عنوان: OpenCVSharp #16
نویسنده: وحید نصیری
تاریخ: ۱۳۹۴/۰۳/۳۱ ۱۴:۱۰
آدرس: www.dotnettips.info
گروه‌ها: OpenCV

در قسمت قبل با نحوه‌ی استفاده از یک trained data از پیش آماده شده‌ی تشخیص چهره، آشنا شدیم. در این قسمت قصد داریم با نحوه‌ی تولید این فایل‌های XML آشنا شویم و یک تشخیص دهنده‌ی سفارشی را طراحی کنیم.

طراحی classifier سفارشی تشخیص خودروها

برای طراحی یک تشخیص دهنده‌ی سفارشی مبتنی بر الگوریتم‌های Machine learning، نیاز به تعداد زیادی تصویر داریم. در اینجا از بانک تصاویر خودروهای « UIUC Image Database for Car Detection » استفاده خواهیم کرد. در این بسته، یک سری تصویر positive و negative را می‌توان ملاحظه کرد. تصاویر مثبت، تصاویر انواع و اقسام خودروها هستند (550 عدد) و تصاویر منفی، تصاویر غیر خودرویی (500 عدد)؛ یا به عبارتی، هر تصویری، منهای تصاویر خودرو می‌تواند تصویر منفی باشد.

ایجاد فایل برداری از تصاویر خودروها

در ادامه یک فایل متنی را به نام carImages.txt ایجاد می‌کنیم. هر سطر این فایل چنین فرمتی را خواهد داشت:

```
pos/pos-177.pgm 1 0 0 100 40
```

ابتدا مسیر تصویر مشخص می‌شود. سپس عدد 1 به این معنا است که در این تصویر فقط یک عدد خودرو وجود دارد. 4 عدد بعدی، ابعاد مستطیلی تصویر هستند.
در ادامه فایل متنی دیگری را به نام negativeImages.txt جهت درج اطلاعات تصاویر منفی، ایجاد می‌کنیم. اینبار هر سطر این فایل تنها حاوی مسیر تصویر مدنظر است:

```
neg/neg-274.pgm
```

این دو فایل را می‌توان با استفاده از دو متد ذیل، به سرعت تولید کرد:

```
private static void createCarImagesFile()
{
    var sb = new StringBuilder();
    foreach (var file in new DirectoryInfo(@"..\..\CarData\CarData\TrainImages").GetFiles("*pos-
*.pgm"))
    {
        sb.AppendFormat("{0} {1} {2} {3} {4} {5}{6}", file.FullName, 1, 0, 0, 100, 40,
Environment.NewLine);
    }
    File.WriteAllText(@"..\..\CarsInfo\carImages.txt", sb.ToString());
}

private static void createNegativeImagesFile()
{
    var sb = new StringBuilder();
    foreach (var file in new DirectoryInfo(@"..\..\CarData\CarData\TrainImages").GetFiles("*neg-
*.pgm"))
    {
        sb.AppendFormat("{0}{1}", file.FullName,Environment.NewLine);
    }
    File.WriteAllText(@"..\..\CarsInfo\negativeImages.txt", sb.ToString());
}
```

برای کامپایل اطلاعات فایل‌های تولید شده، نیاز به فایل opencv_createsamples.exe است. این فایل را در پوشه‌ی opencv\build\x86\vc12\bin بسته‌ی اصلی OpenCV می‌توانید پیدا کنید.

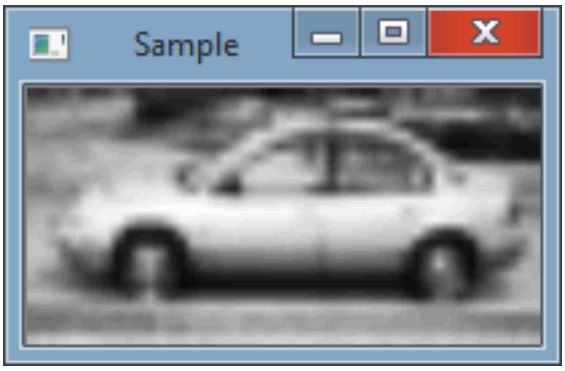
```
opencv_createsamples.exe -info carImages.txt -num 550 -w 48 -h 24 -vec cars.vec
```

پارامترهای این دستور شامل سوئیچ info است؛ به معنای مشخص سازی فایل اطلاعات تصاویر مثبت. سوئیچ num تعداد تصاویر آن‌را تعیین می‌کند و سوئیچ‌های w و h، طول و عرض تصاویر هستند. سوئیچ vec نیز جهت تولید یک فایل vector از این اطلاعات بکار می‌رود.
پس از اجرای این دستور، فایل cars.vec تولید خواهد شد؛ با این خروجی:

```
Info file name: carImages.txt
Img file name: (NULL)
Vec file name: cars.vec
BG  file name: (NULL)
Num: 550
BG color: 0
BG threshold: 80
Invert: FALSE
Max intensity deviation: 40
Max x angle: 1.1
Max y angle: 1.1
Max z angle: 0.5
Show samples: FALSE
Original image will be scaled to:
  Width: $backgroundWidth / 48
  Height: $backgroundHeight / 24
Create training samples from images collection...
Done. Created 550 samples
```

اگر علاقمند هستید که محتویات فایل باینری cars.vec را مشاهده کنید، دستور ذیل را صادر نمائید:

```
"c:\opencv\build\x86\vc12\bin\opencv_createsamples.exe" -vec cars.vec -w 48 -h 24
```



در این پنجره‌ی باز شده، تصاویر بعدی و قبلی را می‌توان با دکمه‌های arrow صفحه کلید، مشاهده کرد.

