2010

Scene classification using spatial pyramid matching and hierarchical Dirichlet processes

Haohui Yin

MS Thesis Report

Scene Classification Using Spatial Pyramid Matching and Hierarchical Dirichlet Processes

Haohui Yin

Department of Computer Science

Rochester Institute of Technology

One Lomb Memorial Drive

Rochester, NY 14623-3464

Phone: 412-320-3629

E-mail: hxy2496@cs.rit.edu

Chair: Dr. Roger Gaborski

Reader: Dr. Peter Anderson

Observer: Yuheng Wang

http://www.cs.rit.edu/hxy2496

Contents

1. Abstract

The goal of scene classification is to automatically assign a scene image to a semantic category (i.e. “building” or “river”) based on analyzing the visual contents of this image. This is a challenging problem due to the scene images’ variability, ambiguity, and a wide range of illumination or scale conditions that may apply. On the contrary, it is a fundamental problem in computer vision and can be used to guide other processes such as image browsing, content-based image retrieval and object recognition by providing contextual information.

This thesis implemented two scene classification systems: one is based on Spatial Pyramid Matching (SPM) and the other one is applying Hierarchical Dirichlet Processes (HDP). Both approaches are based on the most popular “bag-of-words” representation, which is a histogram of quantized visual features.

SPM represents an image as a “spatial pyramid” which is produced by computing histograms of local features for multiple levels with different resolutions. “Spatial Pyramid Matching” is then used to estimate the overall perceptual similarity between images which can be used as a support vector machine (SVM) kernel. In the second approach, HDP is used to model the “bag-of-words” representations of images; each image is described as a mixture of latent themes and each theme is described as a mixture of words. The number of themes is automatically inferred from data. The themes are shared by images not only inside one scene category but also across all categories.

Both systems are tested on three popular datasets from the field and their performances are compared. In addition, the two approaches are combined, resulting in performance improvement over either separate system.

2. Introduction

Though humans understand a real-world scene quickly and accurately, it is not an easy task for computers to classify scenes automatically due to the scene images' variability, ambiguity, and a wide range of illumination and scale conditions that may apply. Scene classification is a fundamental problem in computer vision and provides contextual information to guide other processes, such as browsing, content-based image retrieval and object recognition. This problem has been widely explored and there are many different ways to solve it. After investigating the existing approaches by reading related research papers and analytically comparing their reported advantages in every aspect, I focused on two state-of-art approaches. One is based on Spatial Pyramid Matching (SPM) [5] and the other one is applying Hierarchical Dirichlet Processes(HDP) [7].

This thesis implements two scene classification systems using above two approaches separately. Several experiments are designed and made to compare their performances. Lastly, these two approaches are combined, resulting in performance improvement over either approach alone. The systems are tested on three different sets of images: a 6-class set containing natural images (mountains, forests, open country, coasts, lakes and sky), a 8-class set containing natural and city scenes, and a complicated 15-class set containing both outdoor and indoor scenes. A scene classification system requires a labeled set of training images, which are used to train the system how to classify images, and a labeled set of testing images, to verify performance. Classification accuracy is used as a metric to evaluate the performance of the system. This accuracy is simply the number of correct classifications divided by the number of attempted classifications during testing.

The rest of the report will be divided into the following sections. Section 3 will provide an overview of the field, covering several existing representative approaches. Section 4 will discuss the three popular datasets from the field. Section 5 will discuss the key concepts in detail, while Section 6 will provide the actual implementations. Section 7 will give the experimental results and the conclusion will be outlined in section 8.

3. Overview of the Field

The task of scene classification is to automatically assign a semantic category label to a new image, given a set of labeled images of scenes (for example, coast, forest, city, river, etc.). Scene classification is different than object recognition. Compared to an object class, a specific scene class exhibits a higher degree of variability, characterized by the presence of a large number of different visual entities. This fact has made scene classification quite a challenging problem. On the other hand, understanding scene context plays an important role in computer vision by providing contextual information to guild other processes such as browsing, content-based image retrieval and object recognition. Much research has been devoted to creating systems that enable automatic scene classification. Broadly speaking, the existing methods differ by: representation (how to represent a specific scene category or an image), learning (how to form the classifier given training data) and classification (how the classifier is to be used on novel data). Next, I will discuss several representative approaches in terms of these three aspects, in chronological order, reflecting how substantial progress has been made for the research of scene classification.

3.1. Oliva and Torralba [1] (2001 spatial envelope)

This paper is one of the first papers to address the problem of scene classification without the segmentation and the processing of individual objects or regions. Under the assumption that object information is not a necessary stage for achieving the scene identity level, it proposes to represent a scene image using a set of global image properties (naturalness, openness, roughness, expansion, and ruggedness). These five properties are known as the "spatial envelope" of a scene, and were chosen since humans use these same properties while performing a scene classification task. For a given dataset, training images need to be manually labeled with these properties in order to learn a Discriminant Spectral Template (DST) for each property. The DSTs are based on the Discrete Fourrier Transform (DFT) extracted from the whole image, or from a four-by-four gird. Each DST is able to describe how each spectral component of an image contributes to the corresponding spatial envelope property. Having learned DSTs for a given dataset, the scene attribute (the value of the five properties) of each image in the dataset can be estimated from its global spectral features, which are computed from DFT and thus are holistic as they encode the whole image without splitting it into objects or regions. Once each image has these properties, K-means clustering (with K equal to the number of image classes) is used to group the spatial envelopes into classes. To classify a new image with a known spatial envelope, it simply locates the nearest envelope (using Euclidean distance) and assigns the unknown image the class of this neighbor.

This paper firstly confirms the assumption that scene classification can be achieved by bypassing the segmentation, which is itself an open problem. It also proves the effectiveness of using an intermediate representation by analyzing low-level features in scene classification and

thus inspires future exploration on other more advanced intermediate representations. The dataset being used in this paper becomes standardized and is extensively used in many other papers (including this thesis). This dataset was formed using images from the Corel stock photo library, the authors' own pictures, and pictures from the web, and were all scaled to the same size, resolution and image format. Images span eight classes (mountain, forest, country, coast, highway, street, close-up and tall building), which were used as two separate 4-class problems (natural images and man-made images). Classification accuracy varies between 80-90% for each class.

One main requirement of this paper is the supervised learning of DSTS which involves time-consuming manually ranking each of the hundreds of training scenes into 5 different properties. Also the expert-defined labels are somewhat arbitrary and possibly sub-optimal. This motivates future research on methods for learning intermediate representations directly from the data.

3.2. Vogel and Schiele [2] (2004 Semantic modeling)

While the previous paper [1] represents an image using a low dimensional holistic descriptors based on global features extracted from the whole image, this paper presents an approach to find intermediate semantic models of natural scenes using local, region-based information. It assumes that humans rely on not only local, region-based information but also global, configural information. Both types of information seem to be significant to the same extent for humans to classify scenes.

It can be observed that some common local content are shared in images within a specific category. For example, pictures in the coast category contain mainly water and sand, whereas pictures in the forest category contain much foliage. Because of this fact, this paper came up with an approach to use this local semantic information as intermediate representation for natural scene images. For the 6-class natural scene dataset on which this paper has tested, it specified nine discriminant local semantic concepts: sky, water, grass, trunks, foliage, field, rocks, flowers and sand. Having these semantic concepts, this paper uses three steps to generate the image representation for final scene classification. Firstly, the scene images are divided into an even grid of 10x10 local regions, which are represented by a combination of a color and a texture feature. The color feature is a 84-bin HSI color histogram (H=36 bins, S=32 bins, I=16 bins), and the texture feature is a 72-bin edge-direction histogram. Secondly, through so-called concept classifiers (k-NN or SVM), the local regions are classified into one of the nine concept classes. In this step, a large number of local regions (59,582) of training images need to be annotated manually with the above semantic concepts. Thirdly, each image is finally represented by a concept occurrence vector (COV) which is computed as the histogram of the semantic concepts. To classify a novel image, its COV representation is used as input to an additional SVM to be classified.

This paper firstly shows that SVM outperforms k-NN. Later on, SVM becomes popular in scene classification. Its effective intermediate representation based on local information confirms its assumption that local information is as significant as global information for scene classification. It is another example to show that an appropriate intermediate representation is crucial for scene classification. The dataset used in this paper is characterized by a high degree of variability within a scene category. It becomes another standardized dataset and is used in this thesis. This dataset consists of six categories: coasts, rivers, forests, plains, mountains, and sky.

Like the previous paper [1], this paper needs to manually annotate a large number of local patches into one of nine different "semantic concepts" in order to train concept classifiers.

3.3. Quelhas et al. [3] (2005 BOV and pLSA)

Inspired by the "bag-of-words" (BOW) method in the field of text processing, this paper presents an analogous "bag-of-visterms" (BOV) method to represent a scene image. The construction of the "bag-of-visterms" (BOV) feature vector from an image involves three steps. The first step is to detect interest points automatically. Secondly, local descriptors are computed over the image regions associated with these points. Lastly, all local descriptors are quantized into visterms, and the histogram of visterms is computed to build the BOV representation of the image. This paper tested several different interest point detectors and local descriptors. The combination of Difference-of-Gaussians (DOG) detector and SIFT (Scale Invariant Feature Transform) descriptors was found to work best. SIFT features is known to be both scale- and rotation-invariant, as well as partially invariant to illumination changes, affine warps, and 3D viewpoint changes. The produced simple BOV representation is then used as input to a SVM to classify the corresponding scene image.

One obvious shortcoming of BOV representation is that since the resultant BOV representation is computed as the histogram of visterms, it does not contain the information about the visterm ordering and thus a significant amount of information about spatial layout of the original image is completely removed. Another restriction of BOV is that it cannot address synonymy (different visterms may represent the same scene type) and polysemy (the same visterm may represent different scene types in different contexts) ambiguities. Thus, this paper further proposes to use probabilistic Latent Semantic Analysis (pLSA) to group vistoms into a much smaller number of aspects (also called topics in other papers) and at the same time, it will be possible to combine synonymies into the same aspect. pLSA is a statistical model and it associates a latent variable

$$z_l \in Z = \{z_1, \ldots, z_{N_A}\}$$

where N_A is the number of aspects, with each observation (occurrence of a visterm in a image). It is necessary to define a few probabilities before we understand how pLSA works: $P(d_i)$ defines the probability of an image d_i, the conditional probabilities $P(v_j|z_l)$ represent the likelihood that a randomly selected vistom from topic z_l is the vistom v_j, and $P(z_l|d_i)$ gives the chance that a random vistom from image d_i belongs to the topic z_l.

Assuming that given an aspect z_l, occurrence of a visterm v_j is independent of the image d_i , the joint probability model over images and visterms can be defined as the mixture

$$P(v_j, d_i) = P(d_i) \sum_{l=1}^{N_A} P(z_l|d_i) P(v_j|z_l). \qquad (1)$$

The parameters of the model are estimated using the maximum likelihood principle. More precisely, given a set of training images D, the likelihood of the model parameters θ can be expressed by

$$L(\theta|D) = \prod_{d \in D} \prod_{j=1}^{N_v} p(v_j, d)^{n(d, v_j)}, \qquad (2)$$

where the probability model is given by Eq. 1 [3]. Then the Expectation-Maximization (EM) algorithm is used to maximize $L(\theta|D)$ and learn the aspect distributions $P(v_j|z_l)$ which are independent of images. Having learned $P(v_j|z_l)$, the aspect mixture parameters $P(z_l|d)$ of any image d can be inferred given its BOV representation $h(d)$. Consequently, the second representation of the image proposed by this paper is defined by

$$a(d) = (P(z_l|d))_{l=1 \ldots N_A}. \qquad (3)$$

Extensive experiments on two binary and four multi-class classification tasks (including 3, 5, 6 and 13 classes) show that the BOV approach performs well even in problems with a large amount of classes. Compared to BOV, pLSA deteriorates less as the training set is reduced and at the same time allows for dimensionality reduction by a factor of 17 for 60 aspects. But the performance of pLSA is lower than that one obtained with BOV in cases of a large amount of overlapping classes.

In sum, this paper firstly introduced an intriguing concept to represent a scene image as a mixture of aspects using pLSA. Such aspect-based representation can be learned from data automatically without time-consuming manual and possibly suboptimal labeling required in previous works in [1] and [2]. This paper is also one of the first papers demonstrating the superior performance of SIFT features in scene classification.

3.4. Fei-Fei L and Perona [4] (2005 A Bayesian Hierarchical Model LDA)

Like the work of Quelhas et al. [3], this paper investigates the joint use of local invariant descriptors and probabilistic latent aspect models. It models a scene category (note: not a scene image like in [3]) as a mixture of themes, and each theme is defined by a multinomial distribution over the quantized local descriptors (codewords). In this approach, local regions are first clustered into different intermediate themes, and then into categories. Probability distributions of the local regions as well as the intermediate themes are both learnt in an automatic way, bypassing any human annotation required in previous works [1], [2].

