به نام خدا

دانشگاه تهران

پردیس دانشکده‌های فنی

دانشکده مهندسی برق و کامپیوتر

درس هوش مصنوعی

پروژه 5

طبقه‌بندی تصاویر
با شبکه عصبی چندلایه

Image Classification
with Multi-layer Neural Network

مهلت ارسال تا پایان خرداد ماه

طراحان پروژه: احمدحسین یزدانی، سعید عباسی

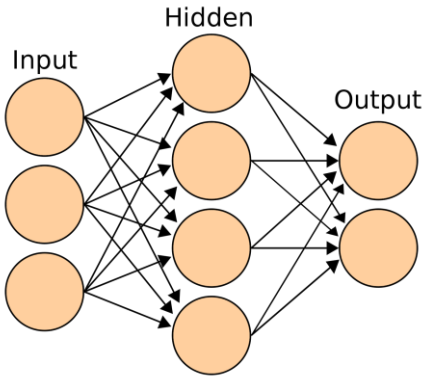
بهار ۹۹-۹۸

مقدمه

در این تمرین، شما باید با استفاده از شبکه‌های عصبی چند لایه، کار آموزش‌دهی ماشین[1] را برای طبقه‌بندی تصاویر[2] انجام دهید.

- ## شبکه های عصبی چند لایه

در شبکه های عصبی Feed Forward که در درس نیز با آن آشنا شده‌اید، هر تصویر ابتدا مسطح شده و به صورت بردار به عنوان ورودی شبکه داده می‌شود. هر درایه (پیکسل) این بردار یک ویژگی برای آن محسوب می‌شود. شبکه قرار است بر اساس این ویژگی‌ها و با ساختن ترکیبات غیر خطی از آن‌ها، وزن اتصالات بین لایه‌هایش را طوری تنظیم کند، که خروجی آن ضمن داشتن کمترین خطا، کلاس تصویر ورودی متناظر را به درستی پیش‌بینی کند.



پس از آخرین لایه تماما متصل معمولا یک لایه Softmax برای تولید احتمالات دسته‌بندی بردار نهایی در کلاس مربوطه استفاده می‌شود.

در این مرحله به ازای هر تصویر ورودی، یک بردار n_class تایی داریم که کلاس متناظر با بیشینه احتمال موجود در این بردار، همان تشخیص شبکه برای کلاس آن ورودی خاص است. (منظور از کلاس‌ها، label هایی است که تصاویر روی آن‌ها طبقه‌بندی می‌شوند.)

