بنام خدا

مریم رضوانی
شماره دانشجویی: ۹۹۲۱۱۶۰۰۱۹

گزارش پیاده سازی:

ابتدا تنسورفلو و کراس و کتابخانه‌هایی که برای ساخت مدل هستند را وارد می‌کنیم.
مجموعه داده Fashion MNIST را بارگیری و پیش پردازش می کند که شامل 60000 تصویر آموزشی و 10000 تصویر آزمایشی از 10 نوع لباس مختلف مانند پیراهن، شلوار، لباس و غیره است. تصاویر در مقیاس خاکستری هستند و اندازه آنها 28 در 28 پیکسل است.کد، مقادیر پیکسل را بین 0 و 1 تغییر می دهد و تصاویر را طوری شکل می‌دهد که یک کانال واحد داشته باشند. مدل شبکه عصبی را با استفاده از API متوالی TensorFlow Keras می‌سازد. این مدل از چندین لایه تشکیل شده است، مانند:
طبق تصویری که داده شده بود لایه ها را پیاده‌سازی کردیم، به این صورت که ورودی های ما ویژگی‌های دیتاست Fashion mnist بودند پس تصاویر با ابعاد ۲۸*۲۸ هستند.
این ورودی ها (inputlayer) را به لایه flatten می‌دهیمFlattenLayer و این لایه (Flatten) ورودی ما را به یک بردار تبدیل می‌کند یعنی:
یک بردار ۲۸*۲۸=۷۸۴
سپس لایه dense هرکدام از ۷۸۴ ورودی لایه قبل را می‌گیرد و طبق روش فولی کانکتد به ۱۶ نورون درخروجی این لایه بصورت خطی متصل می‌کند.
سپس batch normalizationرا روی داده‌ها اعمال می‌کنیم تا داده‌ها نرمالایز شوند. و روی خروجی این لایه تابع فعال ساز relu را اعمال می‌کنیم. و در نهایت تکنیک دراپ اوت(توضیح این روش در بخش سؤالات داده شد) را به صورت رندوم بین این لایه اعمال می‌کنیم(بطور رندوم ۲۰ درصد از نورون های لایه را حذف می‌کند.) و در گام بعدی یک لایه dense دیگر که هرکدام از خروجی‌های لایه‌ای که شامل دراپ اوت شده را می‌گیرد و به ۳۲ نورون خروجی متصل می‌کند را به شبکه اضافه کردیم و دوباره بچ نرمالیزیشن(batchnormalization1) را انجام دادیم و روی خروجی های این لایه تابع فعال ساز relu استفاده کردیم. (relu1) و به همین صورت از دراپ اوت(droupout1) بصورت رندوم(۲۰ درصد داده‌ها حذف) استفاده کردیم.و دوباره یک لایه dense دیگر که نورون های باقی‌مانده لایه قبلی که شامل دراپ اوت نشدند را به ۱۶ خروجی بصورت فولی کانکتد متصل می‌کند(dense2).خروجی هارا به لایه بچ نرمالیزیشن(batchnormalization2) می‌دهیم و تابع فعال ساز (relu2) را روی آن اعمال می‌کنیم و سپس بصورت رندوم ۲۰ درصد نورون‌ها را حذف (dropout)می‌کنیم و در نهایت خروجی این لایه را به ۱۰ نورون خروجی (dense3)که همان ۱۰ کلاس دیتاست است، متصل می‌کنیم.
به این ترتیب با این تیکه کد یک شبکه عصبی چندلایه(۱۰لایه) را ایجاد کردیم.
لایه خروجی که دارای 10 کلاس است و تابع softmax را روی خروجی لایه dense پنجم اعمال می‌کند. این لایه توزیع احتمال را روی 10 کلاس از اقلام لباس تولید می کند.

