پروژه

یادگیری عمیق - دانشکده مهندسی برق - زمستان ۱۴۰۱

پروژه درس یادگیری عمیق طراحی سیستم مولتی‌مودال برای تحلیل احساسات است. در این پروژه ابتدا با مجموعه دادگان این حوزه آشنا خواهید شد و سپس شروع به آموزش مدل‌هایی بر پایه شبکه عصبی برای تحلیل احساسات داده مولتی‌مودال (شامل متن, تصویر و صوت) خواهید کرد. مجموعه دادگان استفاده در این پروژه همگی به زبان انگلیسی هستند.

قواعد پروژه :

- راه ارتباطی با تیم پروژه تنها از طریق گروه درس در تلگرام و یا بخش پرسش و پاسخ کوئرا بوده و اعضای تیم پروژه به سوالات مستقیم پاسخ نخواهند داد.
- پروژه با احتساب فاز صفر مجموعا چهار فاز خواهد داشت و مجموعا ۶ روز تاخیر مجاز. پس از این مدت به ازای هر روز ۲ درصد از نمره بخش مربوطه از دست خواهد رفت. توجه بفرمایید در فاز چهارم امکان تاخیر وجود نداشته و پس از ددلاین این فاز، تحویل پروژه خواهید داشت.
- پس از ارسال کد هر فاز امکان ایجاد تغییر در کد خود برای فارهای بعدی پروژه را خواهید داشت اما ملاک ارزیابی هر فاز کد آپلود شده برای آن فاز می‌باشد نه کد ارائه شده در انتهای پروژه.
- آپلود پروژه از طریق کوئرا انجام می‌شود. برای راحتی دوستان در پروژه استفاده از GitHub اجباری نمی‌باشد اما توصیه اکید می‌شود به منظور مدیریت بهتر کار گروهی از ابزارهای مربوطه استفاده بفرمایید.

گروه‌بندی

با ورود به این لینک نام اعضای گروه خود را وارد کنید. گروه‌ها باید دو یا سه نفره باشند و امکان انجام پروژه به صورت انفرادی وجود ندارد. در صورتی که هیچ همگروهی ندارید می‌توانید نام خود را در داک وارد کرده تا در انتها با افراد تنهای دیگر گروه تشکیل شود. همچنین تفاوتی در ارزیابی گروه‌های دو یا سه نفره وجود نخواهد داشت.

فاز صفر

هدف اصلی این فاز آشنایی با مجموعه دادگان موجود برای تحلیل احساسات به زبان انگلیسی است. یکی از مجموعه دادگانی که قرار است با آن در طول پروژه کار شود به شرح زیر است:

- MSCTD: A Multimodal Sentiment Chat Translation Dataset
    - Github: https://github.com/XL2248/MSCTD
    - Paper: https://aclanthology.org/2022.acl-long.186/

برای فاز صفر استفاده از دیتاست اول - MSCTD - کافی است.

خروجی این فاز می‌بایست یک فایل ژوپیتر نوت‌بوک باشد که **نیازی به آپلود داده از سمت کاربر نباشد**. یعنی هرآنچه که نیاز است با اجرا شدن فایل نوت‌بوک از اینترنت دانلود کند و بقیه مراحل را پیش بگیرد.

توسعه یک Data Loader برای دیتاست MSCTD

بهتر است بخش خواندن داده پروژه شما جدا از بخش‌های دیگر طراحی شود. این موضوع علاوه بر اینکه به خوانایی کد کمک می‌کند, باعث می‌شود تغییرات به سرعت قابل پیاده‌سازی باشند. در این قسمت از شما خواسته شده است که یک کلاس Data Loader برای دیتاست MSCTD بسازید. سعی کنید پیاده‌سازی این کلاس را به گونه‌ای جامع انجام دهید که برای قسمت‌های بعدی پروژه تغییر دیتاست اهمیتی نداشته باشد. Data Loader شما می‌بایست به گونه‌ای دادگان را در اختیار استفاده کننده قرار دهد که جزئیات دیتاست از دید وی پنهان بماند و دیتاست را به عنوان یک ساختمان داده منظم در پایتون ببیند (و نه مجموعه‌ای از فایل‌ها و دایرکتوری‌های تودرتو).

