



یادگیری عمیق

نیم سال دوم ۰۵-۰۴
مدرس: مهدیه سلیمانی

ددلاین تمرین : ۲۲ خرداد

تمرین سوم

- برای ارسال هر تمرین تا ساعت ۲۳:۵۹ روز ددلاین فرصت دارید. مهلت تاخیر (مجاز و غیر مجاز) برای این تمرین، ۵ روز است (یعنی حداکثر تاریخ ارسال تمرین ۲۷ خرداد است)
- 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.
- از تحویل تمارین نظری به شکل تایپ شده (فایل Word-Latex یا هر نوع تایپ شده دیگری) بهره‌یید و حتما پاسخ دست نویس تحویل دهید.
- 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 `HW3_[First-Name]_[Last-Name]_[Student-Id].pdf` بارگذاری شوند.
- در صورت وجود هرگونه ابهام یا مشکل، در کوثرای درس آن مشکل را بیان کنید و از پیغام دادن مستقیم به دستیاران آموزشی خودداری کنید.
- طراحان این تمرین : فاطمه عسکری - دانیال پرنیان- مبین باقریان- علیرضا اعلائی- ارشیا پوسف نیا- حسین شاه آبادی

بخش نظری ۱۱۰ نمره

پرسش ۱. پرسش اول (۱۰ نمره)

در Multi-Head Attention، ورودی از طریق h مجموعه projection مختلف به چند زیرفضای بازنمایی نگاشت می‌شود:

$$\text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V), \quad W_i^Q, W_i^K, W_i^V \in \mathbb{R}^{d_{\text{model}} \times d_k}, \quad d_k = \frac{d_{\text{model}}}{h}.$$

(۱) فرض کنید همه هدها دقیقاً از ماتریس‌های projection یکسان استفاده کنند؛ یعنی برای هر i, j :

$$W_i^Q = W_j^Q, \quad W_i^K = W_j^K, \quad W_i^V = W_j^V.$$

نشان دهید در این حالت، Multi-Head Attention در عمل یک اتنشن یکسان را چند بار محاسبه می‌کند و خروجی آن را می‌توان به صورت یک single-head attention همراه با یک projection خروجی مؤثر بازنویسی کرد.

توجه کنید که این حالت لزوماً فقط معادل ضرب در یک ضریب مقیاس ساده نیست، مگر آنکه ماتریس خروجی W^O ساختار خاصی داشته باشد. این نتیجه چه چیزی درباره اهمیت تنوع بین هدهای اتنشن یادگرفته‌شده نشان می‌دهد؟

(ب) مطالعات تجربی نشان داده‌اند که برخی هدهای اتنشن روی پدیده‌های زبانی خاص، مانند وابستگی‌های نحوی یا coreference، تخصص پیدا می‌کنند. با این حال، بسیاری از هدها نیز افزونه یا تکراری هستند و حذف آن‌ها آسیب جدی به عملکرد مدل نمی‌زند.

اگر فقط بتوانید $k < h$ هد را نگه دارید، یک روش اصولی برای حذف هدهای کم‌اهمیت پیشنهاد دهید. روش شما باید هر دو جنبه زیر را در نظر بگیرد:

(۱) اهمیت هر هد به صورت جداگانه؛

(۲) افزونگی یا شباهت بین هدهای مختلف.

(ج) اکنون می‌خواهیم یک روش معماری‌محور برای انتخاب خودکار هدهای مفید طراحی کنیم. برای ایده گرفتن می‌توانید به مقالات Voita et al. (2019) و Michel et al. (2019) مراجعه کنید.

فرض کنید خروجی هد i را با H_i نشان می‌دهیم و برای هر هد یک متغیر دروازه‌ای یادگرفته‌نی تعریف می‌کنیم:

$$z_i \in [0, 1].$$

خروجی لایه به صورت زیر بازنویسی می‌شود:

$$\text{MHA}_{\text{gated}}(Q, K, V) = \text{Concat}(z_1 H_1, z_2 H_2, \dots, z_h H_h) W^O.$$

هدف این است که پس از آموزش، فقط تعداد کمی از هدهای مهم باقی بمانند و هدهای کم‌اهمیت یا تکراری حذف شوند.

یک تابع loss برای آموزش این دروازه‌ها طراحی کنید که شامل سه بخش باشد: task loss برای حفظ عملکرد مدل، یک جمله تنکی برای تشویق مدل به استفاده از تعداد کمتر هدها، و یک جریمه افزونگی برای جلوگیری از نگه داشتن هدهایی که رفتار بسیار مشابه دارند.

