بسم الله الرحمن الرحیم

دانشگاه صنعتی شریف
دانشکده‌ی مهندسی کامپیوتر

یادگیری عمیق

نیم‌سال دوم ۰۴-۰۳

مدرس: مهدیه سلیمانی

تمرین دوم | ددلاین تمرین : ۱۵ فروردین

- برای ارسال هر تمرین تا ساعت ۲۳:۵۹ روز ددلاین فرصت دارید. مهلت تاخیر (مجاز و غیر مجاز) برای این تمرین، ۷ روز است (یعنی حداکثر تاریخ ارسال تمرین ۲۲ فروردین است)
- مجموع نمرات تمرین نظری برابر ۱۱۰ می باشد که ۱۰ نمره آن جنبه اختیاری دارد و گرفتن نمره ۱۰۰ دریافت نمره کامل بخش نظری کفایت میکند و البته اگر نمره ای بالاتر از ۱۰۰ کسب کنید همان نمره ۱۰۰ برای شما در نظر گرفته میشود
- در هر کدام از سوالات، اگر از منابع خارجی استفاده کرده‌اید باید آن را ذکر کنید. در صورت همفکری با افراد دیگر هم باید نام ایشان را در سوال مورد نظر ذکر نمایید.
- پاسخ تمرین باید ماحصل دانسته‌های خود شما باشد. در صورت رعایت این موضوع، استفاده از ابزارهای هوش مصنوعی با ذکر نحوه و مصداق استفاده بلامانع است.
- پاسخ ارسالی واضح و خوانا باشد. در غیر این صورت ممکن است منجر به از دست دادن نمره شود.
- پاسخ ارسالی باید توسط خود شما نوشته شده باشد. به اسکرین‌شات از منابع یا پاسخ افراد دیگر نمره‌ای تعلق نمی‌گیرد.
- در صورتی که بخشی از سوال‌ها را جای دیگری آپلود کرده و لینک آن را قرار داده باشید، حتما باید تاریخ آپلود مشخص و قابل اتکا باشد.
- محل بارگذاری سوالات نظری و عملی در هر تمرین مجزا خواهد بود. به منظور بارگذاری بایستی تمارین تئوری در یک فایل pdf با نام HW2_[First-Name]_[Last-Name]_[Student-Id].pdf و تمارین عملی نیز در یک فایل مجزای زیپ با نام HW2_[First-Name]_[Last-Name]_[Student-Id].zip بارگذاری شوند.
- در صورت وجود هرگونه ابهام یا مشکل، در کوئرای درس آن مشکل را بیان کنید و از پیغام دادن مستقیم به دستیاران آموزشی خودداری کنید.
- طراحان این تمرین : علی رحیمی اکبر_امیرحسین ایزدی_امیرحسین علمدار_محمد مهدی واحدی_مهرداد مهابادی

بخش نظری (۱۰+۱۰۰ نمره)

پرسش ۱. Batch Normalization (۲۰ نمره)

۱. نحوه انجام نرمال‌سازی بچ در شبکه های تماما متصل و شبکه های پیچشی را با یکدیگر مقایسه کنید. همچنین نحوه اعمال نرمال‌سازی بچ در مرحله آموزش و آزمایش را نیز با یکدیگر مقایسه کنید.

۲. به صورت خلاصه Covariate shift را توضیح دهید و توضیح دهید چرا در نرمالسازی بچ، Covariate shift مابین داده های آموزش و آزمایش منجر به ناپایدار شدن در نتایج مدل برای دادگان آزمایش می شود؟

۳. شبکه CNN ای را در نظر بگیرید که از بلاک‌هایی به فرم زیر استفاده می‌کند:

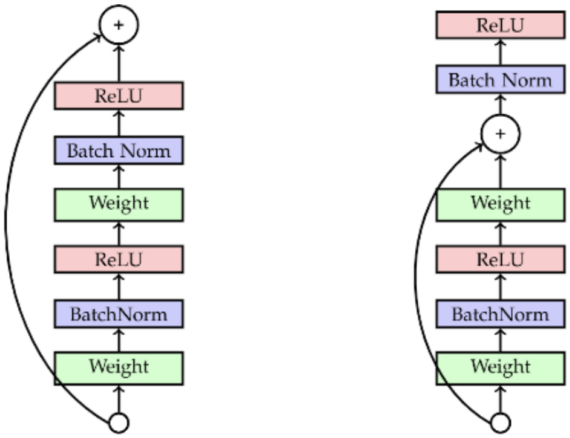
$$(\text{ConvLayer}) \rightarrow (\text{BatchNorm}) \rightarrow (\text{Activation})$$

آیا حذف بایاس (b) از لایه کانولوشن در کارکرد این شبکه اختلال ایجاد می‌کند؟ چرا؟ همچنین فرض کنید شبکه را آموزش داده‌ایم؛ آیا ضرب کردن وزن ها در یک عدد مانند α در زمان آزمایش (Inference) ، عملکرد شبکه را تغییر می‌دهد؟ ضرب کردن این ضریب در تمام درایه‌های ورودی شبکه چطور؟

۴. نشانی دهید که نرمال‌سازی بچ، باعث ایجاد نویزی در برآورد مقادیر گرادیان‌ها در مرحله آموزش می‌شود که به‌طور ضمنی یک منظور ساز است. این اثر را با اثر منظم سازی dropout مقایسه کنید.

۵. بررسی کنید که آیا استفاده پشت سر هم بلوک نرمال‌سازی بچ و dropout عموما می‌تواند مفید باشد؟ چرا؟

۶. شبکه‌های باقی‌مانده (ResNet) نقش مهمی در بهبود یادگیری عمیق داشته‌اند و امکان آموزش مدل‌های بسیار عمیق را فراهم کرده‌اند. با این حال، مکان قرارگیری نرمال‌سازی بچ (BN) نسبت به اتصالات میان‌بر تأثیر قابل توجهی بر پایداری آموزش، تعمیم‌پذیری و رفتار مدل در مرحله تست دارد. دو طراحی متفاوت برای بلوک باقی‌مانده را در نظر بگیرید که به بلوک باقی‌مانده با پیش‌فعال‌سازی و پس‌فعال‌سازی معروف است:



با در نظر گرفتن تأثیر نرمال‌سازی دسته‌ای بر انتشار گرادیان، پایداری بهینه‌سازی، سازگاری بین آموزش و تست، تغییر واریانس، یادگیری نمایش‌های عمیق، تحلیل کنید که چگونه مکان BN می‌تواند دینامیک آموزش شبکه را تحت تأثیر قرار دهد. (توجه کنید که منظور از Weight می تواند لایه تماما متصل و یا پیچشی باشد.)

پاسخ.

۱. – **نحوه انجام نرمال‌سازی بچ در شبکه‌های تماماً متصل:**

* **ورودی:** بردار ویژگی‌ها $(N \times D)$.
* **نرمال‌سازی:**

· میانگین (μ_B) و واریانس (σ_B^2) روی تمام نمونه‌های بچ و هر نورون محاسبه می‌شود:

$$\mu_B = \frac{1}{N}\sum_{i=1}^{N} x_i, \quad \sigma_B^2 = \frac{1}{N}\sum_{i=1}^{N}(x_i - \mu_B)^2.$$

· نرمال‌سازی:

$$\hat{x}_i = \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}}.$$

· مقیاس‌بندی و شیفت:

$$y_i = \gamma \hat{x}_i + \beta.$$

- **نحوه انجام نرمال‌سازی بچ در شبکه‌های پیچشی:**
  - **ورودی:** تنسور چندبعدی $(N \times C \times H \times W)$.
  - **نرمال‌سازی:**
    · میانگین (μ_c) و واریانس (σ_c^2) به صورت جداگانه برای هر کانال محاسبه می‌شود:

$$\mu_c = \frac{1}{NHW} \sum_{n,h,w} x_{n,c,h,w}, \quad \sigma_c^2 = \frac{1}{NHW} \sum_{n,h,w} (x_{n,c,h,w} - \mu_c)^2.$$

· نرمال‌سازی:

$$\hat{x}_{n,c,h,w} = \frac{x_{n,c,h,w} - \mu_c}{\sqrt{\sigma_c^2 + \epsilon}}.$$

· مقیاس‌بندی و شیفت:

$$y_{n,c,h,w} = \gamma_c \hat{x}_{n,c,h,w} + \beta_c.$$

- **مقایسه مرحله آموزش و آزمایش:**
  - **آموزش:**
    · آمارها $(\mu_B$، $\sigma_B^2)$ از نمونه‌های بچ محاسبه می‌شوند.
    · آمار جمعی $(\mu_P$، $\sigma_P^2)$ به‌روزرسانی می‌شود:

$$\mu_P = \alpha \mu_P + (1 - \alpha)\mu_B, \quad \sigma_P^2 = \alpha \sigma_P^2 + (1 - \alpha)\sigma_B^2.$$

  - **آزمایش:**
    · از آمار جمعی $(\mu_P$، $\sigma_P^2)$ استفاده می‌شود:

$$\hat{x} = \frac{x - \mu_P}{\sqrt{\sigma_P^2 + \epsilon}}.$$

۲. **Covariate Shift:** Covariate shift به تغییر توزیع داده‌های ورودی $(\mathrm{P(X)})$ بین داده‌های آموزش و آزمایش اشاره دارد، در حالی که رابطه بین ورودی و خروجی $(\mathrm{P(Y|X)})$ ثابت می‌ماند. به عبارت دیگر، داده‌های آزمایش ممکن است ویژگی‌های آماری متفاوتی نسبت به داده‌های آموزش داشته باشند.
تأثیر Covariate Shift در نرمال‌سازی بچ: - در نرمال‌سازی بچ (Batch Normalization)، آماره‌های میانگین (μ) و واریانس (σ^2) در مرحله آموزش از داده‌های بچ محاسبه می‌شوند، در حالی که در مرحله آزمایش از آماره‌های جمعی $(\mu_P$، $\sigma_P^2)$ استفاده می‌شود. - اگر Covariate shift وجود داشته باشد، توزیع داده‌های آزمایش با داده‌های آموزش متفاوت خواهد بود. این تفاوت باعث می‌شود آماره‌های جمعی که در آموزش محاسبه شده‌اند، دیگر نماینده دقیق توزیع داده‌های آزمایش نباشند. - نتیجه این عدم تطابق، ناپایداری در عملکرد مدل برای داده‌های آزمایش است، زیرا نرمال‌سازی بر اساس آماره‌های نادرست انجام می‌شود و مقادیر ورودی به لایه‌های بعدی شبکه دیگر به درستی نرمال‌سازی نمی‌شوند.

۳. حذف بایاس در لایه‌هایی که از نرمال‌سازی دسته‌ای استفاده می‌کنند معمولاً اختلالی در شبکه ایجاد نمی‌کند. دلیل این امر این است که لایه نرمال‌سازی دسته‌ای شامل پارامترهای قابل یادگیری (مقیاس γ و شیفت β) است که می‌توانند اثر نبود بایاس را جبران کنند. **دلیل:** مرحله تبدیل خطی (Affine Transformation) در نرمال‌سازی دسته‌ای (پس از نرمال‌سازی) انعطاف‌پذیری لازم را برای جبران تغییرات فراهم می‌کند. بنابراین، وجود بایاس ضروری نیست.