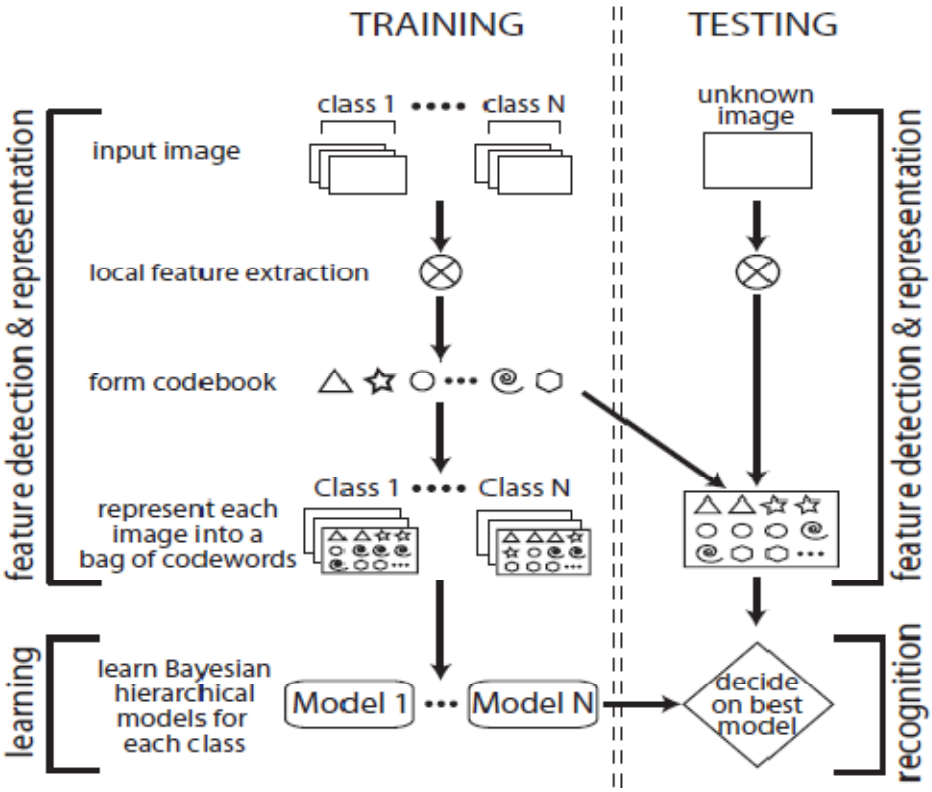

Figure 2. Flow chart of the algorithm.

From paper [4]

Fig.2 (from paper [4]) is a summary of the proposed algorithm in both learning and recognition. An image is modeled as a collection of local patches, each of which is represented by a codeword (like visterm in [3]) from a large vocabulary of codewords. The goal of learning is to achieve a model that best represents the distribution of the codewords in each scene category. In recognition, given an image to be classified, it first identifies all the codewords and then finds the category model that fits best the distribution of the codewords of that particular

image. This paper proposed a variation of Latent Dirichlet Allocation (LDA) to generatively model scene categories. The proposed model is called Bayesian Hierarchical Model. It differs from the basic LDA model by explicitly introducing a category variable for classification.

Different region (patch) detection processes and two kinds of local descriptors (128-dim SIFT vector and normalized 11x11 pixel gray values) were tested to build the codebook. The combination of evenly sampled gird regions spaced at 10x10 pixels and 128-dim SIFT vector was found to work best, outperforming the combination of the DOG detector and SIFT descriptors used in [3]. Thus, evenly sampled SIFT descriptors, being named as dense SIFT features, are extensively used in future research related to scene recognition. On the contrary, local descriptors detected using various feature detectors are called sparse local descriptors and they are mainly used in research related to object recognition.

Unlike all previous studies, this paper classifies a novel image based on the learned models for each category, rather than on k-NN in [1] or SVM in [2], [3]. When asked to categorize one test image, the category label that gives the highest likelihood probability is selected.

This work is very similar to that of Quelhas [3]. Both approaches combine local invariant descriptors (SIFT) with probabilistic latent aspect models. It is worthwhile to note a major difference regarding the way in which the aspect model is applied. This work learns a model for each scene category which can be used to classify a test image directly, whereas that of Quelhas [3] uses the aspect model to infer an image's aspect distribution, which is then used as input to SVM for supervised classification in a second step. Another major difference is: in this work, each training image must be labeled with the category name during learning, while in [3], the aspect representation of an image can be achieved in a fully unsupervised way, without class information.

Although both this work and that of Quelhas[3] are able to learn the intermediate representation automatically without the time-consuming manual labeling required in previous works of [1], [2], they need to specify the number of themes or aspects used in learning their latent aspect models. Researchers decided on this number based on extensive experiments. It might be hard to find a unique number which can be optimal for various datasets.

3.5. Lazebnik [5] (2006 Spatial Pyramid Matching)

The BOV method firstly introduced in [3] represents an image as an orderless collection of local features and thus disregards all information about the spatial layout of the features. In order to improve the severely limited descriptive ability of BOV, this paper presents a novel method based on aggregating statistics of local features over multiple levels with different resolutions. It represents an image as "spatial pyramid" which is produced by computing histograms of local features for multiple levels with different resolutions. The resulting "spatial

pyramid" is an extension of the standard bag-of-words representation. When only one level is considered, it reduces to the standard bag-of-words. Having "spatial pyramid" representation for each image, "spatial pyramid matching" is used to estimate the overall perceptual similarity between images, which then can be used as support vector machine (SVM) kernel.

Since this paper reported its both superior and reliable performance on several challenging scene categorization tasks including the Caltech-101 dataset (101 categories), I chose it as one method to implement the scene classification system. Its key concepts will be detailed in Section 5.

3.6. Sudderth and Torralba [7] (2008 HDP)

Latent aspect models like pLSA and LDA have previously been used to classify natural scenes successfully in the works [3], [4]. One limitation of such parametric models is that the number of latent topics must be specified. This choice is known to significantly impact predictive performance, and computationally expensive cross-validation procedures are often required. This paper proposes a different, data-driven framework for handling uncertainty in the number of latent topics, based on the Hierarchical Dirichlet Process (HDP) - a nonparametric alternative which avoids model selection by defining priors on infinite models. In nonparametric Bayesian statistics, Dirichlet Processes (DPs) are used to learn mixture models whose number of components is automatically inferred from data. A Hierarchical Dirichlet Process (HDP) describes several related datasets by reusing mixture components in different proportions and is used to model object categories for the first time in this paper.

HDP is a statistical model and is not easily understood due to its complicated underlying mathematical theories. Impressed by HDP's ability to model multiple grouped data, I am interested in applying HDP in our scene classification problem, with scene images in one category being considered as one group and all groups in one dataset being related. Section 5 will present the related mathematical formulas of HDP in detail.

4. Datasets

A very important part of one classification system is the dataset used to test it. In this thesis, the systems are tested on three popular datasets from the literature:

1. Oliva and Torralba [1]
2. Vogel and Schiele [2]
3. Lazebnik et al. [5]

We will refer to these datasets as OT, VS, LSP, respectively. Fig. 3 and Fig.4 show example images from each dataset and the contents are summarized here.

OT. Includes 2,688 images classified as eight categories: 360 coasts, 328 forests, 374 mountains, 410 open country, 260 high way, 308 inside of cities, 356 tall buildings, and 292 streets. The average size of each image is 250x250 pixels.

VS. Includes 700 natural scenes consisting of six categories: 142 coasts, 103 forests, 179 mountains, 131 open country, 111 rivers, and 34 sky/clouds. The size of the images is 720x480 (landscape format) or 480x720 (portrait format). Every scene category is characterized by a high degree of diversity and potential ambiguities since it depends strongly on the subjective perception of the viewer.

LSP. Contains 15 categories and is only available in gray scale. This data set consists of the 2,688 images (eight categories) of the OT data set plus: 241 suburban residence, 174 bedroom, 151 kitchen, 289 living room, 216 office, 315 store and 311 industrial. The average size of each image is approximately 250x300 pixels. The major sources of the pictures in the dataset include the COREL collection, personal photographs, and Google image search. This is one of the most complete scene category datasets used in the literature thus far.

For each experiment, the dataset needs to be divided into two separate groups; the training set and the testing set. To ensure that the systems generalizes well (that is, learn to identify a forest as opposed to 20 specific pictures of forests), there will be no overlap between training and testing sets. Experiments with different sizes of training sets are performed to find the relationship between the systems' performance and the training set size.

Fig. 3: Example Images from the data set VS (top) and OT (bottom)

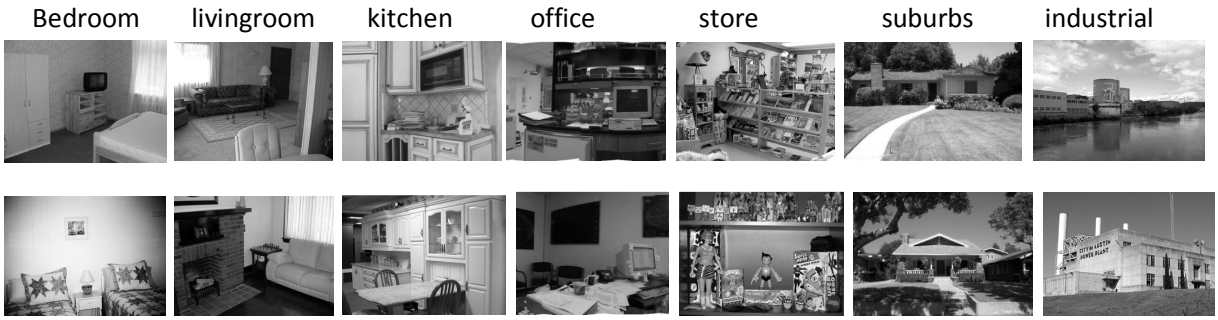

Fig. 4 Example images from the data set LSP (not including those from OT dataset)

5. Key Concepts

5.1. Support Vector Machines (SVMs)

SVM is a widely used approach to data classification that finds the optimal separating hyperplane between two classes. A classification task usually involves labeled training data and unlabeled testing data which consist of some data points in d-dimensional space. A set of d-length vectors and the associated class labels (0 or 1 for 2 class task) from the training data are used to train SVM, which then is able to classify a novel data point from the testing data into one of two classes.

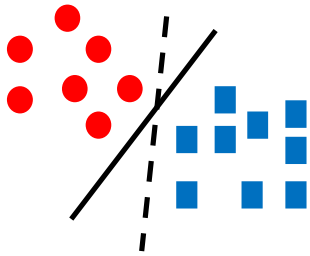

Fig 5: Two possible linear discriminant lines in a binary classification problem

To understand how a SVM works, let's start with a simple example of a binary classification problem in two dimensional data space, where the two data sets are linearly separable (i.e. there exists a line that correctly classifies all the points in the two sets). As shown in Fig 5, though both lines can separate all the data points, the darker line is more discriminant because it is furthest away from all points and small perturbations of any point would not introduce misclassification errors.

There are many ways to find the solid best line, and different SVM implementations will choose different methods. One approach is to find a supporting line for each class so that all points in that class are on one side of that line. The supporting lines are then pushed apart to maximize the distance or margin between them, until they bump into a small number of data points (the support vectors) from each class (see Fig. 6).

In real problems, the data is not always linearly separable, so SVMs must have some tolerance for error. Fig 7 shows an example where a single line cannot separate the two classes; however, the line is still a very good fit for most of the data with the minimum error.

Consider another binary classification problem in Fig. 8, where no simple line can approximate the separation between two classes. In this case, one solution is to map the data

into higher-dimensional space and then apply the existing linear classification algorithm to the expanded dataset in higher-dimensional space, producing a nonlinear discriminant circle in the original data space. For high-dimensional datasets, such kind of nonlinear mapping will cause the dimensionality of the data space exploding exponentially. SVMs get around this issue through the use of kernels, which measure similarity between two data points. Three of the most popular known kernels are given below:

- Polynomial: $K(X_i, X_j) = (\gamma {X_i}^T X_j + r)^d, \gamma > 0.$
- Radial basis function (RBF): $K(X_i, X_j) = exp(-\gamma \|X_i - X_j\|^2), \gamma > 0.$
- Sigmoid: $K(X_i, X_j) = tanh(\gamma {X_i}^T X_j + r)$

Here, X_i and X_j are two data points. γ, r and d are kernel parameters and their appropriate values need to be chosen by cross-validation. By using kernels, a linear SVM classifier can be easily turned into a highly nonlinear SVM classifier.

For more information about SVMs, please refer to [8], where Bennet and Campbell provide a brief overview of SVMs. This paper also provides links to books and websites with more information.

