تشخیص چهره با SVD

تا به اینجا، کاربردهای زیادی از تجزیه مقادیر منفرد(SVD) دیدیم.

اما شاید برایتان جالب باشد که این بحث بسیار مهم در جبرخطی، حتی در تشخیص چهره نیز کاربر دارد.

در این پروژه به کمک دانشی که از جبرخطی کسب کرده‌اید، با استفاده از تجزیه SVD می‌خواهیم به تشخیص چهره های مشابه بر روی یک دیتاست بپردازیم.

امیدواریم از انجام این پروژه به شدت زیبا، از جبرخطی لذت ببرید.

منتظر پروژه‌های قشنگتون هستیم. :)

فهرست مطالب

مقدمه

در این پروژه قرار است سیستمی طراحی شود تا چهره افراد مشابه را تشخیص دهد. به عبارتی بتواند با توجه به تصاویری که از چهره افراد مختلف دیده و یاد گرفته است، تصویر جدیدی که مربوط به یکی از افراد قبلی است اما از زاویه مختلف و با حالت چهره متفاوتی گرفته شده است را شناسایی کند.

قرار نیست در این پروژه از مفاهیمی که در یادگیری ماشین مطرح می‌شود استفاده کنیم.
با استفاده از مفهوم و ابزار قدرتمند SVD و نیز مفهوم فضا و زیرفضای برداری، یک سیستم تشخیص چهره ساده درست میکنیم.

فرض کنید یک دیتاست از چهره افراد مختلف داریم.
هر عکس در این دیتاست از m در n (ابعاد تصویر) پیکسل تشکیل می‌شود.
اگر تعدادی از این عکس ها را به عنوان دیتاست آموزش(train) جدا کنیم و سپس هرکدام از این عکس‌ها را تبدیل به برداری به اندازه $m \times n$ تبدیل کنیم، آنگاه مجموعه تمامی این بردار ها تشکیل یک زیر فضا با نام فضای چهره (face space) می‌دهند که زیر فضایی از فضای کل عکس های دیتاست (و نه صرفا دیتاست آموزش) به نام فضای تصویر (image space) است.

حال سوال مهمی که باید از خودتان بپرسید این است که **این فضا چگونه توصیف می‌شود؟ پایه‌های این فضا به صورت چه هستند؟**
فعلا بدانید که SVD به ما کمک میکند تا پایه‌های این فضا را پیدا کنیم. جوابش را در بخش بعدی به طور مفصل توضیح میدهیم.

سوال بعدی!
به نظر شما پایه‌های این فضای برداری چه کمکی به ما در تشخیص چهره افراد می‌کنند؟
اگر یک چهره جدید که در دیتاست آموزش وجود نداشته است را تبدیل به برداری به اندازه $m \times n$ کنیم، آنگاه با تصویر کردن(project) بردار چهره جدید برروی این پایه ها و سپس با مقایسه‌ی تصویر(project)

چهره‌های دیتاست آموزش، میتوانیم بگوییم این چهره جدید بیشترین شباهت را به کدام فرد شناخته شده از قبل(موجود در دیتاست آموزش) دارد.

به عبارتی، اگر دو عکس متفاوت باشند اما متعلق به چهره یک فرد باشند، آنگاه هویت فرد شناخته می‌شود.

برای مقدمه، این توضیحات کافی است.

در بخش های بعد جزئیات بیشتری برایتان آورده شده است.

تشخیص چهره به کمک SVD

در ابتدا، همانطور که در قسمت مقدمه دیدیم، لازم است که ماتریس تصاویرآموزشی را تشکیل دهیم. برای این کار هر یک از تصاویر $M = m \times n$ آموزشی را به صورت یک بردار با ابعاد M تبدیل می کنیم.

$$\boldsymbol{f}_i \in \mathbb{R}^M$$

حال به واسطه این بردار های تصاویر آموزشی، ماتریس مجموعه برداری آموزشی را می سازیم که N تصویر شناخته شده و جدا از هم را در خود دارد.

$$S = [\boldsymbol{f}_1 \quad \boldsymbol{f}_2 \quad \cdots \quad \boldsymbol{f}_N]$$

