دستور کار: فشرده سازی تصاویر

در این بخش می خواهیم با استفاده از دو روش lossy و lossless تصاویر را فشرده سازی کنیم. چرا به این کار نیاز داریم؟ طبق آمار سال 2022، روزانه 650 میلیون توییت انجام می‌شود که 45% آنها تصاویر هستند! پس فشرده سازی فایل ها از جمله تصاویر نقش حیاتی در ارتباطات دارد :)

۱. بخش اول - سوالات تئوری - *(۴۰ نمره)*

۱-۱. تفاوت فشرده سازی lossy و lossless چیست؟ چه فرمت های تصویری lossy و چه فرمت هایی lossless هستند؟

۱-۲. گام های فشرده سازی تصاویر JPEG را نام برده و هر کدام را مختصرا توضیح دهید.

۱-۳. چهار حالت ذخیره سازی تصاویر با فرمت JPEG را نام برده و شرح دهید.

۱-۴. مزایا و معایب الگوریتم فشرده سازی هافمن (Huffman Coding) را نام ببرید.

۱-۵. فشرده سازی تصویر چیست و چرا اهمیت دارد؟

۱-۶. تفاوت میان فشرده سازی های Spatial و transform-based چیست و مزایا و معایب هرکدام را بگویید.

۱-۷. استفاده های فشرده سازی تصویر در دنیای واقعی چیست؟ مثال هایی از استفاده از فشرده سازی تصویر بیاورید و آن ها را شرح دهید

۱-۸. مشکلات رایج فشرده سازی تصویر چیست و چگونه میتوان از آنها دوری کرد؟

۲. بخش دوم - تمرینات عملی - *(۶۰ نمره)*

۲-۳. الگوریتم فشرده سازی هافمن - *(۲۵ نمره)*

الگوریتم کد کردن هافمن (Huffman) یک روش فشرده سازی lossless است که معمولا برای فرمت های JPEG و MP3 استفاده میشود. در این بخش می خواهیم گام های فشرده سازی تصاویر به وسیله این الگوریتم را پیاده سازی کنیم.

۲-۳-۱. گام اول

تابعی پیاده سازی کنید که یک تصویر رنگی را بخواند و آن را در یک آرایه سه بعدی ndarray بریزد. (height, width, channel)

دستور کار: فشرده سازی تصاویر

2-۳-۲. گام دوم

تابعی که یک تبدیل فضای رنگی را روی تصویر ورودی انجام می‌دهد، مانند تبدیل از RGB به YCbCr، و یک آرایه کم‌رنگ سه‌بعدی از مقادیر پیکسل تبدیل‌شده را برمی‌گرداند.

2-۳-۳. گام سوم

تابعی که مقادیر پیکسل تبدیل‌شده را به بلوک‌های ۸×۸ تقسیم می‌کند و تبدیل کسینوس گسسته (DCT) را روی هر بلوک انجام می‌دهد، که در نتیجه یک آرایه ناقص سه‌بعدی از ضرایب DCT ایجاد می‌شود.

تابعی که یک درخت هافمن را برای یک مجموعه معین از ضرایب DCT، بر اساس فراوانی مقادیر ضرایب در هر بلوک و در همه بلوک‌ها، ایجاد می‌کند.

2-۳-۴. گام چهارم

تابعی که ضرایب DCT را با استفاده از یک ماتریس کوانتیزه‌سازی از پیش تعریف‌شده کوانتیزه می‌کند، که منجر به یک آرایه‌ای سه بعدی از ضرایب کوانتیزه می‌شود.

تابعی که مجموعه معینی از ضرایب کوانتیزه شده را با استفاده از درخت هافمن کد می کند و تصویر کدگذاری شده را به صورت یک رشته باینری برمی گرداند.

2-۳-۵. گام پنجم

تابعی که یک رشته باینری معین را با استفاده از درخت هافمن رمزگشایی می کند و مجموعه رمزگشایی شده از ضرایب کوانتیزه شده را به صورت یک آرایه کمپی سه بعدی برمی گرداند.

تابعی که بر روی ضرایب کوانتیزه شده کوانتیزه معکوس انجام می دهد، با استفاده از همان ماتریس کوانتیزه سازی استفاده شده در مرحله 5، که منجر به یک آرایه ناقص سه بعدی از ضرایب کم شده می شود.

2-۳-۶. گام ششم

تابعی که DCT معکوس را بر روی هر بلوک ضریب dequantize انجام می‌دهد، که منجر به یک آرایه ناقص سه بعدی از مقادیر پیکسل بازسازی‌شده می‌شود. و تابعی که تبدیل فضای رنگی معکوس را بر روی مقادیر پیکسل بازسازی شده انجام می دهد، که منجر به یک آرایه ناپخته سه بعدی از مقادیر پیکسل خروجی نهایی می شود.

