بینایی کامپیوتر

نیم‌سال دوم ۱۴۰۰-۱۴۰۱

مدرس: دکتر مصطفی کمالی

دانشکده علوم ریاضی

تمرین سری چهارم - آشکارسازی چهره

نام و نام‌خانوادگی: نیما کلیدری

شماره دانشجویی: ۹۸۱۰۸۱۲۴

در این سری تمرین از هیچکس مشورت گرفته نشده است. کد نهایی پاسخ به عنوان FaceDetector.py ذخیره شده است.

برای حل این مسئله، در کد سوال ابتدا تابع اصلی تعریف شد که در آن، موقعیت نسبی فولدر های حاوی دیتاست تصاویر و همینطور پارامتر های برخی الگوریتم های مسئله تعریف میشوند. سپس تابعی به نام randomize تعریف شده که با فراخوانی آن، فولدر های Train ، Test و Validation که هر کدام شامل دو فولدر مثبت و منفی هستند، به وجود می‌آیند.سپس ۱۰۰۰۰ تصویر آموزش از فولدر شامل صورت ها و ۱۰۰۰۰ تصویر که تصویر منظره هستند و در آنها صورتی نیست، به تصادف انتخاب شده و درون پوشه های مرتبط ریخته میشوند. سپس ۲۰۰۰ تصویر برای اعتبار سنجی و تست نیز از هر کلاس انتخاب شده و به همین صورت عملیات جابجایی را به پوشه های جدید انجام میدهیم.

حال یک تابع ویژگی hog با مقادیر زیر درست میکنیم:

- ابعاد پنجره: $(20, 20)$
- ابعاد بلوک: $(10, 10)$
- پیشروی بلوک: $(5, 5)$
- ابعاد سلول: $(10, 10)$
- تعداد دسته ها برای گرادیان ها: 12

سپس دقیقا مشابه همان روشی که در سری ۳ اعمال کردیم، تابعی در ادامه میسازیم که وارد فولدر های داده شود، داده های آموزش را انتخاب کرده و بر روی کل تصاویر، تابع hog را اعمال کند و نتیجه را ذخیره کند و همچنین لیبل آن ها را نیز یا ۰ و یا ۱ بسته به صورت داشتن یا نداشتن ذخیره کند. در نهایت داده هایی که داریم و لیبل های آن ها را به تابع svm کتابخانه scikit میدهیم. در این تابع پارامتر های زیر استفاده شده است:

- نوع کرنل: rbf
- C: 1
- gamma: $scale$

پس از آموزش دادن مدل svm که بسیار هم میتواند طول بکشد، نتیجه را در کامپیوتر ذخیره کرده و به عنوان خروجی تابع مذکور نیز برمیگردانیم. و اما ۳ نکته قابل اهمیت در این بخش وجود دارد:

۱. از آنجا که فضای حاشیه اطراف صورت ها بسیار زیاد بود، از هر طرف هر تصویر صورت ۵۰ پیکسل بریده شد و یک تصویر ۱۵۰ در ۱۵۰ پیکسل باقی ماند. از آنجا که دیتاست بدون صورت هم تصویر های ۱۵۰ در ۱۵۰ دارد، دیگر نیاز به تغییر ابعاد وجود ندارد.

۲. در هر تصویر با یک محاسبه، میبینیم که یک بردار حدودا ۲۰ هزار بعدی داریم. اما کتابخانه OpenCv برای پیدا کردن بردار های مشخصه، باگ های زیادی دارد. برای مثال در یک تصویر نامناسب، کمتر از بقیه بردار ها، بعد به ما میدهد. به همین خاطر حدود یک هزارم داده های اینچنینی از داده های مورد بررسی حذف میشود (به دلیل اینکه در ادامه به مشکل بر میخوریم)

۳. پارامتر هایی که در اینجا آمده است، از روی تست روی داده ی اعتبار سنجی است که در بهترین حالت پاسخ 98.5% را به ما میدهد. اما پس از امتحان روی داده تست در مرحله بعد، همین پارامتر ها به ما پاسخ 100% درست دادند.

سپس در تابع اصلی یک خط کد کامنت هست که وقتی برای بار اول مدل آموزش داده شد، خط تابع بالاییش را برداشته و این را قرار میدهیم و ازین به بعد نیازی با آموزش مجدد نیست.

