تمرین دوم

نام درس: یادگیری عمیق

استاد درس: دکتر محمدرضا محمدی

نام: محمد حقیقت

شماره دانشجویی: 403722042

گرایش: هوش مصنوعی

دانشکده: مهندسی کامپیوتر

نیم سال دوم 1403-1404

سوال اول

آ)

مقایسهٔ Grad-CAM و SHAP از نظر ویژگی های خواسته شده

1. نحوه محاسبه اهمیت ویژگی ها (محلی یا جهانی)

: Grad-CAM

ویژگی‌های جهانی: Grad-CAM اهمیت هر کانال (channel) از نقشه ویژگی (feature map) را با استفاده از میانگین گرادیان‌های خروجی کلاس نسبت به پیکسل‌های نقشه ویژگی محاسبه می‌کند. این روش به صورت جهانی (global) به کانال‌های نقشه ویژگی نگاه می‌کند و وزن هر کانال را بر اساس گرادیان‌های کل نقشه تعیین می‌کند. سپس، این وزن‌ها با نقشه‌های ویژگی ترکیب می‌شوند تا نقشه برجستگی (saliency map) تولید شود.

این روش به طور خاص به اهمیت کانال‌ها وابسته است و جزئیات محلی (پیکسل به پیکسل) را به صورت غیرمستقیم از طریق ترکیب وزن‌های کانال و نقشه‌های ویژگی در نظر می‌گیرد.

:Shap-CAM

ویژگی‌های محلی: Shap-CAM اهمیت هر پیکسل را به طور مستقیم با استفاده از مقدار Shapley (مبتنی بر نظریه بازی‌ها) محاسبه می‌کند. این روش به جای تمرکز بر کانال‌ها، به صورت محلی (local) به هر پیکسل در نقشه ویژگی نگاه می‌کند و مشارکت حاشیه‌ای (marginal contribution) آن را در خروجی مدل ارزیابی می‌کند.

Shap-CAM با در نظر گرفتن تعاملات بین پیکسل‌ها، توضیحات دقیق‌تری در سطح پیکسل ارائه می‌دهد و روابط بین پیکسل‌ها را نیز مدل‌سازی می‌کند.

2. وابستگی به ساختار مدل و نیاز به گرادیان

:Grad-CAM

وابستگی به ساختار مدل: Grad-CAM نیازی به تغییر ساختار مدل یا آموزش مجدد ندارد، اما به لایه‌های خاص (مانند لایه‌های کانولوشنی آخر) وابسته است. این روش به گرادیان‌های خروجی

نسبت به نقشه‌های ویژگی نیاز دارد، بنابراین به مدل‌هایی که امکان محاسبه گرادیان را فراهم کنند وابسته است.

نیاز به گرادیان: Grad-CAM به شدت به گرادیان‌ها وابسته است. گرادیان‌ها به عنوان معیار اهمیت کانال‌ها استفاده می‌شوند، که این وابستگی می‌تواند به مشکلاتی مانند حساسیت به نویز یا دستکاری های متخاصم (adversarial manipulations) منجر شود (همان‌طور که در مقاله ذکر شده است).

:Shap-CAM

وابستگی به ساختار مدل: Shap-CAM نیز مانند Grad-CAM نیازی به تغییر ساختار مدل یا آموزش مجدد ندارد. این روش به نقشه‌های ویژگی لایه آخر دسترسی دارد و از خروجی مدل برای محاسبه مشارکت پیکسل‌ها استفاده می‌کند، بنابراین به ساختار کلی مدل وابستگی کمتری دارد.

نیاز به گرادیان: Shap-CAM کاملا بدون گرادیان (gradient-free) است. این روش از مقدار Shapley برای تخمین مشارکت حاشیه‌ای پیکسل‌ها استفاده می‌کند، که باعث می‌شود به گرادیان‌ها وابسته نباشد و در نتیجه از مشکلات مرتبط با گرادیان‌ها (مانند دستکاری متخاصم یا ناپایداری) مصون باشد.

