Univerzitet u Sarajevu
Elektrotehnički fakultet u Sarajevu
Odsjek za računarstvo i informatiku

Leaf recognition

Prepoznavanje oblika i obrada slike
Projektni zadatak

Studenti:
Akšamović Jasmin
Bahtić Nejra
Ahmetović Tarik

Sarajevo, 2018.

Sadržaj

Poglavlje 1

Dataset

Cilj projekta je da sa priložene slike program bude u stanju da prepozna listove na slici.

Link na projekat na githubu `https://github.com/nejrabahtic/PrepoznavanjeOblika`.

Data set je u prilogu projekta.

Data set ima 3 klase.

Klase su:

- Biljka1
- Biljka2
- Biljka3

Svaka klasa ima 60 uzorka. Klase se nalaze u folderu Klase.

Poglavlje 2

DataPreprocessing

2.1 DataPrep1

Anotacije se nalaze u csv fajlovima vezanim za svaku sliku.
Anotacije nisu kreirane automatski.

10% slika svake klase je odvojeno u folderima Validacija.

Maske su kreirane i kao takve su sačuvane u folderima Maske.

Kreiranje maske za jednu sliku implementirano je u globalnoj funkciji mask(x, y, m, n, brojListova) pomoću funkcije poly2mask(x, y, m, n).

```
1 function [bw] = mask(x, y, m, n, brojListova)
2     bw = poly2mask([0,0,0,0], [0,0,0,0], m, n);
3     for i=1:brojListova
4         bwPom = poly2mask(x(i, :), y(i, :), m, n);
5         bw = bw + bwPom;
6     end
```

Kreiranje maske za sve slike u jednom folderu implementirano je u globalnoj funkciji importAnn().

```
1 function [] = importAnn()
2     foldercsv = dir('../../CSV/CSVBiljka1/*.csv');
3     folderslike = dir('../../Slike/SlikeBiljka1/*.png
    ');
4     vel = [];
5     m = 0;
6     n = 0;
7     [folderslikesize, br] = size(folderslike);
8     [foldercsvsize, br] = size(foldercsv);
9     for poredu=1:folderslikesize
```

```
        folderslike(poredu).name
        s = imread(folderslike(poredu).name);
        [m, n, d] = size(s);
        vel = [vel; m, n];
    end
    for poreduu=1:foldercsvsize
        filename = foldercsv(poreduu).name;
        csv = csvread(filename);
        [brojListova,brojKolona] = size(csv);
        x = [];
        y = [];
        for i=1:brojListova
            x = [x; csv(i, 2:2:9)];
            y = [y; csv(i, 3:2:9)];
        end
        bw = [];
        [velsize, br] = size(vel);
        bw = mask (x, y, vel(poreduu,1), vel(poreduu
   ,2), brojListova);
        str = erase(foldercsv(poreduu).name, ".csv");
        fpath = strcat(str, '_mask.png');
        imwrite(bw, fpath);
    end
```

Maske su primjenjene na slike su sačuvane u folderima PrimjenaMaski.

Primjenjivanje maske za jednu sliku implementirano je u globalnoj funkciji primijeniMasku(slika, maska).

```
function [konv] = primijeniMasku(slika, maska)
    [m, n, d] = size(slika);
    konv = zeros(m, n, d);
    for i = 1:m
        for j = 1:n
            if maska(i, j) ~= 255
                slika(i, j, :) = [0, 0 , 0];
            end
        end
    end
    konv = slika;
end
```

Primjenjivanje maski za sve slike u jednom folderu implementirano je u globalnoj funkciji convolution().

```
function [] = convolution()
    folderslike = dir('../../Slike/SlikeBiljka3/*.png');
    foldermaske = dir('../../Maske/MaskeBiljka3/*.png');

    [folderslikesize, br] = size(folderslike);

    rgbC = [];

    for i = 1:folderslikesize
        rgbS = imread(folderslike(i).name);
        rgbM = imread(foldermaske(i).name);
        rgbC = primijeniMasku(rgbS, rgbM);
        imshow(rgbC);
        str = erase(folderslike(i).name, ".png");
        fpath = strcat(str, '_maskApply.png');
        imwrite(rgbC, fpath);
    end
```

2.2 DataPrep2

2.2.1 Uklanjanje šuma

Uklanjanje šuma je omogućeno korištenjem funkcija imnoise, filter2 i na kraju medfilt2. Slike su sačuvane u folderima BezŠuma.

