Computer Vision

Project 4 – Template Matching

Name – Gowri Kurthkoti Sridhara Rao

Introduction

Template matching is a technique that is used in classification tasks. A part of the image is chosen as template and this template is found in the target image. In this project, an image with text written in it is provided along with a text file that denotes the location of each letter. Using this information, template matching is carried out and the accuracy of the matching algorithm is computed.

The steps followed in process are as follows:

1. Template is created for the letter o, e, p, q, x using the input image and information in text file
2. Template is converted to zero-mean template by subtracting the mean value from each template pixel
3. Matched Spatial filter is implemented in which cross correlation is performed.
4. The output of matched spatial filter is then thresholded at a range of values and thresholded image is obtained from which detected matches are classified to TP,FP,TN,FN and TPR and FPR are calculated
5. ROC is plotted using TPR and FPR
6. Optimal T for each letter is found and a cross-confusion matrix is found

Code

1. Create template function

```
function [chars,coords,all_temp] = create_template(img,match)
%create_template This function creates template from the image for the
%given set of letters specified.
%   The function takes two inputs, first being the image and the second
%   being the match values for which templates has to be created. The
%   function returns characters read from the ground truth file,
%   coordinates of each charater, along with a cell that contains the
%   templates of each letter.
fileId = fopen("parenthood_gt.txt",'r'); %Reading the ground truth file
C = textscan(fileId,'%c %d %d'); %Scanning the file into a Cell
fclose(fileId);
chars = [C{1,1}]; %First value in the cell is converted to a character array
coords = [C{1,2},C{1,3}]; %Coordinate values are also coppied to a different
array

match_row=zeros(1,size(match,2)); %Array to hold the value of each char
location
for i = 1:size(match,2)
    for j = 1:size(chars,1)
        if match(i) == chars(j) %Find the first instance of the char in the
ground truth file
            match_row(i)=j; %Copy the row value of each first occurance of
character
```

```
            break; %Stop after first
        end
    end
end
% Using the locations in the char array, respective coordinates are found
% and a template is created for each based on the way it looks. Each of
% them are saved to a temporary variable to display the template and
% eventually it is put into a cell and returned to the main function.
all_temp = {};
o_temp       =       img(coords(match_row(1),2)-3:coords(match_row(1),2)+7,
coords(match_row(1),1)-4:coords(match_row(1),1)+5);              %figure();
imshow(o_temp);
e_temp       =       img(coords(match_row(2),2)-5:coords(match_row(2),2)+7,
coords(match_row(2),1)-3:coords(match_row(2),1)+5);              %figure();
imshow(e_temp);
p_temp=img(coords(match_row(3),2)-5:coords(match_row(3),2)+11,
coords(match_row(3),1)-4:coords(match_row(3),1)+5);       %       figure();
imshow(p_temp);
q_temp=img(coords(match_row(4),2)-3:coords(match_row(4),2)+13,
coords(match_row(4),1)-3:coords(match_row(4),1)+5);%              figure();
imshow(q_temp);
x_temp       =       img(coords(match_row(5),2)-4:coords(match_row(5),2)+9,
coords(match_row(5),1)-3:coords(match_row(5),1)+5);              %figure();
imshow(x_temp);

all_temp{1,1}=o_temp;
all_temp{1,2}=e_temp;
all_temp{1,3}=p_temp;
all_temp{1,4}=q_temp;
all_temp{1,5}=x_temp;
clear o_temp;clear e_temp;clear p_temp;clear q_temp;clear x_temp;
end
```

2. Cross Correlation function

```
function [corr_img] = cross_corr(img,template)
%cross_corr Computes cross correlation between image and template
%   The function takes image and template as two values and computes cross
%   correlation between them and returns it.
img = double(img); %Convert the image to double
[rows,cols]=size(template); %Determine the size of the template
corr_img = zeros(size(img)); %Initialize the correlation output image
for i=1:size(img,1)-rows-1
    for j=1:size(img,2)-cols-1
        Nimage = img(i:i+rows-1,j:j+cols-1); %Part of the image of size of
template
        corr = sum(sum(Nimage.*template)); %Correlation
        corr_img(i,j)= corr/sqrt(sum(sum(Nimage.^2))); %Assigning value to
        %correlation image after some modifications
    end
end
end
```

3. Thresholding function

```
function [thresh_img,img,detected] = thresholding(msf_img,th,img,dr,dc)
%thresholding Thresholds the msf image to a given threshold value and
%returns image with detected letters marked
%   Inputs to the function are msf image, threshold, dr and dc. The
```

```
%   function thresholds and outputs the thresholded image, image with
%   detected points marked and array with detected point values.
thresh_img = zeros(size(msf_img));
y=1;
for i=1:size(msf_img,1)
    for j=1:size(msf_img,2)
        if(msf_img(i,j)>th)
            thresh_img(i+dr-1:i+dr+1,j+dc-1:j+dc+1) = 255; %Set the value in
threshold image =255
            img(i+dr-1:i+dr+1,j+dc-1:j+dc+1) = 0; %Mark the detected values
to 0
            detected(1,y)=i+dr;
            detected(2,y)=j+dc;
            y=y+1;
        end
    end
end
end
```

4.

