


سوال دوم: الف) استخراج اشیا تصویر سه کاناله:

منبع: [OpenCV: Contours : Getting Started](https://docs.opencv.org/4.x/d1/d32/tutorial_py_contour_properties.html)https://docs.opencv.org/4.x/d1/d32/tutorial_py_contour_properties.htmlقبل از هر چیز اگر کتابخانه ای را نصب نداریم، آن را با دستور `!pip install` نصب میکنیم: `!pip install keras-visualizer`

```

Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Collecting keras-visualizer
  Downloading keras_visualizer-2.4-py3-none-any.whl (5.4 kB)
Installing collected packages: keras-visualizer
Successfully installed keras-visualizer-2.4

```

سپس `import` های لازم قرار داده شده را اجرا کردیم.

برای پیاده سازی تابع `compactness` ابتدا باید تصویر را تک کاناله یا خاکستری کنیم که من از `Otsu` برای اینکار استفاده کردم. سپس `Contour` های آن را بیابیم و با استفاده از آن محیط و مساحت شکل مورد نظر را پیدا کنیم بعد اشیا را استخراج کنیم.

```

1 def compactness(image):
2     gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
3     blurred = cv2.GaussianBlur(gray, (5,5),0)
4
5     _, OTsu = cv2.threshold(gray,0,255,cv2.THRESH_BINARY+cv2.THRESH_OTSU)
6     contours, _ = cv2.findContours(OTsu.copy(), cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)
7     max_contour = max(contours, key=cv2.contourArea)
8     S = 4 * np.pi * float(cv2.contourArea(max_contour))
9     P = cv2.arcLength(max_contour, closed=True)
10    P = P**2
11    compactness_score = S / P
12    return compactness_score

```

برای تکمیل تابع `eccentricity` هم باید مراحل بالا را تا پیدا کردن `contour` ها انجام دهیم، سپس از تابع `cv2.fitEllipse` برای فیت کردن یک بیضی روی یک شی استفاده میکنیم. این تابع کانتور ها را در ورودی میگیرد و در خروجی `(x,y)` را به عنوان مرکز بیضی و `(majorAxis, minorAxis)` به عنوان قطر کوچک و بزرگ بیضی و `angle` را به عنوان زاویه ی چرخش بیضی برمیگرداند. برای اینکه این تابع درست کار کند و ارور ندهد و بتواند حداقل ۵ کانتور در همه ی تصاویر پیدا کند اعداد مختلفی را برای حذف نویز امتحان کردم و در نهایت چون نتیجه نگرفتم برای حالت هایی که کمتر از ۵ نقطه پیدا میشود -۱ ریترن کردم.

```

1 def eccentricity(image):
2     gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
3     blurred = cv2.GaussianBlur(gray, (5,5),0)
4
5     _, OTsu = cv2.threshold(blurred,0,255,cv2.THRESH_BINARY+cv2.THRESH_OTSU)
6     contours, _ = cv2.findContours(OTsu.copy(), cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)
7     max_contour = max(contours, key=cv2.contourArea)
8     if len(max_contour) < 5:
9         return -1
10    ((x,y), (majorAxis, minorAxis), angle) = cv2.fitEllipse(max_contour)
11    eccentricity_score = np.sqrt(1 - (majorAxis / minorAxis)**2)
12    return eccentricity_score

```

برای تکمیل بخش `solidity` هم ابتدا همان مراحل قبل را تا پیدا کردن کانتور ها ادامه میدهم سپس از تابع `cv2.convexHull` استفاده میکنیم و فرمول مربوطه را پیاده سازی مینماییم.

```

1 def solidity(image):
2     gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
3     blurred = cv2.GaussianBlur(gray, (5,5),0)
4
5     _, OTsu = cv2.threshold(blurred,0,255,cv2.THRESH_BINARY+cv2.THRESH_OTSU)
6     contours, _ = cv2.findContours(OTsu.copy(), cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)
7     max_contour = max(contours, key=cv2.contourArea)
8     S = float(cv2.contourArea(max_contour))
9     convex = cv2.convexHull(max_contour)
10    hS = cv2.contourArea(convex)
11    solidity_score = S / hS
12    return solidity_score

```

سوال دوم: (ب) محاسبه ی هیستوگرام LBP:

منبع: [Local Binary Pattern for texture classification — skimage v0.19.2 docs \(scikit-image.org\)](https://scikit-image.org/docs/v0.19.2/skimimage.html)