3. دقت در شناسایی نواحی مهم

:Grad-CAM

Grad-CAM نقشه‌های برجستگی با رزولوشن نسبتاً بالا تولید می‌کند که مناطق مهم تصویر را برای تصمیم‌گیری مدل نشان می‌دهد. با این حال، به دلیل وابستگی به گرادیان‌ها و وزن‌دهی کانال‌ها، ممکن است نواحی غیرمرتبط یا نویزدار را نیز برجسته کند. همچنین، این روش تعاملات بین پیکسل‌ها را به طور کامل در نظر نمی‌گیرد، که می‌تواند دقت آن را در شناسایی نواحی واقعاً مهم کاهش دهد.

نتایج تجربی (جدول‌های ۱ و ۲ در مقاله) نشان می‌دهند که Grad-CAM در معیارهای Average Drop و Average Increase نسبت به Shap-CAM عملکرد ضعیف‌تری دارد.

:Shap-CAM

Shap-CAM به دلیل استفاده از مقدار Shapley، که تعاملات بین پیکسل‌ها را مدل‌سازی می‌کند، دقت بالاتری در شناسایی نواحی مهم تصویر دارد. این روش نقشه‌های برجستگی صاف‌تر و با نویز کمتر تولید می‌کند (شکل ۲ در مقاله) و مناطق مرتبط با تصمیم‌گیری مدل را با دقت بیشتری برجسته می‌کند.

4. حساسیت به تغییرات کوچک در ورودی

:Grad-CAM

Grad-CAM به دلیل وابستگی به گرادیان‌ها، به تغییرات کوچک در ورودی (مانند نویز یا دستکاری های متخاصم) حساس است. مقاله اشاره می‌کند که روش‌های مبتنی بر گرادیان می‌توانند به راحتی با دستکاری‌های متخاصم فریب بخورند، بدون اینکه تغییری قابل‌توجه در تصویر اصلی ایجاد شود

این حساسیت باعث می‌شود که Grad-CAM در سناریوهایی که پایداری توضیحات اهمیت دارد، کمتر قابل اعتماد باشد.

:Shap-CAM

Shap-CAM به دلیل عدم وابستگی به گرادیان‌ها و استفاده از مقدار Shapley، که مبتنی بر محاسبه مشارکت حاشیه‌ای در زیرمجموعه‌های مختلف است، به تغییرات کوچک در ورودی حساسیت کمتری دارد. این روش با نمونه‌گیری و میانگین‌گیری از مشارکت‌های پیکسل‌ها، پایداری بیشتری در برابر نویز و دستکاری‌ها ارائه می‌دهد.

مقاله نشان می‌دهد که Shap-CAM در برابر مشکلات مربوط به دستکاری‌های متخاصم مقاوم‌تر است و نقشه‌های برجستگی با نویز کمتر تولید می‌کند (شکل ۲ و بخش ۴.۴).

ب)

۱. آیا انتظار داریم Grad-CAM و Shap-CAM رفتار مشابهی داشته باشند؟ چرا؟

نه، انتظار نداریم رفتارشون مشابه باشه! دلیلش اینه که این دو روش کاملاً متفاوت کار می‌کنن و به تغییرات کوچیک تو ورودی (مثل نویز یا دستکاری‌های متخاصم) واکنش متفاوتی نشون میدن:

:Grad-CAM

این روش به گرادیان‌های مدل وابسته‌ست. یعنی برای اینکه بفهمه کدوم قسمت‌های تصویر مهمن، گرادیان‌های خروجی کلاس نسبت به نقشه‌های ویژگی (feature maps) رو حساب می‌کنه. اگه ورودی یه ذره تغییر کنه (مثلاً یه کم نویز اضافه بشه)، گرادیان‌ها ممکنه به شدت تغییر کنن، چون گرادیان‌ها به تغییرات کوچیک تو مدل حساسن.

مقاله‌ی اول (Shap-CAM) میگه روش‌های مبتنی بر گرادیان مثل Grad-CAM می‌تونن به راحتی با دستکاری‌های متخاصم فریب بخورن، بدون اینکه تصویر اصلی خیلی عوض بشه. این یعنی Grad-CAM تو همچین موقعیت‌هایی احتمالاً نقشه‌های برجستگی (saliency maps) ناپایداری تولید می‌کنه که با تغییرات کوچیک ورودی، خیلی فرق می‌کنن.