5.

6. Calculate TP function

```
function [TP] = calc_TP(thresh_img,rows,cols,dr,dc)
%calc_TP From the thresholded image, true positives in the detected points
%are found
%   Inputs to this function are thresholded image, true locations of the
%   letters, dr and dc values. This function output TP value
TP=0; %Initialize TP value to 0
for i = 1:size(cols,2)
    part_img        =        thresh_img(rows(1,i)-dr:rows(1,i)+dr,cols(1,i)-
dc:cols(1,i)+dc);
    flag=0; % Consider a part of the image of size of the template with
respect
    %to the true location of letter
    for x=1:size(part_img,1)
        for y=1:size(part_img,2)
            if(part_img(x,y)==255) %If a pixel value equals 255,
                TP=TP+1; %Increment TP
                flag=1; %Set flag = 1 to mark detected
                break; %Break out of first loop
            end
        end
        if(flag==1)
            break; %Break out of second loop
        end
    end
end
end
```

7. Main Script

```
clc; clear all; close all;
%% Creating templates
img = imread("parenthood.ppm");
match =['o','e','p','q','x'];
figure();
imshow(img);title("Input Image");
```

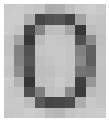
```
%%
[chars,coords,all_temp]=create_template(img,match);

%% Character that has to be matched
var = input('Pick a letter "o,e,p,q,x" (type 1 for "o" and so on)');
char = match(var);
template = double([all_temp{1,var}]);
t8 = uint8(template);
figure(); imshow(t8); title("Template");
%% Zero mean template
% Template has to be subtracted with it's mean to convert it into a
% zero-mean template
mean_template = template - mean(mean(template));
mean_template8 = uint8(mean_template);
figure,imshow(mean_template8); title("Zero mean template");
%% Padding
% With respect to the template size, padding is done to the input image
[dr,dc]=size(template);
dr = round((dr-1)/2); %Half of the number of rows
dc = round((dc-1)/2); %Half of the number of columns
img_pad = padarray(img,[dc,dr],'both'); %Padding image
figure(),imshow(img_pad); title("Padded Image");
%% Finding the true positions of template using ground truth file
j=1; %Iterative variable
clear cols;
clear rows;
for i = 1:size(chars,1)    %Iterate though characters
    if chars(i) == char    %If character is found in the chars array,
        cols(j) = coords(i,1); %Coppy the column and row value from coords
array
        rows(j) = coords(i,2);
        j=j+1;       %Increase count
    end
end
true_detected = img; %Visualizing the expected detected ouput
for i = 1:size(cols,2)
    true_detected(rows(1,i):rows(1,i)+3,cols(1,i):cols(1,i)+3)=0;
    %set expected letter center to 0
end
figure(), imshow(true_detected); title("True locations of template");
%% Calculating the Matched spatial filter image
%Cross correlation between padded image and the template
msf = cross_corr(img_pad,mean_template); %Cross correlation function is
called
msf   =   ((msf   -   min(min(msf)))./(max(max(msf))-min(min(msf)))).*255;
%Normalizing the msf image
msf8 = uint8(msf);
msf8 = msf8(dc+2:size(msf8,1)-(dc+3),dr+2:size(msf8,2)-(dr+5)); %Removing
padding
figure,imshow(msf8);title("MSF Image");
%% Thresholding
iter = 1;
clear TPR;
clear FPR;
for th = 180:10:250
    [thresh_img,out_img,detected]    =    thresholding(msf8,th,img,dr,dc);
%Thresholding MSF image
```

```
        thresh_img = uint8(thresh_img); %Converting threshold image into uint8
        %out_img = uint8(out_img);
        TP = calc_TP(thresh_img,rows,cols,dr,dc); %Calculating True positives
        FP = abs(size(detected,2)-TP); %Calculating FP
        FN = abs(size(cols,2)-TP); %Calculating FN
        TN = abs(size(chars,1)-size(cols,2)-FP); %Calculating TN
        disp('True positive=');disp(TP);
        disp('False positive=');disp(FP);
        disp('False negative=');disp(FN);
        disp('True negative=');disp(TN);
        TPR(iter) = (TP/(TP+FN)); %Calculating TPR
        FPR(iter) = (FP/(FP+TN)); %Calculating FPR
        iter=iter+1;
    end
    disp('TPR=');disp(TPR);
    disp('FPR=');disp(FPR);
    %% ROC curve
    %Plotting ROC curve
    figure(),plot(FPR,TPR,'r*');hold on;
    plot(FPR,TPR,'b');xlabel("FPR");ylabel("TPR"); title("ROC curve");
    %% Cross confusion matrix
    th =230;
    [thresh_img,out_img,detected] = thresholding(msf8,th,img,dr,dc);
    out_img = uint8(out_img);
    figure(), imshow(out_img); title("Teamplates detected for th=230");
    TP = calc_TP(thresh_img,rows,cols,dr,dc); %Calculating True positives
    FP = abs(size(detected,2)-TP); %Calculating FP
```

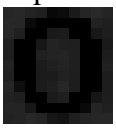
Results

Template matching was performed, and the MSF image was thresholded at value from 50 to 150 with an increment of 10. The ROC curve for each template is as follows.

