

# راهنمای نهایی برای تنظیم دقیق مدل‌های زبان

## بزرگ از مبانی تا پیشرفته:

یک بررسی جامع از فناوری‌ها، تحقیقات، بهترین روش‌ها، چالش‌ها و فرصت‌های تحقیقاتی کاربردی

**نویسنده‌گان:**

ونکاتش بالا و ادانی پارتاساراتی،

احتشام ظفر،

آفاق خان و ارسلان شهید

**رضا کرمی**

[GitHub](#)

۱۴۰۳ آبان

# **The Ultimate Guide to Fine-Tuning LLMs from Basics to Breakthroughs:**

**An Exhaustive Review of Technologies, Research, Best Practices, Applied  
Research Challenges and Opportunities**

**Venkatesh Balavadhani Parthasarathy, Ahtsham Zafar,  
Aafaq Khan, and Arsalan Shahid**

**@ CeADAR Connect Group**

CeADAR: Ireland's Centre for AI, University College Dublin, Belfield, Dublin, Ireland

{ venkatesh.parthasarathy, ahtsham.zafar, aafaq.khan, arsalan.shahid } @ ucd.ie

**Reza Karami**

<https://github.com/RezaKaramiDev/-The-Ultimate-Guide-to-Fine-Tuning-LLMs>

**August 2024**

## چکیده

این گزارش فی بهصورت جامع به بررسی فرآیند تنظیم دقیق مدل‌های زبانی بزرگ (LLMs) پرداخته و بینش‌های نظری و کاربردهای عملی را در هم می‌آمیزد. این گزارش با مرور تاریخچه تکامل مدل‌های زبانی بزرگ، به تحول آن‌ها از مدل‌های سنتی پردازش زبان طبیعی (NLP) و نقش کلیدی‌شان در سیستم‌های هوش مصنوعی مدرن می‌پردازد. در این تحلیل، روش‌های مختلف تنظیم دقیق، شامل رویکردهای ناظارت‌شده، بدون ناظارت و مبتنی بر دستورالعمل تفکیک می‌شوند و تأثیرات هر یک بر وظایف خاص مورد تأکید قرار می‌گیرد.

یک روند ساختاریافته هفت مرحله‌ای برای تنظیم دقیق مدل‌های زبانی بزرگ ارائه می‌شود که تمام مراحل از آماده‌سازی داده‌ها تا استقرار مدل را پوشش می‌دهد. در این مسیر، موارد مهمی از جمله استراتژی‌های جمع‌آوری داده، مدیریت مجموعه‌داده‌های نامتوازن، مقداردهی اولیه مدل و تکنیک‌های بهینه‌سازی با تمرکز ویژه بر تنظیم ابرپارامترها مورد بررسی قرار می‌گیرند. این گزارش همچنین روش‌های تنظیم دقیق کارآمد از نظر پارامترها مانند تطبیق با رتبه پایین (LORA) و تنظیم نیمه‌کامل را معرفی می‌کند که محدودیت‌های منابع را با عملکرد بهینه مدل متوازن می‌کنند.

این گزارش به بررسی تکنیک‌ها و تنظیمات پیشرفت‌هه تنظیم دقیق مانند تنظیم دقیق حافظه، ترکیب شبکه‌های خبره (MoE) و ترکیب عامل‌ها (MoA) می‌پردازد و نشان می‌دهد که چگونه این روش‌ها از شبکه‌های تخصصی و همکاری چند عاملی برای بهبود نتایج استفاده می‌کنند. روش‌های بهینه‌سازی سیاست نزدیک (PPO) و بهینه‌سازی ترجیح مستقیم (DPO) به عنوان رویکردهای نوآورانه برای هم‌راستاسازی مدل‌ها با ترجیحات انسانی مورد بحث قرار می‌گیرند، و همچنین مزایای بهینه‌سازی هرس و مسیریابی برای افزایش کارایی بررسی می‌شوند.

در بخش‌های پایانی، این گزارش به چارچوب‌های اعتبارسنجی، ناظارت پس از استقرار و تکنیک‌های بهینه‌سازی برای استنتاج می‌پردازد. همچنین، استقرار مدل‌های زبانی بزرگ در پلتفرم‌های توزیع شده و مبتنی بر ابر را پوشش می‌دهد. علاوه بر این، موضوعات پیشرفت‌های مانند مدل‌های زبانی چندسانه‌ای و تنظیم دقیق برای پردازش صوت و گفتار نیز مطرح می‌شوند و چالش‌های نوظهوری همچون مقیاس‌پذیری، حریم خصوصی و مسئولیت‌پذیری بررسی می‌شوند.

این گزارش به عنوان یک راهنمای جامع برای پژوهشگران و متخصصان در نظر گرفته شده و بینش‌های کاربردی در زمینه تنظیم دقیق مدل‌های زبانی بزرگ را ارائه می‌دهد و به چالش‌ها و فرصت‌های ذاتی این حوزه در حال تکامل سریع می‌پردازد.

## فهرست

۱	فصل اول - مقدمه
۱۳	فصل دوم - فرایند تنظیم دقیق هفت مرحله ای برای مدل های زبان بزرگ
۱۸	فصل سوم - مرحله اول: آماده سازی داده ها
۲۷	فصل چهارم - مرحله دوم: راه اندازی مدل
۳۲	فصل پنجم - مرحله سوم: تنظیمات آموزش
۴۴	فصل ششم - مرحله چهارم» انتخال تکنیک های تنظیم دقیق و پیکربندی های مناسب مدل
۷۶	فصل هفتم - مرحله پنجم: ارزیابی و اعتبار سنجی
۸۸	فصل هشتم - مرحله ششم: استقرار
۹۸	فصل نهم - مرحله هفتم: نظارت و نگهداری
۱۰۴	فصل دهم - پلتفرم ها و فریمورک های صنعتی برای تنظیم دقیق مدل های زبانی بزرگ
۱۲۶	فصل یازدهم - مدل های چند رسانه ای و تنظیم دقیق آنها
۱۳۸	فصل دوازدهم - چالش های باز و جهت های تحقیق
۱۵۰	واژه نامه
۱۵۵	منابع

# فصل یک

## مقدمه

### یک - یک: پیشینه مدل‌های زبان بزرگ (LLMs<sup>۱</sup>)

مدل‌های زبان بزرگ (LLMs) یک پیشرفت بزرگ در سیستم‌های محاسباتی محسوب می‌شوند که قادر به درک و تولید زبان انسانی هستند. این مدل‌ها بر پایه مدل‌های زبان سنتی مانند مدل‌های N-gram [۱] ساخته شده‌اند و محدودیت‌هایی مانند مدیریت کلمات نادر، بیش‌برازش<sup>۲</sup>، و به دام انداختن الگوهای پیچیده زبانی را برطرف می‌کنند. از نمونه‌های قابل توجه این مدل‌ها می‌توان به GPT-3 و GPT-4 [۲] اشاره کرد که از مکانیزم خودتوجهی (self-attention) در معماری‌های Transformer بهره می‌برند تا داده‌های متوالی را به صورت کارآمد مدیریت کرده و وابستگی‌های بلندمدت را درک کنند.

از جمله پیشرفتهای کلیدی این مدل‌ها، یادگیری درون‌متنی<sup>۳</sup> برای تولید متن‌های منسجم بر اساس دستورات ورودی و یادگیری تقویتی با بازخورد انسانی<sup>۴</sup> (RLHF) [۳] است که از پاسخ‌های انسانی برای بهبود و اصلاح مدل‌ها استفاده می‌کند. تکنیک‌هایی مانند مهندسی دستورات<sup>۵</sup>، پاسخگویی به سوالات و تعاملات مکالمه‌ای باعث پیشرفتهای چشمگیری در حوزه پردازش زبان طبیعی<sup>۶</sup> (NLP) شده‌اند [۴].

<sup>۱</sup> Large Language Models

: در یادگیری ماشین، Overfitting به وضعیتی گفته می‌شود که یک مدل به قدری دقیق و بهخصوص به داده‌های آموزشی می‌چسبد که عملکرد آن در داده‌های جدید یا تست کاهش می‌یابد. به عبارت دیگر، مدل به جزئیات و ناهنجاری‌های داده‌های آموزشی آنقدر حساس شده که نمی‌تواند به خوبی داده‌های نادیده و جدید را تعمیم دهد.

به عنوان مثال، در overfitting، مدل ممکن است به جای یادگیری الگوهای کلی، تمام نویزها و ناهنجاری‌های داده‌های آموزشی را به عنوان ویژگی‌های اصلی در نظر بگیرد. این موضوع باعث می‌شود که در داده‌های جدید و خارج از مجموعه آموزشی عملکرد خوبی نداشته باشد و دقت پایینی نشان دهد. راهکارهایی برای مقابله با overfitting:

- کاهش پیچیدگی مدل (مثل استفاده از تعداد لایه‌ها و نرون‌های کمتر در شبکه عصبی)
- افزایش داده‌های آموزشی (مثل جمع‌آوری داده‌های بیشتر یا استفاده از تکنیک‌های داده‌افزایی)
- استفاده از تکنیک‌های منظم‌سازی (مثل L1 و L2 regularization یا dropout در شبکه‌های عصبی)

<sup>3</sup> in-context learning

<sup>4</sup> Reinforcement Learning from Human Feedback

<sup>5</sup> prompt engineering

<sup>6</sup> natural language processing

## یک – دو: توسعه تاریخی و نقاط عطف کلیدی

مدل‌های زبان به عنوان یک بخش اساسی از پردازش زبان طبیعی (NLP) شناخته می‌شوند و از تکنیک‌های ریاضی برای تعمیم قواعد و دانش زبانی در وظایف مرتبط با پیش‌بینی و تولید استفاده می‌کنند. طی چندین دهه، مدل‌سازی زبان از مدل‌های زبان آماری اولیه<sup>۷</sup> (SLMs) به مدل‌های زبان بزرگ پیشرفته امروزی (LLMs) تکامل یافته است. این پیشرفت سریع به مدل‌های LLM امکان داده تا متن را در سطحی پردازش، درک و تولید کنند که قابل مقایسه با توانایی‌های انسانی است [۵, ۶].

شكل ۱.۱ نشان‌دهنده تکامل مدل‌های زبان بزرگ از رویکردهای آماری اولیه تا مدل‌های پیشرفته کنونی است.

### یک – سه: تکامل از مدل‌های سنتی NLP به مدل‌های زبان بزرگ (LLMs) پیشرفته

برای درک مدل‌های زبان بزرگ (LLMs)، باید مسیر توسعه مدل‌های زبان را از مراحل مختلفی مانند مدل‌های زبان آماری (SLMs)، مدل‌های زبان عصبی<sup>۸</sup> (NLMs)، مدل‌های زبان از پیش آموزش‌دیده<sup>۹</sup> (PLMs)، و در نهایت مدل‌های زبان بزرگ (LLMs) دنبال کرد.

#### یک – سه – یک: مدل‌های زبان آماری (SLMs)

مدل‌های زبان آماری (SLMs) که در دهه ۱۹۹۰ ظهر کردند، از روش‌های احتمالاتی برای تحلیل زبان طبیعی استفاده می‌کنند تا احتمال وقوع جملات در یک متن را تعیین کنند. به عنوان مثال، احتمال جمله *"I am very happy"*

به صورت زیر محاسبه می‌شود:

$$(1.1) \quad P(S) = P(\omega_1, \omega_2, \omega_3, \omega_4) = P(I, am, very, happy)$$

این احتمال را می‌توان با استفاده از احتمالات شرطی محاسبه کرد:

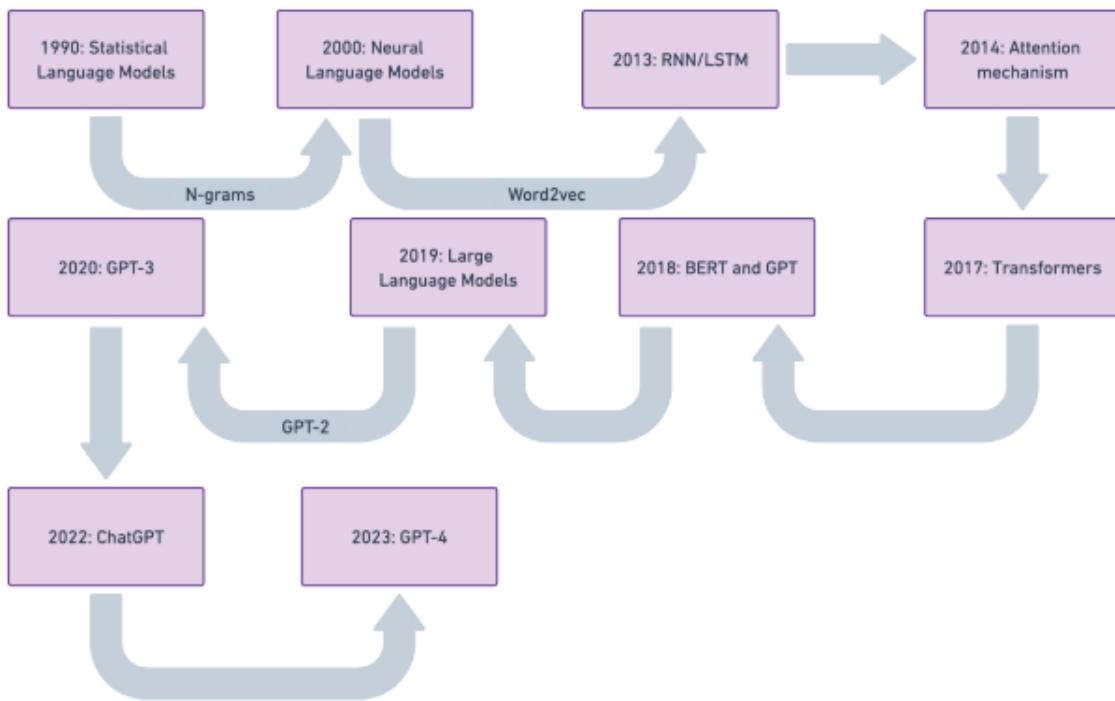
$$(1.2) \quad P(I, am, very, happy) = P(I) \cdot P(am | I) \cdot P(very | I, am) \cdot P(happy | I, am, very)$$

<sup>7</sup> Statistical language models

<sup>8</sup> Neural Language Models

<sup>9</sup> Pre-trained Language Models

احتمالات شرطی با استفاده از روش حداقل احتمال<sup>۱۰</sup> (MLE) تخمین زده می‌شوند.



شکل ۱.۱: یک جدول زمانی تاریخی که تکامل مدل‌های زبان بزرگ (LMs) را از سال ۱۹۹۰ تا ۲۰۲۳ نشان می‌دهد. این پیشرفت با مدل‌های آماری اولیه مانند N-gram آغاز شده، از مدل‌های زبان عصبی مانند Word2Vec و RNN/LSTM عبور می‌کند، و به دوران مدل‌های از پیش آموزش دیده با معرفی ترانسفورمرها و مکانیزم‌های توجه می‌رسد. این شکل، نقاط عطف مهمی را بر جسته می‌کند، از جمله توسعه BERT، GPT، و نوآوری‌های اخیر مانند GPT-4 و ChatGPT که نشان‌دهنده پیشرفت سریع فناوری LMs در طول زمان است. (اقتباس از [۶])

$$P(\omega_i \mid \omega_1 \omega_2 \cdots \omega_{i-1}) = \frac{C(\omega_1 \omega_2 \cdots \omega_i)}{C(\omega_1 \omega_2 \cdots \omega_{i-1})} \quad (1.3)$$

### یک - سه - دو: مدل‌های زبان عصبی (NLMs)

مدل‌های زبان عصبی از شبکه‌های عصبی برای پیش‌بینی توالی کلمات استفاده می‌کنند و محدودیت‌های مدل‌های زبان آماری را برطرف می‌کنند. بردارهای کلمات به رایانه‌ها کمک می‌کنند تا معنای کلمات را درک کنند. ابزارهایی مانند Word2Vec [۷] کلمات را در فضایی از بردارها نمایش می‌دهند، به طوری که روابط معنایی بین کلمات در زوایای بین بردارها منعکس می‌شود. مدل‌های زبان عصبی از نورون‌های متصل به هم تشکیل شده‌اند که در لایه‌هایی سازمان‌دهی شده‌اند و شباهتی با ساختار مغز انسان دارند. لایه ورودی بردارهای

<sup>۱۰</sup> Maximum Likelihood Estimation

کلمات را به هم متصل می‌کند، لایه مخفی یکتابع فعال‌سازی غیرخطی را اعمال می‌کند و لایه خروجی با استفاده از تابع  $\text{Softmax}^{11}$  مقادیر را به یک توزیع احتمالی تبدیل می‌کند تا کلمات بعدی را پیش‌بینی کند.

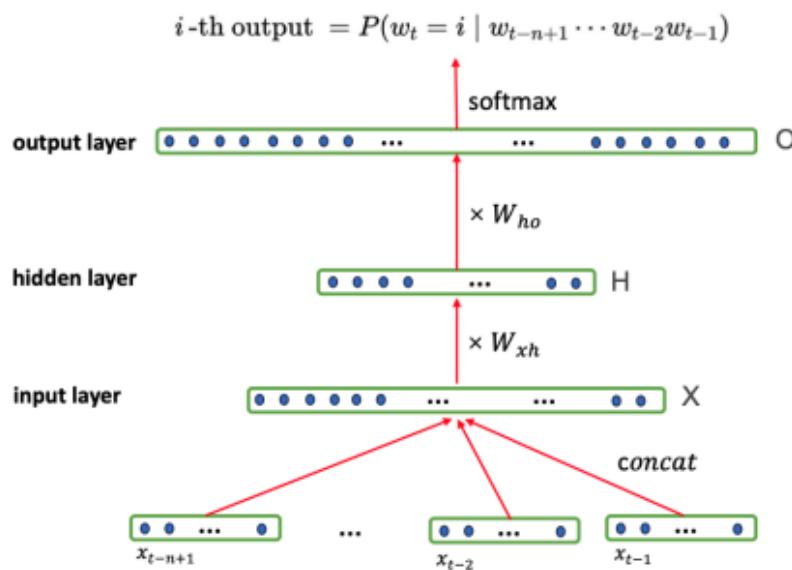
شکل ۱.۲ ساختار مدل‌های زبان عصبی (NLMS) را نشان می‌دهد و لایه‌ها و اتصالاتی که برای پیش‌بینی کلمات بعدی استفاده می‌شوند را بر جسته می‌کند.

### یک – سه – سه: مدل‌های زبان از پیش آموزش دیده (PLMs)

مدل‌های زبان از پیش آموزش دیده (PLMs) ابتدا بر روی حجم زیادی از متون بدون برچسب آموزش می‌بینند تا ساختارهای اساسی زبان را درک کنند (پیش‌آموزش). سپس این مدل‌ها بر روی یک مجموعه داده کوچک‌تر و خاص یک وظیفه تنظیم دقیق می‌شوند. این الگوی پیش‌آموزش و تنظیم دقیق، که در مدل‌هایی مانند GPT-2 [۸] و BERT [۹] دیده می‌شود، به ایجاد معماری‌های متنوع و مؤثر منجر شده است.

### یک – سه – چهار: مدل‌های زبان بزرگ (LLMs)

مدل‌های زبان بزرگ (LLMs) مانند LLaMA، GPT-3، GPT-4، PaLM [۱۰] و [۱۱] بر روی مجموعه‌های عظیمی از متون با ده‌ها میلیارد پارامتر آموزش دیده‌اند. این مدل‌ها یک فرآیند دو مرحله‌ای را طی می‌کنند: ابتدا پیش‌آموزش بر روی یک مجموعه بزرگ از متون و سپس این رویکرد به مدل‌های زبان بزرگ (LLMs) این امکان را می‌دهد که دستورات و ارزش‌های انسانی را بهتر درک کنند.



$\text{Softmax function}^{11}$  یک تابع ریاضی است که برای تبدیل مجموعه‌ای از اعداد به مقادیر احتمال استفاده می‌شود. این تابع، به‌ویژه در مدل‌های دسته‌بندی چند کلاسه (مانند شبکه‌های عصبی) کاربرد دارد و کمک می‌کند که خروجی‌های مدل به صورت توزیع احتمالی در بیاید.

تابع Softmax به هر مقدار عددی در خروجی یک مدل (که معمولاً نمرات اولیه یا logits نامیده می‌شوند) یک مقدار احتمال اختصاص می‌دهد که جمع آن‌ها برابر با ۱ خواهد بود. به عبارت دیگر، Softmax برای هر کلاس یک احتمال نسبی تخصیص می‌دهد و کلاس با بالاترین احتمال به عنوان نتیجه نهایی انتخاب می‌شود.

شکل ۱.۲: یک نمایش شماتیک از مدل‌های زبان عصبی که معماری لایه‌ای آن را به تصویر می‌کشد. در این معماری، لایه ورودی داده‌های توالی را پردازش می‌کند، لایه پنهان وابستگی‌ها را شناسایی می‌کند و لایه خروجی پیش‌بینی‌ها را تولید می‌کند. این شکل تأکید می‌کند بر جریان اطلاعات از طریق تجمعی (concatenation) و ضرب ماتریسی (matrix multiplications) که در نهایت به یک توزیع احتمالی (probability distribution) از طریق تابع سافت‌مکس (softmax function) منجر می‌شود.  
(منبع: [۶])

## یک – چهار: مروری بر مدل‌های زبان بزرگ (LLMs) پیشرو کنونی

مدل‌های زبان بزرگ ابزارهای قدرتمندی در پردازش زبان طبیعی هستند که قادر به انجام کارهایی مانند ترجمه، خلاصه‌سازی و تعاملات گفت‌و‌گویی می‌باشند. پیشرفت‌های صورت‌گرفته در معماری‌های ترانسفورمر، قدرت محاسباتی و مجموعه‌داده‌های وسیع، باعث موفقیت این مدل‌ها شده است. این مدل‌ها تقریباً عملکردی معادل انسان را ارائه می‌دهند و به همین دلیل برای تحقیقات و پیاده‌سازی‌های عملی بسیار بالرزش هستند. توسعه سریع LLM‌ها موجب تحقیق در زمینه نوآوری‌های معماری، استراتژی‌های آموزشی، افزایش طول زمینه<sup>۱۲</sup>، تکنیک‌های تنظیم دقیق و ادغام داده‌های چندمدلی<sup>۱۳</sup> شده است. کاربردهای آن‌ها فراتر از NLP است و به تعاملات انسان-ربات و ایجاد سیستم‌های هوش مصنوعی شهریوری کمک می‌کند. این موضوع اهمیت بررسی‌های جامع که آخرین پیشرفت‌ها را گرد هم می‌آورد، به خوبی نشان می‌دهد [۱۲].

شکل ۱ - ۳ نمای کلی از مدل‌های زبان بزرگ (LLMs) پیشرو کنونی ارائه می‌دهد و توانایی‌ها و کاربردهای آن‌ها را روشن می‌کند.

## یک – پنج: تنظیم دقیق<sup>۱۴</sup> چیست؟

تنظیم دقیق استفاده از یک مدل پیش‌آموزش‌دیده، مانند سری مدل‌های GPT از OpenAI است. این فرآیند شامل آموزش بیشتر بر روی یک مجموعه داده خاص با حجم کمتر است. این رویکرد بر اساس دانش پیشین مدل بنا شده و عملکرد آن را در وظایف خاص با نیازهای داده و محاسباتی کاهش یافته بهبود می‌بخشد.

تنظیم دقیق الگوها و ویژگی‌های آموخته شده مدل پیش‌آموزش‌دیده را به وظایف جدید منتقل می‌کند و بدین ترتیب عملکرد را بهبود بخشدید و نیاز به داده‌های آموزشی را کاهش می‌دهد. این روش در NLP برای کارهایی مانند طبقه‌بندی متن، تحلیل احساسات و پاسخ به سوالات محبوب شده است.

<sup>12</sup> context lengths

<sup>13</sup> integrating multi-modal data

<sup>14</sup> Fine-Tuning



شکل ۱.۳: نقشه ذهنی که ابعاد مختلف مدل‌های زبان بزرگ (LLMs) را پوشش می‌دهد و جنبه‌هایی از روش‌های پیش‌آموزش و تنظیم دقیق تا کارایی، ارزیابی، استنباط و دامنه‌های کاربرد را در بر می‌گیرد. هر بعد به تکنیک‌های خاص، چالش‌ها و مثال‌هایی از مدل‌ها که ویژگی‌های مورد بحث را تجسم می‌بخشند، مرتبط است. این نمودار به عنوان یک نمای کلی از ملاحظات چندبعدی در توسعه و پیاده‌سازی LLM‌ها عمل می‌کند. (منبع: [۱۳])

## یک – شش: انواع تنظیم دقیق LLM

### یک – شش – یک: تنظیم دقیق بدون ناظارت<sup>۱۵</sup>

این روش به داده‌های برچسب‌گذاری شده نیاز ندارد. در عوض، مدل زبان بزرگ (LLM) به یک مجموعه بزرگ از متون بدون برچسب از حوزه هدف دسترسی پیدا می‌کند و درک خود از زبان را بهبود می‌بخشد. این رویکرد برای حوزه‌های جدیدی مانند زمینه‌های حقوقی یا پزشکی مفید است، اما برای وظایف خاصی مانند طبقه‌بندی یا خلاصه‌سازی کمتر دقیق است..

<sup>۱۵</sup> Unsupervised Fine-Tuning

## یک - شش - دو: تنظیم دقیق تحت نظارت<sup>۱۶</sup>

تنظیم دقیق تحت نظارت شامل ارائه داده‌های برچسب‌گذاری شده به LLM است که به وظیفه هدف اختصاص یافته است. به عنوان مثال، تنظیم دقیق یک LLM برای طبقه‌بندی متن در یک زمینه تجاری، از مجموعه داده‌هایی از تکه‌های متنی با برچسب‌های کلاس استفاده می‌کند. در حالی که این روش مؤثر است، نیاز به داده‌های برچسب‌گذاری شده قابل توجهی دارد که می‌تواند هزینه‌بر و زمان‌بر باشد.

## یک - شش - سه: تنظیم دقیق دستوری از طریق مهندسی پرامپت<sup>۱۷</sup>

این روش به ارائه دستورات به زبان طبیعی به LLM متکی است که برای ایجاد دستیارهای تخصصی مفید است. این روش نیاز به مقادیر زیادی از داده‌های برچسب‌گذاری شده را کاهش می‌دهد، اما به شدت به کیفیت پرامپت‌ها وابسته است.

## یک - هفت: پیش‌آموزش در مقابل تنظیم دقیق

جدول ۱.۱ مقایسه‌ای بین پیش‌آموزش و تنظیم دقیق ارائه می‌دهد و ویژگی‌ها و فرآیندهای مربوط به هر یک را هایلایت می‌کند.

Fine-tuning – تنظیم دقیق	Pre-training – پیش آموز	Aspect
سازگاری یک مدل پیش‌آموزش‌دیده برای وظایف خاص	آموزش بر روی مقدار زیادی از داده‌های متنی بدون برچسب	تعریف
داده‌های برچسب‌گذاری شده کوچک و خاص برای هر وظیفه	داده‌های متنی بدون برچسب وسیع و متنوع	نیاز به داده
تخصصی کردن مدل برای وظایف خاص	ساخت دانش عمومی زبانی	هدف
جمع‌آوری داده خاص وظیفه، تغییر لایه آخر برای وظیفه، آموزش بر روی مجموعه داده جدید، تولید خروجی بر اساس وظایف	جمع‌آوری داده، آموزش بر روی مجموعه داده بزرگ، پیش‌بینی کلمه/دبنه بعدی	فرآیند
لایه‌های آخر برای وظیفه جدید سازگار شده‌اند	کل مدل آموزش داده شده	تفصیلات مدل
کمتر (مجموعه داده بزرگ، مدل پیچیده)	بالا (مجموعه داده بزرگ، مدل پیچیده)	هزینه محاسباتی
روزها تا هفته‌ها	هفته‌ها تا ماه‌ها	مدت زمان آموزش
بهبود عملکرد خاص برای وظیفه	درک عمومی زبان	هدف
تنظیم دقیق LLaMA 3 برای خلاصه سازی	LLaMA 3, GPT	مثال‌ها

<sup>۱۶</sup> Supervised Fine-Tuning (SFT)

<sup>۱۷</sup> Instruction Fine-Tuning via Prompt Engineering

جدول ۱.۱: نمای کلی مقایسه‌ای از پیش‌آموزش و تنظیم دقیق در مدل‌های زبان بزرگ (LLMs). این جدول تفاوت‌های کلیدی بین مراحل پیش‌آموزش و تنظیم دقیق را در جنبه‌های مختلفی مانند تعریف، نیازهای داده‌ای، اهداف، فرآیندها، تغییرات مدل، هزینه‌های محاسباتی، مدت زمان آموزش و اهداف مربوطه خود با مثال‌هایی که مدل‌ها و وظایف خاص را نشان می‌دهند، outline می‌کند. پیش‌آموزش شامل آموزش وسیع بر روی حجم زیادی از داده‌های بدون برچسب است تا دانش عمومی زبانی را ایجاد کند، در حالی که تنظیم دقیق مدل‌های پیش‌آموزش دیده را به وظایف تخصصی با استفاده از مجموعه‌های داده کوچک و برچسب‌گذاری شده تطبیق می‌دهد و بر بهبود عملکرد خاص وظیفه تمرکز دارد.