سپس این مدل را با استفاده از بهینه ساز Adam، تابع زیان sparse categorical crossentropy و متریک دقت(accuracy) جمع آوری و کامپایل می‌کند.
با استفاده از اندازه دسته ای 64 و تقسیم اعتبار 0.2، مدل را بر روی تصاویر و برچسب های آموزشی برای 5 دوره آموزش می دهد. به این معنی که 20 درصد از داده های آموزشی برای اعتبارسنجی و بقیه برای آموزش استفاده می شود. این مدل وزن‌ها و بایاس‌های بهینه را یاد می‌گیرد که تابع زیان را به حداقل می‌رساند و دقت داده‌های آموزشی را به حداکثر می‌رساند، در حالی که عملکرد داده‌های اعتبارسنجی را نیز بررسی می‌کند.
مدل را روی تصاویر و برچسب های آزمایشی ارزیابی می کند و دقت تست را که درصدی از تصاویر آزمایشی به درستی طبقه بندی شده است چاپ می کند.

در مرحله ۸ طبق کد دانشجویی خودم ۹۹۲۱۱۶۰۰۱۹ که رقم آخر آن ۹ است تصویر Ankle Boot را بعنوان ورودی به شبکه می‌دهیم و بررسی می‌کنیم که چقدر مدل درست پیش‌بینی می‌کند.
از کلاس ۹ یک نمونه تصویر انتخاب می‌کنیم و به مدل می‌دهیم تا پیش‌بینی

بخش سوالات:

1) Droupout چیست و چه کاربردی دارد؟Dropout یک تکنیک رگولاریزیشن(منظم سازی) برای کاهش اورفیت(بیش‌برازش) در آموزش شبکه عصبی است. دراپ اوت یعنی نادیده گرفتن یکسری از نورون‌ها بصورت تصادفی در فرایند آموزش.دراپ اوت یک هایپر پارامتر دارد که درصد نورون‌های دراپ شده را تعیین می‌کند. در هر بار فوروارد داده در شبکه، بصورت رندم یکسری نورون دراپ اوت می‌شوند. و باعث کاهش وابستگی(co-adaptation) بین نورون ها می‌شود در نتیجه از بیش برازش جلوگیری می کند. با دراپ اوت ممکن است زمان آموزش دو برابر شود اما شبکه مقاوم‌تری حاصل می‌شود. دراپ اوت در فرایند تست هیچ نورونی حذف نمی‌شود اما خروجی باید در احتمال دراپ اوت ضرب شود. عمل‌کرد دراپ اوت در فرایند آموزش و تست باهم فرق دارد که این یک معضل خطاخیز است.
چه زمانی از دراپ اوت استفاده کنیم؟ فولی کانکتدها حجم زیادی پارامتر به یک شبکه عصبی اضافه می‌کنند بنابراین اگر فولی کانکتدهای بزرگ در شبکه دارید از دراپ اوت استفاده می‌کنیم. دراپ اوت رابعد از تابع فعال‌ساز قرار می‌دهیم.

2) لایه batch normalization چیست و چه کاربردی دارد؟ متدی است که روی ورودی لایه شبکه عصبی اجرا می‌شود و ورودی‌های شبکه عصبی را نرمالیزه می‌کند. این روش باعث می‌شود که توزیع داده‌های ورودی به هرلایه ثابت باشد و از تغییرات ناخواسته در طول آموزش جلوگیری کند.باعث می‌شود تا شبکه عصبی سریع‌تر و پایدارتر یادبگیرد و از مشکلاتی مانند محو و انفجار گرادیان و بیش برازش جلوگیری کند. لایه batch normalization دارای چندین مزیت است که از جمله می‌توان به موارد زیر اشاره کرد:

- این لایه به آسانی قابل پیاده‌سازی و اضافه شدن به شبکه عصبی است.

- این لایه به شبکه عصبی اجازه می‌دهد که از نرخ یادگیری بالاتری استفاده کند و در نتیجه سرعت یادگیری را افزایش دهد.
- این لایه دارای اثر منظم سازی است و می‌تواند از بیش‌برازش جلوگیری کند. بنابراین، نیاز به استفاده از تکنیک‌های دیگر منظم‌سازی مانند حذف (dropout) کاهش می‌یابد.
- این لایه باعث می‌شود که شبکه عصبی کمتر به مقداردهی اولیه وزن‌ها و بایاس‌ها وابسته باشد و در نتیجه انتخاب پارامترهای اولیه ساده‌تر شود.
- این لایه می‌تواند به شبکه عصبی کمک کند که از توابع فعال‌سازی غیرخطی مختلف استفاده کند و از مشکلاتی مانند اشباع شدن یا ناپدید شدن گرادیان در توابع فعالسازی اشباع شونده مانند سیگموید یا تانژانت هذلولویی جلوگیری کند.

