تکلیف کامپیوتری پنجم - درس بازشناسی آماری الگو

نام دانشجو
شماره دانشجویی: ۴۰۳۱۳۰۰۴

۱۶ دی ۱۴۰۴

۱ مقدمه

در این تمرین، هدف طراحی، پیاده‌سازی و ارزیابی یک شبکه عصبی کانولوشنی (CNN) برای طبقه‌بندی مجموعه داده KMNIST است. برخلاف شبکه‌های تمام متصل (MLP) که با برداری کردن تصویر، اطلاعات مکانی پیکسل‌ها را از بین می‌برند، شبکه‌های CNN با حفظ ساختار دوبعدی تصویر و استفاده از فیلترها، ویژگی‌های بصری را به شکل مؤثرتری استخراج می‌کنند. در ادامه مراحل آماده‌سازی داده، طراحی معماری شبکه، و ارزیابی مدل با دو روش Hold-out و K-Fold Cross Validation تشریح می‌گردد.

۲ بارگذاری و پیش‌پردازش داده‌ها

برای اجرای این پروژه از فریم‌ورک PyTorch استفاده شده است. مجموعه داده KMNIST شامل تصاویر حروف ژاپنی کوزوشیجی است.

۱.۲ وضعیت مجموعه داده

طبق صورت تمرین، از کل مجموعه داده آموزشی شامل 60000 تصویر برای آموزش و 10000 تصویر برای تست استفاده شد. بررسی توازن کلاس‌ها نشان داد که داده‌ها کاملاً متوازن (Balanced) هستند:

- تعداد کل کلاس‌ها: 10 کلاس
- تعداد نمونه در هر کلاس (در داده‌های آموزش): دقیقاً 6000 نمونه

بنابراین نیازی به تکنیک‌های نمونه‌برداری جهت رفع عدم توازن وجود نداشت.

۲.۲ پیش‌پردازش

عملیات زیر روی تصاویر اعمال شد:

- **تبدیل به تنسور (To Tensor):** تبدیل داده‌ها به فرمت قابل پردازش و نوع داده Float.

- **نرمال‌سازی (Normalization):** مقادیر پیکسل‌ها با میانگین 0.5 و انحراف معیار 0.5 نرمال‌سازی شدند تا دامنه مقادیر به بازه $[-1, 1]$ منتقل شود. این عمل باعث همگرایی سریع‌تر گرادیان‌ها در طول آموزش می‌شود.

۳ طراحی معماری شبکه عصبی (CNN)

مدل پیشنهادی (KMNIST_CNN) شامل سه بلوک استخراج ویژگی و یک طبقه بندی‌کننده نهایی است. طراحی این معماری با هدف کاهش Overfitting و استخراج ویژگی‌های سلسله‌مراتبی انجام شده است. جزئیات لایه‌ها به شرح زیر است:

۱. **لایه‌های کانولوشن (Conv2d):** از سه لایه کانولوشن با کرنل سایز 3×3 و پدینگ 1 استفاده شد. تعداد فیلترها به ترتیب 32، 64 و 128 در نظر گرفته شد تا با عمیق‌تر شدن شبکه، ویژگی‌های پیچیده‌تری استخراج شود.

۲. **نرمال‌سازی دسته‌ای (Batch Normalization):** پس از هر لایه کانولوشن، از لایه BatchNorm2d استفاده شد. این لایه با نرمال کردن خروجی‌های هر لایه میانی، وابستگی به مقداردهی اولیه را کاهش داده و سرعت آموزش را افزایش می‌دهد.

۳. **تابع فعال‌ساز (Activation Function):** در تمامی لایه‌ها از تابع غیرخطی ReLU استفاده شده است.

۴. **لایه ادغام (Max Pooling):** جهت کاهش ابعاد مکانی و حجم محاسبات، از MaxPooling با ابعاد 2×2 بعد از هر بلوک کانولوشن استفاده شد.

۵. **طبقه‌بندی‌کننده (Classifier):** خروجی لایه‌های کانولوشن ابتدا Flatten شده و وارد یک شبکه تمام متصل می‌شود. جهت جلوگیری از بیش‌برازش، یک لایه Dropout با نرخ 0.5 قبل از لایه خروجی قرار داده شد.

۴ تنظیم فرآیند آموزش

پارامترهای مربوط به آموزش مدل به شرح زیر تنظیم شدند:

- **بهینه‌ساز (Optimizer):** الگوریتم Adam به دلیل تطبیق‌پذیری نرخ یادگیری انتخاب شد.
- **تابع هزینه (Loss Function):** تابع CrossEntropyLoss که مناسب مسائل طبقه‌بندی چندکلاسه است.
- **سخت‌افزار:** کلیه محاسبات بر روی GPU (در محیط Google Colab/Local CUDA) اجرا شد.

