دانشگاه صنعتی امیرکبیر
(پلی تکنیک تهران)
دانشکده ریاضی و علوم کامپیوتر

گزارش تمرین پنجم درس داده‌کاوی محاسباتی

فشرده‌سازی مدل‌های عمیق با تجزیه رتبه پایین و ارتقای کیفیت صنعتی

نگارش
زهرا براتی

استاد درس
دکتر مهدی قطعی

تدریسیار
آقای بهنام یوسفی‌مهر

دی ۱۴۰۴

چکیده

در این پژوهش، فشرده‌سازی مدل‌های یادگیری عمیق با تمرکز بر شبکه‌ی ResNet-۱۸ مورد بررسی نظری و تجربی قرار گرفت. در گام نخست، با تحلیل طیفی وزن‌های شبکه و استفاده از تجزیه مقدار منفرد، ساختار طیفی یک لایه‌ی میانی و لایه‌ی نهایی تحلیل شد. نتایج این تحلیل نشان داد که بخش قابل‌توجهی از واریانس وزن‌ها توسط تعداد محدودی از مقادیر منفرد توضیح داده می‌شود که بیانگر وجود افزونگی در پارامترهای مدل و امکان استفاده از تقریب رتبه پایین است.

در ادامه، بر اساس این تحلیل، یک روش فشرده‌سازی عملی پیاده‌سازی شد که در آن لایه‌ی نهایی مدل با یک ساختار دولایه‌ای حاصل از تجزیه مقدار منفرد جایگزین شد. برای تمرکز بر اثر فشرده‌سازی و کاهش هزینه‌ی محاسباتی، بدنه‌ی اصلی شبکه ثابت نگه داشته شد و تنها لایه‌ی خروجی به‌صورت محدود تنظیم دقیق شد. عملکرد مدل پایه و نسخه‌های فشرده‌شده با نرخ‌های مختلف فشرده‌سازی از نظر دقت طبقه‌بندی و تعداد پارامترها مورد مقایسه قرار گرفت.

در بخش پایانی، اثر تنظیم دقیق چند ایپاکی بر بازیابی دقت مدل‌های فشرده‌شده بررسی شد و هم‌زمان سرعت استنتاج روی پردازنده‌ی مرکزی اندازه‌گیری شد. نتایج تجربی نشان داد که فشرده‌سازی شدید لایه‌ی نهایی منجر به افت قابل‌توجه دقت می‌شود و تنظیم دقیق کوتاه‌مدت قادر به جبران کامل این افت نیست. همچنین مشخص شد که فشرده‌سازی صرفاً لایه‌ی نهایی تأثیر محسوسی بر سرعت استنتاج کل شبکه ندارد، زیرا هزینه‌ی محاسباتی غالب در لایه‌های کانولوشنی متمرکز است. به‌طور کلی، این پژوهش نشان می‌دهد که اگرچه تحلیل طیفی ابزار مؤثری برای شناسایی افزونگی در مدل‌های عمیق است، اما دستیابی به مبادله‌ی مناسب میان دقت، اندازه‌ی مدل و سرعت نیازمند راهبردهای فشرده‌سازی و آموزشی دقیق‌تر است.

واژه‌های کلیدی:

فشرده‌سازی مدل، تجزیه مقدار منفرد، تقریب رتبه پایین، شبکه‌های عصبی عمیق

فهرست مطالب

صفحه

فهرست اشکال

صفحه

فهرست جداول

صفحه

فصل اول

مقدمه

مقدمه

در سال‌های اخیر، استفاده از شبکه‌های عصبی عمیق[۱] در مسائل مختلف یادگیری ماشین، پردازش زبان طبیعی و داده‌کاوی به‌طور چشمگیری گسترش یافته است. این شبکه‌ها با بهره‌گیری از معماری‌های عمیق و تعداد زیادی پارامتر، قادر به یادگیری نمایش‌های پیچیده و دستیابی به دقت‌های بالا هستند. بااین‌حال، افزایش عمق و اندازه‌ی مدل‌ها معمولاً با هزینه‌های محاسباتی بالا، مصرف حافظه‌ی زیاد و تأخیر در زمان استنتاج همراه است؛ مسائلی که استفاده از این مدل‌ها را در محیط‌های با منابع محدود با چالش مواجه می‌کند.

یکی از رویکردهای مؤثر برای کاهش این هزینه‌ها، فشرده‌سازی مدل[۲] است. هدف فشرده‌سازی، کاهش تعداد پارامترها و پیچیدگی محاسباتی مدل، با کمترین افت ممکن در دقت عملکرد است. در میان روش‌های مختلف فشرده‌سازی، رویکردهای مبتنی بر روش‌های ماتریسی و تحلیل خطی، به‌ویژه تجزیه مقدار منفرد[۳]، جایگاه ویژه‌ای دارند. این روش‌ها با تحلیل ساختار وزن‌های شبکه، امکان شناسایی افزونگی[۴] و استخراج مؤلفه‌های مؤثر را فراهم می‌کنند.

