

گزارش پروژه کارشناسی

نام و نام خانوادگی :حسین حیاتی

شماره دانشجویی: 9233002

استاد راهنما : دکتر یزدی

استاد ناظر : دکتر ستوده

فهرست:

[مقدمه: 2](#_Toc530469272)

[تشخیص و حذف چهره و نمایش پس زمینه : 3](#_Toc530469273)

[توضیحات بیشتر در مورد dlib facial landmark detector : 4](#_Toc530469274)

[پیدا کردن facial landmarks در ویدیو و بررسی قسمتی از کد ها: 4](#_Toc530469275)

[منابع: 10](#_Toc530469276)

# مقدمه:

شناسایی چهره(face detection) یک تکنولوژی کامپیوتری است که قابلیت شناسایی کردن چهره افراد از روی تصاویر دیچیتالی را داراست. برای این منظور ، برنامه های شناسایی چهره از الگوریتم های ماشین لرنینگ برای شناسایی چهره در تصاویر مختلف استفاده میکنند. که این تصاویر ممکن است شامل اشیا دیگری بغیر از تصویر صورت انسان باشد.

شناسایی چهره (face detection) یک مفهوم گسترده تری از تشخیص چهره (face recognition) می باشد. شناسایی چهره تنها به این معنا خواهد بود که سیستم قادر به تشخیص و پیدا کردن چهره یک انسان در عکس یا فیلم می باشد. شناسایی چهره کاربرد های متعددی دارد که تنها یکی از آن ها تشخیص چهره می باشد. شناسایی چهره میتواند برای دوربین های قابلیت auto focus کاربرد داشته باشد ، همچنین میتواند برای شمارش تعداد افرادی که وارد یک مکان شده اند استفاده شود ، و حتی برای اهداف بازاریابی میتواند مورد استفاده قرار گیرد .برای مثال تبلیغات میتوانند زمانی نشان داده شوند که چهره شخص نمایان شده باشد .

در این جا برای تشخیص چهره همان طور که در ادامه بحث خواهیم کرد ، ابتدا چهره را در تصویر پیدا خواهیم کرد و سپس اعضای اساسی چهره را تشخیص میدهیم و با استفاده از آن ها عملات مورد نظر را میتوانیم روی صورت انجام دهیم ، برای مثال پس از تشخیص قسمت های مختلف صورت میتوانیم استیکر های مختلفی را روی قسمت های مختلف صورت اعمال کنیم ( همانطور که در گوشی های هوشمند وجود دارد ) ، اما در واقع هدف این پروژه این است که پس از تشخیص چهره ، صورت شخص را حذف کنیم و تصویر پس زمینه را که از قبل برای آن تعریف کرده ایم یا حین اجرای پروژه گرفته ایم نشان دهد . ( درواقع این طور به نظر میرست که صورت شخص نامرئی شده و پشت سر وی نشان داده شده است )

# تشخیص و حذف چهره و نمایش پس زمینه :

برای تشخیص چهره ابتدا facial landmarks را باید شناسایی کنیم .

شناسایی facial landmarks زیر مجموعه ای از shape prediction می باشد . در واقع با این روش ، هدف ما این است که قسمت های اساسی صورت را پیدا کنیم .

قسمت های مهم صورت عبارتند از : چشم ها ، ابرو ، بینی ، دهان و چانه . که در این پروژه به دلایلی که بعدا ذکر خواهد شد تنها به تشخیص ابرو و چانه نیاز خواهیم داشت .

برای تشخیص facial landmarks دو مرحله وجود دارد :

1. تشخیص صورت در تصویر
2. تشخیص اعضای مهم صورت

مرحله اول ( تشخیص چهره در تصویر ) با روش های مختلفی ممکن است پیاده سازی شود ، ولی مهم این است که در نهایت محلی را که صورت در آن قرار گرفته را به ما بدهد تا ما بتوانیم مرحله دوم را اجرا نماییم .

شناسایی facial landmarks که در کتابخانه dlib وجود دارد و ما میخواهیم از آن استفاده کنیم در واقع یک پیاده سازی از مقاله One millisecond face alignment with an ensemble of regression trees می باشد که در ضمیمه اضافه خواهم کرد.