## یک – هشت: اهمیت تنظیم دقیق LLM‌ها

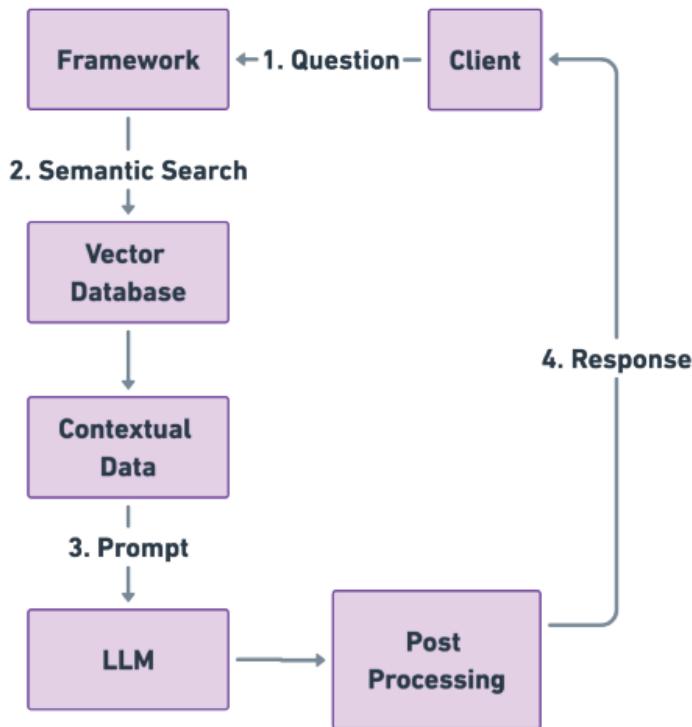
۱. **یادگیری انتقالی:** تنظیم دقیق از دانشی که در طول پیش‌آموزش به دست آمده بهره می‌برد و آن را به وظایف خاص با زمان و منابع محاسباتی کمتری تطبیق می‌دهد.
۲. **کاهش نیاز به داده:** تنظیم دقیق به داده‌های برچسب‌گذاری شده کمتری نیاز دارد و بر تنظیم ویژگی‌های پیش‌آموزش دیده برای وظیفه هدف تمرکز می‌کند.
۳. **بهبود تعمیم‌پذیری:** تنظیم دقیق توانایی مدل را برای تعمیم به وظایف یا حوزه‌های خاص افزایش می‌دهد و ویژگی‌های عمومی زبان را ثبت و شخصی‌سازی می‌کند.
۴. **استقرار کارآمد مدل:** مدل‌های تنظیم دقیق شده برای کاربردهای دنیای واقعی کارآمدتر هستند، زیرا از نظر محاسباتی کارآمد و مناسب برای وظایف خاص هستند.
۵. **سازگاری با وظایف مختلف:** LLM‌های تنظیم دقیق شده می‌توانند به طیف وسیعی از وظایف سازگار شوند و در کاربردهای مختلف به خوبی عمل کنند بدون اینکه به معماری‌های خاص وظیفه نیاز داشته باشند.
۶. **عملکرد خاص حوزه:** تنظیم دقیق به مدل‌ها اجازه می‌دهد در وظایف خاص حوزه عملکرد خوبی داشته باشند و به جزئیات و واژگان حوزه هدف سازگار شوند.
۷. **همگرایی سریع‌تر:** تنظیم دقیق معمولاً همگرایی سریع‌تری را به ارمغان می‌آورد، زیرا با وزن‌هایی آغاز می‌شود که از قبل ویژگی‌های عمومی زبان را ثبت کرده‌اند.

## یک – نه: تولید افزوده با بازیابی<sup>۱۸</sup>

یکی از روش‌های محبوب برای استفاده از داده‌های خودتان، گنجاندن آن در دستور پرسش زمانی است که از مدل LLM سؤال می‌شود. این رویکرد که به عنوان تولید افزوده با بازیابی (RAG) شناخته می‌شود، شامل بازیابی داده‌های مرتبط و استفاده از آن به عنوان زمینه اضافی برای LLM است. به جای اتکا صرف به دانش موجود در داده‌های آموزشی، یک جریان کار RAG اطلاعات مربوطه را می‌کشد و LLM‌های استاتیک را با بازیابی داده‌های

<sup>18</sup> Retrieval Augmented Generation (RAG)

زمان واقعی متصل می‌کند. با معماری RAG، سازمان‌ها می‌توانند هر مدل LLM را مستقر کنند و آن را به گونه‌ای تقویت کنند که نتایج مرتبطی را با ارائه مقدار کمی از داده‌های خود بازگرداند (برای مشاهده جریان کار بصری، به شکل ۱.۴ مراجعه کنید). این فرآیند هزینه‌ها و زمان مرتبط با تنظیم دقیق یا پیش‌آموزش مدل را کنار می‌گذارد.



شکل ۱.۴: تصویری از مراحل خط لوله تولید افزوده با بازیابی (RAG) سنتی، که فرآیند توالی از پرسش مشتری تا تولید پاسخ را نمایش می‌دهد. این خط لوله با پرسش مشتری آغاز می‌شود، و سپس به جستجوی معنایی در یک پایگاه داده برداری می‌پردازد که داده‌ها را به طور متنی غنی‌تر می‌کند، قبل از اینکه یک درخواست برای مدل زبان بزرگ (LLM) تولید کند. پاسخ نهایی پس از پردازش مجدد به مشتری بازگردانده می‌شود.

## یک – نه – یک: مراحل و فرآیند سنتی بازیابی و تولید پاسخ

۱. **فهرست‌گذاری داده‌ها:** سازماندهی داده‌ها به صورت کارآمد برای بازیابی سریع. این شامل پردازش، تقسیم، و ذخیره داده‌ها در یک پایگاه داده برداری با استفاده از استراتژی‌های فهرست‌گذاری مانند فهرست‌گذاری جستجو، فهرست‌گذاری برداری و فهرست‌گذاری ترکیبی است.
۲. **پردازش پرسش ورودی:** پالایش پرسش‌های کاربر برای بهبود سازگاری با داده‌های فهرست‌گذاری شده. این می‌تواند شامل ساده‌سازی یا تبدیل برداری پرسش‌ها برای بهبود کارایی جستجو باشد.
۳. **جستجو و رتبه‌بندی:** بازیابی و رتبه‌بندی داده‌ها بر اساس ارتباط با استفاده از الگوریتم‌های جستجو مانند BM25 و مدل‌های یادگیری عمیق مانند BERT برای تفسیر نیت و زمینه پرسش.

۴. تقویت درخواست: گنجاندن اطلاعات مرتبط از نتایج جستجو در پرسش اصلی برای ارائه زمینه اضافی به LLM، که بهبود دقت و ارتباط پاسخ‌ها را تضمین می‌کند.

۵. تولید پاسخ: استفاده از درخواست تقویت شده برای تولید پاسخ‌هایی که دانش LLM را با داده‌های خاص و بهروز ترکیب می‌کند و به این ترتیب پاسخ‌های با کیفیت و متنی محکم ارائه می‌دهد.

### یک - نه - دو: مزایای استفاده از بازیابی و تولید پاسخ

- پاسخ‌های به روز و دقیق: با استفاده از داده‌های خارجی به روز، پاسخ‌های LLM را بهبود می‌بخشد و دقت و ارتباط آن‌ها را افزایش می‌دهد.

- کاهش پاسخ‌های نادرست: خروجی‌های LLM را بر اساس دانش مرتبط می‌سازد و خطر تولید اطلاعات نادرست را کاهش می‌دهد.

- پاسخ‌های خاص حوزه: پاسخ‌های مناسب با زمینه را ارائه می‌دهد که برای داده‌های اختصاصی سازمان طراحی شده است.

- کارآمدی و صرفه‌جویی در هزینه: یک روش مقرر به صرفه برای سفارشی‌سازی LLM‌ها بدون نیاز به تنظیم دقیق گسترده مدل‌ها ارائه می‌دهد.

### یک - نه - سه: چالش‌ها و ملاحظات در ارائه بازیابی و تولید پاسخ

۱. تجربه کاربر: اطمینان از زمان‌های پاسخ سریع مناسب برای برنامه‌های کاربردی زمان واقعی.

۲. صرفه‌جویی در هزینه: مدیریت هزینه‌های مربوط به ارائه میلیون‌ها پاسخ.

۳. دقت: اطمینان از دقت خروجی‌ها برای جلوگیری از اطلاعات نادرست.

۴. به روز بودن و ارتباط: نگهداری پاسخ‌ها و محتوا با جدیدترین داده‌ها.

۵. آگاهی از زمینه تجاری: هم‌استایی پاسخ‌های LLM با زمینه‌های تجاری خاص.

۶. مقیاس‌پذیری خدمات: مدیریت ظرفیت افزایش یافته در حین کنترل هزینه‌ها.

۷. امنیت و حاکمیت: پیاده‌سازی پروتکل‌ها برای امنیت داده‌ها، حریم خصوصی و حاکمیت.

### یک - نه - چهار: موارد استفاده و مثال‌ها

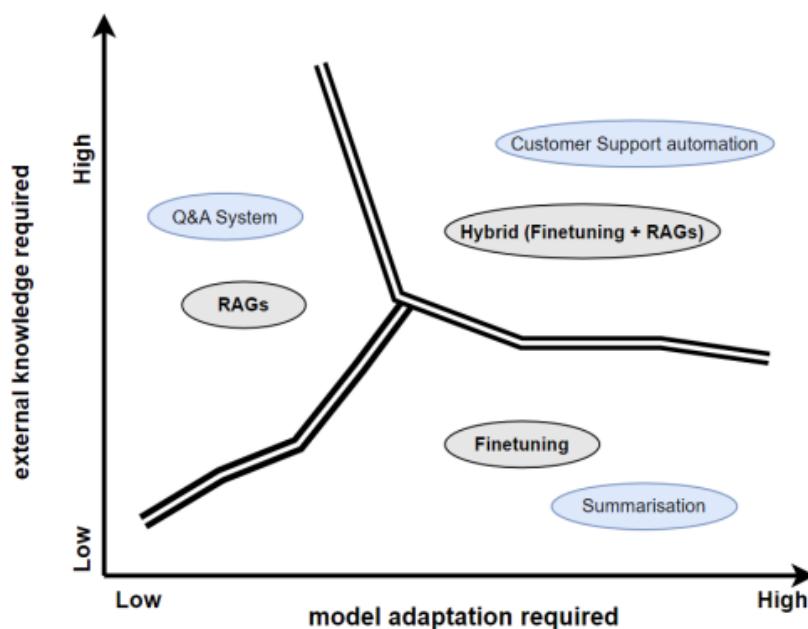
۱. چت‌بات‌های سؤال و پاسخ: ادغام LLM‌ها با چت‌بات‌ها برای تولید پاسخ‌های دقیق از اسناد شرکتی، که به بهبود پشتیبانی مشتری کمک می‌کند.

۲. تقویت جستجو: بهبود موتورهای جستجو با پاسخ‌های تولید شده توسط LLM برای جستجوهای اطلاعاتی دقیق‌تر.

۳. موتور دانش: استفاده از LLMها برای پاسخ به پرسش‌های مربوط به عملکردهای داخلی، مانند منابع انسانی و انطباق، با استفاده از داده‌های شرکتی.

### یک - نه - پنج: ملاحظات برای انتخاب بین بازیابی و تولید پاسخ و تنظیم دقیق

زمانی که به دسترسی به داده‌های خارجی فکر می‌کنید، RAG احتمالاً گزینه برتر برای برنامه‌هایی است که نیاز به دسترسی به منابع داده خارجی دارند. از سوی دیگر، تنظیم دقیق بیشتر مناسب است اگر نیاز داشته باشد که مدل رفتار، سبک نوشتن یا دانش خاص حوزه را تنظیم کند. از نظر کاهش توهمنات و اطمینان از دقت، سیستم‌های بازیابی و تولید پاسخ معمولاً عملکرد بهتری دارند زیرا کمتر مستعد تولید اطلاعات نادرست هستند. اگر داده‌های آموزشی برچسب‌گذاری شده و خاص حوزه فراوانی دارید، تنظیم دقیق می‌تواند منجر به رفتار مدل سفارشی‌تر شود، در حالی که سیستم‌های بازیابی و تولید پاسخ گزینه‌های مقاومی هستند زمانی که چنین داده‌هایی کمیاب هستند. سیستم‌های بازیابی و تولید پاسخ با قابلیت‌های بازیابی داده‌های دینامیک برای محیط‌هایی که داده‌ها به طور مکرر به روزرسانی یا تغییر می‌کنند، مزیت دارند. علاوه بر این، اطمینان از شفافیت و قابلیت تفسیر فرآیند تصمیم‌گیری مدل بسیار مهم است. در این صورت، سیستم‌های بازیابی و تولید پاسخ بینش‌هایی ارائه می‌دهند که معمولاً در مدل‌های صرفاً تنظیم‌شده در دسترس نیستند. شکل ۱.۵ نمای بصری همراه با مثال‌های کاربردی را نشان می‌دهد.



شکل ۱.۵: نموداری که مقایسه‌ای از سازگاری مدل مورد نیاز در مقابل سطح دانش خارجی مورد نیاز در سناریوهای مختلف را نشان می‌دهد، و نقشه‌ای تولید افزوده با بازیابی (RAG)، تنظیم دقیق و کاربردهای ترکیبی آن‌ها در زمینه‌هایی مانند سیستم‌های سؤال و جواب، خودکارسازی پشتیبانی مشتری و وظایف خلاصه‌سازی برجسته می‌کند. (منبع [۱۴])

## یک – ۵: اهداف گزارش

### یک – ۵ – یک: اهداف و دامنه

هدف اصلی این گزارش، انجام یک تحلیل جامع از تکنیک‌های تنظیم دقیق برای مدل‌های زبان بزرگ (LLMs) است. این شامل بررسی مبانی نظری، استراتژی‌های پیاده‌سازی عملی و چالش‌ها می‌شود. گزارش به بررسی روش‌های مختلف تنظیم دقیق، کاربردهای آن‌ها و پیشرفت‌های اخیر می‌پردازد.

### یک – ۵ – ۲: سؤالات و مسائل کلیدی مطرح شده

این گزارش به سؤالات حیاتی در مورد تنظیم دقیق LLM‌ها می‌پردازد و از بینش‌های بنیادی در مورد LLM‌ها، تکامل آن‌ها و اهمیت آن‌ها در پردازش زبان طبیعی (NLP) آغاز می‌شود. این بخش تنظیم دقیق را تعریف می‌کند، آن را از پیش‌آموزش متمایز می‌کند و نقش آن را در سازگار کردن مدل‌ها برای وظایف خاص تأکید می‌کند. اهداف کلیدی شامل بهبود عملکرد مدل برای کاربردها و حوزه‌های هدفمند است.

گزارش یک فرآیند تنظیم دقیق ساختارمند را ترسیم می‌کند که شامل یک فرآیند با نمای بصری و توضیحات دقیق مراحل است. این بخش به پوشش استراتژی‌های پیاده‌سازی عملی می‌پردازد، از جمله پیاده‌سازی اولیه مدل، تعریف ابرپارامترها و تکنیک‌های تنظیم دقیق مانند تنظیم دقیق کارآمد پارامتر<sup>۱۹</sup> (PEFT) و تولید افزوده با بازیابی (RAG). کاربردهای صنعتی، روش‌های ارزیابی، چالش‌های پیاده‌سازی و پیشرفت‌های اخیر نیز مورد بررسی قرار می‌گیرند.

### یک – ۵ – ۳: نمای کلی از ساختار گزارش

بقیه گزارش، درک جامعی از تنظیم دقیق LLM‌ها را ارائه می‌دهد. فصل‌های اصلی شامل نگاهی عمیق به خط لوله تنظیم دقیق، کاربردهای عملی، هم‌راستایی مدل، معیارهای ارزیابی و چالش‌ها است. بخش‌های پایانی به بحث در مورد تکامل تکنیک‌های تنظیم دقیق می‌پردازند، چالش‌های تحقیقاتی جاری را بر جسته می‌کنند و بینش‌هایی برای محققان و کارشناسان ارائه می‌دهند.

<sup>۱۹</sup> Parameter-Efficient Fine-Tuning

## فصل دوم

### فرآیند تنظیم دقیق هفت مرحله‌ای برای مدل‌های زبان بزرگ (LLM)

تنظیم دقیق یک مدل زبان بزرگ (LLM) یک فرآیند جامع است که به هفت مرحله متمايز تقسیم می‌شود که هر یک برای سازگار کردن مدل پیش‌آموزش‌دیده با وظایف خاص و اطمینان از عملکرد بهینه ضروری است. این مراحل شامل همه چیز از آماده‌سازی اولیه داده‌ها تا استقرار و نگهداری نهایی مدل دقیق تنظیم شده می‌شود. با دنبال کردن این مراحل به صورت سیستماتیک، مدل به‌طور دقیق بهینه‌سازی و سفارشی‌سازی می‌شود تا نیازهای مشخص را برآورده کند و در نهایت توانایی آن برای تولید پاسخ‌های دقیق و مناسب با زمینه را بهبود بخشد. هفت مرحله شامل آماده‌سازی داده‌ها، آغاز به کار مدل، تنظیم محیط آموزش، تنظیم دقیق، ارزیابی و اعتبارسنجی، استقرار، و نظارت و نگهداری است.

شكل ۲.۱ فرآیند جامع تنظیم دقیق مدل‌های زبان بزرگ را نشان می‌دهد که شامل تمام مراحل لازم از آماده‌سازی داده‌ها تا نظارت و نگهداری است.

#### دو – یک: مرحله اول: آماده‌سازی داده‌ها

تنظیم دقیق یک مدل زبان بزرگ (LLM) با سازگار کردن مدل پیش‌آموزش‌دیده برای وظایف خاص آغاز می‌شود، به‌طوری‌که پارامترهای آن با استفاده از یک مجموعه داده جدید به‌روزرسانی شود. این شامل پاکسازی و فرمتدی مجموعه داده به‌گونه‌ای است که با وظیفه هدف مطابقت داشته باشد، مانند تنظیم دستورالعمل، تحلیل احساسات، یا نقشه‌برداری موضوع. مجموعه داده شامل جفت‌های <ورودی، خروجی> است که رفتار مورد نظر برای مدل را نشان می‌دهد.

برای مثال، در تنظیم دستورالعمل، مجموعه داده ممکن است به صورت زیر باشد:

```
### Human: $<Input Query>$  
### Assistant: $<Generated Output>$
```

در اینجا، «ورودی پرسش (Input Query)» چیزی است که کاربر می‌پرسد و «خروجی تولیدشده (Generated Output)» پاسخ مدل است. ساختار و سبک این جفت‌ها می‌تواند بر اساس نیازهای خاص وظیفه تنظیم شود.

## دو - دو: مرحله دوم: آغاز به کار مدل

آغاز به کار مدل فرآیند تنظیم پارامترها و پیکربندی‌های اولیه LLM قبل از آموزش یا استقرار آن است. این مرحله برای اطمینان از عملکرد بهینه مدل، آموزش کارآمد و جلوگیری از مشکلاتی مانند گرادیان‌های ناپدیدشونده یا منفجرشونده<sup>۲۰</sup> بسیار حیاتی است.

## دو - سه: مرحله سوم: تنظیم محیط آموزش

تنظیم محیط آموزش برای تنظیم دقیق LLM شامل پیکربندی زیرساخت‌های لازم برای سازگار کردن یک مدل موجود برای وظایف خاص است. این شامل انتخاب داده‌های آموزشی مرتبط، تعریف معماری و هایپرپارامترهای مدل، و اجرای تکرارهای آموزشی برای تنظیم وزن‌ها و بایاس‌های مدل است. هدف این است که عملکرد LLM در تولید خروجی‌های دقیق و مناسب با زمینه که به طور خاص به برنامه‌های خاصی مانند تولید محتوا، ترجمه یا تحلیل احساسات مربوط می‌شود، بهبود یابد. تنظیم دقیق موفق به دقت در آماده‌سازی و آزمایش‌های دقیق بستگی دارد.

---

: در یادگیری ماشین، به ویژه در شبکه‌های عصبی عمیق، مشکل "نابودی یا انفجار گرادیان‌ها، یکی از چالش‌های مهم در طی فرآیند آموزش است. این مشکل زمانی به وجود می‌آید که در طول پس‌انتشار (backpropagation)، گرادیان‌ها به صورت نمایی کوچک یا بزرگ می‌شوند.<sup>۲۰</sup>

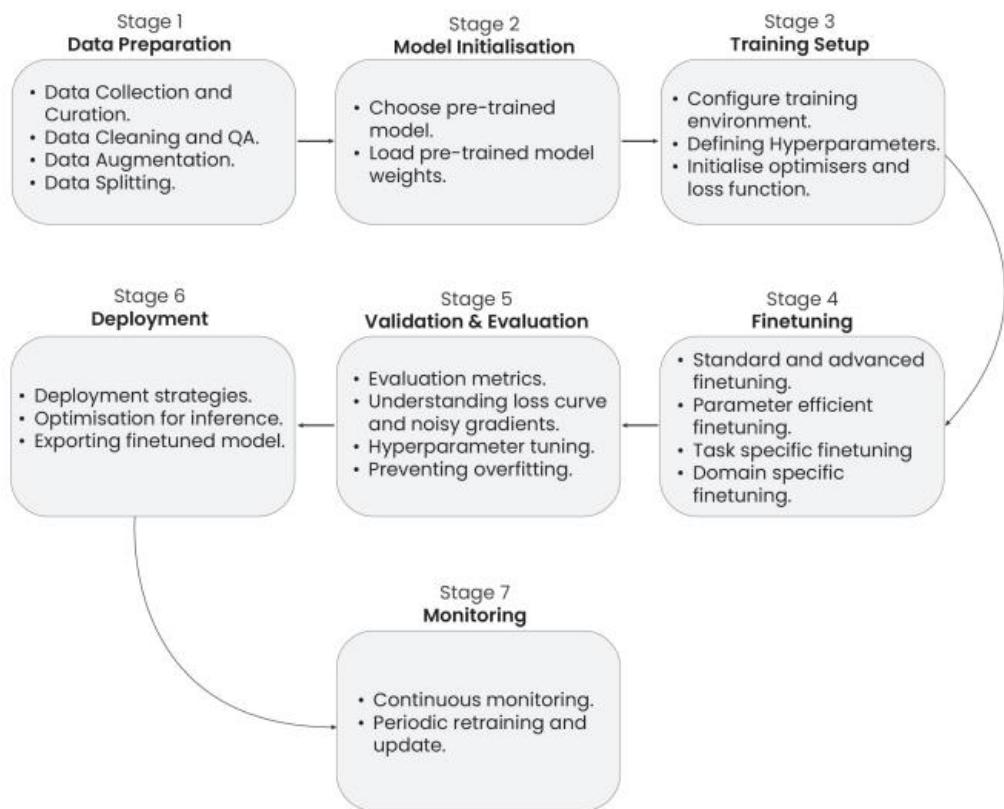
Vanishing Gradients: در این حالت، مقدار گرادیان‌ها در لایه‌های ابتدایی شبکه به‌طور نمایی کوچک می‌شود. به همین دلیل، وزن‌های لایه‌های اولیه بدروستی به روزسانی نمی‌شوند و مدل قادر نیست الگوهای پیچیده را به خوبی یاد بگیرد. این مشکل به خصوص در لایه‌های عمیق و توابع فعال‌سازی مانند سیگموید و تانژانت هایپربولیک (tanh) بیشتر رخ می‌دهد.

Exploding Gradients: در این حالت، مقدار گرادیان‌ها در هنگام پس‌انتشار به صورت نمایی افزایش می‌یابد و به مقادیر بسیار بزرگی می‌رسد. این موضوع باعث می‌شود که وزن‌ها بیش از حد تغییر کنند و مدل ناپایدار شود یا نتواند به درستی یاد بگیرد.

راهکارها برای این مشکل

- تغییر تابع فعال‌سازی: استفاده از توابعی مثل ReLU می‌تواند مشکل نابودی گرادیان را کاهش دهد.
- نرمال‌سازی گرادیان‌ها: مانند گرادیان کلیپینگ (Gradient Clipping) برای محدود کردن اندازه‌ی گرادیان‌ها در هنگام انفجار گرادیان.
- شبکه‌های با حافظه کوتاه- بلند: (LSTM) این شبکه‌ها به دلیل معماری خاص خود کمتر با مشکل نابودی گرادیان مواجه می‌شوند و برای داده‌های سری زمانی و توالی‌ها مؤثر هستند.

این مشکلات یکی از دلایلی هستند که بهبود شبکه‌های عصبی عمیق و استفاده از روش‌های نوین در طراحی مدل‌ها مورد توجه قرار گرفته است.



شکل ۲.۱: یک فرآیند جامع برای تنظیم دقیق مدل‌های زبان بزرگ (LLM) که هفت مرحله ضروری را نشان می‌دهد: آماده‌سازی داده‌ها، اولیه‌سازی مدل، تنظیم محیط آموزشی، تنظیم دقیق، ارزیابی و اعتبارسنجی، استقرار و ناظارت و نگهداری. هر مرحله نقش حیاتی در تطبیق مدل پیش‌آموزش دیده با وظایف خاص و تضمین عملکرد بهینه در طول چرخه عمر آن ایفا می‌کند.

## دو – چهار: مرحله چهارم: تنظیم دقیق جزئی یا کامل

این مرحله شامل بهروزرسانی پارامترهای LLM با استفاده از یک مجموعه داده خاص وظیفه است. تنظیم دقیق کامل، تمامی پارامترهای مدل را بهروزرسانی می‌کند و اطمینان حاصل می‌کند که تطبیق جامعی با وظیفه جدید صورت گیرد. به طور متناوب، تنظیم دقیق نیمه<sup>۲۱</sup> (HFT) یا تنظیم دقیق کارآمد از نظر پارامتر (PEFT)<sup>۲۲</sup> مانند استفاده از لایه‌های آدأپتر می‌تواند برای تنظیم دقیق جزئی مدل به کار رود. این روش لایه‌های اضافی را به مدل پیش‌آموزش دیده متصل می‌کند و امکان تنظیم دقیق کارآمد با تعداد کمتری پارامتر را فراهم می‌آورد، که می‌تواند به چالش‌های مربوط به کارایی محاسباتی، بیش‌فیتی و بهینه‌سازی بپردازد.

<sup>21</sup> Half fine-tuning

<sup>22</sup> Parameter-Efficient Fine-Tuning

## دو - پنجم: مرحله ارزیابی و اعتبارسنجی

ارزیابی و اعتبارسنجی شامل سنجش عملکرد LLM دقیق تنظیم شده بر روی داده‌های دیده نشده است تا اطمینان حاصل شود که مدل به خوبی تعمیم یافته و به اهداف مورد نظر می‌رسد. معیارهای ارزیابی، مانند انتروپی مقاطع<sup>۲۳</sup>، خطاهای پیش‌بینی را اندازه‌گیری می‌کنند، در حالی که اعتبارسنجی منحنی‌های زیان و سایر نشانگرهای عملکرد را برای شناسایی مشکلاتی مانند بیش‌برازش<sup>۲۴</sup> یا کم‌برازش<sup>۲۵</sup> زیر نظر دارد. این مرحله به راهنمایی تنظیم دقیق بیشتر کمک می‌کند تا عملکرد بهینه مدل حاصل شود.

## دو - ششم: مرحله ششم: استقرار

استقرار یک LLM به معنای عملیاتی کردن و در دسترس قرار دادن آن برای کاربردهای خاص است. این شامل پیکربندی مدل برای اجرای کارآمد بر روی سخت‌افزار یا پلتفرم‌های نرم‌افزاری مشخص است و اطمینان حاصل می‌کند که می‌تواند وظایفی مانند پردازش زبان طبیعی، تولید متن یا درک پرسش‌های کاربر را انجام دهد. استقرار همچنین شامل راهاندازی ادغام، تدبیر امنیتی و سیستم‌های نظارتی است تا عملکرد قابل اعتماد و امنی در کاربردهای دنیای واقعی تضمین شود.

---

cross-entropy<sup>۲۶</sup>: به عنوان یک معیار برای اندازه‌گیری اختلاف بین دو توزیع احتمالاتی استفاده می‌شود. این معیار، که غالباً به عنوان یک تابع خطای "loss function" استفاده می‌شود، به مدلی کمک می‌کند که دقت پیش‌بینی خود را بهبود بخشد. Cross-entropy اختلاف بین احتمال پیش‌بینی شده توسط مدل و توزیع واقعی داده‌ها را مشخص می‌کند.

دو نوع متداول cross-entropy در مسائل طبقه‌بندی مورد استفاده قرار می‌گیرد:

۱ - Binary Cross-Entropy: برای مسائل طبقه‌بندی دودویی (مثالاً پیش‌بینی دو کلاس مانند بله/خیر)، که تفاوت بین احتمالات پیش‌بینی شده برای هر کلاس و مقدار واقعی آن را محاسبه می‌کند.

۲ - Categorical Cross-Entropy: برای مسائل طبقه‌بندی چندکلاسی که شامل بیش از دو دسته‌بندی مجزا هستند و خروجی مدل معمولاً به صورت توزیع احتمالاتی برای هر کلاس نمایش داده می‌شود.

در عمل، هدف مدل بهینه‌سازی (minimize) این تابع خطای طریق روش‌هایی مانند گرادیان نزولی است تا تواند پیش‌بینی‌های دقیق‌تری انجام دهد. از این معیار در الگوریتم‌های چون شبکه‌های عصبی و رگرسیون لجستیک استفاده می‌شود، زیرا امکان محاسبه خطای پیش‌بینی و اصلاح مدل برای بهبود دقت آن را فراهم می‌کند.

Overfitting<sup>۲۷</sup>: زمانی رخ می‌دهد که مدل به طور افراطی به داده‌های آموزشی چسبیده و جزئیات و نویز آن‌ها را یاد می‌گیرد. این امر باعث می‌شود که مدل در داده‌های جدید عملکرد ضعیفی داشته باشد زیرا به جای یادگیری الگوهای کلی، جزئیات خاص داده‌های آموزشی را حفظ کرده است.

Underfitting<sup>۲۸</sup>: هنگامی اتفاق می‌افتد که مدل نتواند الگوهای اساسی داده‌ها را به خوبی بیاموزد و عملکرد ضعیفی روی داده‌های آموزشی و جدید داشته باشد.

## **دو – هفت: مرحله هفتم: نظارت و نگهداری**

نظارت و نگهداری یک LLM پس از استقرار برای اطمینان از عملکرد و قابلیت اطمینان مداوم بسیار مهم است. این شامل پیگیری مستمر عملکرد مدل، رسیدگی به هرگونه مشکل پیشآمده و به روزرسانی مدل به منظور تطبیق با داده‌های جدید یا نیازهای در حال تغییر است. نظارت و نگهداری مؤثر به حفظ دقت و کارایی مدل در طول زمان کمک می‌کند.

# فصل سوم

## مرحله اول : آماده‌سازی داده‌ها

### سه – یک: مراحل آماده‌سازی داده‌ها

#### سه – یک – یک: جمع‌آوری داده‌ها

اولین مرحله در آماده‌سازی داده‌ها، جمع‌آوری داده‌ها از منابع مختلف است. این منابع می‌توانند در هر فرمتی مانند CSV، صفحات وب، پایگاه‌های داده SQL، ذخیره‌سازی S3 و غیره باشند. پایتون چندین کتابخانه برای جمع‌آوری داده‌ها به صورت دقیق و کارآمد ارائه می‌دهد. جدول ۳.۱ مجموعه‌ای از فرمتهای رایج داده‌ها را همراه با کتابخانه‌های پایتون مورد استفاده برای جمع‌آوری داده‌ها نشان می‌دهد.

