CS 231A Computer Vision (Fall 2011)
Problem Set 4

Due: Nov. 30th, 2011 (9:30am)

1 Part-based models for Object Recognition (50 points)

One approach to object recognition is to use a deformable part-based model. Unlike global features (e.g. bag of words), a part-based model encodes relative spatial locations of different parts of an object, as shown in Fig. 1(a). The deformable nature of the model allows for relative spatial movement between the parts. Given a test image, the algorithm will find the best match for each part in the image while also considering the cost of deforming the relations between the parts. In this problem you will explore how to pose this problem in a statistical framework. Further, given a set of training images, you will show how the model θ can be learned.

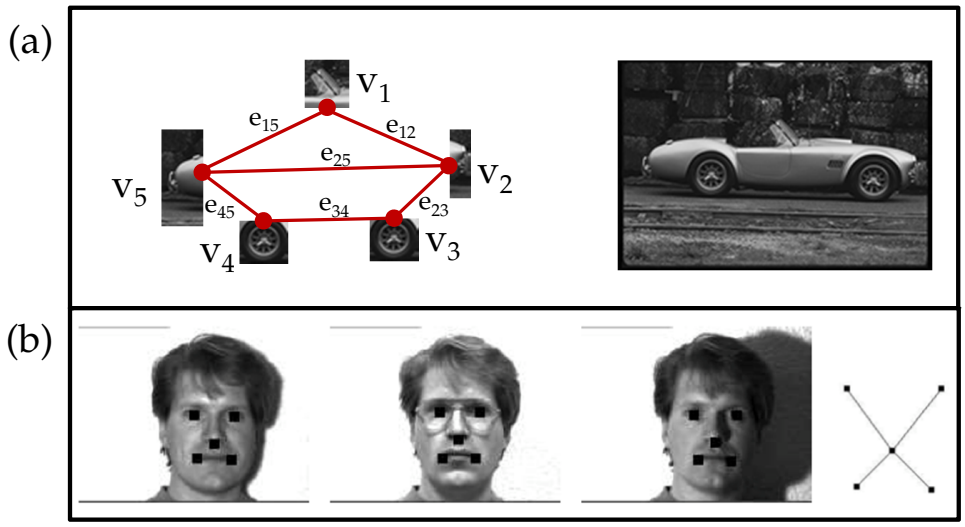

Figure 1: **(a)** An illustration of the parameters of an example deformable part-based model. Note that there is a connection parameter c_{ij} corresponding to each edge e_{ij}, and an appearance parameter u_i corresponding to each vertex v_i.
(b) Three training images showing the part locations and the last image depicting the learned structure model.

A pictorial structure model is a type of part-based model where an object is given by a collection of parts with connections between certain pairs of parts. A pictorial structure model is a graph structure with vertices $V = \{v_1, \ldots, v_n\}$ corresponding to the n parts in the model. It is parameterized by $\theta = \{u, E, c\}$ where $u = \{u_1, \ldots, u_n\}$ are appearance parameters

(such as patches), $E = \{e_{ij}\}$ is the set of edges indicating whether parts are connected, and $c = \{c_{ij} \mid (v_i, v_j) \in E\}$ are connection parameters. Each part v_i has its own appearance parameter u_i (the specifics of this parameter are not needed to solve this problem). For a given image, we would like to find the object configuration $L = (l_1, \ldots, l_n)$, where each l_i specifies the location of part v_i.

In this problem, we specifically find $p(L \mid I, \theta)$ which denotes the probability of an object configuration given an image I, and a model θ. Using Bayes Rule we have $p(L \mid I, \theta) \propto p(I \mid L, \theta)p(L \mid \theta)$ where $p(I \mid L, \theta)$ defines the likelihood of seeing a particular image given that an object is at some location, and $p(L \mid \theta)$ is the likelihood that an object is at a particular location.

To get some intuition of this model let's consider scanning this model over an image. Given a learned model θ, there are two components to evaluate for our model configuration corresponding to each of the probability terms above (from Bayes Rule). First, for each part i, we compare our appearance parameter u_i to the image I and get a measure for how likely the location of the corresponding part is. Second, given a model θ we can calculate the likelihood of a object configuration L. An intuitive way to think of how c_{ij} describes the relation between two parts is to think of the connections between the parts as springs and c_{ij} as the measure of stiffness of those springs. Although we do not know the location of each part given an image we have a general idea of how the object locations will appear relative to each other, and we have a way of measuring how likely a given part is to appear at each location using u.

(a) Write $p(I \mid L, \theta)$ as a function of I, u_i, E, c_{ij} and l_i. Assume that the parts do not overlap. *Hint: What information is not needed to evaluate the likelihood of an image given a part configuration, and the model?*

(b) For this part $p(L \mid \theta)$ has the form

$$p(L \mid \theta) = \frac{\prod_{(v_i,v_j)\in E} p(l_i, l_j \mid \theta)}{\prod_{v_i \in V} p(l_i \mid \theta)^{\deg\, v_i - 1}}$$

where deg v_i is the degree of vertex v_i in the graph defined by E. Also l_i denotes the location of part i. In addition, assume that there is no information associated with the absolute location of an individual part.