1. **Template ‘o’**



Template ‘o’



Zero mean template

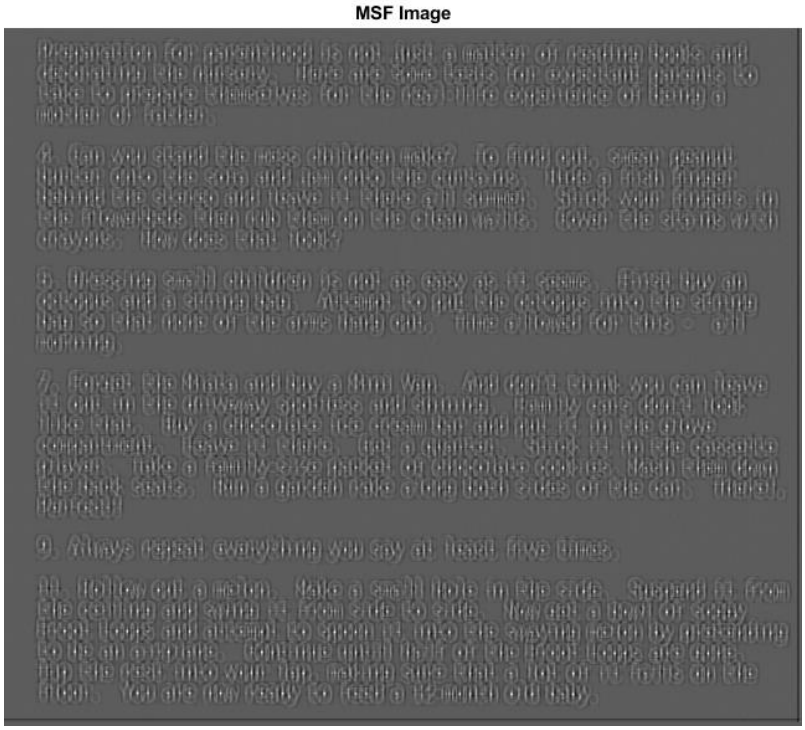
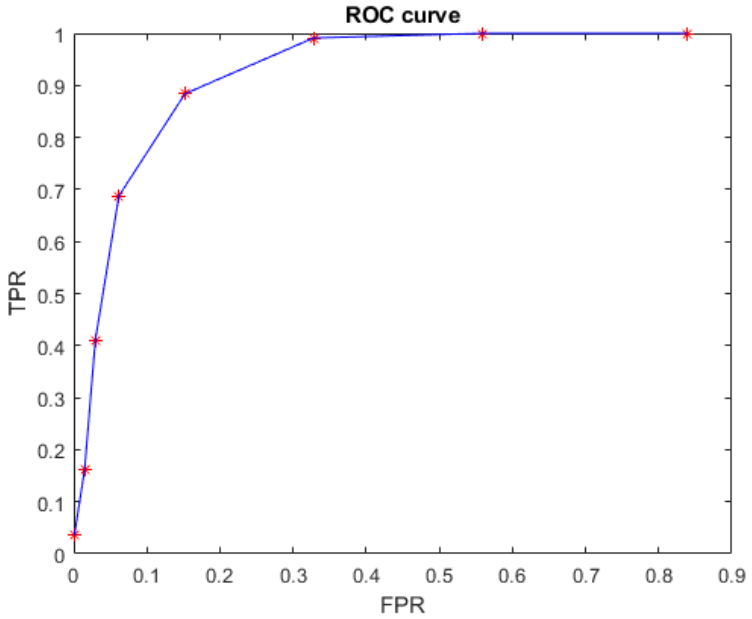
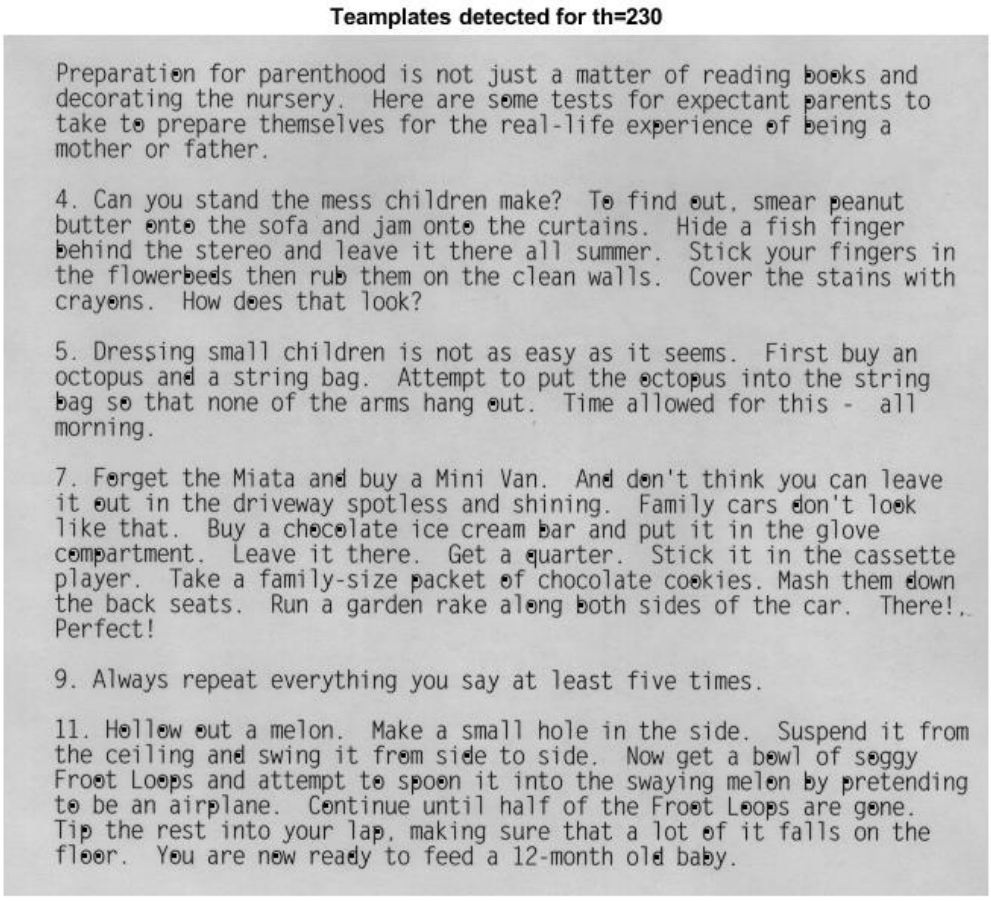