حال در این مرحله لازم است که میانگین بردار های تصویر موجود در S را به دست آوریم.

$$\bar{\boldsymbol{f}} = \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{f}_i$$

در اینجا که بردار تصویر میانگین را به دست آورده ایم، باید برای هم مرکز کردن تمامی بردار های تصویر، بردار میانگین را از تمامی بردار های تصویر کم کنیم و بردار حاصل را a_i می نامیم.

$$\boldsymbol{a}_i = \boldsymbol{f}_i - \bar{\boldsymbol{f}} \;\; (i = 1, 2, \dots, N)$$

در این مرحله، فرض کنید که $rank\, A = r$ می باشد به گونه ای که $r \leq N \ll M$ می باشد. در این حالت قابل اثبات است که ماتریس A دارای تجزیه مقدار منفرد (SVD) زیر می باشد.

$$A = U\Sigma V^T$$

که همانطور که می دانید، در اینجا ماتریس Σ یک ماتریس قطری با هم اندازه با ماتریس A می باشد ($M \times N$) و به فرم زیر است.

$$\Sigma = \begin{bmatrix} \sigma_1 & 0 & \cdots & 0 & 0 & \cdots & 0 \\ 0 & \sigma_2 & \cdots & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \sigma_r & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & \sigma_{r+1} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & 0 & \cdots & \sigma_N \\ 0 & 0 & \cdots & 0 & 0 & \cdots & 0 \end{bmatrix}$$

به ازای σ_i ها ($i = 1, 2, \dots, N$) به عنوان مقدار منفرد (singular value) ماتریس A شناخته می شود و برای آن ها داریم:

$$\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r > 0$$
$$\sigma_{r+1} = \sigma_{r+2} = \cdots = \sigma_N = 0$$

ماتریس V یک ماتریس $N \times N$ و orthogonal می باشد.

$$V = [\boldsymbol{v}_1 \quad \boldsymbol{v}_2 \quad \cdots \quad \boldsymbol{v}_r \quad \boldsymbol{v}_{r+1} \quad \cdots \quad \boldsymbol{v}_N]$$

به این صورت که ستون های ماتریس V تشکیل یک مجموعه برداری orthonormal را می دهند.

$$\boldsymbol{v}_i \cdot \boldsymbol{v}_j = \boldsymbol{v}_i^T \boldsymbol{v}_j = \begin{cases} 1, & i = j \\ 0, & i \neq j \end{cases}$$

ماتریس U نیز یک ماتریس $M \times M$ و orthogonal می باشد.

$$U = [\boldsymbol{u}_1 \quad \boldsymbol{u}_2 \quad \cdots \quad \boldsymbol{u}_r \quad \boldsymbol{u}_{r+1} \quad \cdots \quad \boldsymbol{u}_M]$$

به این صورت که ستون های ماتریس U نیز تشکیل یک مجموعه برداری orthonormal را می دهند.

$$\boldsymbol{u}_i \cdot \boldsymbol{u}_j = \boldsymbol{u}_i^T \boldsymbol{u}_j = \begin{cases} 1, & i = j \\ 0, & i \neq j \end{cases}$$

که ما در اینجا به بردار های $\boldsymbol{v}_i$، بردار های منفرد راست A و به بردار های $\boldsymbol{u}_i$، بردار های منفرد چپ A می گوییم. در ادامه داریم:

$$AV = U\Sigma$$

بنابراین خواهیم داشت:

$$A\boldsymbol{v}_i = \begin{cases} \sigma_i \boldsymbol{u}_i, & i = 1, 2, \dots, r \\ 0, & i = r+1, \dots, N \end{cases}$$

این قابل اثبات است که مجموعه برداری $\{\boldsymbol{u}_1, \boldsymbol{u}_2, \dots, \boldsymbol{u}_r\}$ تشکیل یه پایه orthonormal برای Range ماتریس A ($Col\ A$) می دهد. حال از آنجایی که ماتریس A از ماتریس S با N تصویر چهره ساخته شده است، به Range ماتریس A، زیرفضای چهره (face subspace) در فضای تصویرهای $m \times n$ پیکسل (image space)، گفته می شود. به هر یک از بردار های $\boldsymbol{u}_i$ به ازای $i = 1, 2, \dots, r$ ، یک پایه چهره (base-face) گفته می شود.