تأثیر ضرب کردن وزن‌ها در حین inference:

اگر وزن‌ها را در α ضرب کنیم:

$$x_{\text{in}} \Rightarrow \text{Conv Layer} \Rightarrow \alpha x_{\text{out}} \Rightarrow \text{Batch Norm} \Rightarrow \alpha\gamma\left(\frac{\alpha x_{\text{out}} - \alpha\mu}{|\alpha|\sigma}\right) + \alpha\beta$$

$$= \alpha\left(\gamma \text{sign}\ (\alpha)\left(\frac{x_{\text{out}} - \mu}{\sigma}\right) + \beta\right) = \alpha y$$

حال اگر تابع فعال‌سازی غیرخطی باشد در این صورت مقدار $f(\alpha y)$ رابطه مشخصی با $f(y)$ ندارد، بنابراین عملکرد مدل تحت تأثیر قرار می‌گیرد و رفتار مشابه قبل نخواهد داشت.

تأثیر ضرب کردن ورودی‌ها در حین inference:

داریم:

$$\alpha x_{\text{in}} \Rightarrow \text{Conv Layer} \Rightarrow \alpha x_{\text{out}} \Rightarrow \text{Batch Norm} \Rightarrow \gamma\left(\frac{\alpha x_{\text{out}} - \alpha\mu}{|\alpha|\sigma}\right) + \beta$$

$$= \text{sign}(\alpha)\left(\gamma\left(\frac{x_{\text{out}} - \mu}{\sigma}\right) + \beta\right)$$

می‌توان دید که در صورت مثبت بودن α تغییری در عملکرد مدل ایجاد نمی‌شود.

۴. – نرمال‌سازی بچ آماره‌های μ_B و σ_B^2 را روی بچ محاسبه می‌کند:

$$\mu_B = \frac{1}{m}\sum_{i=1}^{m} x_i, \quad \sigma_B^2 = \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_B)^2.$$

– مشتق $\hat{x}_i$ نسبت به x_i به صورت زیر است:

$$\frac{\partial \hat{x}_i}{\partial x_i} = \frac{1}{\sqrt{\sigma_B^2 + \epsilon}} - \frac{(x_i - \mu_B)}{2(\sigma_B^2 + \epsilon)^{3/2}} \cdot \frac{\partial \sigma_B^2}{\partial x_i}.$$

– این تصادفی بودن در μ_B و σ_B^2 باعث می‌شود که گرادیان‌ها نویز داشته باشند و به طور ضمنی عملکرد یک منظم‌ساز را ایفا کنند.

- **مقایسه با دراپ‌اوت:**

– دراپ‌اوت نورون‌ها را به صورت تصادفی خاموش می‌کند و گرادیان‌ها را تحت تأثیر قرار می‌دهد:

$$\frac{\partial L}{\partial x_i} = \frac{\partial L}{\partial h_i^{\text{drop}}} \cdot m_i,$$

که در آن m_i یک متغیر تصادفی برنولی است.

– تفاوت‌های کلیدی:

* نرمال‌سازی بچ نویز اضافی وارد می‌کند، در حالی که دراپ‌اوت نویز ضربی وارد می‌کند.
* نرمال‌سازی بچ به طور ضمنی عمل می‌کند، در حالی که دراپ‌اوت به طور صریح عمل می‌کند.

۵.

- **Batch Normalization (BN) و Dropout به طور همزمان:**

  - استفاده از Batch Normalization (BN) و Dropout به طور همزمان در یک شبکه عصبی موضوعی بحث‌برانگیز است. در بیشتر موارد، ترکیب این دو تکنیک بدون دقت کافی می‌تواند مضر باشد و حتی منجر به کاهش عملکرد مدل شود. دلایل این امر شامل موارد زیر است:

    * **تداخل در اهداف BN و Dropout:**
      - BN سعی دارد توزیع ورودی‌های لایه را تثبیت کند و واریانس آنها را در کل مینی‌بچ یکسان کند.
      - Dropout به طور تصادفی برخی از نودها را صفر می‌کند که باعث تغییر و ناپایداری در توزیع ورودی‌های BN در هر مینی‌بچ می‌شود.
      - این تضاد می‌تواند باعث شود که BN در حین آموزش مقدار واریانس غیرواقعی‌ای را ذخیره کند، که در نتیجه در زمان تست مقادیر نرمال‌سازی‌شده نامناسب خواهند بود.
    * **کاهش تأثیر Dropout به دلیل BN:**
      - BN باعث نرمال‌سازی خروجی‌ها می‌شود و این باعث می‌شود که اثر تصادفی حذف نودها در Dropout کاهش یابد.
      - به همین دلیل، بسیاری از تحقیقات نشان داده‌اند که در حضور BN، Dropout اغلب دیگر مفید نیست و حتی می‌تواند زیان‌آور باشد.
    * **عدم نیاز به Dropout در بسیاری از مدل‌های مدرن:**
      - در مدل‌هایی که از BN استفاده می‌کنند (مانند ResNet)، اغلب نیازی به Dropout وجود ندارد، زیرا BN به خودی خود تا حدی نقش تنظیم‌کننده (Regularization) را ایفا می‌کند.
      - در واقع، در معماری‌های مدرنی مانند ResNet، EfficientNet و DenseNet، Dropout اغلب فقط در لایه‌های نهایی و برای تنظیم وزن‌های Fully Connected (FC) استفاده می‌شود.
    * **تغییر واریانس بین زمان آموزش و تست (Variance Shift):**
      - Dropout هنگام آموزش فعال است، اما در زمان تست غیرفعال می‌شود. از سوی دیگر، BN از میانگین متحرک و واریانس متحرک برای نرمال‌سازی در زمان تست استفاده می‌کند.
      - اگر Dropout قبل از BN قرار گیرد، BN واریانس‌های نادرستی را در طول آموزش ذخیره می‌کند، که در زمان تست، به علت عدم وجود Dropout، منجر به ناهمخوانی توزیع خروجی‌ها می‌شود و مدل عملکرد ضعیفی خواهد داشت.
    * **مدل‌های خاص که ترکیب BN و Dropout را استفاده می‌کنند:**
      - در مدل‌های خاصی مانند Wide ResNet که لایه‌های بسیار عریض دارند، مشاهده شده که استفاده همزمان BN و Dropout (در جایگاه مناسب) می‌تواند مفید باشد.
      - در این حالت، به دلیل تعداد زیاد نودهای هر لایه، BN به تنهایی کافی نیست و Dropout می‌تواند به جلوگیری از بیش‌برازش کمک کند.

- **اگر مجبور به استفاده از BN و Dropout با تابع فعال‌ساز باشیم، بهترین ترتیب استفاده به چه صورت است؟**

  - اگر مجبور به استفاده از BN و Dropout با تابع فعال‌ساز باشیم، بهترین ترتیب استفاده به این صورت است:

    * $\text{Conv/Fully Connected} \rightarrow \text{BN} \rightarrow \text{Activation} \rightarrow \text{Dropout}$.

  - **چرا این ترتیب مناسب‌تر است؟**

    * ابتدا BN: نرمال‌سازی باید روی خروجی خام لایه قبلی انجام شود تا واریانس ویژگی‌ها را قبل از ورود به تابع فعال‌ساز کنترل کند.
    * سپس تابع فعال‌ساز: پس از نرمال‌سازی، خروجی به تابع فعال‌ساز داده می‌شود که مقادیر غیرخطی ایجاد کند.

* در نهایت Dropout: حذف تصادفی نودها پس از فعال‌سازی بهتر است، زیرا در این حالت ویژگی‌های پردازش‌شده و فعال‌شده از بین می‌روند، نه ویژگی‌های خام.

- **چرا ترتیب‌های دیگر مناسب نیستند؟**

* Dropout → BN → Activation :
  - مشکل: اگر Dropout قبل از BN اعمال شود، میانگین و واریانس محاسبه‌شده توسط BN دچار اعوجاج می‌شود، زیرا در هر بار پردازش برخی نودها حذف شده‌اند. این باعث اختلاف واریانس بین آموزش و تست می‌شود که منجر به کاهش دقت مدل خواهد شد.
* BN → Dropout → Activation:
  - مشکل: اگر Dropout قبل از تابع فعال‌ساز باشد، نودهایی که حذف شده‌اند ممکن است تأثیر مستقیمی بر تعداد نودهای فعال در هر لایه داشته باشند، که رفتار مدل را دچار ناپایداری می‌کند.

- **چرا ترتیب بین BN و Activation مهم است؟**

- یکی دیگر از نکات کلیدی در ترتیب این لایه‌ها، نحوه قرارگیری نرمال‌سازی بچ (BN) نسبت به تابع فعال‌ساز (Activation Function) است. دو ترتیب رایج در استفاده از BN و تابع فعال‌ساز عبارتند از:

* BN → Activation (رایج‌ترین روش):
  - ترتیب: Conv/Fully Connected → BN → Activation.
  - چرا این روش بهتر است؟
  - BN واریانس و میانگین خروجی لایه را قبل از اعمال تابع فعال‌ساز کنترل می‌کند.
  - در صورت استفاده از ReLU، مقادیر منفی را حذف کرده و فقط مقادیر مثبت را نگه می‌داریم، اما این روی نرمال‌سازی تاثیری ندارد زیرا BN قبلاً کار خود را انجام داده است.
  - بسیاری از شبکه‌های معروف مانند ResNet از این ترتیب استفاده می‌کنند.
  - این ترتیب به بهبود پایداری گرادیان و جلوگیری از مشکلاتی مانند vanishing/exploding gradients کمک می‌کند.
* Activation → BN (روش کمتر رایج اما در برخی موارد مفید):
  - مشکل این روش چیست؟
  - در صورت استفاده از ReLU، مقادیر منفی حذف می‌شوند و خروجی‌ها skew (به سمت صفر متمایل) می‌شوند. در این حالت، اگر BN بعد از ReLU قرار گیرد، توزیع خروجی را اصلاح می‌کند، اما این ممکن است تاثیر منفی بر اطلاعات ویژگی‌ها بگذارد.
  - به همین دلیل، برخی تحقیقات نشان داده‌اند که این روش ممکن است در بعضی شرایط باعث کاهش عملکرد شود.
  - ✓پس چه زمانی Activation → BN می‌تواند مفید باشد؟
  - در مدل‌هایی که از Activationهای خاص مانند Swish یا SELU استفاده می‌کنند، ممکن است ابتدا فعال‌سازی و سپس BN نتیجه بهتری بدهد.
  - در برخی مدل‌های GAN، این ترتیب برای حفظ ویژگی‌های نرمال‌سازی‌شده بهتر عمل کرده است.

- **ترتیب نهایی پیشنهادی:**

- (Conv/Fully Connected → BN → Activation → Dropout). این ترتیب بهترین ترکیب برای حفظ پایداری شبکه و جلوگیری از کاهش عملکرد ناشی از ناسازگاری BN و Dropout است. هرچند لازم به ذکر است همانطور که اشاره شد در صورت اجبار استفاده همزمان نرمالساز بچ و دراپ اوت این ترکیب عموما پیشنهاد میشود و در شرایط عادی یکی از بلاک های نرمالساز و یا دراپ اوت حذف خواهد گردید

۶. در ادامه تأثیر جایگاه لایه نرمال‌سازی دسته‌ای (Batch Normalization یا BN) نسبت به اتصال پرشی در بلوک‌های باقی‌مانده بررسی شده است:

- **انتشار گرادیان و پایداری:**

* در پیش‌نرمال‌سازی، تابع پس از جمع همانی است ($f(y) = y$)، که مسیر میان‌بر را بدون تغییر نگه می‌دارد و گرادیان‌ها را بدون تضعیف به لایه‌های ابتدایی می‌رساند. این ویژگی مانع محوشدگی یا انفجار گرادیان در شبکه‌های عمیق می‌شود.
* در پس‌نرمال‌سازی، BN و ReLU پس از جمع، سیگنال میان‌بر را تغییر می‌دهند و مسیر هویت را مختل می‌کنند، که می‌تواند به محوشدگی گرادیان در شبکه‌های عمیق منجر شود.
* پیش‌نرمال‌سازی امکان آموزش شبکه‌های بسیار عمیق (مثل ResNet با ۱۰۰۰+ لایه) را فراهم می‌کند، در حالی که پس‌نرمال‌سازی در چنین اعماقی ناپایدار است.