از منظر نظری، تجزیه مقدار منفرد ارتباط نزدیکی با تحلیل مؤلفه‌های اصلی[۵] دارد و ابزاری مناسب برای بررسی توزیع واریانس در ماتریس‌های وزن محسوب می‌شود. اگر بخش عمده‌ای از واریانس وزن‌ها توسط تعداد محدودی از مؤلفه‌ها توضیح داده شود، می‌توان از تقریب رتبه پایین[۶] به‌عنوان یک راهبرد فشرده‌سازی استفاده کرد. با این حال، انتقال این ایده‌ی نظری به پیاده‌سازی عملی در شبکه‌های عمیق، همواره ساده نیست و می‌تواند با افت دقت یا کاهش پایداری مدل همراه باشد.

---

[۱] Deep Neural Networks

[۲] Model Compression

[۳] Singular Value Decomposition (SVD)

[۴] Redundancy

[۵] Principal Component Analysis (PCA)

[۶] Low-rank Approximation

فصل دوم

تحلیل طیفی وزن‌های مدل

تحلیل طیفی وزن‌های مدل

هدف این فصل، بررسی ساختار طیفی وزن‌های یک شبکه عصبی عمیق از پیش‌آموزش‌دیده (ResNet-۱۸) به‌منظور تحلیل میزان افزونگی و کم‌رتبه‌بودن آن‌هاست. تحلیل طیفی این امکان را فراهم می‌کند که توزیع اطلاعات در وزن‌های لایه‌های مختلف مدل به‌صورت کمّی بررسی شده و ظرفیت بالقوه‌ی آن‌ها برای فشرده‌سازی ارزیابی گردد. در این راستا، یک لایه میانی و لایه نهایی مدل انتخاب و مورد مطالعه قرار گرفته‌اند.

۲-۱- مرور ادبیات

تجزیه مقدار منفرد روشی در جبر خطی است که ماتریس وزن $W_{m\times n}$ را به حاصل ضرب سه ماتریس $U\Sigma V^T$ تجزیه می‌کند. با انتخاب k مقدار منفرد بزرگتر، می توان تقریب رتبه پایینی از وزن‌ها ساخت که بخش عمده اطلاعات را حفظ می‌کند. این به معنای جایگزینی یک لایه با $m \times n$ پارامتر، با دو لایه متوالی با مجموع $k(m+n)$ پارامتر است.

همچنین می‌توان گفت ارتباط نزدیک SVD با تحلیل مؤلفه‌های اصلی نیز سبب شده است که واریانس مؤلفه‌ها به‌صورت مربع مقادیر منفرد تفسیر شود. بر این اساس، نمودار واریانس تجمعی ابزاری رایج برای تعیین تعداد مؤلفه‌های لازم جهت حفظ درصد معینی از اطلاعات (برای مثال ۹۵ درصد) محسوب می‌شود. این ایده، مبنای روش‌های تقریب رتبه پایین و فشرده‌سازی مبتنی بر SVD در مدل‌های یادگیری عمیق است. [۱]

۲-۲- روش انجام پیاده‌سازی

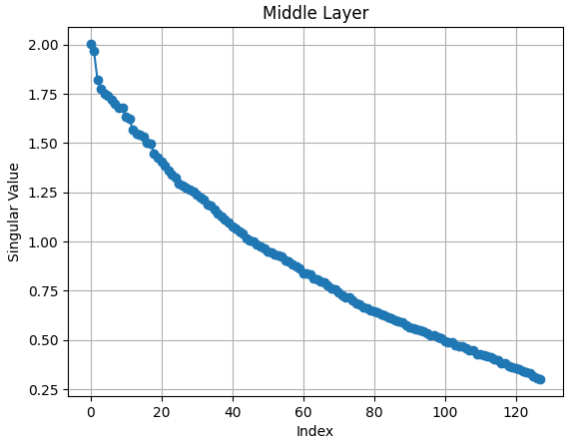
ابتدا یک مدل ResNet-۱۸ از پیش‌آموزش‌دیده بارگذاری و در حالت ارزیابی قرار داده شد. سپس وزن‌های دو لایه‌ی منتخب استخراج شد:

- لایه میانی: لایه‌ی layer۲[۰].conv۱ با ابعاد وزن (۱۲۸,۶۴,۳,۳)
- لایه نهایی: لایه‌ی کاملاً متصل (fc) با ابعاد (۵۱۲، ۱۰۰۰)

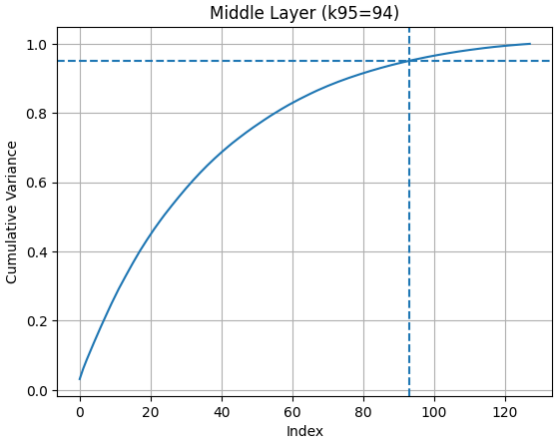
از آن‌جا که اعمال SVD مستلزم وجود یک ماتریس دوبعدی است، وزن‌های لایه کانولوشنی پس از بازآرایی[۷] به ماتریسی با ابعاد (۵۷۶، ۱۲۸) تبدیل شدند. سپس برای هر ماتریس، مقادیر منفرد محاسبه شده و واریانس متناظر با هر مؤلفه به‌صورت مربع مقدار منفرد در نظر گرفته شد. با نرمال‌سازی واریانس تجمعی، کوچک‌ترین تعداد مؤلفه‌ای که منجر به حفظ حداقل ٪۹۵ از واریانس کل می‌شود، به‌عنوان k_{95} استخراج شد.