از آنجایی که یک پایه برای زیر فضای چهره، دارای r بردار می باشد، بنابراین در صورتی که بخواهیم بردار های تصاویر را در این زیرفضا مشخص کنیم، می توانیم باید یک بردار با r مولفه این کار را انجام دهیم.

$$\boldsymbol{x} = [\boldsymbol{u}_1 \quad \boldsymbol{u}_2 \quad \cdots \quad \boldsymbol{u}_r]^T (\boldsymbol{f} - \bar{\boldsymbol{f}})$$

در واقع، این بردار $\boldsymbol{x}$ مشخص می کند که هر تصویر چهره $m \times n$ پیکسلی در چه موقعیتی از زیرفضای چهره قرار دارد. همانطور که از فصل ۶ کتاب به یاد داریم، برای محاسبه مختصات یک بردار دلخواه نسبت به یک پایه orthonormal داریم که مختصات مولفه $\boldsymbol{i}$ام برابر است با تصویر عمودی بردار دلخواه بر روی $\boldsymbol{i}$امین بردار پایه.

$$\begin{aligned} x_1 &= \boldsymbol{u}_1 \cdot (\boldsymbol{f} - \bar{\boldsymbol{f}}) = \boldsymbol{u}_1^T (\boldsymbol{f} - \bar{\boldsymbol{f}}) \\ x_2 &= \boldsymbol{u}_2 \cdot (\boldsymbol{f} - \bar{\boldsymbol{f}}) = \boldsymbol{u}_2^T (\boldsymbol{f} - \bar{\boldsymbol{f}}) \\ &\vdots \\ x_r &= \boldsymbol{u}_r \cdot (\boldsymbol{f} - \bar{\boldsymbol{f}}) = \boldsymbol{u}_r^T (\boldsymbol{f} - \bar{\boldsymbol{f}}) \end{aligned}$$

که می توان این را به صورت ضرب یک ماتریس در بردار نوشت و مستقیما بردار x را محاسبه کرد.

$$\boldsymbol{x} = \begin{bmatrix} \cdots & \boldsymbol{u}_1^T & \cdots \\ \cdots & \boldsymbol{u}_2^T & \cdots \\ & \vdots & \\ \cdots & \boldsymbol{u}_r^T & \cdots \end{bmatrix} (\boldsymbol{f} - \bar{\boldsymbol{f}})$$

در واقع این بردار مختصات $\boldsymbol{x}$، استفاده می شود تا مشخص کند که کدام یک از تصاویر موجود در چهره های آموزشی، بهترین تطابق را با تصویر ورودی $\boldsymbol{f}$ دارد. برای این کار، ما به دنبال تصویر چهره آموزشی $\boldsymbol{f}_i$ میگردیم که در زیرفضای چهره، کمترین فاصله را تا این تصویر چهره جدید داشته باشد. به این صورت که به ازای $i = 1, 2, \ldots, N$ داریم:

$$\varepsilon_i = \|\boldsymbol{x} - \boldsymbol{x}_i\|_2 = [(\boldsymbol{x} - \boldsymbol{x}_i)^T (\boldsymbol{x} - \boldsymbol{x}_i)]^{\frac{1}{2}}$$

به صورتی که بردار $\boldsymbol{x}_i$ بردار مختصات تصویر چهره $\boldsymbol{f}_i$ می باشد و مقدار آن همانند قبل، برابر تصویر عمودی $\boldsymbol{f} - \boldsymbol{f}_i$ بر روی پایه های زیرفضای چهره (base-faces) است.

$$\boldsymbol{x}_i = \begin{bmatrix} \cdots & \boldsymbol{u}_1^T & \cdots \\ \cdots & \boldsymbol{u}_2^T & \cdots \\ & \vdots & \\ \cdots & \boldsymbol{u}_r^T & \cdots \end{bmatrix} (\boldsymbol{f}_i - \bar{\boldsymbol{f}})$$

حال که ما فاصله تصویر چهره جدید $\boldsymbol{f}$ را به دست آوردیم، این تصویر زمانی به عنوان تصویر $\boldsymbol{f}_i$ انتخاب و دسته بندی می شود که مقدار مینیمم ε_i کوچکتر از یک حد آستانه از پیش تعریف شده ε_0 باشد. در غیر این صورت این تصویر چهره جدید، به عنوان یک "چهره نامشخص" دسته بندی می شود.

