به نام خدا

دانشگاه تهران

دانشکده مهندسی برق و کامپیوتر

یادگیری عمیق با کاربردها

تمرین شماره چهار

نام و نام خانوادگی	امید نائیج نژاد
شماره دانشجویی	610301189
تاریخ ارسال گزارش	1404/3/20

فهرست مطالب

فهرست شکل ها

فهرست جدول ها

سوال 1

ابعاد سیگنال قبل از ورود به هر بلاک و پس از خروج از آن در بخش رمزنگار به صورت زیر می باشد:

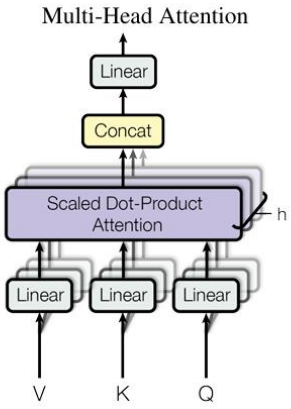
در ابتدا سیگنال داده ای به شکل یک ماتریس با ابعاد (32, 4096) → (Batch, Seq Length) است. در این مرحله هر توکن به صورت یک رشته فعلا ذخیره شده است.

پیش از ورود داده به ساختار مدل های ترنسفورمری، باید با کمک روش های امبدینگ ورودی (تعبیه)، نمایش وان-هات توکن ها را با یک تبدیل خطی به یک فضای معنادار از لحاظ فاصله بردارها منتقل کنیم. (مشابه روشی که در مدل های رمزنگار-رمزگشا مبتنی بر شبکه های عصبی بازگشتی برای کاربرد توصیف تصاویر در تمرین قبلی انجام دادیم). بنابراین ابعاد سیگنال به صورت (32, 4096, 1024) تبدیل خواهد شد. برای نمایش هر توکن از یک بردار به اندازه امبدینگ توکن ورودی یعنی 1024 استفاده کرده ایم.

حالا به بلاک های انکودری (رمزنگار) میرسیم.

- در بلوک هد چندگانه توجه، بردار نهان به تعداد هدها تقسیم میشود تا ورودی و خروجی هر هد به دست بیاید:

✓ هر هد توجه، مطابق با شکل 1، در ابتدا با استفاده از یک تبدیل خطی برای محاسبه Q, V, K ابعاد سیگنال را از (32, 4096, 1024) به (32, 4096, 768) تبدیل میکند و متناسب با تعداد هد ها انها را جدا میکند: (32, 6, 4096, 128) → 768 ÷ 6 = 128



شکل 1 - زیربخش های هد چندگانه توجه

✓ حالا هر هد توجه به صورت جداگانه مطابق با شکل 2، و رابطه 1 خروجی محاسبه میشود:

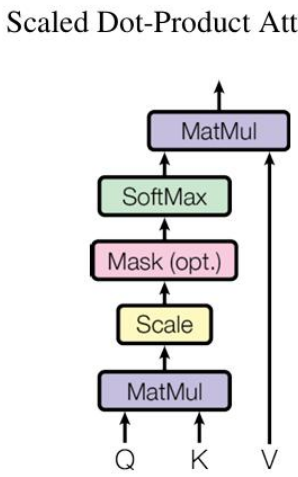
$$softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

1. $QK^T = (32, 6, 4096, 4096)$
2. تقسیم کردن ماتریس بر یک اسکالر و تابع سافت مکس واضحا ابعاد را تغییر نمیدهند.
3. بعد از ضرب کردن ماتریس بدست امده در بردار V، مجددا سیگنال به صورت (32, 6, 4096, 128) خواهد بود.

✓ حالا پس از پردازش های هر هد توجه باید خروجی های بدست امده را به همدیگر بچسبانیم(شکل 1) پس سیگنال دوباره بصورت (32, 4096, 768) درخواهد امد.

- ✓ در انتهای بلاک هد چندگانه توجه، تبدیل خطی خروجی انجام میشود تا سیگنال به حالت پیش از ورود به این بلاک یعنی (32, 4096, 1024) تبدیل شود.

