



دانشگاه صنعتی شریف

دانشکده مهندسی کامپیوتر

هوش مصنوعی

بهار ۱۴۰۰

استاد: محمدحسین رهبان

گردآورندگان: احمد سلیمی

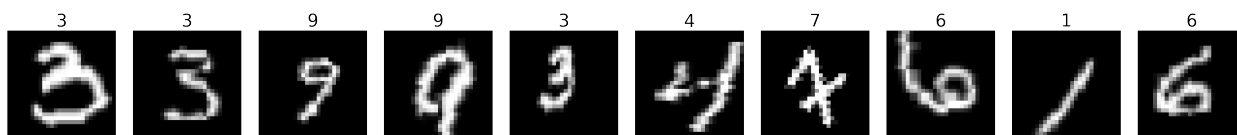
تمرین عملی چهارم

شبکه‌های عصبی

مهلت ارسال: ۲۳ خرداد ۱۴۰۰

تشخیص ارقام دست‌نویس (۴۰ + ۳۰ نمره)

در این تمرین، می‌خواهیم به کمک شبکه‌های عصبی مسئله classification تصویر ارقام دست‌نویس را حل کنیم. مجموعه داده‌های مورد استفاده در این مسئله، برگرفته از مجموعه داده‌های **MNIST** است که برای این تمرین، اندکی مورد پردازش قرار گرفته است. چند نمونه از تصاویر این مجموعه داده در شکل ۱ نشان داده شده‌است. این مجموعه داده شامل ۹۸۰۰۰ تصویر ۲۸ در ۲۸ برای آموزش شبکه (با برجسب) و ۱۰۰۰۰ تصویر (بدون برجسب) برای ارزیابی مدل و نمره‌دهی است.



شکل ۱: چند نمونه از تصاویر مجموعه داده مسئله

به همراه این فایل، یک فایل جویپتر نوت‌بوک نیز در اختیارتان قرار گرفته است که بایستی پیاده‌سازی و آموزش شبکه عصبی خود را در آن انجام دهید. آن چه تحویل خواهید داد، فایل نوت‌بوک مذکور و فایل prediction داده‌های ارزیابی است. فایل hw4_helper.py نیز حاوی توابع کمکی برای دریافت دیتاست و ذخیره prediction است که در ادامه نحوه استفاده از آن شرح داده شده‌است.

۱. (۱۰ + ۱۵ نمره)

دریافت داده‌ها

برای دریافت داده‌ها می‌توانید از توابع زیر استفاده کنید:

```
1 import hw4_helper
2 x_train, y_train = hw4_helper.get_train_data()
3 x_test = hw4_helper.get_test_data()
```

این توابع، برای اولین بار فایل دیتاست را دانلود کرده و در فولدر data_cache ذخیره می‌کنند. در دفعات بعدی چون فایل‌ها در آن مسیر موجود اند، تصاویر بدون دانلود مجدد از همان‌جا load می‌شوند. در صورت بروز هرگونه مشکل با توابع مذکور، می‌توانید فایل **x_train**، **y_train** و **x_test** را دانلود کرده و با استفاده از تابع `np.load('path/to/file')` آن‌ها را load کنید.

نکته مهم: برای آموزش شبکه خود فقط و فقط از **x_train** و **y_train** داده شده استفاده کنید. (به هیچ وجه از خود دیتاست MNIST استفاده نکنید.)

طراحی مدل

پیشنهاد می‌شود برای طراحی مدل خود، از یکی از کتابخانه‌های **PyTorch**، **Tensorflow** یا **Keras** استفاده کنید. مدل شما باید یک batch از تصاویر را که ابعاد آن $28 \times 28 \times B$ است را دریافت کرده و به ازای هر تصویر، یک بردار ۱۰ بعدی بعنوان خروجی برگرداند که هر عنصر آن، احتمال هریک از برچسب‌ها (ارقام ۰ تا ۹) باشد. (ابعاد خروجی باید $10 \times B$ باشد).

ساده‌ترین مدلی که می‌توانید طراحی کنید، فقط شامل لایه‌های **Fully Connected** است. در این حالت، باید تصاویر خود را بصورت یک آرایه یک‌بعدی (با $28 \times 28 = 784$ عنصر) به مدل بدهید. در مدل نیز کافیست بصورت پشت سر هم، لایه‌های خطی را با تابع فعال‌سازی **ReLU** قرار دهید. در آخرین لایه (که شامل ۱۰ نورون است) نیز باید از تابع فعال‌سازی **Softmax** استفاده کنید. این تابع مجموع خروجی‌های هر نمونه را برابر ۱ قرار می‌دهد تا بتوان به هر یک از عناصر آن بعنوان احتمال نگاه کرد.