۵ ارزیابی مدل با روش Hold-out

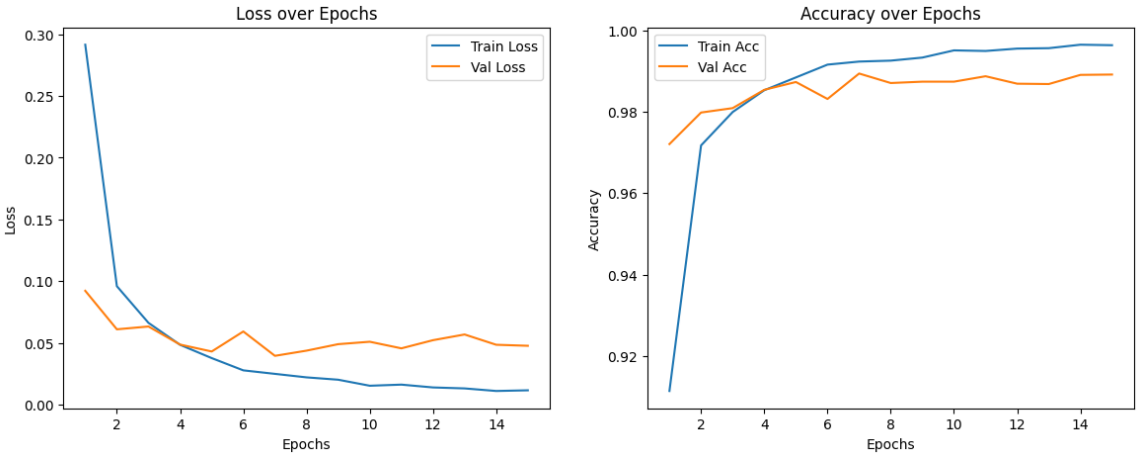
در این روش، داده‌های آموزشی به دو بخش تقسیم شدند:

- **داده‌های آموزش (Train):** 80% از داده‌ها (48000 نمونه).
- **داده‌های اعتبارسنجی (Validation):** 20% از داده‌ها (12000 نمونه).

مجموعه داده تست (10000 نمونه) کاملاً ایزوله نگه داشته شد. مدل با نرخ یادگیری 0.001 و اندازه دسته (Batch Size) برابر با 64 به مدت 15 دور (Epoch) آموزش دید.

۱۰۵ تحلیل نمودارهای آموزش

در شکل ۱ نمودارهای خطا (Loss) و دقت (Accuracy) نمایش داده شده است. همگرایی نمودارهای آموزش و اعتبارسنجی و فاصله اندک بین آن‌ها نشان می‌دهد که مدل دچار بیش‌برازش شدید نشده و به خوبی تعمیم یافته است.



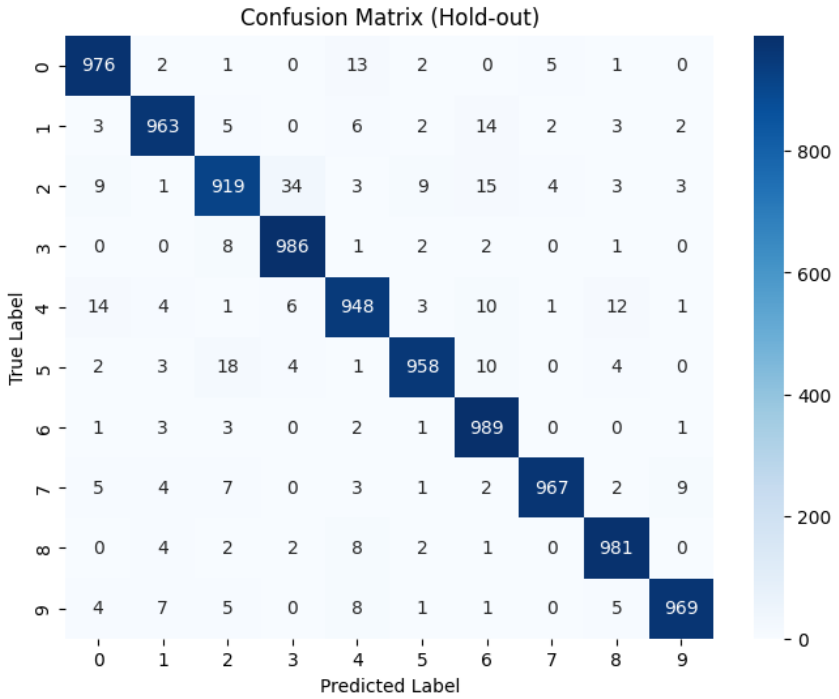
شکل ۱: روند تغییرات خطا و دقت در طول 15 دور آموزش (روش Hold-out)

۲۰۵ نتایج ارزیابی نهایی (Test Data)

پس از پایان آموزش، مدل روی داده‌های تست ارزیابی شد. نتایج کلی عبارتند از:

- **دقت نهایی (Accuracy):** برابر با 0.9656 (یا 96.56%)

ماتریس درهم‌ریختگی در شکل ۲ و جزئیات دقت به تفکیک هر کلاس در جدول ۱ آورده شده است.



