indoor surveillance. We use these videos to evaluate individual algorithm components as well as the complete algorithm.

Three video datasets are used in our experiments, namely the PETS06 set, the Visor set, and the iLIDS set. The PETS06 set PETS2006 (2006) contains pedestrian sequences recorded in a railway station which involves different viewing angles and a certain degree of occlusions. We use both the side-view and front-view sequences in the experiments. The Visor set Visor (2010) is developed by the Imagelab group of the University of Modena and Reggio Emilia. It contains a large set of multimedia data and the corresponding annotations. Some outdoor surveillance videos are selected for evaluation. The iLIDS set iLIDS (2007) is from the Image library for intelligent detection systems (i-LIDS), which is a benchmark for video analytics systems. The videos we choose contain footages recorded in a railway station by a front-mounted camera.

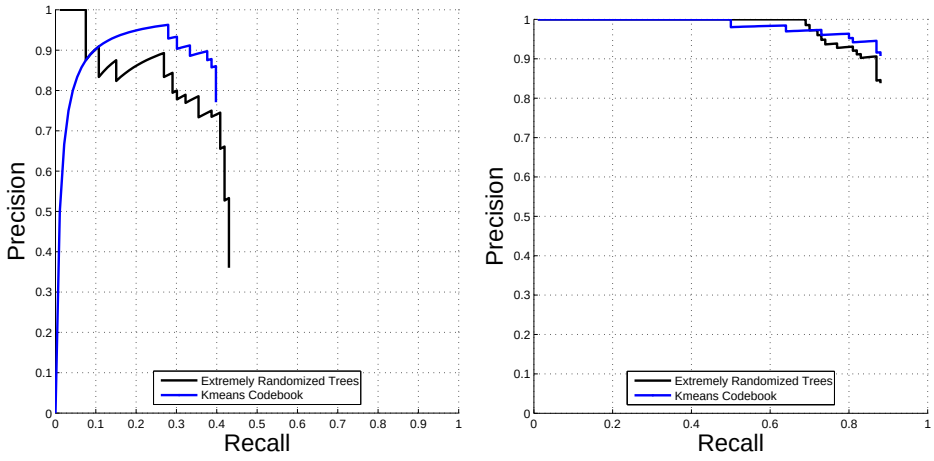
Evaluation Metric We follow the evaluation metric mentioned in Section 4.4.

5.4.1 Randomized Trees for Pedestrian Detection

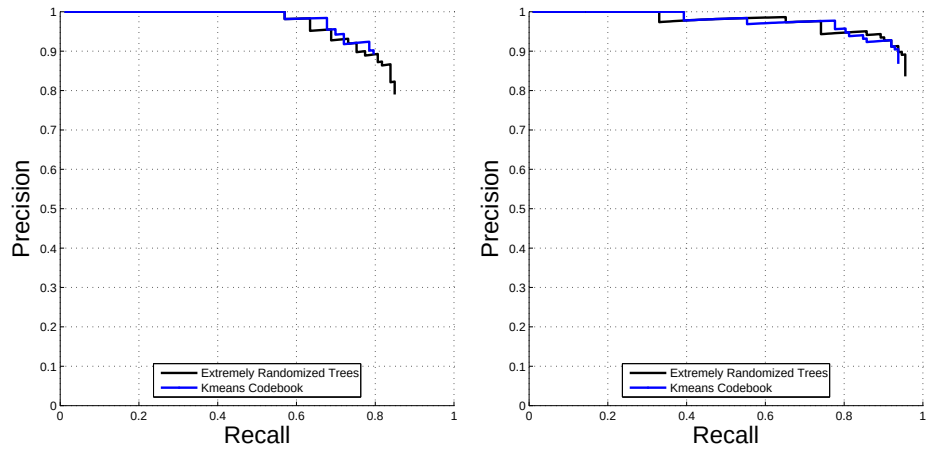
The purpose of the first experiment is to verify the choice of Randomized Trees for pedestrian detection. We compare the detection performance of Randomized Trees and k-means codebook using the same training set.

We use six sequences from the Visor Visor (2010) data set for evaluation, whose lengths range from 400 to 500 frames. For each frame, 50 frames are used for training, and the remaining frames are used for testing. We use the batch kernel density estimation based instance selection to select 10 silhouettes as the training set. Given the same training set, we construct the Randomized Tree based codebook and k-means clustering based codebook respectively.

The Precision Recall curves are shown in Figure 5.3, and sample outputs are demonstrated in Figure 5.4. It can be observed that the Randomized Trees achieve competitive performance than the k-means codebooks. In comparison with the k-means codebook, randomized trees provide a more suitable structure for on-line learning, which is shown in the following experiments.

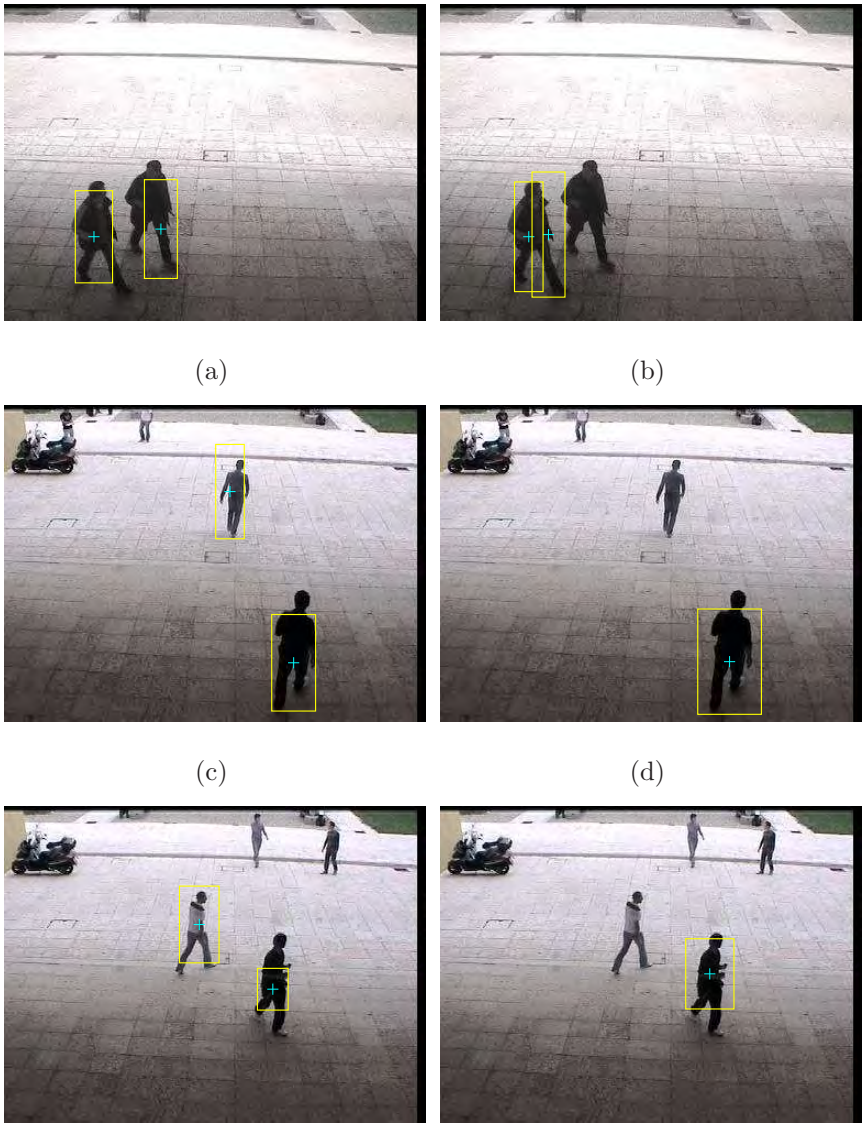


(a) (b)



(c) (d)

Figure 5.3: Performance comparison of Randomized Trees and k-means codebook on the Visor dataset.