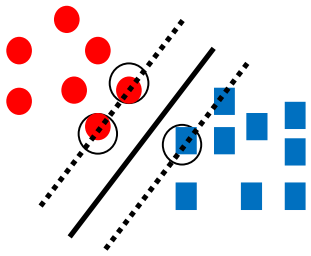

Fig 6: Best line maximizes the margin. Support vectors are outlined in circles.

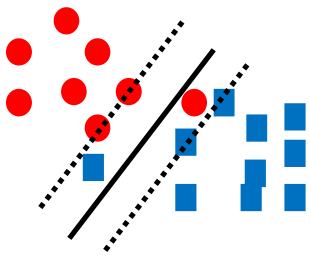

Fig 7: A binary classification problem which is not linearly separable.

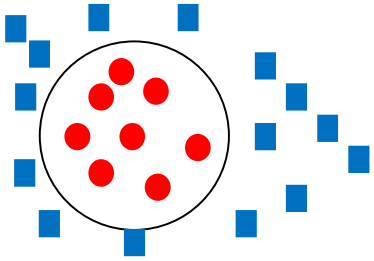

Fig 8: A binary classification problem which requires kernels to separate.

5.2. Spatial Pyramid Matching (SPM) [5]

Spatial Pyramid Matching [5] works by computing rough geometric correspondence on a global scale and it is in fact using an efficient approximation technique adapted from the pyramid matching scheme of Grauman and Darrell [6], which is described initially in the first part below. Then the second part will introduce how the pyramid matching is adapted to address our scene classification problem.

5.2.1 Pyramid Matching Scheme

Let X and Y be two sets of feature vectors in a d-dimensional feature space. Pyramid matching is to measure similarity between these two sets based on approximate correspondences found within a multi-resolution histogram pyramid. It repeatedly places a sequence of increasingly finer girds over the feature space to form the multi-resolution histogram pyramid. The similarity between two feature sets is then defined as the weighted sum of the number of feature matches at each level of the pyramid. At each pyramid level, two points are said to match if they fall into the same histogram bin; matches found at finer levels are weighted more highly than matches found at coarser levels. Let $H_X = (H_X^0, \dots, H_X^L)$ and $H_Y = (H_Y^0, \dots, H_Y^L)$ denote the histogram pyramid of X and Y at levels 0, …, L. Suppose the histogram at level l has 2^l bins (0th level is the coarsest whereas Lth level is the finest) along each dimension, thus the total number of bins will be $D = 2^{dl}$. $H_X^l(i)$ and $H_Y^l(i)$ are the numbers of feature matches from X and Y that fall into the ith histogram bin. Then the number of matches at level l is given by the histogram intersection function Eq. (4):

$$I(H_X^l, H_Y^l) = \sum_{i=1}^{D} \min\left(H_X^l(i), H_Y^l(i)\right). \qquad (4)$$

In the following, we will abbreviate $I(H_X^l, H_Y^l)$ to I^l.

Note that the number of matches found at level l also includes all the matches found at the finer level l+1. Therefore, the number of new matches found at level l is given by $I^l - I^{l+1}$ for l = 0, …, L-1. The weight associated with level l is set to $\frac{1}{2^{L-l}}$, which is inversely proportional to bin width at that level. Intuitively, since matches found in finer levels involve increasingly similar features, we want to weight them more than those newly found in coarser level. When all these levels of weighted histogram intersection are summed together, we get the following definition of a pyramid match kernel:

$$k^L(X, Y) = I^L + \sum_{i=1}^{L-1} \frac{1}{2^{L-l}} (I^i - I^{i+1}) \qquad (5)$$

$$= \frac{1}{2^L} I^0 + \sum_{i=1}^{L} \frac{1}{2^{L-i+1}} I^i \qquad (6)$$

5.2.1 Spatial Pyramid Matching Scheme

While the pyramid matching introduced above is performed in the feature space, this paper adapts it to perform pyramid matching in the two-dimensional image space, and use traditional clustering techniques in the feature space. Specifically, it quantizes all feature vectors into M discrete types (each corresponding to a visual word from the visual vocabulary), and assumes that only features of the same type can be matched to one another. For each channel m, we have two sets of two-dimensional vectors, X_m and Y_m , representing the horizontal and vertical position of features of type m found in respective images. Summing all channel kernels together, we get the final kernel:

$$K^L(X, Y) = \sum_{m=1}^{M} k^L(X_m , Y_m). \qquad (7)$$

In fact, the standard BOV image representation is a special case of the spatial histogram pyramid representation with L = 0. Thus, this approach has the advantage of maintaining continuity with the popular “bag-of-words” framework.

Because the pyramid match kernel Eq.(6) is simply a weighted sum of histogram intersection, and because $\mathrm{cmin}(a, b) = \min(ca, cb)$ for positive numbers, we can implement K^L as a single histogram intersection of “long” vectors formed by concatenating the appropriately weighted histograms of all channels at all resolutions (Fig. 9 from paper [5]).

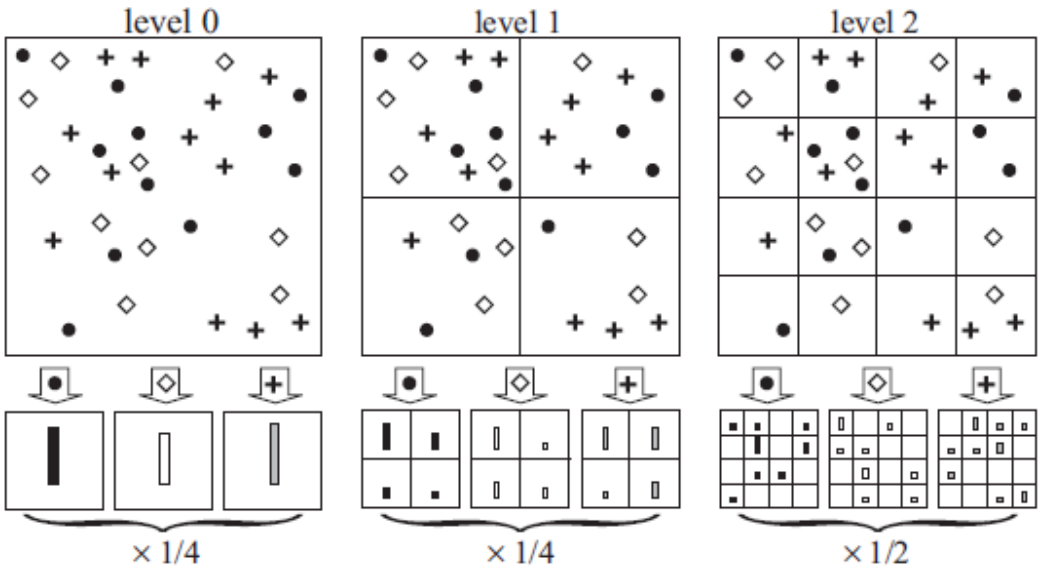

Figure 9 (from paper [5]). Toy example of constructing a three-level pyramid. The image has three feature types, indicated by circles, diamonds, and crosses. At the top, we subdivide the image at three different levels of resolution. Next, for each level of resolution and each channel, we count the features that fall in each spatial bin. Finally, we weight each spatial histogram according to Eq. (6).

5.3. Hierarchical Dirichlet Processes (HDP) [7]

Let's consider grouped data, where each group is associated with a mixture model and where there are also underlying links between these mixture models. The dataset for our scene classification problem is an example of grouped data, each scene category being a group. Data generated from Hierarchical Dirichlet Process (HDP) mixture models exactly satisfy the grouped data characteristic and thus can be used to model the dataset of our problem. An image from one scene category can be viewed as a collection of quantized local features and each feature can be viewed as arising from a number of latent topics, where a topic is generally modeled as a multinomial probability distribution on all features from a basic visual vocabulary. Topics are usually shared among images not only in one scene category but also across categories.

HDP is built on multiple DPs. We need to provide a brief overview of DP before discussing HDP.

5.3.1 Dirichlet process (DP)

DP is a stochastic process whose samples are probability measures with probability one. Let H be a probability measure on some parameter space θ. A Dirichlet Process (DP), denoted by DP(γ, H), is defined to be the distribution of a random probability measure over θ, where the scalar concentration parameter γ controls the amount of variability of samples $G \sim \mathrm{DP}(\gamma, H)$

around the base measure H, the larger γ, the less variability of G to H. For any finite measurable partitions (T1, …, Tl) of θ, the random vector (G(T1),…, G(Tl)) has a finite-dimensional Dirichlet distribution with parameters (γH(T1),…, γH(Tl)):

$$(G(T1),\ldots, G(Tl)) \sim \mathrm{Dir}(\gamma H(T1),\ldots, \gamma H(Tl)). \qquad (8)$$

Samples from DPs are discrete with probability one. This property is made explicit in the following stick-breaking construction [7]:

$$G(\theta) = \sum_{k=1}^{\infty} \beta_k \delta(\theta, \theta_k), \quad \beta'_k \sim \mathrm{Beta}(1, \gamma), \quad \beta_k = \beta'_k \prod_{l=1}^{k-1} (1 - \beta'_l). \qquad (9)$$

$\delta(\theta, \theta_k)$ is a probability measure concentrated at θ_k. Each parameter $\theta_k \sim H$ is independently sampled from the base measure. The weights $\boldsymbol{\beta} = (\beta_1, \beta_2, \ldots)$ use beta random variables to partition a unit-length "stick" of probability mass and satisfies $\sum_{k=1}^{\infty} \beta_k = 1$ with probability one. Thus $\boldsymbol{\beta}$ can be interpreted as a random probability measure on positive integers. For convenience, we write $\boldsymbol{\beta} \sim \mathrm{GEM}(\gamma)$ to denote a sample from this stick-breaking process.

One of the most important applications of DPs is as a nonparametric prior on the parameters of a mixture model with an unknown, and potentially infinite, number of components. Suppose that given $G \sim \mathrm{DP}(\gamma, H)$, observations x_i are generated as follows:

$$\overline{\theta_i} \sim G$$

$$x_i \sim F(\overline{\theta_i})$$

Where $F(\overline{\theta_i})$ denotes the distribution of the observation x_i given $\overline{\theta_i}$. $\overline{\theta_i}$ are conditionally independent given G and x_i is conditionally independent of the other observations given $\overline{\theta_i}$. Note that θ_k is used to denote the unique parameters associated with distinct mixture components, and $\overline{\theta_i}$ is used to denote a copy of one such parameter associated with a particular x_i. This model is referred to as a Dirichlet process mixture model. A graphical model representation of a Dirichlet Process mixture model is shown in Figure 10.

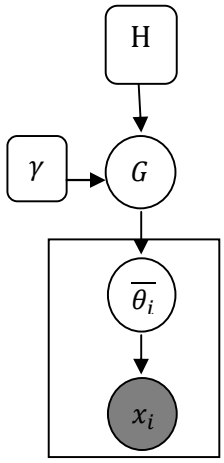

Figure 10: Dirichlet process mixture model. Each node is associated with a random variable, where shading denotes an observed variable. Rectangles denote replication of the model within the rectangle.

Since G can be represented using a stick-breaking construction Eq.(9), $\overline{\theta_i}$ take on values θ_k with probability β_k. For moderate concentrations γ, all but a random, finite subset of the mixture weights $\boldsymbol{\beta}$ are nearly zero, and data points cluster as in finite mixture models. Now suppose the number of distinct values taken by $\overline{\theta_i}$ is K.

To develop computational methods, we introduce an indicator variable z_i which takes on positive integral values and is distributed according to $\boldsymbol{\beta}$ ($z_i \sim \boldsymbol{\beta}$). z_i indicates the unique component of $G(\theta)$ associated with observation $x_i \sim F(\theta_{z_i})$. Integrating out G, these assignments $\mathbf{Z}$ exhibit an important clustering behavior. Letting N_k denote the number of observations already assigned to θ_k, the successive conditional distribution of z_i given $z_1, \ldots, z_{i-1}$ has the following form:

$$p(z_i|z_1, \ldots, z_{i-1}, \gamma) = \frac{1}{\gamma + i - 1}\left[\sum_k N_k \delta(z_i, k) + \gamma\delta(z_i, \bar{k})\right]. \qquad (10)$$

Here, $\bar{k}$ indicates a previously unused mixture component (a priori, all clusters are equivalent). This process is sometimes described by analogy to a Chinese restaurant in which the (infinite collection of) tables correspond to the mixture components θ_k, and customers to observations x_i. Customers are social. The ith customer x_i, sits at the table indexed by $z_i = k$, with probability proportional to the number of customers N_k already seated there (in which case we set $\overline{\theta_i} = \theta_k$), and sits at a new table with probability proportional to γ (increment K, draw $\theta_K \sim H$ and set $\overline{\theta_i} = \theta_K$).

5.3.2 Hierarchical Dirichlet Processes

Standard Dirichlet process mixture model can be used to model one scene category (coast) with an unknown number of mixture components (water, sand, sky, etc). In contrary, Hierarchical Dirichlet Process (HDP) is able to model multiple scene categories, where each category is associated with a mixture model which is shared among all categories.