3) در لایه آخر از چه نوع تابع فعال سازی استفاده کردید؟ چرا؟soft-max؛ بدلیل اینکه ایده طراحی تابع Softmax از این تابع sigmoid گرفته شده است. همان‌طور که بیان شد، براساس خروجی تابع سیگموید که بین بازه 0 تا 1 است، از این تابع برای محاسبه احتمال استفاده می‌شود.با این حال، تابع فعالسازی سیگموئید یک مشکل اساسی دارد. چنانچه چندین خروجی به صورت 0.8، 0.9، 0.6، 0.7 و 0.8 تولید شوند، جمع تمامی خروجی‌ها بیشتر از عدد 1 می‌شود؛ در حالی که جمع مقادیر احتمالاتی باید عدد 1 باشد. این تابع مقادیر ورودی را به یک توزیع احتمال نگاشت می‌کند، بنابراین خروجی‌های تابع مجموع ۱ می‌شوند و می‌توانند به عنوان احتمالات کلاس تفسیر شوند. این کار موجب می‌شود که شبکه عصبی بتواند اطلاعات مربوط به هر کلاس را به صورت مستقل از سایر کلاس‌ها یاد بگیرد و در نتیجه قدرت تمایزدهی شبکه عصبی افزایش یابد.
تابع ریاضی فعالساز Softmax به صورت زیر است:

$$Softmax(z_i) = \frac{exp(z_i)}{\Sigma_j exp(z_j)}$$

تابع فعالسازی Softmax نوعی از تابع سیگموئید به حساب می‌آید. به عبارتی، تابع Softmax ترکیبی از چندین تابع سیگموئید است که احتمالات نسبی را محاسبه می‌کند. به عبارتی، تابع Softmax احتمالات چندین کلاس را مشخص می‌کند.

معمولاً، در تسک‌های «دسته‌بندی چندـ کلاسه» (Multi-Class Classification)، از تابع فعالسازی Softmax در لایه آخر شبکه عصبی استفاده می‌شود.

4) بهینه ساز Adamچیست؟ توضیح دهید. یک الگوریتم بهینه سازی است. این الگوریتم یک نسخه تعمیم یافته از گرادیان کاهشی تصادفی است که از تخمین گشتاور تطبیقی برای به‌روز رسانی پارامترهای شبکه استفاده می‌کند. این الگوریتم در سال 2015 توسط دیدریک کینگما و جیمی با معرفی شد و نام آن از عبارت Adaptive Moment Estimation گرفته شده است.
بهینه ساز آدام چندین مزیت دارد که از جمله می توان به موارد زیر اشاره کرد:
- این الگوریتم به آسانی قابل پیاده سازی و محاسبه است.

- این الگوریتم به حافظه کمی نیاز دارد. به لحاظ محاسباتی بهینه است.
- این الگوریتم برای مسائلی که داده ها یا پارامترها بزرگ هستند مناسب است.
- برای مسائلی که هدف یا گرادیان ناایستا یا نویزدار هستند مناسب است.
- این الگوریتم از نرخ یادگیری متغیر استفاده می کند که با اندازه گرادیان هماهنگ می‌شود.
- این الگوریتم از میانگین متحرک نمایی گرادیان ها استفاده می کند که باعث می شود توزیع گرادیان ثابت و یکنواخت باشد.

5) چرا دقت و مقدار زیان شبکه عصبی متفاوت شد؟

دقت accuracy، زیان loss در این پیاده‌سازی تغییر می‌کند چون مدل بصورت رندوم از بین ۶۴ دسته‌ای که ایجاد کردیم تصویر انتخاب می‌کند پس دقت و زیانی که برای آن‌ها محاسبه می‌شود در هر سری آموزش با یک اختلاف کمی تغییر می‌کند و ثابت نیست.(مدل در حال یادگیری داده ها و آموزش است و وزن و بایاس های خود را به‌روز می‌کند.مدل با مقایسه پیش بینی های خود با برچسب های واقعی و محاسبه خطا یاد می گیرد.)