مقاله‌ی دوم (مقایسه SHAP و Grad-CAM تو HAR) هم تأیید می‌کنه که Grad-CAM بیشتر روی توضیحات فضایی (spatial) تمرکز داره و به فعال‌سازی‌های لایه‌های خاص (مثل لایه کانولوشنی آخر) وابستست. اگه ورودی تغییر کنه، این فعال‌سازی‌ها ممکنه به هم بریزن و توضیحات Grad-CAM عوض بشه.

:Shap-CAM

Shap-CAM به گرادیان وابسته نیست و از مقدار Shapley (یه روش از نظریه بازی‌ها) استفاده می‌کنه تا بفهمه هر پیکسل چقدر تو خروجی مدل نقش داره. این روش با نمونه‌گیری از زیرمجموعه‌های مختلف پیکسل‌ها و محاسبه مشارکت حاشیه‌ای (marginal contribution) هر پیکسل کار می‌کنه. چون میانگین‌گیری از این مشارکت‌ها انجام میشه، تغییرات کوچیک تو ورودی (مثل نویز) تأثیر خیلی کمتری روش دارن.

مقاله‌ی اول میگه Shap-CAM به خاطر این روش نمونه‌گیری و در نظر گرفتن تعاملات بین پیکسل‌ها، نقشه‌های برجستگی صاف‌تر و با نویز کمتری تولید می‌کنه. همچنین، چون به گرادیان وابسته نیست، در برابر دستکاری‌های متخاصم مقاوم‌تره.

مقاله‌ی دوم هم درباره SHAP می‌گه که این روش توضیحات در سطح ویژگی (feature-level) میده و به خاطر در نظر گرفتن تعاملات ویژگی‌ها، پایداری بیشتری تو سناریوهایی داره که ویژگی‌ها به هم وابسته‌ن (مثل داده‌های حرکتی تو HAR). این موضوع برای Shap-CAM هم صدق می‌کنه، چون از همون اصول SHAP استفاده می‌کنه.

چرا رفتارشون فرق داره؟

Grad-CAM به شدت به گرادیان‌های مدل وابستست که به تغییرات کوچیک حساسن، ولی Shap-CAM با یه روش مبتنی بر نمونه‌گیری و میانگین‌گیری کار می‌کنه که باعث می‌شه تغییرات کوچیک تو ورودی کمتر روش تأثیر بذاره. پس تو مدل‌هایی که به ورودی حساسن، Grad-CAM احتمالا نقشه های برجستگی‌ای تولید می‌کنه که با هر تغییر کوچیک ورودی عوض می‌شن، ولی Shap-CAM توضیحات پایدارتری می‌ده.

سوال دوم

الف)

در شبکه‌های عصبی پیچشی (Convolutional Neural Networks یا CNNها) به اشتراک‌گذاری پارامترها (Parameter Sharing) یکی از ویژگی‌های مهم و کلیدی این معماری است.

مفهوم به اشتراک‌گذاری پارامترها در شبکه‌های عصبی پیچشی:

در شبکه‌های عصبی معمولی (Fully Connected)، هر نورون به تمام ورودی‌ها متصل است و هر اتصال، یک وزن منحصر به‌فرد دارد. اما در CNNها، از فیلترها (یا کرنل‌ها) استفاده می‌شود که در سراسر تصویر حرکت می کنند تا ویژگی‌های محلی را استخراج کنند.

به اشتراک‌گذاری پارامترها یعنی:

یک فیلتر با مجموعه‌ای از وزن‌ها که در کل تصویر یا ورودی مکررا استفاده می‌شود.

این فیلتر با همان وزن‌ها روی بخش‌های مختلف تصویر حرکت می‌کند.

یعنی همه‌ی موقعیت‌های مختلف تصویر از یک مجموعه وزن مشترک استفاده می‌کنند.

تأثیر این ویژگی در روند آموزش مدل:

کاهش تعداد پارامترها:

چون همان فیلتر در کل تصویر استفاده می‌شود، نیازی نیست برای هر مکان، وزن جدیدی داشته باشیم و این باعث کاهش شدید حافظه مورد نیاز و پیچیدگی محاسباتی کمتر می‌شود.