Image after correlation with ‘o’ template



ROC of ‘o’ template matching



Detected locations when threshold =230

Above is the image where all the ‘o’ letters are detected at a threshold of 230. In the above figure, templates similar to o were also detected.

O - 46 , e- 0,p-13 , q- 1,x-0, others -20 total = 80

2. **Template ‘e’**

‘e’ template

Zero mean template

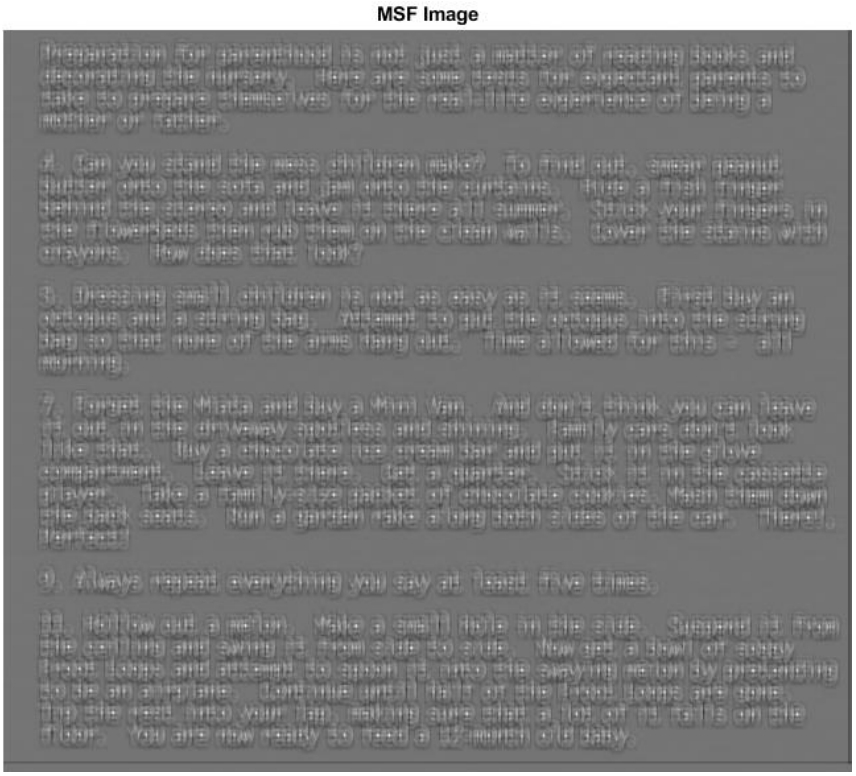
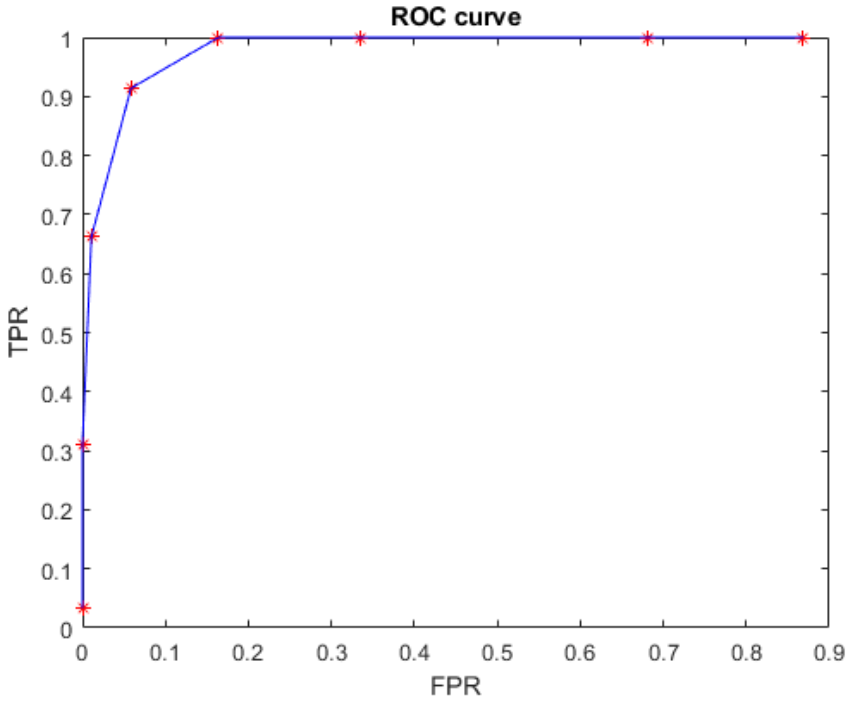
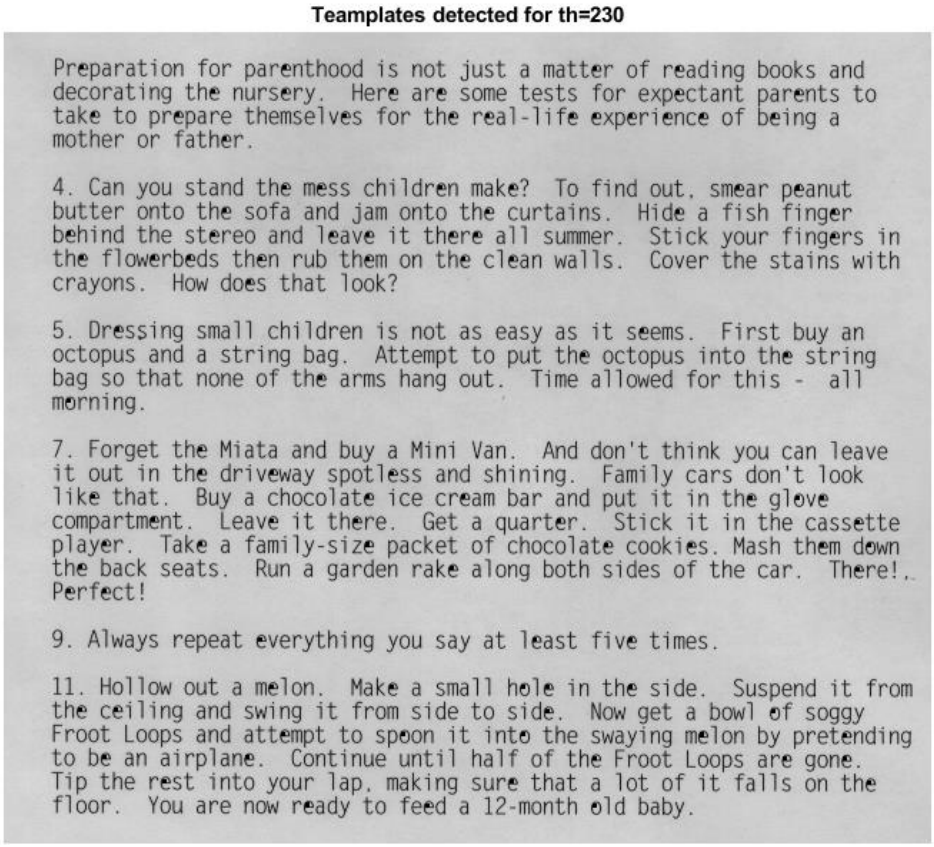


Image after cross correlation with ‘e’ template



ROC of ‘e’ template matching



Detected locations when threshold =230

Above is the image where all the ‘e’ letters are detected at a threshold of 230. In the above figure, templates similar to o were also detected.
O - 8 , e-100,p-0 , q- 0,x-0, others -4 total = 112

