به نام خدا

درس مبانی بینایی کامپیوتر

تمرین سری دهم

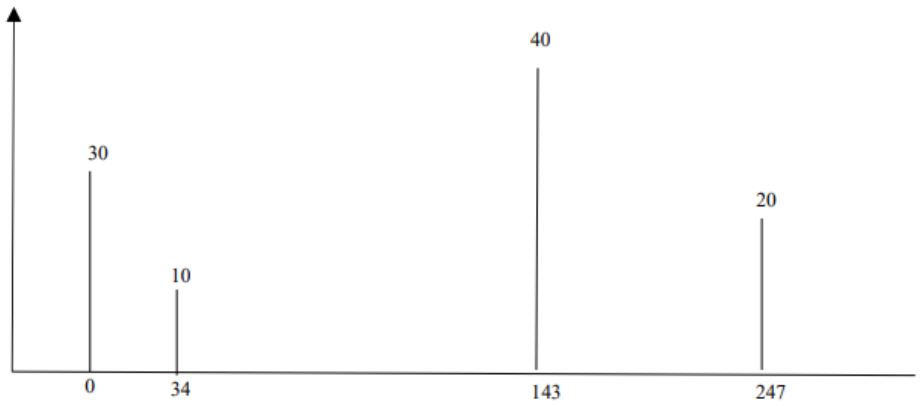
مدرس درس:
جناب آقای دکتر محمدی

تهیه شده توسط:
الناز رضایی ۹۸۴۱۱۳۸۷

تاریخ ارسال: ۱۴۰۱/۱۰/۲۹

سوال ۱:

فرض کنید هیستوگرام LBP_8^2 یک تصویر که به اندازه ۲۷۰ درجه چرخانده شده و شدت روشنایی پیکسل‌های آن نصف شده است، در شکل زیر باشد. هیستوگرام LBP_8^2 تصویر اصلی (بدون چرخش و بدون تغییر شدت روشنایی پیکسل‌های آن) را ترسیم کنید. برای هر دو حالت هیستوگرام LBP_8^2 یکنواخت و مستقل از چرخش را هم رسم کنید و باهم مقایسه کنید. (۲۰ نمره)



پاسخ ۱:

در ابتدا باید اعداد موجود در هیستوگرام زیر را به باینری ۸تایی تبدیل کنیم؛ زیرا در صورت سوال، LBP با ۸ نقطه و شعاع ۲ را داده است.

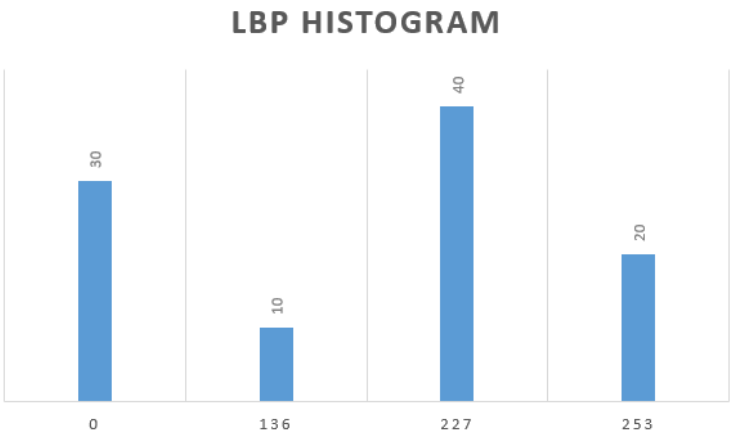
$$0 = (00000000)_2$$
$$34 = (00100010)_2$$
$$143 = (10001111)_2$$
$$247 = (11110111)_2$$