Represent $p(L \mid \theta)$ in terms of I, u_i, E, c_{ij} and l_i. Use the expression for $p(L \mid \theta)$ and the assumption given above.

(c) In the previous parts of this problem we have assumed that our model θ is known. In this part you will derive how we learn this model given training images. Specifically use the maximum likelihood (ML) estimation to find the model $\theta^* = (u^*, E^*, c^*)$.

(i) Write down the ML estimate for θ^*.
Assume that you are given m i.i.d. (independent and identically distributed) training images. Use the expressions you found in parts (a) and (b) to simplify your answer.
Note that we do not expect you to find a closed form solution for θ^*. You only need to pose the optimization problem whose solution is θ^*. If it helps you to understand the problem you can assume that the distributions are all Gaussian.

(ii) Show that our estimate for θ^* can be found by solving two separate maximization problems. Show how to find our estimates for u_i^*,E^*, c_{ij}^* from these problems. Again, in this problem you do not need to provide a closed form solution, you only need to

set up the optimization problems.

(iii) **Extra Credit (1%)**: Give a method for finding E^* and c^* in accordance with the ML estimate of θ^* separately. First assume that E is a fully connected graph, using this find an estimate for c_{ij}^*. Using these values for your c_{ij}^*'s find an estimate for E^*. *Hint: Pose E^* as a common graph problem.*

(d) A question that you should always be asking yourself when you see a new recognition algorithm is what the method is invariant to, and how it accomplishes these invariances. Specifically, is the method developed in this problem invariant to scale, intensity, affine transformation and/or rotation? Provide justification for your answers in a few sentences.

2 Scene Classification and Matching Using ObjectBank (50 points)

In this problem you will experiment with a low-level (based on local image features) and high-level (based on object relevance) approach to scene classification. Furthermore, you will explore the use of different feature vectors with a linear kernel support vector machine (SVM) in order to improve classification performance. The scene classification will be performed on a given subset of the UIUC Sports data-set. The dataset along with all other materials for this problem can be downloaded from the course web page (`PS4_data.zip`).

One of the most exciting branches of modern Computer Vision research is the description of the entire scene depicted in an image. While there are many available solutions to object detection and image description, a conclusive approach to describe the semantic contents of a natural scene has not yet emerged. However, ObjectBank [3, 4] is a new technology that has demonstrated very good results with a relatively simple approach.

ObjectBank (OB) is similar to an array of filters in traditional signal processing insofar that it is an array of object detectors applied to one image. However, instead of outputting a simple measure of the signal inside a frequency band, the output of an object detector is a 3-dimensional response map which quantifies the presence of an object as a function of resolution and (x,y)-coordinates. These response maps can be concatenated to create an overall description of a scene in terms of the presence of objects. For example, Fig. 2 shows some sample responses from a test image of sailboats. The"Sailboat" filter has several large responses that correlate to the

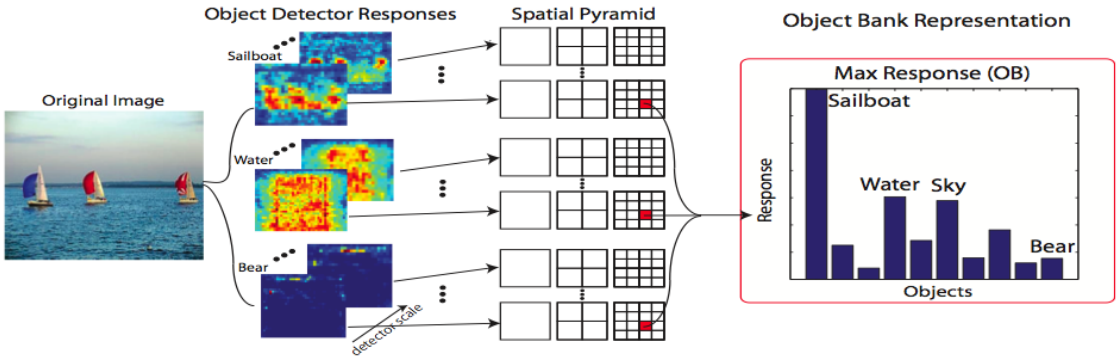

Figure 2: Diagram of computation pipeline for ObjectBank representation of an image.

Algorithm 1 Create ObjectBank Feature Vector

$FeatureVector \Leftarrow []$
for detector d in DetectorSet **do**
  $ResponseMap \Leftarrow Detect(d, image)$
  $SPMPoints \Leftarrow Compress(ResponseMap)$
  $FeatureVector.Append(SPMPoints)$
end for

location of the actual boats, the"Water" filter has a uniformly large response, and the"Bear" filter demonstrates effectively zero signal. These object detector results can be concatenated into a cohesive Feature Vector for the image, which can then be fed into an SVM or other classification technique.