(a) (b)

(c) (d)

(e) (f)

Figure 5.4: Sample output from detector trained by different instance selection schemes on the Visor dataset. The output on the left are produced by the Randomized Tress, whereas the output on the right are produced in by the k-means codebook.

5.4.2 On-line Instance Selection for Pedestrian Detection

As our second experiment, we evaluate the proposed on-line automatic labeler. Given the same video, two sets of training sets are produced by the batch and the

on-line labeler respectively. Two sets of randomized trees are constructed using these two training sets respectively. We then compare their detection performance on the testing sequences.

We extract five sequences from the PETS06 PETS2006 (2006) data set for evaluation, whose lengths range from 100 to 150 frames. We collect the training data from the first sequence, and use the rest for testing. Figure 5.5 depicts

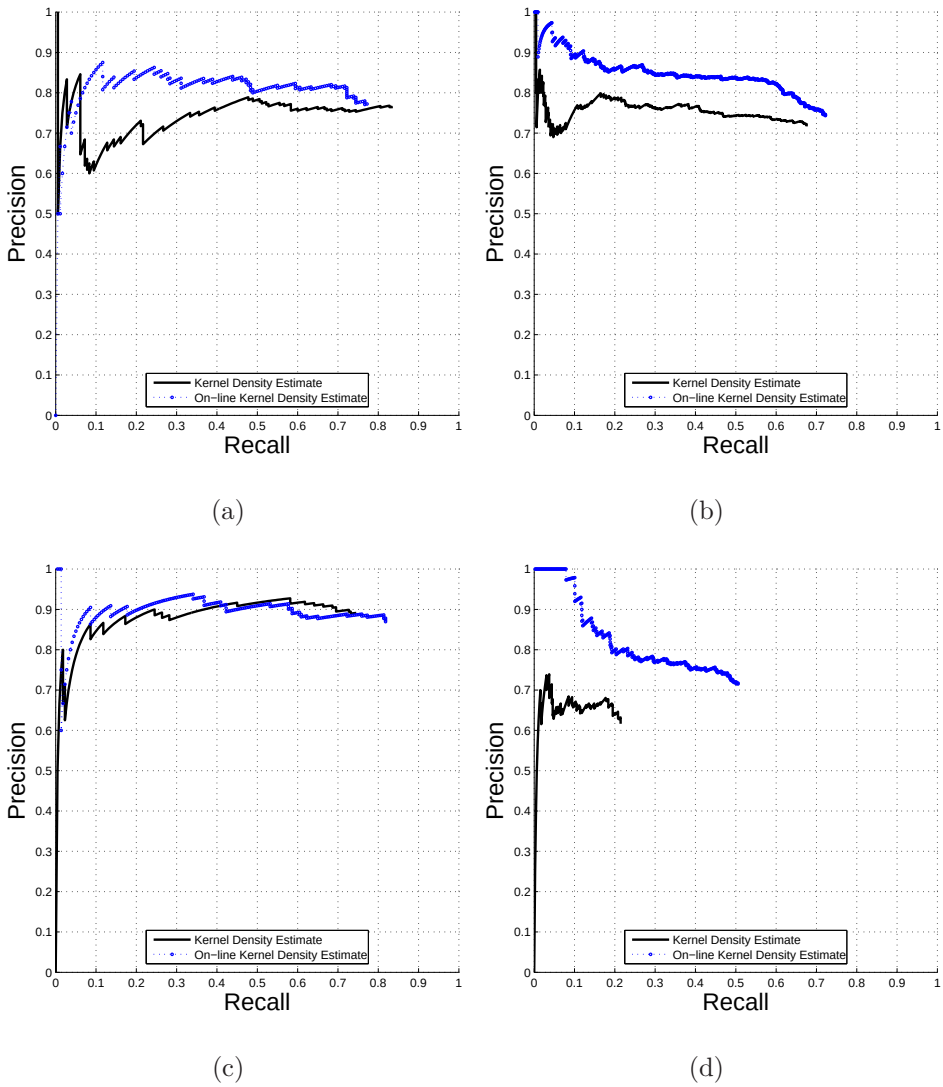


Figure 5.5: Performance comparison of different instance selection methods.

the Precision Recall curves correspond to these two different methods. It can be

observed from the curves that, the on-line instance selection achieves comparable or even better results than the batch instance selection. It seems that the on-line instance selection can capture more meaningful samples from a dynamic distribution compared to the batch selection algorithm.

5.4.3 On-line Randomized Trees for Pedestrian Detection

As our third experiment, we evaluate the proposed on-line learning algorithm for constructing the randomized trees. Given the same training set, two sets of randomized trees are constructed using the on-line and the batch learning algorithm Okada (2009) respectively. To show the advantages of on-line learning with randomized trees over traditional codebook, a k-means codebook is also trained using the on-line kmeans algorithm Bishop (2006). We then compare their detection performance on the testing sequences. We use the five sequences extracted from the PETS06 PETS2006 (2006) data set for evaluation, whose lengths range from 100 to 150 frames. We collect the training data from the first sequence, and use the rest for testing.

Figure 5.6 depicts the Precision Recall curves of the randomized trees. It can be seen from the curves that, the on-line randomized trees achieve similar recall and precision values compared with the batch randomized trees. This shows that the randomized trees can be constructed in an on-line fashion, without sacrifice in performance. Figure 5.7 shows the performance comparisons between the on-line randomized trees and the on-line k-means clustering. It can be seen from the figure that, the on-line randomized trees achieve higher recall than the on-line codebook. This shows that on-line randomized trees possess better adaptation than the on-line codebook. Essentially, on-line randomized trees obtain a better split for the feature space. The reasons might due to the following factors: (i) both positive and negative training samples are used for learning the on-line randomized trees, while only positive samples are used to learn the on-line codebook; (ii) the on-line update of the randomized trees employs an information gain based measure, which is better than the approximate cluster center update used in the on-line codebook updating. Sample results of both detectors can be found in Figure 5.8.