- **پایداری آموزش و منظم‌سازی:**

* پیش‌نرمال‌سازی ورودی هر لایه را نرمال می‌کند، که باعث پایداری بیشتر در آموزش و کاهش ناهماهنگی بین فازهای آموزش و تست می‌شود.
* این معماری اثر منظم‌کننده دارد، زیرا ورودی‌های نرمال‌شده از بیش‌برازش جلوگیری می‌کنند و خطای تست را کاهش می‌دهند (خطای آموزش کمی بالاتر اما تست پایین‌تر).
* در پس‌نرمال‌سازی، سیگنال پس از جمع نرمال‌نشده به لایه بعد می‌رود، که ممکن است به بیش‌برازش یا ناپایداری منجر شود.

- **سرعت همگرایی:**

* پیش‌نرمال‌سازی یادگیری را تسریع می‌کند، زیرا شبکه در ابتدا رفتار نزدیک به هویت دارد و تنها residualهای کوچک را می‌آموزد. این منجر به کاهش سریع خطا و همگرایی پایدار می‌شود.
* در آزمایش‌ها، ResNet-1001 با پیش‌نرمال‌سازی به خطای تست ۴.۹۲% در CIFAR-10 رسید، در حالی که پس‌نرمال‌سازی خطای ۷.۶۱% داشت.
* پس‌نرمال‌سازی در شبکه‌های عمیق یادگیری کندتر و پرنوسانی دارد و گاهی نیاز به ترفندهای اضافی برای همگرایی است.

- **عملکرد در استنتاج:**

* پیش‌نرمال‌سازی اطلاعات (از جمله مقادیر منفی) را بدون قطع حفظ می‌کند، که ظرفیت مدل برای نمایش ویژگی‌های پیچیده را افزایش می‌دهد.
* وجود BN در ابتدای هر بلوک، ورودی‌ها را در تست پایدار نگه می‌دارد و مدل را نسبت به تغییرات توزیع ورودی مقاوم‌تر می‌کند.
* پس‌نرمال‌سازی خروجی هر بلوک را مستقل نرمال می‌کند، اما ممکن است به دلیل ReLU پس از جمع، اطلاعات منفی از دست برود و خروجی bias مثبت داشته باشد.
* هر دو معماری از نظر محاسباتی مشابه‌اند، اما پیش‌نرمال‌سازی به دلیل BN ابتدایی پایداری بیشتری در تست فراهم می‌کند.

- **جمع‌بندی و توصیه:**

* پیش‌نرمال‌سازی به دلیل حفظ مسیر میان‌بر بدون تغییر، انتشار بهتر گرادیان، پایداری آموزش، و همگرایی سریع‌تر، برتری قابل توجهی دارد.

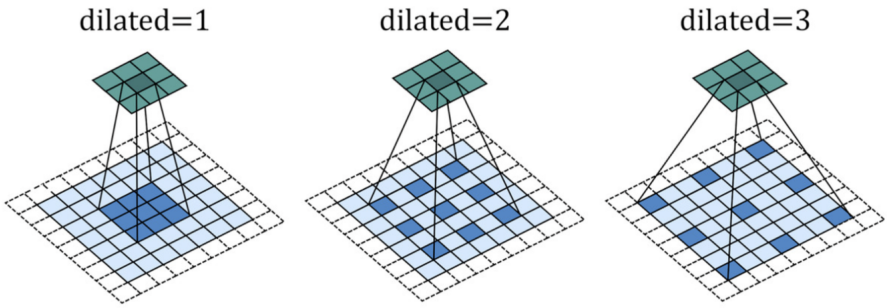
* این معماری در شبکه‌های مدرن بینایی (EfficientNet، ConvNeXt) و زبان (Pre-LN ترنسفورمرها) استاندارد است.

* توصیه می‌شود در طراحی بلوک‌های باقی‌مانده از پیش‌نرمال‌سازی استفاده شود، مگر دلایل خاصی برای پس‌نرمال‌سازی وجود داشته باشد.

▷

پرسش ۲. Dilated Convolution (۱۵ نمره)

در شبکه های پیچشی به صورت متداول از لایه های کانولوشن ساده استفاده می شود که با آن آشنا هستید. نوع دیگری از لایه ها که می توان از آنان ددر شبکه های پیچشی استفاده نمود، لایه کانولوشن گسترش یافته یا متسع است. در شکل ۵ تصویر شهودی از فیلتر کانولوشن گسترش یافته ارائه شده است، این فیلتر ها میان خانه هایی که فیلتر با استفاده از اطلاعات آن‌ها لایه بعد را محاسبه می‌کنند فاصله می‌اندازند یا به بیانی دیگر در زمان اعمال فیلتر و انجام عملیات ضرب کانولوشن، بر روی ورودی با گام (dilated) بزرگتری حرکت می کنیم، توجه کنید طول گام مفهومی متفاوت نسبت به طول گام (stride) در لایه های شبکه کانولوشن دارد.



شهودی از کانولوشن گسترش یافته با گام های متفاوت

همانطور که در شکل ۱ نیز مشخص است این روش، یک روش کم هزینه برای افزایش محدوده دید شبکه های پیچشی است. کانولوشن گسترش یافته بصورت فرم بسته ریاضی زیر تعریف می‌شود.

$$(K \star_D I)(i,j) = \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} K(m,n) I(i + Dm, j + Dn)$$

فرض کنید یک شبکه عصبی کانولوشنال با L لایه طراحی شده است که هر لایه شامل فیلترهای کانولوشن با پارامترهای زیر است:

- اندازه فیلتر (Kernel Size): $k_\ell \times k_\ell$ (مربعی)،
- نرخ اتساع (Dilation Rate): d_ℓ،
- گام (Stride): s_ℓ،
- بدون پدینگ (No Padding).

هدف ما بررسی تأثیر پارامترها بر **گستره دید نسبی** (Relative Receptive Field) است. گستره دید نسبی به صورت نسبت گستره دید در خروجی لایه Lـام به اندازه ورودی اصلی ($M \times N$) تعریف می‌شود.

۱. فرمول کلی گستره دید نسبی $R^{(L)}_{\text{relative}}$ را به صورت تابعی از k_ℓ، d_ℓ، و s_ℓ استخراج کنید.

۲. فرض کنید هدف ما این است که گستره دید نسبی بیشتر از یک حد آستانه (T) باشد، در حالی که هزینه‌های محاسباتی (FLOPs) کمترین مقدار ممکن باشد:

- معادله‌ای برای تعیین شرایط بهینه d_ℓ و s_ℓ بنویسید.
- آیا این شرایط بهینه به تعداد لایه‌ها (L) وابسته است؟ چرا؟

پاسخ.

۱. - **روابط بازگشتی:**

$$\begin{aligned}\text{jump}_\ell &= \text{jump}_{\ell-1} \cdot s_\ell, \\ R_\ell &= R_{\ell-1} + (k_\ell - 1) \cdot d_\ell \cdot \text{jump}_{\ell-1}.\end{aligned}$$

- **فرمول بسته برای گستره دید نسبی:**

$$R^{(L)}_{\text{relative}} = \frac{R_L}{M} = \frac{1 + \sum_{\ell=1}^{L} \left((k_\ell - 1) \cdot d_\ell \cdot \prod_{m=1}^{\ell-1} s_m \right)}{M}.$$

۲. - **معادله شرایط بهینه:** برای رسیدن به گستره دید نسبی بیشتر از T:

$$1 + \sum_{\ell=1}^{L} \left((k_\ell - 1) \cdot d_\ell \cdot \prod_{m=1}^{\ell-1} s_m \right) \geq T \cdot M.$$

برای کاهش هزینه‌ها (FLOPs):

$$\mathcal{O}(d_\ell^2 \cdot k_\ell^2) \quad \text{و} \quad \mathcal{O}\left(\frac{1}{s_\ell^2}\right).$$

- **وابستگی به تعداد لایه‌ها:** شرایط بهینه به L وابسته است، زیرا افزایش L می‌تواند گستره دید را افزایش دهد، اما هزینه‌ها را نیز افزایش می‌دهد.

▷

پرسش ۳. ROI Alignment (۱۰ نمره)

۱. یکی از مسائل در برخی روش‌های تشخیص شی، ROI Alignment است. نحوه‌ی کار کرد درون‌یابی خطی و نزدیک‌ترین همسایه را توضیح دهید. برای درون‌یابی خطی روابط مربوطه را بنویسید.

۲. یک عکس ۳۲ در ۳۲ را در نظر بگیرید، فرض کنید به یک activation map ۱۰ در ۱۰ تبدیل شده باشد. مقدار متناظر با نقطه‌ی x=۴ و y=۸ در عکس اولیه را در نقشه‌ی نهایی برحسب مقادیر پیکسل‌های نقشه محاسبه کنید.

پاسخ.

۱. درون‌یابی خطی یک روش برای تقریب مقدار یک نقطه در یک شبکه‌ی گسسته است. این روش ابتدا در یک جهت (مثلاً x) مقدار را بر اساس نقاط همسایه محاسبه کرده و سپس در جهت دیگر (مثلاً y) مقدار نهایی را پیدا می‌کند. روابط مربوط به درون‌یابی خطی به‌صورت زیر است:

$$f(x, y_1) = \frac{x_2 - x}{x_2 - x_1} f(Q_{11}) + \frac{x - x_1}{x_2 - x_1} f(Q_{21}),$$

$$f(x, y_2) = \frac{x_2 - x}{x_2 - x_1} f(Q_{12}) + \frac{x - x_1}{x_2 - x_1} f(Q_{22}).$$

$$f(x, y) = \frac{y_2 - y}{y_2 - y_1} f(x, y_1) + \frac{y - y_1}{y_2 - y_1} f(x, y_2).$$

در مقابل، روش نزدیک‌ترین همسایه، مقدار نزدیک‌ترین پیکسل را به‌عنوان مقدار درون‌یابی‌شده در نظر می‌گیرد که بسیار ساده‌تر از درون‌یابی خطی است ولی ممکن است باعث از دست رفتن جزئیات شود.

۲. برای محاسبه‌ی مقدار متناظر با نقطه‌ی $(x = 4, y = 8)$ در نقشه‌ی نهایی، ابتدا تبدیل مختصات را انجام می‌دهیم. از آن‌جایی که ابعاد تصویر از 32×32 به 10×10 کاهش یافته است، مقیاس تبدیل برابر است با:

$$S_x = \frac{10}{32}, \quad S_y = \frac{10}{32}$$

بنابراین مختصات متناظر در نقشه‌ی ویژگی برابر است با:

$$x' = 4 \times \frac{10}{32} = 1.25, \quad y' = 8 \times \frac{10}{32} = 2.5$$

حال مقدار متناظر با این نقطه را می‌توان با استفاده از درون‌یابی خطی از مقادیر پیکسل‌های نزدیک محاسبه کرد. نقاط همسایه‌ی این مختصات عبارتند از:

$$(x_1, y_1) = (1, 2), \quad (x_2, y_2) = (2, 3)$$

مقدار متناظر با $f(x', y')$ را می‌توان از روابط بالا محاسبه کرد.