دستور کار: فشرده سازی تصاویر

۲-۳-7. گام هفتم

یک تابع اصلی که توابع بالا را برای فشرده سازی و از حالت فشرده خارج کردن یک فایل تصویر رنگی خاص فراخوانی می کند.

بخش های اضافی

1. **یک الگوریتم کدگذاری پویا هافمن را اجرا کنید که دفترچه کد و ساختار درختی را به عنوان نمادهای جدید در طول رمزگذاری تنظیم می کند.**
2. **با ماتریس‌های کوانتیزاسیون مختلف آزمایش کنید تا ببینید که چگونه بر نسبت فشرده‌سازی و کیفیت تصویر تأثیر می‌گذارند و به کاربر این امکان را می‌دهد که انتخاب کند از کدام ماتریس استفاده کند.**
3. **با استفاده از کدگذاری حسابی به جای کدگذاری هافمن، یک نسخه lossless از الگوریتم فشرده سازی را پیاده سازی کنید و نسبت های فشرده سازی و زمان اجرا را بین دو رویکرد مقایسه کنید.**
4. **با تبدیل فضای رنگی مختلف، مانند تبدیل به فضای رنگی HSV، آزمایش کنید و کیفیت تصویر و نسبت فشرده سازی حاصل را مقایسه کنید.**
5. **یک الگوریتم watermark برگشت پذیر را پیاده سازی کنید که مقدار کمی از اطلاعات را در تصویر فشرده قرار می دهد و بعداً می تواند بدون تاثیر بر کیفیت تصویر از حالت فشرده خارج شود.**
6. **گزینه ای برای visualization ضرایب DCT و درخت هافمن اضافه کنید تا به کاربر در درک نحوه عملکرد الگوریتم فشرده سازی کمک کند.**

۲-۳-۲. الگوریتم K-Means برای فشرده سازی تصاویر - *(۳۵ نمره)*

می‌خواهیم از یک الگوریتم بدون نظارت یا Unsupervised به نام K-Means استفاده کنیم تا تصاویر را فشرده سازی کنیم (برای درک بهتر این الگوریتم این ویدئو را ببینید). برخلاف الگوریتم هافمن که در درس یاد گرفتیم، این الگوریتم lossy است!

روش اجرای این الگوریتم به صورت زیر است:

۱. تعداد دسته ها یا K را تعیین می‌کنیم.

۲. در این گام K نقطه به عنوان Centroid یا مرکز دسته ها انتخاب می‌کنیم.

۳. برای هر نقطه از تصویر، نزدیک ترین centroid به آن را پیدا می‌کنیم.

۴. سپس Centroid های جدید را محاسبه کرده و جایگزین قبلی ها می‌کنیم.

۵. سپس گام های ۳ و ۴ را آنقدر اجرا کرده تا الگوریتم همگرا شود. (مرکز ها تغییر نکنند)

دستور کار: فشرده سازی تصاویر

۶. در گام آخر عکس جدید را می‌سازیم.

سوالات:

- چرا این الگوریتم از نوع lossy است؟ دلیل خود را به طور دقیق شرح دهید.
- تعداد دسته ها چه تاثیری بر قدرت فشرده سازی، میزان lossy بودن و سرعت اجرای الگوریتم دارد؟

نکات پیاده سازی:

- شرط خاتمه الگوریتم میزان همگرا شدن مرکز ها است. مقدار threshold همگرایی به اختیار شماست.
- دقت کنید تصویر شما یک ماتریس ۳ بعدی است (Width, Height, Color-Depth)، پس الگوریتم شما باید روی هر ۳ کانال رنگ اجرا شود.
- استراتژی انتخاب centroid های اولیه، تاثیر بسزایی در عملکرد الگوریتم دارد. یک استراتژی مناسب انتخاب کنید. (انتخاب رندوم نباشد)
- ایده های خلاقانه برای افزایش سرعت الگوریتم و یا بهینه سازی آن نمره اضافی دارد.

نکات دیگر:

- استفاده از کتابخانه *Numpy* برای سریع تر کردن اجرای الگوریتم مجاز **است**.
- استفاده از کتابخانه های آماده مانند *Scikit-Learn* و *Tensorflow* (و نظیر آنها 😉) مجاز **نیست**!
- گزارش کار مختصری تهیه کرده و پاسخ سوالات به همراه شرح مراحل که انجام داده اید تهیه کنید.

دستور کار: فشرده سازی تصاویر

تصویر انتخابی برای این تمرین (سورس اصلی):