این روش با استفاده از موارد زیر شروع میشود :

1. یک training set شامل facial landmark های لیبل خورده روی یک عکس ، که این عکس ها به صورت دستی لیبل خورده اند و مختصات نقاطی که هر یک از اعضای صورت را نشان میدهند را مشخص میکند.
2. Priors ، به طور مشخص تر ، probability of distance بین پیکسل های ورودی

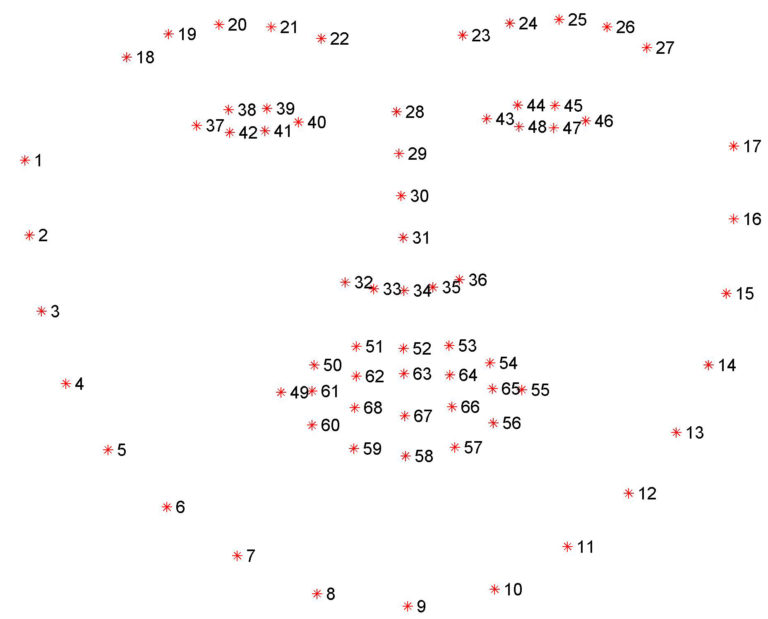
با داشتن این training data ، یک گروه از regression trees ، train میشوند تا موقعیت facial landmarks را به صورت مستقیم از شدت پیکسل ها تخمین بزنند.

که نتیجه نهایی یک facial landmark detector میشود که میتواند facial landmark ها را به صورت real-time تشخیص دهد.

# توضیحات بیشتر در مورد dlib facial landmark detector :

یک شناساگر facial landmark از قبل train شده در کتابخانه dlib برای تخمین مختصات 68 نقطه که روی اعضای مهم صورت map شده اند مورد استفاده قرار میگیرد . البته شناساگر های دیگری نیز وجود دارند که میتواند این کار را برای ما انجام دهند ، برای مثال در HELEN dataset برای تخمین مختصات 198 نقطه روی که روی اعضای مهم صورت map شده اند مورد استفاده قرار میگیرد . در هر صورت در این قصد دارم برای کم کردن حجم دیتاست از کتابخانه dlib استفاده کنم.

به صورت شکل زیر:



# پیدا کردن facial landmarks در ویدیو و بررسی قسمتی از کد ها:

ابتدا پکیج هایی را که میخواهیم با آنها کار کنیم import میکنیم :



حال میخواهیم یک دستور برای command line اضافه کنیم که بتوانیم فایل دیتا ها را از طریق آن به کد معرفی کنیم.



که این نشان میدهد که بعد از python project\_name.py اگر مثلا -p data.dat را بنویسیم میتوانیم فایل دیتا را در کد لود کنیم .

حال باید الگوریتم  HOG + Linear SVM-based face detectorرا آماده سازی کنیم .(HOG مخفف Histogram of Oriented Gradients میباشد)( این الگوریتم برگرفته از مقاله Histograms of Oriented Gradients for Human Detection Navneet Dalal and Bill Triggs میباشد که در ضمیمه اضافه خواهد شد) ، که عملا مراحل زیر را انجام میدهد:

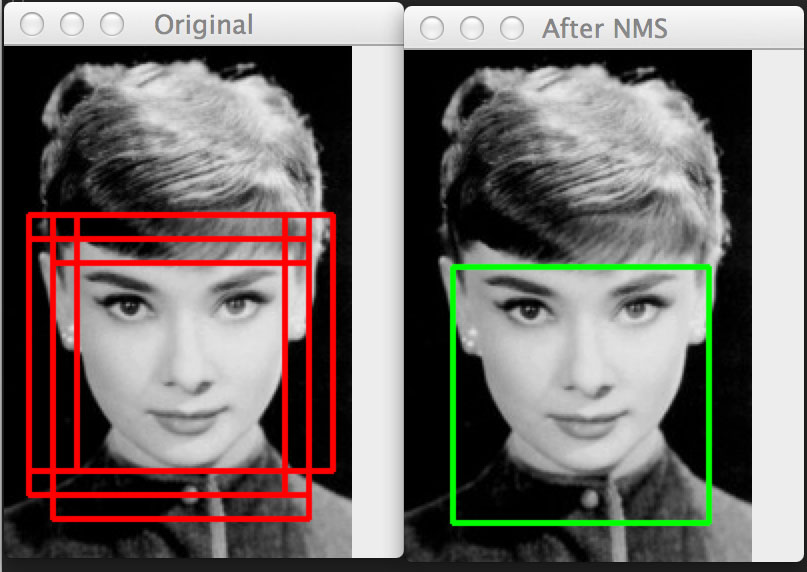
1. از P سمپل درست از شی هایی که میخواهیم detect کنیم و hog descriptor را از آنها استخراج کنیم ، از training data نمونه برداری کنیم
2. از n سمپل که شامل هیج کدام از شی هایی که میخواهیم از ان ها نمونه برداری کنیم نیز نمونه برداری کنیم .
3. Linear Support Vector Machine را برای این دو دسته از نمونه ها train کنیم
4. hard-negative mining را برای همه عکس ها با همه اسکیل های از آن عکس ها اعمال کنیم .

برای این منظور برای همه عکس ها با همه اسکیل ها یک ویندوز را روی عکس جا به جا میکنیم و HOG descriptors را محاسبه کرده و کلاسیفایر را اعمال میکنیم.

سمپل های false-positive که حین مرحله قبل به دست آمده اند را با درجه اطمینانشان مرتب میکنیم و دوباره کلاسیفایر را با نمونه های hard-negative ترین میکنیم .

1. حال کلاسیفایر train شده و میتواند روی دیتاست تستی شما اعمال شود.

حال دوباره مانند مرحله 4 برای هر عکس از دیتا ست تستی و برای هر اسکیل از این عکس ها sliding window technique را اعمال میکنیم. در هر ویندوز توصیفگر های HOG را اعمال کرده و کلاسیفایر را اعمال کنیم.اگر کلاسیفایر یک ویندوز را با احتمال به اندازه کافی زیاد پیدا کرد ، bounding box آن را ذخیره میکنیم . بعد از پایان اسکن کردن عکس ، برای حذف قسمت های زاید و اضافه عکس non-maximum suppression را اعمال میکنیم.



برگردیم به آماده سازی الگوریتم  HOG + Linear SVM-based face detector:



حال میخواهیم VideoStream را آماده سازی کنیم و سنسور های دوربین را اجازه دهیم آماده به کار شوند:



حال یک فریم را میخوانیم ، و از سایز آن استفاده میکنیم تا یک تصویر پس زمینه سفید به صورت موقت ایجاد کنیم :



سپس یک while بینهایت ایجاد میکنیم تا تصویر را به صورت دائم بتوانیم پردازش کنیم و دو شرط تعریف میکنیم که یکی با زدن space عکس بگیرد و به عنوان عکس پس زمینه ذخیره کند و دیگری اینکه با زدن دکمه q برنامه را ببندد.



سپس سایز تصویر بک گراند را که گرفته ام تغییر میدهیم ، فریم ها را به grayscale برده و چهره را در آن مقیاس تشخیص می دهیم.

حال که چهره را تشخیص داده ایم زمان آن رسیده که برای هر چهره تشخیص داده شده facial landmarks را پیدا کنیم و درصورتی که وجود نداشت ، پیام خطا میدهیم که چهره ای یافت نشد .