- بدیهتا در بخش نرمال سازی لایه و ارتباط باقیمانده، ابعاد تغییر نمیکند.
- در لایه پسرو بلاک رمزگشا:
  - ✓ در لایه پسرو اول، ابعاد ورودی نصف میشود: (32, 4096, 512)
  - ✓ در لایه پسرو دوم، ابعاد ورودی تغییر نمیکند: (32, 4096, 512)
  - ✓ در لایه پسرو سوم، ابعاد ورودی دوبرابر میشود: (32, 4096, 1024)
- مجددا در بخش نرمال سازی لایه و ارتباط باقیمانده، ابعاد تغییر نمیکند.



شکل 2 - زیربخش های هر هد توجه

پارامترهای مدل:

- **هر بلوک رمزنگار:** 3,149,056 + 2048 + 1,312,768 + 2048 = 4,465,920
  - هد چندگانه توجه: 2,361,600 + 787,456 = 3,149,056
    - تبدیل خطی Q/K/V: هر کدام یک لایه پسرو از 1024 به 128 هستند پس تعداد پارامترهای این بخش با توجه به اینکه 6 هد توجه داریم برابر است با: 6 × 3 × (1024 × 128 + 128) = 2,361,600
    - تبدیل خطی خروجی: یک لایه پسرو از 768 به 1024 است پس تعداد پارامترهای این بخش برابر است با: 768 × 1024 + 1024 = 787,456
  - نرمالسازی لایه: 2 بردار قابل اموزش به اندازه نرون های آن لایه یعنی 2 × 1024 = 2048 .
  - لایه های پسرو: 524,800 + 262,656 + 525,312 = 1,312,768
    - در لایه پسرو اول، ابعاد ورودی نصف میشود: 1024 × 512 + 512 = 524,800
    - در لایه پسرو دوم، ابعاد ورودی تغییر نمیکند: 512 × 512 + 512 = 262,656
    - در لایه پسرو سوم، ابعاد ورودی دوبرابر میشود: 512 × 1024 + 1024 = 525,312
  - نرمالسازی لایه: 2 بردار قابل اموزش به اندازه نرون های آن لایه یعنی 2 × 1024 = 2048 .

- **هر بلوک رمزگشا:** 3,149,056 + 2048 + 3,149,056 + 2048 + 1,312,768 + 2048 = 7,617,192
  - **هد چندگانه ماسک شده توجه:** 2,361,600 + 787,456 = 3,149,056
    - **تبدیل خطی Q/K/V:** هر کدام یک لایه پسرو از 1024 به 128 هستند پس تعداد پارامترهای این بخش با توجه به اینکه 6 هد توجه داریم برابر است با: 6 × 3 × (1024 × 128 + 128) = 2,361,600
    - **تبدیل خطی خروجی:** یک لایه پسرو از 768 به 1024 است پس تعداد پارامترهای این بخش برابر است با: 768 × 1024 + 1024 = 787,456
  - **نرمالسازی لایه:** 2 بردار قابل اموزش به اندازه نرون های آن لایه یعنی 2048 = 1024 × 2 .
  - **توجه رمزنگار-رمزگشا:** تعداد پارامترها کاملا مشابه با هد چندگانه ماسک شده توجه است: 2,361,600 + 787,456 = 3,149,056
  - **نرمالسازی لایه:** 2 بردار قابل اموزش به اندازه نرون های آن لایه یعنی 2048 = 1024 × 2 .
  - **لایه های پسرو:** 524,800 + 262,656 + 525,312 = 1,312,768
    - در لایه پسرو اول، ابعاد ورودی نصف میشود: 1024 × 512 + 512 = 524,800
    - در لایه پسرو دوم، ابعاد ورودی تغییر نمیکند: 512 × 512 + 512 = 262,656
    - در لایه پسرو سوم، ابعاد ورودی دوبرابر میشود: 512 × 1024 + 1024 = 525,312
  - **نرمالسازی لایه:** 2 بردار قابل اموزش به اندازه نرون های آن لایه یعنی 2048 = 1024 × 2 .