در صورت سوال، گفته شده است که تصویر ۲۷۰ درجه به صورت پادساعتگرد چرخانده شده است، بنابراین برای به دست آوردن هیستوگرام تصویر اصلی و بدون چرخش، باید ۲۷۰ درجه به صورت پادساعتگرد بچرخانیم یا ۹۰ درجه به صورت ساعتگرد بچرخانیم؛ زیرا با چرخش ۹۰ درجه در جهت ساعتگرد، بر ۳۶۰ که معادل همان ۰ درجه چرخش است، منطبق می‌شود. شیفت ۹۰ درجه

ساعتگرد، همان شیفت ۲ بیت به سمت راست است. بنابراین اعداد باینری که در مرحله قبل به دست آوردیم، ۲ بیت به راست شیفت چرخشی می‌دهیم. در مورد شدت روشنایی نیز می‌دانیم که هیستوگرام با ضریب شدت روشنایی تغییر نمی‌کند. بنابراین تغییر ضریب آن تاثیری ندارد.

$$(00000000)_2 = (00000000)_2 = 0$$
$$(00100010)_2 = (10001000)_2 = 136$$
$$(10001111)_2 = (11100011)_2 = 227$$
$$(11110111)_2 = (11111101)_2 = 253$$

بنابراین هیستوگرام تصویر اصلی، مطابق شکل زیر بوده است.

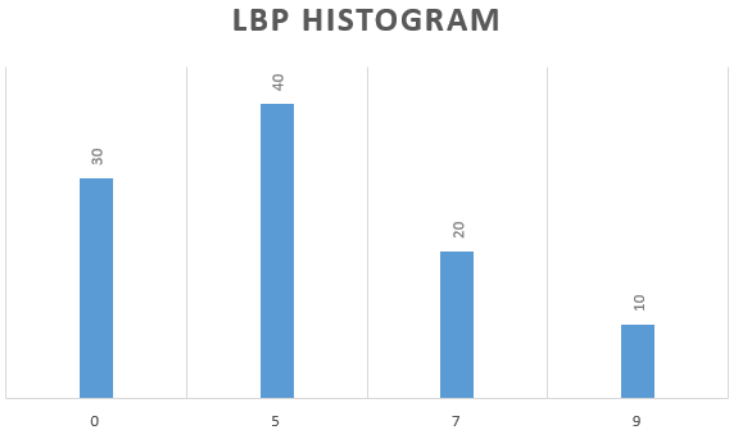


به الگوهایی که کمتر از ۳ تغییر بین ۰ و ۱ داشته باشند، یکنواخت گفته می‌شود. بنابراین طبق این تعریف، ۰، ۳۴ و ۲۴۷ از هیستوگرام اولیه یکنواخت هستند و ۰، ۲۲۷ و ۲۵۳ از هیستوگرام دوم یکنواخت هستند. همچنین مستقل از چرخش نیز یعنی فقط تعداد ۱ها مهم است و جایگاه آن مهم نیستند؛ لذا الگوهایی که تعداد یکسان ۱ دارند را به عدد یکسانی map می‌کنیم. بنابراین برای الگوهای یکنواخت، کد ۰ تا ۸ می‌توانیم داشته باشیم و کد ۹ را به الگوهای غیر یکنواخت نسبت می‌دهیم. بنابر توضیحات داده شده، برای هیستوگرام تصویر چرخش داده شده داریم:

$$0 = (00000000)_2 \Rightarrow 0$$
$$34 = (00100010)_2 \Rightarrow 9$$
$$143 = (10001111)_2 \Rightarrow 5$$

$247 = (11110111)_2 \Rightarrow 7$

هیستوگرام مستقل از چرخش و یکنواخت تصویر چرخش یافته، مطابق شکل زیر است.