برای انجام این کار بهتر است کلاسی بنویسید که **از یکی از** کلاس‌های زیر ارث‌بری کند:

1. *torch.utils.data.Dataset*[1]

کتابخانه pytorch دو کلاس اصلی (*torch.utils.data.DataLoader* و *torch.utils.data.Dataset*) برای لود کردن داده دارد که به کمک آن‌ها می‌توان دادگان از قبل لود شده - موجود در محتویات کتابخانه پایتورچ - و نیز دادگان دلخواه را لود کرد. این کلاس‌ها توابعی دارند که شما می‌توانید با override کردن آن‌ها دسترسی به دادگان را آسان کنید. سه تا از این توابع که پیاده‌سازی آن‌ها در کلاس فرزند این کلاس ضرورت دارند عبارت اند از:

- متد __*init*__: این متد زمان ساخت یک شی جدید از این کلاس فراخوانی می‌شود معمولا در کارهای اولیه‌ای که به محضی لود شدن دیتاست مورد انتظار است انجام می‌شود.
- متد __*len*__: در این متد باید اندازه دیتاست را برگردانید.
- متد __getitem__: این متد یک اندیس به عنوان ورودی می‌گیرد و شما می‌بایست نمونه متناظر با آن اندیس را برگردانید.

2. datasets.Dataset[2]

این کلاس از کتابخانه‌های متن‌بازشرکت Huggingface است که به تازگی معرفی شده است. از ویژگی‌های منحصر به فرد این کلاس این است که به جای اینکه دادگان را بر روی حافظه اصلی لود کند از مکانیزم (ساختار داده؟) arrow استفاده می‌کند که باعث می‌شود در صورت نیاز داده از روی حافظه خارجی خوانده شود. بدین وسیله این امکان را به شما می‌دهد که با دادگانی بسیار بزرگ‌تر از اندازه حافظه اصلی خود کار کنید.

در نهایت می‌بایست Data Loader خود را به گونه ای طراحی کنید که از دید کاربر با کدی شبیه به کد شکل زیر قابل استفاده باشد:

```
train_data = MSCTDDataset("path_to_dataset_train", random_seed=0, transform=ToTensor())
test_data = MSCTDDataset("path_to_dataset_test", transform=ToNumpy())
```

پی‌نوشت ۱: ممکن است نیاز داشته باشید به جای لودکردن بعضی از قسمت‌های دیتاست در هنگام ساخت یک شی جدید از کلاس تنها در هنگام نیاز دیتا مورد نظر را از دیتاست اصلی لود کنید.
پی‌نوشت ۲: در ساختن Data Loader خود خلاق باشید. ممکن است به مرور زمان در طول پروژه نیاز شود که تغییراتی به آن بدهید و ویژگی‌هایی را کم یا زیاد کنید.

---

[1] این قسمت برداشته شده از آموزش‌های pytorch برای کار با داده است. می‌توانید به صفحه اصلی بروید و مطالب را از آن بخوانید.

[2] https://huggingface.co/docs/datasets/v2.8.0/en/package_reference/main_classes#datasets.Dataset

تحلیل دیتاست - MSCTD

به کمک Data Loader خود داده MSCTD را بخوانید و موارد خواسته شده از دیتاست استخراج کنید. برای هر قسمت در صورت امکان تحلیل کنید که آماره مورد نظر چه تاثیری روی نتایج تحلیل احساسات در آینده خواهد داشت.

- توزیع پیشین کلاس‌های عواطف (هیستوگرام تعداد دادگان در هر دسته‌ی عواطف)
- هیستوگرام, میانگین و واریانس طول (تعداد کلمه) جملات
- هیستوگرام, میانگین و واریانس تعداد تصاویر در هر مکالمه
- هیستوگرام, میانگین و واریانس تعداد چهره‌های موجود در هر تصویر (برای این قسمت می‌توانید از کتابخانه‌های آماده استفاده کنید)
- کورولیشن هر کدام از احساسات در مکالمات با طول (تعداد کلمات) مکالمه (نمودار تعداد کلمات بر حسب دسته احساس را رسم کنید)
- هیستوگرام پترن‌های زمانی تغییر احساسات در یک مکالمه
    - یک پترن زمانی احساس از حذف نمونه‌های مشابه در یک مکالمه بدست می‌آید. مثلا اگر سری زمانی احساسات به صورت: "معمولی - معمولی - معمولی - خوشحال - ناراحت - ناراحت" باشد پترن زمانی آن به صورت "معمولی - خوشحال - ناراحت" خواهد بود.
- در آخر با فرض اینکه تمام مکالمات هم طول با طول میانگین مکالمه هستند تعداد پترن‌های احساسات ممکن به صورت تئوری را با تعداد کلاس‌های هیستوگرام آخرین بخش مقایسه کنید.