Neeraj Vashistha - 190573735

School of Electronic Engineering and Computer Science

ECS797

Machine Learning for Visual Data Analysis

Lab 4: Part-based Action localisation

3. Localising actions in image sequences

1. The function ism_test_voting calculates the voting maps for a single image sequence. Calculate the Euclidean matrix between the dictionary elements and the descriptors extracted in an image sequence.

```
aa=sum(a.*a,2); bb=sum(b.*b,2); ab=a*b';
d = sqrt(abs(repmat(aa,[1 size(bb,1)]) + repmat(bb',[size(aa,1) 1]) - 2*ab));
```

This piece of code is referenced from Lab 1 which has been referenced from Roland, I have included credits in the code.

The code calculates the euclidean distance which uses the basic formula of euclidean distance, just that the input and output matrices are of same length.

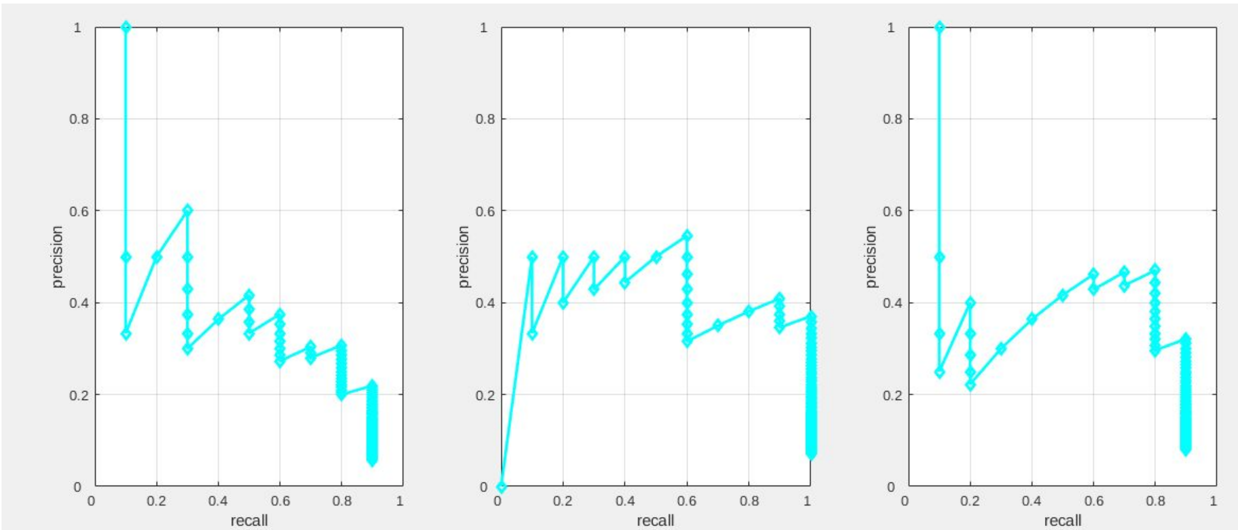
2. Write a function that implements the voting scheme for the following properties: a) the spatial centre of the action at the current frame, b) the start and the end of action and c) the width and the height of the bounding box of the action in the current frame. The details are given in houghvoting.m

```
[X, Y, S, E, V, B1, B2] = deal([]); % init
% dict_size = size(flag_mat, 1);
for p=1:size(patches,2)
    for i=1:size(flag_mat, 1)
        sum_act_cc = sum(flag_mat(i,:));
        if sum_act_cc ~= 0
            % if interest point exists
            num_of_edges = struct_cb.offset(i).tot_cnt; % patches
            act_pos_s = position(:, flag_mat(i,:)); % for X and Y, spatial
            act_frame_t = frame_num(:, flag_mat(i,:)); % start and end, temporal
            V_prob = [];
            for j=1:num_of_edges
                % updadte the weights
                X = [X act_pos_s(1,:) - spa_scale(1, flag_mat(i,:))  * struct_cb.offset(i).spa_offset(1, j)];
                Y = [Y act_pos_s(2,:) - spa_scale(1, flag_mat(i,:))  * struct_cb.offset(i).spa_offset(2, j)];
                S = [S act_frame_t - tem_scale(1, flag_mat(i,:))  * struct_cb.offset(i).st_end_offset(1, j)];
                E = [E act_frame_t - tem_scale(1, flag_mat(i,:))  * struct_cb.offset(i).st_end_offset(2, j)];
                B1 = [B1 struct_cb.offset(i).hei_wid_bb(1, j) * spa_scale(1, flag_mat(i,:)) ];
                B2 = [B2 struct_cb.offset(i).hei_wid_bb(2, j) * spa_scale(1, flag_mat(i,:)) ];
                V_prob = [V_prob repmat(1/(num_of_edges), sum_act_cc, 1)']; % voting
            end
```

```
Command Window
New to MATLAB? See resources for Getting Started.

ratio =

    0.1223


Mis-sclassification rate = 0.100000
>>
```

4. Evaluation

1. Using the provided code in ism_test_voting.m and recall_prec_curve.m, plot the Recall precision curves for each class.