برای مثال، اگر A_i ماتریس اتنشن هد i باشد، می‌توانید شباهت دو هد را به صورت زیر تعریف کنید:

$$R(i, j) = \cos(\text{vec}(A_i), \text{vec}(A_j)).$$

یک فرم ممکن برای تابع هزینه بنویسید و توضیح دهید هر جمله چه نقشی دارد. همچنین توضیح دهید پس از آموزش، چگونه می‌توان از مقدار دروازه‌ها برای انتخاب k هد نهایی استفاده کرد.

در پایان، تحلیل کنید اگر دو هد رفتار کاملاً یکسانی داشته باشند، چرا وجود جریمه افزونگی باعث می‌شود نگه داشتن هم‌زمان هر دو هد مطلوب نباشد.

پرسش ۲. پرسش دوم (۱۰ نمره)

Transformer اصلی از Post-LN استفاده می‌کند:

$$x_{l+1} = \text{LN}(x_l + \text{Sublayer}(x_l)).$$

در بسیاری از معماری‌های جدیدتر، از Pre-LN استفاده می‌شود:

$$x_{l+1} = x_l + \text{Sublayer}(\text{LN}(x_l)).$$

(آ) جریان گرادینان را در یک residual block تحلیل کنید.

برای حالت Post-LN، مشتق زیر را بر حسب گرادینان لایه بعدی به دست آورید:

$$\frac{\partial \mathcal{L}}{\partial x_l}.$$

نشان دهید ژاکوبین LayerNorm چگونه می‌تواند اندازه و جهت گرادینان را تغییر دهد.

سپس این حالت را با Pre-LN مقایسه کنید. در Pre-LN، مسیر residual مستقیماً از normalization عبور نمی‌کند.

توضیح دهید چرا Pre-LN معمولاً آموزش مدل‌های عمیق‌تر را پایدارتر می‌کند و در بسیاری از تنظیمات، نیاز به learning-rate warmup را کاهش می‌دهد یا حتی حذف می‌کند.

(ب) با وجود پایداری بهتر در آموزش، مدل‌های Pre-LN در مدل‌های کوچک‌تر گاهی عملکرد نهایی کمی ضعیف‌تر از Post-LN دارند.

برای این پدیده یک توضیح منطقی ارائه دهید. در پاسخ خود به موارد زیر توجه کنید:

(۱) عمق موثر در تبدیل ویژگی‌ها؛

(۲) توزیع activation‌هایی که وارد لایه نهایی می‌شوند؛

(۳) نقش normalization به عنوان نوعی regularization.

(ج) برخی معماری‌های جدید از روش‌های ترکیبی normalization استفاده می‌کنند یا حتی normalization را حذف می‌کنند و به جای آن از مقداردهی اولیه یا مقیاس‌بندی مناسب شاخه residual استفاده می‌کنند.

یک نمونه از چنین روشی را توضیح دهید؛ مثلاً Fixup initialization، T-Fixup، DeepNorm یا روش‌های مشابه.

توضیح دهید چرا از دید نظری، اگر مقداردهی اولیه، مقیاس‌بندی residual و دینامیک گرادینان درست طراحی شوند، normalization ممکن است برای آموزش پایدار ضروری نباشد.

پرسش ۳. پرسش سوم (۲۰ نمره)

در مدل‌های زبانی بزرگ، دو راه رایج برای بهبود عملکرد مدل بدون آموزش کامل همه پارامترها وجود دارد. راه اول استفاده از روش‌های کم‌پارامتر مانند LoRA در مرحله fine-tuning است. راه دوم، ثابت نگه داشتن وزن‌های مدل و صرف محاسبه بیشتر در زمان استنتاج، مانند self-consistency یا استفاده از verifier است. در این پرسش، این دو رویکرد را از نظر تعداد پارامتر، هزینه استنتاج و کارایی مقایسه می‌کنیم.

(آ) در LoRA، ماتریس وزن اصلی

$$W \in \mathbb{R}^{d \times k}$$

ثابت نگه داشته می‌شود و به جای به‌روزرسانی مستقیم آن، یک به‌روزرسانی کم‌رتبه به آن اضافه می‌شود:

$$W' = W + \Delta W, \quad \Delta W = BA$$

که در آن:

$$B \in \mathbb{R}^{d \times r}, \quad A \in \mathbb{R}^{r \times k}, \quad r \ll \min(d, k)$$

نشان دهید تعداد پارامترهای قابل آموزش در LoRA برابر است با:

$$r(d + k)$$

در حالی که fine-tuning کامل ماتریس W نیاز به آموزش dk پارامتر دارد. سپس برای یک ماتریس projection با مشخصات زیر محاسبه کنید:

$$d = k = 4096, \quad r = 8$$

موارد زیر را به دست آورید:

(۱) تعداد پارامترهای قابل آموزش در fine-tuning کامل؛

(۲) تعداد پارامترهای قابل آموزش در LoRA؛

(۳) درصد کاهش تعداد پارامترها.