در صورتی که تصویر جدید $\boldsymbol{f}$، یک چهره نباشد، فاصله این تصویر تا زیرفضای چهره، مقداری بزرگتر از صفر خواهد بود. چرا که تصویر عمودی بردار $\boldsymbol{f} - \bar{\boldsymbol{f}}$ بر روی زیرفضای چهره را می توان به شکل زیر مصاحبه کرد.

$$\boldsymbol{f}_{projected} = \begin{bmatrix} \vdots & \vdots & & \vdots \\ \boldsymbol{u}_1 & \boldsymbol{u}_2 & \cdots & \boldsymbol{u}_r \\ \vdots & \vdots & & \vdots \end{bmatrix} \boldsymbol{x}$$

در این حالت، فاصله بردار تصویر $\boldsymbol{f}$ تا زیرفضای چهره، همان فاصله $\boldsymbol{f} - \bar{\boldsymbol{f}}$ تا تصویر عمودی آن بر روی این زیرفضاست. ($\boldsymbol{f}_{projected}$)

$$\varepsilon_f = \left\|(\boldsymbol{f} - \bar{\boldsymbol{f}}) - \boldsymbol{f}_{projected}\right\|_2$$
$$= \left[(\boldsymbol{f} - \bar{\boldsymbol{f}} - \boldsymbol{f}_{projected})^T(\boldsymbol{f} - \bar{\boldsymbol{f}} - \boldsymbol{f}_{projected})\right]^{\frac{1}{2}}$$

در صورتی که مقدار محاسبه شده ε_f بزرگتر از حد آستانه از پیش تعریف شده ε_1 باشد، آنگاه می توان بیان کرد که بردار تصویر $\boldsymbol{f}$ یک تصویر چهره نیست.

اما یک سوال بنیادین ممکنه که برای شما هم پیش اومده باشه.

"وقتی که میتونیم دو تا تصویر $\boldsymbol{f}_1$ و $\boldsymbol{f}_2$ رو مستقیما با هم مقایسه کنیم، چرا برای مقایسشون قصد داریم مختصات این دو تا تصویر چهره رو در زیرفضای چهره به دست بیاریم؟ این کار چه فایده ای برامون داره؟"

جواب خیلی سادست. برای مقایسه دو تصویر به طور مستقیم، لازمه که تمامی پیکسل های اونا با هم مقایسه بشن که همونطور که قبلا دیدیم اگه دو تا تصویرمون ابعاد $M = m \times n$ داشته باشند، باید M پیکسل با هم بررسی بشن در حالی که وقتی ما تلاش می کنیم مختصات هر تصویر چهره رو در زیرفضای چهره به دست بیاریم، صرفا یک بردار با ابعاد r داریم که همونطور که قبل تر هم دیدیم $r \leq N \ll M$ هست. پس لازمه که تعداد مولفه های کمتری برای این مقایسه بررسی بشه و به وضوح مشخصه که این روش نسبت به حالت اول، سرعت بالاتری در محاسبات برای ما به همراه داره.

گام‌های پیاده‌سازی

در این پروژه، همانند پروژه دوم، فایلی برایتان قرار داده شده است که در پیاده سازی پروژه به شما کمک می‌کند.