مدل ترنسفورمری ما شامل 6 بلاک انکودر و 8 بلاک دیکدر است پس تعداد کل پارامترهای مدل برابر است با:

4,465,920 × 6 + 7,617,192 × 8 = 26,795,520 + 60,936,192 = **87,731,712**

سوال 2

مدل‌های زبانی بزرگ مانندGPT ، LLaMA و Claude معمولاً در سه مرحله آموزش داده می‌شوند:

1. پیش‌آموزش (Pre-training)
2. تنظیم دقیق با نظارت(Supervised Fine-Tuning - SFT)
3. هم‌راستاسازی با بازخورد انسانی (Alignment via Reinforcement Learning from Human Feedback - RLHF)

هر مرحله نقش خاصی در ساخت مدل‌های قابل استفاده و منطبق با ارزش‌های انسانی دارد.

۱. مرحله پیش آموزش

الف) این مرحله در چه داده هایی و با چه هدفی انجام می شود؟

در این مرحله، مدل با استفاده از داده های متنی خام وبدون برچسب مانند کتابها، مقالات ومحتوای وب، آموزش می بیند. هدف از این بخش، یادگیری الگوهای زبانی عمومی مانند دستور زبان، معناشناسی و دانش عمومی است.

- هدف: یادگیری ساختار زبان و دانش عمومی
- نوع داده: متون خام و بدون برچسب از منابع مختلف (مثل دیالوگ های سریال Friends در این تمرین)
- روش آموزش: با پیش بینی توکن بعدی دریک دنباله، مدل را آموزش می دهیم.

ب) چرا مدل در این مرحله هنوز آماده‌ی استفاده مستقیم در کاربردهای انسانی نیست؟

مدل های پیش آموزش شده توانایی درک یا پیروی از دستورات خاص را ندارند.

ممکن است پاسخ های نامناسب از نظر اخلاقی، مغرضانه یا نادرست تولید کنند. چون مدل ترجیحات انسانها را برای سوال(پرامپت)هایی که میپرسند نمیداند و ارزش منفی کلمات بی ادبانه را نمیفهمد؛ پس امکان تولید خروجی توهین آمیز و نامناسب وجود دارد. بنابراین، نیاز به مراحل بعدی برای تنظیم دقیق و هم راستاسازی دارد.

۲. مرحله آموزش با نظارت

الف) تفاوت این مرحله با پیش آموزش چیست؟

همانطور که گفته شد در مرحله پیش‌آموزش (Pre-training) :

- مدل زبانی برای اولین بار آموزش می‌بیند و با ساختار جملات آشنا میشود.
- در این مرحله، از داده‌های بدون برچسب استفاده می‌شود (مثل تمام محتوای اینترنت یا کتاب‌ها).
- هدف یادگیری ساختار زبان، روابط بین کلمات و جملات و بدست آوردن یک دانش عمومی و کلی است.
- نوع یادگیری اغلب به صورت self-supervised است، مثلاً مدل سعی می‌کند کلمات حذف‌شده را حدس بزند یا جمله بعدی را پیش‌بینی کند.

اما در مرحله آموزش با نظارت (Supervised Fine-Tuning) که بعد از پیش‌آموزش انجام می‌شود:

- از داده‌های برچسب‌خورده استفاده می‌شود؛ یعنی برای هر ورودی، پاسخ درست توسط انسان مشخص شده.
- هدف از این مرحله یادگیری نحوه انجام تسک های متفاوت (مثل پاسخ به سوال، ترجمه، طبقه‌بندی، چت‌کردن، خلاصه‌سازی) است.
- این مرحله باعث می‌شود مدل قابلیت تعامل واقعی با انسان‌ها را به‌دست بیاورد.