شکل ۲: ماتریس درهم‌ریختگی مدل روی داده‌های تست (روش Hold-out)

جدول ۱: دقت مدل به تفکیک کلاس‌ها (Per-class Accuracy)

کلاس	دقت	کلاس	دقت
۰ (o)	0.976	۵ (ha)	0.958
۱ (ki)	0.963	۶ (ma)	0.989
۲ (su)	0.919	۷ (ya)	0.967
۳ (tsu)	0.986	۸ (re)	0.981
۴ (na)	0.948	۹ (wo)	0.969

مشاهده می‌شود که کلاس شماره 2 (su) کمترین دقت را داشته است که با نگاه به ماتریس درهم‌ریختگی مشخص می‌شود بیشترین تداخل را با کلاس‌های دیگر (به خصوص کلاس 3) داشته است.

۶ ارزیابی مدل با روش K-Fold Cross Validation

جهت یافتن بهترین هایپرپارامترها و ارزیابی دقیق‌تر، از روش 10-Fold Cross Validation استفاده شد. دو پیکربندی مختلف بررسی شد:

۱. LR=0.001 ، Batch Size=128 ← میانگین دقت: 98.69%

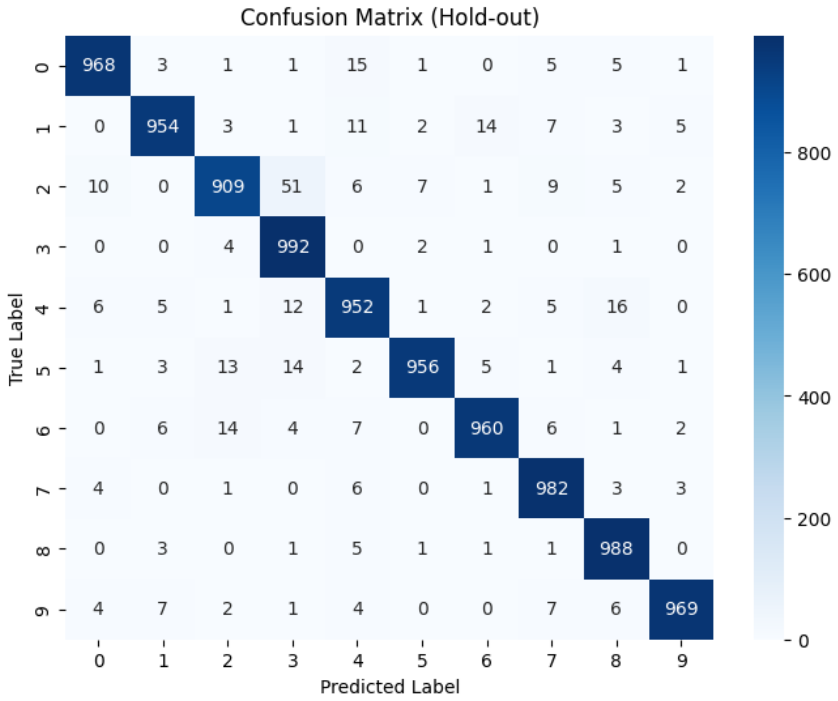
۲. LR=0.0005 ، Batch Size=64 ← میانگین دقت: 98.83%

۱۰۶ آموزش مدل نهایی

بر اساس نتایج بالا، پیکربندی دوم (نرخ یادگیری 0.0005 و اندازه دسته 64) به عنوان بهترین تنظیمات انتخاب شد. مدل نهایی با این تنظیمات روی **کل داده‌های آموزشی** (60000 نمونه) آموزش داده شد و سپس روی داده‌های تست ارزیابی گردید.

- **دقت نهایی تست (K-Fold Best Config):** برابر با 0.9630 (یا 96.30%)

ماتریس درهم‌ریختگی حاصل از این مدل در شکل ۳ نمایش داده شده است.



شکل ۳: ماتریس درهم‌ریختگی مدل بهینه شده با روش K-Fold

۷ نتیجه‌گیری و مقایسه

در این تکلیف، شبکه CNN با موفقیت پیاده‌سازی شد و در هر دو روش ارزیابی به دقتی بالاتر از 95% دست یافت که شرط نمره امتیازی را برآورده می‌کند.

جدول ۲: مقایسه نهایی عملکرد روش‌ها روی داده‌های تست

روش ارزیابی	دقت نهایی تست	توضیحات
Hold-out	0.9656	آموزش سریع‌تر، استفاده از ۸۰ درصد داده
K-Fold CV	0.9630	تنظیم دقیق‌تر پارامترها، استفاده از ۱۰۰ درصد داده

هرچند دقت روش Hold-out اندکی بالاتر به نظر می‌رسد، اما تفاوت ناچیز است (0.2%) و نشان می‌دهد مدل پیشنهادی پایداری خوبی دارد. استفاده از Dropout و Batch Normalization نقش کلیدی در دستیابی به این دقت بالا و جلوگیری از Overfitting داشته‌اند.