به‌خصوص برای تصاویر بزرگ بسیار مهم است.

افزایش قابلیت تعمیم (Generalization):

استفاده از وزن‌های مشترک باعث می‌شود مدل الگوهای مشابه را در مکان‌های مختلف بهتر تشخیص دهد که در نتیجه مدل نسبت به تغییر مکان (Translation) مقاوم‌تر است.

افزایش سرعت آموزش:

به‌دلیل کاهش تعداد پارامترها، فرایند یادگیری سریع‌تر انجام می‌شود و مدل ساده‌تر و آموزش‌پذیرتر خواهد بود.

ب)

1. نظارت بر یک گونه‌ی خاص از گرگ در حیات وحش با پهپاد

مناسبه

دلیل:

این سناریو شامل تحلیل تصویر یا ویدیو از نمای بالا (پهپاد) است. CNNها توی پردازش تصویر و تشخیص اشیا خیلی قوی هستن و می‌تونن یاد بگیرن که ویژگی‌های ظاهری اون گونه خاص از گرگ رو از تصاویر تشخیص بدن، حتی در شرایط نوری مختلف یا زوایای متفاوت.

2. استخراج متن از درون صوت

مناسب است، اما نه به‌تنهایی

دلیل:

این مسئله مربوط به پردازش سیگنال‌های صوتی و تبدیل اون به متن یعنی تشخیص گفتار یا Speech Recognition هست.

CNN ممکنه برای پردازش طیف نگاره‌ی صوت (Spectrogram) استفاده بشه ولی به‌تنهایی برای درک دنباله‌ای از داده‌های زمانی کافی نیست و معماری‌هایی مثل RNN، LSTM یا مدل‌های مبتنی بر ترنسفورمر این‌کار مناسب‌ترن چون صوت ذاتا یک دنباله‌ی زمانی (sequence) است

3. شناسایی عمل انجام شده درون ویدیو

مناسب است، اما نه به‌تنهایی

دلیل:

ویدیو شامل دنباله‌ای از تصاویر (فریم‌ها) هست. که CNNها می‌تونن ویژگی‌های مکانی (فضایی) هر فریم رو استخراج کنن.

اما برای تشخیص حرکات و تغییرات زمانی باید اطلاعات بین فریم‌ها هم بررسی بشه.

ترکیب CNN با RNN یا استفاده از 3D-CNN (که هم ویژگی مکانی و هم زمانی رو در نظر می‌گیره) برای این‌کار مناسب‌تره.

4. داوری انجام حرکت میل‌زنی در مسابقات زورخانه‌ای

مناسب است، اما نه به‌تنهایی

دلیل:

این هم مثل مورد قبلی شامل تحلیل ویدیو و درک حرکات بدن در بازه زمانی مشخصه که CNN برای استخراج ویژگی از فریم‌های تصویری خیلی خوبه.

اما باید به صورت زمانی-پیوسته تحلیل بشه تا بشه داوری کرد که آیا حرکت درست و کامل انجام شده یا نه.

بنابراین می‌تونیم از CNN + LSTM یا 3D-CNN یا حتی Action Recognition Transformers استفاده کنیم.

ج)

معادله‌ی تلفیق (Fusion) لایه‌ی BatchNorm درون یک لایه‌ی Convolution مثل D2Conv

تلفیق(fusion) لایه‌ی Batch Normalization با لایه‌ی Convolution به‌منظور ساده‌سازی شبکه اسفتفاده می شود مخصوصا در مرحله‌ی inference .

این کار باعث افزایش سرعت و کاهش زمان اجرای مدل می‌شه بدون اینکه خروجی مدل تغییر کنه.