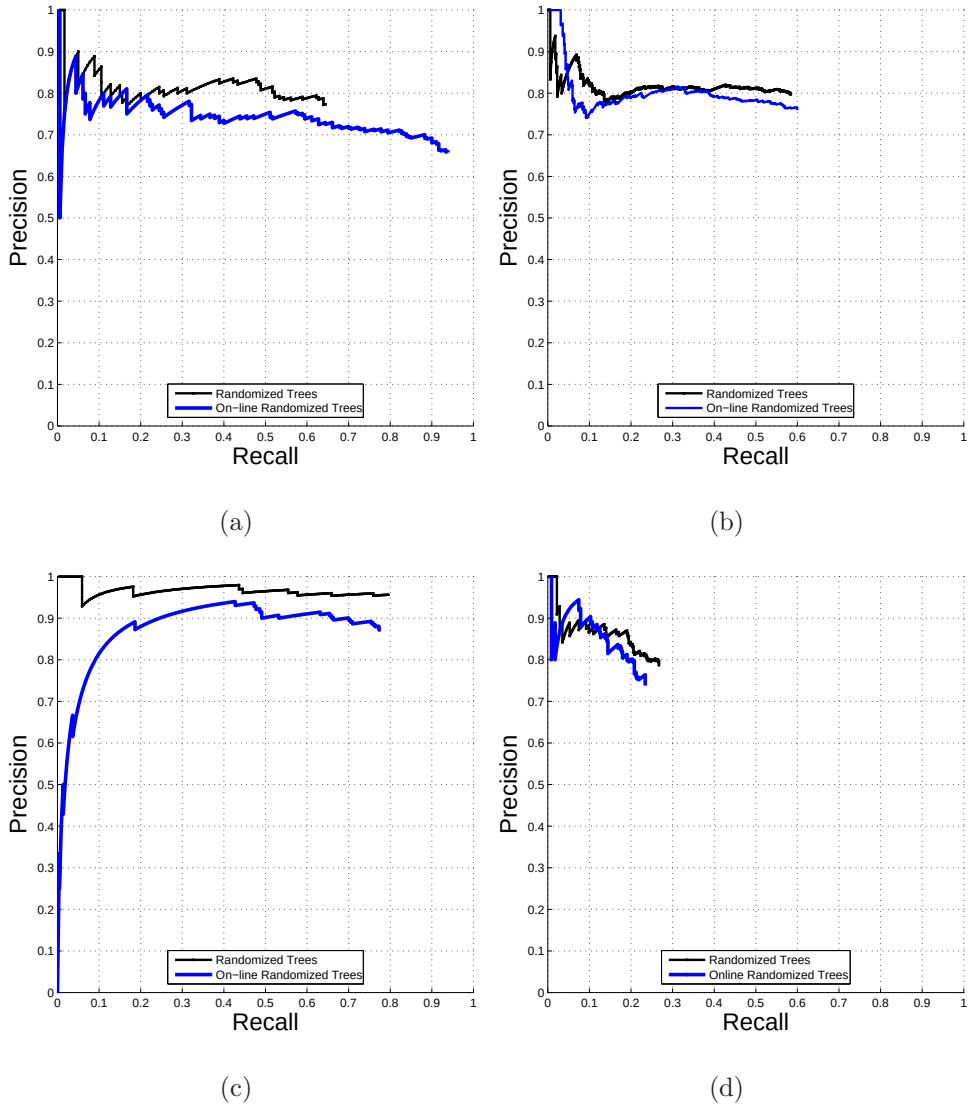


Figure 5.6: Performance comparison of randomized trees on the PETS06 dataset.

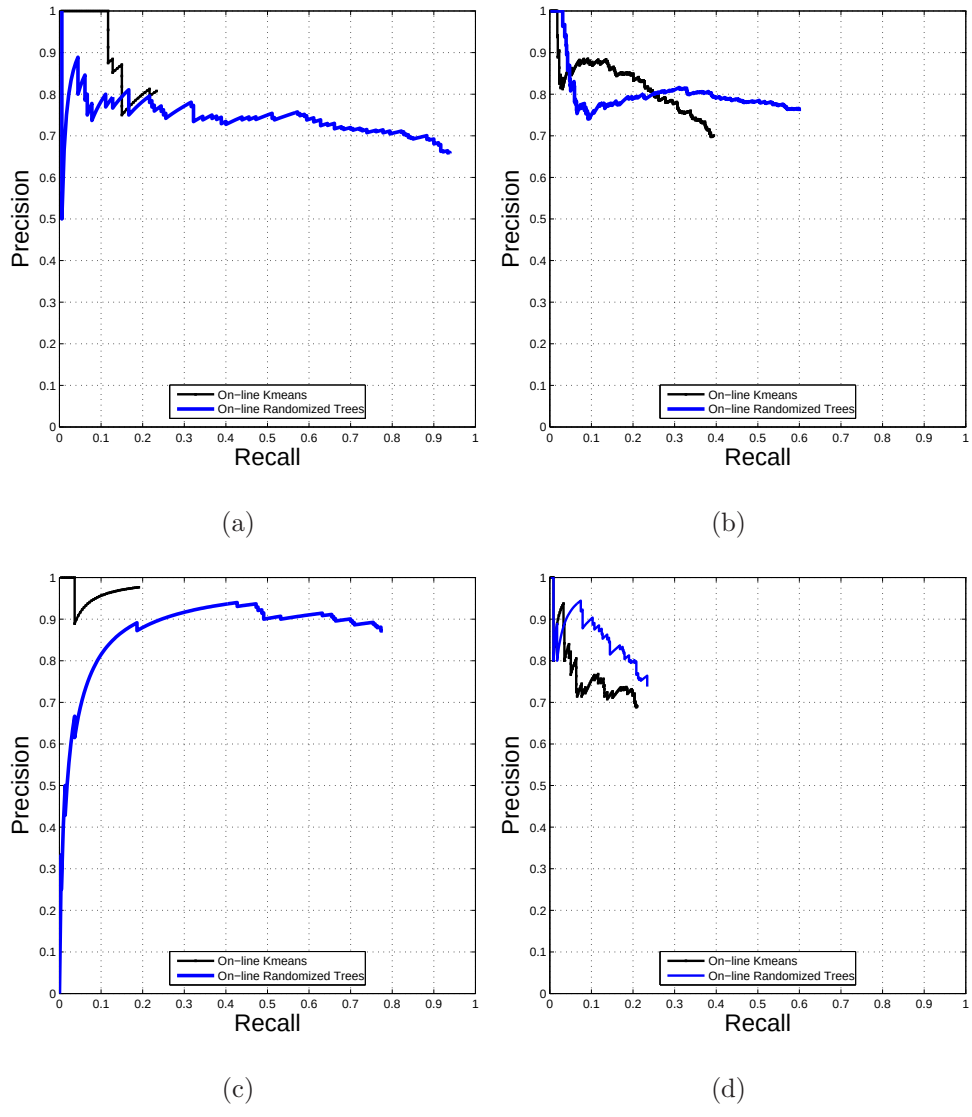
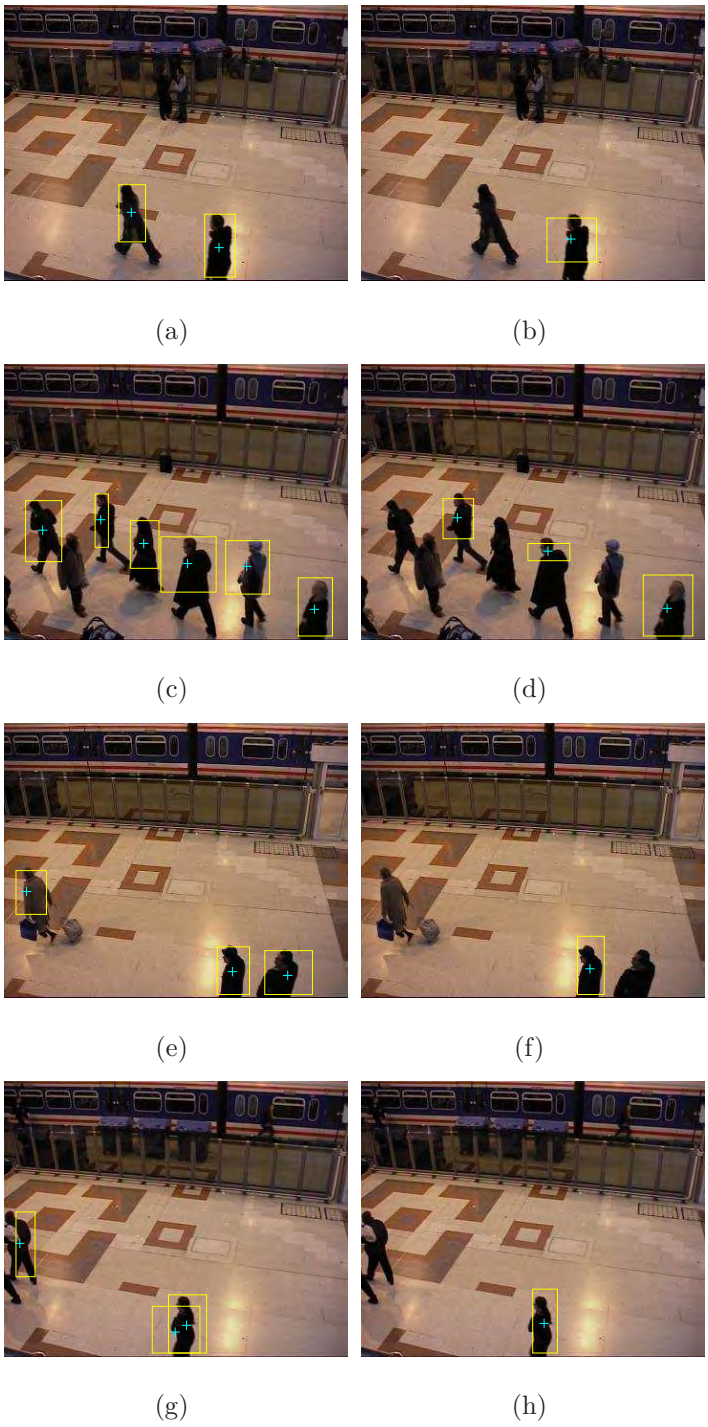