ابتدا در جهت x و سپس در جهت y درون‌یابی می‌کنیم:

$$(0.5)\left[(0.75)v_{1,2} + (0.25)v_{2,2}\right] + (0.5)\left[(0.75)v_{1,3} + (0.25)v_{2,3}\right]$$

◁

پرسش ۴. Convolution Gradient (۱۵ نمره)

۱. بردار یک بعدی $\vec{x}$ با چهار درایه را در نظر بگیرید. فرض کنید روی این بردار یک کانولوشن یک بعدی با سایز کرنل ۳ و Padding ۱ اعمال کنیم. عملیات انجام شده را به فرم ماتریسی بنویسید و خروجی را محاسبه کنید.

۲. حال فرض کنید یک تابع زیان روی خروجی این لایه اعمال شده و به زیان L رسیده‌ایم. مشتق L را نسبت به $\vec{x}$ با کمک قاعده زنجیره‌ای محاسبه کنید.

۳. به طور دقیق مشخص کنید باید چه عملیاتی روی مشتق جزئی تابع زیان نسبت به خروجی این لایه انجام دهیم تا مشتق جزئی زیان نسبت به بردار $\vec{x}$ بدست آید؟

۴. حال فرض کنید بردار $\vec{x}$ به طول چهار به شما داده شده است و می‌خواهید با کمک upsampling آن را به فضای $\mathbb{R}^6$ ببرید. برای اینکار از Transpose Convolution با padding صفر و stride یک استفاده می‌کنیم. اگر کرنل ما $\vec{w}$ به طول سه باشد عملیات را به فرم ماتریسی بنویسید. ماتریس حاصل را با ماتریس بخش ۱ مقایسه کنید. اگر padding یک باشد چه اتفاقی می‌افتد؟

پاسخ.

۱.

$$\underbrace{\begin{bmatrix} w_1 & w_2 & w_3 & o & o & o \\ o & w_1 & w_2 & w_3 & o & o \\ o & o & w_1 & w_2 & w_3 & o \\ o & o & o & w_1 & w_2 & w_3 \end{bmatrix}}_{W\in\mathbb{R}^{4\times 6}} \underbrace{\begin{bmatrix} o \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ o \end{bmatrix}}_{\mathbf{x}'\in\mathbb{R}^6} = \underbrace{\begin{bmatrix} x_1w_2 + x_2w_3 \\ x_1w_1 + x_2w_2 + x_3w_3 \\ x_2w_1 + x_3w_2 + x_4w_3 \\ x_3w_1 + x_4w_2 \end{bmatrix}}_{\mathbf{o}\in\mathbb{R}^4}$$

۲.

$$\frac{\partial L}{\partial x} = \frac{\partial L}{\partial o}\frac{\partial o}{\partial x} = \begin{bmatrix} \frac{\partial L}{\partial o_1} & \frac{\partial L}{\partial o_2} & \frac{\partial L}{\partial o_3} & \frac{\partial L}{\partial o_4} \end{bmatrix} \begin{bmatrix} \frac{\partial o_1}{\partial x_1} & \frac{\partial o_1}{\partial x_2} & \frac{\partial o_1}{\partial x_3} & \frac{\partial o_1}{\partial x_4} \\ \frac{\partial o_2}{\partial x_1} & \frac{\partial o_2}{\partial x_2} & \frac{\partial o_2}{\partial x_3} & \frac{\partial o_2}{\partial x_4} \\ \frac{\partial o_3}{\partial x_1} & \frac{\partial o_3}{\partial x_2} & \frac{\partial o_3}{\partial x_3} & \frac{\partial o_3}{\partial x_4} \\ \frac{\partial o_4}{\partial x_1} & \frac{\partial o_4}{\partial x_2} & \frac{\partial o_4}{\partial x_3} & \frac{\partial o_4}{\partial x_4} \end{bmatrix}$$

$$\frac{\partial L}{\partial x} = \frac{\partial L}{\partial o}\frac{\partial o}{\partial x} = \begin{bmatrix} \frac{\partial L}{\partial o_1} & \frac{\partial L}{\partial o_2} & \frac{\partial L}{\partial o_3} & \frac{\partial L}{\partial o_4} \end{bmatrix} \begin{bmatrix} w_2 & w_3 & 0 & 0 \\ w_1 & w_2 & w_3 & 0 \\ 0 & w_1 & w_2 & w_3 \\ 0 & 0 & w_1 & w_2 \end{bmatrix}$$

۳. طبق معادله‌ای بخش قبل، گرادیان ورودی از بالا را به اندازه‌ی ۱ پد کنیم، سپس با فیلتر معکوس روی آن کانولوشن بزنیم و خروجی را به لایه پایین دهیم.

۴. ماتریس وزن درواقع همان ترنسپوز ماتریس وزن بخش ۱ است. خروجی نیز به اینصورت است:

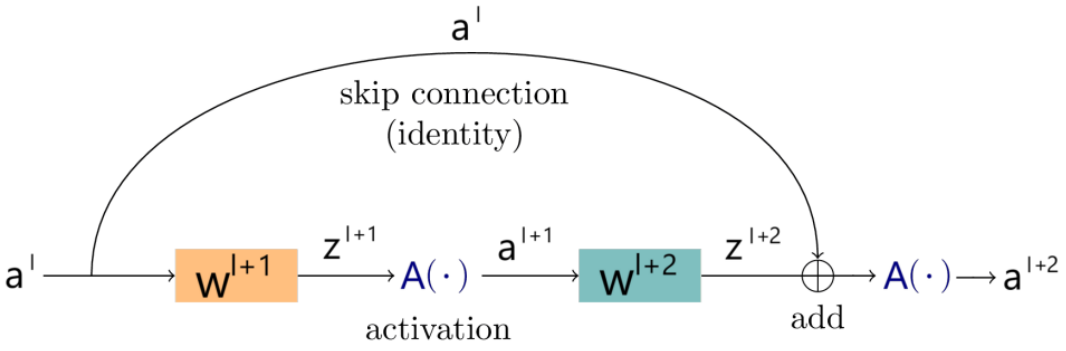
$$\begin{pmatrix} k_1x_1 \\ k_2x_1 + k_1x_2 \\ k_3x_1 + k_2x_2 + k_1x_3 \\ k_3x_2 + k_2x_3 + k_1x_4 \\ k_3x_3 + k_2x_4 \\ k_3x_4 \end{pmatrix}$$

اگر پدینگ صفر اعمال کنیم، درواقع المنت اول و آخر این بردار حذف می‌شوند.

▷

پرسش ۵. محو شدن گرادیان (۱۵ نمره)

یکی از مشکلاتی که در الگوریتم های انتشار به عقب (back propagation) وجود دارد بحث محو شدن گرادیان (Gradient Vanishing) است. این موضوع مهم و قبل از اهمیت زیاد باعث عدم آموزش درست و کامل مدل در روند آموزش می‌شود. در این مسئله قصد داریم به بررسی این اتفاق بپردازیم.



۱. ابتدا مقدار $\frac{\partial a^l}{\partial a^{l+2}}$ را بدون در نظر گرفتن skip connection محاسبه کنید.

۲. حال یکی از راه حل ها که در بسیاری از مدل ها استفاده میشود در نظر گرفتن skip connection می‌باشد. این حالت چه کمکی به مدل می‌کند؟ $\frac{\partial a^l}{\partial a^{l+2}}$ با کمک قاعده زنجیره‌ای محاسبه کنید.

۳. با توجه به نتایج دو قسمت قبل بگویید که این الگوریتم به چه صورت می‌تواند مشکل محو شدن گرادیان را حل کند.(فرض کنید که داریم $W^i < 1 - \epsilon$)

پاسخ.

۱. می دانیم:

$$\begin{aligned} a^{l+2} &= \text{Act}\left(z^{2+l}\right) \\ &= \text{Act}\left(w^{l+2}a^{l+1}\right) \\ &= \text{Act}\left(w^{l+2} \cdot \text{Act}\left(w^{l+1}a^{l}\right)\right) \end{aligned}$$

حال داریم:

$$\begin{aligned} \frac{\partial a^{l+2}}{\partial a^l} &= \frac{\partial \text{Act}\left(z^{l+2}\right)}{\partial a^l} \\ &= \frac{\partial \text{Act}\left(w^{l+2}a^{l+1}\right)}{\partial a^l} \\ &= \frac{\partial \text{Act}\left(w^{l+2} \cdot \text{Act}\left(w^{l+1}a^{l}\right)\right)}{\partial a^l} \\ &= \left[w^{l+2}\frac{\partial \text{Act}\left(w^{l+1}a^{l}\right)}{\partial a^l}\right] \cdot \text{Act'}\left(w^{l+2} \cdot \text{Act}\left(w^{l+1}a^{l}\right)\right) \\ &= \left[w^{l+2} \cdot w^{l+1} \cdot \text{Act'}\left(z^{l+1}\right)\right] \cdot \text{Act'}\left(z^{l+2}\right) \end{aligned}$$

۲. می دانیم:

$$\begin{aligned} a^{l+2} &= \text{Act}\left(a^l + z^{2+l}\right) \\ &= \text{Act}\left(a^l + w^{l+2}a^{l+1}\right) \\ &= \text{Act}\left(a^l + w^{l+2} \cdot \text{Act}\left(w^{l+1}a^{l}\right)\right) \end{aligned}$$

حال داریم:

$$\frac{\partial a^{l+2}}{\partial a^l} = \frac{\partial \text{Act}\left(a^l + z^{l+2}\right)}{\partial a^l}$$

$$= \frac{\partial \text{Act}\left(a^l + w^{l+2}a^{l+1}\right)}{\partial a^l}$$

$$= \frac{\partial \text{Act}\left(a^l + w^{l+2} \cdot \text{Act}\left(w^{l+1}a^l\right)\right)}{\partial a^l}$$

$$= \left[1 + w^{l+2}\frac{\partial \text{Act}\left(w^{l+1}a^l\right)}{\partial a^l}\right] \cdot \text{Act'}\left(a^l + w^{l+2} \cdot \text{Act}\left(w^{l+1}a^l\right)\right)$$

$$= \left[1 + w^{l+2} \cdot w^{l+1} \cdot \text{Act'}\left(z^{l+1}\right)\right] \cdot \text{Act'}\left(a^l + z^{l+2}\right)$$

۳. مشخص است که اضافه شدن این ارتباط در مدل باعث شده است که در زمان انتشار گرادیان به عقب، علاوه بر مسیر معمول برای گرادیان‌ها، گرادیان بدون ضرب شدن در وزن‌های w^{l+1} و w^{l+2} و فقط ضرب شدن در لایه فعال‌سازی به عقب منتقل می‌شود. این موضوع باعث می‌شود که مشکل محو شدن گرادیان ناشی از ضرب وزن‌ها با مقادیر بزرگ به صفر رفته و با انتشار گرادیان‌ها به عقب، اندازه گرادیان‌ها به صفر میل نکند(توجه کنید به فرض داده شده در سوال). توجه کنید که همچنان باید در انتخاب تابع فعال‌سازی دقت داشت.