ب) نوع داده هایی که در این مرحله استفاده میشوند را شرح دهید و یک مثال بزنید؟

- داده‌ها باید شامل جفت‌های (ورودی، خروجی) باشند که توسط انسان تهیه شده‌اند.
- این داده‌ها اغلب در قالب مجموعه‌هایی از سوال و جواب، دستور و پاسخ، جمله و ترجمه، یا متن و برچسب ارائه می‌شوند. پس در این مرحله به این labelled data یا داده‌های برچسب‌خورده نیاز داریم. واضح است که جمع آوری داده مورد نیاز این مرحله بسیار هزینه برتر از مرحله اول است. در مرحله قبلی با نوشتن یک web crawler به راحتی کل متن های موجود در وب سایت ها را میتوانیم دانلود کنیم اما در این مرحله برای آماده سازی دیتا به انسان نیاز داریم.
- مثال‌هایی از داده‌های مورد نیاز برای تسک های مختلفی که از یک مدل زبانی انتظار میرود بتواند انجام دهد:

1. ترجمه:
   - ورودی: «Translate ‘I love you’ to Spanish»
   - خروجی: «Te quiero»
2. پاسخ به سوال:
   - ورودی :

«What is the boiling point of water?»

   - خروجی:

«100 °C at standard atmospheric pressure»

3. طبقه‌بندی احساسات:
   - ورودی: «.I had a terrible day»
   - خروجی: «Negative»
4. خلاصه‌سازی متن:
   - ورودی: یک مقاله یا متن بلند
   - خروجی: یک جمله‌ی خلاصه‌شده شامل مطالب مهم آن

ج) چرا این مرحله نقش مهمی در تعامل انسانی-مدل دارد؟

✓ انسان‌محور شدن مدل:

- مدل بدون آموزش با نظارت، فقط یک زبان‌فهم است، اما نمی‌داند چطور در تعامل واقعی با انسان‌ها پاسخ دقیق و سودمند بدهد.
- آموزش با نظارت به مدل کمک می‌کند تا سبک پاسخ‌دهی مناسب، دقیق و اخلاقی را بیاموزد.

- ✓ کاهش پاسخ‌های نادرست یا بی‌ربط:

  - مدل‌هایی که فقط با پیش‌آموزش تربیت شده‌اند، ممکن است پاسخ‌هایی بدهند که گمراه‌کننده، بی‌معنا یا حتی خطرناک باشد.
  - داده‌های نظارت‌شده مثل راهنماهایی هستند که به مدل نشان می‌دهند: «چه چیزی قابل قبول است و چه چیزی نیست.»

- ✓ افزایش دقت و قابلیت اطمینان:

  - در کاربردهای حساس مثل پزشکی، حقوق، یا آموزش، مدل باید دقیق باشد و علاوه بر یادگیری ساختار زبان، باید با اصطلاحات تخصصی و بار حقوقی بعضی کلمات نیز آشنا شود.
  - آموزش با نظارت باعث می‌شود مدل به خروجی‌های استاندارد و درست نزدیک‌تر شود.

- ✓ پایه‌ای برای آموزش مبتنی بر بازخورد انسانی (RLHF) :

  - این مرحله پیش‌نیاز مرحله بعدی است که در آن مدل از طریق بازخورد انسانی بهینه‌تر می‌شود. اگر این مرحله انجام نشود، RLHF نمی‌تواند مؤثر باشد.

۳. هم راستاسازی

الف) هدف اصلی از این مرحله چیست؟

هدف هم راستاسازی (Alignment) این است که مدل زبانی را طوری تنظیم کنیم که با ارزش‌ها و انتظارات انسانی هماهنگ شود.
یعنی حتی اگر مدل از نظر فنی درست کار کند و جواب درستی از نظر علمی تولید کند ، باید یاد بگیرد که چگونه پاسخ‌هایی مفید، محترمانه، بی‌طرف و ایمن ارائه دهد تا انسان ها بتوانند به راحتی با آن کار کنند و برای کودکان بدآموزی نداشته باشد.

ب) یکی از روش های رایج برای انجام هم راستاسازی را نام ببرید و نحوه‌ی کار آن را به طور خلاصه توضیح دهید.