#### سه – یک – دو: پیش‌پردازش و قالب‌بندی داده‌ها

پیش‌پردازش و قالب‌بندی داده‌ها برای اطمینان از کیفیت بالای داده‌ها جهت تنظیم دقیق بسیار ضروری است. این مرحله شامل کارهایی مانند پاکسازی داده‌ها، مدیریت مقادیر گمشده و قالب‌بندی داده‌ها به گونه‌ای است که با نیازهای خاص وظیفه مطابقت داشته باشد. چندین کتابخانه برای پردازش داده‌های متنی وجود دارند و جدول ۳.۲ شامل برخی از کتابخانه‌های رایج پردازش داده‌ها در پایتون است.

#### سه – یک – سه: مدیریت عدم توازن داده‌ها

مدیریت مجموعه داده‌های نامتوازن برای اطمینان از عملکرد متوازن در تمامی کلاس‌ها بسیار حائز اهمیت است. چندین تکنیک و استراتژی برای این منظور به کار می‌روند:

۱. Under-sampling و Over-sampling: تکنیک‌هایی مانند<sup>26</sup> SMOTE (تکنیک افزایش مصنوعی اقلیت) نمونه‌های مصنوعی برای رسیدن به تعادل ایجاد می‌کنند.

کتابخانه پایتون: [Imbalanced-Learn<sup>27</sup>](#)

<sup>26</sup> Synthetic Minority Over Sampling Technique

<sup>27</sup> <https://imbalanced-learn.org/stable/references/index.html>

توضیح: کتابخانه imbalanced-learn روش‌های مختلفی برای مدیریت مجموعه داده‌های نامتوازن، از جمله تکنیک‌های oversampling مانند SMOTE ارائه می‌دهد.

۲. تنظیم تابع هزینه: تغییر تابع هزینه برای تخصیص وزن بیشتر به کلاس اقلیت، به طوری که وزن کلاس‌ها به صورت معکوس متناسب با فراوانی آن‌ها تنظیم شود.

۳. **Focal Loss**: نوعی از تابع هزینه انتروپی متقطع که فاکتوری برای کاهش وزن مثال‌های ساده و تمرکز بر نمونه‌های سخت به آن اضافه می‌کند.

کتابخانه پایتون: [Focal Loss<sup>28</sup>](#)

توضیح: پکیج focal loss پیاده‌سازی‌های قوی از توابع مختلف loss، از جمله SparseCategoricalFocalLoss و BinaryFocalLoss.

۴. یادگیری حساس به هزینه: ادغام هزینه خطاهای دسته‌بندی مستقیماً در الگوریتم یادگیری، با تخصیص هزینه بالاتر به خطاهای دسته‌بندی نمونه‌های کلاس اقلیت.

۵. روش‌های Ensemble: استفاده از تکنیک‌هایی مانند boosting و bagging برای ترکیب چندین مدل و مدیریت عدم توازن کلاس‌ها.

کتابخانه پایتون: [Sklearn.Ensemble<sup>29</sup>](#)

توضیح: کتابخانه scikit-learn پیاده‌سازی‌های قوی از روش‌های مختلف ensemble، از جمله boosting و bagging را فراهم می‌کند.

---

<sup>28</sup> <https://pypi.org/project/focal-loss/>

<sup>29</sup> <https://scikit-learn.org/stable/modules/ensemble.html>

فرمت داده	کتابخانه پایتون	توضیحات	لینک کتابخانه
فایلهای CSV	pandas	پانداس یک کتابخانه قدرتمند برای تحلیل و دستکاری داده است. این کتابخانه تابع <code>read_csv</code> را برای خواندن آسان و کارآمد فایلهای CSV به داخل اشیاء <code>DataFrame</code> ارائه می‌دهد. همچنین از خواندن داده‌ها به فرمتهای دیگر مانند JSON و غیره پشتیبانی می‌کند.	<a href="https://pandas.pydata.org/pandasdocs/stable/">https://pandas.pydata.org/pandasdocs/stable/</a>
صفحات وب	BeautifulSoup and requests	کتابخانه‌ای برای پردازش استاد HTML و XML است. وقتی با <code>requests</code> که برای ارسال درخواست‌های HTTP استفاده می‌شود، ترکیب شود، استخراج داده‌ها از صفحات وب، که برای انجام کارهای <code>web scraping</code> ضروری است، امکان‌پذیر می‌گردد.	<a href="https://www.crummy.com/software/BeautifulSoup/bs4/doc">https://www.crummy.com/software/BeautifulSoup/bs4/doc</a> <a href="https://requests.readthedocs.io/en/latest/">https://requests.readthedocs.io/en/latest/</a>
پایگاه‌های SQL داده	SQLAlchemy	یک ابزار SQL و کتابخانه ORM (نگاشت شیء به رابطه <sup>۳۰</sup> ) برای پایتون است که مجموعه‌ای کامل از الگوهای نگهداری داده‌های سازمانی ارائه می‌دهد.	<a href="https://www.sqlalchemy.org/">https://www.sqlalchemy.org/</a>
ذخیره‌سازی S3	boto3	آمازون وب سرویس‌ها (AWS) برای پایتون است که به توسعه‌دهندگان امکان استفاده از سرویس‌هایی مانند EC2 و Amazon S3 را می‌دهد. این کتابخانه امکان تعامل با خدمات AWS، از جمله بارگذاری، دانلود و مدیریت فایلهای ذخیره‌شده در S3 bucket را فراهم می‌کند.	<a href="https://boto3.amazonaws.com/v1/documentation/api/latest/index.html">https://boto3.amazonaws.com/v1/documentation/api/latest/index.html</a>
ادغام داده‌ها	RapidMiner	یک محیط جامع برای آماده‌سازی داده‌ها، یادگیری ماشین، و تحلیل پیش‌بینی‌کننده است که امکان پردازش و تبدیل داده‌های خام به اطلاعات قابل استفاده را به طور کارآمد فراهم می‌کند.	<a href="https://rapidminer.com">https://rapidminer.com</a>
پاکسازی داده‌ها	Trifacta Wrangler	بر ساده‌سازی و خودکارسازی فرآیندهای آماده‌سازی داده تمرکز دارد و داده‌های خام را به فرمتهای تمیز و ساختارمند تبدیل می‌کند.	<a href="https://www.alteryx.com/about-us/trifacta-is-now-alteryxdesigner-cloud">https://www.alteryx.com/about-us/trifacta-is-now-alteryxdesigner-cloud</a>

جدول ۳.۱: کتابخانه‌ها و ابزارهای پایتون برای جمع‌آوری و ادغام داده‌ها در فرمتهای مختلف، که نمای کلی از کتابخانه‌های پرکاربرد، عملکردهای آن‌ها و لینک‌های مستندات رسمی‌شان برای مدیریت و پردازش داده‌های کارآمد را ارائه می‌دهد.

<sup>30</sup> Object-Relational Mapping

۶. نمونه‌گیری طبقه‌بندی شده (**Stratified Sampling**): اطمینان از این که هر مینی‌بج در طول آموزش دارای نمایشی متناسب یا برابر از هر کلاس باشد.

### <sup>۳۱</sup> [sklearn.model\\_selection.StratifiedShuffleSplit](https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.StratifiedShuffleSplit.html):

توضیحات: **scikit-learn** ابزارهایی برای نمونه‌گیری طبقه‌بندی شده ارائه می‌دهد که تعادل در نمایش کلاس‌ها را تضمین می‌کند.

۷. پاکسازی داده‌ها (**Data Cleaning**): حذف داده‌های نویزی و اشتباه برچسب‌خورده که می‌تواند به طور نامتناسبی روی کلاس‌های اقلیت تاثیر بگذارد.

### <sup>۳۲</sup> [pandas.DataFrame.sample](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.sample.html):

توضیحات: **pandas** متدهایی برای نمونه‌گیری از DataFrame‌ها فراهم می‌کند که برای پاکسازی و پیش‌پردازش داده‌ها مفید هستند.

۸. استفاده از معیارهای مناسب: معیارهایی مانند Precision-Recall AUC، F1-اسکور، و کohen کاپا (Cohen's Kappa) اطلاعات بیشتری نسبت به دقت (accuracy) در مواجهه با داده‌های نامتوازن ارائه می‌دهند.

### <sup>۳۳</sup> [Sklearn.Metrics](https://scikit-learn.org/stable/modules/classes.html#sklearn-metrics-metrics):

توضیحات: **scikit-learn** مجموعه کاملی از ابزارها برای ارزیابی عملکرد مدل‌های طبقه‌بندی به ویژه در مواجهه با داده‌های نامتوازن ارائه می‌دهد.

نام کتابخانه	گزینه‌های پیش‌پردازش داده	لينك
spaCy	قابلیت‌های قدرمندی برای پیش‌پردازش متن ارائه می‌دهد، از جمله توکن‌سازی(tokenization)، لماتیزاسیون (lemmatization) و تشخیص کارآمد مرزهای جمله.	<a href="https://spacy.io">https://spacy.io</a>
NLTK	مجموعه‌های کامل از ابزارهای پیش‌پردازش داده ارائه می‌دهد، مانند توکن‌سازی، استمینگ (stemming) و حذف کلمات توقف.	<a href="https://www.nltk.org/">https://www.nltk.org/</a>
HuggingFace	هایگینگ فیس از طریق کتابخانه transformers امکانات گسترده‌ای برای پیش‌پردازش متن فراهم می‌کند، از جمله توکن‌سازی و پشتیبانی از مدل‌های پیش‌آموزش‌یافته مختلف.	<a href="https://huggingface.co/">https://huggingface.co/</a>
KNIME	پلتفرم KNIME Analytics امکان طراحی بصری جریان کاری برای ادغام داده‌ها، پیش‌پردازش و دستکاری‌های پیشرفته مانند استخراج متن و تحلیل تصویر را فراهم می‌کند.	<a href="https://www.knime.com/">https://www.knime.com/</a>

<sup>۳۱</sup> [https://scikit-learn.org/stable/modules/generated/sklearn.model\\_selection.StratifiedShuffleSplit.html](https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.StratifiedShuffleSplit.html)

<sup>۳۲</sup> <https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.sample.html>

<sup>۳۳</sup> [https://scikit-learn.org/stable/modules/model\\_evaluation.html](https://scikit-learn.org/stable/modules/model_evaluation.html)

جدول ۳.۲: مروری بر کتابخانه‌های پایتون که معمولاً برای پیش‌پردازش داده‌های متند استفاده می‌شوند، از جمله spaCy، NLTK و KNIME و HuggingFace. این جدول گزینه‌های خاص پیش‌پردازش که توسط هر کتابخانه ارائه می‌شوند را توضیح می‌دهد و پیوندهایی به مستندات رسمی آنها برای کاربرانی که به دنبال راهنمایی‌های بیشتر هستند، فراهم می‌کند.

## سه – یک – چهار: تقسیم مجموعه داده

تقسیم مجموعه داده برای تنظیم دقیق مدل شامل جدا کردن آن به مجموعه‌های آموزش و اعتبارسنجی است که معمولاً با نسبت ۸۰:۲۰ انجام می‌شود. تکنیک‌های مختلف شامل موارد زیر هستند:

۱. نمونه‌گیری تصادفی: انتخاب یک زیرمجموعه از داده‌ها به صورت تصادفی برای ایجاد یک نمونه نماینده.

کتابخانه پایتون: [sklearn.model\\_selection.train\\_test\\_split](#)<sup>۳۴</sup>

۲. نمونه‌گیری طبقه‌بندی شده: تقسیم مجموعه داده به زیر‌گروه‌ها و نمونه‌گیری از هر کدام برای حفظ تعادل کلاسی.

کتابخانه پایتون: [sklearn.model\\_selection.StratifiedShuffleSplit](#)<sup>۳۵</sup>

۳. اعتبارسنجی متقاطع K-بخشی: تقسیم مجموعه داده به K بخش و انجام آموزش و اعتبارسنجی به تعداد K بار.

کتابخانه پایتون: [sklearn.model\\_selection.KFold](#)<sup>۳۶</sup>

۴. اعتبارسنجی متقاطع Leave-One-Out: استفاده از یک نقطه داده به عنوان مجموعه اعتبارسنجی و بقیه برای آموزش، و این فرایند برای هر نقطه داده تکرار می‌شود.

کتابخانه پایتون: [sklearn.model\\_selection.LeaveOneOut](#)<sup>۳۷</sup>

جزئیات بیشتر را می‌توان در این مستندات <sup>۳۸</sup> درباره انتخاب مدل یافت.

## سه – دو: روش‌های تحقیق موجود و بالقوه

### سه – دو – یک: برچسب‌گذاری داده‌ها

برچسب‌گذاری داده شامل الصاق یا برچسب‌گذاری داده‌های متند با ویژگی‌های خاصی است که برای اهداف آموزشی مدل مهم هستند. این فرایند برای وظایف یادگیری تحت نظرارت بسیار مهم است و به طور قابل توجهی بر عملکرد مدل‌های تنظیم دقیق شده تأثیر می‌گذارد. تحقیقات اخیر به روش‌های مختلف برچسب‌گذاری داده‌ها اشاره کرده‌اند:

<sup>۳۴</sup> [https://scikit-learn.org/stable/modules/generated/sklearn.model\\_selection.train\\_test\\_split.html](https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html)

<sup>۳۵</sup> [https://scikit-learn.org/stable/modules/generated/sklearn.model\\_selection.StratifiedShuffleSplit.html](https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.StratifiedShuffleSplit.html)

<sup>۳۶</sup> [https://scikit-learn.org/stable/modules/generated/sklearn.model\\_selection.KFold.html](https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.KFold.html)

<sup>۳۷</sup> [https://scikit-learn.org/stable/modules/generated/sklearn.model\\_selection.LeaveOneOut.html](https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.LeaveOneOut.html)

<sup>۳۸</sup> [https://scikit-learn.org/stable/api/sklearn.model\\_selection.html](https://scikit-learn.org/stable/api/sklearn.model_selection.html)

- برچسب‌گذاری انسانی: برچسب‌گذاری دستی توسط کارشناسان انسانی به دلیل دقیق و درک زمینه‌ای همچنان استاندارد طلایی محسوب می‌شود. با این حال، برای مجموعه‌های داده بزرگ زمان‌بر و هزینه‌بر است. ابزارهایی مانند اکسل،<sup>39</sup> Prodigy<sup>40</sup> و Innodata<sup>41</sup> این فرایند را تسهیل می‌کنند.

- برچسب‌گذاری نیمه‌خودکار: ترکیب الگوریتم‌های یادگیری ماشین با بررسی انسانی برای ایجاد مجموعه‌های داده برچسب‌گذاری شده به صورت کارآمدتر. این روش تعادلی بین دقیق و کارایی ایجاد می‌کند. ابزارهایی مانند Snorkel<sup>42</sup> از نظرات ضعیف برای ایجاد برچسب‌های اولیه استفاده می‌کنند که سپس توسط برچسب‌گذاران انسانی بهبود می‌یابند.

- برچسب‌گذاری خودکار: برچسب‌گذاری کاملاً خودکار از الگوریتم‌های یادگیری ماشین برای برچسب‌گذاری داده‌ها بدون دخالت انسانی استفاده می‌کند و مقیاس‌پذیری و صرفه‌جویی در هزینه را ارائه می‌دهد. سرویس‌هایی مانند Amazon SageMaker Ground Truth<sup>43</sup> از یادگیری ماشین برای خودکارسازی برچسب‌گذاری داده‌ها استفاده می‌کنند، اگرچه دقیق ممکن است بسته به پیچیدگی وظیفه متفاوت باشد.

## سه—دو—دو: افزایش داده

تکنیک‌های افزایش داده (Data Augmentation) به طور مصنوعی مجموعه‌های داده آموزشی را گسترش می‌دهند تا به مسئله کمبود داده بپردازنند و عملکرد مدل را بهبود بخشنند. تکنیک‌های پیشرفته که معمولاً در پردازش زبان طبیعی (NLP) استفاده می‌شوند، شامل موارد زیر هستند:

- جایگزینی با تعبیه‌های کلمه: استفاده از تعبیه‌های کلمه مانند Word2Vec و GloVe برای جایگزینی کلمات با معادل‌های معنایی آنها و در نتیجه ایجاد نمونه‌های جدید داده.

- ترجمه معکوس: ترجمه متن به یک زبان دیگر و سپس ترجمه مجدد آن به زبان اصلی برای ایجاد داده‌های بازنویسی شده.<sup>44</sup> این تکنیک به تولید نمونه‌های آموزشی متنوع کمک می‌کند. ابزارهایی مانند Google Translate API<sup>44</sup> معمولاً برای این منظور استفاده می‌شوند.

- حملات متخصص: ایجاد داده‌های افزایش‌یافته از طریق مثال‌های متخصص که متن اصلی را کمی تغییر می‌دهند تا نمونه‌های جدیدی برای آموزش تولید شود، در حالی که معنای اصلی حفظ می‌شود. کتابخانه‌هایی مانند TextAttack<sup>45</sup> چارچوب‌هایی برای چنین افزایش‌هایی ارائه می‌دهند.

<sup>39</sup> <https://prodi.gy/>

<sup>40</sup> <https://innodata.com/>

<sup>41</sup> <https://snorkel.ai/>

<sup>42</sup> <https://aws.amazon.com/sagemaker/groundtruth/>

<sup>43</sup> paraphrased data

<sup>44</sup> <https://translate.google.com/?sl=auto&tl=en&op=translate>

<sup>45</sup> <https://github.com/QData/TextAttack>

- کتابخانه **NLP-AUG<sup>46</sup>**: این کتابخانه انواع مختلفی از افزونه‌ها را برای کاراکتر، کلمه، جمله، صدا و طیف‌نگار ارائه می‌دهد که تنوع مجموعه داده را افزایش می‌دهد.

#### سه - سه: تولید داده‌های مصنوعی با استفاده از LLM‌ها

مدل‌های زبانی بزرگ (LLM‌ها) می‌توانند داده‌های مصنوعی را از طریق تکنیک‌های نوآورانه تولید کنند، از جمله:

- مهندسی دستورها (**Prompt Engineering**): طراحی دستورهای خاص برای هدایت LLM‌ها مانند GPT-3 در تولید داده‌های مصنوعی مرتبط و با کیفیت بالا.

- تولید چندمرحله‌ای: استفاده از فرآیندهای تولید تکراری که در آن LLM‌ها داده‌های اولیه تولید می‌کنند که سپس در مراحل بعدی اصلاح می‌شوند. این روش می‌تواند داده‌های مصنوعی با کیفیت بالا برای وظایف مختلف، از جمله خلاصه‌سازی و تشخیص سوگیری تولید کند.

تأثیر دقیق و ارتباط داده‌های مصنوعی تولیدشده توسط LLM‌ها پیش از استفاده از آنها برای تنظیم دقیق مدل‌ها بسیار مهم است.

#### سه - سه: چالش‌های آماده‌سازی داده برای تنظیم دقیق LLM‌ها

چالش‌های کلیدی در آماده‌سازی داده شامل موارد زیر هستند:

۱. ارتباط حوزه‌ای: اطمینان از اینکه داده‌ها مرتبط با حوزه خاص هستند تا عملکرد دقیق مدل تضمین شود. داده‌های نامرتبط با حوزه می‌توانند منجر به تعمیم ضعیف و خروجی‌های نادرست شوند.

۲. تنوع داده‌ها: شامل کردن داده‌های متنوع و بهخوبی متعادل شده برای جلوگیری از سوگیری‌های مدل و بهبود تعمیم‌دهی. عدم تنوع می‌تواند باعث شود که مدل در مواجهه با سناریوهای کمتر نماینده ضعیف عمل کند.

۳. اندازه داده‌ها: مدیریت و پردازش مجموعه‌های داده بزرگ، به طوری که حداقل هزار نمونه برای تنظیم دقیق مؤثر توصیه می‌شود. با این حال، مجموعه‌های داده بزرگ چالش‌هایی از نظر ذخیره‌سازی، نیازهای محاسباتی و زمان پردازش ایجاد می‌کنند.

۴. پاکسازی و پیش‌پردازش داده‌ها: حذف نویز، خطاهای ناسازگاری‌ها برای فراهم کردن ورودی‌های تمیز برای مدل بسیار مهم است. داده‌های ضعیف پیش‌پردازش شده می‌توانند عملکرد مدل را به‌طور قابل توجهی کاهش دهند.

<sup>46</sup> <https://github.com/makcedward/nlpaug>

۵. برچسب‌گذاری داده‌ها: اطمینان از برچسب‌گذاری دقیق و سازگار برای وظایفی که به داده‌های برچسب‌گذاری شده نیاز دارند، ضروری است. برچسب‌گذاری ناهمانگ می‌تواند منجر به پیش‌بینی‌های نامطمئن مدل شود.

۶. مدیریت موارد نادر: نمایش مناسب موارد نادر اما مهم در مجموعه داده برای اطمینان از اینکه مدل می‌تواند به سناریوهای کمتر رایج اما حیاتی تعمیم بابد.

۷. ملاحظات اخلاقی: بررسی داده‌ها از نظر محتوای مضر یا سوگیرانه برای جلوگیری از پیامدهای ناخواسته بسیار مهم است. مدیریت اخلاقی داده‌ها شامل حذف سوگیری‌ها و اطمینان از حفظ حریم خصوصی است.

## سه – چهار: مجموعه‌های داده مناسب برای تنظیم دقیق LLM‌ها

برای فهرست جامعی از مجموعه داده‌های مناسب برای تنظیم دقیق LLM‌ها، به منابعی مانند [LLMXplorer](#)<sup>47</sup> که مجموعه‌های داده‌های خاص دامنه و وظیفه را ارائه می‌دهد، مراجعه کنید.

## سه – پنج: بهترین شیوه‌ها

### سه – پنج – یک: جمع‌آوری داده‌های باکیفیت

اطمینان از اینکه داده‌ها باکیفیت، متنوع و نماینده هستند، حیاتی است. استفاده از منابع باکیفیت و پوشش جامع سناریوهای مختلف، باعث افزایش مقاومت مدل می‌شود. ابزارهایی مانند DataRobot Paxata<sup>48</sup> و KNIME Analytics Platform<sup>49</sup> قابلیت‌های پروفایل‌سازی و تبدیل داده‌های قوی را ارائه می‌دهند.

### سه – پنج – دو: پیش‌پردازش مؤثر داده‌ها

پیش‌پردازش مناسب داده‌ها برای عملکرد مدل ضروری است. استفاده از کتابخانه‌هایی مانند NLTK، spaCy و Trifacta HuggingFace Transformers می‌تواند وظایف پیش‌پردازش را ساده‌تر کند. پلتفرم‌هایی مانند RapidMiner و Wrangler وظایف پاک‌سازی داده‌ها را به صورت خودکار انجام می‌دهند و به این ترتیب کارایی را بهبود بخشیده و ثبات را تضمین می‌کنند [۳۰].

### سه – پنج – سه: مدیریت عدم تعادل داده‌ها

<sup>47</sup> [https://docs.google.com/forms/d/e/1FAIpQLSfYVlyJNn5Afpb5YffFSBrYaXSBNwFn\\_8EBTSQO8qswhlusoxQ/viewform?usp=send\\_form](https://docs.google.com/forms/d/e/1FAIpQLSfYVlyJNn5Afpb5YffFSBrYaXSBNwFn_8EBTSQO8qswhlusoxQ/viewform?usp=send_form)

<sup>48</sup> <https://www.datarobot.com/platform/preparation/>

<sup>49</sup> <https://www.knime.com/>

پرداختن به عدم تعادل داده‌ها امری حیاتی است. تکنیک‌هایی مانند افزایش نمونه (over-sampling)، کاهش نمونه (under-sampling) و SMOTE به معادل‌سازی مجموعه‌های داده کمک می‌کنند. کتابخانه‌هایی مانند ensemble methods در scikit-learn و روش‌های جمعی imbalanced-learn ابزارهای قوی برای مدیریت مجموعه‌های داده نامتعادل فراهم می‌کنند [۳۱].

#### سه - پنج - چهار: افزایش و برچسب‌گذاری داده‌ها

افزایش داده‌ها و نشانه‌گذاری باعث بهبود استحکام مدل می‌شوند. ابزارهایی مانند TextAttack، NLP-AUG و Snorkel قابلیت‌های پیشرفته‌ای برای ایجاد مجموعه‌داده‌های متنوع و با برچسب‌گذاری مناسب ارائه می‌دهند [۳۲] .

#### سه - پنج - پنج: مدیریت اخلاقی داده‌ها

اطمینان از مدیریت اخلاقی داده‌ها شامل بررسی دقیق سوگیری‌ها و نگرانی‌های مربوط به حریم خصوصی است. پیاده‌سازی تکنیک‌های حفاظت از حریم خصوصی و فیلتر کردن محتوای مضر بسیار مهم است. سرویس‌هایی مانند Amazon SageMaker Ground Truth امکان برچسب‌گذاری داده‌ها به صورت مقیاس‌پذیر و ایمن را فراهم می‌کنند [۳۴].

#### سه - پنج - شش: ارزیابی و تکرار منظم

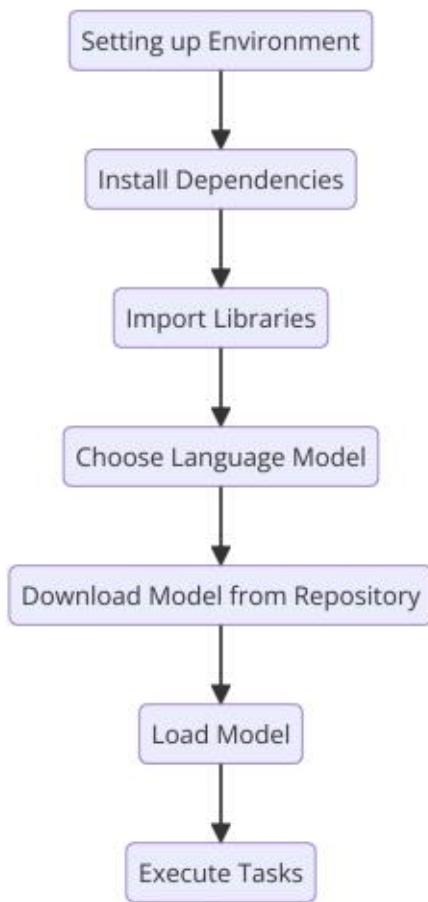
ارزیابی مداوم و تکرار در فرایند آماده‌سازی داده‌ها به حفظ کیفیت و ارتباط داده‌ها کمک می‌کند. استفاده از حلقه‌های بازخورد و معیارهای عملکرد تضمین می‌کند که بهبودهای پیوسته انجام شده و مدل با نیازهای داده‌های جدید سازگار شود.

با ادغام این بهترین شیوه‌ها، محققان و متخصصان می‌توانند اثربخشی تنظیم دقیق مدل‌های زبانی بزرگ (LLM) را افزایش دهنده عملکرد مدل را به صورت مقاوم و قابل اعتماد تضمین کنند.

# فصل چهارم

## مرحله دوم: راهاندازی مدل

### چهار - یک: مراحل مربوط به راهاندازی مدل



شکل ۴.۱: مراحل توالی‌دار مربوط به راهاندازی یک مدل زبان بزرگ (LLM)، که فرآیند را از تنظیم محیط تا اجرای وظایف نشان می‌دهد. هر مرحله برای اطمینان از اینکه LLM به درستی پیکربندی شده و آماده کار است، حیاتی است. این مراحل شامل نصب وابستگی‌های ضروری، وارد کردن کتابخانه‌ها، انتخاب و دانلود مدل زبان مناسب از یک مخزن، و در نهایت، بارگذاری مدل برای انجام وظایف خاص می‌باشد.

۱. تنظیم محیط: محیط خود را پیکربندی کنید، مانند تنظیم استفاده از GPU/TPU در صورت موجود بودن، که می‌تواند به طور قابل توجهی بارگذاری مدل و استنتاج را تسريع کند.
۲. نصب وابستگی‌ها: اطمینان حاصل کنید که تمام نرم‌افزارها و کتابخانه‌های لازم نصب شده‌اند. این معمولاً شامل پکیج منجر مانند pip و فریمورک‌هایی مانند TensorFlow یا PyTorch می‌باشد.
۳. وارد کردن کتابخانه‌ها: کتابخانه‌های مورد نیاز را در اسکریپت یا نوت‌بوک خود وارد کنید. کتابخانه‌های رایج شامل PyTorch برای torch، Hugging Face transforms و سایر کتابخانه‌های کاربردی هستند.
۴. انتخاب مدل زبان: مدل زبان پیش‌آموزش‌داده شده مناسب را بر اساس نیازهای وظیفه خود انتخاب کنید. این می‌تواند مدل‌هایی مانند GPT-3، BERT، یا دیگر مدل‌های موجود در پلتفرم‌هایی مانند Model Hub Hugging Face باشد.
۵. دانلود مدل از ریپازیتوری: از توابع فریمورک انتخاب شده برای دانلود مدل پیش‌آموزش‌داده شده از یک ریپازیتوری آنلاین استفاده کنید. به عنوان مثال، با استفاده از transformers، ممکن است از این استفاده کنید:

```
AutoModel.from_pretrained('model name').
```

۶. بارگذاری مدل در حافظه: مدل را در حافظه بارگذاری کنید تا آماده استنتاج یا تنظیم دقیق بیشتر باشد. این مرحله اطمینان حاصل می‌کند که وزن‌های مدل اولیه‌سازی شده و آماده استفاده هستند.
۷. اجرای تسك‌ها: وظایف مورد نظر را با استفاده از مدل بارگذاری شده انجام دهید. این می‌تواند شامل پیش‌بینی، تولید متن یا تنظیم دقیق مدل بر روی یک مجموعه داده جدید باشد.