◁

پرسش ۶. MobileNet (۳۵ نمره)

معماری‌های MobileNet (شامل نسخه‌های V1، V2، و V3) از جمله شبکه‌های عصبی پیچشی سبک‌وزن هستند که به طور خاص برای اجرا بر روی دستگاه‌های کم‌مصرف مانند گوشی‌های هوشمند و سخت‌افزارهای لبه طراحی شده‌اند. این مدل‌ها با کاهش تعداد محاسبات و پارامترها، بدون افت چشمگیر در دقت، توانسته‌اند به تعادلی میان کارایی و عملکرد دست یابند. از مهم‌ترین نوآوری‌های به‌کاررفته در این معماری‌ها می‌توان به کانولوشن‌های عمقی قابل تفکیک (Depthwise Separable Convolutions)، بلوک‌های باقیمانده معکوس (Inverted Residual Blocks) دارای گلوگاه‌های خطی (Linear Bottlenecks)، مکانیزم‌های فشرده‌سازی و تحریک (SE Blocks) و استفاده از جستجوی معماری عصبی (NAS) اشاره کرد. در این سوال به بررسی برخی از این موارد می‌پردازیم.(لازم به ذکر است برای حل این سوال پیشنهاد اکید میشود از سرچ در منابع مختلف برای mobilenet بهره ببرید)

۱) کانولوشن‌های عمقی قابل تفکیک و تئوری تقریب
کانولوشن‌های عمقی قابل تفکیک سنگ بنای شبکه‌های پیچشی سبک‌وزن هستند که باعث کاهش قابل توجه هزینه‌های محاسباتی می‌شوند و در عین حال ظرفیت نمایشی منطقی را حفظ می‌کنند.

۱-۱. کانولوشن عمقی قابل تفکیک را بطور کامل حین مقایسه با کانولوشن عادی، توضیح دهید(به صورت ریاضی).

۱-۲. عمل ریاضی کانولوشن استاندارد را درنظر بگیرید و آن را بصورت مجموع تنسورهای رتبه یک با استفاده از تجزیه مقادیر تکین بیان کنید. توضیح دهید این تجزیه چگونه به ساختار کانولوشن عمقی قابل تفکیک مرتبط است.

۱-۳. نسبت کاهش FLOPs در کانولوشن‌های عمقی قابل تفکیک نسبت به کانولوشن‌های استاندارد را برای اندازه ورودی $H \times W$، تعداد کانال‌های ورودی C_{in}، تعداد کانال‌های خروجی C_{out} و اندازه فیلتر K را بطور پارامتری محاسبه کنید. آیا بین ظرفیت نمایشی و هزینه‌های محاسباتی ناشی از این تقریب تضادی وجود دارد؟

۲) بلوک‌های باقیمانده معکوس و تحلیل جریان گرادیان
بلوک‌های باقیمانده معکوس، که در MobileNetV2 معرفی شدند، با ترکیب باقی‌مانده معکوس و گلوگاه‌های خطی پیشرفتی در شبکه‌های پیچشی سبک‌وزن ایجاد کردند و باعث بهبود ظرفیت نمایشی در حالی که هزینه‌های محاسباتی کاهش می‌یابد، شدند.

۲-۱. بلوک‌های باقیمانده معکوس را در حین مقایسه با بلوک‌های باقیمانده عادی (ResNet)، توضیح دهید.

۲-۲. آیا این ادعا که کانولوشن‌های عمقی قابل تفکیک اطلاعات مکانی را حفظ می‌کنند، در حالی که گلوگاه‌های خطی اطلاعات کانالی را حفظ می‌کنند، صحیح است؟

۲-۳. تحلیل کنید که چگونه ضریب انبساط t بر جریان گرادیان و پایداری بهینه‌سازی تأثیر می‌گذارد. ثابت کنید که افزایش t خطر ناپدید شدن گرادیان‌ها را کاهش می‌دهد، اما هزینه‌های محاسباتی را افزایش می‌دهد. مقدار بهینه t را که تعادلی بین جریان گرادیان و کارایی ایجاد می‌کند، بر چه اساسی می‌توان انتخاب کرد؟

۳) مکانیزم‌های فشرده‌سازی و تحریک و بازنگری ویژگی‌های کانالی

بلوک‌های فشرده‌سازی و تحریک، که در MobileNetV3 استفاده شدند، تنظیم تطبیقی پاسخ‌های ویژگی در سطح کانال افزایش می‌دهند. این بلوک‌ها با استفاده از یک مکانیزم توجه (attention) وزن‌های کانال‌ها را مطابق با اطلاعات جهانی موجود در نقشه‌های ویژگی ورودی خود تنظیم می‌کنند.

۳-۱. اعمالی که در یک بلوک فشرده‌سازی و تحریک در یک لایه رخ می‌دهد را به صورت ریاضی فرمول بندی کنید. فرض کنید نقش ویژگی ورودی این بلوک با ابعاد $C \times H \times W$ است.

۳-۲. ثابت کنید که بلوک‌های SE ظرفیت نمایشی را با بازنگری تطبیقی پاسخ‌های ویژگی‌های کانالی بهبود می‌دهند. از تئوری اطلاعات برای اندازه‌گیری اطلاعات متقابل بین ورودی و خروجی بلوک SE استفاده کنید. توضیح دهید که ضریب کاهش r چگونه بر تضاد بین هزینه‌های محاسباتی و عملکرد تأثیر می‌گذارد.

۳-۳. FLOPs اضافه شده توسط یک بلوک SE با $r = 16$ برای $C = 64$ را محاسبه کنید. این مقدار را با کل FLOPs یک کانولوشن عمقی قابل تفکیک با پارامترهای مشابه مقایسه کنید.

۴) جستجوی معماری عصبی (NAS) و تکنیک کوچک‌سازی پیش‌رونده

جستجوی معماری عصبی (NAS) برای بهینه‌سازی معماری MobileNetV3 استفاده شد و منجر به بهترین‌سطح عملکرد در دستگاه‌های موبایل شد.

۴-۱. این فرآیند را بطور کامل تحلیل کنید. فضای جستجو برای NAS در MobileNetV3 چه بوده است؟ از نظریه گراف برای مدل‌سازی فضای جستجو به عنوان یک گراف جهت‌دار بدون دور (DAG) چگونه می‌توان استفاده کرد؟

۴-۲. تکنیک کوچک‌سازی پیش‌رونده استفاده‌شده در NAS را توضیح دهید.

پاسخ.

۱) کانولوشن‌های عمقی قابل تفکیک و تئوری تقریب

۱-۱. * در کانولوشن استاندارد، هر فیلتر به تمام کانال‌های ورودی متصل است و یک ترکیب خطی از تمامی این کانال‌ها را با استفاده از کرنلی به ابعاد $K \times K$ استخراج می‌کند. به‌صورت ریاضی، اگر ورودی دارای ابعاد (C_{in}, H, W) باشد و تعداد خروجی‌ها C_{out} باشد، عملیات کانولوشن استاندارد به‌صورت زیر تعریف می‌شود:

$$Y_i(p) = \sum_{j=1}^{C_{in}} \sum_{u=1}^{K} \sum_{v=1}^{K} W_{i,j,u,v}^{(std)} X_j(p + (u, v))$$

که در آن $W^{(std)}_{i,j,u,v}$ وزن‌های فیلتر کانولوشن استاندارد برای خروجی i و کانال ورودی j است. این فرمول نشان می‌دهد که هر فیلتر به‌طور همزمان هم عملیات ترکیب کانال‌ها (Feature Mixing) و هم فیلترینگ مکانی (Spatial Filtering) را انجام می‌دهد.

* در مقابل، کانولوشن‌های عمقی‌ـ‌قابل‌تفکیک (Depthwise Separable Convolutions) این عملیات را به دو مرحله مجزا تقسیم می‌کنند:

- **کانولوشن عمقی (Depthwise Convolution)**: در این مرحله، هر کانال ورودی به‌طور مستقل با یک فیلتر $K \times K$ خاص خود پردازش می‌شود. به‌صورت ریاضی:

$$(X \star D)_j(p) = \sum_{u=1}^{K} \sum_{v=1}^{K} D_j(u,v) X_j(p + (u,v))$$

که در آن D_j کرنل مختص کانال j است. برخلاف کانولوشن استاندارد، این مرحله ترکیب اطلاعات بین کانال‌ها را انجام نمی‌دهد و فقط فیلترینگ مکانی را اجرا می‌کند.

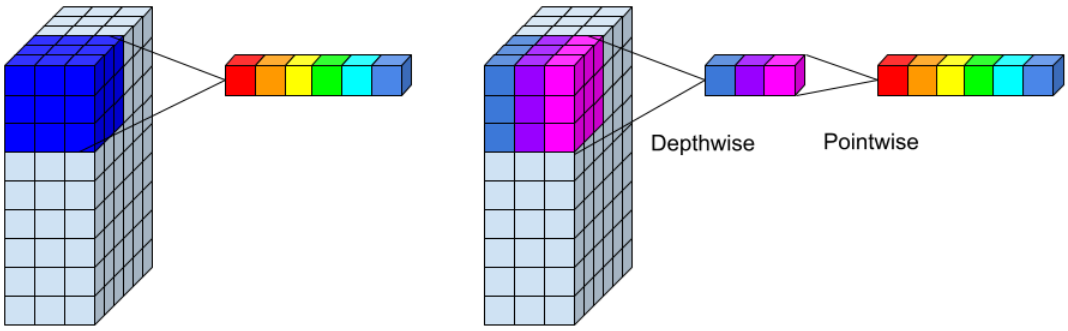
- **کانولوشن نقطه‌ای (Pointwise Convolution)**: پس از اعمال فیلترهای مکانی، یک کانولوشن 1×1 برای ترکیب کانال‌ها به کار گرفته می‌شود. این عمل که با وزن‌های $P_{i,j}$ انجام می‌شود، مقدار نهایی را برای هر خروجی i محاسبه می‌کند:

$$Y_i(p) = \sum_{j=1}^{C_{in}} P_{i,j} [X \star D]_j(p)$$

در این مرحله، اطلاعات بین کانال‌ها مخلوط می‌شود، اما هیچگونه پردازش مکانی صورت نمی‌گیرد.

* **مقایسه کانولوشن استاندارد و عمقی‌ـ‌قابل‌تفکیک**:

- در کانولوشن استاندارد، هر فیلتر شامل C_{in} کرنل با ابعاد $K \times K$ است، که باعث پیچیدگی محاسباتی بالا و افزایش تعداد پارامترها می‌شود.
- در کانولوشن عمقی‌ـ‌قابل‌تفکیک، ابتدا هر کانال مستقل از بقیه پردازش می‌شود، سپس یک ترکیب خطی از خروجی‌ها اعمال می‌گردد. این امر منجر به کاهش شدید تعداد محاسبات و پارامترها می‌شود.
- کانولوشن استاندارد یک فضای جستجوی گسترده‌تر برای یادگیری الگوهای پیچیده‌تر فراهم می‌کند، اما کانولوشن عمقی‌ـ‌قابل‌تفکیک در بسیاری از موارد کارایی مشابهی با هزینه‌ی محاسباتی بسیار کمتر ارائه می‌دهد.



a) Standard Convolution b)Depthwise Sepratable Convolution

۱ - ۲. تجزیه کانولوشن استاندارد با SVD و ارتباط با کانولوشن عمقی قابل تفکیک

* **کانولوشن استاندارد:** فرض کنید ورودی $I \in \mathbb{R}^{H_{\text{in}} \times W_{\text{in}} \times C_{\text{in}}}$ و هسته (کرنل) $K \in \mathbb{R}^{k_H \times k_W \times C_{\text{in}} \times C_{\text{out}}}$ باشد. خروجی $O \in \mathbb{R}^{H_{\text{out}} \times W_{\text{out}} \times C_{\text{out}}}$ برای کانال خروجی j و موقعیت (x, y) به صورت زیر محاسبه می‌شود (بدون در نظر گرفتن بایاس):

$$O[x, y, j] = \sum_{i=0}^{C_{\text{in}}-1} \sum_{h=0}^{k_H-1} \sum_{w=0}^{k_W-1} K[h, w, i, j] \cdot I[x + h', y + w', i]$$

(توجه: $x + h', y + w'$ اندیس‌های مناسب در ورودی I با در نظر گرفتن padding و stride هستند.) این عملیات همزمان اطلاعات مکانی و بین کانالی را ترکیب می‌کند.