فرم کلی لایه‌ی کانولوشن + BatchNorm

فرض کنیم:

خروجی لایه‌ی کانولوشن قبل از BatchNorm به صورت:

$$y = W * x + b$$

که:

- x: ورودی
- W: وزن‌های فیلتر(Kernel)
- b: بایاس
- *: عملیات کانولوشن

حالا این خروجی وارد لایه‌ی BatchNorm می‌شه:

$$\mathrm{BN}(y) = \gamma \cdot \frac{y - \mu}{\sqrt{\sigma^2 + \epsilon}} + \beta$$

که:

- μ: میانگین کانال در BatchNorm
- σ^2: واریانس کانال
- γ: پارامتر مقیاس(scale)
- β: پارامتر انتقال(shift)
- ϵ: عدد کوچیک برای پایداری عددی

تلفیق این دو لایه(Convolution + BatchNorm)

ما می‌خوایم فرمول کانولوشن جدیدی بسازیم که همون خروجی رو بده ولی بدون نیاز به اجرای BatchNorm جداگانه.

با جایگذاری $y = W * x + b$ در معادله‌ی BatchNorm داریم:

$$\mathrm{BN}(W * x + b) = \gamma \cdot \frac{W * x + b - \mu}{\sqrt{\sigma^2 + \epsilon}} + \beta$$

الان می‌تونیم اینو به شکل یک کانولوشن جدید بنویسیم:

معادله‌ی نهایی تلفیق‌شده:

$$\mathrm{W}' = \gamma \cdot \frac{W}{\sqrt{\sigma^2 + \epsilon}}$$

$$\mathrm{b}' = \gamma \cdot \frac{b - \mu}{\sqrt{\sigma^2 + \epsilon}} + \beta$$

یعنی وزن و بایاس جدید می‌شن W' و b' که مستقیما در لایه‌ی کانولوشن جایگزین می‌شوند و از این به بعد دیگه نیازی به BatchNorm نیست.

تاثیر این تلفیق در عملکرد مدل:

افزایش سرعت inference: چرا که یک لایه حذف می‌شه، پس محاسبات کمتر می‌شن.

سادگی شبکه: ساختار گراف محاسباتی ساده‌تر و بهینه‌تر می‌شه.

بدون افت دقت: چون مقداردهی به دقت انجام می‌شه، خروجی نهایی هیچ تغییری نمی‌کنه.

نکته مهم:

این تلفیق فقط در مرحله‌ی inference استفاده انجام می‌شه، نه در آموزش.

در زمان آموزش، BatchNorm نقش مهمی در نرمال‌سازی و کمک به همگرایی سریع ایفا می‌کنه.

د)

مدل دو مرحله‌ای (Question-Guided Region Proposal + Focused VQA)

1. تولید نواحی پیشنهادی (Region Proposals):

با استفاده از یک مدل تشخیص شی مانند Faster R-CNN، DETR یا

(Segment Anything Model) SAM، نواحی مختلف تصویر را شناسایی می‌کنیم. این نواحی می‌توانند شامل اشیا یا بخش‌های معنادار تصویر باشند.

2. انطباق سوال با نواحی (Question-Region Matching):

به‌جای انطباق سوال با کل تصویر، مدل matching فعلی را روی هر کدام از این نواحی اعمال می‌کنیم. یعنی برای هر ناحیه پیشنهادی، similarity بین embedding سوال و embedding آن ناحیه (با یک encoder تصویری مانند ViT یا ResNet) محاسبه می‌شود.

می‌توان از CLIP برای encode کردن سوال و ناحیه‌ها استفاده کرد چون زبان و تصویر را در فضای embedding مشترک نگاشت می‌کند.

3. انتخاب ناحیه‌ی هدف (Target Region Selection):

ناحیه‌ای که بیشترین شباهت مفهومی با سوال دارد، انتخاب می‌شود. به عبارتی، ناحیه‌ای که مدل matching بالاترین امتیاز similarity را برای آن پیش‌بینی کرده، به عنوان ناحیه‌ی مورد نظر انتخاب می‌شود.

4. برش ناحیه و پاسخ‌دهی دقیق (Focused VQA):

ناحیه انتخاب شده را جداگانه به عنوان ورودی تصویری به مدل VQA وارد می‌کنیم، همراه با همان سوال اولیه. مدل حالا تمرکزش را فقط بر این ناحیه می‌گذارد، نه کل تصویر، و در نتیجه می‌تواند بهتر به سوالات جزئی پاسخ دهد.