## چهار – دو: ابزارها و کتابخانه‌ها برای راهاندازی مدل

پایتون مجموعه وسیعی از کتابخانه‌ها را برای راهاندازی مدل‌های زبان بزرگ ارائه می‌دهد و به مدل‌های منبع باز و بسته دسترسی دارد. در اینجا به برخی از کتابخانه‌های قابل توجه اشاره می‌شود:

### ۱ – کتابخانه پایتون: HuggingFace<sup>50</sup>

**توضیحات:** HuggingFace به خاطر حمایت از مدل‌های زبان بزرگ پیش‌آموزش‌داده شده متعدد معروف است، که از Phi-3 mini تا Llama-3 70B متغیر هستند. کتابخانه transformers که بخشی از HuggingFace است، به کاربران این امکان را می‌دهد که به این مدل‌ها از طریق کلاس‌هایی مانند AutoModelForCausalLM دسترسی پیدا کنند. این کتابخانه از بارگذاری مدل‌های

---

<sup>50</sup> <https://huggingface.co/docs/transformers/en/index>

تنظیم شده و همچنین مدل‌های کمی‌سازی شده<sup>۵۱</sup> بیتی پشتیبانی می‌کند. علاوه بر این، کتابخانه transformers شامل ویژگی pipeline است که استفاده از مدل‌های پیش‌آموزش‌داده شده برای وظایف مختلف را آسان می‌کند [۳۵].

## ۲- فریمورک پایتون: PyTorch<sup>۵۲</sup>

توضیحات: PyTorch ابزارها و کتابخانه‌های جامعی برای راهاندازی و تنظیم دقیق مدل‌های زبان بزرگ ارائه می‌دهد. این فریمورک یک پلتفرم انعطاف‌پذیر و کارآمد برای ساخت و استقرار مدل‌های یادگیری عمیق فراهم می‌کند. کتابخانه HuggingFace از transformers شکاف بین PyTorch و سایر فریمورک‌ها را پر می‌کند و قابلیت استفاده از آن را برای مدل‌های زبان پیشرفته بهبود می‌بخشد [۳۶].

## ۳- فریمورک پایتون: TensorFlow<sup>۵۳</sup>

توضیحات: TensorFlow نیز ابزارها و کتابخانه‌های وسیعی برای راهاندازی و تنظیم دقیق مدل‌های زبان بزرگ فراهم می‌کند. مشابه PyTorch، این فریمورک از کتابخانه HuggingFace بهره‌مند است که یک API و رابط کاربری چندمنظوره و کاربرپسند برای کار با آخرین پیشرفت‌ها در مدل‌های زبان بزرگ ارائه می‌دهد [۳۷].

---

<sup>۵۱</sup> کمی‌سازی (Quantization) فرآیندی است که طی آن وزن‌ها و بایاس‌های یک مدل با دقت کمتر (مانند اعداد صحیح به جای اعداد اعشاری با دقت بالا) نمایش داده می‌شوند. هدف از این فرآیند، کاهش حجم مدل و بهبود سرعت پردازش آن، به ویژه برای اجرا در دستگاه‌های با منابع محدود مانند موبایل و سیستم‌های نهفته است. به طور کلی، مدل‌های کمی‌سازی شده دقت کمتری دارند اما برای کاربردهای واقعی که نیازمند سرعت و کارایی بالا هستند، بسیار مناسب‌اند.

<sup>52</sup> <https://pytorch.org/docs/stable/index.html>

<sup>53</sup> <https://www.tensorflow.org/tutorials>

## چهار - سه: چالش‌ها در راه اندازی مدل

چالش	توضیحات
هم راستایی با وظیفه هدف	ضروری است که مدل پیش‌آموزش دیده با وظیفه یا دامنه خاص شما به طور نزدیکی هم راستا باشد. این هم راستایی اولیه به عنوان یک پایه محکم برای تلاش‌های بیشتر در زمینه تنظیم دقیق عمل می‌کند و منجر به بهبود کارایی و نتایج می‌شود [۳۸].
درک مدل پیش‌آموزش دیده	قبل از انتخاب، درک کامل معماری، قابلیتها، محدودیتها و وظایفی که مدل به طور اولیه روی آن‌ها آموزش دیده است، ضروری است. بدون این درک، تلاش‌های تنظیم دقیق ممکن است به نتایج دلخواه نرسد [۲۳].
در دسترس بودن و سازگاری	بررسی دقیق مستندات مدل، مجوز، نگهداری و فرکانس به روزرسانی برای جلوگیری از مشکلات احتمالی و اطمینان از ادغام روان آن در برنامه شما ضروری است.
معماری مدل	همه مدل‌ها در هر وظیفه‌ای برتر نیستند. هر معماری مدل نقاط قوت و ضعف خاص خود را دارد، بنابراین انتخاب یک مدل که با وظیفه خاص شما هم راستا باشد برای دستیابی به نتایج مطلوب ضروری است [۳۹].
محدودیت‌های منابع	بارگذاری مدل‌های پیش‌آموزش دیده LLM مبنابرا است و به محاسبات بیشتری نیاز دارد. این مدل‌ها به پردازنده‌های مرکزی و گرافیکی با عملکرد بالا و همچنین فضای دیسک قابل توجهی نیاز دارند. به عنوان مثال، مدل Llama 3 8B حداقل به ۱۶ گیگابایت حافظه برای بارگذاری و اجرای استنتاج نیاز دارد.
حریم خصوصی	حریم خصوصی و محروم‌گی عوامل حیاتی هنگام انتخاب یک مدل زبان بزرگ (LLM) هستند. بسیاری از کسب‌وکارها تمایل ندارند داده‌های خود را با ارائه‌دهندگان LLM خارجی به اشتراک بگذارند. در چنین مواردی، میزبانی یک LLM بر روی سوررهای محلی یا استفاده از LLM‌های پیش‌آموزش دیده در دسترس از طریق ارائه‌دهندگان ابر خصوصی می‌تواند راه حل‌های مناسبی باشد. این رویکردها اطمینان می‌دهند که داده‌ها در محل شرکت باقی می‌مانند و بنابراین حریم خصوصی و محروم‌گی حفظ می‌شود.
هزینه و نگهداری	میزبانی LLM‌ها بر روی سوررهای محلی به زمان و هزینه زیادی برای راه اندازی و نگهداری مداوم نیاز دارد. در مقابل، استفاده از ارائه‌دهندگان ابری نگرانی‌های مربوط به نگهداری منابع را کاهش می‌دهد، اما هزینه‌های صورتحساب ماهانه را به همراه دارد. این هزینه‌ها معمولاً بر اساس عواملی مانند اندازه مدل و حجم درخواست‌ها در دقیقه محاسبه می‌شوند.
اندازه مدل و کمی‌سازی	استفاده از یک مدل پیش‌آموزش دیده با مصرف حافظه بالا همچنان می‌تواند با استفاده از نسخه کمی‌شده آن ممکن باشد. از طریق کمی‌سازی، وزن‌های پیش‌آموزش دیده می‌توانند با دقت کمتری بارگذاری شوند، معمولاً ۴ بیتی یا ۸ بیتی، که حجم پارامترها را به طور قابل توجهی کاهش می‌دهد و در عین حال دقت قابل توجهی را حفظ می‌کند [۴۰].
داده‌های پیش‌آموزش	داده‌های مورد استفاده برای پیش‌آموزش را بررسی کنید تا درک مدل از زبان را بستجید. این موارد مهم هستند زیرا مدل‌هایی به طور خاص برای تولید کد در دسترس هستند و نمی‌خواهیم از آن مدل‌ها برای طبقه‌بندی متن مالی استفاده کنیم [۴۱].
آگاهی از سوگیری‌ها	نسبت به سوگیری‌های بالقوه در مدل‌های پیش‌آموزش دیده هوشیار باشید، بهویشه اگر پیش‌بینی‌های بدون تعصب لازم باشد. آگاهی از تعصب می‌تواند با آزمایش مدل‌های مختلف و بررسی داده‌های مورد استفاده برای پیش‌آموزش ارزیابی شود [۴۲].

جدول ۴.۱: مرور جامع چالش‌ها در راهاندازی یک مدل زبان بزرگ (LLM). این جدول موارد مهمی را بررسی می‌کند، از جمله اهمیت هم‌راستایی مدل‌های پیش‌آموزش‌دیده با وظایف خاص، درک معماری مدل و سازگاری آن، مدیریت محدودیت‌های منابع و اطمینان از حریم خصوصی داده‌ها. علاوه بر این، چالش‌های مرتبط با هزینه، نگهداری و پیچیدگی‌های اندازه مدل، کمی‌سازی و آگاهی از تعصبات را نیز مورد بحث قرار می‌دهد. هر چالش به منابع خاصی مرتبط است تا درک کامل و پیاده‌سازی صحیح مدل‌ها را تضمین کند.

## چهار – چهار: آموزش‌ها

- ۱ - خلاصه‌سازی با استفاده از Llama 3<sup>۵۴</sup>
- ۲ - آموزش HuggingFace برای شروع کار با LLM‌ها<sup>۵۵</sup>
- ۳ - آموزش PyTorch برای تنظیم دقیق مدل‌ها<sup>۵۶</sup>
- ۴ - آموزش TensorFlow برای مدل‌های ترنسفورمر<sup>۵۷</sup>

<sup>۵۴</sup> <https://medium.com/@manuelescobar-dev/implementing-and-running-llama-3-with-hugging-faces-transformers-library-40e9754d8c80>

<sup>۵۵</sup> [https://huggingface.co/docs/transformers/en/llm\\_tutorial](https://huggingface.co/docs/transformers/en/llm_tutorial)

<sup>۵۶</sup> [https://pytorch.org/tutorials/beginner/finetuning\\_torchvision\\_models\\_tutorial.html](https://pytorch.org/tutorials/beginner/finetuning_torchvision_models_tutorial.html)

<sup>۵۷</sup> <https://www.tensorflow.org/text/tutorials/transformer>

## فصل پنجم

### مرحله سوم: تنظیمات آموزش

#### پنج - یک: مراحل مرتبط با تنظیمات آموزش

۱. راهاندازی محیط آموزشی: هنگام راهاندازی محیط برای آموزش یک مدل زبان بزرگ (LLM)، ضروری است که سختافزار با عملکرد بالا، مانند GPU‌ها یا TPU‌ها را پیکربندی کنید و نصب صحیح اجزای نرمافزاری مورد نیاز مانند TensorFlow، PyTorch و فریمورک‌های یادگیری عمیق مانند CuDNN، CUDA یا CuTorch را تضمین کنید. تأیید شناسایی سختافزار و سازگاری آن با نرمافزار برای بهره‌برداری مؤثر از قدرت محاسباتی، کاهش زمان آموزش و بهبود عملکرد مدل اهمیت دارد.

۲. تعریف هایپرپارامترها: هنگام تعریف هایپرپارامترها برای تنظیم دقیق یک LLM، مهم است که پارامترهای کلیدی مانند نرخ یادگیری، اندازه دسته و تعداد دوره‌ها را با دقت تنظیم کنید تا عملکرد مدل بهینه شود.

۳. راهاندازی بهینه‌سازها و توابع ضرر: هنگام راهاندازی بهینه‌سازها و توابع ضرر برای تنظیم دقیق یک LLM، انتخاب بهینه‌ساز مناسب برای بهروزرسانی مؤثر وزن‌های مدل و انتخاب تابع ضرر صحیح برای اندازه‌گیری عملکرد مدل بسیار حیاتی است [۴۳].

#### پنج - دو: راهاندازی محیط آموزشی

هنگام تنظیم دقیق یک مدل زبان بزرگ (LLM)، محیط محاسباتی نقش حیاتی در تضمین آموزش مؤثر ایفا می‌کند. برای دستیابی به عملکرد بهینه، ضروری است که محیط را با سختافزار با عملکرد بالا، مانند GPU‌ها (واحدهای پردازش گرافیکی) یا TPU‌ها (واحدهای پردازش تنسور) پیکربندی کنید. NVIDIA GPU‌هایی مانند A100 یا V100 به دلیل قابلیت‌های پردازش موازی خود به طور گسترده‌ای برای آموزش مدل‌های یادگیری

عمیق استفاده می‌شوند. برای عملیات در مقیاس بزرگ‌تر، TPUهایی که توسط Google Cloud ارائه می‌شوند می‌توانند شتاب بیشتری فراهم کنند [۴۴].

نخست، اطمینان حاصل کنید که سیستم یا محیط ابری شما سختافزار لازم را نصب کرده است. برای GPUها، این شامل راهاندازی CUDA<sup>58</sup> (معماری دستگاه محاسباتی یکپارچه) و CUDNN<sup>59</sup> (کتابخانه شبکه‌های عصبی عمیق NVIDIA) از CUDA است که برای فعال‌سازی شتابدهی GPU ضروری است. برای استفاده از TPU، معمولاً یک محیط Google Cloud با نمونه‌های TPU راهاندازی می‌کنید که شامل پیکربندی زمان‌اجرای TPU در اسکریپت‌های آموزشی شما می‌باشد.

مطمئن شوید که سختافزار به درستی شناسایی و توسط فریمورک‌های یادگیری عمیق شما مورد استفاده قرار می‌گیرد. به عنوان مثال، در PyTorch، می‌توانید با استفاده از (torch.cuda.is\_available) قابلیت دسترسی GPU را بررسی کنید. راهاندازی و آزمایش صحیح سختافزار اطمینان می‌دهد که فرآیند آموزشی می‌تواند به طور مؤثر از قدرت محاسباتی بهره‌برداری کند و زمان آموزش را کاهش داده و عملکرد مدل را بهبود بخشد [۳۶].

هنگام تنظیم دقیق یک LLM، ملاحظات نرم‌افزاری و سختافزاری از اهمیت بالایی برخوردار است تا فرآیند آموزشی به طور روان و مؤثر انجام شود. از طرف نرم‌افزار، به یک فریمورک یادگیری عمیق سازگار مانند PyTorch یا TensorFlow نیاز دارید. این فریمورک‌ها از LLM‌ها به خوبی پشتیبانی می‌کنند و ابزارهایی برای آموزش و ارزیابی مؤثر مدل‌ها فراهم می‌کنند. نصب آخرین نسخه‌های این فریمورک‌ها، همراه با هرگونه وابستگی‌های لازم، برای بهره‌برداری از جدیدترین ویژگی‌ها و بهبودهای عملکرد ضروری است [۴۵].

علاوه بر این، استفاده از کتابخانه‌هایی مانند transformers از Hugging Face برای ساده‌سازی فرآیند بارگذاری مدل‌های از پیش آموزش دیده و توکنایزرهای توصیه می‌شود. این کتابخانه به ویژه برای کار با LLM‌های مختلف مناسب است و یک رابط کاربری کاربرپسند برای تنظیم دقیق مدل فراهم می‌کند. اطمینان حاصل کنید که تمام اجزای نرم‌افزاری، از جمله کتابخانه‌ها و وابستگی‌ها، با فریمورک و تنظیمات سختافزاری انتخابی شما سازگار هستند [۳۵].

از طرف سختافزار، به نیازهای حافظه مدل و مجموعه داده‌های خود توجه کنید. LLM‌ها معمولاً به حافظه GPU قابل توجهی نیاز دارند، بنابراین انتخاب GPUهایی با VRAM بالاتر (مانند ۱۶ GB یا بیشتر) می‌تواند مفید باشد. اگر مدل شما به طور استثنایی بزرگ باشد یا اگر با مجموعه داده‌های بسیار بزرگی آموزش می‌دهید، ممکن است نیاز به آموزش توزیع شده در چندین GPU یا TPU داشته باشید. این امر نیازمند تنظیم دقیق تکنیک‌های همزمانی داده‌ها یا همزمانی مدل است تا به طور مؤثر از سختافزار موجود استفاده شود [۴۶].

در نهایت، اطمینان حاصل کنید که برای سختافزار خود، سیستم خنک‌کننده و منبع تغذیه قوی فراهم شده باشد، زیرا آموزش LLM‌ها می‌تواند نیازمند به منابع زیاد باشد و گرمای قابل توجهی تولید کند و نیاز به تأمین

<sup>58</sup> <https://developer.nvidia.com/cuda-toolkit>

<sup>59</sup> <https://developer.nvidia.com/cudnn>

مداوم برق دارد. تنظیم صحیح سخت‌افزار نه تنها عملکرد آموزشی را بهبود می‌بخشد بلکه عمر تجهیزات شما را نیز افزایش می‌دهد [۴۷].

## پنج – سه: تعریف هایپرپارامترها

هایپرپارامترهای کلیدی مانند نرخ یادگیری، اندازه بج، و ایپاک‌ها<sup>۶۰</sup> برای بهبود عملکرد مدل و دستیابی به نتایج بهتر بسیار حیاتی هستند. این فرآیند شامل تنظیم هایپرپارامترها و تنظیمات آموزشی به منظور هم‌راستا کردن آن‌ها با مورد استفاده خاص شما است. در زیر هایپرپارامترهای کلیدی آورده شده‌اند:

۱. **نرخ یادگیری**: تنظیم یک LLM شامل استفاده از الگوریتم‌های بهینه‌سازی مانند گرادیان کاهشی تصادفی<sup>۶۱</sup> (SGD) است. این تکنیک با استفاده از نمونه‌هایی از مجموعه داده‌های آموزشی، گرادیان خط را برای وضعیت فعلی مدل تخمین می‌زند و سپس وزن‌های مدل را از طریق الگوریتم بازگشت خطابه‌روزرسانی می‌کند. نرخ یادگیری سرعتی را که مدل به مسئله سازگار می‌شود تعیین می‌کند. نرخ‌های یادگیری کوچکتر به آموزش بیشتری نیاز دارند زیرا تغییرات وزن در هر بهروزرسانی حداقل است، در حالی که نرخ‌های یادگیری بزرگ‌تر منجر به تغییرات سریع‌تری در وزن‌ها می‌شوند [۴۸].

۲. **اندازه بج**: یک بج به زیرمجموعه‌ای از داده‌های آموزشی اطلاق می‌شود که برای بهروزرسانی وزن‌های مدل در طول فرآیند آموزش استفاده می‌شود. آموزش بج شامل تقسیم کل مجموعه آموزشی به گروه‌های کوچک‌تر است و پس از پردازش هر بج، مدل بهروزرسانی می‌شود. اندازه بج یک هایپرپارامتر است که تعداد نمونه‌هایی را که قبل از بهروزرسانی پارامترهای مدل پردازش می‌شود، تعیین می‌کند.

۳. **ایپاک**: ایپاک به یک دور کامل از کل مجموعه داده‌های آموزشی اطلاق می‌شود. این شامل یک دور کامل از عبور به جلو و به عقب از طریق مجموعه داده است. مجموعه داده می‌تواند به عنوان یک بج واحد پردازش شود یا به چندین بج کوچک‌تر تقسیم گردد. یک ایپاک کامل محسوب می‌شود هنگامی که مدل تمام بج‌ها را پردازش کرده و پارامترهای خود را بر اساس ضرر محاسبه‌شده بهروزرسانی کرده باشد.

## پنج – سه – یک: روش‌های تنظیم هایپرپارامتر

تنظیم هایپرپارامترهای LLM شامل تنظیم انواع مختلف هایپرپارامترها در طول فرآیند آموزش برای شناسایی ترکیب بهینه است که بهترین خروجی را تولید می‌کند. این فرآیند معمولاً شامل تلاش و خطای قابل توجهی است

<sup>۶۰</sup> در یادگیری ماشین و شبکه‌های عصبی، epochs (دوره‌ها یا ایپاک‌ها) به تعداد دفعاتی گفته می‌شود که الگوریتم آموزش یک مدل، کل داده‌های آموزشی را از ابتدا تا انتهای پردازش می‌کند. هر بار که مدل یک بار کامل روی کل داده‌های آموزشی تمرین می‌شود، یک ایپاک به پایان می‌رسد. معمولاً با افزایش تعداد ایپاک‌ها، مدل دقیق‌تر پیدا می‌کند، چون فرصت بیشتری دارد تا الگوهای موجود در داده را یاد بگیرد. با این حال، تعداد بسیار زیاد ایپاک‌ها ممکن است باعث overfitting (بیش‌پرازش) شود، که به معنای یادگیری بیش از حد جزئیات داده‌های آموزشی است و ممکن است در مواجهه با داده‌های جدید کارایی خوبی نداشته باشد.

<sup>۶۱</sup> stochastic gradient descent

که در آن هر تنظیم هایپرپارامتر به دقت ردیابی می شود و عملکرد حاصل ثبت می شود. انجام این کار به صورت دستی می تواند زمان بر باشد. برای رفع این مشکل، روش های تنظیم هایپرپارامتر خودکار توسعه یافته اند تا فرآیند را تسهیل کنند. سه روش رایج تنظیم هایپرپارامتر خودکار عبارتند از: جستجوی تصادفی، جستجوی شبکه ای، و بهینه سازی بیزی<sup>۶۲</sup>:

**۱. جستجوی تصادفی:** این روش به طور تصادفی ترکیب های هایپرپارامتر را از یک دامنه مشخص انتخاب و ارزیابی می کند. این یک رویکرد ساده و کارآمد است که می تواند فضای پارامتر بزرگی را کاوش کند. با این حال، ممکن است همیشه بهترین ترکیب هایپرپارامترها را پیدا نکند و می تواند از نظر محاسباتی پرهزینه باشد [۴۹].

**۲. جستجوی شبکه ای:** بر خلاف جستجوی تصادفی، جستجوی شبکه ای به طور جامع هر ترکیب ممکن از هایپرپارامترها را از یک دامنه مشخص ارزیابی می کند. اگرچه این رویکرد از نظر منابع سنگین است، اما اطمینان می دهد که بهترین مجموعه هایپرپارامترها پیدا شود [۵۰].

**۳. بهینه سازی بیزی:** این روش از یک مدل احتمالی برای پیش بینی عملکرد هایپرپارامترهای مختلف استفاده می کند و بهترین ها را انتخاب می کند. این یک روش کارآمد است که می تواند فضای پارامتر بزرگی را بهتر مدیریت کند و از نظر منابع نسبت به جستجوی شبکه ای کمتر هزینه بر است. با این حال، راه اندازی آن پیچیده تر است و ممکن است در شناسایی مجموعه بهینه هایپرپارامترها نسبت به جستجوی شبکه ای کمتر قابل اعتماد باشد.

**۴. تنظیم هایپرپارامتر خودکار:** این امکان را فراهم می آورد که مدل های زبانی متعددی با هر ترکیب منحصر به فردی از هایپرپارامترها توسعه یابند. با آموزش این مدل ها بر روی یک مجموعه داده مشابه، امکان مقایسه خروجی های آن ها و تعیین اینکه کدام پیکربندی برای مورد استفاده مورد نظر بهترین است، فراهم می شود. علاوه بر این، مدل هایی که با مجموعه های مختلفی از هایپرپارامترها تنظیم شده اند، می توانند به کاربردهای خاص مختلفی اختصاص داده شوند.

## پنج - چهار: راه اندازی بهینه سازها و توابع ضرر

انتخاب بهینه ساز و تابع ضرر مناسب برای آموزش و تنظیم LLMها حیاتی است. در زیر توضیحاتی درباره برخی از الگوریتم های بهینه سازی رایج، مزایا، معایب و موارد استفاده مناسب آن ها آورده شده است:

<sup>62</sup> Bayesian optimisation

### پنج - چهار - یک: گرادیان کاهشی (Gradient Descent)

گرادیان کاهشی یک الگوریتم بهینه‌سازی بنیادی است که برای حداقل کردن تابع هزینه در مدل‌های یادگیری ماشین استفاده می‌شود. هدف آن یافتن پارامترهای بهینه برای یک شبکه عصبی است.

نحوه کار: گرادیان کاهشی به صورت تکراری پارامترهای مدل را در جهت گرادیان منفی تابع هزینه به روزرسانی می‌کند. این الگوریتم گرادیان‌ها را برای هر پارامتر محاسبه کرده و به روزرسانی‌ها را در تمام نقاط داده اعمال می‌کند تا به همگرایی برسد. این روش از کل مجموعه داده برای محاسبه گرادیان‌ها استفاده می‌کند و معمولاً به نرخ یادگیری ثابتی نیاز دارد و به مقیاس داده و انتخاب نرخ یادگیری حساس است.

**مزایا:**

- ساده و آسان برای پیاده‌سازی.
- شهودی و آسان برای درک.
- به حداقل کلی برای تابع محدب همگرا می‌شود.<sup>۶۳</sup>
- مناسب برای مشکلات کوچک.

**معایب:**

- از نظر محاسباتی پرهزینه در مجموعه‌های داده بزرگ.
- ممکن است در مینیموم‌های لوکال گیر کند.
- به تعداد زیادی از تکرارها نیاز دارد.
- به انتخاب نرخ یادگیری حساس است.

**زمان استفاده:** گرادیان کاهشی بهترین گزینه برای مجموعه‌های داده کوچک است که در آن هزینه محاسبه گرادیان کم است و سادگی و وضوح مدنظر است.

### پنج - چهار - دو: گرادیان کاهشی تصادفی (SGD)

گرادیان کاهشی تصادفی (Stochastic Gradient Descent) نوعی از گرادیان کاهشی است که بر کاهش محاسبات در هر تکرار تمرکز دارد.

نحوه کار: SGD پارامترها را با استفاده از یک یا چند نقطه داده در هر تکرار به روزرسانی می‌کند و در به روزرسانی‌ها تصادفی بودن را معرفی می‌کند. این روش بار محاسباتی در هر تکرار را کاهش می‌دهد و معمولاً سریع‌تر از گرادیان کاهشی بچی همگرا می‌شود. با این حال، به دلیل واریانس بالاتر به نرخ یادگیری کمتری نیاز دارد و از مومنتوم برای پایدار کردن به روزرسانی‌ها بهره‌مند می‌شود.

**مزایا:**

- سریع و مناسب برای مجموعه‌های داده بزرگ.

<sup>۶۳</sup> به عبارت دیگر یعنی یک روش یا الگوریتم می‌تواند برای تابع محدب، به نقطه‌ای برسد که مقدار آن حداقل مقدار ممکن در کل دامنه تابع باشد.

- استفاده کارآمد از حافظه.
- ساده و آسان برای پیاده‌سازی.
- می‌تواند به دلیل نویز از مینیمومهای محلی فرار کند.

**معایب:**

- واریانس بالا در بهروزرسانی‌ها می‌تواند منجر به ناپایداری شود.
- ممکن است حداقل را بیش از حد رد کند.
- به انتخاب نرخ یادگیری حساس است.
- ممکن است نسبت به روش‌های بچی کندر همگرا شود.

**زمان استفاده:** SGD برای مجموعه‌های داده بزرگ، سناریوهای یادگیری تدریجی و محیط‌های یادگیری زمان واقعی که منابع محاسباتی محدود است، ایده‌آل است.

#### **پنج - چهار - سه: گرادیان کاهشی مینی بج**

گرادیان کاهشی مینی بج، کارایی SGD و ثبات گرادیان کاهشی بج را ترکیب کرده و تعادلی بین رویکردهای بج و تصادفی ارائه می‌دهد.

**نحوه کار:** این روش داده‌ها را به دسته‌های کوچک تقسیم می‌کند و پارامترها را با استفاده از گرادیان‌های میانگین شده بر روی هر مینی بج بهروزرسانی می‌کند. این کار نسبت به SGD و کارایی بیشتری نسبت به گرادیان کاهشی بج دارد و به عمومیت بخشیدن به بهروزرسانی‌ها کمک می‌کند.

**مزایا:**

- تعادل بین کارایی و ثبات برقرار می‌کند.
- بهروزرسانی‌های قابل عمومیت بیشتری ارائه می‌دهد.
- واریانس بهروزرسانی‌های پارامتر را کاهش می‌دهد.
- تعادلی بین SGD و بج ایجاد می‌کند.

**معایب:**

- نیاز به تنظیم اندازه مینی بج دارد.
  - برای مجموعه‌های داده بسیار بزرگ می‌تواند هنوز هم پرهزینه از نظر محاسباتی باشد.
  - پیاده‌سازی پیچیده‌تری دارد.
  - ممکن است به تکرارهای بیشتری نسبت به گرادیان کاهشی بچی نیاز داشته باشد.
- زمان استفاده:** گرادیان کاهشی مینی بج برای اکثر وظایف یادگیری عمیق مناسب است، بهویژه هنگام کار با مجموعه‌های داده متوسط تا بزرگ.

## پنج - چهار - چهار: آدآگراد<sup>۶۴</sup>

الگوریتم گرادیان تطبیقی یا آدآگراد برای داده‌های پراکنده و مدل‌های با ابعاد بالا طراحی شده و نرخ‌های یادگیری را تنظیم می‌کند تا عملکرد را بر روی داده‌های پراکنده بهبود بخشد.

نحوه کار: آدآگراد نرخ یادگیری را برای هر پارامتر بر اساس اطلاعات گرادیان تاریخی<sup>۶۵</sup> تنظیم می‌کند و گرادیان‌های مربع شده<sup>۶۶</sup> را جمع‌آوری می‌کند. این روش از بهروزرسانی‌های بزرگ برای پارامترهای پر تکرار جلوگیری کرده و در مواجهه با ویژگی‌های پراکنده کمک می‌کند.

مزایا:

- نرخ یادگیری را برای هر پارامتر تطبیق می‌دهد.
- مناسب برای داده‌های پراکنده.
- نیازی به تنظیم دستی نرخ‌های یادگیری نیست.
- با داده‌های با ابعاد بالا عملکرد خوبی دارد.

معایب:

- نرخ یادگیری می‌تواند به صفر برسد و یادگیری را متوقف کند.
- ممکن است به تنظیم بیشتری برای همگرایی نیاز داشته باشد.
- جمع‌آوری گرادیان‌های مربع شده می‌تواند منجر به نرخ‌های یادگیری بیش از حد کوچک شود.
- می‌تواند به طور قابل توجهی کند شود.

زمان استفاده: آدآگراد برای مجموعه‌های داده پراکنده مانند متن و تصاویر مفید است که در آن نرخ‌های یادگیری باید به فراوانی ویژگی‌ها تطبیق یابند.

## ۴ - ۵ - انتشار میانگین مربع (RMSprop<sup>۶۷</sup>)

رایج‌ترین انتشار میانگین مربع یک روش نرخ یادگیری تطبیقی است که برای عملکرد بهتر در مشکلات غیر ایستایی و آنلاین طراحی شده است.

نحوه کار: RMSprop آدآگراد را با استفاده از میانگین متحرک گرادیان‌های مربع شده تغییر می‌دهد تا نرخ‌های یادگیری را بر اساس بزرگی‌های گرادیان‌های اخیر تنظیم کند. این روش میانگین جاری از گرادیان‌های مربع شده را حفظ می‌کند تا به حفظ نرخ‌های یادگیری ثابت کمک کند.

مزایا:

- مشکل کاهش نرخ یادگیری آدآگراد را حل می‌کند.