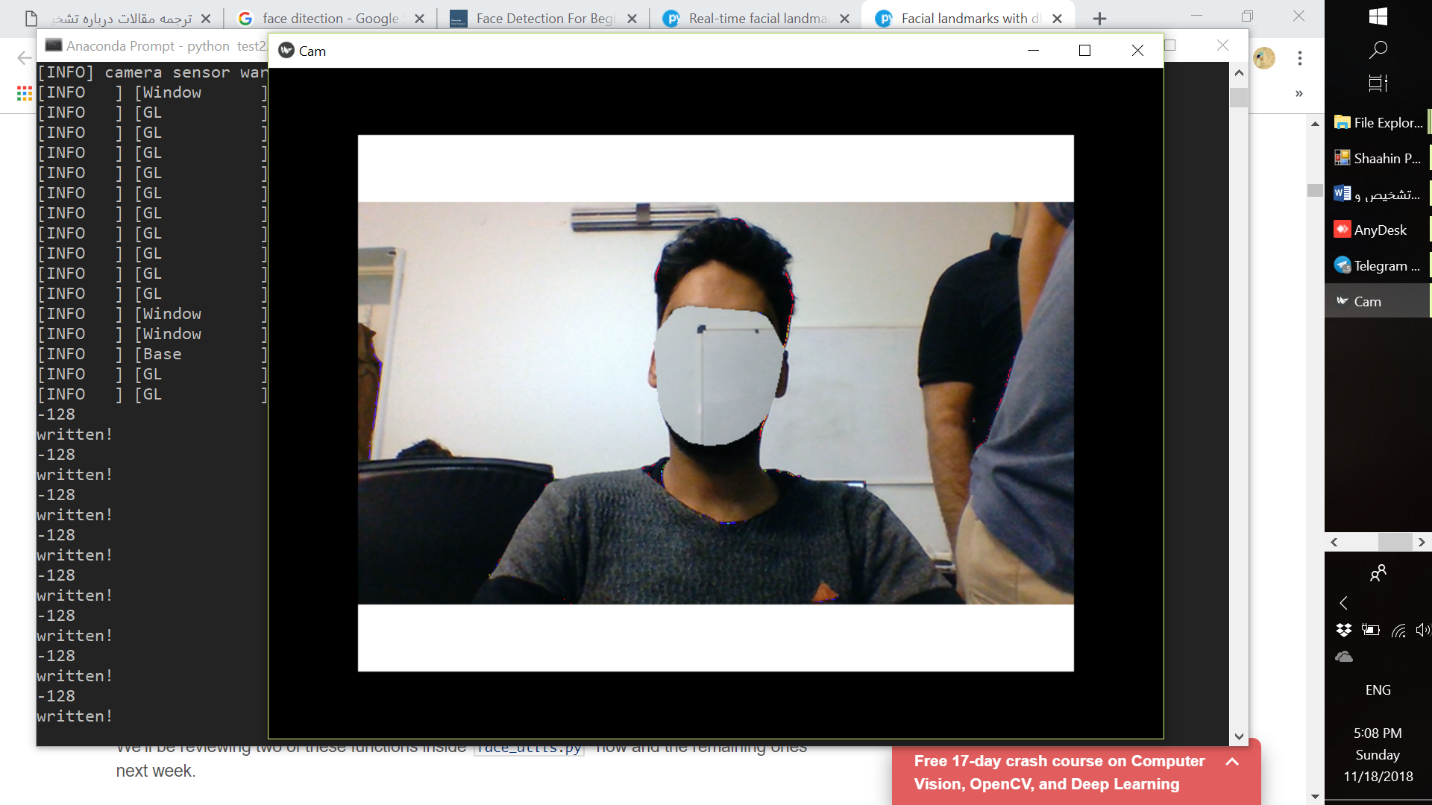
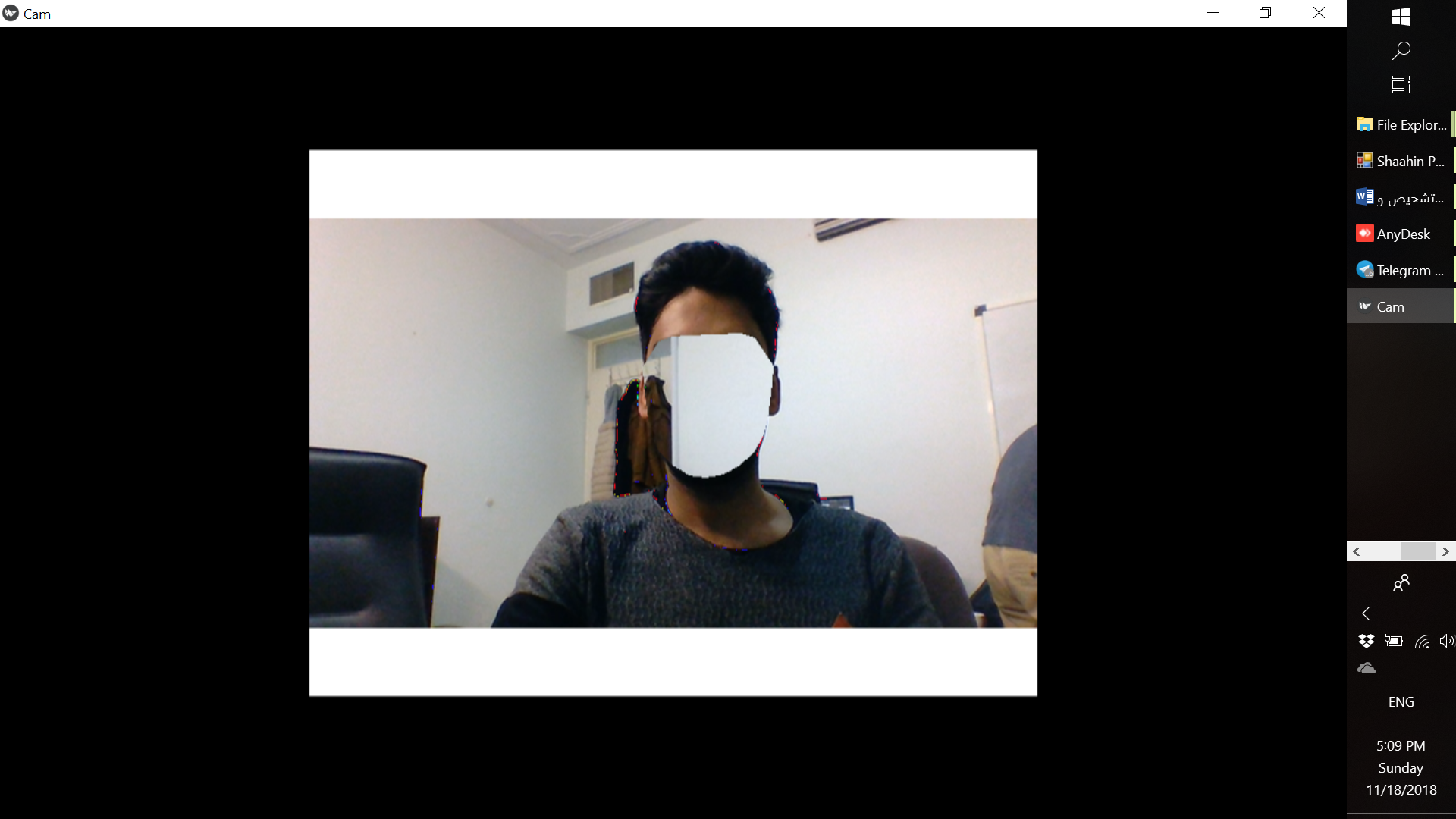
حال با استفاده از دیتاست لود شده ، و با استفاده از نقاط مشخص شده در صورت ، تمام صورت را که میتوان گفت بین دو ابرو و چانه است را باید مشخص کینم ، و سپس نقاط مشخص شده را با خط به هم وصل میکنیم   
داخل صورت را با ماسک پر میکنیم .



سپس تصویر پیش زمینه که همان ویدیو هست و تصویر پس زمینه که عکس آن را از قبل گرفته بودیم و خود ماسک به تنهایی را به به تابع cmb که از قبل ساخته ایم میدهیم ، و به این صورت تصویری که در نهایت به ما میدهد صورت حذف شده خواهد بود که پس زمینه را میتوان در آن سوی چهره مشاهده کرد .



و در نهایت نمونه هایی از برنامه در حال اجرا را قرار میدهم :

# منابع:

1. One Millisecond Face Alignment with an Ensemble of Regression Trees paper by Kazemi and Sullivan (2014).
2. Histograms of Oriented Gradients for Human Detection Navneet Dalal and Bill Triggs(2004)
3. http://dlib.net
4. stackoverflow.com
5. youtube.com
6. https://www.pyimagesearch.com/2014/11/10/histogram-oriented-gradients-object-detection/