۲-۳- نتایج عددی

در لایه‌ی میانی تعداد کل مقادیر منفرد برابر ۱۲۸، تعداد مؤلفه‌های لازم برای حفظ ٪۹۵ واریانس برابر ۹۴ و معادل ٪۷۳.۴۴ از مقادیر است. همچنین در لایه نهایی این مقادیر به ترتیب برابر ۵۱۲، ۲۹۱ و ٪۵۶.۸۴ است.

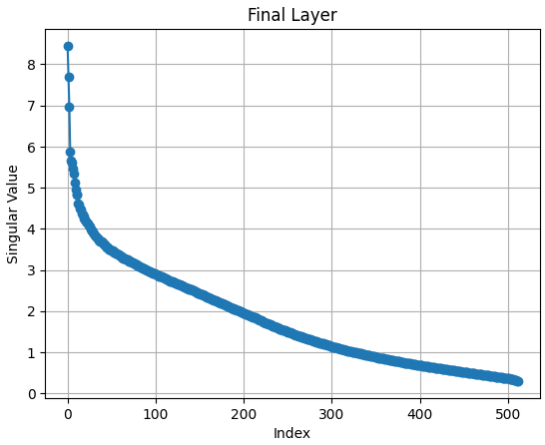


شکل ۲-۱ : نمودار پراکندگی مقادیر منفرد لایه‌ی میانی

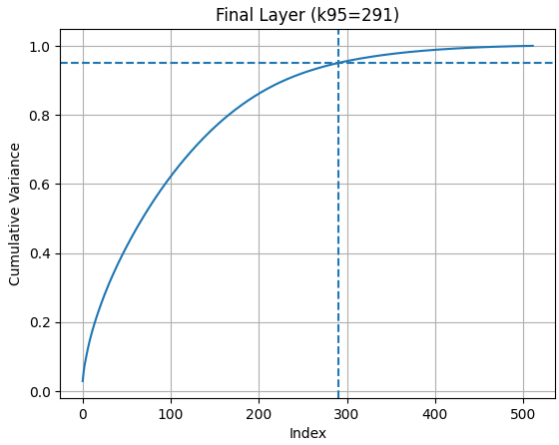


شکل ۲-۲ : نمودار واریانس تجمعی مقادیر منفرد لایه‌ی میانی

[۷] reshape



شکل ۲-۳ : نمودار پراکندگی مقادیر منفرد لایه‌ی نهایی



شکل ۲-۴ : نمودار واریانس تجمعی مقادیر منفرد لایه‌ی نهایی

همچنین نمودارهای مقادیر منفرد (شکل ۲-۱ و شکل ۲-۳) نشان می‌دهند که افت اولیه‌ی سریعی در مقدار مؤلفه‌ها وجود دارد که این موضوع بیانگر تمرکز اطلاعات در تعداد محدودی از ابعاد مؤثر است. همچنین، نمودارهای واریانس تجمعی (شکل ۲-۲ و شکل ۲-۴) تأیید می‌کنند که بخش قابل‌توجهی از اطلاعات وزن‌ها را می‌توان با تعداد کمتری از مؤلفه‌ها حفظ کرد.

۲-۴- تحلیل

نتایج به‌دست‌آمده نشان می‌دهد که لایه‌ی نهایی مدل، نسبت به لایه‌ی میانی، از ساختار طیفی فشرده‌تری برخوردار است. به بیان دیگر، درصد کمتری از مؤلفه‌های طیفی در لایه‌ی fc برای حفظ بخش عمده‌ای از واریانس کافی است. این موضوع حاکی از وجود افزونگی بالاتر در وزن‌های لایه‌ی نهایی بوده و نشان می‌دهد که این لایه، نامزد مناسبی برای اعمال فشرده‌سازی رتبه پایین است.

در مقابل، لایه‌ی میانی برای دستیابی به همان سطح حفظ واریانس، نیازمند تعداد بیشتری از مؤلفه‌هاست که می‌تواند بیانگر توزیع یکنواخت‌تر اطلاعات در جهات مختلف فضای ویژگی باشد. این تفاوت رفتاری میان لایه‌ها با نقش آن‌ها در شبکه نیز سازگار است؛ لایه‌های میانی مسئول استخراج نمایش‌های غنی‌تر و متنوع‌تری از داده هستند، درحالی‌که لایه‌ی نهایی بیشتر نقش نگاشت این نمایش‌ها به فضای خروجی را بر عهده دارد.