<sup>64</sup> Adaptive Gradient Algorithm (AdaGrad)

<sup>65</sup> historical gradient

<sup>66</sup> squared gradients

<sup>67</sup> Root Mean Square Propagation

- نرخ یادگیری را بر اساس گرادیان‌های اخیر تطبیق می‌دهد.
- برای شبکه‌های عصبی بازگشتی مؤثر است.
- در برابر اهداف غیر ایستایی مقاوم‌تر است.

**معایب:**

- ممکن است هنوز در مینیموم‌های محلی<sup>68</sup> در مشکلات غیر محدب<sup>69</sup> گیر کند.
- نیاز به تنظیم ابرپارامترها دارد.
- نیاز به تنظیم دقیق نرخ کاهش دارد.
- ممکن است به نرخ یادگیری اولیه حساس باشد.

**زمان استفاده:** بهترین گزینه برای مشکلات بهینه‌سازی غیر محدب، آموزش RNN‌ها و LSTM‌ها و مواجهه با اهداف نویزی یا غیر ایستایی است.

### پنج - چهار - شش: آدا دلتا

آدا دلتا<sup>70</sup> بهبود یافته‌ای بر آدأگراد و RMSprop است که بر نرخ‌های یادگیری تطبیقی تمرکز می‌کند بدون اینکه خیلی سریع کاهش یابد.

**نحوه کار:** آدا دلتا نیاز به تعیین نرخ یادگیری پیش‌فرض را با استفاده از یک پنجره متحرک از بهروزرسانی‌های گرادیان، از بین می‌برد. این روش نرخ‌های یادگیری را بر اساس بزرگی‌های گرادیان‌های اخیر تنظیم می‌کند تا اطمینان حاصل شود که بهروزرسانی‌ها حتی با گرادیان‌های پراکنده نیز ثابت باقی بمانند.

**مزایا:**

- نیاز به تعیین نرخ یادگیری پیش‌فرض را از بین می‌برد.
- به مشکل کاهش نرخ یادگیری رسیدگی می‌کند.
- نیازی به تنظیم دستی نرخ یادگیری ندارد.
- به خوبی با پراکنده‌گی گرادیان‌ها برخورد می‌کند.

**معایب:**

- پیچیده‌تر از RMSprop و آدأگراد است.
- ممکن است در ابتدا همگرایی کمتری داشته باشد.
- ممکن است به تکرارهای بیشتری برای همگرایی نیاز داشته باشد.
- پیاده‌سازی می‌تواند پیچیده‌تر باشد.

---

<sup>68</sup> local minima

<sup>69</sup> non-convex problems

<sup>70</sup> Adaptive Delta

**زمان استفاده:** آدا دلتا برای سناریوهای مشابه RMSprop مناسب است اما در موقعی که از تنظیم دستی نرخ یادگیری اجتناب می‌شود، ترجیح داده می‌شود.

#### ۴-۷-۵ آدام

تخمین لحظه‌ای تطبیقی<sup>۷۱</sup> (آدام) مزایای آدادر و RMSprop را ترکیب می‌کند و برای مشکلات با مجموعه‌های داده بزرگ و فضاهای با ابعاد بالا مناسب است.

**نحوه کار:** آدام از میانگین‌های متحرک هر دو گرادیان و مقادیر مربع شده آنها برای محاسبه نرخ‌های یادگیری تطبیقی برای هر پارامتر استفاده می‌کند. این روش شامل اصلاح بایاس است و معمولاً همگرایی سریعتری نسبت به سایر روش‌ها را به دست می‌آورد.

**مزایا:**

- مزایای آدادر و RMSprop را ترکیب می‌کند.
- نرخ‌های یادگیری تطبیقی.
- شامل اصلاح بایاس است.
- همگرایی سریع.
- با مجموعه‌های داده بزرگ و فضاهای با ابعاد بالا به خوبی کار می‌کند.

**معایب:**

- نیاز به تنظیم ابرپارامترها دارد (اگرچه معمولاً با پیش‌فرضها خوب عمل می‌کند).
- نیاز به محاسبات سنگینی دارد.
- اگر به درستی منظم نشود، ممکن است منجر به بیش‌برازش شود.
- به حافظه بیشتری نیاز دارد.

**زمان استفاده:** آدام به طور گسترده‌ای در اکثر برنامه‌های یادگیری عمیق به دلیل کارایی و اثربخشی آن استفاده می‌شود، بهویژه در معماری‌های پیچیده شبکه‌های عصبی.

#### پنج - چهار - هشت: آدامدبلیو

آدامدبلیو (AdamW) یک اکستنشن از آدام است که شامل تنظیم‌پذیری وزن<sup>۷۲</sup> برای رسیدگی به مشکلات بیش‌برازش موجود در آدام می‌باشد.

<sup>71</sup> Adaptive Moment Estimation (Adam)

<sup>72</sup> weight decay

نحوه کار: آدامدبلیو تنظیم‌پذیری  $L_2$ <sup>73</sup> را به طور مستقیم در بهروزسازی‌های پارامتر ادغام می‌کند و کاهش وزن را از نرخ یادگیری جدا می‌کند. این کار به بهبود عمومیت کمک می‌کند و برای تنظیم دقیق مدل‌های بزرگ مناسب است.

مزایا:

- شامل تنظیم‌پذیری وزن برای بهبود منظم‌سازی است.
- نرخ یادگیری تطبیقی آدام را با تنظیم‌پذیری  $L_2$  ترکیب می‌کند.
- بهبود عمومیت.
- نسبت به آدام، بیش‌براش را کاهش می‌دهد.

معایب:

- کمی پیچیده‌تر از آدام است.
- نیاز به تنظیم دقیق پارامتر کاهش وزن دارد.
- به دلیل محاسبات اضافی، کمی کندتر از آدام است.
- نیاز به حافظه بیشتری دارد.

زمان استفاده: آدامدبلیو برای سناریوهایی که نیاز به منظم‌سازی دارند ایده‌آل است، مانند جلوگیری از بیش‌براش در مدل‌های بزرگ و تنظیم دقیق مدل‌های از پیش آموزش دیده. یک مجموعه جامع از الگوریتم‌های بهینه‌سازی که در کتابخانه PyTorch پیاده‌سازی شده‌اند را می‌توانید در این [اینجا](#)<sup>74</sup> پیدا کنید. پکیج Hugging Face Transformers همچنین انواع مختلفی از بهینه‌سازها برای راهاندازی و تنظیم دقیق مدل‌های زبانی را ارائه می‌دهد که [اینجا](#)<sup>75</sup> در دسترس است.

## پنج - پنج: چالش‌ها در تنظیم آموزش

۱. اطمینان از سازگاری و پیکربندی صحیح سخت‌افزارهای با عملکرد بالا مانند GPU‌ها یا TPU‌ها می‌تواند پیچیده و زمان‌بر باشد.
۲. مدیریت وابستگی‌ها و نسخه‌های فریمورک‌ها و کتابخانه‌های یادگیری عمیق برای جلوگیری از تعارضات و بهره‌گیری از جدیدترین ویژگی‌ها.
۳. انتخاب نرخ یادگیری مناسب بسیار حیاتی است، زیرا نرخ بسیار بالا می‌تواند منجر به همگرایی غیر بهینه شود، در حالی که نرخ بسیار پایین می‌تواند فرآیند آموزش را به طور غیرضروری کند.

<sup>73</sup> [sL2regularisation](#)

<sup>74</sup> <https://pytorch.org/docs/stable/optim.html>

<sup>75</sup> <https://huggingface.co/datasets/huggingface/transformers-metadata>

۴. تعیین اندازه مینی‌بچ بهینه که تعادلی بین محدودیت‌های حافظه و کارایی آموزش ایجاد کند، به ویژه با توجه به نیازهای بالای حافظه مدل‌های بزرگ زبانی (LLMs).
۵. انتخاب تعداد مناسب ایپاک‌ها (epochs) برای جلوگیری از کمبرازش یا بیشبرازش مدل که نیاز به نظارت و اعتبارسنجی دقیق دارد.
۶. انتخاب بهینه‌ساز مناسب برای وظیفه آموزشی خاص به منظور به روزرسانی کارآمد وزن‌های مدل.
۷. انتخاب تابع ضرر صحیح برای اندازه‌گیری دقیق عملکرد مدل و راهنمایی فرآیند بهینه‌سازی.

## ۶ - بهترین شیوه‌ها

- نرخ یادگیری بهینه: از نرخ یادگیری پایین‌تری استفاده کنید، معمولاً بین  $1e-4$  تا  $2e-4$ <sup>۷۶</sup>؛ تا از همگرایی پایدار اطمینان حاصل شود. یک برنامه‌ریزی نرخ یادگیری، مانند افزایش نرخ یادگیری و به دنبال آن کاهش خطی، نیز می‌تواند مفید باشد. این کمک می‌کند تا در ابتدا آموزش پایدار شود و سپس به مدل اجازه می‌دهد تا به طور دقیق‌تری همگرا شود.
- ملاحظات اندازه مینی‌بچ: اندازه مینی‌بچ را انتخاب کنید که تعادلی بین محدودیت‌های حافظه و کارایی آموزش برقرار کند. اندازه‌های کوچکتر مینی‌بچ می‌توانند در دستیابی به همگرایی سریع‌تر کمک کنند، اما ممکن است نیاز به به روزرسانی‌های مکرر داشته باشند. بر عکس، اندازه‌های بزرگتر مینی‌بچ می‌توانند حافظه‌برتر باشند، اما ممکن است به به روزرسانی‌های پایدارتر منجر شوند. با اندازه‌های مختلف مینی‌بچ آزمایش کنید تا تعادل بهینه را برای مورد خاص خود پیدا کنید.
- ذخیره نقاط بازرسی به طور منظم: به طور منظم وزن‌های مدل را در فواصل مختلف در طول ۵ تا ۸ ایپاک (epoch) ذخیره کنید تا عملکرد بهینه را بدون بیشبرازش ثبت کنید. مکانیزم‌های توقف زودهنگام را پیاده‌سازی کنید تا آموزش را زمانی متوقف کنید که عملکرد مدل بر روی مجموعه اعتبارسنجی شروع به کاهش کند، و بدین ترتیب از بیشبرازش جلوگیری شود.
- تنظیم ابرپارامترها: از روش‌های تنظیم ابرپارامترها مانند جستجوی شبکه‌ای، جستجوی تصادفی و بهینه‌سازی بیزی استفاده کنید تا مجموعه بهینه‌ای از ابرپارامترها را پیدا کنید. ابزارهایی مانند Optuna و Ray Tune و Hyperopt می‌توانند این فرایند را خودکار کنند و به طور مؤثری فضای ابرپارامترها را کاوش کنند.
- موازی‌سازی داده و مدل: برای آموزش در مقیاس بزرگ، استفاده از تکنیک‌های موازی‌سازی داده یا موازی‌سازی مدل را در نظر بگیرید تا بار کاری آموزش را بین چندین GPU یا TPU توزیع کنید. کتابخانه‌های

<sup>۷۶</sup> این  $1e-4$  معادل **0.0001** است و  $2e-4$  معادل **0.0002** است. این نوع بازه‌ها معمولاً در تنظیم مقادیر کوچک، مثل نرخ یادگیری (learning rate) یا پارامترهای دیگر الگوریتم‌های بهینه‌سازی، استفاده می‌شوند.

مانند DeepSpeed و Horovod می‌توانند آموزش توزیع شده مؤثر را تسهیل کنند و به کاهش زمان آموزش و مدیریت مؤثر مصرف حافظه کمک کنند.

- نظارت و ثبت اطلاعات منظم؛ نظارت و ثبت اطلاعات جامعی را پیاده‌سازی کنید تا متريک‌های آموزشی، استفاده از منابع و گلوگاه‌های احتمالی را رهگیری کنید. ابزارهایی مانند Weights & TensorBoard می‌توانند بیشتر های بلاذرنگی از فرایند آموزش ارائه دهند و امكان مداخله و تنظیمات به موقع را فراهم کنند.

- مدیریت بیش‌برازش و کم‌برازش: اطمینان حاصل کنید که مدل شما به خوبی تعمیم می‌یابد و از تکنیک‌هایی برای مدیریت بیش‌برازش و کم‌برازش استفاده کنید. تکنیک‌های منظم‌سازی مانند منظم‌سازی L2، دراپ‌اوت و افزایش داده می‌توانند از بیش‌برازش جلوگیری کنند. بر عکس، اگر مدل شما کم‌برازش است، افزایش پیچیدگی مدل یا آموزش برای دوره‌های بیشتر را در نظر بگیرید.

- استفاده از آموزش با دقت مختلط: آموزش با دقت مختلط شامل استفاده از انواع نقطه شناور ۱۶ بیتی و ۳۲ بیتی برای کاهش استفاده از حافظه و افزایش کارایی محاسباتی است. این تکنیک می‌تواند آموزش را به طور قابل توجهی سرعت بخشد و اندازه حافظه مورد نیاز را کاهش دهد، به ویژه هنگام استفاده از مدل‌های بزرگ. Apex، NVIDIA API و TensorFlow پشتیبانی برای پیاده‌سازی آموزش با دقت مختلط ارائه می‌دهند.

- ارزیابی و تکرار: به طور مداوم عملکرد مدل را با استفاده از مجموعه اعتبارسنجی جداگانه ارزیابی کنید و بر اساس نتایج، فرایند آموزش را تکرار کنید. به طور منظم داده‌های آموزشی خود را به روزرسانی کرده و مدل را دوباره آموزش دهید تا آن را با روندها و الگوهای جدید داده به روز نگه دارید.

- مستندسازی و قابلیت تکرار: مستندات کاملی از تنظیمات آموزشی خود، از جمله پیکربندی سخت‌افزار، محیط نرم‌افزاری و ابرپارامترهای استفاده شده نگه‌داری کنید. با تنظیم بذرها تصادفی و ارائه سوابق دقیق از فرایند آموزش، قابلیت تکرار را تضمین کنید. این عمل نه تنها در عیب‌یابی و توسعه بیشتر کمک می‌کند، بلکه همکاری و به اشتراک‌گذاری نتایج با جامعه تحقیقاتی گسترده‌تر را تسهیل می‌کند.

## فصل ششم

### مرحله چهارم: انتخاب تکنیک‌های تنظیم دقیق و پیکربندی‌های مناسب مدل

این فصل بر روی انتخاب تکنیک‌های مناسب تنظیم دقیق و پیکربندی‌های مدل که متناسب با نیازهای خاص وظایف مختلف است، تمرکز دارد. تنظیم دقیق مرحله‌ای حیاتی است که در آن مدل‌های پیش‌آموزش دیده به وظایف یا حوزه‌های خاص تطبیق داده می‌شوند.

#### شش – یک: مراحل مربوط به تنظیم دقیق

مراحل زیر فرآیند تنظیم دقیق را خلاصه کرده و تکنیک‌های پیشرفته و بهترین شیوه‌ها را ادغام می‌کند.

۱. ابتدای بارگذاری توکنایزر و مدل پیش‌آموزش دیده: با بارگذاری توکنایزر<sup>۷۷</sup> و مدل پیش‌آموزش دیده شروع کنید. توکنایزر اطمینان حاصل می‌کند که متن ورودی به فرمت قابل پردازش برای مدل تبدیل می‌شود، در حالی که مدل پیش‌آموزش دیده به عنوان پایه‌ای برای سازگاری‌های بیشتر عمل می‌کند. بسته به وظیفه، مدلی را انتخاب کنید که روی داده‌های مرتبط پیش‌آموزش دیده باشد تا نقطه شروع قوی‌تری فراهم کند.
۲. تنظیم لایه خروجی مدل: لایه خروجی مدل را برای هم‌راستا کردن با نیازهای خاص وظیفه هدف تنظیم کنید. این ممکن است شامل تغییر لایه‌های موجود یا اضافه کردن لایه‌های جدید باشد. به عنوان مثال، وظایفی مانند طبقه‌بندی ممکن است به یک لایه softmax با تعداد کلاس‌های مناسب نیاز داشته باشد، در حالی که وظایف تولید متن ممکن است شامل تغییراتی در مکانیزم رمزگشایی باشد.

---

Tokenizer<sup>۷۷</sup> یا توکن‌ساز ابزاری در پردازش زبان طبیعی و یادگیری ماشین است که یک متن را به بخش‌های کوچک‌تر به نام "توکن" تقسیم می‌کند. این توکن‌ها معمولاً شامل کلمات، زیرکلمات یا حتی کاراکترها هستند و به مدل کمک می‌کنند تا متن را بهتر درک و پردازش کند. به عنوان مثال، در جمله‌ی «کتاب خواندن لذتبخش است»، توکن‌ساز می‌تواند این جمله را به توکن‌هایی مثل «کتاب»، «خواندن»، «لذتبخش»، و «است» تجزیه کند. این فرایند به مدل اجازه می‌دهد تا با اجزای جداگانه‌ی جمله کار کرده و مفهوم کلی متن را تحلیل کند.

**۳. انتخاب استراتژی مناسب تنظیم دقیق:** استراتژی تنظیم دقیقی که بهترین تناسب را با وظیفه و معماری مدل دارد انتخاب کنید. برخی از گزینه‌ها شامل:

- **تنظیم دقیق وظیفه خاص:** برای وظایفی مانند خلاصه‌سازی متن، تولید کد، طبقه‌بندی و پاسخ‌گویی به سوالات، مدل را با استفاده از داده‌های مرتبط تطبیق دهید.

- **تنظیم دقیق دامنه:** مدل را به گونه‌ای سفارشی کنید که متن‌های مرتبط با دامنه‌های خاص، مانند پزشکی، مالی یا حقوقی را درک و تولید کند.

- **تنظیم دقیق بهینه از نظر پارامتر<sup>۷۸</sup>:** تکنیک‌هایی مانند QLORA، LoRA و adapters به تنظیم دقیق با هزینه‌های محاسباتی کاهش یافته با بروزرسانی یک زیرمجموعه کوچک از پارامترهای مدل کمک می‌کنند.

- **تنظیم دقیق<sup>۷۹</sup>:** تعادلی بین حفظ دانش پیش‌آموزش‌دهی و یادگیری وظایف جدید با بروزرسانی تنها نیمی از پارامترهای مدل در هر دور تنظیم دقیق ایجاد کنید.

**۴. راهاندازی حلقه آموزش:** حلقه آموزشی را ایجاد کنید که استراتژی تنظیم دقیق انتخاب شده را شامل شود. حلقه باید شامل بارگذاری داده‌ها، محاسبه از دست دادن (loss)، بازگشت خطای (backpropagation) و بهروزرسانی پارامترها باشد. هنگام استفاده از روش‌های PEFT، اطمینان حاصل کنید که تنها پارامترهای مرتبط به روزرسانی شوند تا حداکثر کارایی حاصل شود. تکنیک‌هایی مانند نرخ یادگیری پویا و توقف زودهنگام را پیاده‌سازی کنید تا فرآیند آموزش را بهبود ببخشید.

**۵. ادغام تکنیک‌ها برای مدیریت چندین وظیفه:** اگر تنظیم دقیق برای چندین وظیفه انجام می‌دهید، استراتژی‌هایی مانند تنظیم دقیق با چندین adapter یا استفاده از معماری‌های Mixture of Experts (MoE) را در نظر بگیرید. این روش‌ها به یک مدل واحد اجازه می‌دهند که وظایف مختلف را با استفاده از زیر شبکه‌های تخصصی یا adapterها برای هر وظیفه مدیریت کند.

**۶. نظارت بر عملکرد در مجموعه اعتبارسنجی:** به طور منظم عملکرد مدل را در یک مجموعه اعتبارسنجی ارزیابی کنید تا اطمینان حاصل کنید که به خوبی به داده‌های دیده‌نشده تعمیم می‌یابد. ابرپارامترهایی مانند نرخ یادگیری، اندازه مینی‌باتج و نرخ‌های دراپ‌اوت را بر اساس عملکرد اعتبارسنجی تنظیم کنید. از ابزارهای نظارتی پیشرفته برای پیگیری متريک‌هایی مانند دقت، از دست دادن و بیش‌برازش استفاده کنید.

**۷. بهینه‌سازی مدل با استفاده از تکنیک‌های پیشرفته:** از تکنیک‌هایی مانند بهینه‌سازی سیاست مجاور<sup>۸۰</sup> (PPO) برای سناریوهای یادگیری تقویتی یا بهینه‌سازی ترجیحات مستقیم<sup>۸۱</sup> (DPO) برای هم‌راستا

<sup>78</sup> Parameter-Efficient Fine-Tuning (PEFT)

<sup>79</sup> Half Fine-Tuning (HFT)

<sup>80</sup> Proximal Policy Optimization

<sup>81</sup> Direct Preference Optimization

کردن خروجی‌های مدل با ترجیحات انسانی استفاده کنید. این تکنیک‌ها به ویژه در تنظیم دقیق مدل‌ها برای وظایفی که نیاز به تصمیم‌گیری دقیق یا پاسخ‌های شبیه به انسان دارند، مفید هستند.

**۸. تنظیم و بهینه‌سازی مدل (دو صورت نیاز):** برای پیاده‌سازی مدل در محیط‌های محدود از لحاظ منابع، از تکنیک‌های هرس (pruning) برای کاهش اندازه و پیچیدگی آن استفاده کنید. این شامل حذف پارامترها یا اجزای غیرضروری بدون تأثیر قابل توجه بر عملکرد است. از روش‌های هرس پویا در حین استنتاج (inference) برای بهینه‌سازی مدل به‌طور آنی در سناریوهای مختلف استفاده کنید.

**۹. ارزیابی و تکرار مستمر:** به‌طور مداوم عملکرد مدل را در میان وظایف مختلف با استفاده از معیارهای مناسب ارزیابی کنید. در فرآیند تنظیم دقیق تکرار کنید و بر اساس متريک‌های عملکرد و آزمایش‌های دنیای واقعی تنظیمات لازم را انجام دهید. این رویکرد تکراری به بهبود مدل در راستای برآورده کردن معیارهای خاص عملکرد کمک می‌کند.

## شش – دو: استراتژی‌های تنظیم دقیق برای مدل‌های زبانی بزرگ (LLMs)

### شش – دو – یک: تنظیم دقیق مبتنی بر وظیفه

تنظیم دقیق مبتنی بر وظیفه، مدل‌های زبانی بزرگ (LLMs) را برای وظایف خاص و پایین‌دستی تطبیق می‌دهد. این کار با استفاده از داده‌های به‌خوبی قالب‌بندی و تمیز شده انجام می‌شود. در ادامه، خلاصه‌ای از وظایف کلیدی که مناسب تنظیم دقیق مدل‌های زبانی بزرگ هستند ارائه شده است، از جمله مثال‌هایی از مدل‌هایی که برای این وظایف خاص تنظیم شده‌اند.

مدل‌های کلیدی	توضیحات	وظیفه
BERTSUM, GPT-3, T5	فسرده‌سازی متن طولانی به خلاصه‌هایی منسجم و حفظ اطلاعات کلیدی. روش‌ها شامل خلاصه‌سازی استخراجی (انتخاب جملات کلیدی) و انتزاعی (تولید جملات جدید) هستند.	خلاصه‌سازی متن
Codex, GPT-3 ، CodeBERT	تولید خودکار کد برنامه‌نویسی بر اساس توضیحات زبان طبیعی، بخش‌های کد موجود، یا ورودی‌های داده‌ای ساختار یافته.	تولید کد
BERT, RoBERTa ، GPT-4	دسته‌بندی متن به برچسب‌های از پیش تعیین شده مانند تحلیل احساسات، طبقه‌بندی موضوعی و طبقه‌بندی موجودیت‌ها.	طبقه‌بندی
BERT, GPT-3, T5	درک و تولید پاسخ‌های دقیق و مرتبط به سوالات زبان طبیعی در یک زمینه مشخص.	پرسش و پاسخ

جدول ۶.۱: مروری بر وظایفی همچون خلاصه‌سازی متن، تولید کد، طبقه‌بندی و پرسش و پاسخ، همراه با مدل‌های کلیدی زبان‌های بزرگ (LLMs) و توضیحات آنها.

### شش - دو - دو: تنظیم دقیق مبتنی بر حوزه

تنظیم دقیق مبتنی بر حوزه بر انطباق مدل با فهم و تولید متونی متمرکز است که به یک حوزه یا صنعت خاص مرتبط هستند. با تنظیم دقیق مدل بر روی یک مجموعه داده از حوزه هدف، توانایی مدل در فهم متنی و دانش تخصصی در وظایف مرتبط با آن حوزه افزایش می‌یابد. در زیر، نمونه‌هایی از مدل‌های LLM تخصصی در حوزه‌های مختلف آورده شده است.

#### حوزه پزشکی

- توضیحات مدل: Med-PaLM 2 بر روی مجموعه داده‌های پزشکی بدقت گردآوری شده آموزش دیده و توانایی پاسخ‌گویی دقیق به سوالات پزشکی را دارد، به طوری که عملکردی قابل مقایسه با متخصصان پزشکی رائمه می‌دهد [۵۵].
- مدل پایه: PaLM 2
- پارامترهای مدل تنظیم شده: نامشخص
- تکنیک‌های استفاده شده برای تنظیم دقیق: تنظیم دقیق بر اساس دستورالعمل (fine-tuning)

#### مجموعه داده‌های استفاده شده:

MedQA –

MedMCQA –

LiveQA –

MedicationQA –

HealthSearchQA –

- نتایج: Med-PaLM 2 در تعدادی از معیارهای پزشکی کلیدی عملکرد بهتری نسبت به GPT-4 داشت و توانایی برتری در مدیریت وظایف پیچیده دانش و استدلال پزشکی از خود نشان داد.

#### حوزه مالی

- توضیحات مدل: FinGPT یک مدل LLM منبع باز و ویژه حوزه مالی است که با تسهیل دسترسی به داده‌ها و مدیریت مسائلی مانند کسب و کیفیت داده‌ها، پژوهش و همکاری در زمینه مالی را بهبود می‌بخشد [۵۶].
- مدل پایه: ChatGLM، LLaMA و دیگر مدل‌های ترنسفورمر
- پارامترهای مدل تنظیم شده: نامشخص
- تکنیک‌های استفاده شده برای تنظیم دقیق: LoRA، یادگیری تقویتی بر اساس قیمت سهام (RLSP)

## - مجموعه داده‌های استفاده شده:

- اخبار مالی (رویترز، CNBC، یاهو فاینانس)
- شبکه‌های اجتماعی (توییتر، فیسبوک، ردیت، ویبو)
- گزارش‌های نظارتی (مثل گزارش‌های SEC)
- ترندتها (Seeking Alpha)
- مجموعه داده‌های دانشگاهی
- نتایج: ندارد

## حوزه حقوقی

- توضیحات مدل: LAWGPT، اولین مدل منبع باز به‌طور خاص طراحی شده برای کاربردهای حقوقی در چین است و توانایی بالایی در انجام وظایف حقوقی چینی دارد [۵۷].
- مدل پایه: مدل پایه چینی Alpaca Plus 7B
- پارامترهای مدل تنظیم شده: نامشخص
- تکنیک‌های استفاده شده برای تنظیم دقیق: LoRA با قالب Alpaca
- مجموعه داده‌های استفاده شده:
- مجموعه داده‌های منبع باز: ۲۰۰,۰۰۰ مثال شامل پیش‌بینی نوع جرم و مشاوره جرم
- مجموعه داده JEC-QA: ۲۰,۰۰۰ مثال شامل وظایف پاسخ به سوالات حقوقی
- مجموعه داده حقوقی ساخته شده: ۸۰,۰۰۰ مثال که از مجموعه‌های منبع باز و JEC-QA با استفاده از ChatGPT اصلاح شده‌اند.
- نتایج: LAWGPT بهبود قابل توجهی نسبت به مدل LLaMA 7B در وظایف حقوقی مختلف نشان داده است، اما هنوز نسبت به مدل‌های اختصاصی مانند GPT-3.5 Turbo و GPT-4 عقب‌تر است.

## حوزه داروسازی

- توضیحات مدل: PharmaGPT، مجموعه‌ای از مدل‌های بزرگ زبانی ویژه حوزه‌های دارویی و شیمیایی است که استاندارد جدیدی برای دقت در این حوزه‌ها ارائه می‌دهد [۵۸].
- مدل پایه: سری LLaMA
- پارامترهای مدل تنظیم شده: B13 و B70
- تکنیک‌های استفاده شده برای تنظیم دقیق: تنظیم دقیق بر اساس دستورالعمل و RLHF
- مجموعه داده‌های استفاده شده:
- داده‌های حوزه تخصصی از مقالات علمی و گزارش‌های بالینی

- داده‌های متنی از فرمتهای مجموعه داده NLP (مانند سوال و جواب، خلاصه‌سازی، گفت‌و‌گو)
- مجموعه داده تنظیم دقیق بر اساس دستورالعمل برای یادگیری چند وظیفه‌ای
- مجموعه داده RLHF با دستورالعمل‌های حاشیه‌نویسی شده توسط کارشناسان
- نتایج: مدل‌های PharmaGPT عملکرد چشم‌گیری در معیارهای حوزه داروسازی نشان داده و به طور پیوسته از GPT-3.5 Turbo پیشی گرفته‌اند.

## حوزه مالی

- توضیحات مدل: Palmyra-Fin-70B-32K، که توسط Writer توسعه داده شده، یک مدل بزرگ زبانی پیشرو و به طور خاص طراحی شده برای حوزه مالی است [۵۹].
- مدل پایه: LLaMA
- پارامترهای مدل تنظیم شده: BYO
- تکنیک‌های استفاده شده برای تنظیم دقیق: نامشخص
- مجموعه داده‌های استفاده شده: نامشخص
- نتایج: Palmyra-Fin-70B-32K عملکردی در سطح جهانی داشته و نتایج برتری را در مجموعه داده‌های مالی مختلف به دست آورده است، بهویژه در تحلیل اسناد مالی، پیش‌بینی روند بازار و ارزیابی ریسک.