* **تجزیه با استفاده از SVD:**

- ابتدا تنسور کرنل ۴ بعدی K را به یک ماتریس ۲ بعدی تبدیل می‌کنیم. یک روش رایج، تبدیل آن به ماتریس $\widetilde{K} \in \mathbb{R}^{C_{\text{out}} \times (k_H k_W C_{\text{in}})}$ است، که هر سطر آن نماینده وزن‌های یک فیلتر خروجی است.

- سپس تجزیه مقادیر تکین (SVD) را روی ماتریس $\widetilde{K}$ اعمال می‌کنیم:

$$\widetilde{K} = U \Sigma V^\top = \sum_{r=1}^{R} \sigma_r \mathbf{u}_r \mathbf{v}_r^\top$$

که R رتبه ماتریس $\widetilde{K}$ است، σ_r مقادیر تکین، $\mathbf{u}_r \in \mathbb{R}^{C_{\text{out}}}$ بردارهای تکین چپ (مرتبط با ترکیب کانال‌های خروجی) و $\mathbf{v}_r \in \mathbb{R}^{k_H k_W C_{\text{in}}}$ بردارهای تکین راست (مرتبط با فیلتر کردن مکانی و کانال‌های ورودی) هستند.

- هر جمله $\sigma_r \mathbf{u}_r \mathbf{v}_r^\top$ یک ماتریس رتبه ۱ است. با بازآرایی $\mathbf{v}_r$ به شکل تنسور $(k_H, k_W, C_{\text{in}})$، می‌توان عمل کانولوشن اصلی را به صورت مجموعی از کانولوشن‌ها با کرنل‌های مؤثر رتبه ۱ (در فضای بازآرایی شده) بیان کرد:

$$O \approx \sum_{r=1}^{R} \text{Conv}(\text{Input} = I, \text{Kernel} = \text{reshape}(\sigma_r \mathbf{u}_r \mathbf{v}_r^\top))$$

این نشان می‌دهد که کانولوشن استاندارد قابل تجزیه به مولفه‌هایی است که به نوعی عملیات مکانی-ورودی و ترکیب خروجی را جدا می‌کنند.

* **کانولوشن عمقی قابل تفکیک (Depthwise Separable Convolution):** این ساختار عملیات را به دو مرحله صریح تفکیک می‌کند:

- کانولوشن عمقی (Depthwise): اعمال فیلترهای مجزای مکانی $(k_H, k_W, 1)$ روی هر کانال ورودی C_{in} به طور مستقل. خروجی دارای ابعاد $(H_{\text{out}}, W_{\text{out}}, C_{\text{in}})$ است. (فقط پردازش مکانی)
- کانولوشن نقطه‌ای (Pointwise): اعمال فیلترهای 1×1 با ابعاد $(1, 1, C_{\text{in}}, C_{\text{out}})$ برای ترکیب خروجی مرحله قبل در راستای کانال‌ها. (فقط ترکیب کانال‌ها)

* **ارتباط تجزیه SVD با کانولوشن عمقی قابل تفکیک:**

- تجزیه SVD نشان می‌دهد که کرنل کانولوشن استاندارد می‌تواند به صورت مجموعی از مولفه‌های رتبه ۱ بیان شود، که هر مولفه به طور ضمنی نوعی جداسازی بین جنبه‌های مکانی/ورودی ($\mathbf{v}_r$) و جنبه‌های ترکیب خروجی ($\mathbf{u}_r$) دارد.
- این تجزیه، ایده اصلی پشت کانولوشن عمقی قابل تفکیک را توجیه ریاضی می‌کند: یعنی امکان‌پذیری شکستن (فاکتورگیری) عملیات کانولوشن به بخش‌های مرتبط با فیلترینگ مکانی و ترکیب کانالی.
- کانولوشن عمقی قابل تفکیک، این ایده فاکتورگیری را به صورت یک معماری صریح و کارآمد پیاده‌سازی می‌کند که در آن جداسازی بین عملیات مکانی (Depthwise) و ترکیب کانالی (Pointwise) به طور کامل و ساختاری انجام می‌شود.
- تفاوت کلیدی: SVD یک تجزیه ریاضی (که می‌تواند برای تقریب‌سازی استفاده شود) ارائه می‌دهد، در حالی که کانولوشن عمقی قابل تفکیک یک ساختار معماری جایگزین است که ذاتاً عملیات را به شکلی خاص تفکیک می‌کند.

۱-۳. * **کاهش پیچیدگی محاسباتی (FLOPs Reduction):**

- در یک کانولوشن استاندارد، تعداد عملیات شناور مورد نیاز به‌صورت زیر محاسبه می‌شود:

$$\text{FLOPs}_{\text{std}} = HWC_{\text{out}}C_{\text{in}}K^2$$

- در کانولوشن جداسازی‌پذیر عمقی، محاسبات به دو بخش تقسیم می‌شود:
- **مرحله عمقی**: در این مرحله، هر کانال به‌طور مستقل پردازش شده و پیچیدگی آن برابر است با:

$$\text{FLOPs}_{\text{depthwise}} = HWC_{\text{in}}K^2$$

- **مرحله نقطه‌ای**: این مرحله، کانال‌های خروجی را ترکیب می‌کند و پیچیدگی آن به‌صورت زیر است:

$$\text{FLOPs}_{\text{pointwise}} = HWC_{\text{in}}C_{\text{out}}$$

- بنابراین، پیچیدگی کلی کانولوشن جداسازی‌پذیر عمقی برابر است با:

$$\text{FLOPs}_{\text{dw-sep}} = HWC_{\text{in}}(K^2 + C_{\text{out}})$$

- نسبت کاهش تعداد عملیات در مقایسه با کانولوشن استاندارد به‌صورت زیر است:

$$\frac{\text{FLOPs}_{\text{dw-sep}}}{\text{FLOPs}_{\text{std}}} = \frac{C_{\text{in}}(K^2 + C_{\text{out}})}{C_{\text{out}}C_{\text{in}}K^2} = \frac{K^2 + C_{\text{out}}}{K^2 C_{\text{out}}}$$

- برای تنظیمات معمولی (مثلاً $K = 3$ و $C_{\text{out}} = 50$)، این کاهش قابل توجه است. برای مثال، در این شرایط، کانولوشن جداسازی‌پذیر عمقی تنها $1/7.6 \approx 13\%$ از FLOPs کانولوشن استاندارد را نیاز دارد.

- در حد $C_{\text{out}} \gg K^2$، این نسبت به $1/K^2$ میل می‌کند (مثلاً برای 3×3، حدوداً $1/9$ یا کاهش 89% در تعداد عملیات).

* **کاهش تعداد پارامترها**:

- در کانولوشن استاندارد، تعداد پارامترها برابر است با:

$$C_{\text{out}}C_{\text{in}}K^2$$

- در کانولوشن جداسازی‌پذیر عمقی، تعداد پارامترها کاهش یافته و برابر است با:

$$C_{\text{in}}K^2 + C_{\text{in}}C_{\text{out}}$$

- این کاهش پارامترها در مدل‌هایی مانند MobileNet و Xception منجر به طراحی شبکه‌های سبک‌تر و سریع‌تر شده است.

* **موازنه بین ظرفیت نمایش و هزینه محاسباتی (Capacity vs. Cost Trade-off)**:

- مزیت کانولوشن جداسازی‌پذیر عمقی این است که باعث کاهش قابل توجه پیچیدگی محاسباتی و تعداد پارامترها می‌شود.

- اما این روش یک محدودیت نیز دارد: یک کانولوشن استاندارد می‌تواند فیلترهای فضایی منحصربه‌فردی برای هر جفت ورودی–خروجی (j, i) یاد بگیرد، در حالی که در کانولوشن جداسازی‌پذیر عمقی، تمام خروجی‌ها یک فیلتر فضایی یکسان را برای یک کانال ورودی به اشتراک می‌گذارند و فقط توسط وزن‌های نقطه‌ای مقیاس‌بندی می‌شوند.

- این مسئله منجر به کاهش قدرت نمایش شبکه می‌شود، زیرا بعضی از الگوهای پیچیده با مرتبه بالا در ماتریس کانولوشن استاندارد ممکن است نیاز به چندین لایه از کانولوشن‌های جداسازی‌پذیر داشته باشند تا به‌درستی مدل‌سازی شوند.

- به بیان دیگر، کانولوشن جداسازی‌پذیر عمقی یک تقریب **کم‌مرتبه** (Low-rank Approximation) از یک کانولوشن کامل است.

- اگر تابع هدف دارای درهم‌تنیدگی قابل توجه بین کانال‌ها و فضا باشد، یک لایه جداسازی‌پذیر ممکن است نتواند آن را به‌درستی مدل‌سازی کند و در نتیجه عملکرد مدل کاهش یابد.

- با این حال، در عمل بسیاری از ویژگی‌های استخراج شده در شبکه‌های کانولوشنی به‌طور قابل توجهی **قابل تجزیه** هستند. یعنی بسیاری از الگوهای فضایی که در کانال‌های مختلف اعمال می‌شوند، می‌توانند با ترکیب خطی ساده‌ای مدل‌سازی شوند.

- همچنین، استفاده از **چندین لایه از کانولوشن‌های جداسازی‌پذیر** می‌تواند تقریباً همان عملکرد یک کانولوشن استاندارد را ارائه دهد، در حالی که هزینه محاسباتی را به‌طور قابل توجهی کاهش می‌دهد.

- بنابراین، کانولوشن جداسازی‌پذیر عمقی تعادلی میان **کاهش هزینه محاسباتی و ظرفیت نمایش مدل** برقرار می‌کند. این تکنیک زمانی مناسب است که نمایش تابع مورد نظر تقریباً درون یک ساختار کم‌مرتبه قرار گیرد.

- شواهد تجربی از شبکه‌هایی مانند MobileNet و Xception نشان می‌دهد که در اکثر موارد، این کاهش درجه آزادی منجر به افت عملکرد قابل توجهی نمی‌شود و به دلیل کاهش هزینه محاسباتی، مزایای آن بر محدودیت‌هایش غلبه دارد.

۲) بلوک‌های باقیمانده معکوس و تحلیل جریان گرادیان

۲-۱. – بلوک‌های باقیمانده عادی (ResNet):

* در شبکه‌های ResNet، بلوک‌های باقیمانده شامل یک مسیر میان‌بر (skip connection) هستند که مستقیماً ورودی را به خروجی اضافه می‌کند.

* این بلوک‌ها معمولاً دارای یک ساختار bottleneck هستند که شامل سه لایه است: یک لایه 1×1 برای کاهش تعداد کانال‌ها، یک لایه 3×3 برای پردازش فضایی، و یک لایه 1×1 برای بازیابی تعداد کانال‌های اصلی.

* مسیر میان‌بر کمک می‌کند که گرادیان‌ها در حین پس‌انتشار (backpropagation) حفظ شوند و مسأله ناپدید شدن گرادیان را کاهش می‌دهد.

– بلوک‌های باقیمانده معکوس (MobileNetV2):

* در MobileNetV2، طراحی بلوک‌های باقیمانده معکوس، روند کاهش و افزایش تعداد کانال‌ها را معکوس می‌کند.

* ابتدا ورودی با یک لایه 1×1 به یک فضای ویژگی با بعد بالاتر (expansion) نگاشته می‌شود.