Figure 5.7: Performance comparison of the on-line randomized Trees and the on-line k-means codebook on the PETS06 dataset



(a) (b)

(c) (d)

(e) (f)

(g) (h)

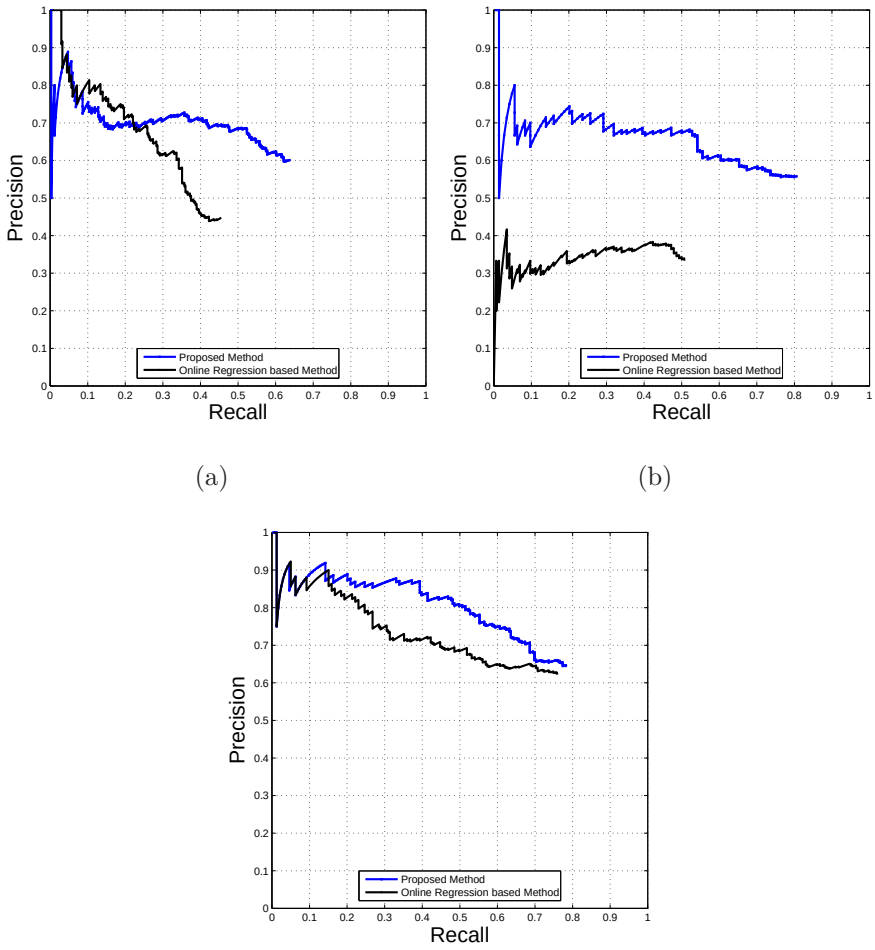
Figure 5.8: Sample output on the PETS06 dataset. The output on the left are produced by the on-line randomized trees, whereas the output on the right are produced by the on-line k-means codebook.

5.4.4 Unsupervised On-line Learning for Pedestrian Detection

In our last experiment, we compare our unsupervised on-line learning framework with the state-of-the-art conservative learning framework Ikizler-Cinbis *et al.* (2009). A conservative labeler which usually produces a biased training set, whose bias is caused by rigid requirements and also high thresholds for selection.

We employ both frameworks to train pedestrian detectors in an on-line fashion. Two sets of randomized trees are trained and updated using the produced training set respectively. Due to the rigid requirements, the conservative labeler only considers the foreground blobs whose aspect ratio are within some predefined range. Our labeler does not have such requirements, and therefore it can capture the multi-modal nature of the data.

We extract three sequences from the iLIDs dataset, and each of them contains 1000 frames. Given one sequence, pedestrian detectors are automatically trained by the proposed framework and the conservative learning framework. The precision recall curves of both frameworks are presented in Figure 5.9, and the sample results are shown in Figure 5.10. As shown in Figure 5.9, the proposed framework outperforms the state-of-the-art conservative learning framework in the sense that the former achieves higher accuracies and recalls. Most of the sequences contain pedestrians that walk close to each other for some time, whose corresponding foregrounds do not fit the aspect ratio for a single person. As a result, the distributions of the silhouettes can be considered as multi-modal distributions. Due to its conservativeness, the conservative learning ignores the modal that corresponds to the case of two close pedestrians. On the other hand, our labeler can still cope with foreground blobs that contains multiple persons. As shown in the sample outputs, our proposed framework can successfully detect each individual pedestrian. That is probably the reason why the conservative learning fail to achieve high accuracy and recall.



(a) (b)

(c)

Figure 5.9: Performance comparison of the proposed framework and the conservative learning framework.

5.5 Conclusions

We have presented a unsupervised on-line learning framework for object detection. We proposed an on-line instance selection algorithm based on Multiple Instance Learning. On the basis of the selected instances, a set of Extremely Randomized Trees can be constructed in an on-line manner. We have evaluated the algorithms on three video datasets. The experimental results demonstrate that our framework outperforms the state-of-the-art on-line conservative learning

(a) (b)

(c) (d)

(e) (f)

Figure 5.10: Sample output on the iLIDs dataset. The output on the left are produced by the proposed framework, whereas the output on the right are produced by the state-of-the-art conservative learning framework.

framework.

Chapter 6

Conclusions

6.1 Summary

In this thesis, we have developed on-line and unsupervised learning algorithms for codebook based visual recognition methods. The goal of these algorithms is to reduce human effort when dealing with a large amount of data.

We focus on on-line learning in chapter 3, where we develop an on-line learning algorithm for the Probabilistic Latent Semantic Analysis (PLSA) model. The proposed algorithm enables the PLSA to deal with data that arrive sequentially. We study the unsupervised learning in chapter 4, where we propose extensions to the Implicit Shape Model (ISM). Our contributions include two automatic training data selection methods, and also a moving edge detection method. The proposed algorithms enable the ISM to detect moving objects without human supervision. Finally we look at both unsupervised and on-line learning in chapter 5, where our contributions include an on-line training data selection scheme, and the on-line Extremely Randomized Tree algorithm. These two algorithms constitute a framework that can adaptively detect moving objects without human supervision.

6.2 Future Work

In this section we present the details for the future work of the proposed algorithms.