<https://numpy.org/doc/stable/reference/generated/numpy.histogram.html>

ابتدا تصویر را تک کاناله میکنیم. سپس با تابع `local_binary_pattern(img, n_points, radius, METHOD)` هیستوگرام `lbp` آن را حساب میکنیم. سپس از تابع `numpy.histogram(a, bins=10, range=None, density=None, weights=None)` برای محاسبه ی هیستوگرام استفاده کردیم. این تابع دو خروجی دارد که اولی آرایه ای شامل مقادیر هیستوگرام و دومی آرایه ای از `bin edge` هاست.

```

1 def histogram_of_LBP(image, numPoints, radius, eps=1e-7):
2     one_channel_image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
3     bins = np.arange(0, numPoints + 3)
4     range = (0, numPoints + 2)
5     lbp = feature.local_binary_pattern(one_channel_image, numPoints, radius, "uniform")
6     hist, bin_edges= np.histogram(lbp.ravel(), bins=bins, range=range)
7     hist_values = hist.astype("float")/(hist.astype("float").sum() + eps)
8     return hist_values

```

سوال دوم: (ج) خواندن دیتا و آموزش و تست

توابع پیاده سازی شده را با استفاده از تابع `validating_func` رو دو تا از تصویر ها امتحان کردیم:

```
1 validating_func("dataset/ship/2196336.jpg", "dataset/airplane/airplane1.jpg")
```

Result for ship image:

compactness is : 0.19646747199902298

eccentricity is : 0.9934357842652797

solidity is : 0.8249648052557484

Result for airplane image:

compactness is : 0.16123865611799323

eccentricity is : 0.7850845697375121

solidity is : 0.9248123180500923

سوال دوم: (د) استخراج ویژگی تصاویر:

برای تکمیل تابع `get_featureMatrix` باید اطلاعات تصاویر را به صورت آرایه ای ذخیره کنیم.

```

1 def get_featureMatrix(data):
2     feature_matrix = []
3     data = np.array(data)
4     for im in data:
5         im = np.array(im)
6         h = [compactness(im), eccentricity(im), solidity(im), histogram_of_LBP(im, 8, 1)[0], histogram_of_LBP(im, 8, 1)[1]]
7         feature_matrix.append(h.copy())
8
9     return np.array(feature_matrix)

```

و در نهایت مدل خواسته شده را به صورت زیر ساختیم:

```
1 print(feature_matrix_train)
2 classifier = svm.LinearSVC()
3 classifier.fit(feature_matrix_train, y_train)
```

```
[[0.5107296 0.83043092 0.9568531 0.02152423 0.05233578]
 [0.39202215 0.59742074 0.92268966 0.02192283 0.04944595]
 [0.04285039 0.92610289 0.52283276 0.03623246 0.07101004]
 ...
 [0.43334203 0.95968813 0.91408771 0.02331792 0.0411352 ]
 [0.41540985 0.84036864 0.84261964 0.02321827 0.04444356]
 [0.10148655 0.29208802 0.80157453 0.02547034 0.06305804]]
LinearSVC()
```

سوال دوم: ح)

```
1 #test on test dataset
2 test_feature = get_featureMatrix(x_test)
3 test_label = classifier.predict(test_feature)
4 accuracy = accuracy_score(y_test, test_label)
5 print('accuracy on test data set:', accuracy*100, '%')
```

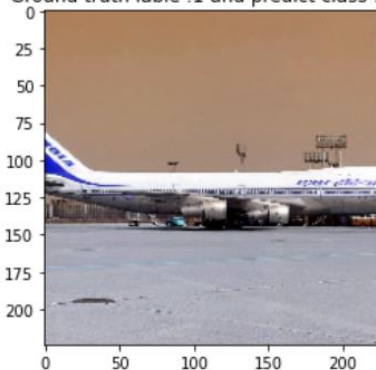
accuracy on test data set: 78.125 %

دقت مدل روی داده ی تست ۷۸ درصد شد.

سوال دوم: خ)

```
1 #test visualize
2 index = random.randint(0, len(x_test)-1)
3 prediction = classifier.predict(get_featureMatrix(np.array([x_test[index]])))
4 plt.title(f"Ground truth lable :{y_test[index]} and predict class : {prediction}")
5 plt.imshow(x_test[index])
6 plt.show()
```

Ground truth lable :1 and predict class : [1]



در این نمونه ی رندوم که عکس هواپیما است باید خروجی ۱ میشد که مدل درست تشخیص داده و ۱ برگردانده است .