۲-۵- جمع‌بندی

در این فصل، با استفاده از تحلیل طیفی مبتنی بر SVD، ساختار وزن‌های یک مدل ResNet-۱۸ از پیش‌آموزش‌دیده مورد بررسی قرار گرفت. نتایج نشان داد که:

- لایه‌ی میانی برای حفظ ٪۹۵ واریانس به حدود ٪۷۳ از مؤلفه‌های طیفی خود نیاز دارد.
- لایه‌ی نهایی همان سطح از واریانس را تنها با حدود ٪۵۷ از مؤلفه‌ها حفظ می‌کند.

این مشاهدات به‌صورت کمّی تأیید می‌کنند که لایه‌ی نهایی از پتانسیل بالاتری برای فشرده‌سازی رتبه پایین برخوردار است.

فصل سوم

پیاده‌سازی عملی فشرده‌سازی

پیاده‌سازی عملی فشرده‌سازی

در این فصل، یک روش فشرده‌سازی مبتنی بر تقریب رتبه پایین وزن‌های لایه‌ی نهایی شبکه‌ی ResNet-۱۸ پیاده‌سازی شد. ایده‌ی اصلی این است که به‌جای یک لایه‌ی خطی بزرگ، آن را با دو لایه‌ی خطی متوالی جایگزین کنیم که حاصل تجزیه مقدار منفرد و نگه‌داشتن تنها مؤلفه‌های مهم‌تر است.

۳-۱- مرور ادبیات

در روش‌های مبتنی بر SVD، یک ماتریس را می‌توان به مؤلفه‌هایی تفکیک کرد که به‌صورت مرتب‌شده، «جهت‌های غالب تغییرات داده» را ارائه می‌دهند؛ به همین دلیل، نگه‌داشتن تعداد محدودی از مؤلفه‌ها معمولاً بخش مهمی از اطلاعات را حفظ می‌کند .همچنین ارتباط SVD و PCA از این جهت مهم است که واریانس نمونه‌ای مؤلفه‌ها به مقادیر منفرد وابسته است و این وابستگی مبنای تصمیم‌گیری برای کاهش‌بعد/کم‌رتبه‌سازی می‌شود.[۲]

۳-۲- روش انجام پیاده‌سازی

لایه‌ی نهایی (fc) یک نگاشت خطی از ۵۱۲ ویژگی به ۱۰ کلاس دارد. وزن‌های این لایه یک ماتریس W هستند. در پیاده‌سازی، W به کمک SVD به صورت $W = U\Sigma V^T$ تجزیه می‌شود و سپس به‌جای W، تقریب رتبه پایین آن ساخته می‌شود؛ یعنی فقط k مؤلفه‌ی اول نگه داشته شده و نتیجه به شکل دو لایه پیاده‌سازی می‌شود:

- لایه‌ی اول: از بعد ۵۱۲ به k (بدون بایاس)
- لایه‌ی دوم: از k به ۱۰ (با بایاس)

این کار دقیقاً همان ایده‌ی «نگه‌داشتن جهت‌های غالب تغییرات» است که SVD به‌صورت مرتب‌شده در اختیار می‌گذارد.

در این فصل، دو حالت فشرده‌سازی ٪۵۰ و ٪۸۰ تعریف شد. هدف این بخش، مقایسه‌ی اثر فشرده‌سازی

در ساده‌ترین حالت است. برای اینکه زمان آموزش کم شود و اثر فشرده‌سازی روی همان بخشِ جایگزین‌شده قابل مشاهده باشد، تمام لایه‌های بدنه‌ی اصلی شبکه[۸] ثابت[۹] شدند و فقط سرِ دسته‌بند یعنی fc اجازه‌ی یادگیری داشت. این تصمیم باعث می‌شود تغییرات دقت عمدتاً ناشی از کیفیت fc و فشرده‌سازی آن، نه بازآموزی کامل شبکه باشد.

از آن‌جا که مدل ResNet-۱۸ مورد استفاده، از روی دیتاست ImageNet آموزش دیده است، نرمال‌سازی ورودی‌ها با استفاده از میانگین و انحراف معیار همان دیتاست انجام شده است. این مقادیر برابر با [۰.۴۰۶, ۰.۴۵۶, ۰.۴۸۵] برای میانگین و [۰.۲۲۵, ۰.۲۲۴, ۰.۲۲۹] برای انحراف معیار هستند و استفاده از آن‌ها باعث می‌شود توزیع آماری داده‌های ورودی با توزیعی که مدل در مرحله‌ی پیش‌آموزش مشاهده کرده است، هم‌راستا باقی بماند.

درعین‌حال، اندازه‌ی ورودی تصاویر نسبت به تنظیمات پیش‌فرض ImageNet کاهش داده شد. درحالی‌که تنظیمات استاندارد شامل تغییر اندازه به ۲۵۶ و برش مرکزی ۲۲۴ پیکسل است، در این آزمایش تصاویر به اندازه‌ی ۱۲۸ تغییر اندازه داده شده و برش مرکزی ۱۲۸ اعمال شد. این انتخاب با هدف کاهش هزینه‌ی محاسباتی، تسهیل اجرای آزمایش روی پردازنده‌ی مرکزی "CPU" و افزایش سرعت اجرای کل فرآیند انجام شده است. هرچند این کاهش اندازه می‌تواند منجر به افت جزئی دقت شود، اما برای مقایسه‌ی نسبی اثر فشرده‌سازی، مناسب است.

