16720J: Homework 3 - Object Detection

Name - Jiaxin Chen

October 24, 2016

1 Warming up with some theory (9pts)

Question 1.1 (2pts, 1 line)

I can classify $(M - w + 1) \times (N - h + 1)$ windows in this image.

Question 1.2 (5pts, 2-3 lines)

Given the following Model1 and Model2 as example:

Predicted M1	Negative	Positive
Negative Cases	9700	150
Positive Cases	50	100

Predicted M2	Negative	Positive
Negative Cases	9850	0
Positive Cases	150	0

We can calculate the accuracy :

$$Accuracy(M1) = \frac{TP + TN}{TP + TN + FP + FN} = \frac{9700 + 100}{9700 + 50 + 150 + 100} = 0.980 \tag{1}$$

$$Accuracy(M2) = \frac{TP + TN}{TP + TN + FP + FN} = \frac{9850 + 0}{9850 + 150 + 0 + 0} = 0.985 \tag{2}$$

Model2 reduces the rate of inaccurate predictions from 2% to 1.5%. This is an apparent improvement of 25%.The Model2 shows fewer incorrect predictions and markedly improved accuracy, as compared to Model1, which is obviously useless and cause accuracy paradox. Therefore, accuracy paradox with a class imbalance when classification accuracy alone cannot be trusted to select a well performing model. We need precision/recall as the measure of a classifiers exactness and completeness.

Question 1.3 (2 pts, 1 line)

One train can have 1 Exemplar detector as well as 1 Dalal-Triggs type template detector.

2 Object Detection via DPMs and Non-Maximum Suppression (40 pts)

2.1 Mean-Shift Clustering (20 pts)

Question 2.1.1 `MeanShift.m`

```matlab
% Created by chenjx65 on 2016-10-12.

function [CCenters,CMemberships] = MeanShift(data,bandwidth,stopThresh)
% Intialize X and associated positive weight
X = data(:, 1:end-1);
w = data(:, end);
[N, F] = size(X);
XNew = zeros(N, F);

for i = 1:N
    Xi = X(i, :);
    while 1
        % Calculate the distance between current point and other point.
        distance = sqrt(sum(bsxfun(@minus, X, Xi).^2, 2));
        % Get the index of points within the bandwidth and calculate Xmean.
        index = distance < (bandwidth / 2);
        Xmean = sum(bsxfun(@times, w(index,:), X(index,:)))/sum(w(index,:));
        % Check convergence of the Mean-Shift algorithm and update Xmean.
        if norm(Xmean - Xi) < stopThresh
            break;
        end
        Xi = Xmean;
    end
    XNew(i, :) = Xmean;
end

% Initialize and define merge threshold
CMemberships = zeros(N, 1);
CCenters = XNew(1, :);
mergeThresh = norm(max(X) - min(X)) / 100;
for i = 1:N
    centerDist = sqrt(sum(bsxfun(@minus, CCenters, XNew(i, :)).^2, 2));
    [certerDistMin, pos] = min(centerDist);
    % Merge to the corresponding cluster
    if certerDistMin < mergeThresh
        CMemberships(i) = pos;
    % Create a new cluster
    else
        CCenters = [CCenters; XNew(i, :)];
        CMemberships(i) = size(CCenters, 1);
    end
end
save('q21_result.mat', 'CCenters', 'CMemberships');
end
```

Question 2.1.2

Save your CCenters and CMemberships into q21_result.mat. Also save the visualization result from q21_test.m, as q21_clustering.jpg and include here.