سوال 3

محاسبه تعداد پارامتر:

$$\text{Parameters} = (K_H \times K_W \times C_{in} + 1) \times C_{out}$$

عملیات ضرب و جمع:

$$\text{ضرب ها} = K_H \times K_W \times C_{in} \times C_{out} \times H_{out} \times W_{out}$$

$$\text{جمع ها} = (K_H \times K_W \times C_{in} - 1 + 1) \times C_{out} \times H_{out} \times W_{out}$$

اون 1 + در فرمول جمع برای بایاس در نظر گرفته شده.

میدان دید موثر(Receptive Field) :

$$RF_{new} = RF_{prev} + (kernel\_size - 1) \times cumulative\_stride$$

خروجی:

$$out\_size = \frac{(W - K + 2P)}{S} + 1$$

Layer1

- تعداد فیلترها: 32
- اندازه‌ی فیلتر: 7×7
- Stride: 1
- Padding: same

تعداد پارامتر:

$(7 \times 7 \times 3 + 1) \times 32 = 4736$

تعداد ضرب و جمع :

ضرب‌ها = $7 \times 7 \times 3 \times 32 \times 256 \times 256 = 308{,}281{,}344$

جمع‌ها = $(7 \times 7 \times 3 - 1 + 1) \times 32 \times 256 \times 256 = 308{,}281{,}344$

میدان دید موثر:

میدان دید اولیه: 1 (پیکسل ورودی)

1:cumulative_stride

RF_new = 1 + (7 - 1) × 1 = 7

هر پیکسل خروجی، ۷×۷ پیکسل ورودی را می‌بیند.

خروجی این لایه:

256 × 256 × 32

BN1

تعداد پارامتر:

لایه های bn 2 تا پارامتر قابل آموزش برای هر کانال دارند:

1. گاما
2. بتا

پارامتر ها:

32 + 32 = 64

اگر بخواهیم پارامترهای آماری mean و var را هم بشماریم میشه 64 تا دیگه:

مجموع کل با آماری

32 × 4 = 128

اما پارامترهای قابل آموزش فقط 64 هستند.

تعداد ضرب و جمع :

عملیات نرمال‌سازی:

$$\mathrm{BN(x)} = \gamma \cdot \frac{x - \mu}{\sqrt{\sigma^2 + \epsilon}} + \beta$$

پس در مجموع:

۲ جمع/تفریق (x - mean و beta +)

۲ ضرب/تقسیم (تقسیم بر std و ضرب در gamma)

و گاهی sqrt و تقسیم رو هم در FLOPs به‌صورت تقریبی ضرب در نظر می‌گیرن (برای سادگی).

بنابراین برای هر پیکسل در هر کانال 4 عملیات داریم

تعداد کل پیکسل ها:

256 × 256 = 65536

تعداد کانال ها : 32

کل عملیات:

65536 × 32 × 4 = 8388608

میدان دید موثر تغییری نمی کند:

۷×۷

خروجی تغییری نمی کند:

256 × 256 × 32

:Layer2

تعداد پارامتر:

(5 × 5 × 32 + 1) × 64 = 51264

تعداد ضرب و جمع :

ضرب‌ها = 5 × 5 × 32 × 64 × 126 × 126 = 812,851,200

جمع‌ها = (5 × 5 × 32 − 1 + 1) × 64 × 126 × 126 = 812,851,200

میدان دید موثر:

میدان دید قبلی: 7×7

1:cumulative_stride

RF_new = 7 + (5 - 1) × 1 = 11

cumulative_stride = 1 × 2 = 2

خروجی این لایه:

$$out_{size} = \frac{(256 - 5 + 2 \times 0\ )}{2} + 1 = \left\lfloor \frac{251}{2} \right\rfloor + 1 = 126$$

126 × 126

BN2

تعداد پارامتر:

64 + 64 = 128

مجموع کل پارامترها با آماری:

64 × 4 = 256

اما پارامترهای قابل آموزش فقط 128 هستند.