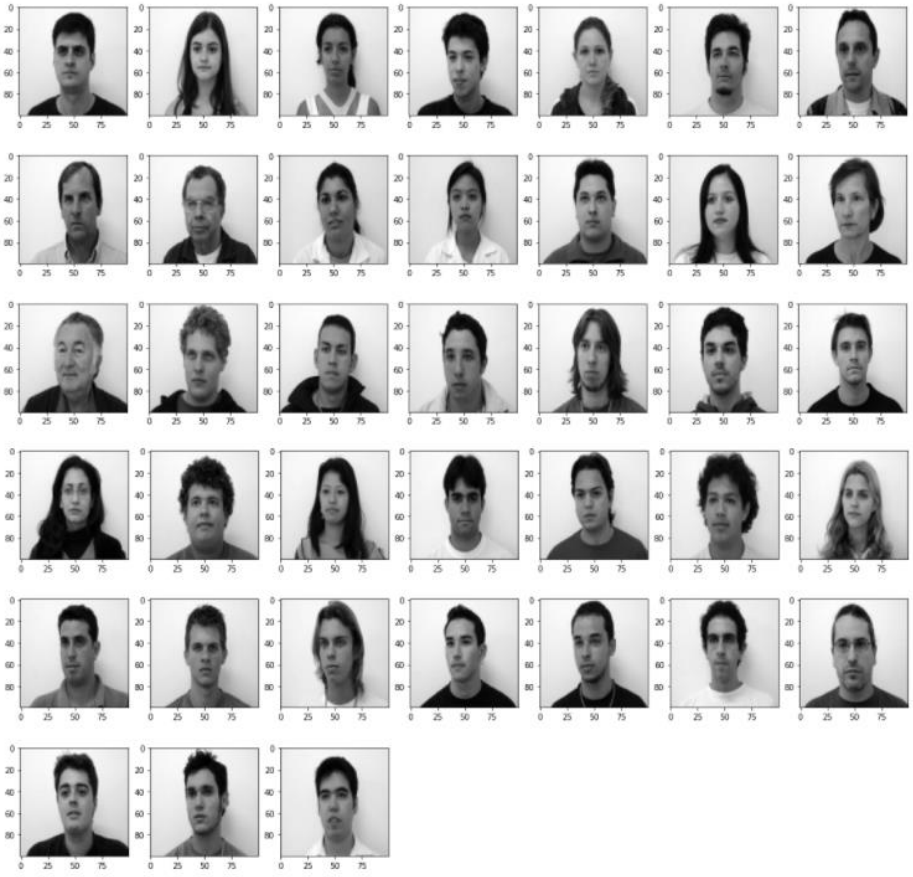
تنها کافیست بخش‌های مشخص شده TODO در فایل main.py را تکمیل کنید.

گام ۱ : خواندن داده‌های train و test

این گام برای شما در تابع load_images_train_and_test به طور کامل پیاده‌سازی شده است.

نمونه ای از چهره افراد مختلف موجود در دیتاست داده شده را در تصویر پایین مشاهده می‌کنید.

در پوشه ی Dataset داده های Train و در پوشه ی Testset داده های Test می باشد . تصاویری که عدد اول اسمشان یکسان است ، یعنی یک نفر می باشند .

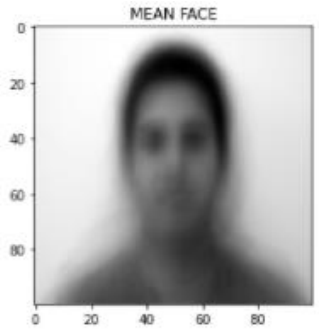


گام ۲: نرمال سازی داده‌های train و test

در این بخش باید تابع normalize را کامل کنید.

ابتدا تصویر میانگین را با استفاده از تابع np.mean به دست آورید و سپس تصویر میانگین را از کل داده های train و test کم کنید تا نرمال شوند.

در تصویر زیر، تصویر میانگین برایتان رسم شده است تا دید بهتری بگیرید.



گام ۳ : تجزیه SVD

در این مرحله کافیست با استفاده از تابع np.linalg.svd، ماتریس داده‌های train را تجزیه کنید.

دقت کنید که حتماً پارامتر full_matrices=False باشد.

گام ۴ (امتیازی)

(در صورت انجام ، علاوه بر نمره خود پروژه ، ۴۰ درصد از نمره این پروژه به نمره کل پروژه ها اضافه می شود)

گام سوم را بدون استفاده از np.linalg.svd پیاده‌سازی کنید. به عبارتی، تجزیه SVD را تنها با استفاده از اعمال پایه ماتریسی (همانند ضرب ماتریس ها، وارون، کاهش ردیفی و ...) پیاده کنید.

