به نام خدا گزارش کار پروژه دوم کامران کیانی 810101497

بخش اول:

در مرحله اول همان طور که در دستور کار امده است با دستور uigetfile مطابق شکل زیر عکس را انتخاب و محتویات ان را در picture میریزیم

```
clear;
% SELECTING THE TEST DATA
[file,path]=uigetfile({'*.jpg;*.bmp;*.png;*.tif'},'Choose an image');
s=[path,file];
picture=imread(s);
```

که در ان ابتدا فرمت های قابل قبول را تعریف میکنیم که jpg , bmp, png,tif هستند و بالای پنجره انتخاب عکس choose an image مینویسیم بعد از ان در متغیر file نام فایل و در متغیر path مسیر عکس ذخیره شده اند که محتویات فایل را در متغیر s ریخته و s عکس ما میشود حال با دستور imread عکس را باز کرده و در ماتریس picture ذخیره میکنیم که سطر و ستون ان هر عددی میتواند باشد اما ماتریس سه بعدی است مثلا 144*616*3

در مرحله دوم با استفاده از دستور imresize طول و عرض تصویر را تغیر میدهیم

```
8 -    picture=imresize(picture,[300 500]);
```

که حال ماتریس picture یک ماتریس 300*500*3 خواهد بود

در مرحله سوم باید سه بعد را به یک بعد تبدیل کنیم تا عکسی سیاه و سفید داشته باشیم

```
%RGB2GRAY
gray=mygrayfun(picture);
```

از تابع mygrayfun استفاده میکنیم که بصورت زیر است .

```
function gray=mygrayfun(image)
    gray=zeros(300,500);
    for i=1:300
        for j=1:500
            gray(i,j)=image(i,j,1)*0.299+image(i,j,2)*0.578+image(i,j,3)*0.114;
        end
    end
end
```

از طرفی میدانیم ماتریس RGB است یعنی درایه اول red درایه دوم green و درایه سوم blue است هر کدام از پیکسل ها را در ضریبی خاص که در صورت پروژه داده شده است ضرب کرده تا در نهایت فقط عکس ما طول و عرض وداشته باشد بدین صورت که با دو حلقه تو در تو ابتدا روی سطر ها (300) و بعد روی ستون ها (500) جابجا شده و هر پیکشل را بعد R را در 0.299 و بعد G را در 0.578 و بعد B را در 0.114 ضرب میکنیم و جمع انان را در درایه متاظر ماتریس gray قرار میدهیم .

در مرحله چهارم میخواهیم تصویر را binary کنیم و چون تصویر از قبل سیاه و سفید است یعنی هر پیکسل عددی بین 0 تا 255 دارد حال باید یک استانه قرار بدهیم که اگر از ان بیش تر بود ان را سیاه (0) در نظر گرفته و اگر از ان کمتر بود ان را سفید (1) در نظر بگیرد

```
thero=100;
picbinary=mybinaryfun(gray,thero);
```

که تابع mybinaryfun بصورت زیر است

```
function picbinary=mybinaryfun(pic,thero)
    picbinary=zeros(300,500);
    for i=1:300
        for j=1:500
            if pic(i,j)<=thero
                picbinary(i,j)=1;
            elseif pic(i,j)>thero
                picbinary(i,j)=0;
            end
        end
    end
end
```

که در ان ورودی اول تابع همان عکس سیاه و سفید است و ورودی دوم حد استانه که در اینجا 100 در نظر گرفتیم دوباره بکمک دو حلقه تو در تو ابتدا روی سطر ها (300) و بعد روی ستون ها(500) جابجا میشویم و بعد میگوییم اگر پیکسلی که روی ان هستیم از حد استانه کمتر بود درایه متناظر با ان را در ماتریس picbinary 1 (سفید) و اگر بیش تر بود ان را 0 در نظر بگیریم .

حال ماتریس picbinary یک ماتریس 500*300 با فرمت double با مقادیر 0 و 1 است .