تعداد ضرب و جمع :

در لایه قبلی bn محاسبه کردیم برای هر پیکسل در هر کانال 4 عملیات داریم

تعداد کل عملیات:

126 × 126 × 64 × 4 = 4,064,256

میدان دید موثر تغییری نمی کند:

11 × 11

خروجی تغییری نمی کند:

126 × 126 × 64

Layer3

خروجی این لایه:

$$out_{size} = \frac{(126 - 2 + 2 \times 0\ )}{2} + 1 = \left\lfloor \frac{124}{2} \right\rfloor + 1 = 63$$

63 × 63

تعداد پارامترها:

AvgPool هیچ پارامتر قابل یادگیری ندارد و فقط میانگین می گیرد.

عملیات‌های محاسباتی (FLOPs):

هر عملیات میانگین‌گیری برای یک پنجره 2×2:

جمع: 3 عمل (برای 4 عدد)

تقسیم: 1 عمل

کل عملیات:

63 × 63 × 64 × 4 = 1,016,064

میدان دید موثر:

میدان دید قبلی: 11×11

2:cumulative_stride

RF_new = 11 + (2 - 1) × 2 = 13

cumulative_stride = 2 × 2 = 4

Layer4

تعداد پارامتر:

(3 × 3 × 64 + 1) × 128 = 73,856

تعداد ضرب و جمع :

ضرب‌ها = 3 × 3 × 64 × 128 × 63 × 63 = 292,626,432

جمع‌ها = (3 × 3 × 64 − 1 + 1) × 128 × 63 × 63 = 292,626,432

میدان دید موثر:

میدان دید قبلی: 13×13

4:cumulative_stride

کرنل مؤثر با dilation:

Effective kernel size: 3 + (3−1) × (2−1) = 5

RF_new = 13 + (5−1) × 4 = 13 + 16 = 29

cumulative_stride = 4 × 1 = 4

خروجی این لایه(با استفاده از dilation) :

$$out_{size} = \frac{(H_{in} + 2 \times \text{padding} - \text{ dilation} \times (k-1) - 1\ )}{stride} + 1 =$$

$$out_{size} = \frac{(63 + 2 \times 0 - \ 2 \times (5-1) - 1\ )}{1} + 1 = \frac{58}{1} + 1 = 59$$

59 × 59 × 128

:Bn3

تعداد پارامتر:

128 + 128 = 256

مجموع کل پارامترها با آماری:

128 × 4 = 512

اما پارامترهای قابل آموزش فقط 256 هستند.

تعداد ضرب و جمع :

در لایه قبلی bn محاسبه کردیم برای هر پیکسل در هر کانال 4 عملیات داریم

تعداد کل عملیات:

59 × 59 × 128 × 4 = 1,783,552

میدان دید موثر تغییری نمی کند:

29 × 29

خروجی تغییری نمی کند:

59 × 59 × 128

Layer5

تعداد پارامتر:

(3 × 3 × 128 + 1) × 128 = 147,584

تعداد ضرب و جمع :

ضرب‌ها = 3 × 3 × 128 × 128 × 59 × 59 = 513,294,336

جمع‌ها = (3 × 3 × 128 − 1 + 1) × 128 × 59 × 59 = 513,294,336

میدان دید موثر:

میدان دید قبلی: 29 × 29

4:cumulative_stride

RF_new = 29 + (3−1) × 4 = 29 + 8 = 37

cumulative_stride = 4 × 1 = 4

خروجی این لایه:

$$out_{size} = \frac{(59 - 3 + 2 \times 0\,)}{1} + 1 = \left\lfloor \frac{56}{1} \right\rfloor + 1 = 57$$

57 × 57 × 128

:Bn4

تعداد پارامتر:

128 + 128 = 256

مجموع کل پارامترها با آماری:

128 × 4 = 512

اما پارامترهای قابل آموزش فقط 256 هستند.

تعداد ضرب و جمع :

در لایه قبلی bn محاسبه کردیم برای هر پیکسل در هر کانال 4 عملیات داریم

تعداد کل عملیات:

57 × 57 × 128 × 4 = 1,663,488

میدان دید موثر تغییری نمی کند:

37 × 37

خروجی تغییری نمی کند:

57 × 57 × 128

Layer6

خروجی این لایه:

$$out_{size} = \frac{(57 - 2 + 2 \times 0\ )}{2} + 1 = \left\lfloor \frac{55}{2} \right\rfloor + 1 = 28$$

28 × 28 × 128

تعداد پارامترها:

AvgPool هیچ پارامتر قابل یادگیری ندارد و فقط میانگین می گیرد.