The on-line PLSA proposed in Chapter 3 can be improved in the following ways. Firstly, our on-line PLSA assumes a static distribution for data, however the data distribution can be dynamic. Enabling the PLSA to learn a dynamic distribution for streaming data is a direction for future work. This might probably require a different optimization algorithm rather than the on-line EM, since the on-line EM is assumed to converge only under a static distribution. Though the QB-PLSA has already been proposed for text mining, there is still room for a more flexible PLSA algorithm in visual recognition. Secondly, the number of topics for the PLSA is set prior to the learning. This variable is usually data dependent, and therefore we should be able to learn it from the given data. A more flexible on-line algorithm that can learn the topic number automatically is another direction for future work. Thirdly, the learning of our on-line PLSA is actually subbatch based learning. The reason for the subbatch learning probably lies in the fact that the original EM algorithm derives a closed form estimation for PLSA. The initialization for the EM plays an important role under the closed form, and therefore a single data sample is not enough for getting a good estimate. A more flexible on-line algorithm should process a data sample each time. To enable the PLSA to process a data sample at a time, we might have to employ a different optimization algorithm such as the gradient descent. A non-closed form of parameter estimation should be derived based on the chosen optimization algorithm. This is the most interesting research direction for future work.

The proposed algorithms in chapter 4 and 5 can be considered as similar algorithms with different focus. Future work can be investigated in several ways. Firstly, different combinations of feature descriptors and object detectors can be employed under the proposed framework. Experimental evaluation on different combinations of feature descriptors and object detectors is a immediate direction for future work. Secondly, the training and the testing phase are independent from each other in the sense that there is no feedback from the testing phase. It is expected the feedback from the testing phase can be used to improve the

training data selection. Finally, the proposed framework can only distinguish two object classes, namely foreground objects (pedestrians) and the background. However there can be moving objects from different classes in a video. As a result, a framework that can distinguish moving objects from multiple classes is another research direction.

Appendix A

List of Publications and Patents

A.1 Publications related to the thesis

- Jie Xu, Yang Wang, Getian Ye, Wei Wang, Jun Yang.(2010). "On-line PLSA for Visual Recognition." *Asian Conference on Computer Vision.*

- Jie Xu, Yang Wang, Wei Wang, Jun Yang.(2010). "Unsupervised Moving Object Detection with On-line Generalized Hough Transform." *Asian Conference on Computer Vision.*

- Jie Xu, Getian Ye, Gunawan Herman, Bang Zhang. (2008). "Detecting and recognizing moving pedestrians in video." *IEEE International Workshop on Multimedia Signal Processing.*

A.2 Other Publications

- Jie Xu, Yang Wang, Fang Chen, Ho Choi, Guanzhong Li, Siyuan Chen, Sazzad Hussain.(2011). "Pupillary Response Based Cognitive Workload Index under Luminance and Emotional Changes." *ACM CHI Conference on Human Factors in Computing Systems.*

- Jun Yang, Yang Wang, Arcot Sowmya, Bang Zhang, Jie Xu and Zhidong Li.(2010). "Spatial-Temporal Affinity Propagation for Feature Clustering with Application to Traffic Video Analysis." *Asian Conference on Computer Vision.*

- Zhidong Li, Jie Xu, Yang Wang, Glenn Geers, Jun Yang.(2011). "Saliency Detection Based on Proto-Object and Topic Model." *IEEE Workshop on Applications of Computer Vision.*

- Zhidong Li, Yang Wang, Jing Chen, Jie Xu, and John Laird.(2010). "Image Topic Discovery with Saliency Detection." *British Machine Vision Conference.*

- Bang Zhang, Getian Ye, Yang Wang, Wei Wang, Jie Xu, Gunawan Herman and Jun Yang.(2010). "Multi-class Graph Boosting with Subgraph Sharing for Object Recognition." *International Conference on Pattern Recognition.*

- Jun Yang, Yang Wang, Getian Ye, Arcot Sowmya, Bang Zhang and Jie Xu.(2009). "Feature Clustering for Vehicle Detection and Tracking in Road Traffic Surveillance." *International Conference on Image Processing.*

- Jie Xu, Getian Ye, Yang Wang, Gunawan Herman, Bang Zhang.(2009). "Incremental EM for Probabilistic Latent Semantic Analysis on Human Action Recognition." *IEEE International Conference on Advanced Video and Signal Based Surveillance.*

- Bang Zhang, Getian Ye, Yang Wang, Wei Wang, Jie Xu, Gunawan Herman and Jun Yang.(2009). "Informative Frequent Assembled Feature for Face Detection." *IEEE International Conference on Image Processing.*

- Bang Zhang, Getian Ye, Yang Wang, Jie Xu, Gunawan Herman.(2009). "Finding Shareable Informative Patterns and Optimal Coding Matrix for Multiclass Boosting", *IEEE International Conference on Computer Vision.*

- Jie Xu, Getian Ye, Gunawan Herman, Bang Zhang.(2008). "An efficient approach to detecting pedestrians in video", *ACM International Conference on Multimedia.*

- Gunawan Herman, Getian Ye, Jie Xu, Bang Zhang.(2008). "Improving object detection by removing noisy samples from training sets", *ACM International Conference on Multimedia Information Retrieval.*

A.3 Patents

- Getian Ye, Jie XU.(2008). "Traffic Information about Objects in a Video." *Australian Provisional Patent.* **2008903430.**

References

Aksoy, S., Koperski, K., Tusk, C., Marchisio, G. & Tilton, J. (2003). Learning Bayesian classifiers for a visual grammar. In *IEEE Workshop on Advances in Techniques for Analysis of Remotely Sensed Data*, 212–218. 31

AlSumait, L., Barbará, D. & Domeniconi, C. (2008). Online LDA: Adaptive Topic Model for Mining Text Streams with Application on Topic Detection and Tracking. In *International Conference on Data Mining*. 54

An, S., Peursum, P., Liu, W. & Venkatesh, S. (2009). Efficient algorithms for subwindow search in object detection and localization. In *IEEE Conference on Computer Vision and Pattern Recognition*. 18

Andriluka, M., Roth, S. & Schiele, B. (2008). People-tracking-by-detection and people-detection-by-tracking. In *IEEE Conference on Computer Vision and Pattern Recognition*. 68, 97

Balcan, M., Blum, A. & Yang, K. (2005). Co-training and expansion: Towards bridging theory and practice. In *Neural Information Processing Systems*. 16, 70

Ballard, D. (1981). Generalizing the Hough transform to detect arbitary shapes. *Pattern Recognition*, **13**, 111–122. 22

Bay, H., Tuytelaars, T. & Van Gool, L. (2006). Surf: Speeded up robust features. In *European Conference of Computer Vision*, 404–417. 13

Belongie, S., Malik, J. & Puzicha, J. (2002). Shape matching and object recognition using shape contexts. *IEEE Transaction on Pattern Analysis and Machine Intelligence*. 13, 77

Biederman, I., Mezzanotte, R. & Rabinowitz, J. (1982). Scene perception: Detecting and judging objects undergoing relational violations. *Cognitive Psychology*, **14**, 143–177. 27, 69

Bishop, C. (2006). *Pattern recognition and machine learning*. Springer New York. 115

Bissacco, A., Yang, M. & Soatto, S. (2007). Detecting humans via their pose. In *Advances in Neural Information Processing Systems*. 41

Blei, D., Ng, A. & Jordan, M. (2003). Latent dirichlet allocation. *The Journal of Machine Learning Research*, **3**, 993–1022. 14

Bosch, A., Zisserman, A. & Munoz, X. (2006). Scene classification via pLSA. In *European Conference of Computer Vision*, 517–530. 34