یکی از رایج‌ترین روش‌ها یادگیری تقویتی با بازخورد انسانی (RLHF = Reinforcement Learning from Human Feedback) نام دارد. نحوه‌ی کار این روش به صورت زیر است:

1. ابتدا مدل روی یک مجموعه داده‌ی با نظارت آموزش می‌بیند (مرحله قبلی).
2. سپس انسان‌ها چند خروجی مختلف مدل را برای یک ورودی مشخص رتبه‌بندی می‌کنند (مثلاً پاسخ بهتر، محترمانه‌تر یا دقیق‌تر را انتخاب می‌کنند).
3. با استفاده از این رتبه‌بندی‌ها، یک مدل پاداش (reward model) آموزش داده می‌شود.
4. در نهایت، با استفاده از الگوریتم‌های یادگیری تقویتی (مثل PPO)، مدل زبانی به‌گونه‌ای بهینه می‌شود که پاسخ‌هایی تولید کند که مطابق با پاداش انسانی باشند.

سوال 3

دانلود و آماده سازی داده

از لینک داده شده، دیتاست را دانلود کرده و با استفاده از ستون text آن، corpus را میسازیم سپس آن را به دنباله هایی با طول یکسان و برابر با $block\_size$ تقسیم میکنیم تا برای مدل ترنسفورمری ما قابل پردازش باشد.

تنظیم هایپر پارامترها

کاملا مشابه با (جدول ۲: ابرپارامترهای پیشنهادی برای آموزش مدل) موجود در فایل توضیح تمرین، تنظیمات انجام شد.

پیاده سازی ماژول های مدل

در ابتدا در سطح کاراکتر توکنایزیشن را انجام میدهیم. در ادامه مقایسه ای بین روشهای مختلف توکنایزیشن را ارائه میدهیم:

1. توکن سازی در سطح کاراکتر

- مزایا:
  - تعداد واژگان بسیار کم (در حد مثلاً 100-200 کاراکتر)
  - عدم وجود مشکل در مدیریت کلمات ناشناخته (Out of Vocab)

- معایب:
  - دارای سرعت پردازش و تولید بسیار کمتر نسبت به حالت های بعدی
  - عدم درک ساختارهای معنایی سطح بالا مانند کلمات، عبارات وقواعد نحوی.
  - مدل باید وابستگی های طولانی مدت را در سطح کاراکتر یاد بگیرد، که بدیهتا دشوارتر از یادگیری وابستگی های طولانی مدت در سطح کلمه است.

2. توکن سازی در سطح کلمه

- مزایا
  - توالی های کوتاه تر و پردازش سریعتر
  - درک بهتر از ساختار معنایی جملات

- معایب
  - واژگان بسیار بزرگ (صدها هزار کلمه) میشود که برای مدل های چند زبانه مشکل ساز خواهد بود.
  - وجود مشکل در مدیریت کلمات ناشناخته (Out of Vocab) برعکس روش در سطح کاراکتر

3. توکن سازی در سطح زیرکلمه (Subword-Level)

- مزایا
  - ترکیبی از مزایای دو روش بالا را دارد.

- واژگان با اندازه متوسط (مثلاً 30-50 هزار توکن)
- توانایی مدیریت کلمات خارج از واژگان با تقسیم آن ها به زیرکلمات

- معایب
  - پیاده سازی پیچیده تر نسبت به توکن سازی در سطح کاراکتر

بنابراین:

- ✓ برای مدل‌های کوچک یا پروژه‌های آموزشی، توکن‌سازی در سطح کاراکتر مناسب است.
- ✓ برای مدل‌های بزرگ و کاربردهای واقعی، توکن‌سازی در سطح زیرکلمه (مانند BPE یا SentencePiece) ترجیح داده می‌شود، زیرا تعادل خوبی بین اندازه واژگان و توانایی مدل در درک ساختار زبان فراهم می‌کند.