(ب) توضیح دهید چرا یک به‌روزرسانی کم‌رتبه از فرم

$$\Delta W = BA$$

می‌تواند برای سازگار کردن یک مدل pretrained بزرگ با یک وظیفه جدید مؤثر باشد. در پاسخ خود به این ایده اشاره کنید که تغییرات لازم در fine-tuning ممکن است در یک زیرفضای کم‌بعد قرار داشته باشند.

سپس دو روش استفاده از LoRA در زمان inference را مقایسه کنید:

$$Wx + BAx$$

و:

$$W' = W + BA$$

این دو حالت را از نظر تاخیر در inference، مصرف حافظه، و انعطاف‌پذیری هنگام سرو کردن چند adapter مختلف توضیح دهید.

(ج) اکنون فرض کنید وزن‌های مدل ثابت‌اند و می‌خواهیم فقط با صرف محاسبه بیشتر در زمان inference احتمال رسیدن به پاسخ درست را افزایش دهیم.
برای یک مسئله استدلالی، احتمال درست بودن یک پاسخ منفرد برابر است با:

$$p = 0.35$$

تعداد N پاسخ مستقل تولید می‌کنیم. سپس یک verifier یکی از پاسخ‌ها را انتخاب می‌کند. اگر حداقل یک پاسخ درست بین N پاسخ وجود داشته باشد، verifier با احتمال:

$$r = 0.8$$

یک پاسخ درست را انتخاب می‌کند. اگر هیچ پاسخ درستی وجود نداشته باشد، پاسخ نهایی غلط است.
فرمول احتمال درست بودن پاسخ نهایی را بر حسب N ، p و r به دست آورید. سپس این احتمال را برای مقادیر زیر محاسبه کنید:

$$N = 1, 2, 4, 8$$

همچنین نشان دهید اگر:

$$P_N = r(1 - (1 - p)^N)$$

باشد، سود افزایشی تولید نمونه‌های بیشتر، یعنی:

$$P_N - P_{N-1}$$

با افزایش N کاهش می‌یابد.

(د) اکنون دو راهبرد زیر را برای استفاده از بودجه استنتاج مقایسه می‌کنیم. بودجه کل برابر با ۶۰۰۰ توکن پردازش شده یا تولید شده است.

راهبرد اول: مثال‌های درون‌متنی

در ورودی، تعداد k مثال حل‌شده قرار می‌دهیم. هر مثال ۲۰۰ توکن دارد. خود سؤال نیز ۲۰۰ توکن است و مدل در پایان یک پاسخ ۴۰۰ توکنی تولید می‌کند. دقت تخمینی این روش به صورت زیر است:

$$p_A(k) = 0.35 + 0.08k$$

این رابطه فقط برای:

$$0 \leq k \leq 5$$

معتبر است.

راهبرد دوم: خودسازگاری با verifier

هیچ مثال حل‌شده‌ای در ورودی قرار نمی‌دهیم. برای هر نمونه، ورودی شامل سؤال ۲۰۰ توکنی است و مدل یک پاسخ ۴۰۰ توکنی تولید می‌کند. بنابراین هزینه هر نمونه برابر با ۶۰۰ توکن است. احتمال درست بودن هر پاسخ منفرد برابر با:

$$p = 0.45$$

است و verifier مانند بخش قبل با احتمال:

$$r = 0.8$$

در صورت وجود حداقل یک پاسخ درست، پاسخ درست را انتخاب می‌کند.
موارد زیر را محاسبه کنید:

(۱) بیشترین مقدار مجاز k در راهبرد اول؛

(۲) بهترین دقت تخمینی راهبرد اول؛

(۳) بیشترین مقدار مجاز N در راهبرد دوم؛

(۴) بهترین دقت تخمینی راهبرد دوم؛

(۵) با توجه به فرض‌های مسئله، کدام راهبرد بهتر است؟

در پایان، در یک پاراگراف توضیح دهید در چه شرایطی استفاده از LoRA مناسب‌تر است و در چه شرایطی inference-time scaling مناسب‌تر است. در پاسخ خود به تفاوت هزینه آموزش و هزینه استنتاج، وظایف تکرارشونده، مسائل استدلالی موردی، و امکان ترکیب دو روش اشاره کنید.