برای آموزش، از دیتاست CIFAR-۱۰ استفاده شد، اما نه به‌منظور آموزش کامل شبکه. بدنه‌ی اصلی مدل به‌طور کامل ثابت نگه داشته شد و تنها لایه‌ی نهایی (fc) اجازه‌ی یادگیری داشت. فرآیند تنظیم دقیق[۱۰] نیز به‌صورت بسیار محدود و تنها روی ۱۰ بچ انجام شد. این کار به‌منظور ایجاد حداقل سازگاری بین لایه‌ی خروجی و توزیع داده‌های CIFAR-۱۰ انجام شده و هدف آن، صرفاً فراهم کردن یک مبنای منصفانه برای مقایسه‌ی مدل پایه و نسخه‌های فشرده‌شده بوده است، نه رسیدن به بهترین عملکرد ممکن.

---

[۸] backbone

[۹] freeze

[۱۰] Fine-tuning

همچنین، برای کاهش زمان اجرا و امکان تکرار سریع آزمایش‌ها، از زیرمجموعه‌های کوچک دیتاست استفاده شد؛ به‌طوری که آموزش روی ۱۰۲۴ نمونه و ارزیابی روی ۵۱۲ نمونه انجام گرفت. اگرچه این حجم داده نماینده‌ی کامل عملکرد مدل در مقیاس بزرگ نیست، اما برای تحلیل نسبی تأثیر فشرده‌سازی‌های مختلف (بدون فشرده‌سازی، فشرده‌سازی متوسط و فشرده‌سازی شدید) کفایت می‌کند.

در مجموع، این تنظیمات به‌گونه‌ای انتخاب شده‌اند که ضمن حفظ منطق علمی آزمایش، زمان اجرا کاهش یافته و تمرکز اصلی گزارش بر رفتار فشرده‌سازی و اثر آن بر دقت مدل باقی بماند.

۳-۳- نتایج عددی

پس از یک مرحله تنظیم دقیق محدود روی fc و سپس فشرده‌سازیfc داریم:

جدول ۳-۱ : نتایج فشرده‌سازی

حالت مدل	تعداد پارامتر	دقت
پایه (بدون فشرده‌سازی)	۱۱,۱۸۱,۶۴۲	۵۰.۷۸٪
فشرده‌سازی با keep ratio = ۰.۵	۱۱,۱۷۹,۱۳۲	۳۶.۷۲٪
فشرده‌سازی با keep ratio = ۰.۲	۱۱,۱۷۷,۵۶۶	۱۷.۵۸٪

طبق نتایج جدول ۳-۱ می‌توان گفت هرچه رتبه‌ی مؤثر کوچک‌تر شده، دقت افت کرده است. این رفتار با منطق تقریب رتبه پایین سازگار است: با حذف مؤلفه‌های بیشتر، ظرفیت نمایش لایه کاهش می‌یابد و برای مسئله‌ی طبقه‌بندی، خطای بیشتری ایجاد می‌شود.[۳]

۴-۳- جمع‌بندی

در این فصل، فشرده‌سازی لایه‌ی خطی نهایی با استفاده از تقریب رتبه پایین مبتنی بر SVD انجام شد و نشان داده شد که:

۱. جایگزینی fc با دو لایه‌ی کوچک‌تر، یک پیاده‌سازی مستقیم از مفهوم تقریب رتبه پایین است که SVD جهت‌های غالب را به‌ترتیب اهمیت ارائه می‌کند.

۲. برای سرعت، بدنه‌ی اصلی شبکه ثابت شد و فقط fc با تعداد محدودی بچ آموزش دید (۱۰ بچ) تا هم سازگاری حداقلی با CIFAR-۱۰ ایجاد شود و هم زمان اجرا پایین بماند.
۳. نرمال‌سازی ImageNet حفظ شد، اما اندازه‌ی ورودی از ۲۲۴ به ۱۲۸ کاهش یافت تا هزینه‌ی محاسباتی کم شود.
٤. نتایج نشان داد فشرده‌سازی شدیدتر، افت دقت بیشتری ایجاد می‌کند؛ بنابراین انتخاب نسبت فشرده‌سازی باید با توجه به محدودیت محاسباتی و افت دقت قابل‌قبول انجام شود.

فصل چهارم

بازیابی دقت و ارزیابی سرعت

بازیابی دقت و ارزیابی سرعت

هدف این بخش، بررسی این موضوع است که آیا می‌توان پس از فشرده‌سازی مبتنی بر تقریب رتبه پایین، با انجام تنظیم دقیق بخشی از افت دقت را جبران کرد یا خیر. علاوه‌برآن، اثر فشرده‌سازی بر سرعت استنتاج[۱۱] روی پردازنده مرکزی CPU اندازه‌گیری و با مدل پایه مقایسه می‌شود. در نهایت، رابطه‌ی بین نرخ فشرده‌سازی و دقت با رسم نمودار نمایش داده می‌شود.