با استفاده از شبکه‌های **Fully Connected** می‌توانید به دقت مورد انتظار برای دریافت نمره کامل در این مسئله برسید. اما برای دقت‌های بالاتر و دریافت نمره امتیازی، بهتر است از شبکه‌های عصبی کانولوشنی استفاده کنید. برای این کار، ابتدا توسط لایه‌های کانولوشنی دوبعدی و تابع فعال‌سازی **ReLU**، ابعاد تصویر را کاهش، و تعداد کانال‌های آن را افزایش دهید. (برای کاهش ابعاد تصاویر، هم می‌توانید از لایه‌ی کانولوشنی با **stride** برابر ۲ استفاده کنید، هم می‌توانید از لایه‌های **Pooling** مثل **Max Pooling** استفاده کنید.) پس از کاهش ابعاد تصاویر و افزایش کانال‌های آن به اندازه کافی، آن را بصورت یک‌بعدی در آورده و توسط یک شبکه **Fully Connected**، مشابه چیزی که بالاتر شرح داده شد خروجی را تولید کنید.

همچنین برای جلوگیری از **overfitting**، می‌توانید از لایه‌های **Dropout** و **Batch Normalization** در میان لایه‌های شبکه خود استفاده کنید. در **این لینک**، می‌توانید یک پیاده‌سازی بسیار ساده از یک شبکه عصبی کانولوشنی در کتابخانه **PyTorch** را مشاهده کنید. توجه کنید که استفاده از لایه‌های کانولوشنی و **Pooling** در مدل، نمره امتیازی دارد.

۲. (۱۰ نمره)

تابع هزینه و روش بهینه‌سازی

برای مسئله **classification** چند کلاسه، عموماً از تابع هزینه **cross-entropy** استفاده می‌شود. روش‌های بهینه‌سازی متنوعی نیز قابل استفاده اند، که از جمله آن‌ها می‌توان به **Adam** و **SGD** اشاره کرد.

آموزش شبکه

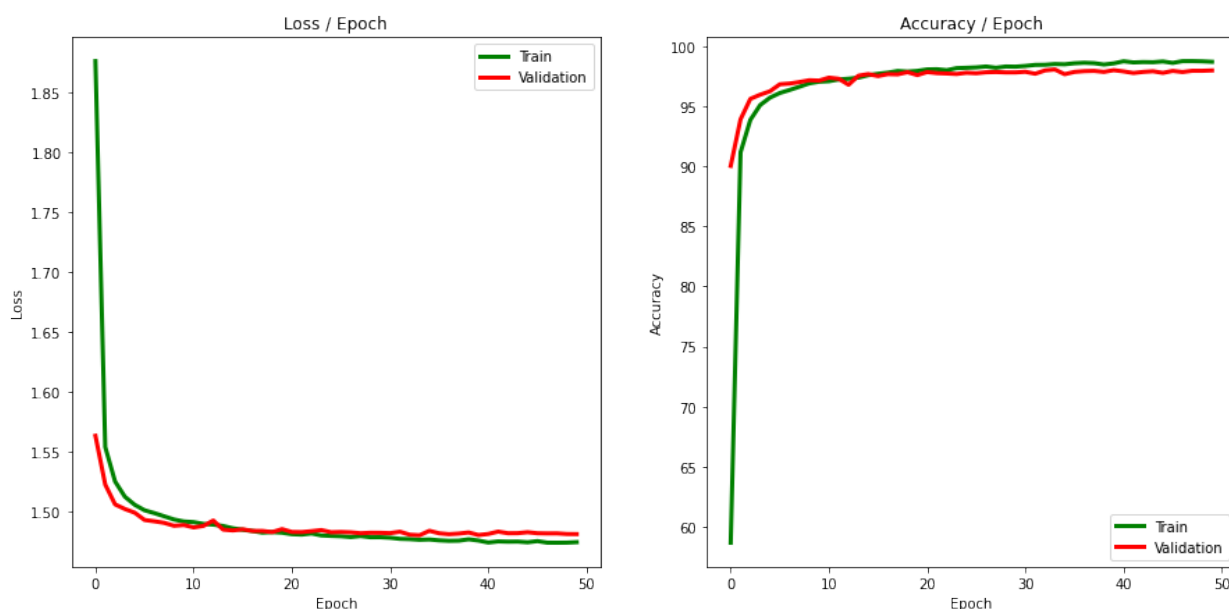
رای آموزش شبکه خود، بخش کوچکی از داده‌های آموزشی (مثلاً ۱۰ درصد) را برای اعتبارسنجی (**validation**) در نظر بگیرید. (می‌توانید از **این** تابع استفاده کنید.) سپس در هر **epoch**، داده‌های آموزشی را به تعدادی **batch** تقسیم کرده، و هر یک را به مدل دهید. با استفاده از تابع هزینه، هزینه خروجی محاسبه شده را بدست آورید و میانگین آن را بعنوان هزینه آن **epoch** در نظر بگیرید. سپس با استفاده از روش بهینه‌سازی، وزن‌های مدل خود را به‌روزرسانی کنید. علاوه بر محاسبه هزینه میانگین هر **epoch**، دقت (نسبت تعداد تشخیص‌های درست به کل تصاویر) را نیز محاسبه کرده و در یک لیست ذخیره کنید.