پرسش ۴. پرسش چهارم (۲۰ نمره)

در این پرسش می‌خواهیم رابطه میان Self-Attention، Local Attention و Convolution را هم از نظر مفهومی و هم از نظر محاسباتی بررسی کنیم.
فرض کنید ورودی یک لایه ترنسفورمر به صورت زیر باشد:

$$Z \in \mathbb{R}^{b \times n \times d}$$

که در آن b اندازه batch، n طول دنباله و $d = d_{\text{model}}$ بعد مدل است. در Multi-Head Self-Attention داریم:

$$Q_i = ZW_i^Q, \quad K_i = ZW_i^K, \quad V_i = ZW_i^V$$

که در آن:

$$W_i^Q, W_i^K, W_i^V \in \mathbb{R}^{d \times d_k}, \quad d_k = \frac{d}{h}$$

و خروجی هد i به صورت زیر تعریف می‌شود:

$$H_i = \text{softmax} \left(\frac{Q_i K_i^\top}{\sqrt{d_k}} \right) V_i$$

خروجی نهایی لایه نیز برابر است با:

$$\text{MHA}(Z) = \text{Concat}(H_1, \dots, H_h) W^O$$

که در آن:

$$W^O \in \mathbb{R}^{d \times d}$$

در این سؤال هزینه softmax، biasها، LayerNorm و residual connection را نادیده بگیرید و هر multiply-add را یک MAC در نظر بگیرید. مشخصات مدل به صورت زیر است:

$$b = 4, \quad n = 4096, \quad d = 1024, \quad h = 16, \quad d_k = 64$$

همچنین یک نسخه local causal attention را در نظر بگیرید که در آن هر توکن فقط به $w = 128$ توکن قبلی، شامل خودش، توجه می‌کند.

(آ) هزینه محاسباتی تقریبی MAC برای یک لایه standard full Multi-Head Self-Attention را به دست آورید. در محاسبه خود، هزینه‌های زیر را لحاظ کنید:

(۱) محاسبه Q,K,V projections؛

(۲) محاسبه ماتریس‌های امتیاز توجه QK^T ؛

(۳) ضرب attention probabilities در V ؛

(۴) output projection.

سپس مقدار عددی کل MAC را برای مشخصات داده‌شده محاسبه کنید.

(ب) همین محاسبه را برای local causal attention با پنجره $w = 128$ انجام دهید و نشان دهید که جمله درجه دوم نسبت به n به جمله‌ای خطی نسبت به nw تبدیل می‌شود. مقدار عددی کل MAC را نیز محاسبه کنید.

(ج) یک causal convolution معمولی با اندازه کرنل $w = 128$ را در نظر بگیرید که ورودی و خروجی آن هر دو بعد $d = 1024$ دارند:

$$Y_t = \sum_{r=0}^{w-1} Z_{t-r} C_r$$

که در آن:

$$C_r \in \mathbb{R}^{d \times d}$$

هزینه محاسباتی تقریبی این کانولوشن را محاسبه کنید و آن را با full attention و local attention مقایسه کنید. نتیجه چه چیزی درباره تفاوت channel mixing در کانولوشن و اتنشن نشان می‌دهد؟

(د) فرض کنید در یک هد از local self-attention، نمره توجه بین توکن t و توکن $t-r$ مستقل از محتوای ورودی باشد و فقط به فاصله نسبی r بستگی داشته باشد. یعنی:

$$\frac{q_t k_{t-r}^T}{\sqrt{d_k}} = \beta_r$$

برای $r = 0, \dots, w-1$.

نشان دهید در این حالت خروجی این هد معادل یک causal convolution با کرنل‌هایی وابسته به β_r و W^V است. سپس این نتیجه را برای چند هد و ماتریس W^O تعمیم دهید.

(ه) توضیح دهید چرا self-attention در حالت کلی فقط یک کانولوشن معمولی نیست. برای پاسخ، از این نکته استفاده کنید که وزن‌های توجه در حالت کلی تابعی از خود ورودی هستند:

$$\alpha_{t,j}(Z) = \text{softmax}_j \left(\frac{z_t^T W^Q (W^K)^T z_j}{\sqrt{d_k}} \right)$$

نشان دهید که این موضوع باعث می‌شود self-attention شبیه یک dynamic convolution یا input-dependent convolution رفتار کند، نه یک کانولوشن ثابت.

پرسش ۵. پرسش پنجم (۱۰ نمره)

فرض کنید یک مدل زبانی از پیش‌آموزش‌دیده‌ی مبتنی بر Transformer داریم که مشخصات زیر را دارد:

- ۲۴ لایه‌ی Transformer

- اندازه‌ی حالت پنهان (hidden state) برابر با $d = 1024$

- ۱۶ سر attention

- در مجموع ۳۵۰ میلیون پارامتر

می‌خواهیم این مدل را برای ۲۰ وظیفه‌ی پایین‌دستی (downstream task) مختلف سازگار کنیم. سه روش full fine-tuning، Prompt Tuning و Prefix Tuning را در نظر بگیرید و به پرسش‌های زیر پاسخ دهید.