Uklanjanje šuma je implementirano glavnom funkcijom removenoisefilter().

```
function [] = removenoisefilter()
    folderslike = dir('../../Slike/SlikeBiljka3/*.png');
    [folderslikesize, br] = size(folderslike);
    for slika=1:folderslikesize
        folderslike(slika).name
        s = rgb2gray(imread(folderslike(slika).name));
        J = imnoise(s,'salt & pepper',0.02);
        Kaverage = filter2(fspecial('average',3), J)/255;
        Kmedian = medfilt2(J);
        imshow(Kmedian);
        str = erase(folderslike(slika).name, ".png");
        fpath = strcat(str, '_removeNoiseFilter.png');
```

```
13          imwrite(Kmedian, fpath);
14      end
```

Maskiranje neoštrina je primjenjeno sa k = 1.2. Slike su sačuvane u folderima Neoštrine.

Maskiranje neoštrina je implementirano u globalnoj funkciji unsharpingmask().

```
1  function [] = unsharpmasking()
2      folderslike = dir('../../Slike/SlikeBiljka1/*.png
       ');
3      [folderslikesize, br] = size(folderslike);
4      for slika=1:folderslikesize
5          rgb = imread(folderslike(slika).name);
6          gray = rgb2gray(rgb);
7          imshow(gray);
8          h = fspecial('average', 3);
9          gray_z = imfilter(slika, h);
10         maska = gray - gray_z;
11         k = 1.2;
12         grayFinal = gray + k*maska;
13         imshow(grayFinal,[]);
14         str = erase(folderslike(slika).name, ".png");
15         fpath = strcat(str, '_unasharpMasking.png');
16         imwrite(grayFinal, fpath);
17     end
```

2.3 DataPrep3

2.3.1 Poboljšanje kvaliteta slike

Filteri koji su korišteni su:

- Kontrast - Slike su sačuvane u folderima Kontrast
- Osvjetljenje - Slike su sačuvane u folderima Osvjetljenje
- Ujednačavanje histograma - Slike su sačuvane u folderima Histogram

Za kontrast korišten je noise filter speckle. Pomoću njega smo postigli dodavanje šuma ponajviše na zelenim nijansama listova na slikama biljaka.

Kontrast je implementiran u globalnoj funkciji noisefilter().

```
function [] = noiseFilter()
    folderslike = dir('../../Slike/SlikeBiljka1/*.png');
    [folderslikesize, br] = size(folderslike);
    for slika=1:folderslikesize
        rgb = imread(folderslike(slika).name);
        J = imnoise(rgb,'speckle', 0.08);
        imshow(J);
        str = erase(folderslike(slika).name, ".png");
        fpath = strcat(str, '_noiseFilter.png');
        imwrite(J, fpath);
    end
```

Za osvjetljenje je korišten filter imreducehaze u kombinaciji sa filterom imcomplement.

Osvjetljenje je implementirano u globalnoj funkciji enhancementFilter().

```
function [] = enhancementFilter()
    folderslike = dir('../../Slike/SlikeBiljka1/*.png');
    [folderslikesize, br] = size(folderslike);
    for slika=1:folderslikesize
        rgb = imread(folderslike(slika).name);
        rgbInv = imcomplement(rgb);
        pomInv = imreducehaze(rgbInv, 'ContrastEnhancement', 'none');
        pom = imcomplement(pomInv);
        imshow(pom);
        str = erase(folderslike(slika).name, ".png");
        fpath = strcat(str, '_enhancementFilter.png');
        imwrite(pom, fpath);
    end
```

Za ujednačavanje histograma je korišten filter histeq.