عملیات‌های محاسباتی (FLOPs):

در لایه قبلی محاسبه کردیم AvgPool 4 عمل دارد.

کل عملیات:

28 × 28 × 128 × 4 = 401,408

میدان دید موثر:

میدان دید قبلی: 37×37

4:cumulative_stride

RF_new = 37 + (2 - 1) × 4 = 41

cumulative_stride = 4 × 2 = 8

layer7

تعداد پارامتر:

(3 × 3 × 128 + 1) × 256 = 295,168

تعداد ضرب و جمع :

ضرب‌ها = 3 × 3 × 128 × 256 × 28 × 28 = 231,211,008

جمع‌ها = (3 × 3 × 128 − 1 + 1) × 256 × 28 × 28 = 231,211,008

میدان دید موثر:

میدان دید قبلی: 41 × 41

8:cumulative_stride

RF_new = 41 + (3−1) × 8 = 41 + 16 = 57

cumulative_stride = 8 × 1 = 8

خروجی این لایه:

$$out_{size} = \frac{(28 - 3 + 2 \times 0\ )}{1} + 1 = \left\lfloor \frac{25}{1} \right\rfloor + 1 = 26$$

26 × 26 × 256

:Bn4

تعداد پارامتر:

256 + 256 = 512

مجموع کل پارامترها با آماری:

256 × 4 = 1024

اما پارامترهای قابل آموزش فقط 512 هستند.

تعداد ضرب و جمع :

در لایه قبلی bn محاسبه کردیم برای هر پیکسل در هر کانال 4 عملیات داریم

تعداد کل عملیات:

26 × 26 × 256 × 4 = 692,224

میدان دید موثر تغییری نمی کند:

57 × 57

خروجی تغییری نمی کند:

26 × 26 × 256

Layer6

خروجی این لایه:

$$out_{size} = \frac{(26 - 2 + 2 \times 0\ )}{2} + 1 = \left\lfloor \frac{24}{2} \right\rfloor + 1 = 13$$

13 × 13 × 256

تعداد پارامترها:

AvgPool هیچ پارامتر قابل یادگیری ندارد و فقط میانگین می گیرد.

عملیات‌های محاسباتی (FLOPs):

در لایه قبلی محاسبه کردیم AvgPool 4 عمل دارد.

کل عملیات:

13 × 13 × 256 × 4 = 173,056

میدان دید موثر:

میدان دید قبلی: 57×57

8:cumulative_stride

RF_new = 57 + (2 - 1) × 8 = 65

cumulative_stride = 8 × 2 = 16

fc1

تعداد پارامتر:

43,264 × 1024 + 1024 = 44,303,360

تعداد ضرب و جمع :

ضرب‌ها = 43,264 × 1024 = 44,302,336

جمع‌ها = 43,264 × 1024 + 1024 = 44,303,360

میدان دید موثر:

لایه Linear روی RF تأثیری ندارد

65 × 65

خروجی این لایه:

1,024

Fc2

تعداد پارامتر:

1024 × 1024 + 1024 = 1,049,600

تعداد ضرب و جمع :

ضرب‌ها = 1024 × 1024 = 1,048,576

جمع‌ها = 1024 × 1024 + 1024 = 1,049,600

میدان دید موثر:

لایه Linear روی RF تأثیری ندارد

65 × 65

خروجی این لایه:

1,024

Dropout

تعداد پارامتر:

هیچ پارامتر قابل یادگیری ندارد

تعداد ضرب و جمع :

ندارد

میدان دید موثر:

Dropout روی RF تأثیری ندارد

65 × 65

خروجی این لایه:

تغییری نمی کند

1,024

Fc3

تعداد پارامتر:

1024 × 10 + 10 = 10,250

تعداد ضرب و جمع :

ضرب‌ها = 1024 × 10 = 10,240

جمع‌ها = 1024 × 10 + 10 = 10,250

میدان دید موثر:

لایه Linear روی RF تأثیری ندارد

65 × 65

برای رفع برخی ایرادات و ابهامات از AI استفاده شده است.