که ان را نمایش میدهیم


```
figure
subplot(1,2,1)
imshow(picbinary)
```

در مرحله پنجم باید تا حد امکان تکه های کوچکی که حاوی مقادیر مفید اطلاعات نیستند را حذف کرد (مثلا در مثال ما پرچ های پلاک یا هر گونه نویز اضافی رو پلاک)

```
% Removing the small objects and background
CC=myremovecom(picbinary,300);
background=myremovecom(picbinary,3300);
picc=CC-background;
picc=~mybinaryfun(picc,thero);
subplot(1,2,2)
imshow(picc)
```

که در ان بکمک تابع myremovecom پیکسل های بهم چسبیده کوچکتر از مثلا در اینجا 300 پیکسل را جدا کرده و عکس را در CC میریزیم بعد از ان دوباره بکمک myremovecom این بار پیکسل های بهم چسبیده کوچکتر از 3300 که عددی بسیار بزرگ است را حذف میکنیم و ان را در background میریزیم گویی تمام عکس بجز مثلا قاب دور و ... حذف میشود البته ممکن است قاب حذف نشود به دلیل این که اگر قاب را یکپارچه تشخیص ندهد تعداد پیکسل های قاب هم کمتر از مثلا 3300 میشود و حذف میشود اما به هر حال تشکیل background کاری مفید است

در نهایت تصویرcc که اشیا کوچک فقط حذف شده اند را منهای background که تقریبا همه چی بجز پس زمینه حذف شده است میکنیم تا ماتریس picc را بدست بیاوریم بعد از ان چون picc از حالت binary در امده دوباره با تابع mybinaryfun که در بالا نوشته ایم ان را binary و فقط صفر و یک میکنیم و در نهایت تصویر را نمایش میدهیم

همانطور که در عکس نهایی دیده میشود جای پرچ ها و مقداری نویز حذف شده است.

حال به سراغ تابع myremovecom که نوشته ایم میرویم

```
function CC=myremovecom(picbinary,n)
 CC = zeros(300, 500);
 flag = 1;
 memory = zeros(300, 500);
 for i = 1:300
    for j = 1:500
        if picbinary(i, j) == 1 && memory(i, j) == 0
            S= [i, j];
            memory(i, j) = flag;
            while ~isempty(S)
                pixel = S(1, :);
                S(1, :) = [];
                for k = -1:1
                    for l = -1:1
                        new_x = pixel(1) + k;
                        new_y = pixel(2) + l;
                        if new_x >= 1 && new_y >= 1 && new_x <= 300 && new_y <= 500 && ...
                           picbinary(new_x, new_y) == 1 && memory(new_x, new_y) == 0
                           S = [S; new_x, new_y];
                           memory(new_x, new_y) = flag;
                        end
                    end
                end
            end
            size = sum(memory(:) == flag);
            if size >= n
                CC(memory == flag) = 255;
            end
            flag = flag + 1;
```

که در آن ورودی عکس مدنظر و تعداد پیکسلی که اگر از ان کمتر بود حذف کند است

بعد از ان ماتریس CC که خروجی نهایی است را ابتدا تعریف میکنیم و بعد از ان ماتریس memory که تعداد پیکسل های بهم چسبیده را قرار است هر بار در خودش ذخیره کند تعریف میکنیم

بعد از ان وارد سطر ها میشویم و بعد از ان وارد ستون ها میشویم و شروع به پیمایش میکنیم میدانیم تمام تصویر بجز جاهایی که اعداد یا حرف هستند صفر است پس پیمایش تا رسیدن به اولین 1 (سفید) ادامه پیدا میکند اگر پیکسلی برابر 1 بود و درایه متناظر با ان در ماتریس memory برابر صفر بود :

ابتدا سطر و ستون ان را در متغیر S ذخیره میکنیم و بعد از ان درایه متناظر در ماتریس memoryرا نیز 1 میکنیم

بعد از ان S که مشخصات پیکسلی است که سفید بوده است(بطور کلی فرض میکنیم S ماتریس n سطر در 2 ستون است) را سطر اولش را در ماتریس pixel ذخیره میکینم که ماتریس pixel در حقیقت مختصات اولین 1است که دو درایه دارد حال به سراغ همسایه های پیکسل (i,j) میرویم که همسایه های ان پیکسل های (i-1,j-1),(i-1,j),(i-1,j-1),(i+1,j+1),(i+1,j),(i+1,j-1),(i,j-1),(i,j+1),(i-1,j+1) هستند گویی در مرکز یک مربع 3 در 3 هستیم و همسایه های ان پیکسل های اطراف هستند برای این که 8 تا پیکسل ننویسیم از دو حلقه تو در تو برای پیمایش روی طول و رووی عرض عکس استفاده میکنیم و new_x و new_y را با توجه به عددی که در حلقه هستیم انتخاب میکنیم حال میگوییم اگر

```
new_x >= 1 && new_y >= 1 && new_x <= 300 && new_y <= 500
```

