دانشکده مهندسی و علوم کامپیوتر

پروژه نهایی

سیگنال و سیستم ها

استاد: دکتر سلیمی‌بدر

نکته) حتماً به توضیحات انتهای پروژه توجه کنید

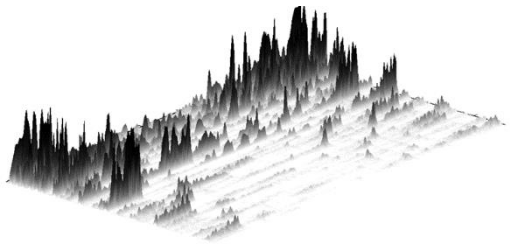
بخش اول)

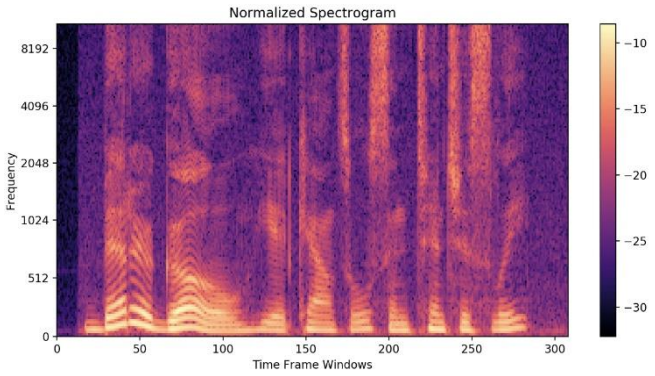
پردازش سیگنال صوتی

پیش نیاز:

در بخش اول این پروژه میخواهیم با کمک مباحثی که در درس آموختید، یک سری پردازش بر روی یک فایل صوتی اعمال کنیم و با نحوه ی تصویر سازی خروجی با کمک اسپکتروگرام (spectrogram) آشنا شویم. اسپکتروگرام ها برای نمایش محتوای فرکانسی سیگنال ها در طول زمان به کار می روند و به کمک آنها می‌توان از سیگنال های تک بعدی تصویر دوبعدی ایجاد کرد و شهود بهتری از ویژگی‌های سیگنال به دست آورد. اسپکتروگرام در زمینه ی پردازش صوت و یا کاربرد های پزشکی به فراوانی استفاده میشود. اسپکتروگرام ها معمولا نمودار های دوبعدی (یا سه بعدی) هستند که در یک محور زمان و در محور دیگر فرکانس را نمایش میدهند. بعد دیگر آنها که با شدت رنگ های متفاوت نمایش داده میشود، دامنه ی یک فرکانس در زمان مشخص را نمایش میدهد.

یک نمونه اسپکتروگرام:





برای مطالعه بیشتر در مورد اسپکتروگرام ها می توانید از این لینک استفاده کنید.

شرح تمرین:

یک فایل صوتی به نام noisy.wav در کنار فایل های تمرین ضمیمه شده. این فایل شامل نویز است و از کیفیت مناسبی برخوردار نیست. بنابراین:

- در ابتدا باید سیگنال را به نحوی پردازش کنید که نویز ها از آن حذف شود و فایل صوتی با کیفیت مناسب را با نام clean.wav ذخیره کنید.
- پس از استخراج فایل صوتی بدون نویز باید آنرا مجددا خوانده و سرعت پخش آنرا **دو برابر** و **نصف** کنید و از هرکدام خروجی‌هایی به ترتیب به نام های fast.wav و slow.wav ذخیره کنید. این کار باید به کمک تبدیل فوریه در حوزه فرکانس صورت گیرد.
- برای هرکدام از فایل های خروجی و همینطور فایل اصلی (noisy) نمودار سیگنال در حوزه‌ی زمان، فرکانس و همینطور اسپکتروگرام آن سیگنال را به کمک دستور subplot در کنار هم رسم کنید. (برای هر تصویر یک plot داریم که هرکدام شامل سه تصویر زمان، فرکانس و اسپکتروگرام هستند)
- اسپکتروگرام های هرکدام از خروجی ها را با سیگنال اولیه مقایسه کنید و ویژگی های آنها را تحلیل کنید.

