تمرین عملی سری سوم

در درس با تجزیه SVD آشنا شدیم. یکی از کاربردهای تجزیه SVD، کاهش حجم داده‌ها است. در این تمرین می‌خواهیم عکس‌هایی با فرمت BMP را که فشرده‌سازی نشده‌اند به کمک SVD فشرده کنیم و حجم آن‌ها را کاهش دهیم.

در فرمت BMP فشرده‌سازی بر روی عکس انجام نمی‌شود و اطلاعات پیکسل‌ها به شکل خام و به صورت یک عدد بین ۰ تا ۲۵۵ ذخیره می‌شوند. فایل BMP در تصاویر RGB، از ۳ آرایه ۲ بعدی تشکیل شده است که هر یک از این سه آرایه مربوط به یک کانال رنگی است. برای باز کردن یک فایل BMP در پایتون، می‌توانید از تابع imread و برای نمایش آن می‌توانید از تابع imshow در کتابخانه matplotlib استفاده کنید.

مراحل کار:

۱. یکی از فایل‌های BMP که در اختیارتان قرار گرفته است را به کمک matplotlib در قالب یک آرایه ۳ بعدی numpy (مثلا به نام img) لود کرده و سپس آن را نمایش دهید.
۲. کانال‌های رنگی مختلف را جدا کرده و در ماتریس‌های جداگانه ذخیره کنید.

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

۳. به کمک توابع کتابخانه‌ای برای هر یک از ۳ ماتریس مرحله قبل تجزیه SVD را محاسبه کنید (تابع svd در numpy). حاصل تجزیه به شکل زیر است:

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

عبارت بالا را می‌توان به شکل

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

$$= \sigma_1 \mathbf{u_1} \mathbf{v_1}^T + \sigma_2 \mathbf{u_2} \mathbf{v_2}^T + \dots + \sigma_r \mathbf{u_r} \mathbf{v_r}^T$$

نیز نوشت. از آنجا که مقادیر تکین در قطر ماتریس Σ به شکل نزولی مرتب شده‌اند، تاثیر جملات ابتدایی عبارت بالا از جملات بعدی بیشتر است. در نتیجه می‌توان با در نظر k جمله اول، تخمین مناسبی از ماتریس اولیه داشته باشیم. یعنی:

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

۱. عبارت بالا را برای $k = 50$ 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 برای $k = 10$، $k = 150$ و $k = 250$ تکرار کنید. مشاهده می‌کنید که با افزایش k وضوح تصویر بهتر شده و به تصویر اصلی نزدیک‌تر می‌شود.

توجه کنید از آنجا که ماتریس تصویر حاصل هم‌اندازه ماتریس اصلی است، به نظر می‌رسد در اینجا فشرده‌سازی صورت نگرفته است. اما دقت کنید در اینجا نیازی به ذخیره‌سازی ماتریس نهایی برای 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 1920×1080 را ذخیره کنیم، نیاز به $1920 \times 1080 \times 3 = 6,220,800$ درایه در ماتریس آن داریم. اما اگر از طریق بسط SVD تا جمله $k = 150$ آن را ذخیره‌کنیم، 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 ۴,۶ برابر کوچک‌تر شده است (توجه کنید نیازی به ذخیره عکس‌ها و باز تولید آن‌ها با روش گفته شده نیست).

موفق باشید

تیم تدریس‌یاری جبر خطی کاربردی

بهار ۱۴۰۰