۴-۱- روش انجام پیاده‌سازی

در فصل سوم، لایه‌ی نهایی مدل با ساختار دولایه‌ای جایگزین شد که حاصل تجزیه مقدار منفرد بود. در این فصل، دو مدل فشرده‌شده (با نرخ‌های ٪۵۰ و ٪۸۰) برای ۳ ایپاک[۱۲] تنظیم دقیق شدند:

نکته‌ی مهم این است که در این مرحله، برخلاف فصل قبل که برای سرعت تنها ۱۰ بچ آموزش داده شده بود، max_batches=None قرار داده شد؛ یعنی تنظیم دقیق روی کل داده‌های موجود در بارگذار آموزشی انجام گرفته است. بنابراین، از نظر طراحی آزمایش، این فصل تلاش می‌کند یک فرصت واقعی‌تر برای بازیابی دقت در مدل‌های فشرده فراهم کند.

برای اندازه‌گیری سرعت استنتاج، تابع cpu_ms_per_batch تعریف شد که زمان میانگین اجرای مدل روی یک بچ تصادفی از ورودی‌ها را بر حسب میلی‌ثانیه گزارش می‌دهد. در این تابع:

- ورودی مصنوعی با توزیع نرمال استاندارد تولید می‌شود:

$$(\text{batch\_size} = ۶۴, \text{channels} = ۳, \text{input\_size} = ۱۲۸, \text{input\_size} = ۱۲۸)$$

- ابتدا چند بار اجرای Warm-up انجام می‌شود تا اثر هزینه‌های اولیه (مانند آماده‌سازی کش و مسیرهای اجرایی) کاهش یابد.
- سپس مدل به تعداد $\text{runs} = ۵۰$ بار اجرا شده و زمان متوسط هر اجرا گزارش می‌شود.

---

[۱۱] Inference Speed

[۱۲] Epoch

این روش برای مقایسه‌ی نسبی سرعت مدل‌ها مناسب است، زیرا شرایط ورودی، تعداد اجراها و تنظیمات برای همه‌ی مدل‌ها ثابت نگه داشته شده است.

پس از تنظیم دقیق، دقت برای سه مدل پایه، مدل با فشرده‌سازی ٪۵۰ و مدل با فشرده‌سازی ٪۸۰ اندازه‌گیری شد. در نهایت، نمودار دقت بر حسب نرخ فشرده‌سازی رسم شد تا روند تغییرات عملکرد به‌صورت بصری مشخص شود.

۴-۲- نتایج عددی

جدول ۴-۱ : نتایج استفاده از تنظیم دقیق

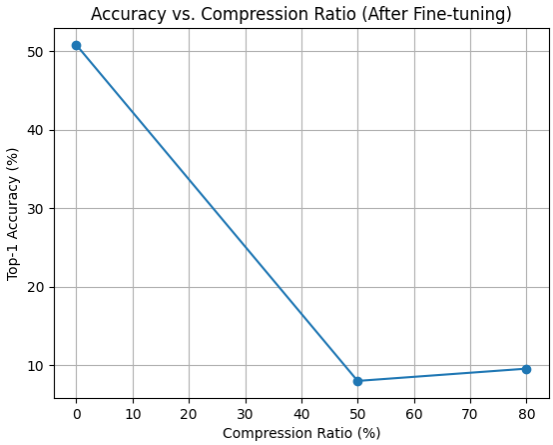
حالت مدل	تعداد پارامتر	دقت	زمان اجرا
پایه (بدون فشرده‌سازی)	۱۱,۱۸۱,۶۴۲	٪۵۰.۷۸	۱۸۳۰.۲۸
فشرده‌سازی با keep ratio = ۰.۵	۱۱,۱۷۹,۱۳۲	٪۸.۰۱	۱۸۵۴.۵۷
فشرده‌سازی با keep ratio = ۰.۲	۱۱,۱۷۷,۵۶۶	٪۹.۵۷	۱۸۲۱.۸۰

نتایج به‌دست‌آمده در جدول ۴-۱ نشان می‌دهد که فشرده‌سازی لایه‌ی نهایی، اگرچه منجر به کاهش تعداد پارامترها می‌شود، اما تأثیر آن بر سرعت استنتاج بسیار محدود بوده و در مقابل، افت دقت قابل‌توجهی ایجاد کرده است.

از نظر سرعت استنتاج، زمان اجرای مدل پایه برابر با حدود ۱۸۳۰.۲ میلی‌ثانیه بر بچ اندازه‌گیری شد، در حالی که این مقدار برای مدل با فشرده‌سازی ٪۵۰ حدود ۱۸۵۴.۵۷ میلی‌ثانیه و برای مدل با فشرده‌سازی ٪۸۰ حدود ۱۸۲۱.۸۰ میلی‌ثانیه بود. این مقادیر نشان می‌دهند که اختلاف زمان اجرا میان مدل‌ها ناچیز است و فشرده‌سازی لایه‌ی نهایی تأثیر محسوسی بر کاهش زمان استنتاج نداشته است. دلیل اصلی این رفتار آن است که در معماری داده، بخش عمده‌ی هزینه‌ی محاسباتی مربوط به لایه‌های کانولوشنی و عملیات‌های میانی شبکه است و لایه‌ی نهایی سهم کوچکی از زمان کل اجرا را به خود اختصاص می‌دهد. بنابراین، کاهش تعداد پارامترهای این لایه به‌تنهایی نمی‌تواند به بهبود چشمگیر سرعت استنتاج منجر

شود.