2. Assign each sequence to a class according to which hypothesis received the higher number of votes (hint: use the values of the matrix TP_FP_mat). Report the misclassification error, or build the confusion matrix.

```
function er = eval_error()
% calculate the misclassification error which is
% FP+FN/Total
load('struct_TP_FP');
count = 0;
num_miss = 0;
for i = 1:3
    class_size = size(struct_TP_FP.class(i).seq,2);
    for j = 1:class_size
        % from reall_prec_curv.m, get temp_ind_pos and temp_class
        temp_ind_pos = ((struct_TP_FP.class(i).seq(j).array(1,:)==1) & (struct_TP_FP.class(i).seq(j).array(3,:)==i));
        temp_class = struct_TP_FP.class(i).seq(j).array(3,temp_ind_pos);
        % FP found
        if cumsum(temp_ind_pos) == 0
            num_miss = num_miss + 1;
        % FN
        elseif temp_class(1) ~= i
            num_miss = num_miss + 1;
        end
        count = count + 1;
    end
end
er = num_miss/count;
```

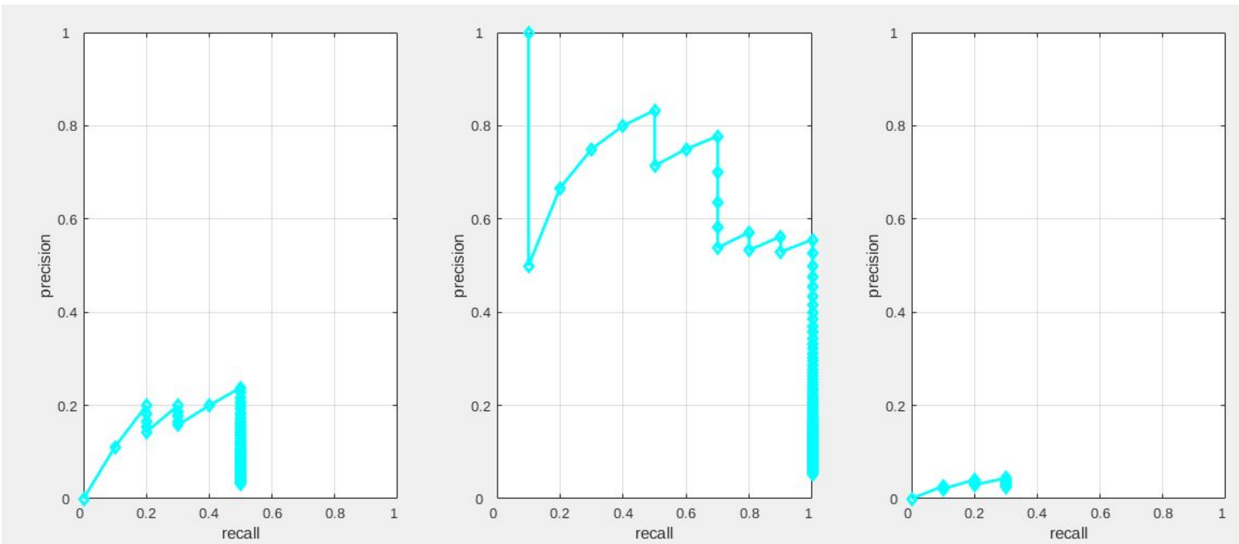
Misclassification error is

```
Mis-sclassification rate = 0.100000
>>
```

5. Dictionary size

1. Perform the localisation experiment using a very small dictionary and report the precision – recall curves. Hint: Cluster the descriptors into a small number of clusters (e.g. 20).

codebook_sz = [10 15 5];




```
Mis-sclassification rate = 0.400000
fx >>
```

2. Explain the drop in the performance.

From the above graphs where the class 1, class 2, class 3 has [10, 15, 5] as the descriptors. The ROC curve for class 1 and class 3 is worse than class 2. This shows that with less descriptors it becomes difficult to be precise and the model accuracy decreases. This is evident from the high overall misclassification rate (40%). The features which are required are encapsulated in the codebook, decreasing those will decrease the overall accuracy.