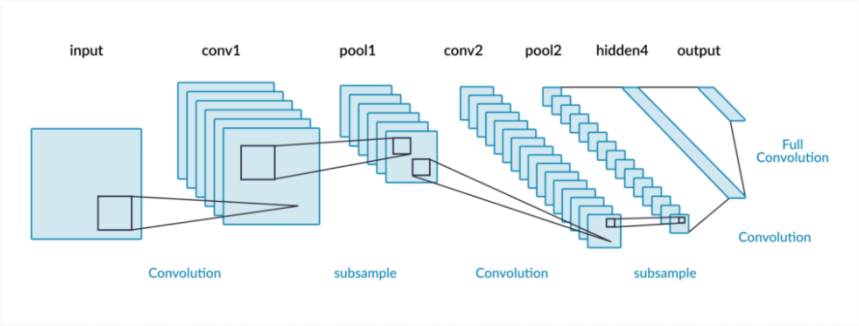
نکات) برای تولید spectrogram و اعمال تبدیل فوریه استفاده از کتابخانه های جانبی مثل scipy مجاز است. اما ساخت فیلتر و اعمال آن باید توسط خودتان انجام شود.

بخش دوم)

پیاده سازی و اعمال کانوولوشن دو بعدی روی تصاویر

پیش نیاز:

همانطور که در کلاس حل تمرین توضیح داده شده، شبکه های عصبی پیچشی (convolutional neural networks) از عملگر کانوولوشن دو بعدی برای استخراج ویژگی های مفید از تصاویر استفاده می‌کنند تا از این ویژگی ها در ادامه برای کارهایی مانند دسته بندی، تشخیص چهره و ... استفاده شود. پیش از به وجود آمدن این نوع شبکه عصبی، متخصصان پردازش تصویر بصورت دستی یک سری فیلتر برای استخراج ویژگی‌های خاصی مثل لبه‌ها، یا smooth کردن و sharp کردن تصویر طراحی میکردند، اما شبکه‌های عصبی پیچشی (CNN به اختصار)، می توانند با کمک الگوریتم‌های یادگیری ماشین این فیلتر ها را یاد بگیرند و دیگر نیاز به طراحی دستی فیلترها نیست.



عملکرد CNN ها از دو فاز روبه جلو (forward pass) و رو به عقب (backward pass) تشکیل شده. در فاز روبه جلو فیلتر ها رو تصاویر اعمال میشوند و در فاز رو به عقب به کمک الگوریتم پس انتشار (backpropagation) مقادیر فیلتر ها اصلاح میشوند تا خروجی به نتیجه دلخواه نزدیک تر شود. این توالی forward pass و backward pass به دفعات تکرار میشود. فاز backward pass و اصلاح مقادیر فیلترها برای بدست آوردن فیلترهای بهینه از چهارچوب درس سیگنال و سیستم خارج است و بنابراین نیاز به دانستن جزئییات آن نیست، اما لازم است فاز رو به جلو (اعمال فیلتر) را که جزو درس است در این تمرین پیاده سازی نمایید.

شرح تمرین:

دوست شما این ترم پروژه کارشناسی دارد و باید یک شبکه عصبی پیچشی (convolutional neural networks) را پیاده سازی کند، اما فرصتش بسیار محدود است او میداند که شما این ترم درس سیگنال و سیستم را گذرانده اید، بنابراین از شما کمک خواسته تا فقط بخش رو به جلو (اعمال کانوولوشن) را برایش پیاده سازی کنید.

شما باید تابع زیر را پیاده سازی کنید: (امضای تابع در متلب)

```
function [feature_map] = conv2D(img, filters, stride, padding)
```

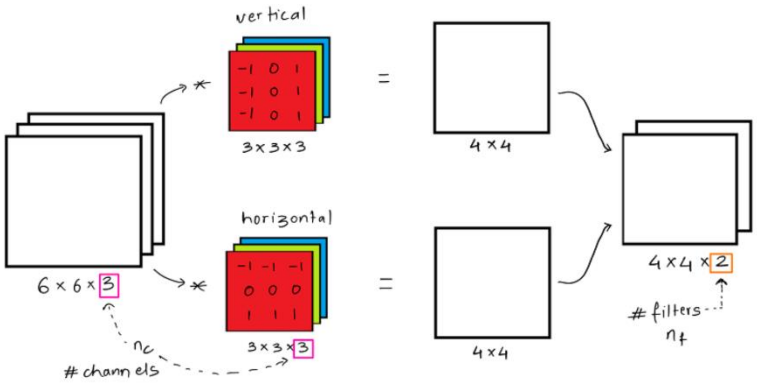
img: یک آرایه ورودی به سایز M x N x C است، برای تصاویر رنگی C=3 است، چون 3 کانال RGB داریم، برای تصاویر سیاه و سفید C=1 است. برای feature map های حاصل از خروجی یک لایه کانوولوشن این عدد برابر تعداد فیلتر های لایه قبل است و لزومی ندارد حتماً ۱ یا ۳ باشد، در ادامه بیشتر توضیح داده میشود.