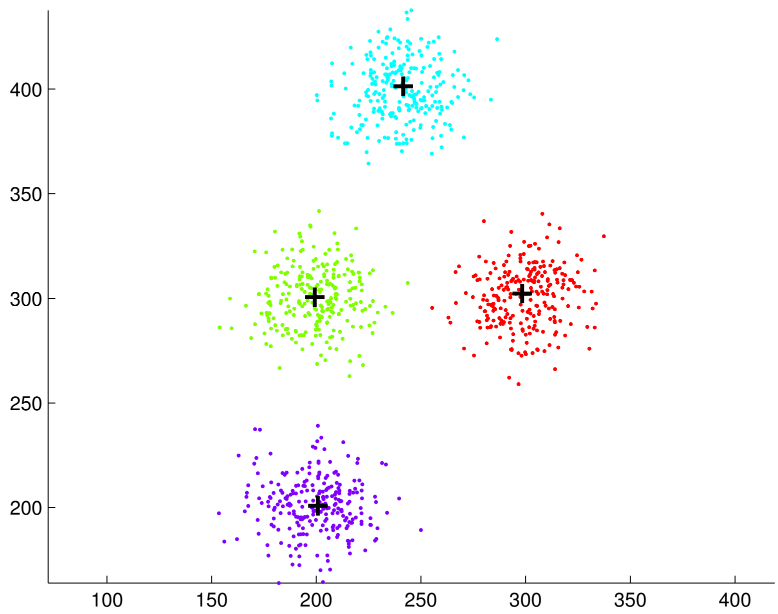

Figure 1: Mean Shift Clusters

Question 2.1.3 (at most 3 lines in your write-up)

Data will be divided into lots of small clusters for low bandwidth or merged into a big cluster for high bandwidth, which is opposite to the reality. Bandwidth is chosen to match the distance between the cluster center and its points as well as the distance among different cluster centers.

2.2 Detecting using Deformable Part Models (DPMs) (20 pts)

Question 2.2.1

Submit your `nms` function and include here

```
% Created by chenjx65 on 2016-10-12.

function [refinedBBoxes] = nms(bboxes, bandwidth,K)
% Calculate stopThresh and normalize the score
stopThresh = 0.001 * bandwidth;
bboxes(:, end) = mat2gray(bboxes(:,end));
% Calculate CCenters and CMemberships by MeanShift
[CCenters,CMemberships] = MeanShift(bboxes, bandwidth, stopThresh);

refinedBBoxes = [];
M = size(CCenters, 1);
for i = 1:M
    % Find the candidate boxes for M clusters
    candidatebboxes = bboxes(CMemberships == i, :);
    % Choose the refinedBBoxes with the highest score
    [¬, pos] = max(candidatebboxes(:, end));
    refinedBBoxes = [refinedBBoxes; candidatebboxes(pos, :)];
end

% Obtain the top K detectors by sorting refinedBBoxes descendingly.
if K < size(refinedBBoxes, 1)
    [¬,index] = sort(refinedBBoxes(:, end), 'descend');
    refinedBBoxes = refinedBBoxes(index, :);
    refinedBBoxes = refinedBBoxes(1:K, :);
```

```
end

end
```

Question 2.2.2 (at most 3 lines in your write-up)

I classify the candidate boxes for each cluster and choose from the candidate boxes with the highest score as refinedBBoxe. And I sort the refinedBBoxes in descending order according to its score to obtain the top K detectors.

Question 2.2.3

Your result images here:

(a) NMS Result 1 - 008360.jpg (b) NMS Result 2 - 009105.jpg (c) NMS Result 3 - 009126.jpg

3 Reducing Exemplar Detectors (55 pts)

3.1 Detecting using Exemplar Detectors (10 pts)

Question 3.1.1 (10 pts)

```
% Created by chenjx65 on 2016-10-13.

function [boundingBoxes] = batchDetectImageESVM(imageNames, models, params)

imageDir = '../../data/voc2007';
% Initialize
N = length(imageNames);
boundingBoxes = cell(1, N);
% Create boundingBoxes for each image
for i = 1:N
    I = imread(fullfile(imageDir, imageNames{i}));
    boundingBoxes{i} = esvm_detect(I, models, params);
end
end
```

3.2 Evaluating Detection Performance (15 pts)

Question 3.2.1 Theory (5 pts, 2 lines)

Average precision computes the average value of p(r) over the interval $r \in [0,1]$: $AP = \int_0^1 p(r)\mathrm{d}r$. By computing a precision and recall at every position in the ranked sequence of documents, we can plot this precision-recall curve.

