باسمه تعالی

دانشگاه صنعتی شریف

دانشکده مهندسی برق

گروه دکتر امینی - روش های ریاضی در مهندسی
نیمسال اول ۰۲-۱۴۰۱

تمرین سری چهارم (بخش عملی)

---

۱. مهلت تحویل این تمرین مطابق تاریخ اعلام شده در سامانه CW می باشد.

۲. ۱۰ روز تاخیر مجاز برای تحویل تمرین های تئوری در اختیار شما خواهد بود.

۳. سقف تاخیر برای تحویل هر تمرین ۷ روز خواهد بود و پس از آن پاسخنامه تمرین منتشر خواهد شد.

۴. ابهامات و مشکلات خود در مورد این تمرین را می توانید با دستیاران طراح، آقای زرگران و خانم حریقی مطرح کنید.

@cloner_z4 , @SN_HAR

در درس با تجزیه SVD آشنا شدیم. یکی از کاربردهاي تجزیه SVD ،کاهش حجم داده ها است. در این تمرین می خواهیم عکس هایی با فرمت BMP را که فشرده سازي نشده اند، به کمک SVD فشرده کنیم و حجم آنها را کاهش دهیم.
در فرمت BMP فشرده سازي بر روي عکس انجام نمی شود و اطلاعات پیکسل ها به شکل خام و به صورت یک عدد بین ۰ تا۲۵۵ ذخیره می شوند. فایل BMP در تصاویر RGB ، از ۳ آرایه ۲ بعدي تشکیل شده است که هر یک از این سه آرایه، مربوط به یک کانال رنگی است. براي باز کردن یک فایل BMP در پایتون، می توانید از تابع *imread* و برای نمایش آن می توانید از تابع *imshow* در کتابخانه *matplotlib* استفاده کنید.
مراحل کار:

۱. یکی از فایل هاي BMP که در اختیارتان قرار گرفته است را به کمک *matplotlib* در قالب یک آرایه ۳ بعدي *numpy* (مثلا به نام img) لود کرده و سپس آن را نمایش دهید.

۲. کانالهای رنگی مختلف را جدا کرده و در ماتریس هاي جداگانه ذخیره کنید.

$$r = img[:, :, 0]$$
$$b = img[:, :, 1]$$
$$g = img[:, :, 2]$$

۳. به کمک توابع کتابخانه ای، براي هر یک از ۳ ماتریس مرحله قبل تجزیه SVD را محاسبه کنید (تابع svd در $numpy$). حاصل تجزیه مطابق ۱ است:

$$\mathbf{A} = \begin{bmatrix} \mathbf{u_1} & \mathbf{u_2} & \dots & \mathbf{u_m} \end{bmatrix} \begin{bmatrix} \sigma_1 & 0 & \dots & 0 & 0 & \dots & 0 \\ 0 & \sigma_2 & \dots & 0 & 0 & \dots & 0 \\ \vdots & \vdots & \dots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \dots & \sigma_r & 0 & \dots & 0 \\ 0 & 0 & \dots & 0 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \dots & 0 & 0 & \dots & 0 \end{bmatrix} \begin{bmatrix} \mathbf{v_1}^T \\ \mathbf{v_2}^T \\ \vdots \\ \mathbf{v_n}^T \end{bmatrix}$$

شکل ۱

عبارت ۱ را می توان به صورت ۲ نیز نوشت:

$$\mathbf{A} = \begin{bmatrix}\mathbf{u_1} & \mathbf{u_2} & \dots & \mathbf{u_m}\end{bmatrix} \left.\begin{bmatrix}\sigma_1\mathbf{v_1}^T \\ \sigma_2\mathbf{v_2}^T \\ \vdots \\ \sigma_r\mathbf{v_r}^T \\ 0 \\ \vdots \\ 0\end{bmatrix}\right\}\text{m rows}$$
$$= \sigma_1\mathbf{u_1}\mathbf{v_1}^T + \sigma_2\mathbf{u_2}\mathbf{v_2}^T + \dots + \sigma_r\mathbf{u_r}\mathbf{v_r}^T$$

شکل ۲

از آنجا که مقادیر تکین در قطر ماتریس Σ به شکل نزولی مرتب شده اند، تاثیر جملات ابتدایی عبارت ۲ از جملات بعدی بیشتر است. در نتیجه می توان با در نظر گرفتن k جمله اول، تخمین مناسبی از ماتریس اولیه داشته باشیم (۳).

$$\mathbf{A_k} = \begin{bmatrix}\mathbf{u_1} & \mathbf{u_2} & \dots & \mathbf{u_k}\end{bmatrix} \begin{bmatrix}\sigma_1 & 0 & \dots & 0 \\ 0 & \sigma_2 & \dots & 0 \\ \vdots & \vdots & \dots & \vdots \\ 0 & 0 & \dots & \sigma_k\end{bmatrix} \begin{bmatrix}\mathbf{v_1}^T \\ \mathbf{v_2}^T \\ \vdots \\ \mathbf{v_k}^T\end{bmatrix}$$
$$= \sigma_1\mathbf{u_1}\mathbf{v_1}^T + \sigma_2\mathbf{u_2}\mathbf{v_2}^T + \dots + \sigma_k\mathbf{u_k}\mathbf{v_k}^T$$

شکل ۳

۱. عبارت ۳ را براي $k = 50$ محاسبه کنید. سپس ۳ ماتریس حاصل را با یکدیگر ترکیب کرده تا ماتریس کامل تصویر ایجاد شود. با رسم تصویر به دست آمده، وضوح آن را با تصویر اصلی مقایسه کنید.

۲. مرحله قبل را براي $k = 10$ ، $k = 150$ و $k = 250$ تکرار کنید. مشاهده می کنید که با افزایش k وضوح تصویر بهتر شده و به تصویر اصلی نزدیکتر می شود.

توجه کنید از آنجا که ماتریس تصویر حاصل هم اندازه ماتریس اصلی است، به نظر می رسد در اینجا فشرده سازي صورت نگرفته است. اما دقت کنید در اینجا نیازي به ذخیره سازي ماتریس نهایی براي عکس نداریم بلکه کافیست ستونهایی از U ، مقادیر تکین Σ و سطرهایی از V^T را که مربوط به k جمله ابتدایی بسط SVD است را ذخیره کنیم و در هنگام نمایش عکس، آن را باز تولید کنیم.
به عنوان مثال اگر یک تصویر RGB با ابعاد 1080×1920 را ذخیره کنیم، نیاز به $3 \times 1080 \times 1920$ درایه در ماتریس آن داریم. اما اگر از طریق بسط SVD تا جمله $k = 150$ آن را ذخیره کنیم، نیاز به ذخیره ۱۵۰ ستون از U ، ۱۵۰ مقدار تکین و ۱۵۰ سطر از V^T را داریم. یعنی در مجموع نیاز به ذخیره

$$3 \times (150 \times 1920 + 150 + 150 \times 1080) = 1,350,450$$

درایه داریم. در نتیجه تصویر اصلی حدود 4.6 برابر کوچکتر شده است (توجه کنید نیازي به ذخیره عکسها و باز تولید آنها با روش گفته شده نیست).