آموزش مدل

تابع train_model پیاده سازی شده است و با ورودی گرفتن optimizer, scheduler, tokenizer, max_iter, eval_interval, batch_size مدل را آموزش میدهد. باید توجه کرد که برای آموزش مدل های زبانی بزرگ برخلاف شبکه های عصبی کانولوشنی و بازگشتی که در هر ایپاک روی کل داده آموزشی گرادیان تابع هزینه را محاسبه میکنیم و با استفاده از الگوریتم پس انتشار خطا وزن ها شبکه را آپدیت میکنیم، اینجا در هر ایپاک یک نمونه تصادفی از کل داده آموزشی را انتخاب میکنیم و گرادیان تابع هزینه را صرفا روی یک بسته محاسبه میکنیم. چون در مدل های زبانی با یک دنباله سر و کار داریم که میتواند هر چیزی باشد (برخلاف تسک های بینایی ماشین و پردازش زبان طبیعی که مثلا صرفا یک طبقه بندی ساده است) و با این نمونه گیری تصادفی به نوعی کوئری کاربر را داریم شبیه سازی میکنیم. به طور مشابه نحوه عملکرد مدل روی داده های ارزیابی نیز با همین منطق روی صرفا یک بسته اندازه گیری و گزارش شده است.

برای آموزش مدل های زبانی، بهینه ساز AdamW و CosineAnnealingLR برای Learning Rate Scheduler پیشنهاد شده است. در زیر به طور خلاصه توضیحیاتی درباره این دو روش نوشته ام:

AdamW یک الگوریتم بهینه‌سازی است که از بهینه‌ساز Adam مشتق شده است. تفاوت اصلی بین Adam و AdamW در نحوه مدیریت کاهش وزن (تنظیم‌کننده L2) است.

- در بهینه‌ساز اصلیAdam ، تنظیم‌کننده L2 (کاهش وزن) با افزودن یک عبارت متناسب با وزن‌ها مستقیماً به گرادیان‌ها اعمال می‌شود. این کار به‌طور مؤثری کاهش وزن را با نرخ‌های یادگیری تطبیقی ترکیب می‌کند که منجر به عملکرد نامطلوب یا دشوارتر شدن تنظیم مؤثر کاهش وزن می‌شود. اعمال کاهش وزن مستقیماً به گرادیان‌ها به این معنی است که این مقیاس‌بندی بر عبارت کاهش وزن نیز تأثیر می‌گذارد، که ممکن است مطلوب نباشد.
- AdamWکه توسط لوشچیلوف و هوتر معرفی شد، کاهش وزن را از فرآیند به‌روزرسانی گرادیان جدا می‌کند و به جای افزودن عبارت کاهش وزن به گرادیان‌ها، آن را مستقیماً پس از به‌روزرسانی گرادیان به وزن‌ها اعمال می‌کند. این جداسازی تضمین می‌کند که نرخ‌های یادگیری تطبیقی Adam با کاهش وزن تداخل پیدا نمی‌کنند و امکان کنترل دقیق‌تر بر تنظیم‌کننده را فراهم می‌کند.

چرا **AdamW** برای آموزش**LLM** ها مفید است:

- بهبود تعمیم‌پذیری LLM : الگوریتم AdamW با جدا کردن کاهش وزن، یک مکانیزم تنظیم‌کننده مؤثرتر را فراهم می‌کند. این به جلوگیری از حفظ داده‌های آموزشی توسط مدل کمک می‌کند و در عوض آن را به یادگیری ویژگی‌های تعمیم‌پذیرتر ترغیب می‌کند که منجر به عملکرد بهتر در داده‌های دیده نشده می‌شود.
- عملکرد بهتر :شواهد تجربی در جامعه توسعه دهندگان LLM نشان می‌دهد که AdamW اغلب منجر به همگرایی برتر و عملکرد نهایی بهتر در مقایسه با Adam اصلی می‌شود، به‌ویژه برای معماری‌های مبتنی بر ترانسفورمر در مقیاس بزرگ.