Question 3.2.2 (10 pts)

Submit your script as q3_2_2.m and include here. Include an interpretation of your graph here.

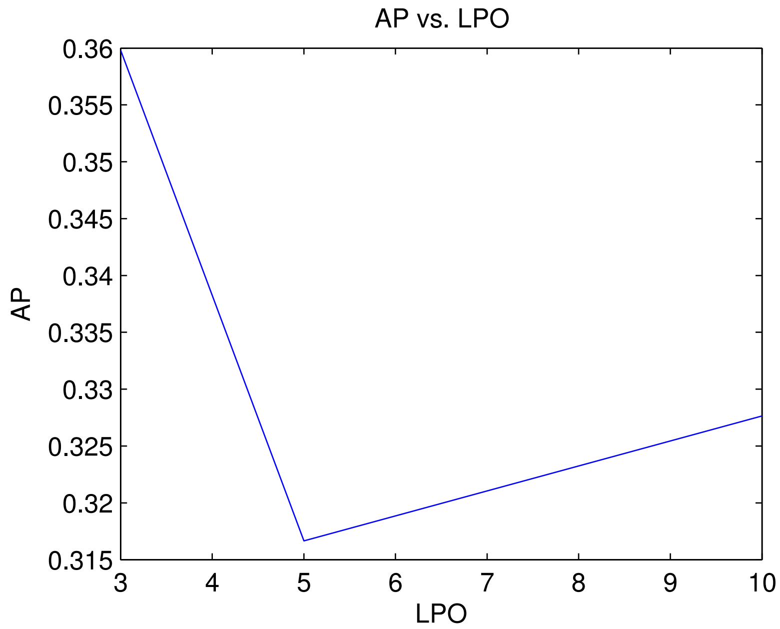

Figure 3: AP vs LPO (test set)

My interpretation of the graph:
The AP will vary with the variation of LPO. The bigger LPO is, the more resizings of an image we have. LPO = 3 has the highest AP because 3 rescaled images already contain enough useful information for detector. However, when LPO get bigger, the resized image is too small for the detector so it increases the noise to the detector on the contrary, that's why the AP decreases.

```
% Created by chenjx65 on 2016-10-13.

% Q3.2.2

addpath(genpath('../utils'));
addpath(genpath('../lib/esvm'));
load('../../data/bus_esvm.mat');
load('../../data/bus_data.mat');
params = esvm_get_default_params();

% Initialize
lpo = [3, 5, 10];
len = length(lpo);
recall = cell(1,len);
precion = cell(1,len);
AP = zeros(1,len);

for i = 1:len
    % Update params and compute the corresponding boundingBoxes.
    params.detect_levels_per_octave = lpo(i);
    [boundingBoxes] = batchDetectImageESVM(gtImages, models, params);
    [recall{i}, precion{i}, AP(i)] = evalAP(gtBoxes,boundingBoxes);
end
```

```
% plot AP vs. lpo graph
plot(lpo, AP);
title('AP vs. LPO');
xlabel('LPO');
ylabel('AP');
```

3.3 Compacting the set of exemplar detectors

Question 3.3.1 (20 pts)