One of the complications of this technique is the need for data compression or reduction. Each object detector yields an output of roughly 12,000 datapoints, and with 177 objects currently in the ObjectBank, the feature vector (dimension $= 12000 * 177 \approx 2.1mil$) would be high dimensional and unlikely to work in a standard machine learning technique due to overfitting, etc. The current solution for this problem is to continually subdivide the image and extract the maximum value in each subdivision in a modification of the Spatial Pyramid Matching (SPM) [2] technique as shown in Fig. 3.

For the purposes of this assignment, we will "black box" the detection portion of the system. *If you are curious to learn more, please refer to the following papers: [1, 3, 4] (not required for this assignment)*

Note: You need to print & submit your code for all parts of this problem to receive full credit as the final results have already been provided in the starter code. You do not need to submit any files that were not modified by you.

(a) Before exploring the object-based approach of ObjectBank, you will experiment with the low-level bag of words (BoW) representation. That is, you will create a feature vector using the BoW approach discussed in class and train a linear SVM to classify scenes.

We have provided you with a pre-trained code-book to form the feature vector. Assign each SIFT descriptor to its nearest code-book entry using L_2 distance and aggregate the results in a histogram to form the BoW feature. Use the Liblinear SVM package (same as

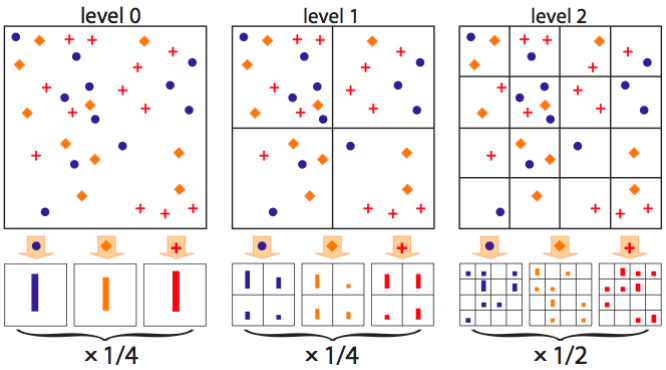

Figure 3: Overview of the Spatial Pyramid Kernel in which histograms of interest point (denoted by the three markers) are created for each subdivision in a 2-level pyramid. The method used in ObjectBank is very similar, except it only extracts the local max inside each subdivision.

PS0) to train and test your classifier on the given subset of the UIUC Sports data-set. To compare your classification accuracy to the one we obtained, run the `Problem2a.m` script.

(b) Using your result in part (a) as a benchmark, we will implement ObjectBank in several steps. We have given you the response maps corresponding to the data-set in the folder `response_maps`. To speed up the processing of the data, we will be using only 50 of the 177 objects for this assignment.

(i) To motivate the switch from BoW to OB use a 0-level SPM to create your feature vector (i.e. select the max of each response map and add it to the feature vector). Train a linear SVM and compare the classification accuracy to the one we obtain for this part, and the result that you obtained for part (a). The code and specific instructions for this part of the assignment are in `Problem2bi.m`.

(ii) Now use a 2-level SPM to form the feature vector form the given response maps. Train a linear SVM and compare your performance results to the given benchmark. The code and specific instructions for this part of the assignment are in `Problem2bii.m`.

(c) Assume that you are given a single image (without a label) and asked to find the "most similar" image from your dataset (also without labels). How would you approach this problem? Describe and implement a method **using the OB response maps** to generate features for the purpose of selecting the closest semantic match to the given image. Essentially, we would like to build a Google for images without using textual input.

This problem is intended to be more open-ended and provide a lot of room for creativity. To begin with, be clear on the type of similarity you are trying to achieve. Some possible definitions are pixel-by-pixel accuracy, OB feature correspondence, or local image-patch matching. Once you have a sense of your metric, you could try several approaches that we have discussed in the course: clustering, norm of differences, 1-vs-all SVMs, correlation, etc.. To get full credit for this problem, please write a clear description of your methodology and an explanation of trade-offs compared to other methods to demonstrate your understanding of the material.

The sample data that you are given in this problem is a mix of data with clear ground truths (i.e., a picture of people playing badminton should match with another picture of people playing badminton) and intentionally poor matches. It is interesting to note what matches you get and to share insights into why the results make sense to you. **Please submit your code, a description of your method, and a printout of image match results displayed by `Problem2c.m`.**

(a) (b)

Figure 4: Sample Test Images

References

[1] Pedro Felzenszwalb, David McAllester, and Deva Ramanan. A discriminatively trained, multiscale, deformable part model. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR) Anchorage, Alaska, June 2008.*, 2008.

[2] Svetlana Lazebni, Cordelia Schmid, and Jean Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2169–2178, 2006.

[3] Li-Jia Li, Hao Su, Yongwhan Lim, and Li Fei-Fei. Objects as attributes for scene classification. In *European Conference of Computer Vision (ECCV), International Workshop on Parts and Attributes*, Crete, Greece, September 2010.

[4] Li-Jia Li, Hao Su, Eric P. Xing, and Li Fei-Fei. Object bank: A high-level image representation for scene classification & semantic feature sparsification. In *Neural Information Processing Systems (NIPS)*, Vancouver, Canada, December 2010.