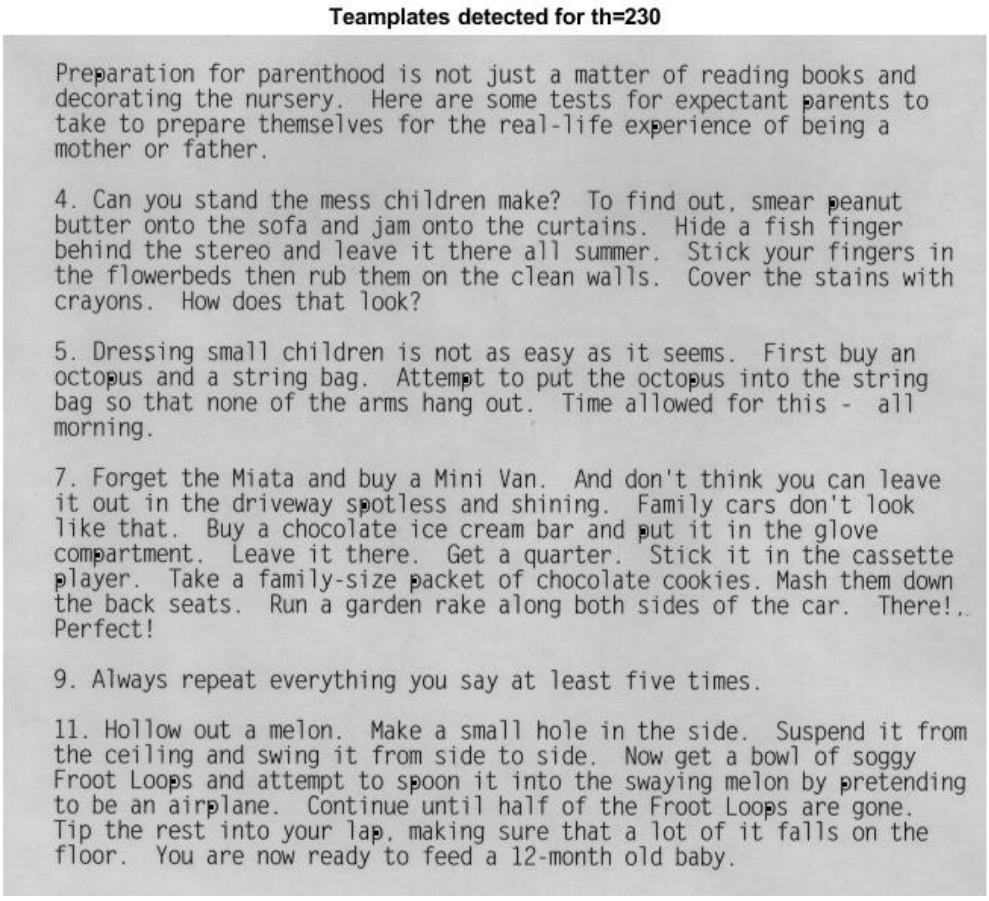
3. Template ‘p’

‘p’ template

Zero mean template

Image after cross correlation with ‘p’ template

ROC of ‘p’ template matching



Detected locations when threshold =230

Above is the image where all the ‘p’ letters are detected at a threshold of 230. In the above figure, templates similar to o were also detected.

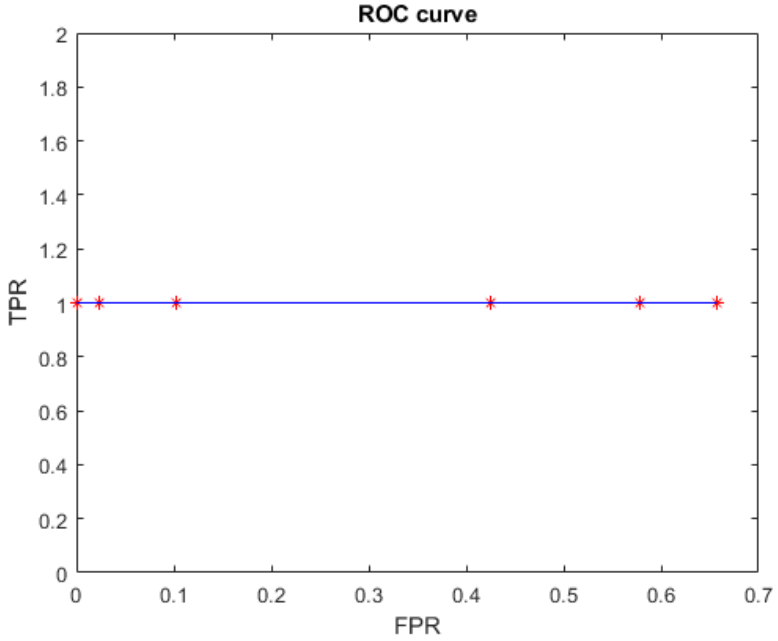
O - 0 , e-0,p-18 , q- 0,x-0, others -4 total = 22