---

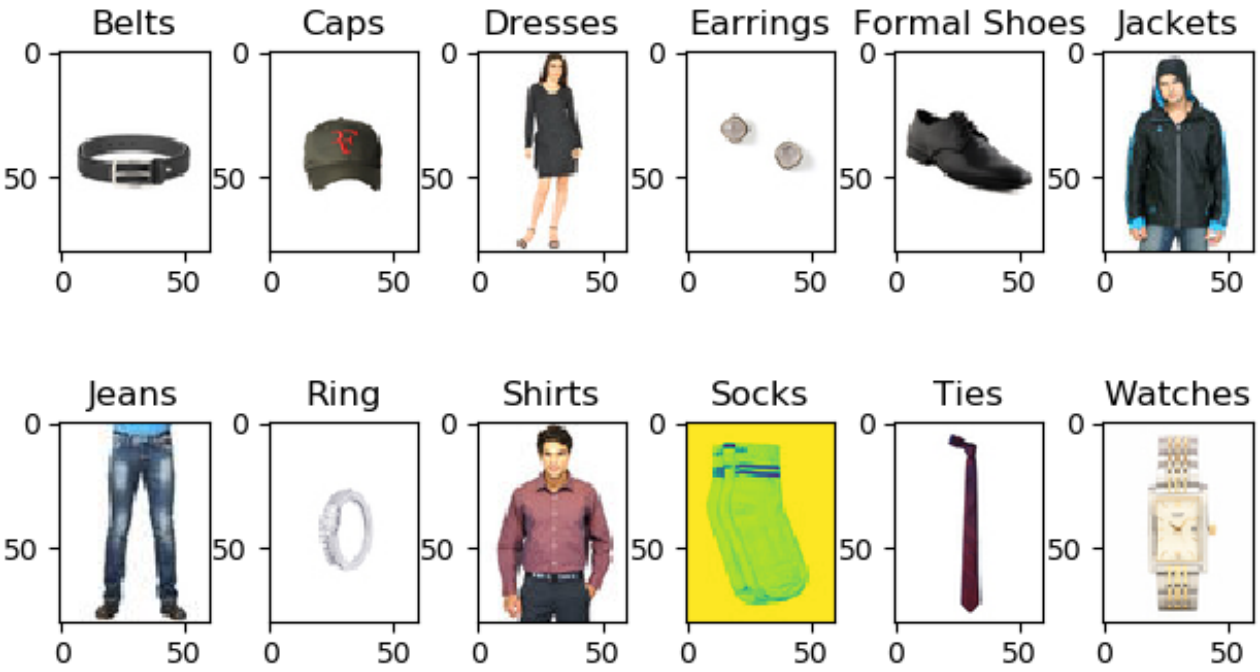
[1] Machine Learning

[2] Image Classification

تعریف مسأله

- **پیش بینی نوع محصول**

در این تمرین شما با یک مجموعه داده[3] شامل تصاویری از ۵۲ نوع محصول مختلف، کار خواهید کرد، که فایل زیپ‌شده آن در این لینک قرار دارد. این مجموعه داده شامل تعدادی فولدر به ازای هر نوع از محصول است که در آن تصاویر مرتبط با آن نوع محصول با نام های به فرمت id.jpg آمده اند که نوع آن محصول همان نام فولدر حاوی آن می باشد. (id تصاویر در سطح تمام محصولات مجموعه داده یکتا هستند)



نکات پیاده سازی

در ادامه شما باید از کتابخانه ی pytorch برای کار کردن با شبکه های عصبی استفاده کنید. بدین منظور یک شبکه Feed Forward متشکل از حداقل ۲ لایه پنهان طراحی کنید. (بنابراین شبکه ی شما باید با احتساب لایه ی ورودی و خروجی بدون در نظر گرفتن لایه softmax باید حداقل ۴ لایه داشته باشد.)

همچنین نت بوکی در این لینک، حاوی قطعه کدهای بارگیری دیتاست در اختیار شما قرار گرفته است. بنابراین **استفاده از jupyter notebook در این پروژه اجباری است.**

---

[3] Dataset

تجربه نشان داده که شبکه های عصبی می توانند به مواردی مثل رنگ، موقعیت شی و دوران حساس هستند و این موارد می تواند در عملکرد شبکه تاثیر بگذارد. در این پروژه برای سادگی، از سیاه‌وسفید شده تصاویر استفاده می‌کنیم. بنابراین در نت بوک داده شده ابتدا تمام تصاویر سیاه سفید شده و سپس به tensor تبدیل می شوند.

- **استفاده از Google Colab**

با توجه به حجم عملیاتی که در شبکه های عصبی انجام می شود که همه ی آنها می توانند در سطح داده موازی شوند (یعنی آن که می توان مثلا یک بردار را به بخش های مختلف تقسیم کرد و با استفاده از امکانات سخت افزار به طور موازی یک عمل مشخص روی این بخش ها انجام داد)، می توان یک شبکه ی عصبی بسیار بزرگ که اجرای آن روی CPU بسیار زمان بر است را روی GPU اجرا کرد که به تعداد زیادی (در حدود چند هزار تا) هسته ی اجرایی دارد و می تواند چند هزار عملیات که عملکرد یکسان ولی روی داده های مختلف اجرا می شوند را در مدت زمان بسیار کوتاه تری انجام دهد. اگر روی سیستم خود GPU ندارید، می توانید از سیستمی که Google برای پروژه های یادگیری ماشین تدارک دیده است، به نام Google Colab استفاده کنید که به صورت رایگان به شما GPU اختصاص می دهد. با استفاده از Google Colab شما می توانید پروژه ی خود را روی یک ماشین روی Google Cloud اجرا کنید. توصیه می شود که اندکی با آن آشنا شوید چرا که می تواند در زندگی حرفه ای شما (در صورتی که یادگیری ماشین را برای ادامه ی حرفه ی خود انتخاب می کنید) بسیار کمک حالتان باشد. می توانید در منابع ۱ تا ۳ با Google Colab آشنا شوید.