A Hierarchical Dicichlet Process is a distribution over a set of random probability measures over parameter space θ. Let H denote a Dirichlet prior on feature parameter distributions. To construct an HDP, a global random probability measure $G_0 \sim \mathrm{DP}(\gamma, H)$ is first used to define an infinite set of shared topics:

$$G_0(\theta) = \sum_{k=1}^{\infty} \beta_k \delta(\theta, \theta_k), \quad \boldsymbol{\beta} \sim \mathrm{GEM}(\gamma), \quad \theta_k \sim H. \qquad (11)$$

Then, the process defines a set of random probability measures G_l for each scene category $l = 1, \ldots, L$. G_l are conditionally independent given G_0, with distributions given by a Dirichlet process with base probability measure G_0. In another words, $G_l \sim \mathrm{DP}(\alpha, G_0)$. The sample from G_l is a category-specific reweighting of the global topics from G_0, so we have:

$$G_l(\theta) = \sum_{t=1}^{\infty} \tilde{\pi}_{lt} \delta\left(\theta, \tilde{\theta}_{lt}\right), \quad \tilde{\pi}_l \sim \mathrm{GEM}(\alpha), \quad \tilde{\theta}_{lt} \sim G_0, t = 1, 2, \ldots \qquad (12)$$

Each local part t (see Eq.12) has parameters $\tilde{\theta}_{lt}$ copied from some global part $\theta_{k_{lt}}$, indicated by $k_{lt} \sim \boldsymbol{\beta}$. Aggregating the probabilities associated with these copies, we can also directly express each scene category's appearance via the distinct, global parts:

$$G_l(\theta) = \sum_{k=1}^{\infty} \pi_{lk} \delta(\theta, \theta_k), \quad \pi_{lk} \sim \sum_{t|k_{lt}=k} \tilde{\pi}_{lt}. \qquad (13)$$

Using Eq. 8, it can be shown that $\boldsymbol{\pi_l} \sim \mathrm{DP}(\alpha, \boldsymbol{\beta})$, where $\boldsymbol{\beta}$ and $\boldsymbol{\pi_l}$ are interpreted as measures on the positive integers. Thus $\boldsymbol{\beta}$ determines the average importance of each global part ($E(\pi_{lk}) = \beta_k$), while α controls the degree to which parts are reused across scene categories.

The HDP graphical model for scene classification is shown in Fig. 11. Let's see how an image j of scene category s_j containing N_j features w_j is generated from this model. A scene image includes a collection of spatially constrained parts which are represented as groups of features that are spatially clustered, and have predictable appearances. Each of the L scene categories is characterized by a probability distribution $\boldsymbol{\pi_l}$ over a global set of potentially infinite shared parts. Each distinct part generates a different typical feature w_{ji}. Each feature is associated with

a particular part in $\mathbf{z_j} = \left(z_{j1}, \ldots z_{jN_j}\right)$ which are independently sampled from a category-specific probability distribution $\boldsymbol{\pi}_{\mathbf{s_j}}$, so that $z_{ji} \sim \boldsymbol{\pi}_{\mathbf{s_j}}$. Each part is then defined by a multinomial distribution η_k on the discrete set of W descriptors:

$$w_{ji} \sim \eta_{z_{ji}}, \qquad (14)$$

In other words, each feature w_{ji} is generated by choosing a part $z_{ji} \sim \pi_{s_j}$, and then sampling from the part's feature distributions, as in Eq. 14. Marginalizing these unobserved assignments of features to parts z_{ji}, image feature is defined by an infinite mixture model:

$$p\left(w_{ji} \middle| s_j = l\right) = \sum_{k=1}^{\infty} \pi_{lk} \eta_k(w_{ji}). \qquad (15)$$

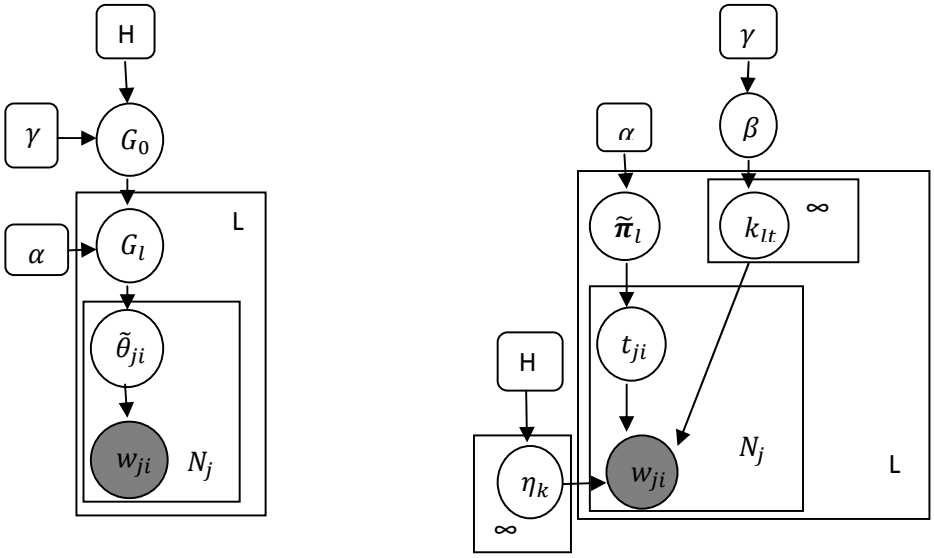

Figure 11 Nonparametric, hierarchical DP model for the visual appearance of L scene categories. Left: $G_0 \sim DP(\gamma, H)$ defines an infinite set of global parts, and scene categories reuse those parts via the reweighted distribution $G_l \sim DP(\alpha, G_0)$. $\tilde{\theta}_{ji} \sim G_l$ are then the part parameters used to generate feature w_{ji}. Right: Equivalent, Chinese restaurant franchise representation of the HDP. The explicit assignment variables k_{lt}, t_{ji} are used in Gibbs sampling algorithms

The HDP can also be represented by Chinese restaurant franchise which is an analog of the Chinese restaurant process for DP. The graphical model of the Chinese restaurant franchise representation for HDP is demonstrated in Fig.11 (right). In this interpretation, the Chinese restaurant process is extended to allow multiple restaurants which share a set of dishes. In our

application, a restaurant represents one specific scene category, customers (observed features w_{ji}) sit at tables (clusters or parts) t_{ji} which share a single dish (feature parameter) $\tilde{\theta}_{lt}$ being ordered from a global menu G_0 shared among restaurants (scene categories). Let $\mathbf{k_l} = \{k_{lt}\}$ denote the global parts assigned to all tables (local parts) of category l. We may then marginalize G_0 and G_l, as in Eq 10, to find the conditional distribution of these assignment variables:

$$p(t_{ji}|t_{j1}, \dots, t_{ji-1}, \alpha) \propto \sum_t N_{jt}\delta(t_{ji}, t) + \alpha\,\delta(t_{ji}, \bar{t}), \qquad (16)$$

$$p(k_{lt}|k_1, \dots k_{l-1}, k_{l1}, \dots, k_{lt-1}, \gamma) \propto \sum_k M_k\delta(k_{lt}, k) + \gamma\,\delta(k_{lt}, \bar{k}). \qquad (17)$$

Here, M_k is the number of tables previously assigned to θ_k, and N_{jt} is the number of customers already seated at the t^{th} table in group j. As before, customers prefer tables t at which many customers are already seated (see Eq. 16), but sometimes choose a new table $\bar{t}$. Each new table is assigned a dish $k_{l\bar{t}}$ according to Eq 17. Popular dishes are more likely to be ordered, but a new dish $\theta_{\bar{k}} \sim H$ may also be selected. In this way, scene categories sometimes reuse parts from other scenes, but may also create a new part capturing distinctive features.

The same as the work of Fei-Fei L and Perona [4], this approach learns a model for each scene category which are used to classify a given test image directly and quickly. It does not need to store any training data for classification, whereas both K-nearest neighbor and SVM must typically retain a large proportion of training images for later testing. As scene recognition systems are applied to larger datasets, such savings in storage and computation become increasingly important. Note that while each scene category's model in [4] is independent with each other, the models learned for each scene category in HDP framework are closely related by sharing the mixture components from the basic DP. Also this approach is able to infer from the data the number of mixture components which on contrary needs to be specified in [4]. Please refer to [12] for more details about DPs and HDPs.

6. Implementations

This section will cover the actual implementation of the two scene classification systems. The first part will present the system based on SPM, while another system applying HDP will be discussed in the second part. Both systems are implemented in MATLAB. In addition, two approaches are combined to produce improved performance.

6.1. The SPM-based system

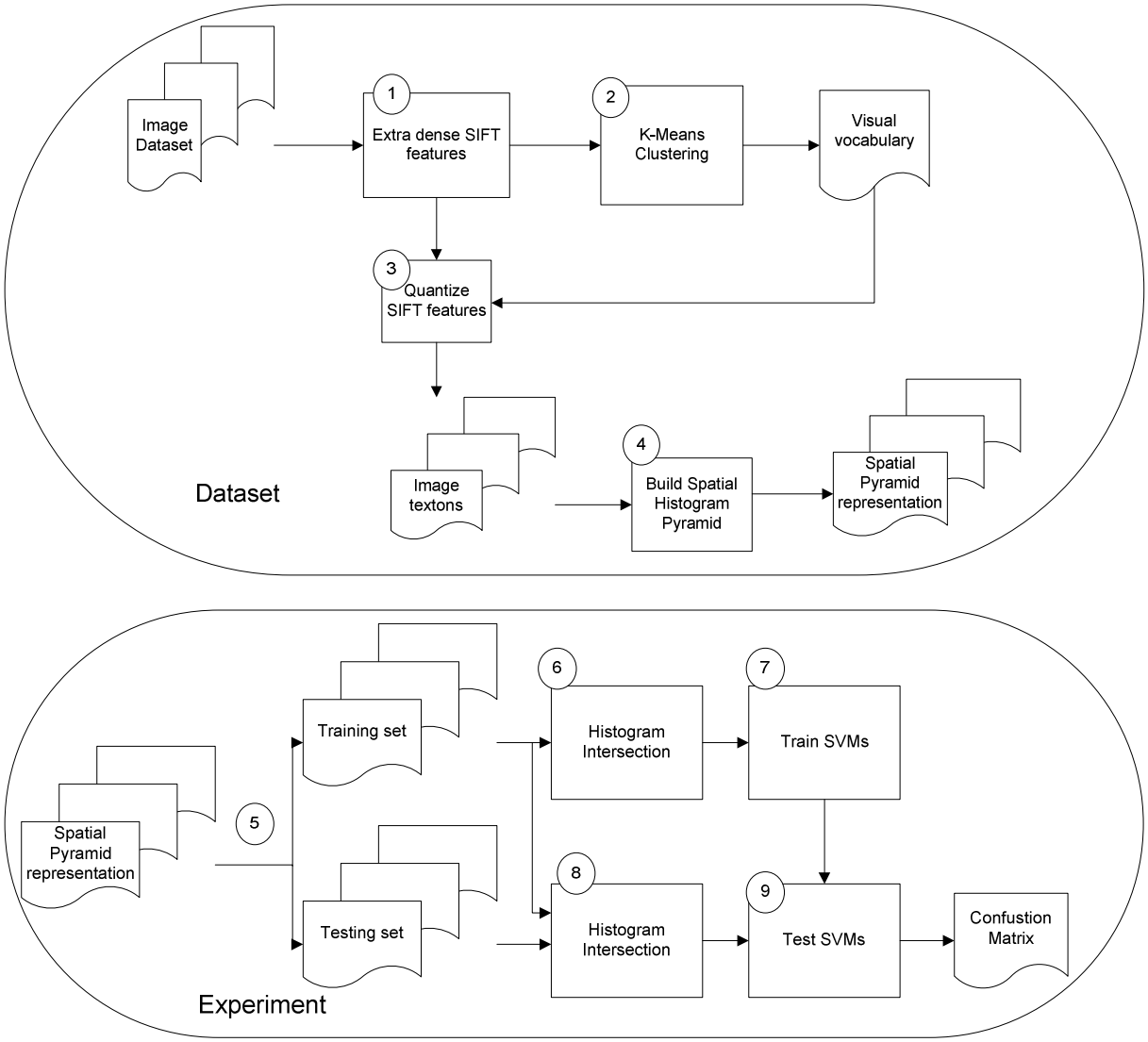

Fig. 12: A summarized architecture for SPM-based system. After the "Dataset" portion is run once for the dataset, multiple experiments can be run with just the "Experiment" portion.

The basic architecture of the SPM-based system is illustrated in Fig. 12.

