



مقدمه

در این تمرین، یک مدل چندکلاسه مبتنی بر PyTorch را روی مجموعه داده Fashion-MNIST طراحی و ارزیابی می‌کنید. هدف این دو بخش آن است که با فرآیند ساخت، آموزش و تحلیل شبکه‌های عصبی بصورت عملی و دقیق آشنا شوید.

طبقه‌بندی تصاویر

از دیتاست Fashion-MNIST استفاده می‌کنیم. این دیتاست شامل:

- ۶۰,۰۰۰ تصویر آموزشی
- ۱۰,۰۰۰ تصویر تست
- اندازه‌ی هر تصویر: 28×28
- تک‌کاناله (Grayscale)
- ۱۰ کلاس مختلف شامل: کفش، کیف، تی‌شرت، کت، بوت و...

۱_ بارگذاری دیتاست

دیتاست را از طریق یکی از روش‌های زیر بارگذاری کنید:

- دانلود دیتاست Fashion-MNIST با `torchvision.datasets.FashionMNIST` به شکل زیر:

```
from torchvision import datasets, transforms

# Download training and testing data
transform = transforms.Compose([transforms.ToTensor(),
                                transforms.Normalize((0.5,), (0.5,))])
train_ds = datasets.FashionMNIST('F_MNIST_data', download=True, train=True, transform=transform)
test_ds = datasets.FashionMNIST('F_MNIST_data', download=True, train=False, transform=transform)
```

- یا دانلود مستقیم از سایت MNIST و بارگذاری با NumPy ([لینک](#))

بررسی داده‌ها:

- چاپ شکل هر batch
- نمایش چند نمونه تصویر همراه با برچسب‌شان
- توضیح کوتاه درباره این‌که هر تصویر چه ابعادی دارد و چرا باید Flatten شود

۲- طراحی مدل شبکه عصبی

ساختار مدل (مدل پیشنهادی - پیاده سازی سایر مدل هایی که به $accuracy: 85\%$ یا بالاتر برسد قابل قبول است)

مدل چندلایه طبق ساختار زیر:

یک مدل چندلایه طبق ساختار زیر بسازید:

- مرحله Flatten: $1 \times 28 \times 28 \rightarrow 784$
- لایه Fully Connected اول: $784 \rightarrow 256$ با تابع فعال سازی ReLU
- لایه Fully Connected دوم: $256 \rightarrow 128$ با تابع فعال سازی ReLU
- لایه Fully Connected سوم (خروجی): $128 \rightarrow 10$ (بدون اعمال softmax در مدل، زیرا تابع loss از جمله CrossEntropyLoss خودش پیش پردازش لازم را انجام می‌دهد)

موارد موردنیاز

- مدل را به صورت یک کلاس مشتق از nn.Module پیاده سازی کنید.
- ساختار و تعداد پارامترها را چاپ کنید . `print(model)`
- توضیح مختصر درباره تعداد پارامترها و نقش لایه‌ها

۳- تعریف Loss و Optimizer

موارد موردنیاز

❖ استفاده از:

- Loss: `nn.CrossEntropyLoss()`
- Optimizer: `torch.optim.Adam(model.parameters(), lr=0.001)`
- ❖ توضیح بسیار کوتاه درباره دلیل انتخاب CrossEntropy برای طبقه‌بندی چندکلاسی

۴- آموزش مدل

موارد موردنیاز

- ❖ نوشتن حلقه آموزش کامل شامل:
 - قرار دادن مدل در حالت train
 - عبور دادن batch ها از مدل
 - محاسبه loss
 - backward
 - بروزرسانی وزن‌ها
- ❖ ذخیره loss دوره‌ای برای رسم نمودار
- ❖ چاپ دقت (accuracy) در هر epoch
- ❖ تعداد epoch پیشنهادی: حداقل ۱۰ تا

۵- ارزیابی مدل

موارد موردنیاز

- قرار دادن مدل در حالت eval
- محاسبه accuracy روی داده‌های تست
- نمایش چند نمونه پیش‌بینی شده همراه با برچسب صحیح
- بررسی چند نمونه اشتباه و تحلیل اینکه چه نوع لباس‌هایی سخت‌تر تشخیص داده می‌شوند

۶- رسم نمودارها

موارد موردنیاز

- رسم نمودار loss بر حسب epoch
- رسم نمودار accuracy بر حسب epoch
- تحلیل کوتاه اینکه مدل دچار overfitting شده یا نه

۷- بهبود مدل (اختیاری - نمره اضافه)

حداقل یکی از موارد زیر را انجام دهید:

- اضافه کردن Dropout و مقایسه دقت
- آزمایش BatchNorm
- افزایش/کاهش تعداد نرون‌ها و تحلیل تأثیر آن
- تغییر optimizer یا learning rate و مقایسه

نکات نهایی

۱. ددلاین تمرین تاریخ ۲۶ دی ۱۴۰۴ است.
 ۲. تمرین دارای ارائه ۱۰ دقیقه‌ای می‌باشد و باید آمادگی ارائه آن را داشته باشید.
 ۳. تمرین به صورت گروه‌های دو نفره که در تمرین اول مشخص می‌شود، قابل انجام است. حتما در گزارش کار نام اعضا و شماره دانشجویی نوشته شود؛ در غیر این صورت نمره برای اسامی نوشته‌نشده تعلق نمی‌گیرد.
 ۴. فایل کد و گزارش کار را در پوشه‌ای با نام زیر در سامانه آپلود کنید.
- CV-HW4-std#1-std#2
۵. نوشتن گزارش کار الزامی است و می‌تواند در Jupyter Notebook نیز نوشته شود.
 ۶. در صورت پیدا شدن موارد تقلب، نمره تمرین صفر می‌شود.
 ۷. شما مجاز به استفاده از کتابخانه‌های رایج پایتون هستید.