مواردی که باید پیاده سازی کنید

۱. در مجموعه داده اولیه، از ۱۲ نوع از محصولات، به دلخواه یک تصویر انتخاب کرده و (و آن‌ها را در ۲ یا ۳ ردیف) نمایش دهید. برای تصویر هر محصول، نوع (کلاس) آن را نیز به همراه تصویر نمایش دهید.

۲. **الف)** نمودار میله‌ای تعداد تصاویر هر نوع محصول را به صورت نزولی ترسیم کنید و نحوه توزیع تصاویر در انواع محصولات را ببینید.

ب) از هر نوع از محصول ۲۰ درصد تصاویر آن را (به صورت تصادفی) در مجموعه تست و باقی تصاویرش را در مجموعه آموزش قرار دهید تا مجموعه داده به دو مجموعه داده آموزش و تست تقسیم شود.

در تمام موارد پیاده‌سازی خواسته شده در قسمت‌های بعد، به طور پیش فرض:

- تابع هزینه ای که برای شبکه استفاده می کنید باید تابع Cross Entropy باشد.

- روش بهینه سازی شما باید روش Stochastic Gradient Descent[۴] باشد.
- batch_size شما باید ۶۴ باشد.
- تعداد دور[۵] های آموزش شما باید به طور پیش فرض ۱۰ دور باشد.
- learning rate ای که استفاده می کنید ۰.۰۱ باشد.
- activation function ای که در تمام لایه ها استفاده می کنید تابع relu باشد.
- مقداردهی اولیه وزن‌ها و بایاس‌های شبکه ی شما باید به صورت تصادفی انجام شود (می‌توانید فرض کنید pytorch به صورت پیش‌فرض وزن ها را به صورت تصادفی مقداردهی اولیه می کند).

در تمام موارد خواسته شده، باید نتایجتان را به صورت پیش‌فرض با ترکیب پارامترهای بالا بدست آورید و فقط پارامترهای خواسته شده در هر مرحله را تغییر دهید.

پس از هر مرحله آموزش شبکه، موارد زیر را به دست آورید:

- دقت مدل روی داده های آموزش و تست.
- رسم نمودار میله‌ای دقت مدل روی مجموعه داده‌های تست و آموزش به ازای هر کلاس در کنار هم.
- نمودار تغییرات میانگین loss داده های آموزش.

۳. در این قسمت شما باید یک مدل شبکه عصبی چند لایه برای فیت کردن تصاویر به آن طراحی کنید. با آزمون و خطا کردن، تعداد مناسب لایه ها و تعداد نورون های هر لایه را به دست آورید. (در pytorch نیازی به استفاده از sofmax در لایه ی آخر شبکه ندارید.)

الف) پیش از شروع آموزش تعداد پارامترهای قابل آموزش یا همان وزن‌های شبکه خود در هر لایه و در کل شبکه را با استفاده از دستورات pytorch به دست آورده و این عددها با توجه به ساختار شبکه خود را توجیه کنید.

ب) حال شبکه را روی مجموعه داده آموزش، با پارامترهای پیش‌فرض ذکر شده آموزش دهید.

۴. در این قسمت داده های خود را نرمال سازی کنید به این شکل که تمام مقادیر RGB پیکسل ها را بر ۲۵۵ تقسیم کنید. با این کار بازه ی تمام مقادیر RGB به بازه ی ۰ و ۱ انتقال داده میشود. سپس شبکه ای که در قسمت قبل گرفته اید را بار دیگر آموزش دهید. نتایج به دست آمده را با قسمت قبل مقایسه کرده و در گزارش ذکر کنید.