First, given a whole dataset, function *GenerateSceneSiftDescriptor*s is used to extract the dense SIFT descriptors for all images in the dataset (see component indicated by 1). Dense SIFT descriptors allow us to have features spread out across the entire image evenly to provide a good idea of which textures are present. This is necessary to capture uniform regions, such as sky, calm water, or road surface which are significant visual contents in scene images. This system is designed to compute dense SIFT descriptors of 16x16 pixel patches over a grid with spacing of 8 pixels. Since it is reported that color SIFT features worked only slightly better than gray SIFT features but with 2 times more computation and storage space, each color image is converted into gray scale in order to much speed up the system, and at the same time reduce a lot of required space. Note that for a 256x256 pixel image, nearly 1024 features are produced, each with 128 floating point values. This can produce a large amount of data for even a reasonable sized dataset, so the results for each image are written to a separate file to purge RAM, as well as provide backups in case the process is stopped. The time for processing a 720x480 image in this step is about 5 seconds.

The huge set of all SIFT descriptors from the whole dataset are too varied, and must be reduced to build a general visual vocabulary, which is done by clustering the descriptors. K-means clustering is a widely used method to do clustering, since it is relatively quick and allows the user to choose the desired number of clusters. In order to prevent the RAM being used up by the immense set of all descriptors from the dataset, only descriptors from a random subset of images (about 50 for each scene category) are clustered, and the number of descriptors is restricted to no more than 100,000. The number of clusters (i.e. the size of the vocabulary) needs to be known beforehand. In [5], the authors reported that 200 clusters produced almost the same results as that of 400 clusters. Thus we use 200 in our k-means clustering (see component 2). This step is implemented in *CalculateDictionary.m* and takes about 5 minutes.

After the visual vocabulary is produced using a subset of the images, all the SIFT descriptors from each image will be quantized to the nearest visual word in the vocabulary with the minimum distance. The outcome of this step is that each image is represented as a vector of texton labels, associating each descriptor in the image to the corresponding visual word. One texton label can be stored in a 16-bit unsigned integer, whereas each element of one 128-dim descriptor needs to be stored as a 64-bit floating value, totaling 128*64 bit space per descriptor. Since the amount of data is reduced by a factor of 512, the speed of subsequent processing on textons instead of SIFT descriptors is largely increased (component 3). The time of processing one image in this step is about one second. The texton labels for each image are saved into a separate file.

Component 4 will build the spatial histogram pyramid for each image based on its texton labels. Given the number of pyramid levels L, it firstly divides the image into 2^L x 2^L sub-regions and computes histogram for each sub-region. Then from l = L-1 to 0, the image is divided into 2^l x 2^l sub-regions. For each sub-region, its histogram is computed as the sum of the histograms of the four sub-regions at the level l+1 which occupy the same image space as itself. The resulted histogram at level l is also weighted by 2^{-l}. The final spatial histogram pyramid is then generated by concatenating all weighted histograms from level L to level 0, producing a very long vector. The length of the vector for all images in the dataset is the same. In the case where vocabulary size is 200 and the number of pyramid levels is 2, the length will be 200 + 200*(2*2) + 200 * (4*4) = 4200. Each image's final spatial histogram pyramid is also saved into a separate file. This step is performed very quickly, since it only involves counting the number of each visual word label being occurred in the image's texton labels.

Once the previous expensive operations are completed, only the spatial histogram pyramids are necessary to fully test the classification system. First, the set of images is divided into two sets: the training set and the testing set (component 5). Then the spatial histogram pyramid of training images are used to evaluate the kernel values between every two training images (component 6). Having the spatial histogram pyramid vectors of two images, the kernel between these two images is computed as the sum of the minimum of each corresponding values (histogram intersection) from the two vectors. The resulted kernel matrix for all training images and the associated category labels are then used to train SVMs (component 7). Finally, the kernel matrix between all test images and all training images are evaluated in the same way as that between all training images (component 8), and the trained SVMs use this kernel matrix to classify all test images (component 9). The confusion matrix and accuracy for this run are printed to the screen.

SVMs are implemented by the SVM library called LIBSVM [9]. Since SVMs are used to solve binary classification problems, a one-vs-all method is employed to train SVMs for multi-class problems. In this scheme, each SVM is trained to discriminate one class from all others; in testing, the kernel values between one test image and all training images are run through each trained SVM classifier, and the classifier with the strongest response is selected as the winner. LIBSVM has built-in support for four kernel functions. If we choose to use one built-in kernel, SVMs needs to take as input the spatial histogram pyramid (i.e. a vector with length = 4200 for vocabulary size = 200) of all training images and then compute the kernel between every two training images using the chosen kernel function, in order to be trained. This is the same for testing SVMs. In our system, since the kernel of SVMs is precomputed using the very simple pyramid matching scheme and it is the only necessary input to train and test SVMs, SVM classification becomes much more efficient in terms of speed and space.

This system is implemented mainly based on the prototype provided by the author which can be downloaded from http://www.cs.unc.edu/~lazebnik/. This prototype implemented most operations in the Dataset portion, producing a spatial histogram pyramid for one image without taking care of a large dataset. It also does not include functionalities in the Experiment portion. I integrated the LIBSVM library into this prototype and also added necessary functions to make the system able to run on large scale dataset smoothly.

6.2. The HDP-based system

Fig. 13 demonstrates the basic architecture of the HDP-based system.

The functionalities provided by component 1, 2 and 3 in the Dataset portion are the exactly same as that of SPM-based system. But they are implemented using the Vlfeat open source library [10], which is written in C for efficiency and compatibility, with interfaces in MATLAB for ease of use. The vocabulary size is set as 600, and the time on building the vocabulary is about 10 minutes.

The Experiment portion is performed on image textons. After images are divided into a training set and a testing set (component 4), a straightforward Gibbs sampler based on the Chinese restaurant franchise runs on training images' textons to infer the HDP model for the dataset (component 5). The sampler involves two kinds of sampling: one is to sample feature assignments $\mathbf{t}_j$ for each training image j, and another one is to sample table or local part assignments $\mathbf{k}_l$ for each scene category l (see Fig 11 right). One iteration of sampling is performed in two stages. In the first stage, each training image j is considered in turn and its feature assignments $\mathbf{t}_j$ are resampled. The second stage then examines each scene category l, and samples assignment $\mathbf{k}_l$ of local to global parts. At all times, the statistics of global parts are updated accordingly. The sampler maintains dynamic lists of those tables to which at least one feature is assigned, and the global parts which are associated with these tables. These lists grow when new tables or global parts are randomly chosen and shrink when a previously occupied table or global part no longer has assigned features. Such sampling will be iterated by 200 times. The learned HDP model is finally characterized by the number of local parts (tables) in each scene category (restaurant), the assignment of each local part (table) to global part (menu), and the statistics of each global part. Gibbs sampling is very time-consuming, since one iteration of sampling needs to resample all features in all training images, one feature at a time. For a training size of 50 images per category and 15 categories, it takes about 2 days to learn the HDP model. Actually research on how to speed up the inference of HDP model is an active topic.

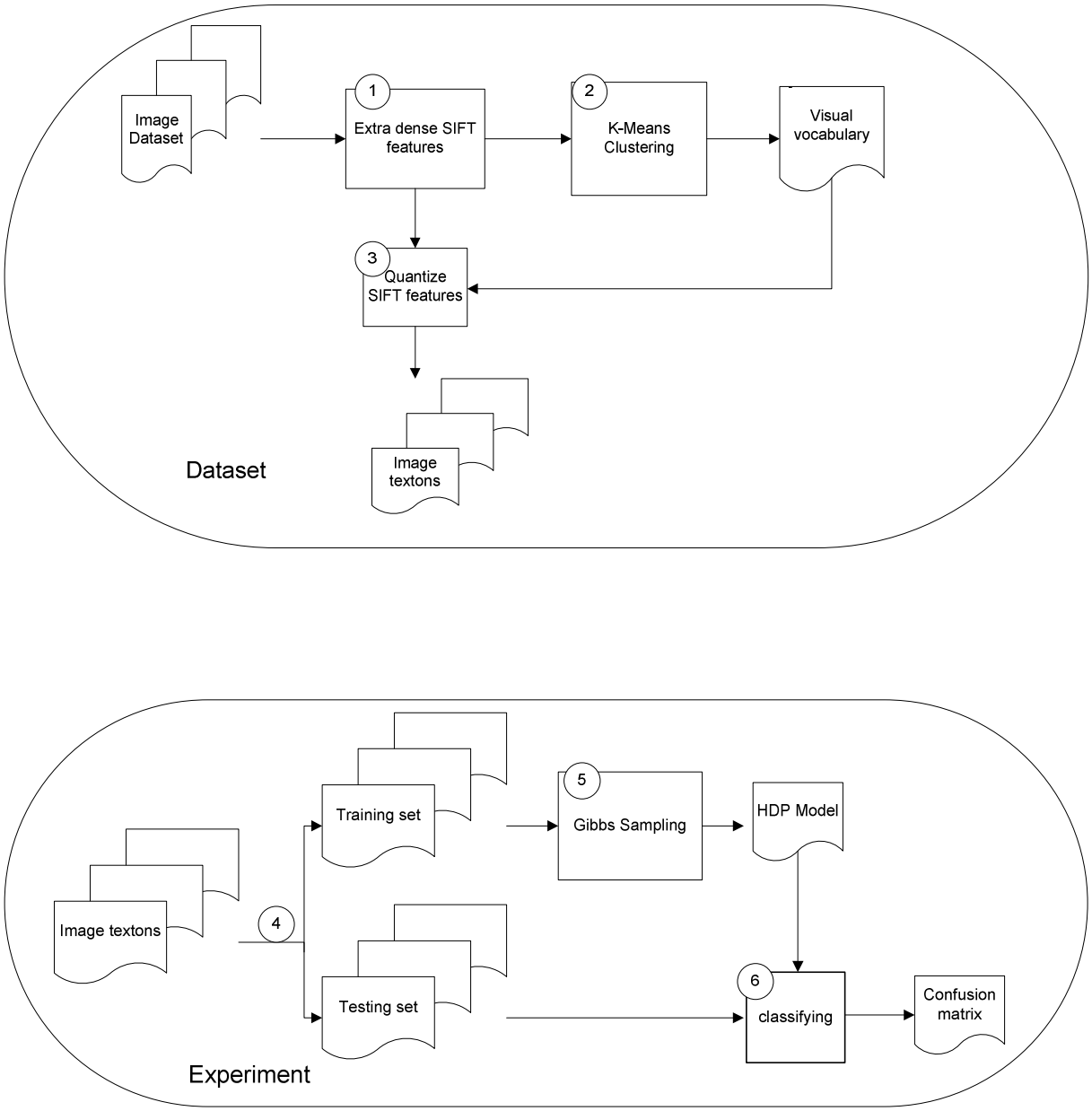

Fig. 13: A summarized architecture for HDP-based system. After the “Dataset” portion is run once for the dataset, multiple experiments can be run with just the “Experiment” portion.

Having learned the HDP model, the likelihood of a given test image is estimated for each scene category using current assignments of the final HDP model, and the category label with the maximum likelihood is assigned to the test image (component 6). Specifically, it involves two steps to compute the likelihood of a test image for one scene category: the first step is to assign each feature of the test image to the local part, which is associated to a global part with the maximum probability to generate this feature; then the likelihood that this test image belongs to this specific category is given by summing up the probabilities of all features in this image. Note that this operation takes as input only the test data and the learned HDP model which is characterized by a small compact set of parameters. It doesn't need to store any training data for classification. A test image is thus classified very quickly.

The implementation of Gibbs sampler is provided by the author who applied it in the object classification task. Please refer to http://www.cs.brown.edu/~sudderth/software.html for the original code.

6.3. The combined system

For a given test image, the SPM-based system gives the classification response from each scene category while the HDP-based system gives the likelihood that this test image belongs to each scene category. These two results are consistent for most test images, where any weighted sum of them will enhance the correct classification. On the other hand, some test images are misclassified in one system, but are classified correctly in another system. In this case, it is possible that the sum of the two appropriate weighted results will give the correct classification. Through the experiments, the combination below give improved performance on all three datasets:

Combine_resul = 0.05 * HDP_result + 0.95 * SPM_result

7. Results

7.1. The SPM-based system

In this section, we report the SPM-based system's performances against the number of spatial pyramid levels L. Table 1 shows results of classification experiments using %50/50% training/testing split, which is the same as David's [11] setting. The vocabulary size is given by M. For VS and LSP datasets, results improve dramatically (more than 5%) from L = 0 (standard BOV) to a multi-level setup. For the OT dataset, the improvement is slight because its performance with L = 0 is already good enough. We believe that for the OT dataset, the performance improvement of multi-level spatial pyramid over standard BOV will become more obvious with less training data. It is interesting to notice that the performance drops as we go from L = 2 to L =3 for all three datasets. This means that the highest level of the L = 3 pyramid is too finely subdivided, with individual spatial bins yielding too few matches. Generally L = 2 will give the best result, in terms of both classification accuracy and the amount of computation. Let us look at the performance against different vocabulary size for LSP. Increasing the size from M = 200 to M = 300 results in a small performance increase at L = 0, but this difference is much less at higher pyramid levels. Thus we can conclude that the geometric cues provided by the pyramid have more discriminative power than an enlarged visual vocabulary. David's system has been tested on VS and OT, and the reported results were attained using the standard BOW method. Compared to his results, our system gives much better performance for OT, but only slightly better for VS. This might confirm the fact that VS dataset is much more ambiguous and is much harder for classification, even by humans.