Bosch, A., Zisserman, A. & Munoz, X. (2008). Scene classification using a hybrid generative/discriminative approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **30**, 712–727. 44, 45, 58

Bouchard, G., Triggs, B., LEAR, G. & Montbonnot, F. (2005). Hierarchical part-based visual object categorization. In *IEEE Conference on Computer Vision and Pattern Recognition*. 69, 97

Bourdev, L. & Brandt, J. (2005). Robust object detection via soft cascade. In *IEEE Conference on Computer Vision and Pattern Recognition*, vol. 2, 236–243. 19

Bregonzio, M., Gong, S. & Xiang, T. (2009). Recognising action as clouds of space-time interest points. In *IEEE Conference on Computer Vision and Pattern Recognition*. 40, 45

Breu, H., Gil, J., Kirkpatrick, D. & Werman, M. (1995). Linear time Euclidean distance transform algorithms. *IEEE Transaction on Pattern Analysis and Machine Intelligence*. 75

Brinks, R. (2008). On the convergence of derivatives of B-splines to derivatives of the Gaussian function. *Computational & Applied Mathematics*, **27**, 79–92. 11

Bugeau, A. & Pérez, P. (2007). Detection and segmentation of moving objects in highly dynamic scenes. In *IEEE Conference on Computer Vision and Pattern Recognition*, 1–8. 26

Canny, J. (1986). A computational approach to edge detection. *IEEE Transaction on Pattern Analysis and Machine Intelligence*. 79, 108

Canny, J. (1987). A computational approach to edge detection. *Readings in computer visifn: issues, problems, principles, and paradigms*, **184**. 11

Carlos Niebles, J., Wang, H. & Li, F.F. (2008). Unsupervsied Learning of Human Action Categories Using Spatial-Temporal Words. *International Journal of Computer Vision*, **42**, 993–1022. 44, 45, 48, 65

Chien, J. & Wu, M. (2008). Adaptive Bayesian latent semantic analysis. *IEEE Transactions on Audio, Speech, and Language Processing*, **16**, 198–207. 54, 57, 61

Choi, M., Lim, J., Torralba, A. & Willsky, A. (2010). Exploiting hierarchical context on a large database of object categories. In *IEEE Conference on Computer Vision and Pattern Recognition*. 27, 69

Chou, T. & Chen, M. (2008). Using incremental PLSI for threshold-resilient online event analysis. *IEEE Transactions on Knowledge and Data Engineering*, **20**, 289. 54

Comaniciu, D. & Meer, P. (2002). Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **24**, 603–619. 80

Csurka, G., Dance, C., Fan, L., Willamowski, J. & Bray, C. (2004). Visual categorization with bags of keypoints. In *ECCV Workshop on Statistical Learning in Computer Vision*, vol. 1, 22. 4

Cucchiara, R., Grana, C., Piccardi, M. & Prati, A. (2003). Detecting moving objects, ghosts, and shadows in video streams. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1337–1342. 73

Cutler, R. & Davis, L. (2000). Real-time periodic motion detection, analysis, and applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **22**, 781–796. 37, 45

D. Comaniciu, P.M. (2002). Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 26

Dalai, N., Triggs, B., Rhone-Alps, I. & Montbonnot, F. (2005). Histograms of oriented gradients for human detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1. 24, 97

Dalal, N. & Triggs, B. (2005). Histograms of oriented gradients for human detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, 886–893. 17, 68

Davis, J. & Bobick, A. (1997). The representation and recognition of human movement using temporal templates. In *IEEE Conference on Computer Vision and Pattern Recognition*. 37, 45

Dempster, A., Laird, N., Rubin, D. *et al.* (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, **39**, 1–38. 47

Divvala, S., Hoiem, D., Hays, J., Efros, A. & Hebert, M. (2009). An empirical study of context in object detection. In *IEEE Conference on Computer Vision and Pattern Recognition*. 27, 69

Dollar, P., Rabaud, V., Cottrell, G. & Belongie, S. (2005a). Behavior recognition via sparse spatio-temporal features. In *Joint IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, 65–72. 39

Dollar, P., Vincent, R. & Garrison, C. (2005b). Behavior recognition via sparse spatio-temporal features. In *VS-PETS*. 45, 64

Duda, R., Hart, P. & Stork, D. (2001). *Pattern classification*, vol. 2. Wiley-Interscience. 10, 75

Dundar, M. & Bi, J. (2007). Joint optimization of cascaded classifiers for computer aided detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, 1–8. xvi, 19, 20

Elgammal, A., Harwood, D. & Davis, L. (2000). Non-parametric model for background subtraction. In *European Conference on Computer Vision*, 751–767. 98

Fan, J., Gao, Y., Luo, H. & Xu, G. (2005). Statistical modeling and conceptualization of natural images. *Pattern Recognition*, **38**, 865–885. 30

Fei-Fei, L. & Perona, P. (2005). A Bayesian hierarchical model for learning natural scene categories. In *IEEE Conference on Computer Vision and Pattern Recognition*. 34, 44, 45

Felzenszwalb, P. & Huttenlocher, D. (2005). Pictorial structures for object recognition. *International Journal of Computer Vision*, **61**, 55–79. 21

Felzenszwalb, P., McAllester, D. & Ramanan, D. (2008). A discriminatively trained, multiscale, deformable part model. In *IEEE Conference on Computer Vision and Pattern Recognition*. xvi, 24, 25, 69, 97

Felzenszwalb, P., Girshick, R. & McAllester, D. (2010). Cascade object detection with deformable part models. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2241–2248. 25

Feng, X. & Perona, P. (2002). Human action recognition by sequence of movelet codewords. In *International Symposium on 3D Data Processing Visualization and Transmission*. 38

Fergus, R., Perona, P. & Zisserman, A. (2007). Weakly supervised scale-invariant learning of models for visual recognition. *International Journal of Computer Vision*. 68

Fidler, S., Berginc, G. & Leonardis, A. (2006). Hierarchical statistical learning of generic parts of object structure. In *IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, 182–189. 22

Fischler, M. & Elschlager, R. (1973). The representation and matching of pictorial structures. *IEEE Transactions on Computers*, **100**, 67–92. xvi, 20, 21

Fredembach, C., Schroder, M. & Susstrunk, S. (2004). Eigenregions for image classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **26**, 1645–1649. 31

Freund, Y. & Schapire, R.E. (1995). A desicion-theoretic generalization of on-line learning and an application to boosting. In *Computational learning theory*, 23–37. 15

Gall, J. & Lempitsky, V. (2009). Class-Specific Hough Forests for Object Detection. In *IEEE Conference on Computer Vision and Pattern Recognition*. 24, 97, 106

Galleguillos, C., Rabinovich, A. & Belongie, S. (2008). Object categorization using co-occurrence, location and appearance. In *IEEE Conference on Computer Vision and Pattern Recognition*. 27, 69

Galleguillos, C., McFee, B., Belongie, S. & Lanckriet, G. (2010). Multi-class object localization by combining local contextual interactions. In *IEEE Conference on Computer Vision and Pattern Recognition*. 27

Gavrila, D. (2000). Pedestrian detection from a moving vehicle. In *European Conference on Computer Vision*. 68

Geurts, P., Ernst, D. & Wehenkel, L. (2006). Extremely randomized trees. *Machine Learning*. 24, 106, 107

Girolami, M. & Kabán, A. (2003). On an equivalence between PLSI and LDA. In *ACM SIGIR*, 434. 32, 44

Google (2009). Developer's Guide to Picasa Web Albums Data API. *http://code.google.com/apis/picasaweb/overview.html*. 58

Gorelick, L., Blank, M., Shechtman, E., Irani, M. & Basri, R. (2007). Actions as space-time shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **29**, 2247–2253. 38