Ujednačavanje histograma je implementirano u globalnoj funkciji histogramfilter().

```
function [] = histogramfilter()
    folderslike = dir('../../Slike/SlikeBiljka1/*.png');
    [folderslikesize, br] = size(folderslike);
```

```
    for slika=1:folderslikesize
        rgb = imread(folderslike(slika).name);
        gray = rgb2gray(rgb);
        histgray = histeq(gray);
        hfig = figure;
        imhist(histgray, 64);
        str = erase(folderslike(slika).name, ".png");
        fpath = strcat(str, '_histogramFilter.png');
        saveas(hfig, fpath);
    end
```

2.4 DataPrep4

Slike su podijeljene u dva disjunktna podskupa train i test i sačuvane su u folderima Train i Test. Svaka klasa je imala 60 slika.

Slike su podijeljene na način 10% (6 slika) Validacija, a ostale 54 slike su podijeljene na Train i Test grupe. Train sadrži 38 slika po klasi (70%), a Test 16 slika po klasi (30%).

Odvajanje u dva diskjunktna skupa je implementirano u globalnoj funkciji test_and_train().

```
from sklearn.model_selection import train_test_split
from save_image import saveImage
import os
import numpy as np
from PIL import Image
import random
import cv2

path_biljka1 = 'C:/Users/NejraBahtic/Desktop/POOS/
   Slike/SlikeBiljka1'
path_biljka2 = 'C:/Users/NejraBahtic/Desktop/POOS/
   Slike/SlikeBiljka2'
path_biljka3 = 'C:/Users/NejraBahtic/Desktop/POOS/
   Slike/SlikeBiljka3'

valid_images = [".jpg",".jpeg", '.png']

br = 1
image_list = []

for f in os.listdir(path_biljka1):
    ext = os.path.splitext(f)[1]
```

```
20      if ext.lower() not in valid_images:
21          continue
22      image = Image.open(os.path.join(path_biljka1, f))
23
24      image_list.append(image)
25
26 random.shuffle(image_list)
27
28 train_data, test_data = train_test_split(image_list,
       test_size=0.2)
29
30 br=1
31 for img in train_data:
32      newpath = 'C:/Users/NejraBahtic/Desktop/POOS/
       Divide/Train'
33      img = np.array(img)
34      saveImage(cv2.cvtColor(img, cv2.COLOR_BGR2RGB),
       newpath, str(br)+".jpg")
35      br += 1
36
37 for img in test_data:
38      newpath = 'C:/Users/NejraBahtic/Desktop/POOS/
       Divide/Test'
39      img = np.array(img)
40      saveImage(cv2.cvtColor(img, cv2.COLOR_BGR2RGB),
       newpath, str(br)+".jpg")
41      br += 1
```

Poglavlje 3

Specifikacija detalja oko projekta

3.1 Izbor modela za prepoznavanje koji odgovara problemu

Izabrani model je

```
1 model = svm.SVC(kernel='linear', C=1.0)
```

3.2 Izbor deskriptora koji odgovara problemu

Izabrani deskriptor je **HOG** *(Histogram of Oriented Gradient)*. Koraci pri kreiranju deskriptora su:

- Preprocessing
- Calculate the Gradient Images
- Calculate Histogram of Gradients in ells
- Calculate the HOG feature vector
- Apply HOG

3.3 Izbor metoda poboljšavanja koje će biti primijenjene nad slikama

Metod za poboljšavanje predstavljen je funkcijom unsharpmasking() koju smo iskoristili nad regionima od interesa. Regioni od interesa nalaze se u folderu Deskriptor, subfolderi SL1, SL2, SL3.