* سپس یک کانولوشن عمقی جداپذیر (depthwise convolution) روی فضای ویژگی گسترده شده اعمال می‌شود که باعث کاهش محاسبات می‌شود.

* در نهایت، یک لایه 1×1 دیگر، ویژگی‌های استخراج‌شده را دوباره به تعداد کانال‌های اولیه فشرده می‌کند (linear bottleneck).

* مسیر میان‌بر در اینجا بین لایه‌های فشرده شده قرار دارد، نه در فضای گسترده، که باعث می‌شود مدل اطلاعات بیشتری را در حین پردازش حفظ کند.

– مقایسه بلوک‌های باقیمانده عادی و معکوس:

* در ResNet، مسیر میان‌بر روی فضای ویژگی‌های **وسیع** اعمال می‌شود، در حالی که در MobileNetV2، مسیر میان‌بر روی **فضای فشرده شده** اعمال می‌شود.

* بلوک‌های ResNet از کانولوشن‌های استاندارد استفاده می‌کنند که پارامتر و محاسبات بیشتری دارند، در حالی که بلوک‌های MobileNetV2 از کانولوشن‌های جداپذیر عمقی استفاده می‌کنند که محاسبات را کاهش می‌دهد.

* در MobileNetV2، کاهش بعدی (bottleneck) بدون تابع غیرخطی انجام می‌شود تا اطلاعات بیشتری حفظ شود، در حالی که در ResNet، این مرحله شامل تابع غیرخطی است.

۲-۲. گلوگاه خطی (Linear Bottleneck) که در بلوک‌های باقیمانده معکوس (Inverted Residual Blocks) استفاده می‌شود، یک لایه 1×1 کانولوشن بدون تابع غیرخطی (مانند ReLU) است. دلیل این امر این است که اگر یک لایه کانولوشن 1×1 همراه با ReLU به‌کار رود، بسیاری از مقادیر ویژگی ممکن است صفر شوند و در نتیجه اطلاعات از دست برود. با استفاده از یک گلوگاه خطی، فشرده‌سازی کانالی به‌صورت یک تبدیل خطی بدون حذف اطلاعات انجام می‌شود. این بدان معناست که در این مرحله، ویژگی‌های کانالی که در مرحله انبساط تولید شده‌اند، تا حد ممکن بدون تحریف حفظ می‌شوند. به این ترتیب، گلوگاه خطی مانع از از دست رفتن اطلاعات مفید کانالی می‌شود که ممکن است در صورت استفاده از تابع غیرخطی تخریب شوند.

- **نتیجه‌گیری:** این ادعا که «کانولوشن‌های جداساز عمقی اطلاعات مکانی را حفظ می‌کنند، در حالی که گلوگاه‌های خطی اطلاعات کانالی را حفظ می‌کنند» **صحیح** است.

  - کانولوشن‌های جداساز عمقی، به دلیل عدم ترکیب اطلاعات بین کانالی، ساختار فضایی ویژگی‌ها را در هر کانال دست‌نخورده نگه می‌دارند.
  - گلوگاه‌های خطی، به دلیل نبود تابع غیرخطی، تضمین می‌کنند که اطلاعات کانالی که در مرحله انبساط استخراج شده است، بدون حذف یا از بین رفتن حفظ شود.

بنابراین، ترکیب این دو مؤلفه در بلوک‌های باقیمانده معکوس باعث می‌شود که مدل بتواند ضمن کاهش هزینه‌های محاسباتی، اطلاعات حیاتی فضایی و کانالی را حفظ کند.

۲-۳. – **تأثیر ضریب انبساط t بر جریان گرادیان و پایداری بهینه‌سازی:** ضریب انبساط t در بلوک‌های باقیمانده معکوس، تعداد کانال‌های ویژگی را در سطح گلوگاه گسترش می‌دهد و تأثیر مستقیمی بر جریان گرادیان دارد. بلوک‌های باقیمانده معکوس از سه مرحله اصلی تشکیل شده‌اند:

  * گسترش: x با کانال‌های ورودی C_{in} از طریق یک کانولوشن 1×1 به $C_{\text{exp}} = t \cdot C_{in}$ گسترش می‌یابد.
  * کانولوشن عمقی: اعمال کانولوشن 3×3 (یا 5×5) به‌صورت مستقل روی هر کانال.
  * فشرده‌سازی: کاهش ابعاد از C_{exp} به C_{out} با یک کانولوشن 1×1 بدون غیرخطیت.

مقدار t تأثیر مستقیمی بر جریان گرادیان و نرخ یادگیری دارد. افزایش t باعث تقویت انتشار گرادیان می‌شود، زیرا ویژگی‌های با ابعاد بالاتر ظرفیت بیشتری برای حمل اطلاعات و حفظ گرادیان‌ها دارند.

– **اثبات کاهش خطر ناپدید شدن گرادیان‌ها با افزایش t:** گرادیان خروجی q نسبت به ورودی x به صورت زیر محاسبه می‌شود:

$$\frac{\partial L}{\partial x} = W_f^T \Big[g^* \Big(W_p^T \frac{\partial L}{\partial q} \Big) \Big]$$

که در آن:

  * W_f ماتریس کانولوشن 1×1 انبساطی است.
  * g^* عملگر بازگشتی کانولوشن عمقی است.
  * W_p ماتریس کانولوشن 1×1 فشرده‌سازی است.

افزایش t باعث می‌شود که W_f و W_p از نظر ابعادی گسترده‌تر شوند، به‌طوری‌که:

$$W_f \in \mathbb{R}^{tC_{in} \times C_{in}}, \quad W_p \in \mathbb{R}^{C_{out} \times tC_{in}}$$

این امر موجب افزایش دامنه ویژه‌ی مقادیر ماتریس ترکیبی W_pW_f می‌شود، و مقدار کمینه‌ی مقدار ویژه کاهش نمی‌یابد، که باعث جلوگیری از محو شدن گرادیان‌ها می‌شود. علاوه بر این، در صورت استفاده از ReLU، افزایش t احتمال خاموش شدن کانال‌ها را کاهش می‌دهد، زیرا شبکه در یک فضای ویژگی با ابعاد بالاتر عمل می‌کند که احتمال صفر شدن خروجی‌ها را کاهش می‌دهد.

- **اثبات افزایش هزینه‌های محاسباتی با افزایش t:** هزینه‌ی محاسباتی هر بلوک را می‌توان به صورت زیر محاسبه کرد:

$$\text{FLOPs} = HW\left(C_{in}t + C_{in}tK^2 + C_{in}tC_{out}\right)$$

که در آن:

* H و W ابعاد فضایی ویژگی‌ها هستند.
* K^2 اندازه‌ی فیلتر کانولوشن عمقی است.

با افزایش t، هر سه جمله‌ی بالا به‌صورت خطی افزایش می‌یابند، در نتیجه هزینه‌ی محاسباتی بلوک به‌طور مستقیم متناسب با t افزایش می‌یابد.

- **انتخاب مقدار بهینه‌ی t:** مقدار t باید به‌گونه‌ای تنظیم شود که هم جریان گرادیان حفظ شود و هم هزینه‌ی محاسباتی بهینه باشد. معیارهای انتخاب مقدار بهینه‌ی t عبارت‌اند از:

* پایداری گرادیان: مقدار t نباید خیلی کوچک باشد، زیرا در این صورت اطلاعات در سطح گلوگاه فشرده شده و کانال‌های زیادی در ReLU صفر می‌شوند که منجر به ناپدید شدن گرادیان‌ها می‌شود.
* محدودیت‌های محاسباتی: مقدار t نباید بیش از حد بزرگ باشد، زیرا باعث افزایش تعداد عملیات شناور (FLOPs) می‌شود که منجر به کاهش کارایی در سخت‌افزارهای موبایل می‌شود.
* نتایج تجربی: در MobileNetV۲ مقدار $t = 6$ به عنوان مقدار بهینه شناسایی شد که تعادل مناسبی بین دقت و کارایی ایجاد می‌کند.

به‌طور کلی، مقدار t باید چنان تنظیم شود که گرادیان‌ها پایدار بمانند و درعین‌حال بار محاسباتی اضافی تحمیل نشود. یک روش عملی برای انتخاب t استفاده از جستجوی شبکه‌ای یا تنظیم تجربی بر اساس معماری و سخت‌افزار هدف است.

۳) مکانیزم‌های فشرده‌سازی و تحریک و بازنگری ویژگی‌های کانالی

۳-۱. * **فرمول‌بندی ریاضی بلوک فشرده‌سازی و تحریک:** بلوک‌های فشرده‌سازی و تحریک شامل دو مرحله‌ی اصلی هستند:

· **مرحله‌ی فشرده‌سازی (Squeeze):** در این مرحله، اطلاعات مکانی از ویژگی‌های ورودی حذف شده و تنها اطلاعات سطح کانال حفظ می‌شود. این کار با یک Global Average Pooling (GAP) انجام می‌شود. فرض کنید $\mathbf{U} \in \mathbb{R}^{C \times H \times W}$ یک نقشه

ویژگی با C کانال و ابعاد مکانی $H \times W$ باشد. عملیات فشرده‌سازی به‌صورت زیر تعریف می‌شود:

$$s_i = F_{\text{sq}}(U)_i = \frac{1}{HW} \sum_{u=1}^{H} \sum_{v=1}^{W} U_i(u, v), \quad \forall i \in \{1, 2, ..., C\}$$

که در آن، $s \in \mathbb{R}^{C \times 1}$ برداری است که هر درایه‌ی آن نمایانگر میانگین شدت مقدار در کانال مربوطه است.

- **مرحله‌ی تحریک (Excitation):** این مرحله برای تولید مقادیر وزن‌دهی شده‌ی تطبیقی برای کانال‌ها استفاده می‌شود. این کار توسط یک شبکه کاملاً متصل (FC) دو لایه‌ای صورت می‌گیرد که بردار فشرده‌شده s را به یک بردار وزن‌دهی $\mathbf{z}$ نگاشت می‌کند. این فرایند به صورت زیر مدل‌سازی می‌شود:

$$z = \sigma(W_2 \delta(W_1 s))$$

که در آن:

- $W_1 \in \mathbb{R}^{C/r \times C}$ ماتریس وزن لایه‌ی اول (کاهش ابعاد با ضریب کاهش r).
- $\delta(\cdot)$ تابع فعال‌سازی ReLU است.
- $W_2 \in \mathbb{R}^{C \times C/r}$ ماتریس وزن لایه‌ی دوم (افزایش ابعاد به C کانال اولیه).
- $\sigma(\cdot)$ تابع فعال‌سازی sigmoid است که خروجی را به بازه‌ی $(0, 1)$ نگاشت می‌کند تا ضرایب وزنی برای کانال‌ها را تولید کند.

- **مرحله‌ی بازتوزیع ویژگی‌ها (Scaling):** در این مرحله، ضرایب وزنی حاصل از مرحله‌ی تحریک، بر روی ویژگی‌های اولیه اعمال شده و وزن‌دهی کانالی انجام می‌شود:

$$\tilde{U}_i(u, v) = z_i \cdot U_i(u, v), \quad \forall i \in \{1, 2, \ldots, C\}, \quad \forall (u, v) \in \{1, \ldots, H\} \times \{1, \ldots, W\}$$

که در آن، $\tilde{U}$ همان ویژگی‌های مقیاس‌بندی شده‌ی خروجی است که پس از تنظیم وزن‌های کانالی به مدل ارسال می‌شود.