---

[4] Stochastic Gradient Descent

[5] epoch

برای تمام موارد خواسته شده بعدی، شبکه قسمت ۴ که داده‌های آن نرمال شده‌اند را معیار قرار دهید.

۵. همانطور که گفته شد، مقدار اولیه وزن‌ها در آموزش شبکه اهمیت دارد.

الف) در این مرحله مقدار اولیه تمام وزن‌های شبکه خود را برابر صفر قرار داده و دوباره شبکه را آموزش دهید. نتیجه را با قسمت قبل (که در آن وزن‌ها به صورت پیش‌فرض با مقدار تصادفی مقداردهی اولیه می‌شدند)، مقایسه کنید.

ب) به نظر شما چه راه حلی برای استفاده از بهترین ترکیب اولیه وزن‌ها، برای آموزش شبکه وجود دارد؟ فقط شرح دهید.

۶. یکی از پارامتر های مهم در آموزش دادن شبکه های عصبی، learning rate می باشد.

الف) حال با کم و زیاد کردن این پارامتر، شبکه را آموزش دهید و پس از یافتن مقدار بهینه برای شبکه خود، نتیجه را گزارش کنید.

ب) همچنین رفتار شبکه را برای learning rate بالاتر (مثلا ۰.۱) و پایین تر (مثلا ۰.۰۰۱) نسبت به حالت بهینه به دست آمده توجیه کنید. به نظر شما learning rate خیلی بالا یا خیلی پایین چه معایبی دارد؟ چرا؟

برای تمام قسمت‌های بعد، از learning rate بهینه‌ای که به دست آورده‌اید استفاده کنید.

۷. یکی دیگر از پارامتر های مهم یک شبکه ی عصبی، سایز batch می باشد.

الف) این بار batch size خود را مقادیر ۳۲ و ۱۲۸ قرار دهید و نتایج به دست آمده را در گزارش خود ثبت کنید. (مزایا و معایب احتمالی batch size بسیار کوچک و بسیار بزرگ را شرح دهید.)

ب) به نظر شما با افزایش batch size آیا باید learning rate را افزایش دهیم یا کاهش؟ برای این کار در حالتی که batch size را ۱۲۸ گذاشته اید learning rate را یک بار افزایش و یک بار کاهش دهید و نتیجه را تفسیر کنید.

۸. یکی دیگر از پارامتر هایی می تواند روی عملکرد شبکه تاثیر بگذارد، پارامتر momentum در optimizer است.

الف) عملکرد این پارامتر را به اختصار توضیح دهید و مزایا و معایب آن را به حالت بدون momentum شرح دهید.

ب) سپس بار دیگر شبکه ی به دست آمده از مورد ۵ را با momentum های ۰.۵، ۰.۹ و ۰.۹۸ آموزش دهید و نتایج را تفسیر کنید.

پ) آیا افزایش momentum لزوما باعث افزایش دقت و آموزش بهتر شبکه می شود؟

در موارد بعدی از مقدار momentum بهینه استفاده نمایید.

۹. تعداد **epoch** هایی که یک شبکه ی عصبی به آن اندازه آموزش داده می شود نیز در نتیجه نهایی موثر است.

الف) حال تعداد epoch هایی که شبکه را آموزش می دهید را به ۲۰ epoch تغییر دهید و نتایج را مقایسه و تفسیر کنید.

ب) به طور کلی شرح دهید که چرا شبکه را باید برای چند epoch (بیش از ۱ بار) آموزش داد؟

پ) آیا لزوما افزایش دادن تعداد epoch ها به دقت بیشتر منجر می شود؟ چرا؟

۱۰. در تمام مراحل، شما تاکنون به طور پیش فرض از تابع relu به عنوان **activation function** استفاده کرده اید. حال این تابع را با توابع tanh و leaky relu جایگزین کنید و نتایج را مقایسه کنید.

۱۱. یکی از تکنیک هایی که در شبکه های عصبی استفاده می شود و می تواند از overfit شدن جلوگیری کند **regularization** می باشد.