4. **Template ‘q’**

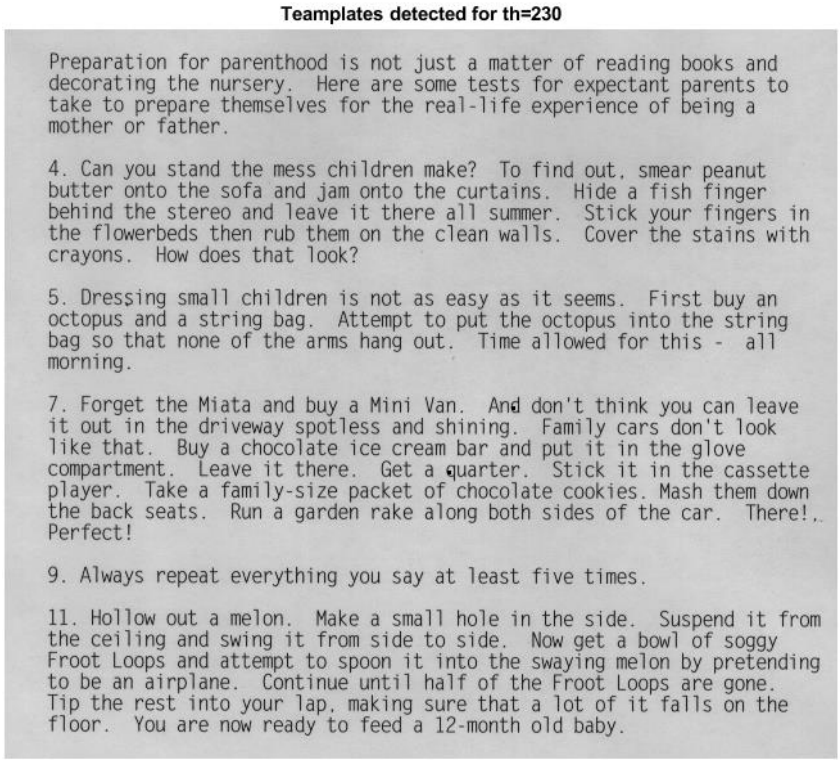
‘q’ template

Zero mean template

Image after correlation with ‘q’ template



ROC of ‘q’ template matching



Detected locations when threshold =230

Above is the image where all the ‘q’ letters are detected at a threshold of 230. In the above figure, templates similar to o were also detected.