گام ۵ : محاسبه وزن w برای test و train

در تابع project_and_calculate_weights(img, u) ، ضرب درایه ای img در u (Element-wise Multiplication) را به دست آورید و آن را برگردانید.

گام ۶ : پیشبینی چهره صحیح

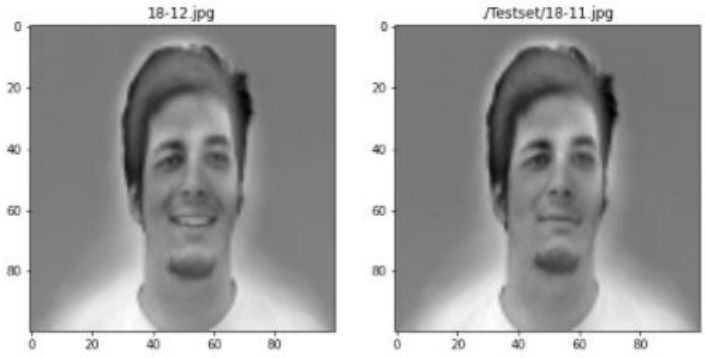
در تابع predict باید شبیه‌ترین چهره را در داده‌های train به داده ی test، بدست آورید.
برای تعیین شبیه‌ترین داده، کافیست داده ای در train که norm خطای آن کمترین است را به دست آورید.

$$error(i) = train(i) - test$$

در نهایت موقعیت(index) شبیه‌ترین داده را برگردانید.

گام ۷ : بررسی درستی یا نادرستی پیشبینی و رسم به وسیله matplotlib

تابع plot_face(test, predicted) که دو داده تصویر ورودی(test) و داده تصویر پیشبینی‌شده(predicted) را می‌گیرد و ترجیحاً در کنارهم نمایش می‌دهد را با استفاده از کتابخانه matplotlib پیاده‌سازی کنید. (در صورتیکه تصویر حالت سبز مانند داشت ، مشکلی ندارد با این حال می توانید با تغییر colormap (cmap = "gray") آن را سیاه سفید نشان دهید)
این تابع ، پیشبینی های درست یا نادرست شما را در هر داده ی test نمایش خواهد داد و می توانید برای سریعتر دیباگ کردن ، با کامنت کردن خطوطی که از آن استفاده شده ، نتیجه را صرفا در terminal خود ببینید .
نمونه یک پیشبینی درست در تصویر پایین آورده شده است.



در انتها اگر کدتان به صورت درست پیاده سازی شده باشد حدود ۹۷ درصد از داده های test درست تشخیص داده خواهد شد.

قوانین و ددلاین

- ددلاین این پروژه،ساعت ۲۳:۵۹ روز ۵ بهمن می‌باشد.

- تنها نیاز است کد main.py خود را بعد از تکمیل، در صفحه کوئرا آپلود کنید.

- هر دانشجو می‌بایست پروژه را به صورت انفرادی انجام دهد. تقلب‌ها به صورت خودکار، توسط سامانه کوئرا بررسی خواهد شد.

- از آن جایی که زبان برنامه نویسی پایتون، یکی از زبان‌های پرکاربرد در حوزه جبر خطی است و آموزش‌های مربوط به این زبان و کتابخانه‌های آن، توسط تیم تدریس‌یاری در اختیار شما قرار گرفته است، بنابراین برای پیاده سازی این پروژه تنها مجاز به استفاده از زبان پایتون و کتابخانه های Matplotlib، Numpy و Pandas در کنار توابع و کتابخانه‌های پیش فرض پایتون هستید. استفاده از هر زبان برنامه‌نویسی یا کتابخانه‌ای دیگر قابل قبول نبوده و در صورت استفاده، نمره‌ای به شما تعلق نخواهد گرفت.

- این پروژه تحویل آنلاین خواهد داشت و پس از پایان مهلت تحویل این پروژه، زمانبندی مربوط به تحویل به زودی اعلام خواهد شد.

- رعایت تمیزی کد برای پیاده سازی پروژه به شدت استقبال می‌شود.

با آرزوی موفقیت و سلامتی

تیم تدریس یاری جبر خطی کاربردی، پاییز ۱۴۰۰