از سوی دیگر، بررسی دقت طبقه‌بندی پس از تنظیم دقیق نشان می‌دهد که مدل پایه همچنان دقت ٪۵۰.۷۸ را حفظ کرده است، در حالی که دقت مدل با فشرده‌سازی ٪۵۰ به ٪۸.۰۱ و مدل با فشرده‌سازی ٪۸۰ به ٪۹.۵۷ کاهش یافته است. این نتایج بیانگر آن است که فشرده‌سازی رتبه پایین شدید، ظرفیت نمایش لایه‌ی خروجی را به‌طور قابل‌توجهی کاهش داده و تنظیم دقیق ۳ ایپاک نتوانسته است این کاهش ظرفیت را جبران کند. به‌عبارت دیگر، اطلاعاتی که در فرآیند کم‌رتبه‌سازی حذف شده‌اند، برای جداسازی مناسب کلاس‌ها در مسئله‌ی CIFAR-۱۰ حیاتی بوده‌اند و بازیابی آن‌ها با یک تنظیم دقیق کوتاه‌مدت امکان‌پذیر نبوده است.



شکل ۴-۱ : دقت بر حسب نرخ فشرده‌سازی

نمودار شکل ۴-۱ نیز این روند را به‌صورت بصری تأیید می‌کند و نشان می‌دهد که با افزایش نرخ فشرده‌سازی، دقت مدل به‌شدت افت می‌کند. این نتیجه حاکی از آن است که در این سناریو، فشرده‌سازی لایه‌ی نهایی بدون به‌کارگیری راهبردهای تکمیلی (مانند تنظیم دقیق طولانی‌تر، بازآموزی بخشی از بدنه‌ی شبکه، یا فشرده‌سازی تدریجی) منجر به مبادله‌ی نامطلوبی میان دقت و کاهش پیچیدگی مدل می‌شود.

فصل پنجم

سؤالات تحلیلی

سؤالات تحلیلی

۱. تأثیر تأخیر ناشی از افزایش تعداد لایه‌ها را در محیط‌های مختلف تحلیل کنید.

افزایش تعداد لایه‌ها در شبکه‌های عصبی عمیق، به‌طور مستقیم منجر به افزایش تأخیر محاسباتی[۱۳] می‌شود، اما شدت و ماهیت این تأخیر به محیط اجرا وابسته است.

در پردازنده مرکزی (CPU)، افزایش تعداد لایه‌ها معمولاً باعث افزایش تقریباً خطی زمان استنتاج می‌شود. دلیل اصلی این موضوع، محدودیت در موازی‌سازی عملیات و وابستگی شدید محاسبات لایه‌ها به یکدیگر است. هر لایه جدید شامل ضرب‌های ماتریسی و اعمال غیرخطی اضافی است که باید به‌صورت ترتیبی اجرا شوند، در نتیجه تأخیر کلی افزایش می‌یابد.

در مقابل، در پردازنده گرافیکی (GPU)، به دلیل قابلیت موازی‌سازی گسترده، افزایش تعداد لایه‌ها الزاماً به همان نسبت باعث افزایش تأخیر نمی‌شود. بااین‌حال، در شبکه‌های بسیار عمیق، هزینه‌های جانبی مانند انتقال داده، همگام‌سازی هسته‌ها و اشباع حافظه می‌توانند موجب کاهش بازدهی شوند.

به‌طور خلاصه، هرچه محیط اجرا محدودتر باشد، حساسیت سیستم نسبت به افزایش تعداد لایه‌ها بیشتر خواهد بود و طراحی مدل باید با در نظر گرفتن این محدودیت‌ها انجام شود.

۲. چگونه می توان از مفاهیم «عدد وضعیت» برای انتخاب لایه های مستعد فشرده سازی استفاده کرد؟

با محاسبه‌ی عدد حالت ماتریس وزن هر لایه، لایه‌هایی که عدد حالت کوچک‌تری دارند به‌عنوان لایه‌های پایدارتر و کمتر حساس به خطا شناسایی می‌شوند و بنابراین گزینه‌های مناسب‌تری برای فشرده‌سازی شدید هستند، درحالی‌که لایه‌های با عدد حالت خیلی بزرگ را باید کمتر فشرده کرد یا دست‌نخورده گذاشت تا بی‌ثباتی عددی ایجاد نشود. همچنین می‌توان از تقریب‌های رتبه پایین برای افزایش پایداری مدل بهره برد.

[۱۳] Computational Latency

فصل ششم

جمع‌بندی و نتیجه‌گیری

جمع‌بندی و نتیجه‌گیری

در این پژوهش، به بررسی عملی و تحلیلی فشرده‌سازی مدل‌های یادگیری عمیق با تمرکز بر شبکه‌ی ResNet-۱۸ پرداختیم و تلاش شد ارتباط میان مفاهیم نظری روش‌های ماتریسی و رفتار تجربی شبکه‌های عصبی به‌صورت گام‌به‌گام روشن شود. فرآیند کار از تحلیل طیفی وزن‌ها آغاز، سپس فشرده‌سازی عملی پیاده‌سازی شد و در نهایت، اثر این فشرده‌سازی بر دقت و سرعت مدل مورد ارزیابی قرار گرفت.