Table 1: Classification accuracies with the number of pyramid levels being 0, 1,2 and 3, respectively.

L	VS (M=200) David(65.9)	OT (M=200) David(83.1)	LSP (M=200)	LSP(M=300)
0	60.8	86.53	76.08	78.13
1	**66.2**	87.35	80.34	80.78
2	65.06	**88.02**	**81.43**	**81.87**
3	63.92	87.20	81.024	81.51

7.2. The HDP-based system

Let us examine how the learned HDP model is consistent with the dataset being modeled. A HDP model for one dataset is characterized by the number of global parts for the whole dataset and the number of local parts for each scene category, each local part being associated with a global part. Table 2 to Table 5 shows these numbers for VS, OT and LSP, respectively. With the number of categories increasing from VS (6) through OT (8) to LSP (15), their HDP model's number of global parts is increased from 38 through 50 to 91. This proves the HDP model's ability in automatically inferring the number of global mixtures from the data consistently. It is also observed that the number of local parts in the forest category is the smallest in all three datasets. This reflects the fact that the forest category exhibits lowest degree variability and thus is the easiest to be classified among all categories. The classification accuracies for forest category are 88.68% (VS), 90.65% (OT), 90.29% (LSP), all being the highest in each dataset.

Table 2: VS: number of local parts in each category (number of global parts = 38)

coast	forest	mountain	opencountry	river	sky
14	3	15	11	14	9

Table 3: OT: number of local parts in each category (number of global parts = 50)

coast	forest	mountain	opencountry	highway	insidecity	street	tallbuilding
10	5	11	9	13	12	14	10

Table 4: LSP: number of local parts in the 8 categories from OT (number of global parts = 91)

coast	forest	mountain	opencountry	highway	insidecity	street	tallbuilding
6	4	9	8	17	16	13	17

Table 5: LSP: number of local parts in the remaining 7 categories (number of global parts = 91)

bedroom	livingroom	kitchen	office	store	suburb	industrial
17	13	18	18	13	10	15

7.3. Comparison among the three systems

Experiments are conducted to establish the relationship between the performance and the training size. For each different training/testing split, the experiments are performed on the SPM-based system, the HDP-based system and the combined approach as well. Systems are tested with the number of training images per category being from 10 to 50, spaced in units of 10.

7.3.1. VS

The comparison for VS is shown in Fig. 14. Classification precision is computed as the number of test images being correctly classified, divided by the number of test images being tempted. The red lines are attained, assuming that a test image is classified correctly only if the classified category with the highest probability is the truth category. The blue lines are attained, when assuming that a test image is classified correctly if its truth category falls in the categories with the first two highest probabilities.

From Fig. 14, it is observed that the performance of any system is increasing with the training size. For all training size, the SPM-based system outperforms the HDP-based system and the combined approach outperforms either system.

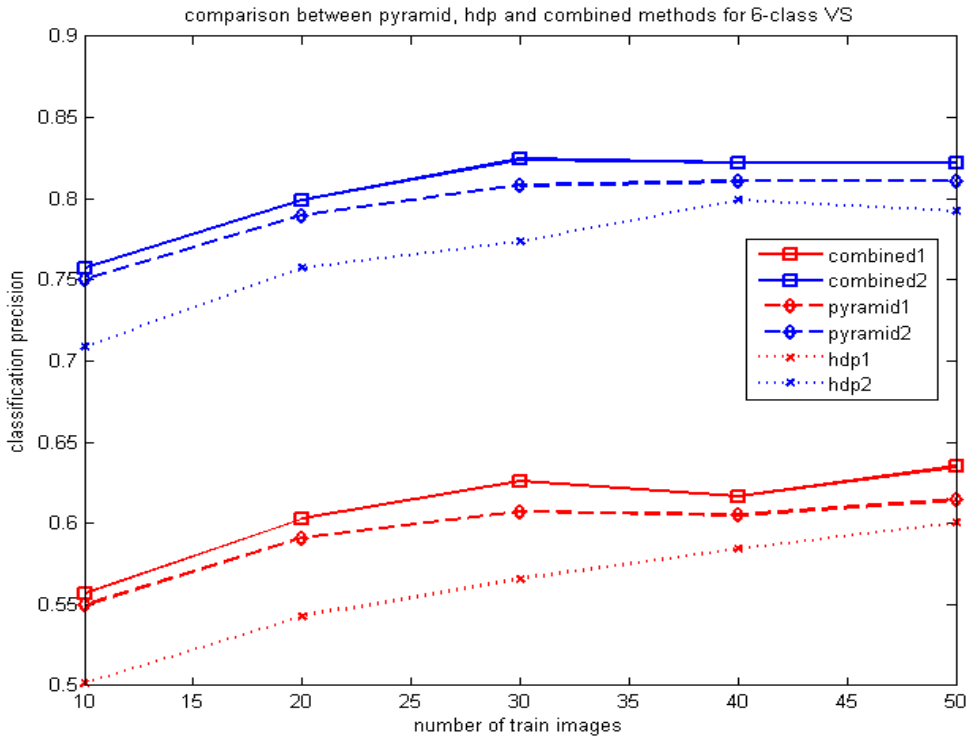

Fig. 14: comparison for VS

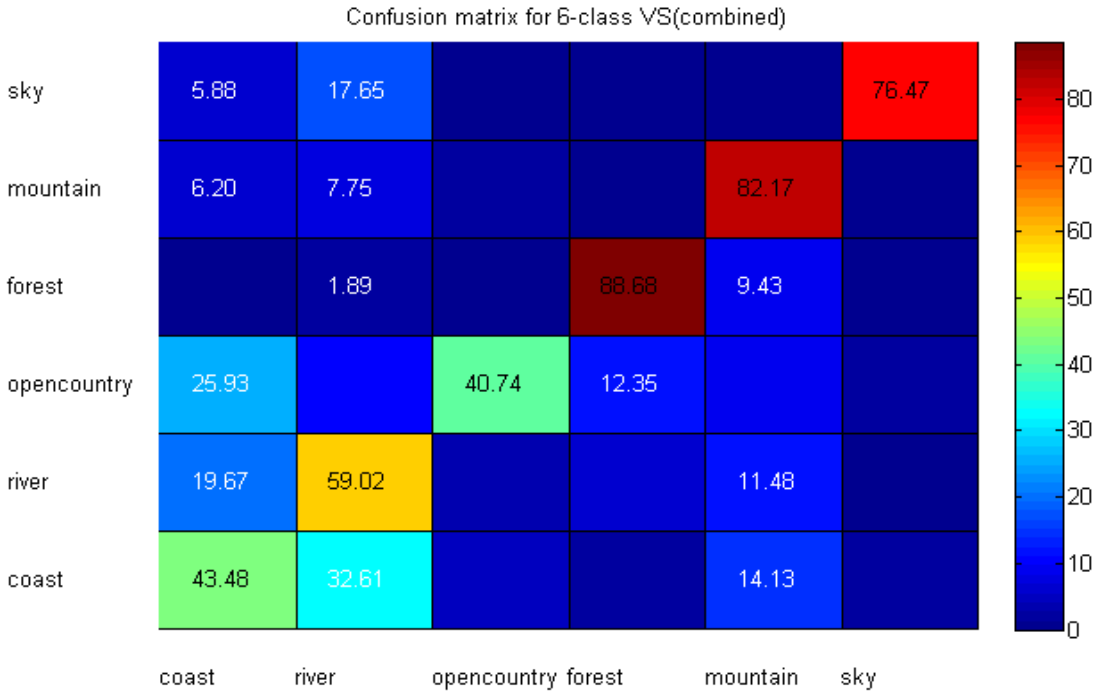

Fig. 15: Confusion matrix for VS with overall accuracy of 63.51(combined)

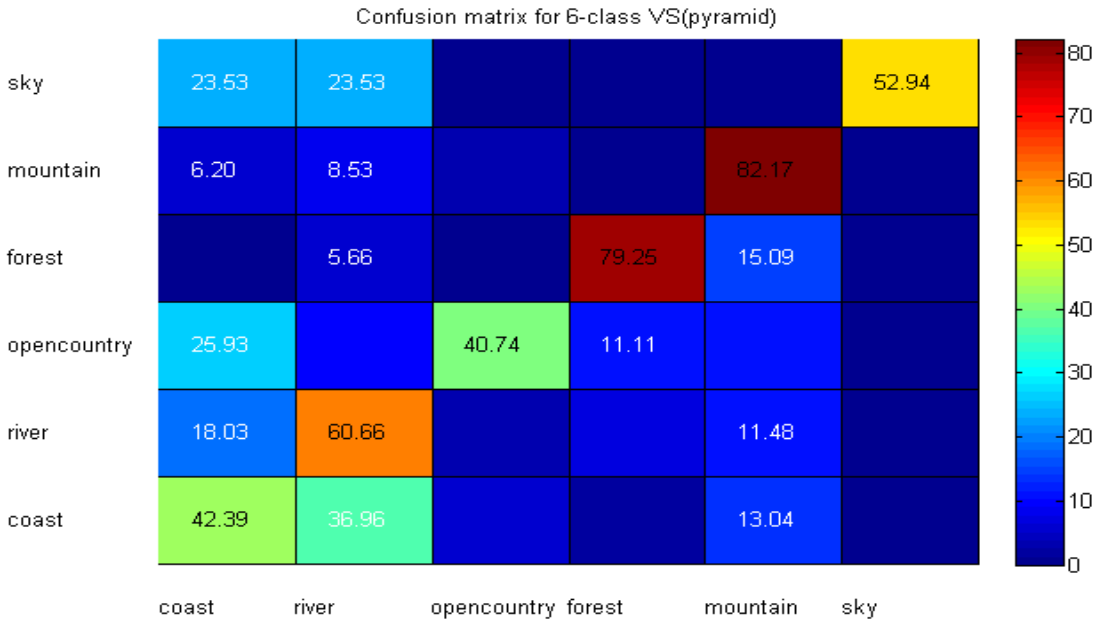

Fig. 16: Confusion matrix for VS with overall accuracy of 61.43(SPM)

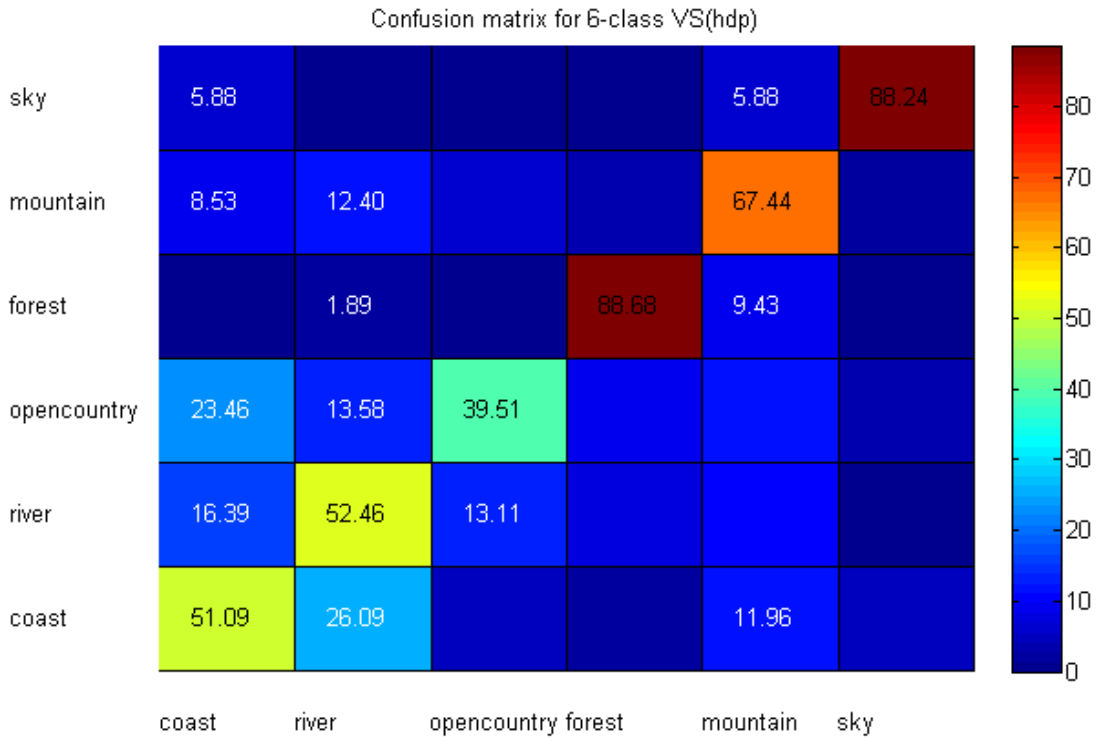

Fig. 17: Confusion matrix for VS with overall accuracy of 60.05(HDP)

The detailed results of each category for each system are presented in the form of a confusion matrix (see Fig. 15 to Fig. 17 for the combined system, the SPM-based system and the HDP-based system, respectively. The training size is 50 images per category.). A confusion matrix is a square matrix where each row and column represents one image category, and entries denote the percentage chance that the specified image category will be assigned a certain label. In our matrices, the rows are the truth categories and the columns are the labels assigned by the system. Entries along the diagonal represent correct classifications, so the ideal confusion matrix would be the identity matrix. Only the three highest percentage numbers for each category are listed in the confusion matrix.