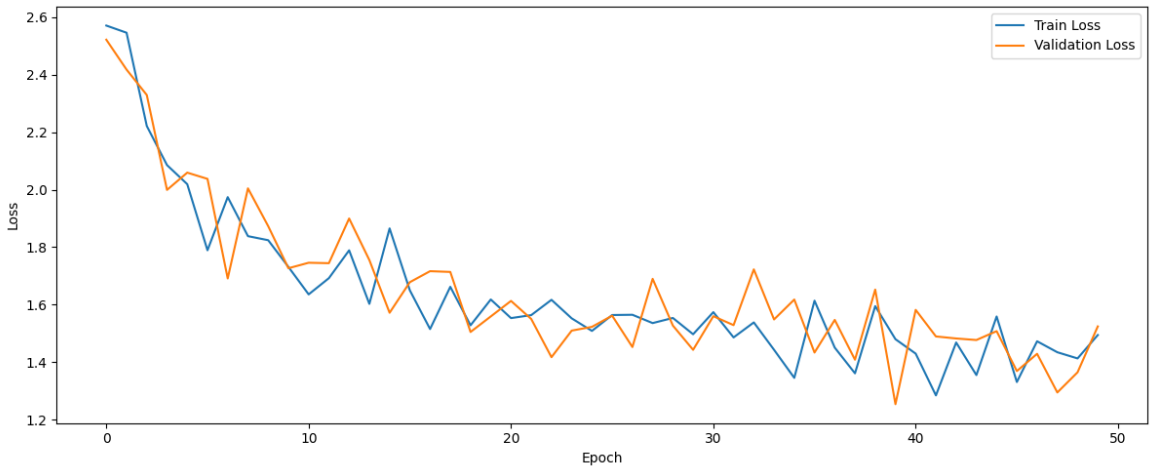
CosineAnnealingLRیک برنامه‌ریز نرخ یادگیری است که نرخ یادگیری را در طول آموزش به دنبال یک منحنی کسینوسی تنظیم می‌کند:

$$\eta_t = \eta_{min} + \frac{1}{2}(\eta_{max} - \eta_{min})\left(1 + cos\left(\frac{T_{Cur}}{T_{max}}\pi\right)\right)$$

که در آن:

- η_{min} حداقل نرخ یادگیری است.
- η_{max} نرخ یادگیری اولیه (حداکثر) است. در این تمرین 0.001.
- T_{Cur} تعداد تکرارها/اپوک‌ها از ابتدای آموزش است.
- T_{max} تعداد کل تکرارها/اپوک‌ها برای چرخه کاهنده کسینوسی است. در این تمرین 5000.

مقدار اولیه بالای نرخ یادگیری، باعث میشود تا مدل در ابتدا با سرعت خوبی آموزش ببیند و در ادامه با کاهش تدریجی نرخ یادگیری به همگراشدن مدل کمک میکند.



شکل 3- نمودار تابع هزینه حین آموزش

تولید متن

در این مثال، خروجی مدل را با استفاده از top_k decoding (top_k = 5) و temperature = 0.8 میبینید:

ورودی اولیه:

“Joey: “

خروجی مدل:

“their fored.

Yeah, three and star”

در این مثال، خروجی مدل را با استفاده از Beam Search Decoding (beam_width=3) و temperature = 1 میبینید:

ورودی اولیه:

“Joey: “

خروجی مدل:

“their fored.

Yeah, three and star”

باتوجه به نحوه پیاده سازی تابع generate_text که لاجیت ها را بر مقدار temperature تقسیم میکند:

- اگر مقدار temperature کمتر از یک باشد، خروجی مدل پیش بینی پذیرتر و دقیقتر و بدون خلاقیت خواهد بود.
- اگر مقدار temperature بیشتر از یک باشد، مدل خلاق تر و متنوع تر عمل خواهد کرد.

مراجع

1. A. Vaswani et al., “Attention is all you need,” Adv. Neural Inf. Process. Syst., vol. 30, 2017.
2. [2504.12501v1] Reinforcement Learning from Human Feedback
3. Decoupled Weight Decay Regularization