در فصل دوم، با تحلیل طیفی وزن‌های یک لایه‌ی میانی و لایه‌ی نهایی، نشان داده شد که بخش قابل‌توجهی از واریانس وزن‌ها توسط تعداد محدودی از مقادیر منفرد توضیح داده می‌شود. به‌طور مشخص، برای حفظ حدود ٪۹۵ واریانس، تنها حدود ٪۵۷ از مقادیر منفرد در لایه‌ی نهایی و حدود ٪۷۳ در لایه‌ی میانی کافی بوده است. این مشاهده بیانگر وجود افزونگی ساختاری در وزن‌هاست و از منظر نظری، امکان استفاده از تقریب رتبه پایین را توجیه می‌کند. این نتایج با مبانی نظری ارائه‌شده در روش‌های مبتنی بر تجزیه مقدار منفرد و تحلیل مؤلفه‌های اصلی هم‌راستا هستند.

در فصل سوم، این تحلیل نظری به یک پیاده‌سازی عملی منجر شد و لایه‌ی نهایی مدل با یک ساختار دولایه‌ای حاصل از تجزیه مقدار منفرد جایگزین شد. برای تمرکز بر اثر فشرده‌سازی و کاهش هزینه‌ی محاسباتی، بدنه‌ی اصلی شبکه ثابت نگه داشته شد و تنها لایه‌ی خروجی به‌صورت محدود تنظیم دقیق شد. نتایج نشان داد که با افزایش نرخ فشرده‌سازی، دقت مدل کاهش می‌یابد؛ به‌گونه‌ای که فشرده‌سازی شدیدتر منجر به افت محسوس عملکرد می‌شود. این رفتار بیانگر آن است که اگرچه وزن‌ها دارای افزونگی هستند، اما حذف بیش از حد مؤلفه‌های مؤثر می‌تواند ظرفیت نمایش مدل را برای حل مسئله‌ی طبقه‌بندی کاهش دهد.

در فصل چهارم، تلاش شد با انجام تنظیم دقیق چند ایپاکی، بخشی از افت دقت ناشی از فشرده‌سازی جبران شود و هم‌زمان اثر فشرده‌سازی بر سرعت استنتاج روی پردازنده‌ی مرکزی بررسی شود. نتایج تجربی نشان داد که تنظیم دقیق کوتاه‌مدت (۳ ایپاک) نتوانست دقت مدل‌های فشرده‌شده را به‌طور معنادار بازیابی کند. این موضوع نشان می‌دهد که اطلاعات حذف‌شده در فرآیند کم‌رتبه‌سازی، به‌ویژه در نرخ‌های فشرده‌سازی بالا، برای تفکیک کلاس‌ها حیاتی بوده‌اند و بازیابی آن‌ها با آموزش محدود امکان‌پذیر نیست. از سوی دیگر، اندازه‌گیری سرعت استنتاج نشان داد که فشرده‌سازی صرفاً لایه‌ی نهایی تأثیر محسوسی بر

زمان اجرای کل مدل ندارد، زیرا هزینه‌ی اصلی محاسبات در لایه‌های کانولوشنی شبکه متمرکز است.

در مجموع، نتایج این پژوهش نشان می‌دهد که اگرچه تحلیل طیفی وزن‌ها ابزار قدرتمندی برای شناسایی افزونگی و امکان‌سنجی فشرده‌سازی است، اما انتقال مستقیم این تحلیل به یک فشرده‌سازی عملی، نیازمند ملاحظات بیشتری است. فشرده‌سازی رتبه پایین لایه‌های منفرد، به‌تنهایی تضمین‌کننده‌ی بهبود سرعت یا حفظ دقت نیست و برای دستیابی به مبادله‌ی مناسب میان دقت، اندازه‌ی مدل و سرعت، باید راهبردهای ترکیبی‌تری مانند فشرده‌سازی چندلایه‌ای، تنظیم دقیق طولانی‌تر یا بازآموزی بخشی از بدنه‌ی شبکه در نظر گرفته شود. به‌طور کلی، نشان داده شد که موفقیت روش‌های فشرده‌سازی در عمل، به میزان زیادی به محل اعمال فشرده‌سازی، شدت آن و راهبرد آموزشی پس از آن وابسته است.

منابع و مراجع

[۱] G. H. Golub and C. F. Van Loan, *Matrix Methods in Data Mining and Pattern Recognition*. Philadelphia, PA, USA: SIAM, ۲۰۰۸

[۲] I. T. Jolliffe, *Principal Component Analysis*, ۲nd ed. New York, NY, USA: Springer, ۲۰۰۲

[۳] T. N. Sainath, V. Sindhwani, and S. Kumar, "Low-rank matrix factorization for deep neural network training with high-dimensional output targets," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Vancouver, BC, Canada, ۲۰۱۳, pp. ٦٦٥٥_٦٦٥٩

پیوست‌ها

لینک گیت‌هاب پروژه:

https://github.com/ZahraBarati۹۹/Deep-Model-Compression-via-Low-Rank-Decomposition