(آ) در روش full fine-tuning، برای هر وظیفه همه‌ی ۳۵۰ میلیون پارامتر مدل ریزتنظیم می‌شوند و یک نسخه‌ی جداگانه از کل مدل ذخیره می‌شود.

برای ۲۰ وظیفه، محاسبه کنید در مجموع چند پارامتر اختصاصی وظیفه باید ذخیره شود.

(ب) در روش Prompt Tuning، مدل از پیش‌آموزش‌دیده ثابت نگه داشته می‌شود. برای هر وظیفه، یک soft prompt با طول $m = 50$ یاد می‌گیریم. هر بردار soft prompt دارای بعد $d = 1024$ است. در این روش، تنها پارامترهای اختصاصی هر وظیفه همان بردارهای soft prompt هستند و یک نسخه‌ی مشترک از مدل پایه ذخیره می‌شود.

موارد زیر را محاسبه کنید:

(۱) تعداد پارامترهای آموزش‌پذیر برای هر وظیفه؛

(۲) تعداد کل پارامترهای اختصاصی وظیفه برای ۲۰ وظیفه.

(ج) در روش Prefix Tuning، مدل از پیش‌آموزش‌دیده ثابت نگه داشته می‌شود. برای هر وظیفه، در هر لایه‌ی Transformer تعداد $p = 10$ بردار prefix برای بخش Key و تعداد $p = 10$ بردار prefix برای بخش Value یاد می‌گیریم. فرض کنید هر بردار prefix بعد $d = 1024$ دارد و از هزینه‌ی پارامترهای اضافی کوچک مانند MLP reparameterization صرف‌نظر می‌کنیم.

موارد زیر را محاسبه کنید:

(۱) تعداد پارامترهای آموزش‌پذیر برای هر وظیفه؛

(۲) تعداد کل پارامترهای اختصاصی وظیفه برای ۲۰ وظیفه.

(د) نسبت کاهش ذخیره‌سازی پارامترهای اختصاصی وظیفه را در دو روش زیر نسبت به full fine-tuning محاسبه کنید:

(۱) Prompt Tuning

(۲) Prefix Tuning

(ه) تفاوت اصلی Prompt Tuning و Prefix Tuning را توضیح دهید. برای هر کدام، یک مزیت و یک محدودیت بیان کنید.

پرسش ۶. پرسش ششم (۲۰ نمره)

در این پرسش، معماری‌های بینایی مبتنی بر ترنسفورمر را از دیدگاه طراحی معماری، نحوه ورود اطلاعات مکانی، مدل‌سازی وابستگی‌های محلی و سراسری، و طراحی تابع هزینه در مسئله تشخیص اشیا بررسی می‌کنیم.

(آ) فرض کنید می‌خواهید یک مدل مبتنی بر Vision Transformer (ViT) برای طبقه‌بندی تصویر طراحی کنید. داده آموزشی محدود است و می‌دانیم الگوهای محلی تصویر، مانند لبه‌ها، بافت‌ها و بخش‌های کوچک اشیا، برای مسئله مهم هستند.

یک نسخه اصلاح‌شده از ViT طراحی کنید که بخشی از سوگیری محلی CNN را وارد مدل کند، اما همچنان هسته اصلی مدل مبتنی بر ترنسفورمر باقی بماند. در پاسخ خود، فقط یکی از ایده‌های زیر را انتخاب کنید و همان را دقیق توضیح دهید:

- (۱) استفاده از convolutional stem پیش از ورود به ترنسفورمر؛
- (۲) کاهش اندازه patch و حفظ جزئیات مکانی بیشتر؛
- (۳) طراحی ساختار سلسله‌مراتبی شبیه مدل‌های pyramid-based؛
- (۴) ترکیب local attention در لایه‌های ابتدایی و global attention در لایه‌های بالاتر.

در طراحی پیشنهادی خود، موارد زیر را مشخص کنید:

- (۱) ورودی تصویر چگونه به توکن تبدیل می‌شود؟
- (۲) اطلاعات مکانی چگونه حفظ یا تقویت می‌شود؟
- (۳) این تغییر چه نوع inductive bias جدیدی نسبت به ViT ساده اضافه می‌کند؟
- (۴) این طراحی چه اثری بر نیاز مدل به داده یا پیش‌آموزش دارد؟
- (۵) این طراحی چه محدودیتی یا هزینه‌ای نسبت به ViT ساده ایجاد می‌کند؟

(ب) در Swin Transformer، به جای اعمال self-attention روی همه توکن‌های تصویر، از window attention و سپس shifted window attention استفاده می‌شود.