که یعنی اگر در اول و اخر عکس نباشیم و همچنین

```
picbinary(new_x, new_y) == 1 && memory(new_x, new_y) == 0
```

باشد یا به طوری اگر پیکسل همسایه پیکسل قبل 1 باشد و در ماتریس memoryمقدار ان 0 باشد :

به ماتریس سطر بعدی ماتریس s مقادیر new_x و new_y اضافه میشود یعنی گویی ما مختصات هایی که پیمایش کرده ایم را در ماتریس S ذخیره میکنیم .

و درایه متناظر با مختصات new_x و new_y را در ماتریس memoryنیز 1(سفید) میکنیم .

این کار تا زمانی ادامه پیدا میکند که تمامی همسایه های ان مقدار 1 بگیرند (یک عدد نمیتواند دوبار حساب شود چون در ماتریس S ذخیره شده است)

حال ماتریس memoryبرای یکی از اشیای روی تصویر مقادیر را 1 را دارد و برای بقیه تصویر صفر است برای این که حد استانه را چک کنیم تعداد 1 ها را با sum جمع کرده و در متغیر size ذخیره میکنیم اگر بیش تر از حد استانه بود مقدار ان را 255 (یا همان سفید) قرار میدهیم

دلیل عدد 255 هم این است که دوباره در اینجا ما از logical خارج شده ایم و در سیاه و سفید قرار داریم اگر 0 باشد (سیاه) و اگر 255 باشد به نوعی دیگر 1 خالص(یا سفید) است

همین طور سراغ شی های بعدی در شکل رفته و دونه دونه در حلقه ها بررسی میشود و و تمام مشخصات خانه های سفید در s و کل عکس در memory ذخیره میشود و حد استانه چک میشود و سراغ object بعدی میرویم تا اخر عکس

در مرحله ششم:

قصد داریم segmentبندی انجام بدهیم

```
% Labeling connected components
[L,Ne]=mysegmention(picc);
propied=regionprops(L,'BoundingBox');
hold on
for n=1:size(propied,1)
    rectangle('Position',propied(n).BoundingBox,'EdgeColor','g','LineWidth',2)
end
hold off
```

که با تابع mysegmention این کار را انجام دادیم بعد از ان خروجی این تابع که L و Ne که تعداد عناصر در Segment بندی است را در می اوریم Lدر حقیقت ای نشکلی است که شی اول را تمام عناصر ان 1 است شی دوم تمام عناصر 2 شی سوم تمام عناصر 3 و ....

بعد از ان با دستورات اماده regionprops و retangle تمام عناصری که مقادری غیر یک دارند کادر بندی شده و به زنگ سبز نمایش داده میشوند

حال به سراغ تابع mysegmention میرویم :

```
function [L,Ne] = mysegmention(picc)
  flag = 1;
  L = zeros(300,500);
  for i = 1:500
     for j = 1:300
         if picc(j, i) == 1 && L(j, i) == 0
             S = [j, i];
             L(j, i) = flag;
             while ~isempty(S)
                 current_pixel = S(1, :);
                 S(1, :) = [];
                 for k = -1:1
                     for l = -1:1
                         new_x = current_pixel(1) + k;
                         new_y = current_pixel(2) + l;
                         if new_x >= 1 && new_x <= 300 && new_y >= 1 && new_y <= 500 &&...
                             picc(new_x, new_y) == 1 && L(new_x, new_y) == 0
                         S = [S; new_x, new_y];
                         L(new_x, new_y) = flag;
                         end
                     end
                 end
             end
             Ne=flag;
             flag = flag + 1;
         end
     end
  end
  end
```