در هر **epoch** پس از آموزش شبکه، هزینه و دقت را روی داده‌های اعتبارسنجی نیز انجام دهید و نتایج را ذخیره کنید. توجه کنید که در اعتبارسنجی، به هیچ وجه از بهینه‌سازی استفاده نکنید.

۳. (۵ نمره)

رسم نمودار آموزش شبکه

مطابق شکل ۲، نمودارهای روند پیش‌روی هزینه و دقت مدل روی داده‌های آموزشی و اعتبارسنجی مدل را رسم کنید.

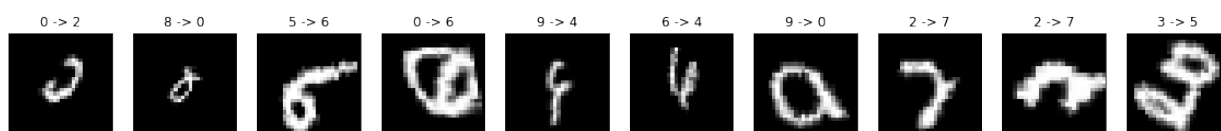


شکل ۲: نمودار آموزش شبکه

۴. (۵ نمره)

نمایش نمونه‌های اشتباه تشخیص داده شده

مطابق شکل ۳، ۱۰ نمونه از مجموعه داده‌های اعتبارسنجی خود را که مدل‌تان به اشتباه آن‌ها را تشخیص می‌دهد، به همراه برچسب واقعی و برچسبی که مدل به آن می‌زند، نمایش دهید.



شکل ۳: نمونه‌های اشتباه تشخیص داده شده

اجرای مدل روی داده‌های تست و ذخیره خروجی آن در فایل

در نهایت، مدل با بیشترین دقت روی داده‌های اعتبارسنجی را انتخاب کرده، و خروجی آن به ازای داده‌های تست بدست آورید. خروجی احتمال را با استفاده از تابع `argmax` به برچسب تبدیل کنید. (به ازای هر نمونه، یک عدد بین ۰ تا ۹ به آن نسبت دهید.) سپس بردار یک‌بعدی حاصل را با استفاده از تابع زیر ذخیره کنید.

```
hw4_helper.export_prediction(prediction)
```

این تابع، فایل `prediction.npy` را در کنار نوت‌بوک می‌سازد. این فایل را به همراه نوت‌بوک نهایی به فرمت `zip` در آورده و در کوئرا بارگذاری کنید. همچنین با اجرای تابع زیر، می‌توانید دقت تشخیص مدل‌تان روی داده تست را مشاهده کنید. (توجه کنید که این تابع، بردار تشخیص (`prediction`) شما را به یک API ارسال کرده و نتیجه را نمایش می‌دهد. برای جلوگیری از شلوغی سرور، در استفاده از این API صرفه‌جویی نمایید.)

```
hw4_helper.test(prediction)
```

نکته مهم: بدیهی است که صرف استفاده از تابع `test` به منزله نمره‌دهی به شما نیست و شما حتماً بایستی فایل `prediction.npy` که توسط تابع `export_prediction` تولید می‌شود را در فایل زیپ نهایی خود در کوئرا بارگذاری نمایید. نمره شما از این بخش، طبق فرمول زیر محاسبه خواهد شد. یعنی با دقت ۸۵ درصد، کل نمره اجباری (۱۰ نمره) این بخش را خواهید گرفت، و با افزایش دقت تا ۱۰۰ درصد، می‌توانید نمره این بخش را به ۲۵ برسانید.

$$0.0088x^2 - 0.63x$$