برای ساده‌سازی، یک دنباله یک‌بعدی از توکن‌ها را در نظر بگیرید:

$$x_1, x_2, \dots, x_8$$

و فرض کنید اندازه هر پنجره برابر با ۴ باشد. در لایه اول، پنجره‌ها به صورت زیر هستند:

$$[x_1, x_2, x_3, x_4], \quad [x_5, x_6, x_7, x_8]$$

ابتدا توضیح دهید چرا در این لایه، توکن‌های x_4 و x_5 نمی‌توانند مستقیماً با هم attention برقرار کنند، با وجود آنکه در تصویر یا دنباله مجاور یکدیگر هستند.

سپس فرض کنید در لایه بعدی پنجره‌ها به اندازه ۲ توکن جابه‌جا شوند، به طوری که یکی از پنجره‌های جدید شامل توکن‌های زیر باشد:

$$[x_3, x_4, x_5, x_6]$$

نشان دهید در این حالت، توکن‌های x_4 و x_5 می‌توانند در یک پنجره مشترک قرار بگیرند و اطلاعات ردوبدل کنند.

در ادامه، نتیجه بگیرید که تکرار window attention و shifted window attention در چند لایه چگونه باعث گسترش تدریجی receptive field می‌شود. همچنین توضیح دهید این طراحی چه تفاوتی با full self-attention و چه شباهتی با سوگیری محلی در CNN دارد.

(ج) فرض کنید می‌خواهید از یک مدل ترنسفورمری برای object detection استفاده کنید. یک معماری کلی از نوع DETR طراحی کنید که شامل اجزای زیر باشد:

- (۱) یک backbone برای استخراج ویژگی‌های تصویری؛
- (۲) یک Transformer Encoder برای مدل‌سازی وابستگی‌های سراسری بین موقعیت‌های تصویر؛
- (۳) یک Transformer Decoder با تعداد ثابتی object query؛
- (۴) دو سر خروجی برای پیش‌بینی کلاس و جعبه مکانی هر شیء.

توضیح دهید object query‌ها در این معماری چه نقشی دارند. چرا می‌توان هر object query را به عنوان یک جایگاه قابل یادگیری برای جست‌وجوی یک شیء در تصویر در نظر گرفت؟ همچنین توضیح دهید چرا خروجی مدل به صورت یک مجموعه از پیش‌بینی‌ها تفسیر می‌شود، نه یک دنباله وابسته به ترتیب.

(د) برای پاسخ به این بخش، ابتدا ایده set prediction و Hungarian Matching در مقاله DETR: End-to-End Object Detection with Transformers را مطالعه کنید.

در DETR، برای آموزش مدل لازم است بین پیش‌بینی‌های مدل و اشیای واقعی تصویر یک تطابق یک‌به‌یک پیدا شود. فرض کنید مدل N پیش‌بینی تولید می‌کند:

$$\hat{y}_i = (\hat{p}_i, \hat{b}_i), \quad i = 1, \dots, N$$

که در آن $\hat{p}_i$ توزیع احتمالی روی کلاس‌ها و $\hat{b}_i$ جعبه پیش‌بینی شده است. همچنین فرض کنید تصویر شامل m شیء واقعی باشد:

$$y_j = (c_j, b_j), \quad j = 1, \dots, m$$

ابتدا الگوریتم تطبیق مجارستانی، یعنی Hungarian Matching، را توضیح دهید و بیان کنید چرا برای فرمول‌بندی set prediction در DETR به یک تطابق یک‌به‌یک بین پیش‌بینی‌ها و اشیای واقعی نیاز داریم. سپس یک تابع هزینه برای تطبیق پیش‌بینی i با شیء واقعی j طراحی کنید که شامل سه جزء زیر باشد:

$$\text{classification cost}, \quad \text{L1 box cost}, \quad \text{GIoU cost}$$

برای مثال می‌توانید هزینه را به صورت زیر در نظر بگیرید:

$$C_{ij} = -\log \hat{p}_i(c_j) + \lambda_1 \|\hat{b}_i - b_j\|_1 + \lambda_{\text{giou}} (1 - \text{GIoU}(\hat{b}_i, b_j))$$

توضیح دهید هر یک از سه جمله بالا چه نقشی در انتخاب تطابق مناسب دارد. همچنین توضیح دهید پیش‌بینی‌هایی که با هیچ شیء واقعی تطبیق داده نمی‌شوند، در مرحله آموزش چگونه در تابع هزینه وارد می‌شوند.