From Fig. 15, it is obvious that coast images and the river images are mostly confused with each other. This is consistent with our observation that both the river images and the coast images are very similar, and they both are characterized with a large area of water visual content. The classification accuracy of the opencountry category is the lowest since there is not a single texture which is unique to this category. The opencountry images often contain grass, foliage, sand and rock etc. (the identifying features of other categories) with the almost same amount. The examples in Fig. 18 reflect a high degree of variability presented in the opencountry category. For the river images, in addition to be mostly confused with coast images, they are also easily misclassified into mountain because most river images contain mountains. Fig. 19 shows both the correctly classified river images and the misclassified river images. A lot of confusions also occur between forest&mountain and mountain&river due to the fact that most forest images have mountains and most mountain images present rivers. Examples of these kinds of misclassification are shown in Fig. 20.

Fig. 18: Opencountry images from VS. Images along the top are classified correctly, while images below are misclassified into coast, forest, mountain and river, respectively.

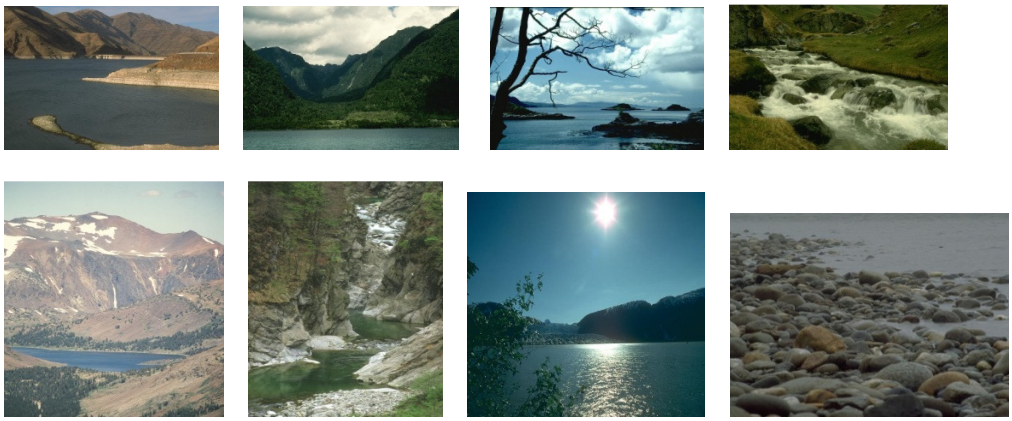

Fig. 19: River images from VS. Images along the top are correctly classified, while the first two and the last two images below are misclassified into mountain and coast, respectively.

Fig. 20: The first two forest images are misclassified into mountain, while the last two mountain images are misclassified into river images.

In sum, VS is a pretty hard dataset for scene classification mainly due to the difficult open country category which exhibits a very high degree of diversity and the very ambiguous categories of coast & river.

7.3.2. OT

Fig. 21 is the comparison for OT, which exhibits the same characteristics as that of VS.

From Fig. 22 to Fig. 24, the matrix confusion tables for the combined system, the SPM-based system, and the HDP-based system are illustrated. Compared to VS, OT has much more improved performance though it has more categories. Without the ambiguous river category which is introduced in VS, the classification accuracies for all four natural categories are much higher than that of VS. It is still observed that the open country is the most difficult category among all categories due to its high degree variability. It is worth to note that almost all confusions are mainly located inside the 4 natural categories or the 4 man-made categories, except the highway category which has the almost same extent confusion with the coast (3.33%), the mountain (3.81%) and the street (3.12%) (see Fig. 25 for examples). This can be explained by the fact that highways are presented in both natural scenes and in cities. Thus, our

systems can be used to classify OT dataset into the natural scene group, which is mainly characterized with rock, foliage, water, sand etc, and the man-made group, which is mainly made of road, brick, metal, glass etc.

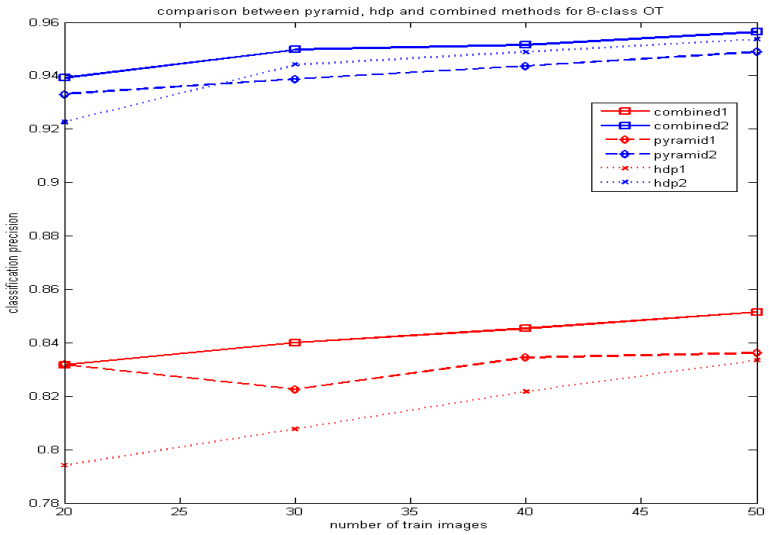

Fig. 21: comparison for OT

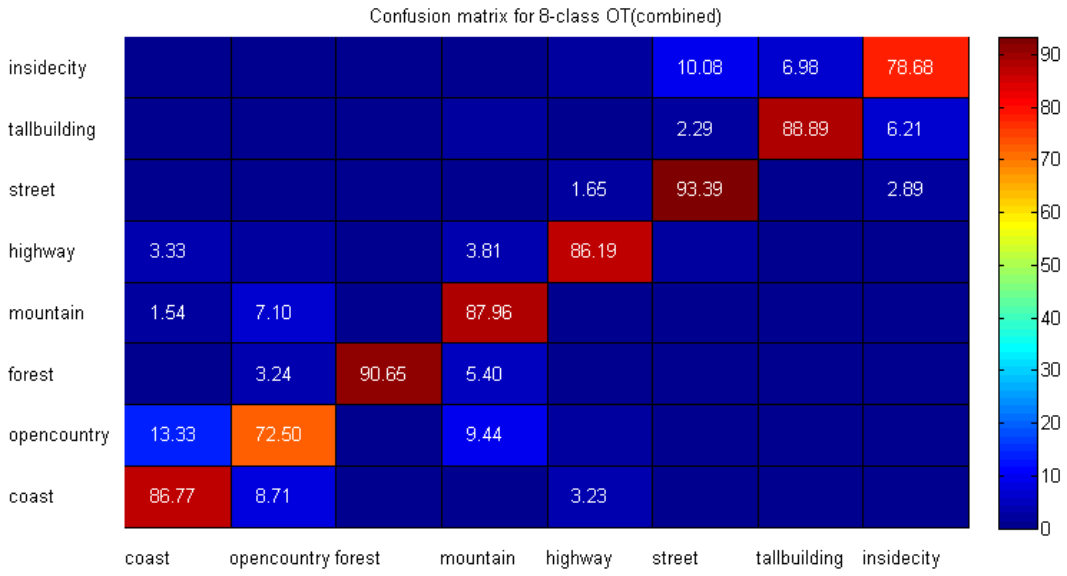

Fig. 22: Confusion matrix for OT with overall accuracy of 85.18(combined)

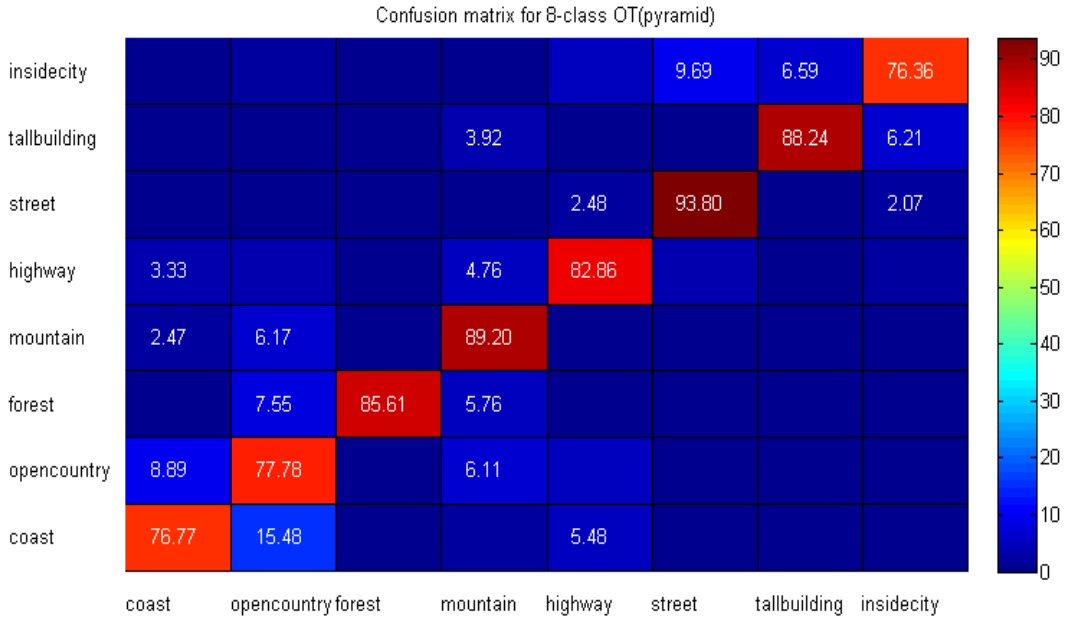

Fig. 23: Confusion matrix for OT with overall accuracy of 83.61(SPM)

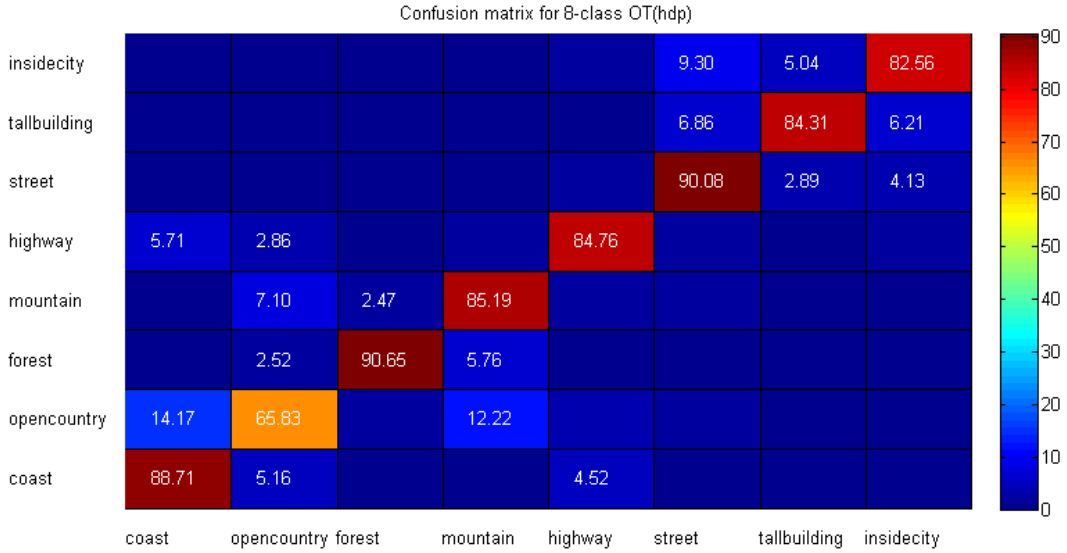

Fig. 24: Confusion matrix for OT with overall accuracy of 83.35(HDP)

The highest confusion inside man-made categories (10.08%) occurs between the inside city and the street, and the examples are shown in Fig. 26. For natural categories, the confusion between the open country and the coast (13.33%) is the highest and please see Fig. 27 for the examples.