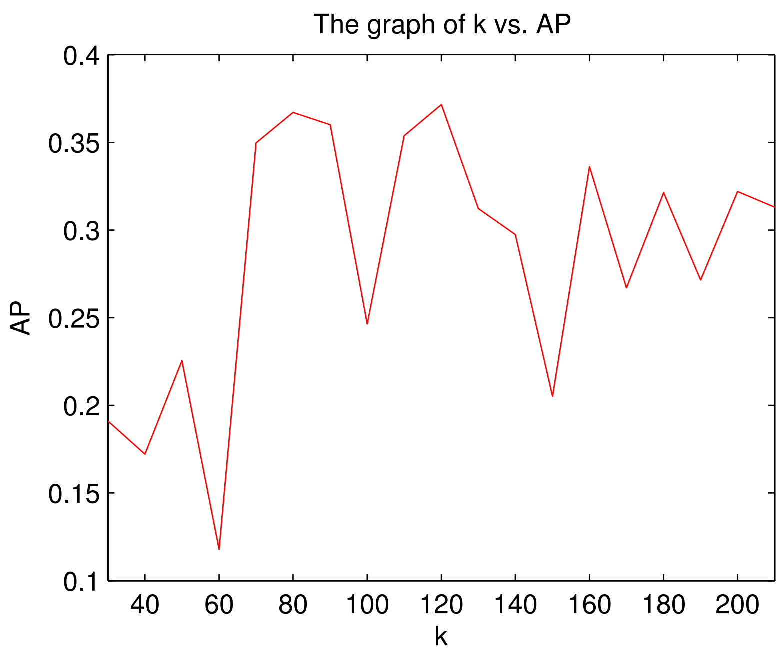

Figure 4: K vs AP

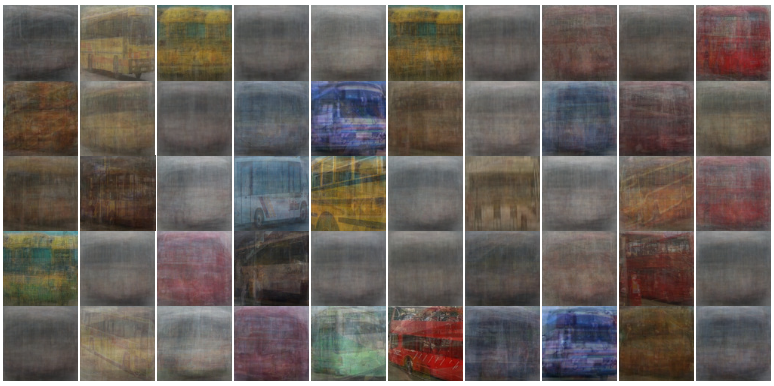

Figure 5: Average Images

```
% Created by chenjx65 on 2016-10-13.

% Q3.3.1

addpath(genpath('../utils'));
```

```
addpath(genpath('../lib/esvm'));
addpath(genpath('../external'));
load('../../data/bus_esvm.mat');
load('../../data/bus_data.mat');
imageDir = '../../data/voc2007/';

% Parameters
alpha = 100;
K = 100;
resize = 100;

% Create filterBank
filterBank = createFilterBank();

% Initialize
T = length(modelImageNames);
N = length(filterBank);
filterResponses = zeros(alpha*T, N*3);
IResizedBoxes = cell(1, T);
averageImgs = cell(1, K);
alphaImgIndex = zeros(alpha*T, 1);
clusterImg = zeros(1, K);

%% Calculate filterResponses and cluster them.
for i = 1:T
    % Obtain resized box of each image
    I = imread([imageDir, modelImageNames{i}]);
    imgBBox = modelBoxes{i};
    IBox = I(imgBBox(2): imgBBox(4), imgBBox(1): imgBBox(3), :);
    IResizedBoxes{i} = imresize(IBox, [resize resize]);
    alphaImgIndex((i-1)*alpha+1: i*alpha) = i*ones(alpha, 1);

    % Select alpha imgResponses into filterResponses
    imgResponses = extractFilterResponses(IBox, filterBank);
    alphaResponses = imgResponses(randperm(size(imgResponses, 1), alpha), :);
    filterResponses((i-1)*alpha+1 : i*alpha, :) = alphaResponses;
end
% Cluster filterResponses by kmeans function
[clusterIndex, ¬, ¬, clusterDistance] = kmeans(filterResponses, K, ...
    'EmptyAction', 'drop');

%% Calculate the averageImg for each cluster
for i = 1:K
    averageImg = zeros(resize, resize, 3);
    imgIndex = alphaImgIndex(clusterIndex == i);
    len = length(imgIndex);
    for j = 1:len
        averageImg = averageImg + double(IResizedBoxes{imgIndex(j)});
    end
    averageImgs{i} = uint8(averageImg / len);
end
imdisp(averageImgs, 'Size', 10);

set(gcf,'position',[0 0 800 800]);
set(gcf, 'Color', 'w');
print -dpdf q3_3_1.pdf
export_fig q3_3_1.pdf

%% Calculate boundingboxes and AP
for i = 1:K
```

```
    [¬, pos] = min(clusterDistance(:, i));
    clusterImg(i) = alphaImgIndex(pos);
end
refined_models = models(unique(clusterImg));
params = esvm_get_default_params();
params.detect_levels_per_octave = 3;
[boundingBoxes] = batchDetectImageESVM(gtImages, refined_models, params);
[¬,¬,ap] = evalAP(gtBoxes, boundingBoxes);
fprintf('K = %i, AP = %d\n',K,ap);
```