(ه) پس از پیدا شدن تطابق بهینه، تابع loss نهایی DETR چگونه ساخته می‌شود؟ پاسخ خود را به صورت یک طراحی مرحله‌به‌مرحله بنویسید.
در پاسخ خود مشخص کنید:

- (۱) برای همه object query‌ها چه نوع classification loss محاسبه می‌شود؛
- (۲) چرا box regression loss فقط برای هایی query محاسبه می‌شود که با یک شیء واقعی تطبیق یافته‌اند؛
- (۳) کلاس no-object چه نقشی در آموزش های query بدون تطابق دارد؛
- (۴) ضرایب λ_1 و λ_{giou} چگونه تعادل بین دقت طبقه‌بندی و دقت مکانی جعبه‌ها را کنترل می‌کنند.

در ادامه، یک ضعف معماری DETR استاندارد را بیان کنید؛ برای مثال همگرایی کند یا دشواری تشخیص اشیای کوچک. سپس یک تغییر معماری برای بهبود آن پیشنهاد دهید. در پاسخ خود می‌توانید از ایده‌هایی مانند ویژگی‌های چندمقیاسی، feature pyramid، یا deformable attention استفاده کنید و توضیح دهید چرا این تغییر می‌تواند برای تشخیص اشیای کوچک یا افزایش سرعت آموزش مفید باشد.

پرسش ۷. پرسش هفتم (۲۰ نمره)

در ترنسفورمرها، سازوکار self-attention به‌تنهایی ترتیب توکن‌ها را به‌صورت صریح مدل نمی‌کند. بنابراین لازم است اطلاعات مکانی به شکلی وارد مدل شود. در این پرسش، ابتدا خاصیت جابه‌جایی در sinusoidal positional encoding را بررسی می‌کنیم و سپس نشان می‌دهیم Rotary Positional Embedding یا RoPE چگونه اطلاعات فاصله نسبی را مستقیماً وارد امتیاز توجه می‌کند.

(آ) در sinusoidal positional encoding ترنسفورمر اصلی، برای هر موقعیت t و هر جفت بعد داریم:

$$p_i(t) = \begin{bmatrix} \sin(\omega_i t) \\ \cos(\omega_i t) \end{bmatrix}$$

که در آن:

$$\omega_i = \frac{1}{10000^{2i/d}}$$

است. نشان دهید برای هر جابه‌جایی ثابت k ، یک ماتریس خطی $A_i(k) \in \mathbb{R}^{2 \times 2}$ وجود دارد به طوری که:

$$p_i(t+k) = A_i(k)p_i(t)$$

ماتریس $A_i(k)$ را به دست آورید. سپس نشان دهید:

$$A_i(k_1 + k_2) = A_i(k_1)A_i(k_2)$$

و توضیح دهید این رابطه چه چیزی درباره ساختار منظم جابه‌جایی‌ها در sinusoidal positional encoding نشان می‌دهد.

(ب) برای دو موقعیت m و n نشان دهید:

$$p_i(m)^\top p_i(n) = \cos(\omega_i(m-n))$$

اکنون فرض کنید کل بردار مکانی $PE(t)$ از الحاق همه جفت‌های دوبعدی $p_i(t)$ ساخته شده است. نشان دهید:

$$PE(m)^\top PE(n) = \sum_i \cos(\omega_i(m-n))$$

نتیجه بگیرید که شباهت ساده بین دو بردار مکانی در sinusoidal positional encoding تابعی از فاصله نسبی $m-n$ است، نه صرفاً تابعی جداگانه از دو موقعیت مطلق m و n .

(ج) اکنون یک امتیاز مکانی کلی‌تر را در نظر بگیرید:

$$g_B(m, n) = PE(m)^\top B PE(n)$$

که در آن:

$$B \in \mathbb{R}^{d \times d}$$

می‌تواند بخشی از ماتریس مؤثر $W^Q(W^K)^\top$ در امتیاز توجه باشد. فرض کنید B نسبت به جفت ابعادهای سینوسی-کسینوسی، بلوکی باشد و هر بلوک آن به صورت زیر تعریف شود:

$$B_i = a_i I + b_i J$$

که:

$$I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad J = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$$

است. نشان دهید:

$$p_i(m)^\top B_i p_i(n) = a_i \cos(\omega_i(m - n)) + b_i \sin(\omega_i(n - m))$$

و نتیجه بگیرید که در این حالت، $g_B(m, n)$ فقط به فاصله نسبی دو موقعیت وابسته است. آیا برای یک ماتریس دلخواه B نیز لزوماً $g_B(m, n)$ فقط تابعی از $m - n$ است؟ توضیح دهید.