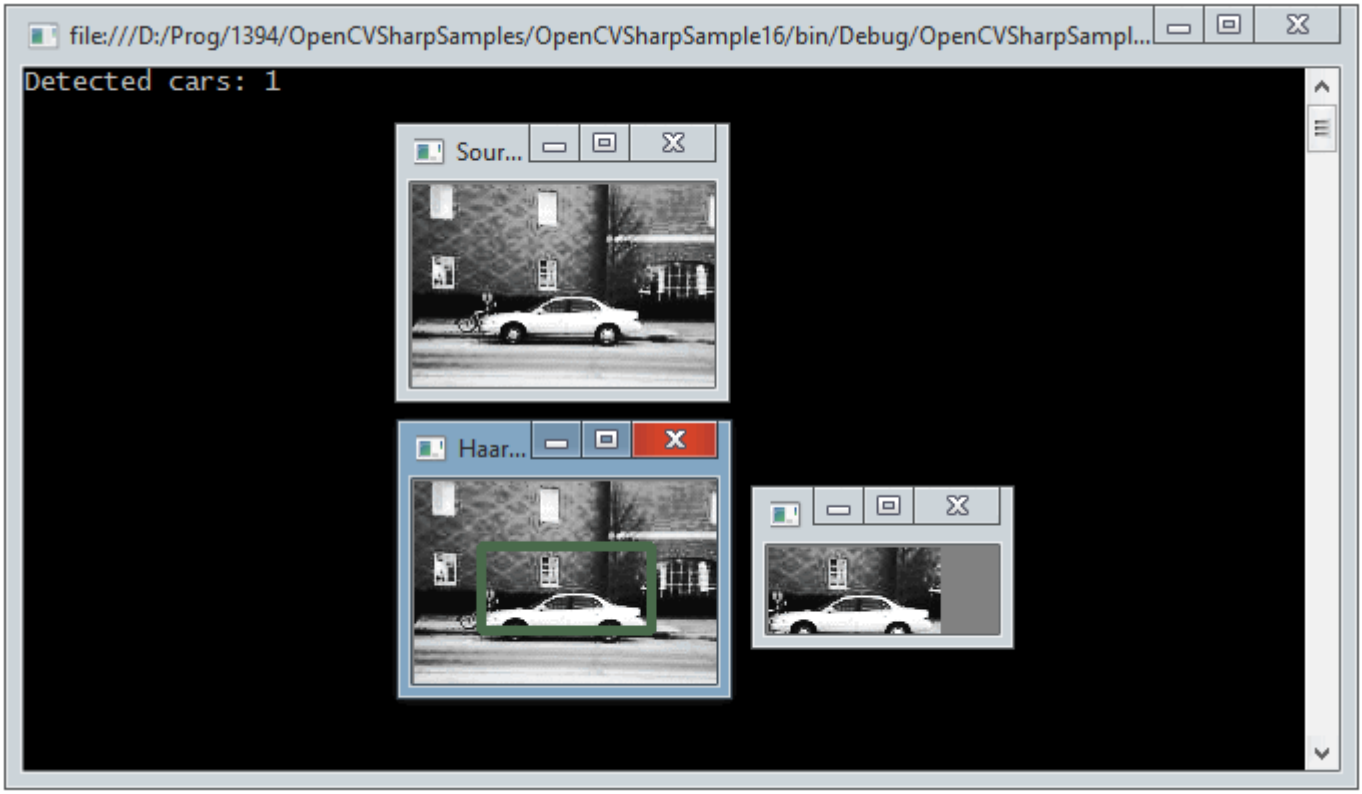
تبدیل فایل برداری تصاویر خودروها به trained data

تا اینجا موفق شدیم بیش از 500 تصویر خودرو را تبدیل به یک فایل برداری سازگار با OpenCV کنیم. اکنون نیاز است، این اطلاعات پردازش شده و trained data مخصوص الگوریتم‌های machine learning تولید شود. این‌کار را توسط برنامه‌ی opencv_traincascade.exe انجام خواهیم داد. این فایل نیز در پوشه‌ی opencv\build\x86\vc12\bin بسته‌ی اصلی OpenCV موجود است.
دستور ذیل در پوشه‌ی data، بر اساس اطلاعات برداری cars.vec و همچنین تصاویر منفی مشخص شده‌ی در فایل negativeImages.txt، با تعداد هر کدام 500 عدد (این عدد را توصیه شده‌است که اندکی کمتر از تعداد max موجود مشخص کنیم) و تعداد مراحل 2 (هر چقدر این تعداد مراحل بیشتر باشد، فایل نهایی تولید شده دقت بالاتری خواهد داشت؛ اما تولید آن به زمان

بیشتری نیاز دارد) اجرا می‌شود. در اینجا featureType به LBP یا Local binary Pattern، تنظیم شده‌است. این الگوریتم از Haar cascade سریعتر است.

```
"E:\opencv\bin\opencv_traincascade.exe" -data data -vec cars.vec -bg negativeImages.txt -numPos 500 -numNeg 500 -numStages 2 -w 48 -h 24 -featureType LBP
```

خروجی اجرای این دستور را می‌توانید در پوشه‌ی data با نام cascade.xml، پیدا کنید. پس از آن، روش استفاده‌ی از این فایل، با مطلب تشخیص چهره تفاوتی ندارد.



کدهای کامل این مثال را از اینجا می‌توانید دریافت کنید.