Question 3.3.2 (10 pts)

I utilize SURF as the feature detector to capture filter responses.

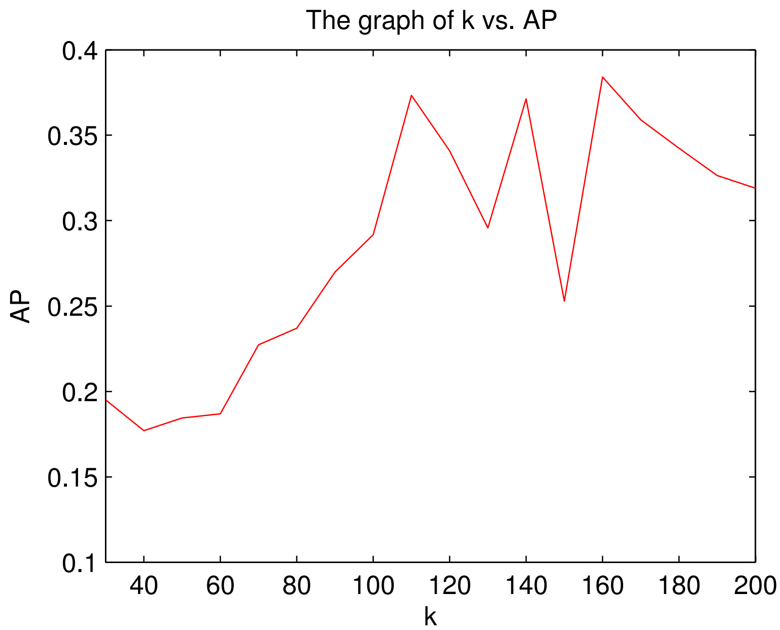

Figure 6: K vs AP

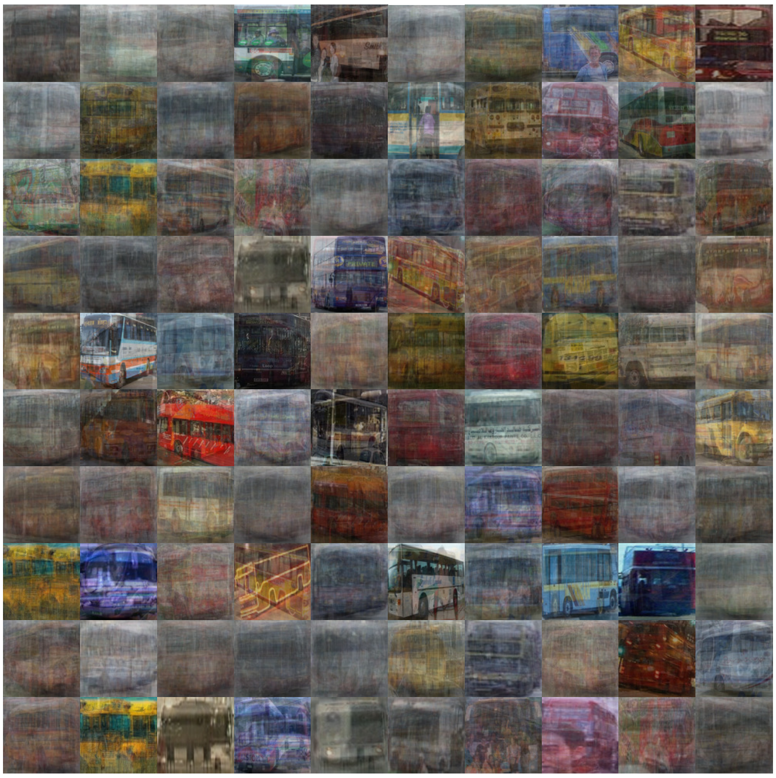

Figure 7: Average Images

```
% Created by chenjx65 on 2016-10-14.

% Q3.3.2

addpath(genpath('../utils'));
addpath(genpath('../lib/esvm'));
addpath(genpath('../external'));
load('../../data/bus_esvm.mat');
load('../../data/bus_data.mat');
imageDir = '../../data/voc2007/';

% Parameters
alpha = 20;
K = 100;
resize = 100;

% Create filterBank
filterBank = createFilterBank();

% Initialize
T = length(modelImageNames);
N = length(filterBank);
filterResponses = zeros(alpha*T, N*3);
IResizedBoxes = cell(1, T);
averageImgs = cell(1, K);
alphaImgIndex = zeros(alpha*T, 1);
clusterImg = zeros(1, K);

%% Calculate filterResponses and cluster them.
for i = 1:T
    % Obtain resized box of each image
```

```
    I = imread([imageDir, modelImageNames{i}]);
    imgBBox = modelBoxes{i};
    IBox = I(imgBBox(2): imgBBox(4), imgBBox(1): imgBBox(3), :);
    IResizedBoxes{i} = imresize(IBox, [resize resize]);
    alphaImgIndex((i-1)*alpha+1: i*alpha) = i*ones(alpha, 1);

    % Select alpha imgResponses into filterResponses
    imgResponses = extractFilterResponses(IBox, filterBank);

    % Detect the SURF Features and get the corrdinate of strongest points
    points = detectSURFFeatures(IBox(:,:,1));
    strongestPoints = points.selectStrongest(alpha);
    pointsCor = round(strongestPoints.Location);
    if size(pointsCor, 1) == 0
        pointsCor = randi([1 min(size(IBox))], alpha, 2);