Poglavlje 4

Kreiranje deskriptora svih slika

Regioni od interesa nalaze se u folderu Deskriptor, subfolderi Deskriptor-Biljka{1,2,3}.

```
1  import cv2
2  import os
3  import numpy as np
4  from PIL import Image
5  from skimage.feature import hog
6  from resizeimage import resizeimage
7
8  imageDir = "/home/nejra/Documents/PrepoznavanjeOblika
       /Deskriptor/SL1"
9  hogPath = "/home/nejra/Documents/PrepoznavanjeOblika/
       Deskriptor/DeskriptorBiljka1"
10
11 def createDescriptor(image):
12     image = resizeimage.resize_cover(image, [50, 50])
13     gray = cv2.cvtColor(np.array(image), cv2.
       COLOR_BGR2GRAY)
14     fd, hog_image = hog(image, orientations=8,
       pixels_per_cell=(12, 12), cells_per_block=(4, 4),
       block_norm='L2', visualize=True)
15
16     return fd, hog_image
17
18 hog_images = []
19 hog_features = []
20 image_path_list = []
21
22 valid_images = [".jpg", ".jpeg", ".png"]
23 br = 1
24
25 for file in os.listdir(imageDir):
```

```
26        extension = os.path.splitext(file)[1]
27    if extension.lower() not in valid_images:
28      continue
29    image_path_list.append(os.path.join(imageDir, file)
      )
30
31 for imagePath in image_path_list:
32      img = Image.open(imagePath)
33
34      fd, hog_image = createDescriptor(img)
35      hog_images.append(hog_image)
36      hog_features.append(fd)
37
38      fullpath = os.path.join(hogPath, 'hog_'+imagePath
      [56:])
39      cv2.imwrite(fullpath, hog_image)
40
41      br += 1
```

Poglavlje 5

Kreiranje i treniranje modela za prepoznavanje

U projekat je dodan novi folder TrainNew u koji su dodane i slike bez listova. Slike bez listova su labelirane kao 0, dok su ostale labelirane kao 1.

```
1  import cv2
2  import numpy as np
3  import os, os.path
4  from PIL import Image
5  from sklearn import svm
6  from skimage.feature import hog
7  from sklearn.externals import joblib
8  from createDescriptor import *
9
10 trainData = []
11 labels = []
12 valid_images = [".jpg",".jpeg", ".png"]
13
14 imageDir = "/home/nejra/Documents/PrepoznavanjeOblika
      /TrainNew"
15
16 for f in os.listdir(imageDir):
17     name, ext = os.path.splitext(f)
18     if ext.lower() not in valid_images:
19         continue
20     if (f == '1.jpg' or f == '2.jpg' or f == '3.jpg'
      or f == '4.jpg' or f == '5.jpg' or f == '6.jpg' or
      f == '7.jpg' or f == '8.jpg' or f == '9.jpg' or f
      == '10.jpg' or f == '11.jpg' or f == '12.jpg' or f
      == '13.jpg' or f == '14.jpg' or f == '15.jpg' or f
      == '16.jpg' or f == '17.jpg' or f == '18.jpg' or f
      == '19.jpg' or f == '20.jpg'):
21   labels.append(0)
```

```
22      else:
23    labels.append(1)
24
25      image = Image.open(os.path.join(imageDir,f))
26      hf, hi = createDescriptor(image)
27      trainData.append(hf)
28
29 clf = svm.SVC(kernel='linear', C=1.0)
30 clf.fit(np.array(trainData), labels)
31
32 #eksport modela
33 model = 'final_leaf_model.sav'
34 joblib.dump(clf, model)
```