که این تابع دقیقا مشابه تابع قبلی برای جداسازی است با این تفاوت که ورودی ان حتما باید فقط logicalباشد که در قسمت قبل این کار را انجام دادیم و تفاوت دیگر این است که در هر مرحله flagعوض میشود یعنی برای شی اول عددی که جایگزین میشود عدد 1 است برای شی دوم دو و ... واگرنه در تشخیص اشیا به هم چبسیده و ... هیچ تفاوتی با قبلی ندارد

در ضمن در این تابع ابتدا ستونی بررسی شده و بعد از ان سطری به دلیل این که اگر سطری بررسی شود ممکن است شکل پلاک کج باشد و حرف یا عدد اخر عدد اول خوانده شود و در اخر تابع هم تعداد اشیا (یا همان flag) در هر مرحله را Ne نامیده که هر بار یکی اضافه میشود و خروجی تابع است .

مرحله هفتم:

در این مرحله باید map set را لود کنیم و هر کدام را با هر کدام از سگمنت هایمان تطبیق بدهیم

```
% Loading the mapset
load TRAININGSET;
totalLetters=size(TRAIN,2);
figure
final_output=[];
t=[];
for n=1:Ne
    [r,c]=find(L==n);
    Y=picc(min(r):max(r),min(c):max(c));
    Y=imresize(Y,[42,24]);
    imshow(Y)
    pause(0.2)
    ro=zeros(1,totalLetters);
    for k=1:totalLetters
        ro(k)=corr2(TRAIN{1,k},Y);
    end
    [MAXRO,pos]=max(ro);
    if MAXRO>.45
        out=cell2mat(TRAIN(2,pos));
        final_output=[final_output out];
    end
end
```

که در ان ابتدا با دستور load مپ ست را لود کردیم بعد از ان تعداد حروفی و اعدادی که در مپ ست از قبل تعین شده است را با دستور size در متغیر totalLetters میریزیم که در اینجا 62 است (چون حروف بزرگ و کوچک و اعداد جدا تعریف شده اند)

بعد از ان متغیر final_output و t را تعریف میکنیم و از 1 تا Ne که همان تعداد اشیایی که در شکل داریم وارد حلقه میشویم

با دستور find(L==n) درابتدا شی اول در L را پیدا میکنیم و از طرفی چون میدانیم مپ ست ما هر عکس 42*24 است هر کدام از سگمنت ها را با imresize طول و عرضشان را یکی میکنیم و شکل ان را نمایش میدهیم و 0.2 ثانیه برای دیدن عکس برنامه را متوقف میکنیم

حال ماتریس جدید ro به اندازه حروفی که در مپ ست داریم تعریف میکنیم و با حلقه و با دستور corr2 که کورلیشن میگیرد بررسی میکنیم ببینیم هر کدام از سگمنت ها چقدر مطابقت داردند با مپ ست ما و مطابقت هر کدام را با هر کدام از مپ ست در ماتریس Ro میریزیم در نهایت ماکسیمم را برداشته و به عنوان جواب انتخاب مکنیم

حال یک نکته مهم را باید برررسی کنیم مثلا اگر در مرحله background نتوانستیم قاب را حذف کنیم یا هر چیز اضافه ای در عکس بود و مطابقت ان با تمام مپ ست بسیار کم بود باید ان را حذف کنیم یا به اصطلاح باید برای تطابق ها theroshold بگذاریم

که با شرط اگر 0.45<MAXRO باشد گویی این کار را میکنیم و میگوییم خانه شماره (2,pos) را از مپ ست برداشته و با دستور cell2mat ان را به نوشته تبدیل کرده و در outذخیره میکنیم و در نهایت out را به final_output اضافه میکنیم

مرحله هشتم:

حال باید جواب را چاپ و در یک فایل txt ذخیره کنیم

```
% Printing the plate
file = fopen('number_Plate.txt', 'wt');
fprintf(file,'%s\n',final_output);
fclose(file);
winopen('number_Plate.txt')
```

که با دستور fopen فایل number_plate را باز کرده (منظور از wt این است که میتوانیم روی ان بنویسیم)

بعد با دستور fprintf میگوییم file را باز کن و بصورت %s (یا کاراکتری) final_output را روی ان بنویس

و با دستور fclose فایل را میبندیم یا به اصطلاح ذخیره میشود

در نهایت هم با دستور winopen فایل را برای دیدن باز میکنیم

***من تمامی توابع را در اسکریپت p1.m نوشته ام**

بخش دوم:

مپ ستی که دارم شامل حروف ب /ج/د/س/ص/ط/ق/ل/م/ن/و/ه/ی/اعداد 0 تا 9 است

که نام mapset برابر TRAININGFARSISET است و وقتی ان را لود میکنیم FTRAIN را به ما میدهد که 2*33 است و هر کدام از انان 50*60 هستند

نحوه کار دقیقا و دقیقا مشابه سوال یک است با این تفاوت به جای mygrayfunاز تابع اماده rgb2gray و بجای تابع mybinaryfun از تابع اماده imbinarize و بجای تابع myremovecom از تابع اماده bwarreaopen و بجای تابع mysegmentation از تابع اماده bwlabel استفاده میکنیم

کد بصورت زیر است

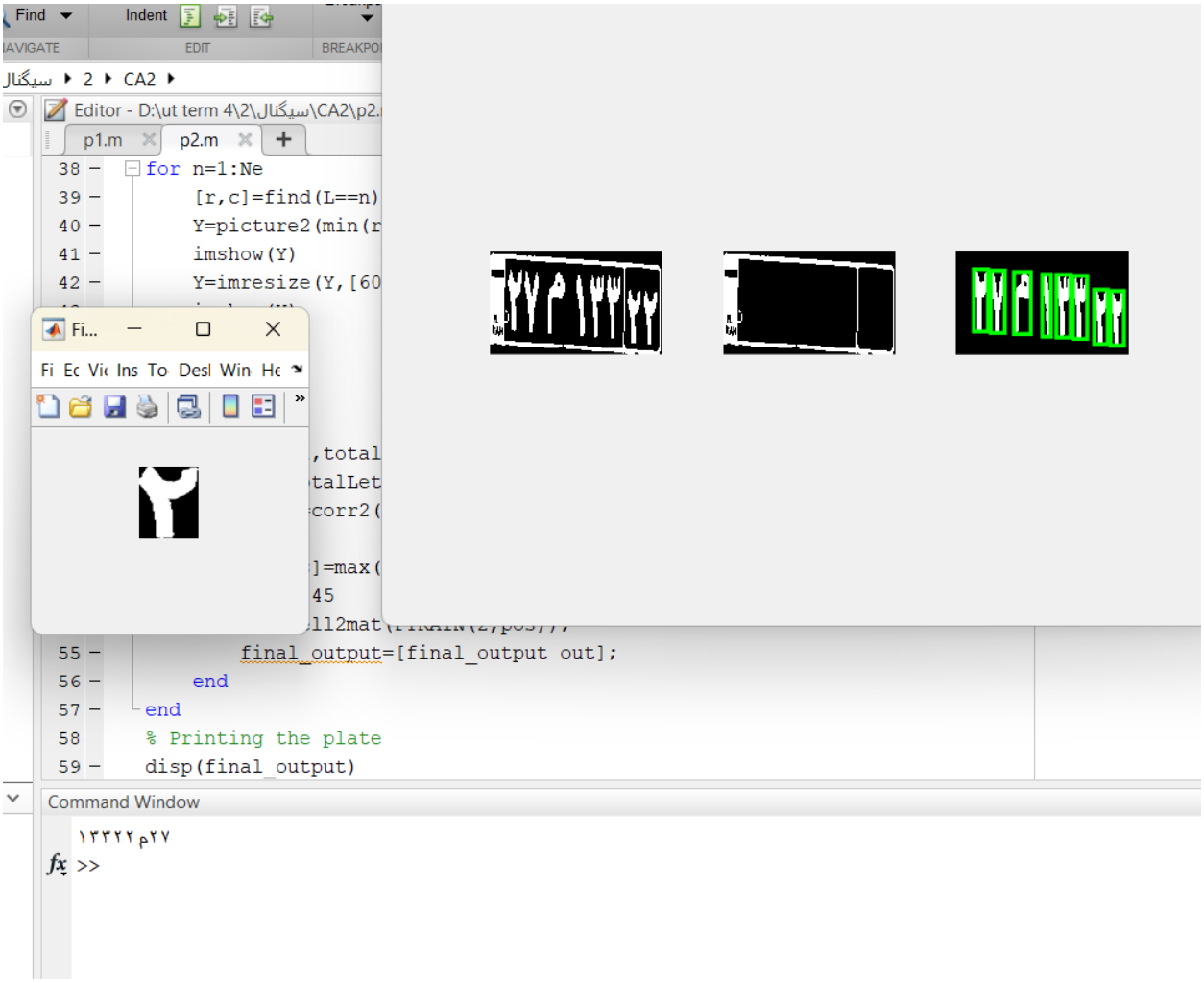
```
4       % SELECTING THE TEST DATA
5 -     [file,path]=uigetfile({'*.jpg;*.bmp;*.png;*.tif'},'Choose an image');
6 -     s=[path,file];
7 -     picture=imread(s);
8 -     picture=imresize(picture,[300 500]);
9       %RGB2GRAY
10 -    picture=rgb2gray(picture);
11      % THRESHOLDIG and CONVERSION TO A BINARY IMAGE
12 -    threshold = graythresh(picture);
13 -    picture =~imbinarize(picture,threshold);
14      % Removing the small objects and background
15 -    picture = bwareaopen(picture,500);
16 -    subplot(1,3,1)
17 -    imshow(picture)
18 -    background=bwareaopen(picture,5000);
19 -    subplot(1,3,2)
20 -    imshow(background)
21 -    picture2=picture-background;
22 -    subplot(1,3,3)
23 -    imshow(picture2)
```

```
24      % Labeling connected components
25 -    [L,Ne]=bwlabel(picture2);
26 -    propied=regionprops(L,'BoundingBox');
27 -    hold on
28 -    for n=1:size(propied,1)
29 -        rectangle('Position',propied(n).BoundingBox,'EdgeColor','g','LineWidth',2)
30 -    end
31 -    hold off
32      % Loading the mapset
33 -    load TRAININGFARSISET;
34 -    totalLetters=size(FTRAIN,2);
35 -    figure
36 -    final_output=[];
37 -    t=[];
38 -    for n=1:Ne
39 -        [r,c]=find(L==n);
40 -        Y=picture2(min(r):max(r),min(c):max(c));
41 -        imshow(Y)
42 -        Y=imresize(Y,[60,50]);
43 -        imshow(Y)
44 -        pause(0.2)
```

```
45 -        if n==7
46 -            hg=1;
47 -        end
48 -        ro=zeros(1,totalLetters);
49 -        for k=1:totalLetters
50 -            ro(k)=corr2(FTRAIN{1,k},Y);
51 -        end
52 -        [MAXRO,pos]=max(ro);
53 -        if MAXRO>.45
54 -            out=cell2mat(FTRAIN(2,pos));
55 -            final_output=[final_output out];
56 -        end
57 -    end
58      % Printing the plate
59 -    disp(final_output)
```