Gupta, P., Arrabolu, S., Brown, M. & Savarese, S. (2009). Video Scene Categorization by 3D Hierarchical Histogram Matching. In *International Conference on Computer Vision*. 44

Hamilton, J. (1991). A quasi-Bayesian approach to estimating parameters for mixtures of normal distributions. *Journal of Business & Economic Statistics*, **9**, 27–39. 54

Han, M., Xu, W. & Gong, Y. (2006). Video object segmentation by motion-based sequential feature clustering. In *ACM international conference on Multimedia*, 773–782. 26, 69, 96

Harris, C. & Stephens, M. (1988). A combined corner and edge detector. In *Alvey vision conference*, vol. 15, 50. 12

Heinz, C. & Seeger, B. (2006). Resource-aware kernel density estimators over streaming data. In *ACM international conference on Information and knowledge management*, 870–871. 102

Hofmann, T. (1999). Probabilistic latent semantic analysis. In *Uncertainty in Artificial Intelligence*, 289–296. 14

Hofmann, T. (2001). Unsupervised Learning by Probabilistic Latent Semantic Analysis. *Machine Learning*, **42**, 177–196. xvi, 32, 33, 43, 46, 47, 48

Horster, E., Lienhart, R. & Slaney, M. (2008). Continuous visual vocabulary modelsfor plsa-based scene recognition. In *International Conference on Image and Video Retrieval*. 61, 62

Ikizler-Cinbis, N., Cinbis, R. & Sclaroff, S. (2009). Learning actions from the web. In *IEEE International Conference on Computer Vision*. 100, 119

iLIDS (2007). http://www.elec.qmul.ac.uk/staffinfo/andrea/avss2007.html. 82, 87, 111

Javed, O., Ali, S. & Shah, M. (2005). Online detection and classification of moving objects using progressively improving detectors. In *IEEE Conference on Computer Vision and Pattern Recognition*. 70

Johansson, G. (1975). Visual motion perception. *Scientific American*, **232**, 76–88. 37

Juang, B. & Rabiner, L. (1991). Hidden Markov models for speech recognition. *Technometrics*, **33**, 251–272. 14, 38

Ke, Y. & Sukthankar, R. (2004). PCA-SIFT: A more distinctive representation for local image descriptors. In *IEEE Conference on Computer Vision and Pattern Recognition*. 13

Ke, Y., Sukthankar, R. & Hebert, M. (2005). Efficient visual event detection using volumetric features. In *IEEE International Conference on Computer Vision*. 40, 45

Khan, S. & Shah, M. (2001). Object based segmentation of video using color, motion and spatial information. In *IEEE Conference on Computer Vision and Pattern Recognition*. 26, 96

Lafferty, J., McCallum, A. & Pereira, F. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *International Conference on Machine Learning*, 282–289. 15

Lampert, C., Blaschko, M. & Hofmann, T. (2008). Beyond sliding windows: Object localization by efficient subwindow search. In *IEEE Conference on Computer Vision and Pattern Recognition*, 1–8. 17

Laptev, I. & Lindeberg, T. (2003). Space-time interest points. In *IEEE Conference on Computer Vision and Pattern Recognition*. 39, 45

Lazebnik, S., Schmid, C. & Ponce, J. (2006). Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *IEEE Conference on Computer Vision and Pattern Recognition*, vol. 2. xvi, 34, 35

Leibe, B., Seemann, E. & Schiele, B. (2005). Pedestrian detection in crowded scenes. In *IEEE Conference on Computer Vision and Pattern Recognition*. xvi, 22, 23, 69

Leibe, B., Leonardis, A. & Schiele, B. (2008). Robust Object Detection with Interleaved Categorization and Segmentation. *International Journal of Computer Vision*. 8, 69, 70, 79, 82, 83, 85, 97, 104, 106

Levin, A., Viola, P. & Freund, Y. (2003). Unsupervised improvement of visual detectors using co-training. In *International Conference on Computer Vision*. 70

Li, L., Socher, R. & Fei-Fei, L. (2009). Towards Total Scene Understanding Classification, Annotation and Segmentation in an Automatic Framework. In *IEEE Conference on Computer Vision and Pattern Recognition*. 44, 45

Lindeberg, T. (1993). *Scale-space theory in computer vision*. Springer. 12

Little, J. & Boyd, J. (1998). Recognizing people by their gait: the shape of motion. *Videre: Journal of Computer Vision Research*, **1**, 1–32. 37, 45

Liu, F. & Gleicher, M. (2009). Learning color and locality cues for moving object detection and segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*. 26

Liu, J. & Shah, M. (2008). Learning human actions via information maximization. In *IEEE Conference on Computer Vision and Pattern Recognition*. 40

Lowe, D. (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, **60**, 91–110. 12, 13, 58, 61

Lu, Z. & Ip, H. (2009). Image categorization with spatial mismatch kernels. In *IEEE Conference on Computer Vision and Pattern Recognition*. 35

Luo, J., Savakis, A. & Singhal, A. (2005). A Bayesian network-based framework for semantic image understanding. *Pattern Recognition*, **38**, 919–934. 30, 31

M. Blei, D., Y. Ng, A. & I.Jordan, M. (2003). Latent Dirichlet Allocation. *The Journal of Machine Learning Research*, **3**, 993–1022. xvi, 32, 33, 44

Maji, S. & Malik, J. (2009). Object detection using a max-margin hough transform. In *International Conference on Computer Vision*. 23, 106

Maron, O. & Lozano-Pérez, T. (1998). A framework for multiple-instance learning. *Neural Information Processing Systems*. 75

Matas, J., Chum, O., Urban, M. & Pajdla, T. (2004). Robust wide-baseline stereo from maximally stable extremal regions. *Image and Vision Computing*, **22**, 761–767. 12

Micilotta, A., Ong, E. & Bowden, R. (2005). Detection and tracking of humans by probabilistic body part assembly. In *British Machine Vision Conference*. 69

Mikolajczyk, K. & Schmid, C. (2004). Scale & affine invariant interest point detectors. *International Journal of Computer Vision*, **60**, 63–86. 12

Mikolajczyk, K. & Schmid, C. (2005). A performance evaluation of local descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1615–1630. 11, 13

Moosmann, F., Triggs, B. & Jurie, F. (2006). Fast discriminative visual codebooks using randomized clustering forests. In *Advances in neural information processing systems*, vol. 19, 985. 4

Moravec, H. (1979). Visual mapping by a robot rover. In *International Joint Conference on Artificial Intelligence*, 598–600. 12

Munder, S. & Gavrila, D. (2006). An experimental study on pedestrian classification. *IEEE Transaction on Pattern Analysis and Machine Intelligence*. 97

Nair, V. & Clark, J. (2004). An unsupervised, online learning framework for moving object detection. In *IEEE Conference on Computer Vision and Pattern Recognition*. xviii, 71, 76, 87, 92, 94, 99

Ni, B., Yan, S. & Kassim, A. (2009). Recognizing human group activities with localized causalities. In *IEEE Conference on Computer Vision and Pattern Recognition*. 39, 45

Niebles, J., Wang, H. & Fei-Fei, L. (2008). Unsupervised learning of human action categories using spatial-temporal words. *International Journal of Computer Vision*, **79**. 41

Nowozin, S., Bakir, G. & Tsuda, K. (2007). Discriminative subsequence mining for action classification. In *IEEE International Conference on Computer Vision*. 40

Okada, R. (2009). Discriminative Generalized Hough Transform for Object Detection . In *International Conference on Computer Vision*. 24, 97, 106, 107, 115