    elseif size(pointsCor, 1) < alpha
         copy = ceil((alpha - size(pointsCor, 1))/size(pointsCor, 1)) + 1;
        corAppend = repmat(pointsCor, copy, 1);
        pointsCor = corAppend(1:alpha, :);
    end

    rowIndex = ((pointsCor(:,1)-1)*size(IBox,1) + pointsCor(:, 2))';
    alphaResponses = imgResponses(rowIndex, :);
    filterResponses((i-1)*alpha+1 : i*alpha, :) = alphaResponses;
end
% Cluster filterResponses by kmeans function
[clusterIndex, ¬, ¬, clusterDistance] = kmeans(filterResponses, K, ...
    'EmptyAction', 'drop');

%% Calculate the averageImg for each cluster
for i = 1:K
    averageImg = zeros(resize, resize, 3);
    imgIndex = alphaImgIndex(clusterIndex == i);
    len = length(imgIndex);
    for j = 1:len
        averageImg = averageImg + double(IResizedBoxes{imgIndex(j)});
    end
    averageImgs{i} = uint8(averageImg / len);
end
imdisp(averageImgs, 'Size', 10);

set(gcf,'position',[0 0 800 800]);
set(gcf, 'Color', 'w');
print -dpdf q3_3_2.pdf
export_fig q3_3_2.pdf

%% Calculate boundingboxes and AP
for i = 1:K
    [¬, pos] = min(clusterDistance(:, i));
    clusterImg(i) = alphaImgIndex(pos);
end
refined_models = models(unique(clusterImg));
params = esvm_get_default_params();
params.detect_levels_per_octave = 3;
[boundingBoxes] = batchDetectImageESVM(gtImages, refined_models, params);
[¬,¬,ap] = evalAP(gtBoxes, boundingBoxes);
fprintf('K = %i, AP = %d\n',K,ap);
```

4 Extra credit: Segmentation transfer using ESVM (20 pts)

If you have attempted this extra-credit section please include a summary of your efforts here and include all relevant work in the folder `segTransfer`.