در نهایت پلاک در command window نمایش داده میشود فقط این که به دلیل حروف فارسی ممکن است برعکس چاپ شود .

بطور مثال



این خروجی ما به ازای شکل ورودی است .

بخش سوم:

در این بخش ابتدا شکل های بسیار غیر معقول برای پلاکک مثل طول یا عرض یا ارتفاع زیاد را حذف میکنیم بعد از ان پلاک را از عکس جدا کرده و مثل قسمت 2 عمل میکنیم

```
%remove extra data
[L,Ne]=bwlabel(picture);
for i = 1:Ne
    U = find(L == i);
    [r,c] = find(L == i);
    if length(U) > 400 || (max(r) - min(r)) > 35 || (max(c) - min(c)) > 35
        for j = 1:length(U)
            picture(U(j)) = 0;
        end
    end
    if (max(r) - min(r)) < 7 || (max(c) - min(c)) < 3
        for j = 1:length(U)
            picture(U(j)) = 0;
        end
    end
end
figure
imshow(picture);
```

که در ان پیکسل های با ارتفاع بیش تر از 400 و طول و عرض بیش تر 35 تا حذف میشوند (طول و عرض را با کم کردن max ازmin در هر شی بدست می اوریم

بعد از ان انهایی که بسیار کوچیکند در حد 3 4 5 پیکسل را حذف میکنیم (دوباره با کم کردن max از min)

شکل بصورت زیر خواهد بود

حال با resize کردن عکس دوباره مثل قسمت 2 است