Oliva, A. & Torralba, A. (2001). Modeling the shape of the scene: A holistic representation of the spatial envelope. *International Journal of Computer Vision*, **42**, 145–175. 35, 61

Oliver, N., Garg, A. & Horvitz, E. (2004). Layered representations for learning and inferring office activity from multiple sensory channels. *Computer Vision and Image Understanding*, **96**, 163–180. 38, 45

Opelt, A., Pinz, A. & Zisserman, A. (2006). A boundary-fragment-model for object detection. In *European Conference on Computer Vision*, 575–588. 23

Papoulis, A. & Pillai, S.U. (2002). *Probability, Random Variables, and Stochastic Processes*. McGraw-Hills. 49

Parikh, D., Zitnick, C. & Chen, T. (2008). From appearance to context-based recognition: Dense labeling in small images. In *IEEE Conference on Computer Vision and Pattern Recognition*. 27

PETS2006 (2006). http://www.cvg.rdg.ac.uk/pets2006/data.html. 81, 111, 114, 115

Prabhakar, K., Oh, S., Wang, P., Abowd, G. & Rehg, J. (2010). Temporal causality for the analysis of visual events. In *IEEE Conference on Computer Vision and Pattern Recognition*. 39, 45

Quattoni, A. & Torralb, A. (2009). Recognizing Indoor Scenes. In *IEEE Conference on Computer Vision and Pattern Recognition*. 44, 45

Quattoni, A. & Torralba, A. (2009). Recognizing indoor scenes. In *IEEE Conference on Computer Vision and Pattern Recognition*. xvii, 36

Quelhas, P., Monay, F., Odobez, J., Gatica-Perez, D., Tuytelaars, T. & Van Gool, L. (2005). Modeling scenes with local descriptors and latent aspects. In *International Conference on Computer Vision*. 33, 34

R. Fergus, L.F.F. & Torralba, A. (2005). ICCV 2005 short course on Object Recognition. *http://people.csail.mit.edu/fergus/iccv2005/bagwords.html*. 58

Rabiner, L. & Juang, B. (1993). *Fundamentals of speech recognition*. Prentice hall Englewood Cliffs, New Jersey. 21

Rodriguez, M. & Shah, M. (2007). Detecting and segmenting humans in crowded scenes. In *ACM International Conference on Multimedia*. 70, 79

Roth, P., Grabner, H., Skocaj, D., Bischof, H. & Leonardis, A. (2005). On-line conservative learning for person detection . In *VS-PETS*. 71, 99

Rubner, Y., Tomasi, C. & Guibas, L. (1998). A metric for distributions with applications to image databases. In *International Conference on Computer Vision*, 59–66. 14

Saffari, A., Leistner, C., Santner, J., Godec, M. & Bischof, H. (2009). On-line Random Forests. In *The 3rd On-line learning for Computer Vision Workshop*. 107

Sato, M. (2000). Convergence of on-line EM algorithm. *ICONIP*, **1**, 476–481. 53

Savarese, S., DelPozo, A., Niebles, J. & Fei-Fei, L. (2008). Spatial-temporal correlations for unsupervised action classification. In *IEEE Workshop on Applications of Computer Vision*. 44

Schneiderman, H. & Kanade, T. (2004). Object detection using the statistics of parts. *International Journal of Computer Vision*, **56**, 151–177. 20, 21

Schuldt, C., Laptev, I. & Caputo, B. (2004). Recognizing human actions: A local svm approach. In *IEEE Conference on Computer Vision and Pattern Recognition*. 64

Serrano, N., Savakis, A. & Luo, J. (2004). Improved scene classification using efficient low-level features and semantic cues. *Pattern Recognition*, **37**, 1773–1784. 29

Shechtman, E. & Irani, M. (2005). Space-time behavior based correlation. In *IEEE Conference on Computer Vision and Pattern Recognition*. 37, 45

Shotton, J., Johnson, M. & Cipolla, R. (2008). Semantic texton forests for image categorization and segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*. 27, 69

Sivic, J., Russell, B., Efros, A., Zisserman, A. & Freeman, W. (2005). Discovering objects and their location in images. In *International Conference on Computer Vision*. 4, 32, 34, 44

Stauffer, C. & Grimson, W. (1999). Adaptive background mixture models for real-time tracking. In *IEEE Conference on Computer Vision and Pattern Recognition*, vol. 2. 96, 98

Stauffer, C. & Grimson, W. (2002). Adaptive background mixture models for real-time tracking. In *IEEE Conference on Computer Vision and Pattern Recognition*, vol. 2. 102

Steinwart, I. & Christmann, A. (2008). *Support vector machines*. Springer Verlag. 15

Swain, M. & Ballard, D. (1991). Color indexing. *International Journal of computer vision*, **7**, 11–32. 35

Szummer, M. & Picard, R. (1998). Indoor-outdoor image classification. In *IEEE International Workshop on Content-Based Access of Image and Video Database*. 29

Tanaka, Y., Okamoto, A., X-H, H. & Y-W, C. (2010). Scene image recognition with multi level resolution semantic modeling. In *International Conference on Software Engineering and Data Mining*, 644–647. 31

Vailaya, A., Jain, A. & Zhang, H. (1998). On image classification: City images vs. landscapes. *Pattern Recognition*, **31**, 1921–1935. 28

Vailaya, A., Figueiredo, M., Jain, A., Zhang, H., Technol, A. & Alto, P. (2001). Image classification for content-based indexing. *IEEE Transaction on Image Processing*, **10**, 117–130. 44

Viola, P. & Jones, M. (2001). Rapid object detection using a boosted cascade of simple features. In *IEEE Conference on Computer Vision and Pattern Recognition*. 68

Viola, P. & Jones, M. (2002). Robust real-time object detection. *International Journal of Computer Vision*, **57**, 137–154. xvi, 18

Visor (2010). http://imagelab.ing.unimore.it/visor/video_categories.asp. 82, 85, 111

Vogel, J. & Schiele, B. (2004). Natural scene retrieval based on a semantic modeling step. *Lecture notes in computer science*, 207–215. 44, 45

Wang, L., Shi, J., Song, G. & Shen, I. (2007). Object detection combining recognition and segmentation. In *Asian Conference on Computer Vision*. 14

Wang, Y. & Ji, Q. (2005). A dynamic conditional random field model for object segmentation in image sequences. In *IEEE Conference on Computer Vision and Pattern Recognition*. 26, 69, 96

Wang, Y. & Mori, G. (2009). Human Action Recognition by Semi-Latent Topic Models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **31**. 41

Wong, S., Kim, T. & Cipolla, R. (2007). Learning motion categories using both semantic and structural information. In *IEEE Conference on Computer Vision and Pattern Recognition*, 1–6. 41

Wu, B. & Nevatia, R. (2007). Improving part based object detection by unsupervised, online boosting. In *IEEE Conference on Computer Vision and Pattern Recognition*. 70

Wu, J. & Rehg, J. (2008). Where am I: Place instance and category recognition using spatial PACT. In *IEEE Conference on Computer Vision and Pattern Recognition*. 44

Yamato, J., Ohya, J. & Ishii, K. (1992). Recognizing human action in time-sequential images using hidden Markov model. In *IEEE Conference on Computer Vision and Pattern Recognition*, 379–385. 38

Yeh, T., Lee, J. & Darrell, T. (2009). Fast concurrent object localization and recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*. 18

Zhou, A., Cai, Z., Wei, L. & Qian, W. (2003). M-Kernel merging: Towards density estimation over data streams. In *Database Systems for Advanced Applications*, 285–292. 102

Zhu, L. & Yuille, A. (2006). A hierarchical compositional system for rapid object detection. In *Advances in Neural Information Processing Systems*, vol. 18, 1633. xvi, 21, 22