O - 0 , e-0,p-0 , q-1 ,x-0, others -1 total = 2

5. **Template ‘x’**

‘x’ template

Zero mean template

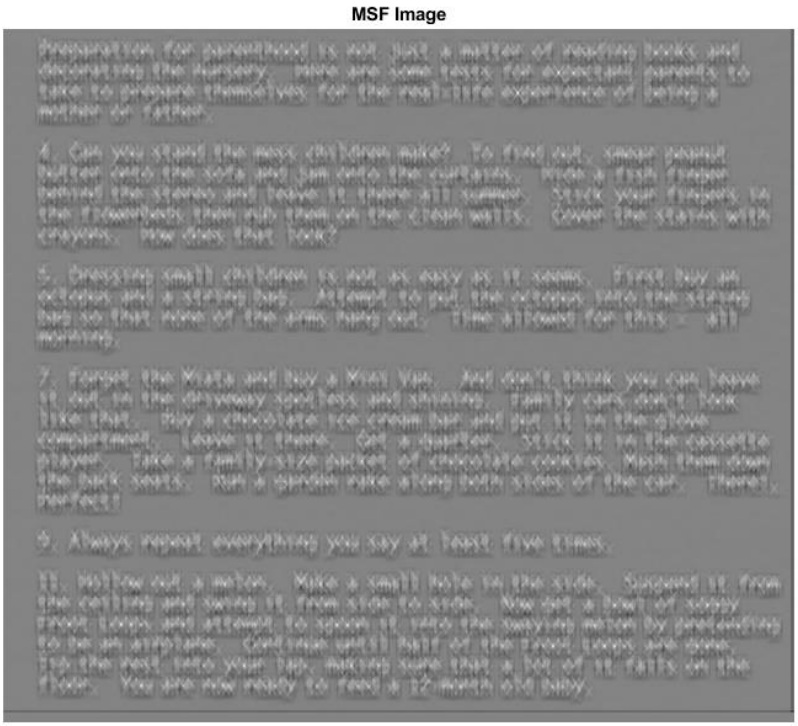
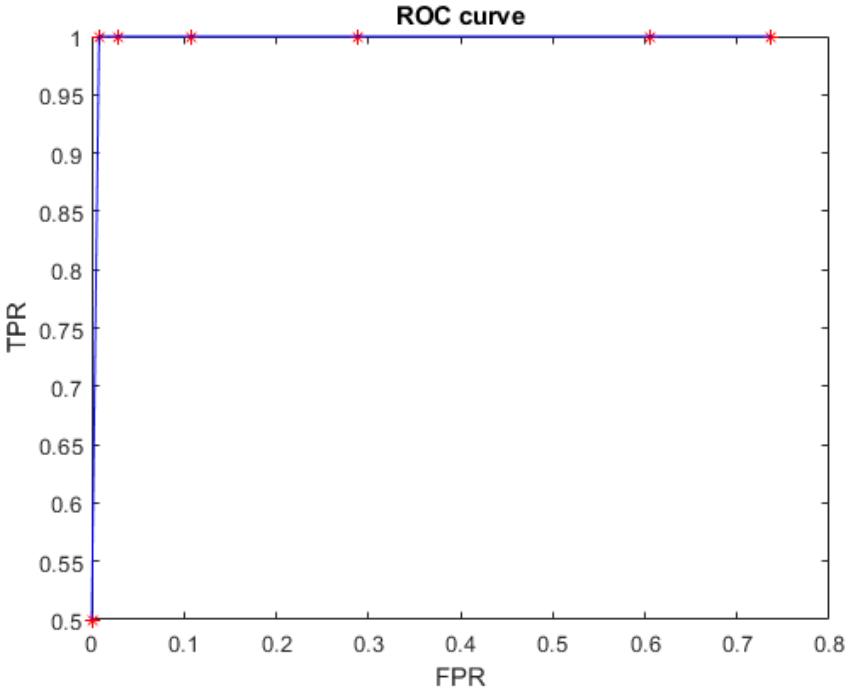
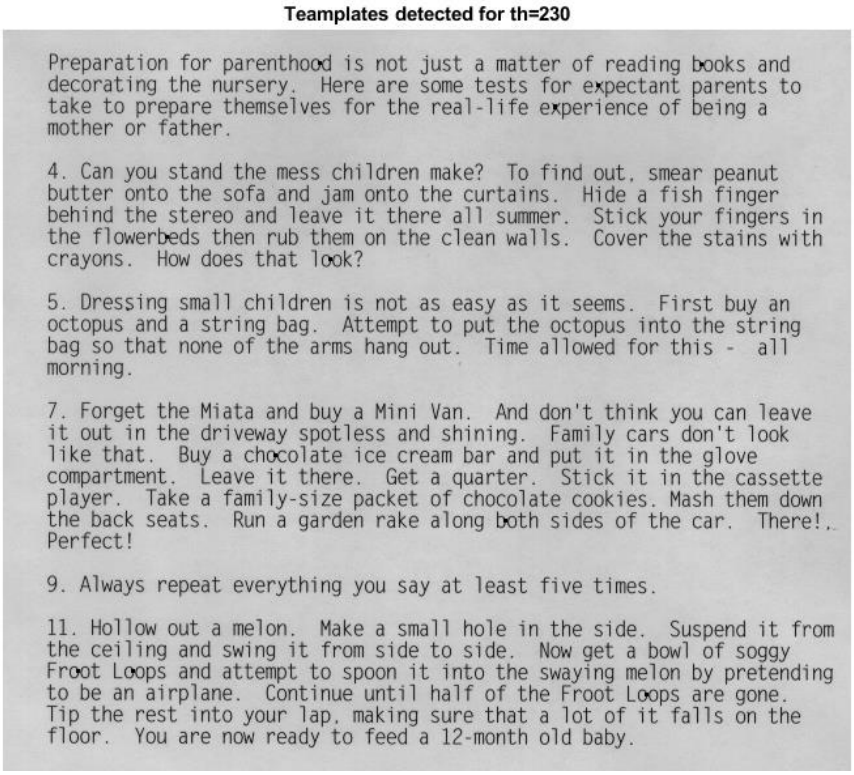


Image after correlation with ‘x’ template



Roc of ‘x’ template matching



Detected locations when threshold =230

Above is the image where all the ‘x’ letters are detected at a threshold of 230. In the above figure, templates similar to o were also detected.

O - 0 , e-0,p-0 , q-0 ,x-2, others -10 total = 12

Confusion Matrix

Confusion matrix is created using the information gathered from the output images of each template matching. Below is the confusion matrix that is generated:

	o	e	p	q	x	others
o	46/80 =57.5%	0	13/80 =16.25%	1/80 = 1.25%	0	20/80 =25%
e	8/112 =7%	100/112 =89%	0	0	0	4/112 =4%
p	0	0	18/22 =82%	0	0	4/22 =18%
q	0	0	0	1/2 = 50%	0	1/2 = 50%
x	0	0	0	0	2/12 = 17%	10/12 = 83%

The reduced accuracy in letters ‘q’ and ‘x’ is because, there is only 1 ‘q’ in the image and 2 ‘x’ in the image. Hence, even if one letter gets detected along with ‘q’, it will be considered false positive. Threshold has to be set to a very high value for the accuracy to increase to 100%. In ‘x’, the others

column has more values compared to other columns because, the matched spatial filter has detected the area between the letters like “oo”, “bo”, “be” etc. as ‘x’. Hence the reduced detection rate.

Conclusion

In this project, template matching of 5 letters was performed on an image with letters. Matched Spatial Filter was used for template matching and the results were compared for different threshold values and ROC curve for each template was plotted. Confusion matrix is generated considering the threshold value to be 230 and the results are reported in this document.