Poglavlje 6

Testirati model sa testnim podacima

6.1 Izračunati performanse modela: sp, sens, acc

```
cm1 = confusion_matrix(labels, pred)
print('Confusion Matrix : \n' % cm1)

total1=sum(sum(cm1))

accuracy1=(cm1[0,0]+cm1[1,1])/total1
print ('Accuracy : ', accuracy1)

sensitivity1 = cm1[0,0]/(cm1[0,0]+cm1[0,1])
print('Sensitivity : ', sensitivity1 )

specificity1 = cm1[1,1]/(cm1[1,0]+cm1[1,1])
print('Specificity : ', specificity1)
```

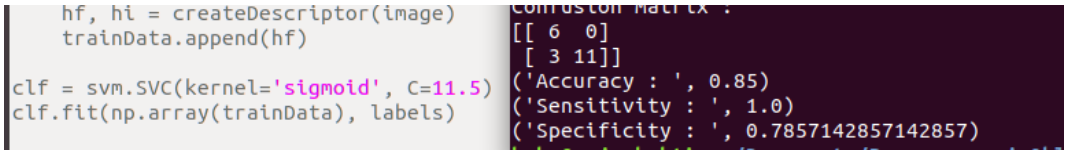
```
Confusion Matrix :
[[ 6  0]
 [ 3 11]]
('Accuracy : ', 0.85)
('Sensitivity : ', 1.0)
('Specificity : ', 0.7857142857142857)
```

Poglavlje 7

Poboljšavanje performansi modela za prepoznavanje na osnovu performansi testiranja modela

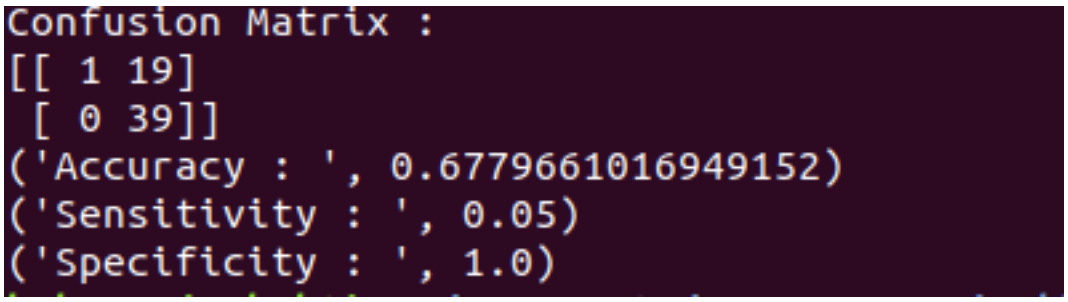
7.1 Izmjena parametara odgovarajućeg modela

Model koji smo koristili jeste linear. Ukoliko izmjenimo u 'sigmoid' dobijemo iste performanse, što je prikazano na slici ispod



7.2 Drugačija podjela podataka na trening/test skup

Kada zamijenimo TrainNew i TestNew dobijemo performanse na slici ispod



7.3 Clustering

Zbog jednostavnosti modela i podataka nad kojima je rađen model, zaključujemo da korištenje clusteringa ne bi na bilo koji način poboljšalo performanse modela. Zbog toga clustering nije urađen.

Set podataka se sastoji od listova istih osobina te zbog toga nije bilo ciljnih osobina na osnovu kojih bi clustering mogao biti urađen.

7.4 Izbacivanje outliera

Izbačena je slika koja je uticala na kreiranje performansi modela.

Poglavlje 8

Export modela za prepoznavanje

```
1 #eksport modela
2 model = 'final_leaf_model.sav'
3 joblib.dump(clf, model)
```

Poglavlje 9

Primijeniti poboljšavanja ili konverzije nad slikama koja su primijenjena nad onim iz seta za treniranje

Primijenjeni su drugi filteri, sve funkcije su napisane u okviru ovog dokumenta.

Poglavlje 10

Kreirati deskriptore slika koji su korišteni pri treniranju

Kreirani su deskriptori na osnovu drugih foldera koji su definisani u ovom dokumentu.