الف) توضیح دهید که افزودن ترم regularizer در تابع loss چگونه می تواند از overfit کردن شبکه جلوگیری کند؟

ب) در این قسمت باید از تکنیک weight decay برای regularize کردن شبکه تان (با اضافه کردن پارامتر weight_decay در SGD Optimizer) استفاده کنید. عملکرد این روش را توضیح دهید و بگویید چرا این کار معادل با اضافه کردن ترم regularizer در تابع loss است.

پ) سپس با استفاده از این روش مدل خود را بار دیگر آموزش دهید (پارامتر weight decay را برابر با ۰.۱ بگذارید) و نتایج را شرح دهید.

ت) قسمت قبل را بار دیگر، این بار با مقدار ۰.۰۱ انجام دهید و نتیجه را با قسمت قبل و همچنین نتیجه بهینه مورد ۸ مقایسه کنید. از این کار چه نتیجه ای می‌گیرید؟

منابع

[1] https://colab.research.google.com/notebooks/intro.ipynb

[2] https://zerowithdot.com/colab-github-workflow/

[3] https://colab.research.google.com/github/googlecolab/colabtools/blob/master/notebooks/colab-github-demo.ipynb

ملاحظات

- دقت داشته باشید که در این پروژه گزارش‌نویسی و تحلیل نتایج از سوی شما به مراتب بیشتر از کدی که نوشته‌اید اهمیت دارد.
- **نکته مهم:** از ابتدای نوشتن کد، توابع کلی برای آموزش شبکه، تست شبکه، رسم نمودار و ... را طوری پیاده سازی کنید که با گرفتن پارامترهای مورد نیاز، در قسمت‌های بعدی با یک فراخوانی ساده reuse شوند.
- **توصیه مهم:** بسیار توصیه می شود این پروژه را زود شروع کنید، چرا که یک تسک داده‌ای نسبتا کامل است و انجام دادن موارد خواسته شده ممکن است زمان زیادی از شما صرف کند. از همین رو حجم و نمره آن از پروژه‌های قبلی بیشتر است.
- استفاده از jupyter notebook برای نوشتن کد در این پروژه اجباری است.
- تمامی نتایج باید در یک فایل فشرده با عنوان AI-CA5-<#STID>.zip تحویل داده شود. این فایل باید شامل موارد زیر باشد:
  - گزارش پروژه با فرمت PDF و شامل شرح تمامی کارهای انجام شده، نتایج به دست آمده و تحلیل‌ها و بررسی‌های خواسته شده در صورت پروژه (در صورتی که گزارش خود را در notebook می نویسید به این کار نیازی نیست).
  - فایل notebook شامل کدهای خواسته شده. (حتما خروجی html فایل notebook خود را نیز در کنار notebook قرار دهید)

- توجه داشته باشید که علاوه بر ارسال فایل‌های پروژه، این پروژه به صورت غیرحضوری نیز تحویل گرفته خواهد شد. بنابراین تمام بخش‌های پروژه باید قابلیت اجرای مجدد در زمان تحویل را داشته باشند. همچنین درصورت عدم حضور در تحویل، نمره‌ای دریافت نخواهید کرد.
- هیچگونه شباهتی در انجام این پروژه بین افراد مختلف پذیرفته نمی‌شود. در صورت کشف هرگونه تقلب برای همه افراد متقلب نمره ۱۰۰- در نظر گرفته می‌شود.
- استفاده از مراجع با ارجاع به آنها بلامانع است. اما در صورتی که گزارش شما ترجمه عینی از آن‌ها باشد، یا از گزارش افراد دیگر استفاده کرده باشید کار شما تقلب محسوب می‌شود.
- در صورتی که سوالی در مورد پروژه داشتید بهتر است در فروم درس مطرح کنید تا بقیه از آن استفاده کننده، در غیر این صورت به طراحان پروژه ایمیل بزنید.

ahmad.yazdani@ut.ac.ir
saeed.abbc@gmail.com

موفق باشید!