filters: یک تنسور (آرایه سه بعدی) به سایز f x f x number_of_filters است که f مشخص کننده سایز فیلتر ها است و number_of_filters تعداد فیلتر های استفاده شده است. در واقع این تنسور شامل number_of_filters فیلتر f x f است.

padding: متغیری از جنس رشته است که مقدار آن یکی از رشته های ’valid‘ یا ’same‘ است. اگر same باشد یعنی به اندازه ای که سایز خروجی کانولوشن برابر تصویر اصلی شود zero-padding بر روی تصویر اولیه اعمال میکنیم (مراجعه به اسلاید کلاس حل تمرین)، در غیر اینصورت اعمال نشود.

output (feature map): خروجی حاصل از اعمال کانوولوشن با یک یا چند فیلتر، یک تنسور به سایز M_new x N_new x number_of_filters که کانال iام آن خروجی حاصل از فیلتر i ام است. این خروجی ممکن است به عنوان ورودی یک تابع Conv2D دیگر استفاده شود (بعنوان img)

توضیحات مربوط به عملگر کانوولوشن روی تصویر در کلاس داده شده و کد نسخه ی ساده تر آن زده شده، برای تسلط بیشتر روی عملکرد آن به کلاس و اسلاید های آن مراجعه کنید، تصویر زیر برای درک بهتر آورده شده:



در این شکل یک تصویر به سایز $6 \times 6 \times 3$ به عنوان ورودی به تابع داده شده است. سپس دو فیلتر تشخیص دهنده لبه‌های افقی و عمودی به سایز 3×3 با عکس ورودی کانوالو شده اند. (دلیل اینکه در شکل بالا سایز فیلترها $3 \times 3 \times 3$ است را باید طبق صحبت‌های تدریس شده خودتان بدانید) خروجی حاصل از عمل کانوولوشن عکس ورودی با هرکدام از فیلترها یک آرایه دو بعدی به سایز 4×4 است.(اینکه عدد 4 از کجا بدست آمده است هم خودتان باید بدانید) در نهایت با الحاق خروجی های بدست آمده، یک feature map با سایز $4 \times 4 \times 2$ بدست آمده است.

در شبکه ی عصبی پیچشی، تعداد فیلتر لایه ی قبلی، تعداد کانال های لایه ی بعدی میشوند. مثلاً در این تصویر اگر بعد از این لایه دوباره تابع Conv2D فراخوانی شود ورودی آن (img) 4x4x2 خواهد بود.

پس از پایان پیاده سازی این تابع، باید آنرا به کمک فیلتر هایی که به شما داده شده روی تصاویر ضمیمه تست کنید. در واقع شما باید مرحله ی اعمال فیلتر را در یک شبکه ی عصبی پیچشی سه لایه انجام دهید و نتیجه را گزارش کنید.

در این تمرین در لایه اول ۴ فیلتر 3x3 به شما داده شده که باید با اعمال آنها feature map خروجی را به دست بیاورید، سپس در لایه ی دوم بر روی feature map خروجی لایه اول دو فیلتر دیگر اعمال کنید در نهایت در لایه سوم یک فیلتر 5x5 اعمال کنید و بر روی خروجی max pooling اعمال کنید. هرکدام از تصاویر موجود در feature map های هر لایه را به همراه تصویر اصلی به کمک دستور subplot در یک plot رسم کنید. مثلاً اگر ۴ فیلتر در یک لایه داشتید، plot شما برای این لایه باید شامل ۵ subplot باشد. این plot ها را در **گزارش خود قرار دهید**.

فیلتر های لایه ی اول و دوم و سوم به همراه دیگر پارامتر ها و توضیحات تکمیلی در فایل kernels.txt در ضمیمه در اختیار شما قرار خواهد گرفت و **حتماً آنرا پیش از شروع پیاده سازی مطالعه کنید**.

نکته: به هیچ عنوان برای پیاده سازی کانوولوشن از کتابخانه های جانبی (مثل tensorflow, keras, openCV و ...) استفاده نکنید، در این سوال فقط استفاده از کتابخانه numpy و matplotlib در پایتون مجاز است. در صورت استفاده از متلب هیچ کتابخانه ای جانبی مجاز نیست.