Fig. 25: High way images from OT. Images along the top are correctly classified. First two, middle two and last two images below are misclassified into coast, mountain and street.

Fig. 26: Inside city images being misclassified into street

Fig. 27: open country images being misclassified into coast

7.3.3. LSP

LSP is the most complicated data set including 15 categories. Fig. 28 shows the comparison for LSP. The performance of the SPM-based system becomes much more improved over the HDP-based system compared to that of VS and OT. This indicates that the SPM-based system's performance is more robust and reliable regarding the category numbers. In terms of both the total performances over all three datasets and the running speed of the whole system, the SPM-based system is the best choice.

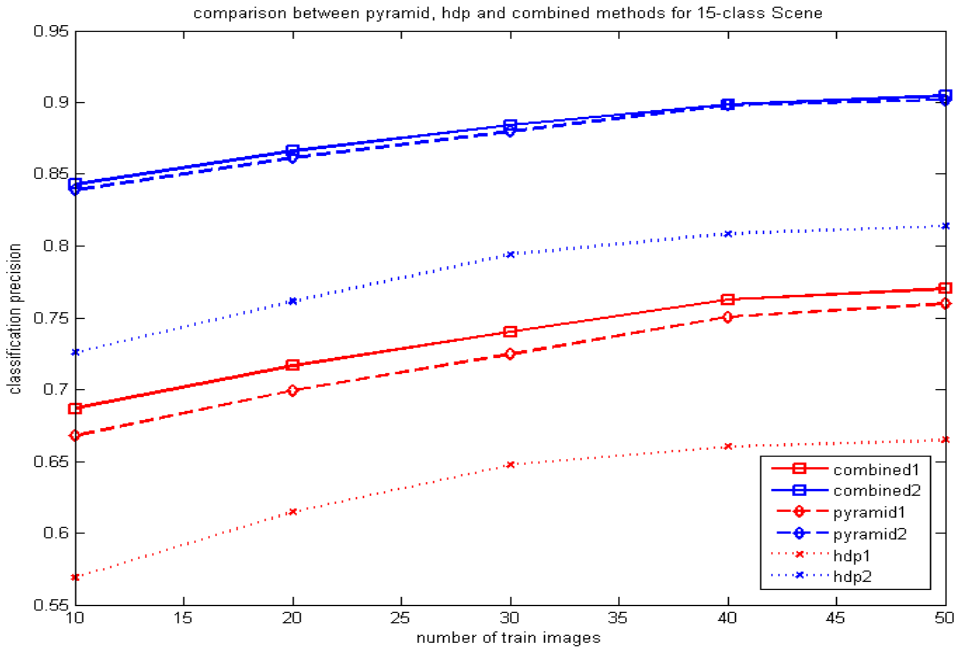

Fig. 28: Comparison for LSP

The confusion matrixes are shown in from Fig. 29 to Fig. 31 for the combined system, the SPM-based system and the HDP-based system, respectively. The confusion situation among the eight categories from the OT dataset is almost the same as that discussed above for OT. In addition, there is another observed confusion which occurs among the four indoor categories (kitchen, bedroom, living room, office), with the highest confusion being between the bedroom and the livingroom. From Fig. 32, it is obvious that images from the bedroom and the livingroom are very similar, being characterized with table, carpet, and frames on the wall. The texture of sofa in livingroom looks very like that of bed in bedroom too.

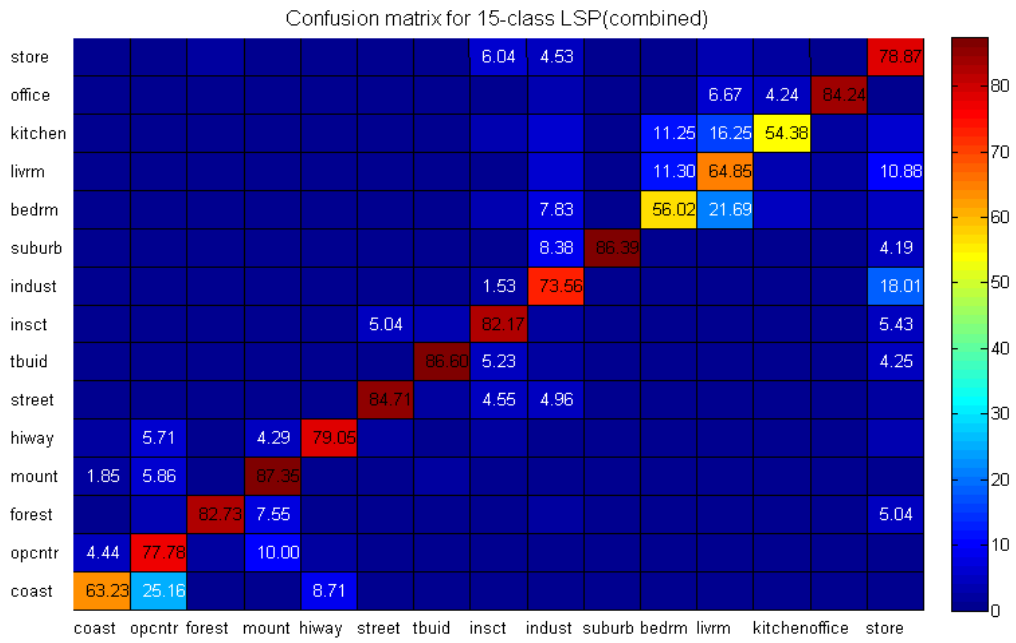

Fig. 29: Confusion matrix for LSP with overall accuracy of 77.03(combined)

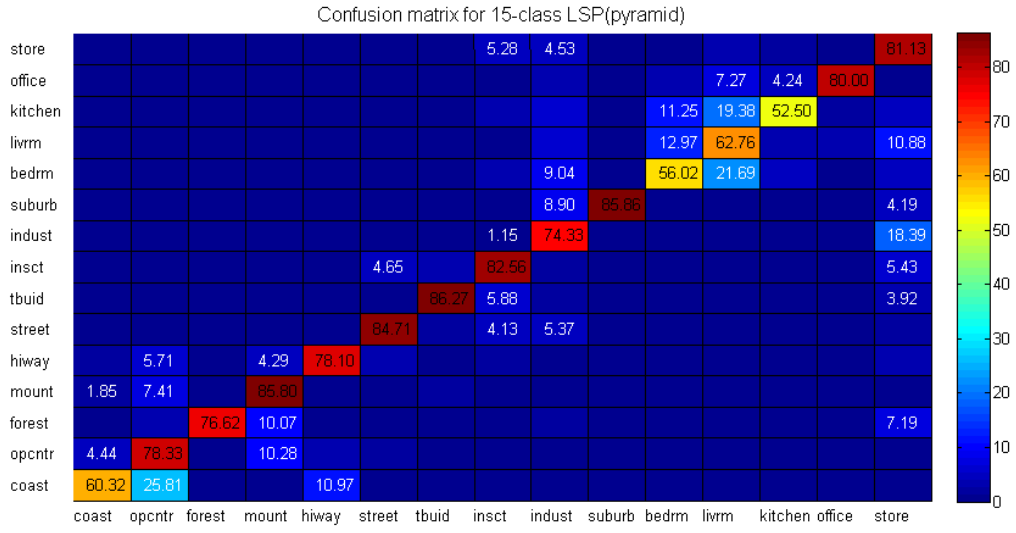

Fig. 30: Confusion matrix for LSP with overall accuracy of 75.98(SPM)

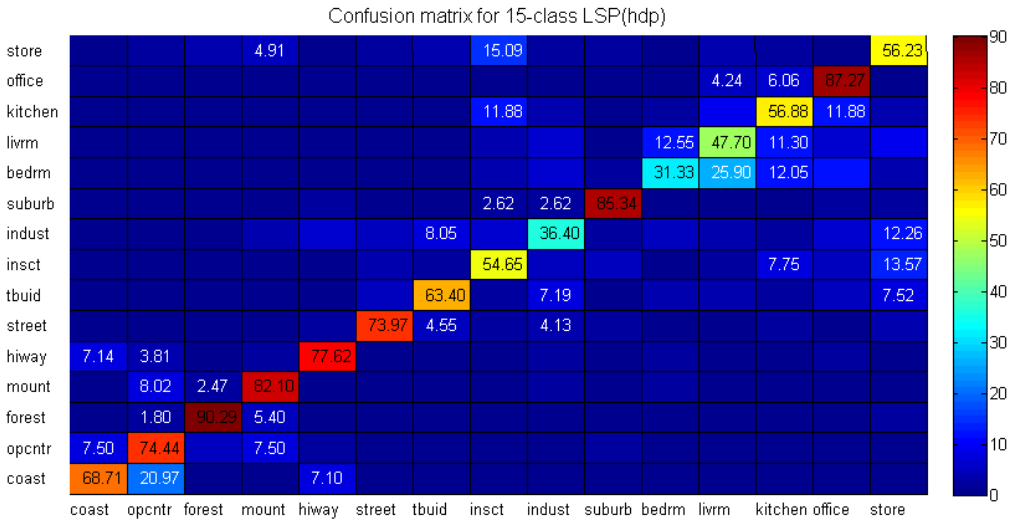

Fig. 31: Confusion matrix for LSP with overall accuracy of 66.48(HDP)

Fig. 32: Bedroom images along the top are misclassified to livingroom, while livingroom images below are misclassified to bedroom.

8. Conclusions

This paper has presented two scene classification systems: one is based on SPM and the other one is applying HDP. Both systems use dense SIFT descriptors and produce a visual vocabulary, which is then used to map features into textons.

In the SPM-based system, based on image textons, an image is represented by the spatial histogram pyramid. Spatial histogram pyramid is a simple and computationally efficient extension of an orderless bag-of-feature image representation, and it makes wise use of the information about the spatial layout of the features. The kernel of SVMs is precomputed using the Spatial Pyramid matching scheme and is the only input to train and test SVMs for classification. Through extensive experiments, it shows significantly improved performance on all three challenging datasets. Experiments with different levels of pyramid also confirm the more discriminative power of the multi-level pyramid representation.

The HDP-based system learns the HDP model using Gibbs sampling, which is very time-consuming. But the latent topics learned from the data can be used to do more than scene classification, such as object recognition and segmentation. Analyzing the HDP models for the three datasets shows HDP's ability in modeling grouped data consistently, allowing topics being shared between images both inside one scene category and across categories. The learned HDP model is characterized by a compact set of parameters and is directly used to classify novel scenes without need of the training data. As scene recognition systems are applied to large datasets, such savings in storage and computation become increasingly important.

By combining SPM and HDP, the system's performance is improved over either the SPM-base system or the HDP-based system.

9. References

[1] Oliva, A. & Torralba, A. “Modeling the Shape of the Scene: A Holistic Represen- tation of the Spatial Envelope.” International Journal of Computer Vision, 42(3), 145-175. 2001.

[2] Vogel, J. & Schiele, B. “Natural scene retrieval based on a semantic modeling step.” CIVR, Dublin, Irelan. 2004

[3] P. Quelhas, F. Monay, J.M. Odobez, D. Gatica-Perex, T. Tuytelaars and L. Van Gool, “Modeling Scenes with Local Descriptors and Latent Aspects.” Proc. Int’l Conf. Computer Vision, pp. 883-890, Oct. 2005.

[4] Fei-Fei, L. & Perona, P. “A Bayesian Hierarchical Model for Learning Natural Scene Categories.” Proceedings of the IEEE Computer Vision and Pattern Recog- nition, 2, 524-531. 2005.

[5] S. Lazebnik, C. Schmid, and J. Ponce. “Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories.” CVPR, volume 2, pages 2169–2178,
New York, NY, June 17–22, 2006.

[6] K. Grauman and T. Darrell. “Pyramid match kernels: Discriminative classification with sets of image features.” In Proc. ICCV, 2005.

[7] E.Sudderth, A. Torralba, W. T. Freeman, and A. Willsky. “Describing Visual Scenes Using Transformed Objects and Parts.” International Journal of Computer Vision, No. 1-3, May 2008, pp. 291-330.

[8] Bennett, K. & Campbell, C. “Support Vector Machines: Hype of Hallelujah?” SIGKDD Explorations, 2, 2. 2000.

[9] Chang, C. & Lin, C. “LIBSVM – A library for Support Vector Machines.” LIBSVM – A Library for Support Vector Machines. http://www.csie. edu.tw/cjlin/libsvm/.

[10] Andrea Vedaldi & Brian Fulkerson, VLFeat Open Source library. http://www.vlfeat.org/index.html

[11] David Rubel, “Scene Classification Using pLSA and Spatial Information”, RIT MS Project Report

[12] Y. W. Tech, M. I. Jordan, M. J. Beal, and D. M. Blei. Hierarchical Dirchlet processes. J. Amer. Stat. Assoc., 101 (476): 1566-1581, Dec. 2006