سپس یک تابع FaceDetector تعریف میکنیم که در آن، آدرس پوشه تصاویر تست داده شده و سپس بردار های ویژگی و کلاس های مربوطشان از آن گرفته میشود (همین کار برای داده های آموزش هم شد).سپس در ادامه، مدل و بردار های ویژگی را به تابع predict میدهیم و لیست خروجی را با لیبل هایی که زده بودیم مقایسه میکنیم و به این ترتیب در ابتدا دقت مدل را پیدا میکنیم. سپس تابع decision-function را صدا میزنیم و مقادیر بدست آمده را به همراه لیبل های اصلی، به دو تابع ساخته شده ی ROC و Precision-Recall میدهیم و این دو تابع با استفاده از ورودی های پاسخ مدل و کتابخانه scikit ، منحنی های لازم را رسم میکند. دقت میانگین (AP) را نیز خود این توابع برای داده تست، برابر $AP = 1.00$ بدست آوردند.

حال که نتایج اول و دوم را بدست آوردیم، لازم است تا چهره ها را از تصاویر داده شده پیدا کرده و مشخص کنیم. به این منظور تابعی تعریف می‌کنیم که در آن نام تصویر را گرفته و به عنوان خروجی تصویری که چهره ها روی آن مشخص است را خروجی می‌دهد. نحوه عملکرد این بخش به این صورت است که در ابتدا به سمت چپ و راست و بالا و پایین تصویر ورودی، ۵۰ پیکسل اضافه میکنیم (که چهره های حاشیه ای را تشخیص دهیم). سپس تصویر و مدل svm و ابعاد شروع و پایان پنجره را به یک تابع میدهیم. در این آزمایش ها، ابعاد شروع پنجره ها ۱۰۰ و ابعاد پایان ۲۱۰ و همچنین گام افزایش ۱۵ داشتند.

حال گام پیشروی پنجره را، برابر با یک دوازدهم ابعاد پنجره قرار میدهیم. پس از ایجاد ۲ تا حلقه برای پیشروی افقی و عمودی پنجره به گام مشخص، بردار ویژگی هر پنجره را از دادن پنجره که به ابعاد ۱۵۰ در ۱۵۰ تغییر اندازه داده شده به تابع hog بدست می‌آوریم، و سپس صحت سنجی ابعاد بردار ویژگی را انجام داده و بردار را به تابع decision-function میدهیم. این تابع به ما یک عدد میدهد که میتواند مثبت یا منفی باشد. در این مسئله در صورت مثبت بودن آن را چهره تشخیص میدهد که این اصلا برای ما مطلوب نیست و پاسخ غلط زیادی بدست می‌آید. بنابر این یک مقدار آستانه (که در این مسئله برابر با ۱.۱ است) تعریف میکنیم و اگر خروجی تابع از این مقدار بزرگتر بود، مرکز مربع به دست آمده، ابعاد آن و پاسخ بدست آمده را در یک لیست ذخیره میکنیم. حال در ادامه خوشه بندی به روش ساده میکنیم. (این روش از تمرین سری ۲ پردازش تصویر سوال ۲ گرفته شده است و تغییراتی بر روی آن اعمال شده است.)

روش کار به این صورت است که هر دو مرکزی که فاصله ی کمتر از 1.45 برابر ابعاد میانگین دسته را داشته باشند، در یک دسته قرار می‌گیرند. (روش های مختلف خوشه بندی در اینجا به دلیل معایبی که هر کدام دارد، به خوبی پیاده سازی نمی‌شوند.) حال در پایان که دسته ها بدست آمد، ابعاد هر مربع نهایی برابر ابعاد میانگین دسته و مرکز مربع، مرکز میانگین دسته است. ضریب بدست آمده برای هر دسته هم حداکثر ضریب حاصل شده برای آن دسته هستند. در نهایت نیز با استفاده از تقسیم ضریب هر دسته بر حداکثر مقدار ضریب یک دسته در 1.05 و گرفتن ریشه دوم آن و سپس ضرب آن در ۱۰۰ ، یک درصد بدست می‌آید که نشان دهنده ی خوبی از ضریب اطمینان برای آن دسته (چهره پیدا شده) است. پس از رسم این مربع ها و نمایش درصد روی آنها، خروجی نهایی که ۵۰ پیکسل از هر طرف آن قطع میکنیم ، به عنوان خروجی تابع بازگردانده میشود و تصویر ذخیره شده و کد به انتها میرسد.