* **توضیح مدل‌سازی ریاضی:** فرایند فوق شامل سه تبدیل متوالی است:

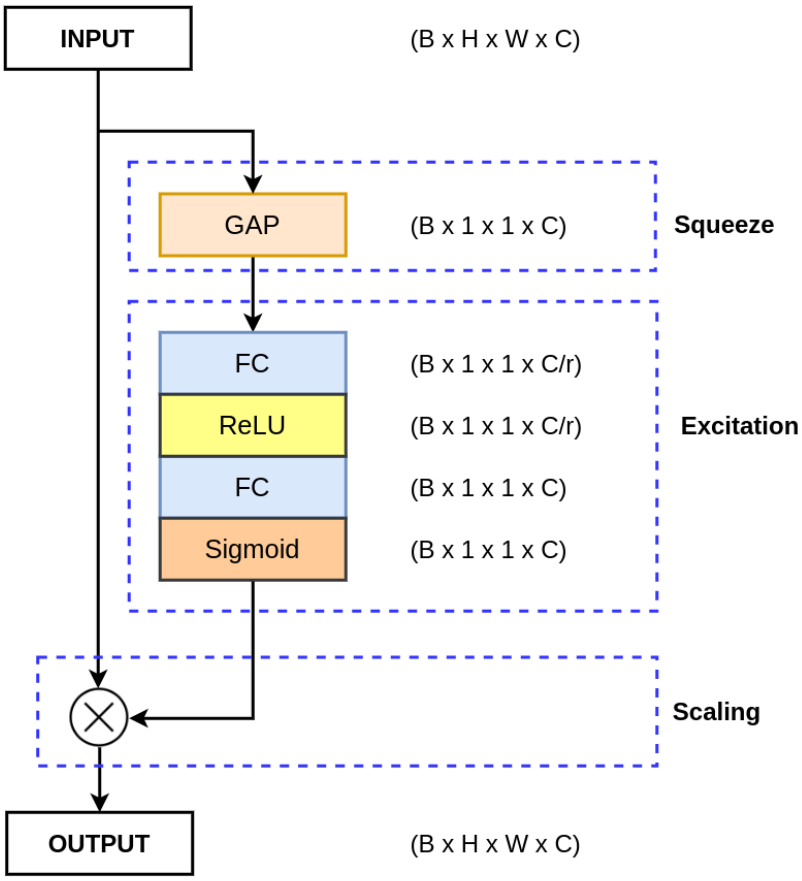
۱. تبدیل $F_{\text{sq}} : \mathbb{R}^{C \times H \times W} \to \mathbb{R}^{C \times 1}$ (فشرده‌سازی مکانی).

۲. تبدیل $F_{\text{ex}} : \mathbb{R}^{C \times 1} \to \mathbb{R}^{C \times 1}$ (تولید ضرایب مقیاس).

۳. تبدیل $F_{\text{scale}} : \mathbb{R}^{C \times H \times W} \times \mathbb{R}^{C \times 1} \to \mathbb{R}^{C \times H \times W}$ (بازتوزیع ویژگی‌ها).

بنابراین، فرمول‌بندی نهایی بلوک فشرده‌سازی و تحریک به شکل زیر خلاصه می‌شود:

$$\tilde{U} = F_{\text{scale}}(U, F_{\text{ex}}(F_{\text{sq}}(U)))$$



۳-۲. اثبات بهبود ظرفیت نمایشی بلوک‌های SE:

* **اطلاعات متقابل (Mutual Information):**

$$I(X;Y) = H(X) - H(X|Y).$$

* بلوک SE با کاهش $H(X|Y)$ و حفظ $H(X)$، اطلاعات متقابل $I(X;Y)$ را افزایش می‌دهد و ظرفیت نمایشی مدل را بهبود می‌بخشد.

تأثیر ضریب کاهش r:

* **هزینه‌های محاسباتی:**
  - کاهش r: افزایش هزینه‌های محاسباتی.
  - افزایش r: کاهش هزینه‌های محاسباتی.
* **عملکرد:**
  - کاهش r: افزایش دقت.
  - افزایش r: کاهش دقت.
* **تضاد:** انتخاب r باید بر اساس توازن بین هزینه‌های محاسباتی و عملکرد انجام شود.

۳-۳. ❊ **محاسبه FLOPs بلوک SE:**

· فشرده‌سازی (Squeeze):

$$\text{FLOPs}_{\text{Squeeze}} = H \times W \times C.$$

· تحریک (Excitation):

$$\text{FLOPs}_{\text{Excitation}} = 4 \times \frac{C^2}{r}.$$

· جمع FLOPs بلوک SE:

$$\text{FLOPs}_{\text{SE}} = H \times W \times C + 4 \times \frac{C^2}{r}.$$

برای $C = 64$ و $r = 16$:

$$\text{FLOPs}_{\text{SE}} = H \times W \times 64 + 1024.$$

❊ **محاسبه FLOPs کانولوشن عمقی تفکیک پذیر:**

· عمقی تفکیک پذیر (Depthwise Convolution):

$$\text{FLOPs}_{\text{Depthwise}} = H \times W \times C \times K \times K.$$

$$\text{FLOPs}_{\text{Depthwise}} = H \times W \times 576.$$

· نقطه‌ای (Pointwise Convolution):

$$\text{FLOPs}_{\text{Pointwise}} = H \times W \times C_{\text{in}} \times C_{\text{out}}.$$

$$\text{FLOPs}_{\text{Pointwise}} = H \times W \times 4096.$$

· جمع FLOPs کانولوشن عمقی تفکیک پذیر:

$$\text{FLOPs}_{\text{DepthwiseSeparable}} = H \times W \times 4672.$$

❊ **نسبت FLOPs:**

· برای نقشه‌های ویژگی کوچک ($H = W = 7$):

$$\text{Ratio} = \frac{4160}{228928} \approx 0.0182 \quad .$$

· برای نقشه‌های ویژگی بزرگ ($H, W \to \infty$):

$$\text{Ratio} = \frac{64}{4672} \approx 0.0137 \quad .$$

· **محدوده نسبت FLOPs:**

$$\boxed{1.37\% \text{ تا } 1.82\%}.$$

۴-۱ & ۴-۲.

- **جستجوی معماری عصبی (NAS) در MobileNetV3:**

جستجوی معماری عصبی (Neural Architecture Search یا NAS) در MobileNetV۳ ترکیبی از جستجوی خودکار و طراحی انسانی برای بهینه‌سازی معماری شبکه بوده است. فرآیند کلی شامل دو مرحله است:

- **مرحله اول: جستجوی معماری در سطح بلاک‌ها** – در این مرحله، یک NAS مبتنی بر یادگیری تقویتی (مشابه MnasNet) برای یافتن ساختار کلی شبکه اعمال شد. این جستجو شامل بهینه‌سازی یک تابع چندهدفه است که هم دقت شبکه را روی ImageNet و هم تأخیر آن بر روی سخت‌افزار موبایل در نظر می‌گیرد. هر مرحله در معماری شامل بلوک‌های مختلفی است که باید بهینه شوند، مانند انتخاب اندازه کرنل (3×3 یا 5×5)، وجود یا عدم وجود مکانیزم Squeeze-and-Excitation (SE) و مقدار فاکتور گسترش t است.

- **مرحله دوم: تنظیم دقیق تعداد کانال‌ها با NetAdapt** – پس از تعیین ساختار کلی، از الگوریتم NetAdapt برای تنظیم دقیق تعداد فیلترها در هر لایه استفاده شد. NetAdapt از یک روش **کوچک‌سازی پیش‌رونده** استفاده می‌کند که به جای کاهش ناگهانی اندازه مدل، به صورت تدریجی کانال‌ها را کم کرده و شبکه را برای حفظ عملکرد خود تنظیم مجدد (fine-tune) می‌کند.

- **مدل‌سازی فضای جستجو با استفاده از نظریه گراف:**

فضای جستجوی NAS را می‌توان به عنوان یک **گراف جهت‌دار بدون دور (DAG)** مدل‌سازی کرد، جایی که:

- **گره‌ها** (V) نشان‌دهنده‌ی تنسورهای ویژگی میانجی در معماری شبکه هستند (یعنی نگاشت‌های خروجی لایه‌های مختلف).

- **یال‌ها** (E) نشان‌دهنده‌ی عملیات بین این گره‌ها هستند (یعنی انواع مختلف بلاک‌های شبکه مانند بلاک‌های Inverted Residual، بلاک‌های SE، یا انتخاب کرنل‌های مختلف).

یک شبکه عصبی را می‌توان به صورت یک DAG در نظر گرفت، که در آن هر مسیر یک پیوند قانونی از ورودی به خروجی را تشکیل می‌دهد. در فضای جستجوی MobileNetV۳، هر لایه از شبکه دارای مجموعه‌ای از انتخاب‌های ممکن است، مانند:

- انتخاب نوع بلاک (مثلاً یک بلاک 3×3 با SE یا یک بلاک 5×5 بدون SE).
- مقدار فاکتور گسترش t در بلاک‌های Inverted Residual.
- حضور یا عدم حضور Squeeze-and-Excitation (SE).

هر مسیر در DAG یک معماری بالقوه را نشان می‌دهد، و NAS وظیفه دارد بهترین زیرگراف را پیدا کند که منجر به دقت بالا با هزینه محاسباتی پایین شود. استفاده از نظریه گراف باعث می‌شود که بتوان از الگوریتم‌های جستجو (مانند یادگیری تقویتی یا جستجوی تکاملی) برای کشف ساختار بهینه بهره برد.

- **تکنیک کوچک‌سازی پیش‌رونده در NAS:**

تکنیک کوچک‌سازی پیش‌رونده (Progressive Shrinking) در NAS به دو صورت به کار گرفته شده است:

۱. **NetAdapt برای کاهش تدریجی تعداد کانال‌ها:** این تکنیک به جای حذف یک‌باره‌ی تعداد زیادی کانال، کانال‌ها را به صورت تدریجی کاهش می‌دهد. در هر مرحله، شبکه به‌طور موقت کوچک می‌شود، دقت آن ارزیابی می‌شود، و در صورتی که دقت کاهش زیادی نداشته باشد، این تغییر حفظ شده و مرحله بعدی کوچک‌سازی انجام می‌شود. این فرآیند تا زمانی که محدودیت‌های سخت‌افزاری (مثلاً تأخیر پردازشی یا حافظه مصرفی) رعایت شود، ادامه دارد.

۲. **جستجوی یک‌شبکه‌ای (One-Shot NAS) و آموزش تدریجی:** در تکنیک‌های مدرن‌تر NAS (مانند Once-For-All یا OFA)، یک شبکه بزرگ آموزش داده می‌شود که شامل بسیاری از زیرمدل‌های کوچکتر است. سپس، به‌طور تدریجی، قسمت‌هایی از این شبکه (مثلاً عرض کانال‌ها، عمق شبکه یا اندازه کرنل‌ها) کوچک شده و مدل به گونه‌ای آموزش داده می‌شود که بتواند چندین معماری را در بر بگیرد. این تکنیک باعث می‌شود که به جای آموزش جداگانه‌ی چندین مدل، یک مدل واحد برای چندین پیکربندی مختلف قابل استفاده باشد.

به طور کلی، تکنیک کوچک‌سازی پیش‌رونده به NAS اجازه می‌دهد که از یک فضای جستجوی گسترده آغاز کند و به‌تدریج مدل را بهینه‌سازی کند تا به یک معماری کارآمد و متناسب با محدودیت‌های سخت‌افزاری دست یابد.

▷

بخش عملی (۱۰۰ نمره)

در این مجموعه سه تمرین ارائه شده است. از شما خواسته می‌شود هر تمرین را به‌صورت جداگانه در سامانهٔ Quera بارگذاری نمایید. هر سوال دارای توضیحات لازمه داخل نوتبوک خود میباشد.