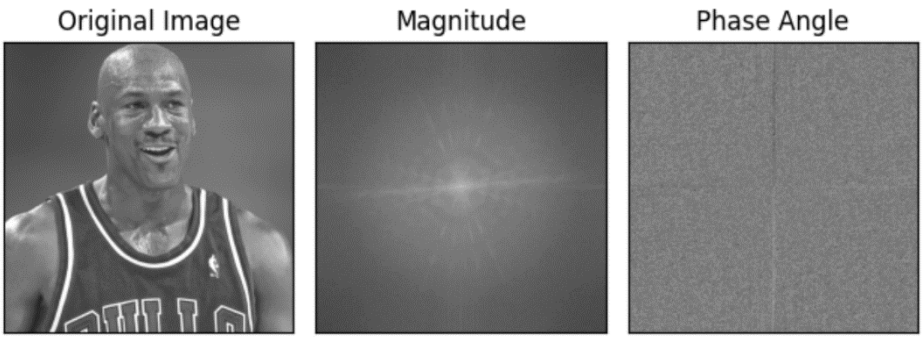
بخش سوم)

انجام این بخش برای گروه های دو نفره امتیازی و برای گروه های سه نفره بخشی از نمره کل و اجباری است

فیلترینگ تصاویر در حوزه‌ی فرکانس

پیش نیاز:

همانطور که می‌دانید تصاویر سیگنال‌های دو بعدی هستند و مانند سایر سیگنال ها می‌توان روی آنها تبدیل فوریه انجام داد. تبدیل فوریه در پردازش تصویر کاربردهای زیادی مثل بازسازی تصویر، فشرده‌سازی، بهبود تصاویر و ... دارد. در زیر یک تصویر را در کنار اندازه و فاز تبدیل فوریه آن مشاهده میکنید.



شرح تمرین:

- از هرکدام از تصاویر هر یک از feature map ها خروجی لایه های قسمت قبل تبدیل فوریه بگیرید و اندازه و فاز آنها را نمایش دهید. به عنوان مثال اگر در feature map خروجی ۴ تصویر وجود دارد، از هر چهار تصویر تبدیل فوریه بگیرید و فاز و اندازه هرکدام را plot کنید.
- هرکدام از تصاویر را با تبدیل فوریه آنها مقایسه کنید و نتایج را تحلیل کنید (بر اساس شکل تبدیل فوریه توضیح دهید که هر فیلتر چه کاری میکند)
- در حوزه ی فرکانس یک فیلتر lowpass و یک فیلتر highpass بسازید و روی تصاویر قسمت قبل اعمال کنید (در حوزه فرکانس) و خروجی را plot کنید. (انواع مختلفی از فیلتر highpass و lowpass در حوزه فرکانس وجود دارند که باید با بررسی آنها یکی را انتخاب کنید و پیاده سازی کنید)

نکات

- بارم بندی برای گروه های دو نفره (یا تکی): **۴۵** نمره بخش اول، **۵۰** نمره بخش دوم، **۵** نمره تمیزی کد، +**۲۵ نمره امتیازی (بخش سوم + خلاقیت)**

- بارم بندی برای گروه های سه نفره: **۳۵** نمره بخش اول، **۴۰** نمره بخش دوم، **۲۰** نمره بخش سوم، **۵** نمره تمیزی کد، +**۵ نمره امتیازی** برای خلاقیت احتمالی و کارهای اضافی

- پروژه تحویل خواهد داشت و لازم است در زمان تحویل همه ی اعضا بر **تمام** قسمت های پروژه تسلط داشته باشند.

- استفاده از زبان های **پایتون و متلب** مجاز است (فایل های قابل قبول: .py ، .ipynb و .m)
- اگر از jupyter notebook استفاده میکنید، حتماً هرکدام از سوالات را در نوت بوک جداگانه انجام دهید.
- گزارشی تهیه کنید و نتایج خود را در آن بیاورید. نیازی به توضیح کد ها نیست اما اگر جایی پارامتری استفاده کردید آنرا ذکر کنید و حتماً تصاویر خروجی ها را در گزارش قرار دهید. اگر از نوت بوک استفاده میکنید تهیه گزارش الزامی نیست ولی تصاویر خروجی ها باید در نوت بوک باشند.
- از کد ها و اسلاید های کلاس حل تمرین راهنمایی بگیرید و سوالات و ابهامات خود را در گروه مطرح کنید.

موفق باشید