## شش – سه: تکنیک‌های تنظیم دقیق بهینه از نظر پارامتر (PEFT)

تنظیم دقیق کارآمد بر اساس پارامتر (PEFT<sup>82</sup> یا Parameter Efficient Fine Tuning) یک تکنیک مؤثر در پردازش زبان طبیعی (NLP) است که مدل‌های زبان از پیش‌آموزش دیده را با کارآیی بالا به کاربردهای مختلف تطبیق می‌دهد. روش‌های PEFT تنها یک زیرمجموعه کوچک از پارامترهای اضافی مدل را تنظیم دقیق می‌کنند، در حالی که بیشتر پارامترهای مدل زبان بزرگ (LLM) از پیش‌آموزش دیده را ثابت نگه می‌دارند، و به این ترتیب به‌طور قابل توجهی هزینه‌های محاسباتی و ذخیره‌سازی را کاهش می‌دهند. این رویکرد مشکل فراموشی فاجعه‌آمیز<sup>۸۳</sup> را کاهش می‌دهد؛ پدیده‌ای که در آن شبکه‌های عصبی دانش پیشین را از دست می‌دهند و با آموزش بر روی مجموعه داده‌های جدید دچار کاهش عملکرد قابل توجهی در وظایف یادگرفته شده قبلی می‌شوند. روش‌های PEFT در مقایسه با تنظیم دقیق کامل، بهویژه در سناریوهای با داده کم، عملکرد برتری از خود نشان داده و تعمیم بهتری برای حوزه‌های خارج از دامنه اصلی دارند. این تکنیک در موارد مختلفی مانند طبقه‌بندی احساسات مالی و ترجمه ماشینی اصطلاحات پزشکی کاربرد دارد. یک طبقه‌بندی از رویکردهای تنظیم دقیق

<sup>82</sup> <https://github.com/huggingface/peft>

<sup>83</sup> catastrophic forgetting

مبتنی بر PEFT در شکل ۶.۱ آورده شده است. در بخش‌های بعدی، چند مورد از رویکردهای کلیدی مبتنی بر PEFT را بررسی خواهیم کرد.

### شش – سه – یک: آداتورها

روش‌های مبتنی بر آداتور، پارامترهای قابل آموزش اضافی را پس از لایه‌های توجه و لایه‌های کاملاً متصل در یک مدل پیش‌آموزش دیده و ثابت اضافه می‌کنند تا میزان حافظه مصرفی کاهش یابد و آموزش سرعت پیدا کند. روش خاص هر آداتور ممکن است شامل افزودن یک لایه اضافی یا نمایش تغییرات وزن دلتا ( $W$ ) به عنوان یک تجزیه پایین‌رتبه از ماتریس وزن باشد. بدون توجه به روش، آداتورها به‌طور کلی کوچک هستند، اما عملکردی مشابه مدل‌های کاملاً تنظیم شده را ارائه می‌دهند، که به آموزش مدل‌های بزرگ‌تر با منابع کمتر کمک می‌کند. کتابخانه HuggingFace از پیکربندی‌های آداتور از طریق کتابخانه PEFT پشتیبانی می‌کند. در طول تنظیم دقیق، آداتورهای جدید با استفاده از LoraConfig به مدل اضافه می‌شوند. HuggingFace از PeftConfig برای بارگذاری مدل‌های از پیش‌آموزش دیده موجود و اعمال تکنیک‌های PEFT استفاده می‌کند. علاوه بر این، HuggingFace از قابلیت پشتیبانی داخلی برای اجرای فرایند تنظیم دقیق در هر نوع پیکربندی توزیع شده با استفاده از Accelerate2 برخوردار است که آموزش و استنتاج در مقیاس بزرگ را ساده، کارآمد و قابل انطباق می‌کند.



شکل ۶.۱: طبقه‌بندی جامع روش‌های تنظیم دقیق کارآمد بر اساس پارامتر (PEFT) برای مدل‌های زبان بزرگ (LLMs)، این شکل، تکنیک‌های مختلف PEFT را دسته‌بندی می‌کند و به رویکردهای متمایز آن‌ها، از تنظیم دقیق افزودنی و انتخابی تا روش‌های بازپارامتری شده و ترکیبی اشاره می‌کند. همچنین استراتژی‌های خاص در هر دسته، مانند تنظیم دقیق مبتنی بر آداتور، تنظیم دقیق مبتنی بر نرم‌پرامپت و زیرتکنیک‌های مربوطه مانند LORA و مشتقات آن را نمایش می‌دهد، که تنوع و روند تکاملی تنظیم دقیق در LLM‌ها را به نمایش می‌گذارد (اقتباس از [۶۰]).

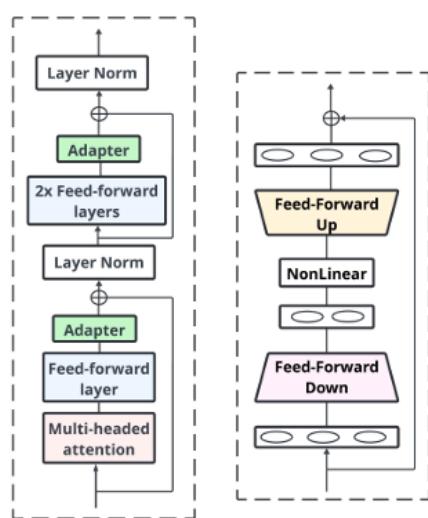
### ۶-۳-۲- تطبیق کمرتبه (LORA)

تطبیق کمرتبه (LORA<sup>84</sup>) [۶۲] تکنیکی برای تنظیم دقیق مدل‌های زبان بزرگ است که فرآیند تنظیم دقیق را با ثابت نگهداشتن وزن‌های اصلی مدل و اعمال تغییرات به مجموعه‌ای جداگانه از وزن‌ها، که به پارامترهای اصلی اضافه می‌شوند، اصلاح می‌کند. LORA پارامترهای مدل را به ابعاد کمرتبه تبدیل می‌کند، تعداد پارامترهای قابل آموزش را کاهش داده، فرآیند را سرعت بخشیده و هزینه‌ها را کاهش می‌دهد. این روش بهویژه در شرایطی مفید است که چندین کاربر مدل‌های تنظیم شده دقیق برای کاربردهای مختلف نیاز دارند، زیرا امکان ایجاد وزن‌های خاص برای هر کاربرد را بدون نیاز به مدل‌های جداگانه فراهم می‌آورد. با استفاده از روش‌های تقریب کمرتبه، LORA به‌طور مؤثری نیازهای محاسباتی و منابع را کاهش می‌دهد و در عین حال قابلیت تطبیق مدل از پیش‌آموزش دیده را برای وظایف یا حوزه‌های خاص حفظ می‌کند.

#### مزایای استفاده از LORA

۱. کارایی پارامتری: LORA تعداد پارامترهایی که باید آموزش داده شوند را با تمرکز فقط روی ماتریس‌های کمرتبه به‌طور قابل توجهی کاهش می‌دهد، که منجر به کاهش نیاز به حافظه و فضای ذخیره‌سازی در مقایسه با تنظیم دقیق کامل می‌شود.

۲. ذخیره‌سازی کارآمد: ذخیره‌سازی مدل آموزش دیده با LORA کارآمدتر است زیرا تنها نیاز به ذخیره ماتریس‌های کمرتبه است، نه وزن‌های کامل مدل.



<sup>84</sup> Low-Rank Adaptation

شکل ۶.۲: نمای شماتیک معماری آداتپر مورد استفاده در مدل‌های زبان بزرگ (LLMs). این دیاگرام، نحوه ادغام آداتپورها در معماری ترنسفورمر را نشان می‌دهد، شامل لایه‌های پیش‌رونده<sup>۸۵</sup> بالا و پایین و نقش آن‌ها در تطبیق کارآمد مدل از طریق افروزن پارامترهای اضافی در حالی که ساختار اصلی مدل حفظ می‌شود (اقتباس از [۶۱]).

۳. کاهش بار محاسباتی: آموزش با ماتریس‌های کمرتبه به منابع محاسباتی کمتری نیاز دارد که باعث سریع‌تر شدن و مقیاس‌پذیری بیشتر می‌شود.

۴. ردپای حافظه کمتر: از آنجایی که پارامترهای کمتری به روزرسانی می‌شوند، ردپای حافظه<sup>۸۶</sup> حین آموزش کاهش یافته و امکان استفاده از سایزهای بزرگ‌تر بچ یا مدل‌های پیچیده‌تر در همان محدودیت‌های سخت‌افزاری فراهم می‌شود.

۵. انعطاف‌پذیری: LORA می‌تواند به راحتی با مدل‌های از پیش‌آموزش‌دیده یکپارچه شود بدون نیاز به تغییرات گسترده در معماری مدل.

۶. سازگاری: این تکنیک می‌تواند در کنار سایر تکنیک‌های تنظیم دقیق، مانند لایه‌های آداتپر یا پرامپت-تیونینگ<sup>۸۷</sup>، استفاده شود تا عملکرد را بهبود بخشد.

۷. نتایج قابل مقایسه: با وجود کاهش تعداد پارامترهای قابل آموزش، LORA در بسیاری از وظایف عملکردی معادل با تنظیم دقیق کامل به دست آورده است.

۸. تطبیق وظیفه‌محور: این تکنیک به طور مؤثر مدل از پیش‌آموزش‌دیده را برای وظایف خاص تطبیق می‌دهد و از دانش از پیش جاسازی‌شده در مدل اصلی بهره می‌گیرد.

۹. جلوگیری از بیش‌برازش: با تمرکز بر به روزرسانی‌های کمرتبه، LORA می‌تواند به کاهش بیش‌برازش کمک کند، به ویژه هنگام کار با مجموعه‌داده‌های کوچک وظیفه‌محور.

## محدودیت‌ها

در حالی که LORA قابلیت‌های قابل توجهی نشان می‌دهد، چالش‌هایی نیز دارد:

- دامنه تنظیم دقیق: ممکن است LORA در مواجهه با وظایفی که به تغییرات عمدی در نمایش‌های داخلی مدل از پیش‌آموزش‌دیده نیاز دارند، دچار مشکل شود.

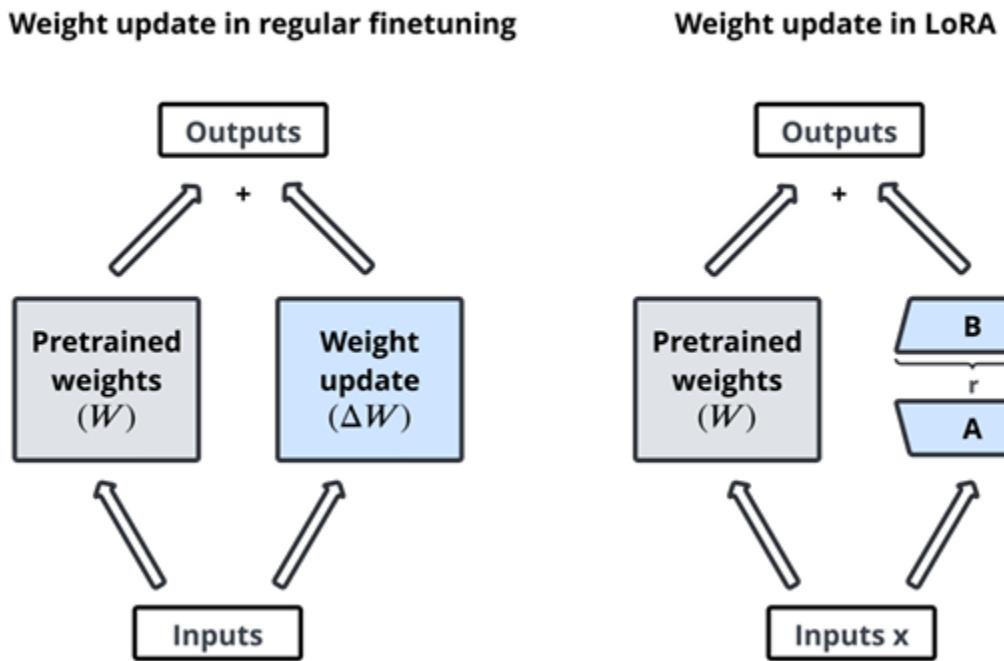
<sup>۸۵</sup> در علوم رایانه و یادگیری ماشین، به ساختاری در شبکه‌های عصبی اشاره دارد که در آن داده‌ها تنها در یک جهت از لایه ورودی به سمت لایه خروجی جریان پیدا می‌کنند، بدون هیچ‌گونه حلقه یا بازخورد (feedback) به لایه‌های قبلی. به این معنی که هر لایه فقط داده‌های خود را به لایه بعدی می‌فرستد. در شبکه‌های عصبی feed-forward، اطلاعات از ورودی به خروجی جریان می‌یابد و این فرآیند یک بار انجام می‌شود؛ برخلاف شبکه‌های بازگشتی (Recurrent Neural Networks) که دارای حلقه‌ها هستند و اطلاعات می‌توانند به لایه‌های قبلی بازگردند. این نوع شبکه‌ها به دلیل سادگی و سرعت محاسباتی، اغلب در مسائل دسته‌بندی و پیش‌بینی داده استفاده می‌شوند.

<sup>۸۶</sup> Memory Footprint

<sup>۸۷</sup> Prompt-tuning

- بهینه‌سازی ابرپارامتر: تنظیم پارامتر رتبه "۲" نیاز به تنظیمات دقیق برای دستیابی به عملکرد بهینه دارد.

- تحقیقات در حال پیشرفت: با وجود وعده‌های آن، LoRA هنوز در مراحل تحقیقاتی فعال است و تأثیرات بلندمدت آن به‌طور کامل بررسی نشده است.



شکل ۶.۳: مقایسه‌ای بین بهروزرسانی وزن‌ها در تنظیم دقیق معمولی و تنظیم دقیق LoRA . در تنظیم دقیق معمولی، کل ماتریس بهروزرسانی وزن ( $W\Delta$ ) به وزن‌های از پیش‌آموزش دیده اعمال می‌شود. در مقابل، تنظیم دقیق LoRA دو ماتریس کم‌رتبه ( $A$  و  $B$ ) را معرفی می‌کند که ماتریس بهروزرسانی وزن ( $W\Delta$ ) را تقریبی‌سازی می‌کنند و تعداد پارامترهای قابل آموزش را با استفاده از بعد داخلي (۲) که یک ابرپارامتر است، بهشت کاهش می‌دهد. این روش از نظر حافظه و محاسبات کارآمدتر است و برای تنظیم دقیق مدل‌های بزرگ ایده‌آل است (اقتباس از [۶۳]).

با وجود این چالش‌ها، LoRA به عنوان یک تکنیک پیشرو با پتانسیل بالا شناخته می‌شود که می‌تواند دسترسی به قابلیت‌های مدل‌های زبان بزرگ (LLMs) را دموکراتیک کند. تحقیقات و توسعه‌های مستمر، چشم‌انداز غله بر محدودیت‌های فعلی و دستیابی به کارایی و سازگاری بیشتر را ارائه می‌دهند.

### آموزش برای تنظیم دقیق LLM با استفاده از LoRA

یک قالب متن‌باز برای تنظیم دقیق مدل‌های زبان بزرگ با روش LoRA و با استفاده از کتابخانه Hugging Face در این [لينك<sup>۸۸</sup>](#) دسترسی است. این قالب به‌طور ویژه برای تطبیق مدل‌های زبان بزرگ در فرآیندهای تنظیم دقیق دستورالعمل‌ها طراحی شده است.

<sup>۸۸</sup> [https://gitlab.com/CeADARIreland\\_Public/llm-resources](https://gitlab.com/CeADARIreland_Public/llm-resources)

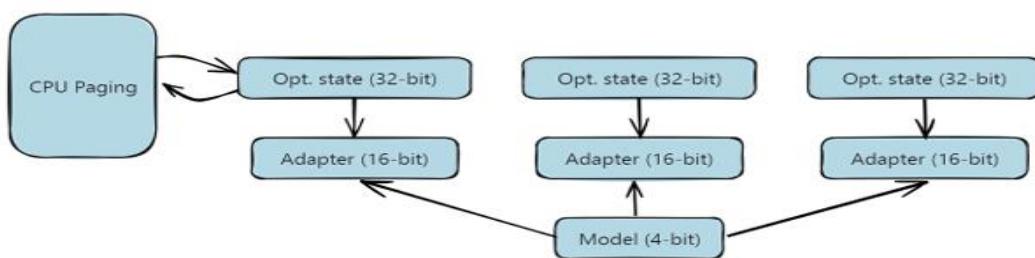
## شش - سه - سه: QLoRA

[۶۴] نسخه‌ای توسعه‌یافته از LORA است که برای افزایش کارایی حافظه در مدل‌های زبان بزرگ (LLMs) با کوانتیزه کردن پارامترهای وزن به دقت ۴ بیت طراحی شده است. به طور معمول، پارامترهای در قالب ۳۲ بیتی ذخیره می‌شوند، اما QLoRA آن‌ها را به ۴ بیت فشرده می‌کند که بهشت ردپایی حافظه را کاهش می‌دهد. این امر امکان تنظیم دقیق را روی سخت‌افزارهای کمتر قدرتمند، از جمله GPU‌های مصرفی، فراهم می‌کند. QLoRA همچنین وزن‌های آداتورهای Lora را از ۸ بیت به ۴ بیت کوانتیزه می‌کند و نیازهای حافظه و ذخیره‌سازی را بیشتر کاهش می‌دهد (به شکل ۶.۴ مراجعه کنید). علیرغم کاهش دقت بیت، سطح عملکردی مشابه تنظیم دقیق ۱۶ بیتی سنتی را حفظ می‌کند.

این عملکرد را از طریق پس انتشار گرادیان‌ها<sup>۸۹</sup> در یک مدل زبان از پیش‌آموزش دیده، منجمد و ۴ بیتی کوانتیزه شده به آداتورهای کم‌رتبه به دست می‌آورد که فرآیند تنظیم دقیق را کارآمد کرده و در عین حال اثربخشی مدل را حفظ می‌کند. پیکربندی QLoRA از طریق کتابخانه PEFT پشتیبانی می‌شود که از BitsAndBytesConfig و LoraConfig برای کوانتیزاسیون استفاده می‌کند. نوآوری‌هایی مانند یک نوع داده بهینه ۴ بیتی، کوانتیزاسیون مضاعف ثابت‌ها و مدیریت افزایش حافظه، باعث می‌شوند که QLoRA استفاده از حافظه را از ۹۶ بیت به ازای هر پارامتر در تنظیم دقیق سنتی به ۵.۲ بیت به ازای هر پارامتر کاهش دهد که به معنای کاهش ۱۸ برابری است.

از نظر عملکرد، QLoRA از کوانتیزاسیون ۴ بیتی ساده بهتر عمل کرده و با مدل‌های کوانتیزاسیون ۱۶ بیتی در بنچمارک‌ها برابر می‌کند. همچنین، QLoRA امکان تنظیم دقیق یک چتبات ۴ بیتی با کیفیت بالا را با استفاده از یک GPU و در ۲۴ ساعت فراهم کرد که کیفیتی قابل مقایسه با ChatGPT به دست آورد.

این آموزش<sup>۹۰</sup> مراحل کامل تنظیم دقیق QLoRA را بر روی یک مجموعه‌داده سفارشی برای مدل Phi-2 توضیح می‌دهد.



یک الگوریتم کلیدی در آموزش شبکه‌های عصبی است که به طور خودکار وزن‌های شبکه را تنظیم می‌کند تا خطای پیش‌بینی‌ها کاهش یابد. در این فرآیند، ابتدا داده‌ها از طریق شبکه به جلو گردیان پیدا می‌کنند (feed-forward) و خطای خروجی محاسبه می‌شود. سپس، این خطای خروجی به سمت عقب شبکه، لایه به لایه، انتشار می‌باید تا گرادیان‌ها (نرخ تغییرات خطای نسبت به وزن‌ها) به دست آید. با داشتن گرادیان‌ها، از روشی مانند گرادیان کاهشی (Gradient Descent) برای بهروزرسانی وزن‌های شبکه استفاده می‌شود. هدف این است که با کم کردن خطای در هر مرحله، مدل به تدریج به وزن‌هایی برسد که باعث پیش‌بینی‌های دقیق‌تر می‌شوند. پس انتشار گرادیان‌ها یکی از مهم‌ترین روش‌ها در یادگیری عمیق است، زیرا شبکه‌های عصبی پیچیده را به طور کارآمد و قابل اعتماد آموزش می‌دهد.

<sup>89</sup> <https://dassum.medium.com/fine-tune-large-language-model-lm-on-a-custom-dataset-with-qlora-fb60abdeba07>

شکل ۴.۶: جریان کاری بهینه‌سازی با استفاده از تطبیق کمرتبه کوانتیزه شده (QLORA). این شکل فرآیند بهینه‌سازی QLORA را نشان می‌دهد و تعاملات بین وضعیت‌های بهینه‌سازی، آداپتورها و مدل را در حین تنظیم دقیق به نمایش می‌گذارد. این نمودار نشان می‌دهد که چگونه از پهنهای بیت‌های مختلف (۳۲ بیت، ۱۶ بیت و ۴ بیت) برای بهینه‌سازی حافظه و کارایی محاسباتی در تنظیم دقیق مدل‌های زبان بزرگ استفاده می‌شود (اقتباس از [۶۵]).

### شش - سه - چهار: تطبیق کمرتبه با تجزیه وزن<sup>۹۱</sup> (DoRA)

در زمینه بهینه‌سازی تنظیم دقیق مدل، تحلیل الگوهای LORA و تنظیم دقیق کامل<sup>۹۲</sup> (FT) تفاوت‌های قابل توجهی در رفتارهای یادگیری و بهروزرسانی‌ها را نشان می‌دهد. LORA با به کارگیری استراتژی بهروزرسانی تدریجی وزن‌های از پیش‌آموزش دیده با استفاده از حاصل ضرب دو ماتریس کمرتبه، وزن‌های اصلی را عمدتاً ثابت نگه می‌دارد که این امر به کارایی در استنتاج کمک می‌کند. با وجود کارایی محاسباتی آن، مطالعات قبلی نشان داده‌اند که تعداد محدود پارامترهای قابل آموزش در LORA ممکن است به تفاوت‌های عملکردی آن نسبت به تنظیم دقیق کامل (FT) منجر شود.

تطبیق کمرتبه با تجزیه وزن (DoRA) [۶۶] یک روش جدید برای تنظیم دقیق است که با تجزیه وزن‌های مدل‌های از پیش‌آموزش دیده به مؤلفه‌های دامنه و جهت، آن‌ها را بهینه‌سازی می‌کند. این روش از کارایی LORA برای بهروزرسانی‌های جهت‌دار بهره می‌برد و بهروزرسانی‌های قابل توجهی در پارامترها بدون تغییر در کل ساختار مدل امکان‌پذیر می‌سازد. DoRA چالش‌های محاسباتی مرتبط با تنظیم دقیق کامل سنتی (FT) را با حفظ سادگی مدل و کارایی در استنتاج رفع می‌کند و به طور همزمان فاصله عملکردی معمولاً مشاهده شده بین FT و LORA را پر می‌کند. ارزیابی‌های تجربی و نظری نشان می‌دهند که DoRA نه تنها نتایج یادگیری مشابهی با FT در وظایف متنوع (از جمله پردازش زبان طبیعی و برنامه‌های بینایی-زبان<sup>۹۳</sup>) ارائه می‌دهد، بلکه در عملکرد نیز به طور مداوم از LORA پیشی می‌گیرد و راهکاری قوی برای افزایش سازگاری و کارایی مدل‌های بزرگ مقیاس فراهم می‌کند.

کتابخانه پایتون - DoRA از طریق بسته LoraConfig در کتابخانه HuggingFace پشتیبانی می‌شود. برای ادغام DoRA در فرآیند تنظیم دقیق، لازم است پارامتر `use\_dora = True` را در پیکربندی DoRA مشخص کنید. اطلاعات بیشتر در مورد تنظیم اولیه را می‌توانید [\[ینجا\]<sup>۹۴</sup>](#) پیدا کنید.

#### مزایای DoRA

۱. ظرفیت یادگیری افزایش یافته: DoRA با تجزیه وزن‌های از پیش‌آموزش دیده به مؤلفه‌های دامنه و جهت، ظرفیتی شبیه به تنظیم دقیق کامل (FT) را برای یادگیری فراهم می‌کند و امکان بهروزرسانی‌های دقیق‌تر را می‌دهد.

<sup>۹۱</sup> Weight-Decomposed Low-Rank Adaptation (DoRA)

<sup>۹۲</sup> Full Fine-Tuning

<sup>۹۳</sup> vision-language applications

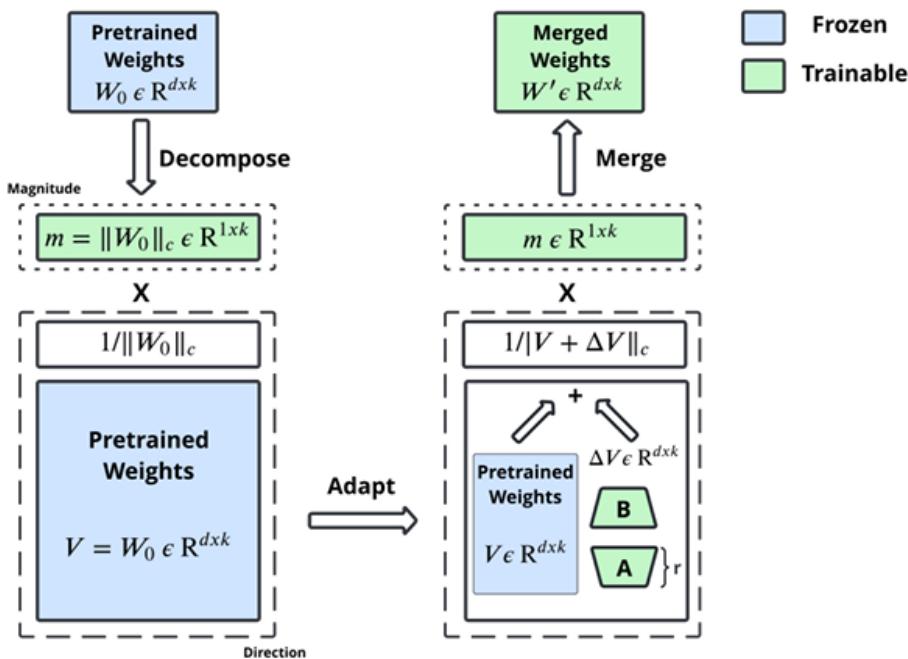
<sup>۹۴</sup> [https://huggingface.co/docs/peft/v0.8.2/en/package\\_reference/lora](https://huggingface.co/docs/peft/v0.8.2/en/package_reference/lora)

۲. تنظیم دقیق کارآمد: با استفاده از مزایای ساختاری LoRA برای بهروزرسانی‌های جهتدار، تنظیم دقیق کارآمدی را بدون تغییر در کل ساختار مدل ممکن می‌سازد.

۳. بدون تأخیر اضافی در استنتاج: با وجود قابلیت‌های بهبودیافته یادگیری، DoRA هیچ تأخیر اضافی در استنتاج نسبت به LoRA ایجاد نمی‌کند و سادگی و کارایی مدل را حفظ می‌کند.

۴. عملکرد برتر: نتایج تجربی نشان می‌دهند که DoRA به طور مداوم در طیف وسیعی از وظایف، از جمله پردازش زبان طبیعی (NLP)، تنظیم دقیق بصری دستورات و درک تصویر/ویدیو - متن، از LoRA پیشی می‌گیرد. به عنوان مثال، DoRA بهبودهای قابل توجهی در معیارهای استدلال مبتنی بر عقل سليم و تنظیم دقیق بصری دستورات نشان می‌دهد.

۵. انعطاف‌پذیری در میان مدل‌های پایه: DoRA در میان مدل‌های پایه مختلف، از جمله مدل‌های زبان بزرگ (LLM) و مدل‌های زبان - بینایی<sup>۹۵</sup> (LVLM)، اعتبارسنجی شده است که نشان‌دهنده کاربرد گسترده و پایداری آن در حوزه‌های مختلف است.



شکل ۶.۵: نمای کلی از DoRA (تطبیق کمرتبه با تجزیه وزن) این شکل، روش تجزیه و تطبیق وزن‌ها در DoRA را نشان می‌دهد. در بخش سمت چپ، وزن‌های از پیش‌آموزش دیده به دو مؤلفه دامنه و جهت تجزیه می‌شوند. بخش سمت راست نحوه ادغام این وزن‌های تجزیه شده با پارامترهای قابل آموزش در حین تنظیم دقیق را نشان می‌دهد، که در نهایت به وزن‌های بهروزرسانی شده‌ای منجر می‌شود که ترکیبی از مؤلفه‌های ثابت (آبی) و قابل آموزش (سبز) هستند. این فرآیند با تمرکز بر جهت‌های مهم در فضای پارامترها، به تطبیق کارآمد کمک می‌کند و تنظیم دقیق مؤثری را فراهم می‌آورد، در حالی که انسجام مدل اصلی حفظ می‌شود (اقتباس از [۶۶]).

<sup>95</sup> vision-language models

۶. تحلیل نوآورانه: معرفی یک تحلیل نوین تجزیه وزن<sup>۹۶</sup> به کشف تفاوت‌های بنیادی در الگوهای یادگیری تنظیم دقیق کامل (FT) و روش‌های مختلف تنظیم دقیق با پارامترهای کارآمد (PEFT) کمک می‌کند و به درک عمیق‌تری از دینامیک تنظیم دقیق مدل می‌افزاید.

### مقایسه بین DoRA و LoRA

تطبیق کمرتبه LoRA و تطبیق کمرتبه با تجزیه وزن DoRA هر دو تکنیک‌های پیشرفته‌ای هستند که برای بهبود کارایی و اثربخشی تنظیم دقیق مدل‌های بزرگ از پیش‌آموزش دیده طراحی شده‌اند. در حالی که هر دو هدف مشترک کاهش سربار محاسباتی را دنبال می‌کنند، از راهبردهای متفاوتی برای دستیابی به این هدف استفاده می‌کنند (به جدول ۶.۲ مراجعه کنید).

معیارها	LoRA (تطبیق کم رتبه)	DoRA (تطبیق کمرتبه با تجزیه وزن)
هدف	ارائه یک روش کارآمد برای تنظیم دقیق مدل‌های از پیش‌آموزش دیده با استفاده از ضرب ماتریس‌های کمرتبه برای بهروزرسانی تدریجی وزن‌ها بدون افزایش تأخیر استنتاج.	ظرفیت یادگیری را با تقلید نزدیک از الگوهای یادگیری تنظیم دقیق کامل بهبود می‌بخشد و دامنه و جهت را به طور جداگانه بهینه‌سازی می‌کند.
رویکرد	یک تجزیه کمرتبه را پیاده‌سازی می‌کند که در آن بهروزرسانی وزن به صورت محصول دو ماتریس کمرتبه (A and B) مدل‌سازی می‌شود و وزن‌های اصلی ثابت باقی می‌مانند.	از تحلیل تجزیه وزن برای پارامترسازی مجدد ماتریس وزن به مؤلفه‌های جداگانه دامنه و جهت برای بهروزرسانی‌های مجزا استفاده می‌کند.
معماری مدل	ماتریس وزن از پیش‌آموزش دیده (W0) را تغییر نمی‌دهد و بهروزرسانی‌ها را با استفاده از ماتریس‌های کمرتبه (A and B) اعمال می‌کند. ماتریس A با توزیع یکنواخت Kaiming <sup>۹۷</sup> مقداردهی اولیه می‌شود، در حالی که B در ابتدا به صفر تنظیم می‌شود.	ماتریس وزن را به مؤلفه‌های دامنه و جهت مجدد ساختار می‌دهد و اطمینان حاصل می‌کند که بردارهای جهتی، بردارهای واحد هستند تا تنظیمات دقیق‌تری انجام شود.