(د) اکنون RoPE را در نظر بگیرید. در این روش، به جای اضافه کردن بردار مکانی به embedding توکن، بردارهای query و key بر اساس موقعیت خود چرخانده می‌شوند. برای یک جفت بعدی، ماتریس چرخش را به صورت زیر تعریف می‌کنیم:

$$R(\phi) = \begin{bmatrix} \cos \phi & -\sin \phi \\ \sin \phi & \cos \phi \end{bmatrix}$$

اگر $q, k \in \mathbb{R}^2$ باشند و در موقعیت‌های m و n داشته باشیم:

$$\tilde{q}_m = R(m\theta)q, \quad \tilde{k}_n = R(n\theta)k$$

نشان دهید:

$$\tilde{q}_m^\top \tilde{k}_n = q^\top R((n - m)\theta)k$$

این رابطه چه چیزی درباره وابستگی attention score به فاصله نسبی دو توکن نشان می‌دهد؟

۵) در حالت کلی، در RoPE بردارهای q و k به چند زیربردار دوبعدی تقسیم می‌شوند و هر زیربردار با فرکانس متفاوتی چرخانده می‌شود. فرض کنید:

$$q = [q^{(0)}, q^{(1)}, \dots, q^{(d/2-1)}]$$

$$k = [k^{(0)}, k^{(1)}, \dots, k^{(d/2-1)}]$$

که در آن:

$$q^{(i)}, k^{(i)} \in \mathbb{R}^2$$

و برای هر جفت دوبعدی فرکانس θ_i استفاده می‌شود. نشان دهید:

$$\tilde{q}_m^\top \tilde{k}_n = \sum_{i=0}^{d/2-1} (q^{(i)})^\top R((n-m)\theta_i) k^{(i)}$$

سپس تفاوت اصلی absolute positional embedding، sinusoidal positional encoding و RoPE را توضیح دهید. مشخص کنید اطلاعات مکانی در هر روش به کدام بخش وارد می‌شود:

input embedding, query/key representation, attention score

در پایان توضیح دهید چرا RoPE نسبت به absolute positional embedding ساختار مناسب‌تری برای مدل‌سازی روابط نسبی و تعمیم به طول کانتکست‌های بزرگ‌تر فراهم می‌کند.

بخش عملی (۱۰۰ نمره)

در این بخش سه تمرین عملی ارائه شده است. تمرین‌های اول و دوم هر کدام ۳۵ نمره و تمرین سوم ۳۰ نمره دارند. از شما خواسته می‌شود هر تمرین را به‌صورت جداگانه در سامانه‌ی Quera بارگذاری نمایید. نوت‌بوک‌ها باید از قبل به‌صورت کامل اجرا شده باشند و خروجی‌های لازم نیز در فایل نهایی باقی بمانند؛ در غیر این صورت نمره‌ی مربوطه تعلق نخواهد گرفت.

پرسش عملی ۱: سازگار کردن مدل‌های زبانی با PEFT و QLoRA (۳۵ نمره)

در این تمرین، یک مدل زبانی از پیش‌آموزش‌دیده را با استفاده از روش‌های کم‌پارامتر مانند LoRA و QLoRA برای چند وظیفه‌ی زبانی سازگار می‌کنید. ابتدا داده‌های محلی مربوط به وظایف مختلف را بارگذاری و به قالب مناسب prompt/target تبدیل کنید. سپس آداپتورهای تخصصی را آموزش داده، اثر رتبه‌ی LoRA را بررسی کرده و عملکرد مدل پایه و مدل‌های سازگار شده را با هم مقایسه و تحلیل کنید.

پرسش عملی ۲: روش‌های پیشرفته‌ی استنتاج و استدلال در مدل‌های زبانی کوچک (۳۵ نمره)

در این تمرین، یک مدل زبانی کوچک را روی زیرمجموعه‌ای از GSM8K اجرا می‌کنید و چند راهبرد مختلف استنتاج را با هم مقایسه می‌کنید. روش‌هایی مانند پاسخ مستقیم، chain-of-thought، نمونه‌گیری چندباره، self-consistency

و استفاده از سازوکار انتخاب پاسخ را پیاده‌سازی و ارزیابی کنید. در پایان، دقت، هزینه‌ی توکنی و نقاط قوت و ضعف هر روش را گزارش و تحلیل نمایید.

پرسش عملی ۳: پیاده‌سازی سامانه‌ی Retrieval-Augmented Generation (RAG) (۳۰ نمره)

در این تمرین، یک سامانه‌ی RAG را از مرحله‌ی آماده‌سازی اسناد تا بازیابی و تولید پاسخ پیاده‌سازی می‌کنید. ابتدا اسناد را بارگذاری و به قطعه‌های کوچک‌تر تقسیم کنید، سپس با استفاده از SentenceTransformer و FAISS یک بازیاب بسازید. در ادامه، پاسخ‌گویی با مدل‌های مولد مختلف را بررسی کرده و نسخه‌ی ساده‌ی RAG را با نسخه‌ی مبتنی بر مکالمه یا حافظه مقایسه کنید.