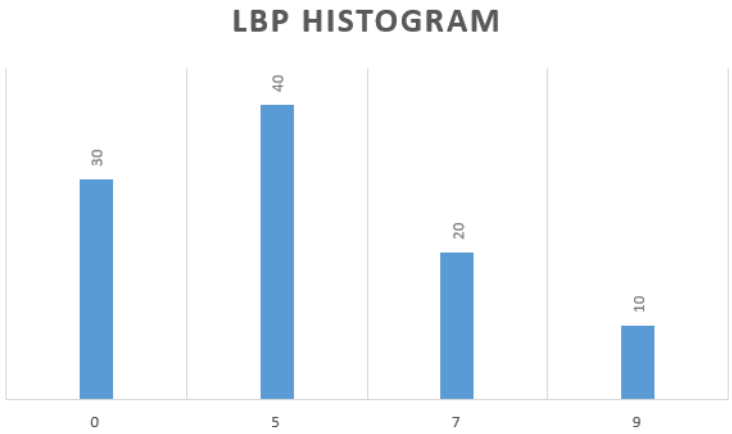
حال، برای تصویر اصلی این کدها را به دست آورده و هیستوگرام آن را رسم می‌کنیم.

$(00000000)_2 \Rightarrow 0$

$(10001000)_2 \Rightarrow 9$

$(11100011)_2 \Rightarrow 5$

$(11111101)_2 \Rightarrow 7$



همانطور که مشخص است، هیستوگرام مستقل از چرخش، برای هر دو حالت تصویر چرخش یافته

و تصویر اصلی یکسان است. علت این موضوع است که در چرخش، الگوها و تعداد یک‌ها تغییری نمی‌کنند و فقط جابه‌جا می‌شوند و همانطور که از اسمش پیداست، مستقل از چرخش است و کدهای یکسانی به اعداد هیستوگرام اولیه و ثانویه نسبت داده شد.

سوال ۲:

در نوت بوک HW[10].ipynb پیوست شده، بخش‌های خواسته شده در زیر را تکمیل کنید. در این سوال می‌خواهیم با استفاده از ویژگی‌های مطالعه شده در درس، اقدام به دسته‌بندی تصاویر به دودسته کشتی و هواپیما بکنیم.

الف) در بخشهای ۱ تا ۳ به ترتیب توابع مربوط به ویژگی فشردگی، گریز از مرکز و چگالی هر تصویر را تکمیل کنید. ورودی هر یک از این توابع تصویر سه کاناله میباشد که ابتدا باید تصویر به تک کاناله تبدیل شود سپس با روش‌های خوانده‌شده در درس باینری شوند (برای نمونه از هرکدام از روش‌های otsu، adaptiveThreshold می‌توانید استفاده کنید) سپس با استفاده از تابع cv2.findConturs اقدام به استخراج شی موردنظر در تصویر نموده و هرکدام از ویژگی‌های مربوطه را متناسب با تابع آن به دست آورید. (۳۰ نمره)

ب) در بخش ۴ در تابع LBP برای تصویر ورودی باید هیستوگرام LBP به دست آورده شود به همین منظور فقط تصویر ورودی را به تصویر تک کاناله تبدیل کنید و با استفاده از تابع pattern_binary_local.feature موجود در کتابخانه skimage هیستوگرام تصویر را به دست آورید و به عنوان خروجی این تابع بازگردانید.(۱۰ نمره)

ج) بخش ۵ مربوط به آماده‌سازی مجموعه داده می‌باشد که تکمیل شده است در این بخش ابتدا تصاویر خوانده شده و به دو بخش آموزش و تست تقسیم می‌شوند با استفاده از تصاویر آموزشی دسته‌بند نحوه تفکیک تصاویر به دودسته کشتی و هواپیما را می‌آموزد و با تصاویر تست دقت دسته‌بند را بر روی تصاویر مشاهده نشده در بخش آموزش می‌سنجیم تا اطمینان حاصل کنیم دسته‌بند تصاویر را همراه با برچسب‌هایشان حفظ نکرده است و با استفاده از ویژگی‌های استخراج شده از تصاویر، ساختار و ویژگی هر دسته را یاد گرفته است. (این بخش نیاز به تغییر نمی‌باشد)

د) در بخش ۶ ابتدا تابع get_featureMatrix را تکمیل کنید به طوری که ورودی تابع مجموعه تصاویر و خروجی آن ماتریس ویژگی‌های استخراج شده باشد همانند ماتریس قرار داده شده در جدول