جدول ۶.۲: مقایسه‌ای دقیق بین LoRA (تطبیق کمرتبه) و DoRA (تطبیق کمرتبه با تجزیه وزن)، که اهداف، رویکردها و استراتژی‌های معما ری خاصی که برای تنظیم دقیق مدل‌های زبان بزرگ به کار می‌برند، را بر جسته می‌کند.

### آموزش تنظیم دقیق LLM با استفاده از DoRA

این راهنمای<sup>۹۸</sup> یک توضیح عمیق و دقیق از مراحل مربوط به پیاده‌سازی DoRA از ابتدا ارائه می‌دهد و همچنین بینش‌هایی درباره فرآیند تنظیم دقیق که برای بهینه‌سازی عملکرد ضروری است، ارائه می‌دهد.

<sup>96</sup> novel weight decomposition analysis

<sup>97</sup> uniform Kaiming distribution

<sup>98</sup> <https://www.kaggle.com/code/aisuko/dora-from-scratch>

## شش - سه - پنج: تنظیم دقیق با چندین آداتور

در طول تنظیم دقیق، ما روش ثابت نگهداشتن پارامترهای LLM و تمرکز صرفاً بر تنظیم دقیق چند میلیون پارامتر قابل آموزش با استفاده از LORA را بررسی کردیم. به عنوان مثال، تنظیم دقیق یک LLM برای ترجمه شامل آموزش یک آداتور ترجمه با داده‌های مربوطه است. این رویکرد به ما این امکان را می‌دهد که آداتورهای جداگانه‌ای را برای هر کار خاصی که می‌خواهیم LLM انجام دهد، تنظیم کنیم. با این حال، یک سوال کلیدی پیش می‌آید: آیا می‌توانیم چندین آداتور را در یک آداتور چندوظیفه‌ای یکپارچه کنیم؟ به عنوان مثال، اگر ما آداتورهای جداگانه‌ای برای وظایف ترجمه و خلاصه‌سازی داشته باشیم، آیا می‌توانیم آن‌ها را ترکیب کنیم به طوری که LLM بتواند هر دو وظیفه را به خوبی انجام دهد؟ (از طریق شکل ۶.۶ نشان داده شده است). کتابخانه PEFT فرآیند ترکیب آداتورها را باتابع افزودن آداتور وزنی<sup>۹۹</sup> خود ساده می‌کند که سه روش مختلف را ارائه می‌دهد:

۱. **الحق:** این روش ساده، پارامترهای آداتورها را الحق می‌کند. به عنوان مثال، اگر دو آداتور هر کدام دارای رتبه ۱۶ باشند، آداتور حاصل دارای رتبه ۳۲ خواهد بود. این روش بسیار کارآمد است.
  ۲. **ترکیب خطی:** اگرچه کمتر مستند شده است، به نظر می‌رسد این روش مجموع وزنی پارامترهای آداتورها را انجام می‌دهد.
  ۳. **تحلیل SVD:** روش پیش‌فرض از تجزیه مقادیر منفرد از طریق `torch.linalg.svd` استفاده می‌کند. در حالی که این روش چندمنظوره است، به طور قابل توجهی کندر از سایر روش‌های است، به ویژه برای آداتورهایی با رتبه‌های بالا (بیش از ۱۰۰) که ممکن است چندین ساعت به طول بینجامد. هر روش به شما این امکان را می‌دهد که با تنظیم وزن‌ها، ترکیب را سفارشی کنید. به عنوان مثال، هنگام ترکیب دو آداتور، X و Y، اختصاص وزن بیشتر به X اطمینان می‌دهد که آداتور حاصل رفتار مشابه X را بر Y اولویت می‌دهد.
- این رویکرد به ویژه برای یکپارچه‌سازی یک LLM برای انجام چندین وظیفه مناسب است و دیگر نیازی به ایجاد مدل‌های جداگانه برای هر حوزه وظیفه نیست. با پذیرش این روش، دیگر نیازی به تنظیم دقیق فردی یک مدل برای هر وظیفه نیست. در عوض، یک لایه آداتور واحد برای هر وظیفه تنظیم دقیق شود و اجازه می‌دهد تا پرسش‌ها به طور کارآمد پاسخ‌های مورد نظر را تولید کنند.

<sup>۹۹</sup> add weighted adapter function :

[https://huggingface.co/docs/peft/main/en/package\\_reference/lora#peft.LoraModel.add\\_weighted\\_adapter](https://huggingface.co/docs/peft/main/en/package_reference/lora#peft.LoraModel.add_weighted_adapter)



شکل ۶.۶: نمای کلی از نحوه استفاده از چندین آداتپتور با یک LLM پیشآموزش دیده برای تنظیم دقیق آن برای وظایف خاص مختلف، مانند خلاصه‌سازی، ویرایش، تحلیل احساسات و غیره. (منبع: [۶۷])

### مراحل تنظیم دقیق LLM با استفاده از LoRA برای وظایف و آداتپتورهای متعدد

۱. ایجاد آداتپتور: چندین آداتپتور ایجاد کنید که هر کدام برای وظایف خاصی با استفاده از قالب‌های متفاوت پرامپت یا تگ‌های شناسایی وظیفه (مانند [chat]، [translate fren]، [translate]) بهینه‌سازی شده‌اند.
۲. یکپارچه‌سازی LoRA: LoRA را برای یکپارچه‌سازی کارآمد این آداتپتورها در LLM پیشآموزش دیده پیاده‌سازی کنید. از روش‌های LORA مانند الحق، ترکیب خطی یا تجزیه مقادیر منفرد (SVD) برای ترکیب آداتپورها استفاده کنید در حالی که بار محاسباتی را به حداقل می‌رسانید و عملکرد را حفظ می‌کنید.

۳. **تطبیق وظیفه خاص**<sup>۱۰۰</sup>: هر آدپتور را با داده‌های وظیفه خاص<sup>۱۰۱</sup> تنظیم دقیق کنید تا عملکرد را برای وظایف فردی بهبود ببخشد. اطمینان حاصل کنید که آدپتورها با داده‌های مربوط به وظایف خود آموزش می‌بینند و توانایی آن‌ها برای تولید پاسخ‌های دقیق بهینه می‌شود.

۴. **تنظیم رفتار**: رفتار آدپتورهای ترکیب شده را تحت نظر داشته باشید تا هر گونه رفتار نامطلوب به ارث رسیده از آدپتورهای فردی را شناسایی کنید (مانند تولید پاسخ‌های کوتاه از یک آدپتور ترجمه). وزن‌ها یا نوع‌های ترکیب را تنظیم کنید تا رفتار آدپتور را در صورت لزوم تغییر دهید و اطمینان حاصل کنید که هر آدپتور به طور بهینه برای وظیفه مورد نظر خود عمل می‌کند.

۵. **ارزیابی و تکرار**: عملکرد مدل ترکیبی را در چندین وظیفه با استفاده از مجموعه‌های داده اعتبارسنجی ارزیابی کنید. فرآیند تنظیم دقیق را تکرار کرده و تنظیمات ترکیب آدپتورها و پارامترهای آموزشی را بر اساس معیارهای عملکرد و بازخورد کاربران انجام دهید.

بنابراین، برای عملکرد بهینه، توصیه می‌شود آدپتورها را که با فرمت پرامپت‌های کاملاً متفاوت<sup>۱۰۲</sup> تنظیم دقیق شده‌اند، ترکیب کنید. با این حال، حتی هنگام استفاده از آدپتورهای با فرمت پرامپت‌های مختلف، آدپتور حاصل ممکن است رفتار مطلوبی از خود نشان ندهد. به عنوان مثال، یک آدپتور جدید ترکیب شده که برای چت طراحی شده است، ممکن است تنها پاسخ‌های کوتاهی تولید کند و این تمایل را از آدپتوری که به‌طور اصلی برای متوقف شدن پس از تولید یک جمله آموزش دیده بود، به ارت ببرد. برای تنظیم رفتار آدپتور ترکیب شده، می‌توان تأثیر یک آدپتور خاص را در حین فرآیند ترکیب اولویت‌بندی کرد و یا روش ترکیبی مورد استفاده را تغییر داد. یک راهنمای بصری که نحوه تنظیم دقیق مدل‌های زبان بزرگ (LLMs) با استفاده از لایه‌های آدپتور متعدد برای وظایف مختلف را نشان می‌دهد، در [اینجا](#)<sup>۱۰۳</sup> موجود است.

## شش – چهار: تنظیم دقیق نیمه

تنظیم دقیق نیمه<sup>۱۰۴</sup> (HFT) [۶۸] یک تکنیک است که به منظور تعادل بین حفظ دانش بنیادی و کسب مهارت‌های جدید در مدل‌های زبان بزرگ (LLMs) طراحی شده است. HFT شامل ثابت نگهداشت نیمی از پارامترهای مدل در هر دور تنظیم دقیق در حالی که نیمه دیگر را به‌روزرسانی می‌کند، می‌باشد و به مدل این امکان را می‌دهد که دانش پیش‌آموزش دیده را حفظ کرده و عملکرد وظایف جدید را بدون تغییر معماری مدل بهبود بخشد. هر لایه ترنسفورمر تکراری به سه بلوک تقسیم می‌شود: توجه خودکار، پیش‌روند و نرمال‌سازی لایه، به‌طوری که نیمی از پارامترها در هر بلوک به‌روزرسانی شده و نیمی دیگر ثابت می‌مانند که با هر دور متفاوت

<sup>100</sup> Task-Specific Adaptation

<sup>101</sup> Task-Specific Data

<sup>102</sup> distinctly varied prompt formats

<sup>103</sup> <https://newsletter.kaitchup.com/p/combine-multiple-lora-adapters-for>

<sup>104</sup> Half Fine-Tuning

است. این به روزرسانی پارامتر استراتژیک به حفظ تعادل دانش در طول دورهای آموزشی کمک می‌کند و مقیاس‌پذیری را در جلسات آموزشی متوالی بهبود می‌بخشد.

تحقیقات در مورد مدل‌هایی مانند LLAMA 2-7B نشان داد که HFT می‌تواند به طور قابل توجهی دانش بنیادی فراموش شده را بازیابی کند در حالی که عملکرد کلی بالا را حفظ می‌کند. robustness و کارآمدی این روش آن را به طور وسیع در سناریوهای مختلف تنظیم دقیق، از جمله تنظیم دقیق تحت نظرارت، بهینه‌سازی مستقیم ترجیحات و یادگیری مداوم، قابل اجرا می‌کند. علاوه بر این، قابلیت HFT در حفظ معماری مدل، پیاده‌سازی آن را ساده کرده و سازگاری با سیستم‌های موجود را تضمین می‌کند و به این ترتیب، پذیرش عملی آن را ترویج می‌دهد.

### شش - چهار - یک: مزایای استفاده از تنظیم دقیق نیمه

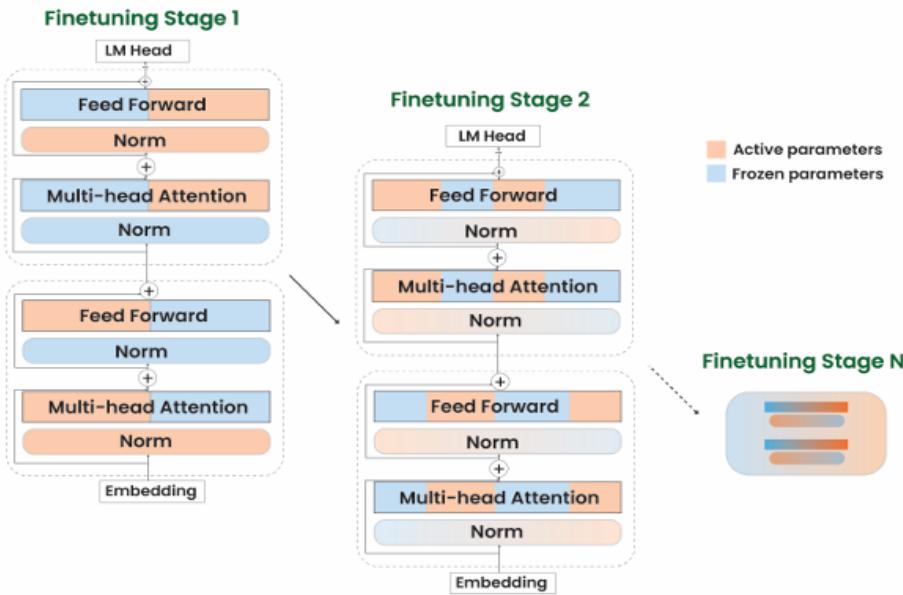
۱. بازیابی دانش پیش‌آموزش دیده: با بازگشت نیمی از پارامترهای تنظیم دقیق شده به وضعیت پیش‌آموزش شده، HFT به طور مؤثری بخشی از دانش اصلی را بازیابی می‌کند و در نتیجه فراموشی فاجعه‌بار قابلیت‌های قبل‌اً کسب شده را کاهش می‌دهد.

۲. عملکرد بهبود یافته: آزمایش‌های تحقیقاتی نشان می‌دهد که HFT عملکرد را در وظایف پایین‌دستی حفظ کرده یا حتی از عملکرد تنظیم دقیق کامل (FFT) پیشی می‌گیرد و اثربخشی آن را در تعادل بین حفظ دانش و یادگیری خاص وظیفه نشان می‌دهد.

۳. استحکام: این روش در برابر استراتژی‌های انتخاب مختلف و تعداد پارامترهای انتخاب شده برای به روزرسانی مقاوم است و عملکرد ثابت را در پیکربندی‌های مختلف تضمین می‌کند.

۴. سادگی و مقیاس‌پذیری: HFT معماری مدل را تغییر نمی‌دهد، که پیاده‌سازی را ساده کرده و اجازه می‌دهد تا کاربردهای مقیاس‌پذیر داشته باشد که به ویژه در سناریوهای تنظیم دقیق متوالی مفید است.

۵. چندمنظوره بودن: این تکنیک در سناریوهای مختلف تنظیم دقیق از جمله تنظیم دقیق تحت نظرارت، بهینه‌سازی مستقیم ترجیحات و یادگیری مداوم مؤثر بوده است.



شکل ۷.۶: نمودار شماتیک روش تنظیم دقیق نیمه (HFT) که بر روی معماری 2 Llama 2 اعمال شده است. این نمودار مراحل مختلف تنظیم دقیق را نشان می‌دهد که در آن پارامترهای خاص مدل به طور انتخابی فعال شده‌اند (نارنجی) در حالی که سایر پارامترها ثابت می‌مانند (آبی). این رویکرد با کاهش نیازهای محاسباتی، آموزش را بهینه می‌کند و در عین حال مدل را به طور مؤثری به وظایف یا داده‌های جدید سازگار می‌کند. (منبع: [۶۸])

## ۶-۴-۲- مقایسه بین HFT و LoRA

معیار	HFT	LoRA
هدف	هدف، حفظ دانش پایه‌ای کسب شده در طول پیش‌آموزش و یادگیری مهارت‌های جدید خاص وظایف است، که تعادلی بین حفظ قابلیت‌های موجود و کسب قابلیت‌های جدید ایجاد می‌کند.	LoRA هدف دارد نیازهای محاسباتی و حافظه‌ای را در هنگام تنظیم دقیق کاهش دهد، و این کار باعث می‌شود که آموزش مدل‌های بزرگ بر روی منابع سخت‌افزاری محدود، امکان‌پذیرتر و کارآمدتر باشد.
رویکرد	HFT نیمی از پارامترهای مدل را در هر دور تنظیم دقیق ثابت نگه می‌دارد و فقط نیمه دیگر را به روزرسانی می‌کند.	LoRA تعداد پارامترهای قابل آموزش را با معرفی تجزیه کمرتبه در ماتریس‌های وزنی شبکه عصبی کاهش می‌دهد. این روش شامل تزریق ماتریس‌های کمرتبه در لایه‌های مدل در هنگام تنظیم دقیق است.
معماری مدل	HFT معما ری مدل را تغییر نمی‌دهد و پارامترهای جدیدی اضافه نمی‌کند، که این امر اعمال آن را بدون نیاز به تغییرات ساختاری اضافی ساده می‌سازد.	LoRA مدل را با افزودن ماتریس‌های کمرتبه تغییر می‌دهد، که پویایی آموزش را تغییر داده و نیاز به محاسبات اضافی برای به روزرسانی‌های کمرتبه دارد.
عملکرد	تحقیقات نشان داده است که HFT می‌تواند دانش پایه فراموش شده را بازیابی کرده و عملکرد بالایی در توانایی‌های عمومی حفظ کند.	LoRA به گونه‌ای طراحی شده که عملکرد رقابتی با تنظیم دقیق کامل داشته باشد، اما با تعداد پارامترهای قابل آموزش کمتر و هزینه محاسباتی پایین‌تر.

## شش – پنج: تنظیم حافظه لامینی

لامینی (Lamini) یک روش تخصصی برای تنظیم دقیق مدل‌های زبانی بزرگ (LLMs) است که با هدف کاهش توهمنات<sup>۱۰۵</sup> معرفی شده است. انگیزه توسعه این روش، نیاز به بهبود دقت و قابلیت اطمینان LLM‌ها در حوزه‌هایی بود که به بازیابی اطلاعات دقیق نیاز دارند. روش‌های سنتی آموزش عموماً شامل اجرای گرادیان کاهشی تصادفی بر روی داده‌های بزرگ هستند که علی‌رغم توانایی در تطبیق با داده‌های آموزشی، اغلب منجر به تولید مدل‌هایی می‌شوند که توانایی تعمیم‌دهی مناسبی ندارند و به خطاهایی همچون توهمنات دچار می‌شوند.

مدل‌های پایه معمولاً از یک برنامه آموزشی مشابه دستورالعمل‌های چینچیلا<sup>۱۰۶</sup> پیروی می‌کنند که آموزش بر روی یک گرپوس عظیم<sup>۱۰۷</sup> برای یک ایپاک کامل را تجویز می‌کند، مثلاً آموزش مدل Llama 2 7B بر روی حدود یک تریلیون توکن. این رویکرد منجر به کاهش قابل توجهی در خطا می‌شود و به بهبود تعمیم‌دهی و خلاقیت کمک می‌کند، جایی که درجه‌ای از تصادفی بودن در انتخاب توکن‌ها مجاز است. با این حال، این روش برای وظایفی که به دقت واقعی نیاز دارد کافی نیست. در مقابل، تنظیم حافظه لامینی به طور عمیق‌تری به تحلیل خطای فکت‌های منفرد می‌پردازد و دقت یادآوری این فکت‌ها را به طور چشمگیری بهبود می‌بخشد.

با افزودن پارامترهای اضافی برای حافظه به مدل (برای مثال، مدل ۸ میلیارد پارامتری با اضافه کردن ۲ میلیارد پارامتر برای وزن‌ها)، لامینی به مدل امکان می‌دهد که تعداد زیادی از فکت‌ها را به خوبی به خاطر بسپارد و دقیقاً بازگو کند، و عملکردی نزدیک به قوانین مقیاس‌گذاری LLM‌ها داشته باشد، بدون اینکه به تعمیم‌دهی مدل آسیبی وارد شود.

### شش – پنج – یک: لامینی-۱ – یک معماری مدل مبتنی بر لامینی

مدل لامینی-۱ با معماری متفاوت از طرح‌های سنتی مبتنی بر ترانسفورمر، از یک سیستم ترکیبی گسترده از متخصصان حافظه<sup>۱۰۸</sup> (MoME) بهره می‌برد. این سیستم شامل یک ترانسفورمر پیش‌آموزش‌دیده است که با آداتورهایی تقویت می‌شود که به صورت پویا و با استفاده از مکانیزم‌های توجه متقاطع<sup>۱۰۹</sup> از یک شاخص انتخاب می‌شوند. این آداتورها مشابه متخصصان در معماری‌های MoE عمل می‌کنند و شبکه به صورت کاملاً انتها به انتها آموزش داده می‌شود، در حالی که ستون فقرات اصلی آن ثابت می‌ماند. این ساختار امکان ذخیره دقیق فکت‌ها در متخصصان منتخب<sup>۱۱۰</sup> را فراهم می‌کند.

<sup>105</sup> Hallucinations

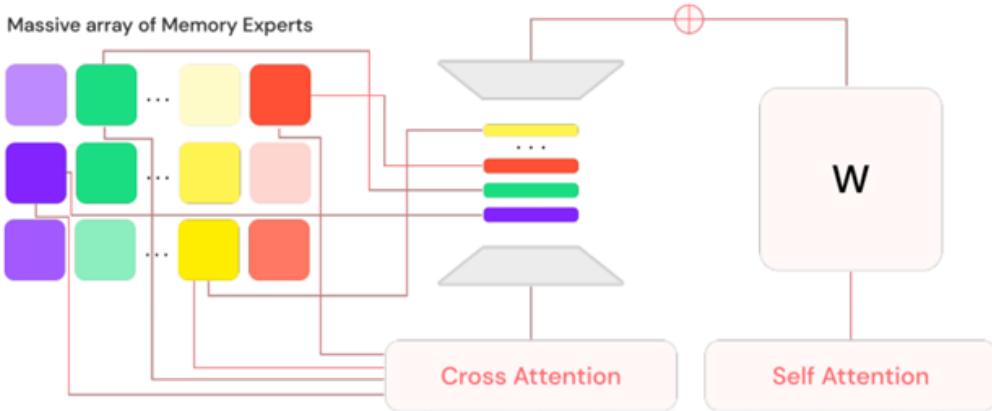
<sup>106</sup> Chinchilla Recipe

<sup>107</sup> Massive Corpus

<sup>108</sup> Massive Mixture of Memory Experts

<sup>109</sup> Cross-Attention Mechanisms

<sup>110</sup> Selected Experts



شکل ۶.۸: نمودار معماری مدل لامینی-۱ که شامل آرایه عظیمی از متخصصان حافظه (MoME) است. این معماری یک ستون فقرات ترانسفورمر پیش‌آموزش دیده را با آداتورهایی که به طور پویا از طریق مکانیزم‌های توجه متقطع انتخاب می‌شوند، ادغام می‌کند. هر آداتور به عنوان یک متخصص حافظه عمل کرده و قادر است داده‌های خاص واقعی را ذخیره کند. (اقتباس از [۶۹])

در زمان استنتاج، تنها متخصصان مرتبط از شاخص بازیابی می‌شوند و به مدل LLM امکان می‌دهد تعداد زیادی از فکت‌ها را ذخیره کند و در عین حال تأخیر استنتاج را پایین نگه دارد. هسته‌های پردازشی تخصصی GPU که با Triton نوشته شده‌اند، برای تسريع در بازیابی متخصصان استفاده می‌شوند و سیستم را برای دسترسی سریع به دانش ذخیره‌شده بهینه می‌کنند.

### بهینه‌سازی‌های سیستم برای رفع توهمات

معماری MoME به گونه‌ای طراحی شده است که تقاضای محاسباتی لازم برای به خاطر سپردن حقایق را به حداقل می‌رساند. در طول آموزش، برای هر واقعیت، یک زیرمجموعه از متخصصان، مثلاً ۳۲ از یک میلیون، انتخاب می‌شوند. وزن‌های شبکه اصلی و توجه متقطع که برای انتخاب متخصص استفاده می‌شود، ثابت می‌مانند و گام‌های گرادیان کاهشی برداشته می‌شود تا خطأ به حدی کاهش یابد که واقعیت را به حافظه بسپارد. این رویکرد، از انتخاب چندباره یک متخصص برای فکت‌های مختلف جلوگیری می‌کند؛ ابتدا مکانیزم انتخاب توجه متقطع در طی مرحله آموزش تعمیم آموزش داده می‌شود و سپس وزن‌های آن ثابت می‌شوند.

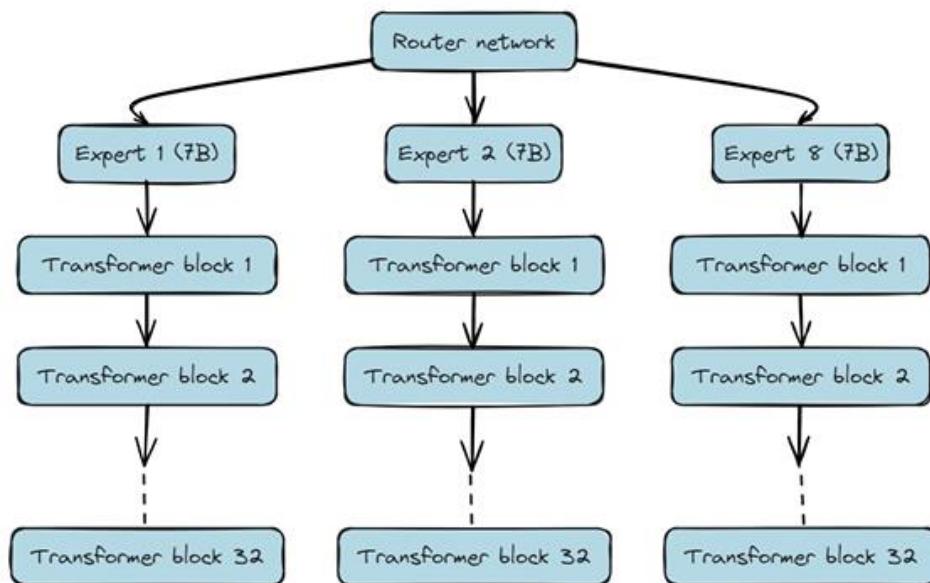
این روش تضمین می‌کند که محاسبات با تعداد مثال‌های آموزشی متناسب است، نه تعداد کل پارامترها، و در نتیجه محاسبات لازم برای تنظیم حافظه را به طور قابل توجهی کاهش می‌دهد. این روش بهینه‌سازی شده به لامینی-۱ اجازه می‌دهد که در تنظیم حافظه بر روی پاسخ‌های واقعی و تصادفی، به طور کارآمد به حداقل خطأ دست یابد و اثربخشی خود را در حذف توهمات و بهبود یادآوری واقعی نشان دهد.

## شش - شش ترکیب متخصصان

ترکیب متخصصان<sup>۱۱۱</sup> (MoE) یک طراحی معماری برای شبکه‌های عصبی است که محاسبات یک لایه یا عملیات (مانند لایه‌های خطی، یا MLP‌ها، یا پروجکشن توجه) را به چندین زیرشبکه متخصصی تقسیم می‌کند که به عنوان "متخصصان" شناخته می‌شوند. هر متخصص محاسبات خود را به طور مستقل انجام می‌دهد و نتایج برای تولید خروجی نهایی لایه MoE ترکیب می‌شوند. معماری‌های MoE می‌توانند به صورت متراکم (dense) باشند، که در آن هر متخصص برای هر ورودی به کار گرفته می‌شود، یا پراکنده (sparse)، که در آن فقط یک زیرمجموعه از متخصصان برای هر ورودی استفاده می‌شود.

### شش - شش - یک: معماری و عملکرد Mixtral 8x7B

Mixtral [70] 8x7B از معماری SMoE<sup>۱۱۲</sup> بهره می‌برد (شکل ۶.۹) که ساختار 7B را بازتاب می‌دهد اما شامل هشت بلوک پیش‌رونده (متخصصان) در هر لایه است. برای هر توکن در هر لایه، یک شبکه مسیریاب دو متخصص را برای پردازش وضعیت فعلی انتخاب کرده و خروجی‌های آن‌ها را ترکیب می‌کند. اگرچه هر توکن در یک زمان فقط با دو متخصص تعامل دارد، متخصصان انتخاب شده می‌توانند در هر گام زمانی متفاوت باشند. بنابراین، هر توکن به ۴۷ میلیارد پارامتر دسترسی دارد، اما در طول استنتاج تنها از ۱۳ میلیارد پارامتر فعال استفاده می‌کند. Mixtral 8x7B نه تنها با Llama 2 70B و GPT-3.5 برابری می‌کند بلکه اغلب در تمام معیارهای ارزیابی از آن‌ها پیشی می‌گیرد. عملکرد آن به ویژه در ریاضیات، تولید کد، و وظایف چندزبانه بهتر از Llama 2 70B است.