۱ باشد. سپس یک دسته‌بند مشخص کنید تا با استفاده از ویژگی‌های استخراج شده و برچسب تصاویر نحوه تفکیک تصاویر به دودسته کشتی و هواپیما را بیاموزد. در این بخش از دسته‌بند ساده همچون svm که یک خط تفکیک می‌آموزد استفاده خواهد شد که ورودی این دستهبند ویژگی‌های استخراج شده و برچسب‌ها می‌باشند برای آشنایی با این دسته‌بند و نحوه استفاده از آن می‌توانید از لینک کمک بگیرید. (لزومی به تسلط کامل به نحوه عملکرد این دسته‌بند نیست می‌توانید دسته‌بند را مانند جعبه سیاهی در نظر بگیرید که ویژگی‌های استخراج شده و برچسب تصاویر مربوطه را به عنوان ورودی می‌گیرد و درون جعبه سیاه یک سری پارامترهایی یاد می‌گیرد که بعدا با داشتن ویژگی‌های هر تصویر برچسب آن را پیش‌بینی کند) (۲۵ نمره)

هر مقدار بازگردانده شده از هیستوگرام LBP یک ستون از ستون های ویژگی را تشکیل خواهد داد

هر سطر یک ویژگی های یک تصویر را نشان می دهد

compactness	eccentricity	solidity		

جدول ۱ - شکل ماتریس ویژگی استخراج شده

ح) بخش ۷ را به نحوی تکمیل کنید که عملکرد دسته‌بند آموزش دیده را بر روی تصاویر تست بسنجد . بدین منظور ویژگی‌های لازم از تصاویر تست با استفاده از تابع get_featureMatrix استخراج شده است و باید با دسته‌بند آموزش دیده برای این تصاویر برچسب پیش‌بینی شود و برچسب‌های اصلی تصاویر با برچسب پیش‌بینی شده مقایسه می‌گردد و دقت لازم گزارش می‌شود. برای این بخش می‌توانید از تابع accuracy_score استفاده کنید. (۱۵ نمره)

خ) در بخش ۸ عملکرد دسته‌بند را بر روی یکی از تصاویر تست مشاهده می‌کنید. (در این بخش نیاز به تغییر کد نیست فقط نتیجه را مشاهده کنید)

پاسخ ۲:

الف) ابتدا با استفاده از تابع زیر، تصویر را باینری کرده و با تابع findcountours شکل را استخراج می‌کنیم و بزرگ‌ترین شکل استخراج شده را به عنوان خروجی برمی‌گردانیم. در تابع‌های compatness، eccenticity و solidity از این تابع برای استخراج شکل استفاده می‌کنیم.

```
def contours(image):

  gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
  blur = cv2.GaussianBlur(gray, (5, 5), 0)
  tresh = cv2.threshold(blur, 100, 255, cv2.THRESH_BINARY)[1]

  cnts = cv2.findContours(tresh, 1, 1)
  cnts = cnts[0] if len(cnts) == 2 else cnts[1]
  cnt = max(cnts, key=cv2.contourArea)

  return cnt
```

سپس با استفاده از رابطه زیر، فشردگی (compactness) شکل را محاسبه می‌کنیم.

$$compactness = \frac{4\pi\, Area}{Perimeter^2}$$

```
def compatness(image):

  cnt = contours(image)

  area = cv2.contourArea(cnt)
  perimeter = cv2.arcLength(cnt, True)

  compactness_score = (4 * np.pi * area) / pow(perimeter, 2)

  return compactness_score
```

با استفاده از فرمول زیر نیز کشیدگی (eccentricity) تصویر را به دست می‌آوریم.

$$eccentricity = \frac{MinorAxisLength}{MajorAxisLength}$$

```python
def eccenticity(image):

    cnt = contours(image)

    ellipse = cv2.fitEllipse(cnt)
    (xc,yc),(d1,d2),angle = ellipse

    major = d2 / 2
    minor = d1 / 2

    eccentricity_score = np.sqrt(1 - pow(minor/major, 2))

    return eccentricity_score
```

همچنین، از رابطه زیر برای محاسبه میزان چگال بودن (solidity) استفاده می‌شود.

$$solidity = \frac{Area}{ConvexArea}$$

```python
def solidity(image):

    cnt = contours(image)

    area = cv2.contourArea(cnt)

    convex = cv2.convexHull(cnt)
    hull = cv2.contourArea(convex)

    solidity_score = float(area) / hull

    return  solidity_score
```

ب) برای به دست آوردن هیستوگرام LBP تصویر، از تابع زیر استفاده می‌کنیم. در اینجا ابتدا تصویر را به حالت یک کاناله در آورده و از تابع blur، برای حذف نویزهای احتمالی استفاده می‌کنیم. سپس با استفاده از تابع local_binary_pattern، مقدار lbp را به دست آورده و هیستوگرام آن را محاسبه می‌کنیم. در انتها هیستوگرام را نرمالیزه کرده و آن را به عنوان خروجی return می‌کنیم.

```python
def histogram_of_LBP(image, numPoints, radius, eps=1e-7):

  gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
  blur = cv2.GaussianBlur(gray, (5, 5), 0)

  hist_values = local_binary_pattern(blur, numPoints, radius, method="uniform")
  (hist, _) = np.histogram(hist_values, density=True, bins=256, range=(0, 256))
  hist = hist.astype("float")
  hist /= (hist.sum() + eps)

  return hist
```

ج) در این بخش کاری توسط ما انجام نشد و فقط run کردیم.

د) در اینجا، ماتریس ویژگی‌ها را ساختیم. به این صورت که ابتدا برای هر تصویر، خروجی توابع compatness، eccenticity، solidity و histogram_of_LBP را به دست آورده و آن‌ها را به ماتریس ویژگی‌هایی که در ابتدای تابع تعریف کردیم، اضافه می‌کنیم. در انتها نیز feature_matrix که شامل ویژگی‌های همه تصاویر است را به عنوان خروجی برمی‌گردانیم. کدی که برای این بخش زده شد، در تصویر زیر نمایش داده شده است.

```python
def get_featureMatrix(data):

  feature_matrix = []
  for d in data:
    com = compatness(d)
    ecc = eccenticity(d)
    sol = solidity(d)
    lbp = histogram_of_LBP(d, 8, 1, eps=1e-7)[0]
    feature_matrix.append([com, ecc, sol, lbp])

  return feature_matrix
```

در ادامه، با استفاده از به دست آوردن ماتریس ویژگی‌ها برای داده‌های train، مدل svm خود را ساخته و آن را روی داده‌های آموزشی، train می‌کنیم.

```python
[28] # model 1
feature_matrix_train = get_featureMatrix(x_train)
feature_matrix_train = np.array(feature_matrix_train)
y_train = np.array(y_train)
#determine classifier and train
model = LinearSVC()
model.fit(feature_matrix_train, y_train)
```

ح) در این بخش، بردار ویژگی را برای داده‌های test به دست آورده و خروجی مدل را به ازای داده‌های تست، در y_predict می‌ریزیم. سپس با مقایسه خروجی پیش‌بینی شده و مقدار واقعی label، دقت (accuracy) را به دست می‌آوریم.

```
#test on test dataset
feature_matrix_test = np.array(get_featureMatrix(x_test))
y_test = np.array(y_test)
y_pred = model.predict(feature_matrix_test)
acc = accuracy_score(y_test, y_pred)
acc
```

```
0.703125
```

خ) با استفاده از کد زیر، نتیجه مدل را برای یک داده تصادفی می‌بینیم.

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

همانطور که در تصویر زیر نیز مشخص است، داده متعلق به کلاس ۱ بود که مدل هم همان را پیش‌بینی کرد.