<sup>۱۱۱</sup> Mixture of Experts (MoE)

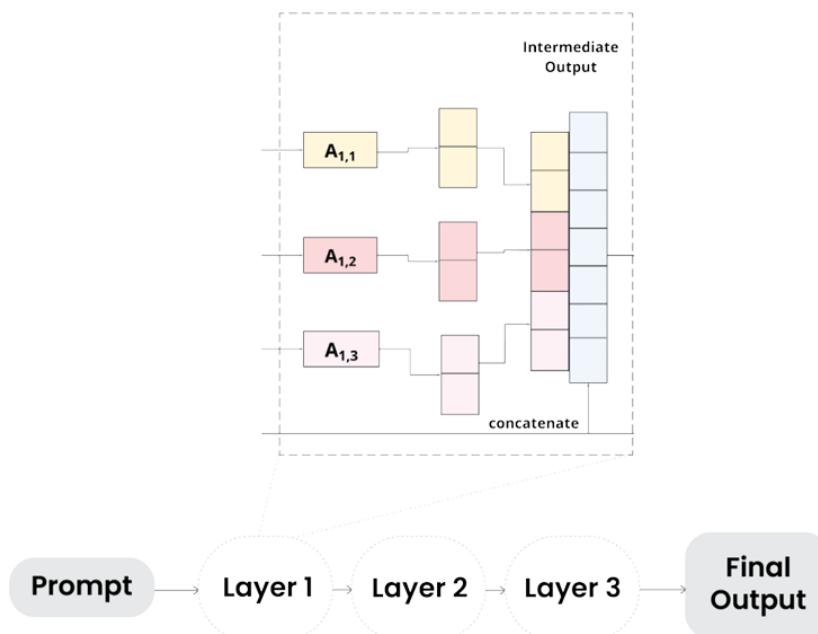
<sup>۱۱۲</sup> Sparse Mixture of Experts (SMoE)

شکل ۶.۹: نمودار معماری مدل Mixtral 8x7B با ترکیب متخصصان (MoE). این مدل شامل یک شبکه مسیریاب است که به طور پویا، مرتبط‌ترین متخصصان را از میان هشت متخصص مبتنی بر ترانسفورمر انتخاب می‌کند که هر کدام ۷ میلیارد پارامتر دارند. این متخصصان در بلوک‌های ترانسفورمر سازمان‌دهی شده‌اند، و مسیریاب داده‌ها را بر اساس ورودی به متخصص مناسب هدایت می‌کند و در نتیجه کارایی محاسباتی و عملکرد مدل را بهینه می‌سازد. این معماری امکان مقیاس‌پذیری و پردازش تخصصی در مدل‌های زبان بزرگ را فراهم می‌کند. (اقتباس از [۷۱])<sup>۱۱۳</sup>

## شش – هفت: ترکیب عامل‌ها

با وجود مدل‌های زبان بزرگ (LLM) متعدد و دستاوردهای قابل توجه آن‌ها، این مدل‌ها همچنان با محدودیت‌های اساسی در زمینه اندازه مدل و داده‌های آموزشی مواجه‌اند. گسترش این مدل‌ها هزینه‌های گرافی دارد و اغلب نیازمند آموزش دوباره با تریلیون‌ها توکن است. در عین حال، مدل‌های LLM مختلف توانمندی‌ها و تخصص‌های متفاوتی در جنبه‌های مختلف وظایف دارند. یک مطالعه اخیر، استفاده از تخصص جمعی چندین مدل LLM را برای توسعه یک مدل توانمندتر و پایدارتر بررسی کرده که این روش با عنوان ترکیب عامل‌ها<sup>۱۱۴</sup> (MoA) شناخته می‌شود [۷۲].

شکل ۶.۱۰. با استفاده از یک معماری لایه‌ای عمل می‌کند، به‌طوری‌که هر لایه شامل چندین عامل LLM است (شکل ۶.۱۰). این ساختار پدیدهای به نام «همکاری LLM‌ها» را آشکار می‌سازد. چارچوب نواورانه MoA از قابلیت‌های ترکیبی چند مدل LLM بهره می‌برد تا مهارت‌های استدلال و تولید زبان را بهبود بخشد. تحقیقات نشان می‌دهد که LLM‌ها به‌طور طبیعی با هم همکاری می‌کنند و کیفیت پاسخ‌دهی بهتری از خود نشان می‌دهند، حتی اگر خروجی‌های سایر مدل‌ها کامل نباشد.



<sup>۱۱۳</sup> Mixture of Agents

شکل ۶.۱۰: تصویر پیکربندی Mixture of Agents (MoA) در مدل‌های زبان بزرگ. این مدل از چندین لایه تشکیل شده است که هر لایه شامل چندین عامل است که به‌طور مستقل ورودی را پردازش کرده و سپس خروجی‌های خود را به هم می‌پیوندند تا یک نتیجه میانجی ایجاد کنند. این فرایند در لایه‌های مختلف ادامه پیدا می‌کند و در هر مرحله خروجی بهینه می‌شود تا در نهایت خروجی نهایی بر اساس ورودی مورد نظر تولید شود. (اقتباس از [۷۲])

### شش - هفت - یک: روش‌شناسی

برای بهبود همکاری میان چندین مدل LLM، شناخت نقاط قوت هر یک از آن‌ها و طبقه‌بندی مناسب آن‌ها ضروری است. این طبقه‌بندی شامل موارد زیر است:

۱. پیشنهادهندگان: این مدل‌ها در تولید پاسخ‌های مرجع ارزشمند برای سایر مدل‌ها مهارت دارند. اگرچه به تهایی ممکن است عملکرد بالایی نداشته باشند، اما با ارائه دیدگاه‌های مختلف و ایجاد زمینه مناسب، به بهبود خروجی نهایی در ترکیب با یک جمع‌کننده کمک می‌کنند.

۲. جمع‌کنندگان: این مدل‌ها در ترکیب پاسخ‌ها از مدل‌های مختلف برای ایجاد یک نتیجه با کیفیت بالا مهارت دارند. یک جمع‌کننده مؤثر باید کیفیت پاسخ نهایی را حفظ کرده یا حتی بهبود بخشد، صرف‌نظر از کیفیت ورودی‌های فردی.

انتخاب دقیق مدل‌ها برای هر لایه MoA بسیار مهم است. معیارهای عملکرد، مانند میانگین نرخ برتری در یک لایه معین، به ارزیابی مناسب بودن مدل‌ها برای لایه‌های بعدی کمک می‌کنند و تولید خروجی‌های با کیفیت بالاتر را تضمین می‌کنند. تنوع در خروجی‌های مدل ضروری است، زیرا پاسخ‌های متنوع از مدل‌های مختلف به‌طور قابل توجهی بهتر از خروجی‌های مشابه از یک مدل واحد هستند. در MoA، برای یک ورودی مشخص، خروجی لایه ۱ ام MoA به صورت زیر محاسبه می‌شود:

$$y_i = \bigoplus_{j=1}^n [A_{i,j}(x_i)] + x_1, \quad x_{i+1} = y_i \quad (6.1)$$

### شش - هفت - دو: شباهت با MoE

ترکیب متخصصان (MoE) یک تکنیک ماشین یادگیری شناخته شده است که در آن چندین شبکه متخصص با مهارت‌های خاص به حل مسائل پیچیده می‌پردازند. این روش در کاربردهای مختلف موفقیت‌های قابل توجهی داشته و به عنوان منبع الهام روش ترکیب عامل‌ها (MoA) محسوب می‌شود. در طراحی معمول MoE، یک پشتۀ از لایه‌ها که به عنوان لایه‌های MoE شناخته می‌شوند، شامل چندین شبکه متخصص، یک شبکه گیت، و اتصالات باقیمانده برای بهبود جریان گرادیان است. خروجی لایه \(\text{O}\_1\) به صورت زیر محاسبه می‌شود:

$$y_i = \sum_{j=1}^n G_{i,j}(x_i) E_{i,j}(x_i) + x_i \quad (6.2)$$

چارچوب MoA مفهوم را با استفاده از تعاملات مبتنی بر ورودی به جای تغییرات داخلی فعال سازی یا وزن ها، در سطح مدل گسترش می دهد. به جای اتکا به شبکه های زیرشاخه تخصصی در یک مدل واحد، MoA از چندین مدل کامل LLM در لایه های مختلف بهره می برد. در این رویکرد، عملکرد شبکه های گیت و متخصص در یک مدل LLM یکپارچه شده و از توانایی آن برای تفسیر ورودی ها و تولید خروجی های هماهنگ بدون نیاز به مکانیزم های هماهنگی اضافی استفاده می کند.

### شش - هفت - سه: چرا MoA عملکرد خوبی دارد؟

- عملکرد برتر MoA:** MoA به طور قابل توجهی بهتر از رتبه بندی کننده های مبتنی بر LLM عمل می کند، که به جای تولید پاسخ جدید، یک پاسخ را از میان پیشنهادها انتخاب می کند. این نشان می دهد که رویکرد MoA در تجمعی تمامی پاسخ های تولید شده نتایج موثر تری نسبت به انتخاب از گزینه های موجود دارد.
- ادغام موثر پیشنهادات:** جمع کننده در MoA تمایل به ادغام بهترین پاسخ های پیشنهاد شده دارد. این امر با همبستگی مثبت بین پاسخ های جمع کننده و معیارهای مختلف شباهت، مانند نمره های BLEU که همپوشانی n-gram ها را اندازه گیری می کنند، پشتیبانی می شود. استفاده از معیارهای جایگزین شباهت نیز همبستگی مثبت با امتیاز های ترجیحی نشان می دهد که جمع کننده به طور موثر از پاسخ های پیشنهادی استفاده می کند.
- تأثیر تنوع مدل و تعداد پیشنهاد دهنده ها:** افزایش تعداد پیشنهاد دهنده ها کیفیت خروجی را بهبود می بخشد و مزایای اطلاعات جانبی بیشتر را نشان می دهد. علاوه بر این، استفاده از مجموعه ای متنوع از مدل های LLM به عنوان پیشنهاد دهنده ها به طور مداوم نتایج بهتری نسبت به استفاده از یک مدل واحد دارد. این موضوع نشان می دهد که تعداد و تنوع عامل های LLM در هر لایه MoA به بهبود عملکرد کمک می کند و امکان بهبود بیشتر از طریق مقیاس پذیری وجود دارد.
- تخصص مدل:** تحلیل نقش مدل ها در اکوسیستم MoA نشان می دهد که مدل های Qwen، GPT-40 و LLaMA-3 در هر دو وظیفه کمک و جمع بندی عملکرد خوبی دارند. در مقابل، مدل WizardLM در پیشنهاد دهنده موفق است اما در جمع بندی پاسخ های سایر مدل ها با مشکل مواجه می شود.

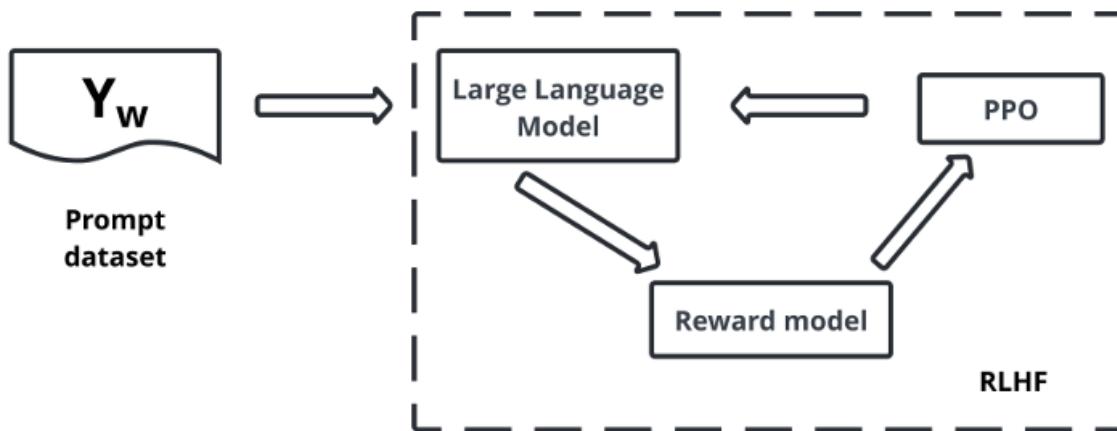
### شش - هشت: بهینه سازی سیاست پروگزیمال (PPO)

PPO<sup>114</sup> [۷۳] یک الگوریتم تقویتی شناخته شده است که برای آموزش عامل ها جهت انجام وظایف در محیط های مختلف استفاده می شود. این الگوریتم از روش های گرادیان سیاست استفاده می کند، که در آن سیاست ها - که با شبکه های عصبی نمایش داده می شوند - اقدام های عامل را بر اساس وضعیت کنونی تعیین

<sup>114</sup> Proximal Policy Optimisation

می‌کنند. PPO به طور موثر با ماهیت پویای داده‌های آموزشی که از طریق تعاملات پیوسته عامل-محیط تولید می‌شوند، تطبیق می‌یابد؛ ویژگی‌ای که آن را از مجموعه داده‌های ایستا در یادگیری نظارت شده متمایز می‌کند. نوآوری PPO در تابع هدف "جایگزین" آن است که از طریق صعود گرادیان تصادفی بهینه‌سازی می‌شود. این رویکرد امکان به روزرسانی‌های متعدد از همان مجموعه داده‌ها را فراهم می‌کند و هم کارایی آموزشی و هم پایداری را نسبت به روش‌های گرادیان سیاست سنتی بهبود می‌بخشد. این الگوریتم توسط OpenAI توسعه یافته و برای ایجاد تعادلی میان سادگی پیاده‌سازی و عملکرد قوی‌تر از الگوریتم‌های پیچیده‌تر مانند بهینه‌سازی سیاست منطقه‌ای قابل اعتماد (TRPO) طراحی شده است، اما بدون پیچیدگی محاسباتی مرتبط با آن.

PPO با به حداکثر رساندن پاداش‌های تجمعی مورد انتظار از طریق تنظیمات مکرر سیاست، که احتمال اقداماتی که به پاداش‌های بالاتر منجر می‌شوند را افزایش می‌دهد، عمل می‌کند. یکی از ویژگی‌های کلیدی PPO، استفاده از یک مکانیسم قطع در تابع هدف است که میزان به روزرسانی‌های سیاست را محدود می‌کند و از تغییرات شدید جلوگیری کرده و پایداری را در طول آموزش حفظ می‌کند.



شکل ۱۱.۶: شماتیک به کارگیری بهینه‌سازی سیاست تقریبی (PPO) در زمینه یادگیری تقویتی از بازخورد انسانی (RLHF) برای تنظیم دقیق یک مدل زبانی بزرگ (LLM). فرآیند شامل استفاده از یک مجموعه داده‌ی پرسشی برای آموزش مدل زبانی است. الگوریتم PPO سیاست مدل زبانی را براساس پاداش‌هایی که از مدل پاداش دریافت می‌شود، تنظیم می‌کند؛ این مدل پاداش از طریق بازخورد انسانی بهبود یافته است. (اقتباس از [۷۳])

## کتابخانه‌ی پایتون – پکیج HuggingFace Transformer Reinforcement Learning (TRL<sup>115</sup>)

از ترینر<sup>116</sup> PPO پشتیبانی می‌کند تا مدل‌های زبانی را با استفاده از داده‌های ترجیحی آموزش دهد. ترینر PPO باید پاسخ تولید شده را با یک پرسش، براساس پاداش‌های به دست آمده از مدل پاداش، هماهنگ کند. در هر مرحله از الگوریتم PPO، دسته‌ای از پرسش‌ها از مجموعه داده نمونه‌برداری می‌شود و سپس از این پرسش‌ها برای تولید پاسخ از مدل SFT استفاده می‌گردد. در مرحله‌ی بعد، مدل پاداش برای محاسبه‌ی پادash‌های پاسخ تولید شده به

<sup>115</sup> <https://huggingface.co/docs/trl/en/index>

<sup>116</sup> Trainer : [https://huggingface.co/docs/trl/main/en/ppo\\_trainer](https://huggingface.co/docs/trl/main/en/ppo_trainer)

کار گرفته می شود. در نهایت، این پاداش ها برای بهینه سازی مدل SFT با استفاده از الگوریتم PPO مورد استفاده قرار می گیرند. بنابراین، مجموعه داده باید یک ستون متنی داشته باشد که بتوان آن را به نام پرسش تغییر نام داد. هر یک از سایر داده های موردنیاز برای بهینه سازی مدل SFT در طول حلقه ای آموزشی به دست می آید.

### شش - هشت - یک: مزایای PPO

۱. پایداری: بهینه سازی سیاست تقریبی (PPO) برای اطمینان از بهروزرسانی های پایدار و قابل اعتماد سیاست طراحی شده است. تابع هدف تقریبی برش خورده، محور این پایداری است، زیرا بهروزرسانی های سیاست را محدود می کند تا از تغییرات بزرگ و احتمالاً ناپایدار جلوگیری کند. این ویژگی منجر به یادگیری روان تر و پایدارتر می شود.

۲. سادگی پیاده سازی: در مقایسه با الگوریتم های پیشرفته مانند TRPO، PPO پیاده سازی نسبتاً ساده ای دارد. این الگوریتم نیازی به تکنیک های بهینه سازی مرتبه دوم ندارد و برای کاربران کم تجربه تر قابل دسترسی تر است.

۳. بهره وری نمونه: PPO با استفاده از تابع هدف تقریبی برش خورده به بهره وری داده می رسد. این مکانیسم بهروزرسانی های سیاست را تنظیم می کند و در حالی که پایداری را حفظ می کند، به طور مؤثر از داده های آموزشی استفاده مجدد می کند. بنابراین، PPO نسبت به سایر الگوریتم های یادگیری تقویتی نمونه ها را با کارآیی بیشتری مصرف می کند و با تعداد کمتری از نمونه ها عملکرد خوبی دارد، که این ویژگی در سناریوهایی که جمع آوری داده ها هزینه بر یا زمان بر است مفید است.

### شش - هشت - دو: محدودیت های PPO

۱. پیچیدگی و هزینه محاسباتی: بهینه سازی سیاست تقریبی (PPO) شامل شبکه های سیاست و ارزش پیچیده است که نیازمند منابع محاسباتی قابل توجهی برای آموزش هستند. این پیچیدگی اغلب به زمان های طولانی تر آموزش و افزایش هزینه های عملیاتی منجر می شود.

۲. حساسیت به ابر پارامترها: عملکرد PPO به شدت به چندین ابر پارامتر وابسته است، مانند دامنه برش، نرخ یادگیری، و عامل تخفیف. دستیابی به عملکرد بهینه نیازمند تنظیم دقیق این پارامترها است. تنظیمات نادرست ممکن است منجر به نتایج سیاست کمتر بهینه یا ناپایداری در طول فرآیند یادگیری شود.

۳. مشکلات پایداری و همگرایی: اگرچه PPO برای بهبود پایداری در مقایسه با روش های قبلی طراحی شده است، همچنان ممکن است در محیط های بسیار پویا یا پیچیده با مشکلات همگرایی مواجه شود. حفظ بهروزرسانی های پایدار سیاست همچنان چالشی قابل توجه است.

۴. وابستگی به سیگنال پاداش: اثربخشی PPO به شدت به یک سیگنال پاداش به خوبی تعریف شده برای هدایت فرآیند یادگیری وابسته است. در شرایطی که طراحی یک تابع پاداش مناسب دشوار یا عملی نیست، PPO ممکن است در دستیابی به نتایج موردنظر با مشکل مواجه شود.

## شش - هشت - سه: آموزشی برای مدل‌ها با استفاده از تکنیک PPO

آموزش برای تنظیم مدل GPT2 برای تولید نقدهای مثبت فیلم براساس مجموعه داده IMDB با استفاده از تکنیک PPO در [ینجا<sup>۱۱۷</sup>](#) قابل دسترسی است.

## شش - نه: بهینه‌سازی مستقیم ترجیحات (DPO)

بهینه‌سازی مستقیم ترجیحات <sup>۱۱۸</sup> (DPO) [۷۴] رویکردی ساده‌تر برای هماراستاسازی مدل‌های زبانی (LMs) با ترجیحات انسانی ارائه می‌دهد و پیچیدگی یادگیری تقویتی از بازخورد انسانی <sup>۱۱۹</sup> (RLHF) را دور می‌زند. مدل‌های زبانی بزرگ‌مقیاس و بدون نظارت دقیق، اغلب فاقد کنترل رفتاری دقیق هستند و این موضوع نیازمند روش‌هایی مانند RLHF است که مدل‌ها را با استفاده از بازخورد انسانی تنظیم دقیق می‌کنند. با این حال، RLHF پیچیده است و شامل ایجاد مدل‌های پاداش و تنظیم دقیق مدل‌ها برای حداکثرسازی پاداش‌های تخمینی می‌شود که می‌تواند ناپایدار و پرهزینه باشد. DPO این چالش‌ها را با بهینه‌سازی مستقیم مدل‌های زبانی با یک هدف طبقه‌بندی ساده که پاسخ‌ها را با ترجیحات انسانی هم راستا می‌کند، برطرف می‌کند. این رویکرد نیاز به مدل‌سازی پاداش صریح و تنظیم دقیق ابرپارامترهای گسترده را از بین می‌برد و پایداری و کارآیی را بهبود می‌بخشد. DPO رفتارهای مطلوب را با افزایش احتمال نسبی پاسخ‌های ترجیحی بهینه‌سازی می‌کند، درحالی که از وزن‌های اهمیتی پویا برای جلوگیری از انحراف مدل استفاده می‌کند. بنابراین، DPO فرآیند یادگیری ترجیحات را ساده می‌کند و آن را به روشی مؤثر برای آموزش مدل‌های زبانی برای تبعیت از ترجیحات انسانی تبدیل می‌کند.

کتابخانه پایتون - پکیج TRL در HuggingFace از ترینر DPO برای آموزش مدل‌های زبانی براساس داده‌های ترجیحی پشتیبانی می‌کند. فرآیند آموزش DPO نیازمند مجموعه داده‌ای با فرمتی بسیار خاص است. اگر از همتراز کننده پیش‌فرض داده DPODataCollatorWithPadding استفاده می‌کنید، شیء نهایی مجموعه داده شما باید شامل سه ورودی خاص باشد که باید به شرح زیر برچسب‌گذاری شوند:

- پرسش (Prompt)

- انتخاب شده (Chosen)

- رد شده (Rejected)

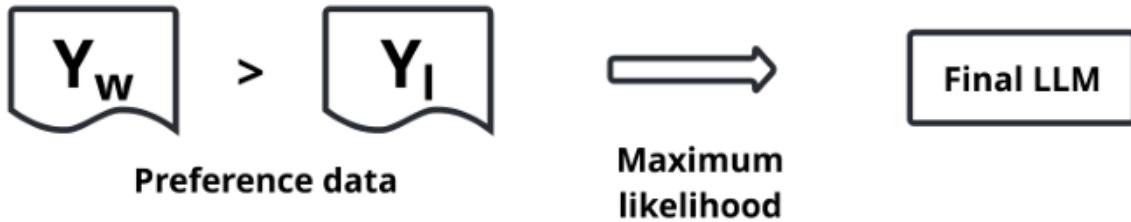
مجموعه داده‌های سازگار با DPO که توسط HuggingFace ارائه شده است از [ینجا<sup>۱۲۰</sup>](#) در دسترس می‌باشد.

<sup>117</sup> <https://github.com/huggingface/trl/blob/main/examples/notebooks/gpt2-sentiment.ipynb>

<sup>118</sup> Direct Preference Optimization

<sup>119</sup> Reinforcement Learning from Human Feedback

<sup>120</sup> <https://huggingface.co/datasets?other=dpo>



شکل ۶.۱۲: جریان فرآیند بهینه‌سازی مستقیم ترجیحات (DPO). این شکل تکنیک بهینه‌سازی مستقیم ترجیحات (DPO) را در فرآیند تنظیم دقیق مدل‌های زبانی بزرگ نشان می‌دهد. فرآیند با داده‌های ترجیحاتی ( $Y_W > Y_I$ ) شروع می‌شود، که در آن  $Y_W$  نشان‌دهنده خروجی‌های ترجیحی و  $Y_I$  نشان‌دهنده خروجی‌های کمارزش‌تر است. با استفاده از فرآیند برآورد حداکثر احتمال، این داده‌های ترجیحاتی برای بهینه‌سازی پارامترهای مدل استفاده می‌شوند و در نتیجه مدل زبانی بزرگ (LLM) نهایی حاصل می‌شود. این روش برای بهبود تطابق خروجی‌های مدل با ترجیحات کاربر طراحی شده است و کارایی مدل را در وظایف خاص افزایش می‌دهد. (اقتباس از [۷۴])

### شش - نه - یک: مزایای DPO

۱. تطابق مستقیم با ترجیحات انسانی: DPO مستقیماً مدل‌ها را برای تولید پاسخ‌هایی که با ترجیحات انسانی همخوانی دارند بهینه‌سازی می‌کند و در نتیجه خروجی‌های مطلوب‌تری تولید می‌کند.
۲. کاهش وابستگی به اهداف جایگزین: در مقابل روش‌هایی که بر پیش‌بینی کلمات بعدی تکیه می‌کنند، DPO از ترجیحات واضح انسانی بهره می‌برد و در نتیجه پاسخ‌هایی تولید می‌کند که بیشتر بازتاب‌دهنده رفتار انسانی هستند.
۳. بهبود عملکرد در وظایف ذهنی: برای وظایفی که به قضاوت‌های ذهنی نیاز دارند، مانند تولید مکالمه یا نوشتمن خلاقانه، DPO در تطابق مدل با ترجیحات انسانی موفق عمل می‌کند.

### شش - نه - دو: بهترین روش‌ها برای DPO

۱. داده‌های ترجیحاتی با کیفیت بالا: عملکرد مدل به شدت تحت تأثیر کیفیت داده‌های ترجیحاتی است. اطمینان حاصل کنید که مجموعه داده شامل ترجیحات انسانی واضح و ثابت است.
۲. مقدار بتا بهینه: با مقادیر مختلف بتا برای مدیریت تأثیر مدل مرجع آزمایش کنید. مقادیر بتای بالاتر ترجیحات مدل مرجع را قوی‌تر اولویت‌بندی می‌کنند.
۳. تنظیم ابرپارامترها: برای به دست آوردن بهترین تنظیمات برای مجموعه داده و وظیفه، ابرپارامترهایی مانند نرخ یادگیری، اندازه بسته و تنظیمات LoRA را بهینه کنید.
۴. ارزیابی در وظایف هدف: عملکرد مدل را به طور مستمر بر روی وظیفه هدف با استفاده از معیارهای مناسب ارزیابی کنید تا پیشرفت را نظارت کرده و از دستیابی به نتایج مطلوب اطمینان حاصل کنید.

**۵. ملاحظات اخلاقی:** به سوگیری‌های احتمالی در داده‌های ترجیحاتی توجه کنید و اقداماتی برای کاهش آن‌ها انجام دهید تا از پذیرش و تقویت این سوگیری‌ها توسط مدل جلوگیری شود.

### شش - نه - سه: آموزش مدل‌ها با تکنیک DPO

آموزش DPO، شامل کد کامل اسکریپت‌های آموزشی برای SFT و DPO، در [اینجا<sup>۱۲۱</sup>](#) در دسترس است.

### شش - نه - چهار: آیا PPO نسبت به DPO برابر با مدل‌های زبانی برتقی دارد؟

مطالعه‌ای جدید درباره DPO و برتقی آن نسبت به PPO در زمینه تطابق مدل‌های زبانی بزرگ (LLM) بررسی می‌کند که آیا روش‌های مبتنی بر پاداش و بدون پاداش در RLHF کارآمدتر هستند یا خیر. روش‌های مبتنی بر پاداش، مانند روش‌های توسعه‌یافته توسط OpenAI، از یک مدل پاداش که از داده‌های ترجیحاتی ساخته شده بهره می‌گیرند و الگوریتم‌های بازیگر-منتقد مانند بهینه‌سازی خطمنشی مجاور (PPO) را برای بهینه‌سازی سیگنال پاداش به کار می‌برند. در مقابل، روش‌های بدون پاداش، مانند بهینه‌سازی مستقیم ترجیحات (RRHF)، از توابع پاداش صریح صرف‌نظر کرده و بهینه‌سازی خطمنشی را از طریق یک نمایندگی لگاریتمی از تابع پاداش انجام می‌دهند.

یکی از اهداف این مطالعه تعیین این است که آیا DPO در حوزه RLHF واقعاً برتقی نسبت به PPO دارد. این مطالعه از تحلیل‌های نظری و تجربی برای کشف محدودیت‌های ذاتی DPO و شناسایی عوامل کلیدی که عملکرد عملی PPO را در RLHF بهبود می‌بخشد، استفاده می‌کند.

یافته‌های نظری نشان می‌دهند که DPO ممکن است با بهره‌گیری از پاسخ‌های خارج از توزیع، راه حل‌های جانبدارانه‌ای ارائه دهد. نتایج تجربی نشان می‌دهند که عملکرد DPO به طور قابل توجهی تحت تأثیر تغییرات توزیعی بین خروجی‌های مدل و مجموعه داده‌های ترجیحاتی قرار دارد. علاوه بر این، مطالعه نشان می‌دهد که در حالی که DPO تکراری ممکن است در مقایسه با آموزش داده‌های ثابت بهبودهایی را ارائه دهد، هنوز در وظایف چالش‌برانگیزی مانند تولید کد موفق به بهبود عملکرد نمی‌شود. مطالعات حذف اجزای PPO نشان می‌دهند که مولفه‌های اساسی برای عملکرد بهینه شامل نرم‌افزاری مزیت، اندازه بسته‌های بزرگ و بهروزرسانی‌های میانگین متحرک نمایی برای پارامترهای مدل مرجع هستند. این یافته‌ها مبنای راهنمایی‌های عملی برای تنظیم PPO را تشکیل داده و توانمندی PPO را در انجام وظایف متنوع و دستیابی به نتایج پیشرفته در وظایف چالش‌برانگیز، مانند رقابت‌های کدنویسی، نشان می‌دهند. به طور خاص، در مجموعه داده CodeContest، مدل PPO با ۴۴ میلیارد پارامتر، از مدل AlphaCode-41B پیشی می‌گیرد و بهبود قابل توجهی در معیارهای عملکرد نشان می‌دهد.

<sup>۱۲۱</sup> <https://github.com/huggingface/blog/blob/main/dpo-trl.md>

## شش - ۵: عملیات بهینه‌سازی مسیریابی و هرس (ORPO)

هرس مدل‌های زبانی بزرگ شامل حذف مولفه‌های غیرضروری یا تکراری از شبکه عصبی است تا اندازه و پیچیدگی آن کاهش یافته و کارایی و عملکرد آن بهبود یابد. این فرآیند به توسعه‌دهندگان و مهندسان هوش مصنوعی کمک می‌کند تا با چالش‌های مرتبط با پیاده‌سازی مدل‌های هوش مصنوعی در محیط‌های با منابع محدود، مانند دستگاه‌های موبایل، محاسبات لبه‌ای یا سیستم‌های تعبیه‌شده، مقابله کنند. روش‌های مختلفی برای هرس مدل‌های هوش مصنوعی وجود دارند که هر کدام متناسب با نوع و ساختار شبکه عصبی، هدف هرس و معیار هرس هستند. موارد زیر از رویکردهای رایج محسوب می‌شوند:

۱. **هرس وزن‌ها**: شامل حذف وزن‌ها یا ارتباطاتی با مقدار یا تأثیر کم بر خروجی است. این روش تعداد پارامترها و عملیات‌ها در مدل را کاهش می‌دهد، اگرچه ممکن است لزوماً حافظه یا تأخیر را کاهش ندهد.

۲. **هرس واحدها**: واحدها یا نورون‌هایی با کمترین فعال‌سازی یا تأثیر در خروجی را حذف می‌کند. این تکنیک می‌تواند حافظه و تأخیر مدل را کاهش دهد، اما ممکن است برای حفظ عملکرد نیاز به بازآموزی یا تنظیم دقیق داشته باشد.

۳. **هرس فیلترها**: شامل حذف فیلترها یا کانال‌های کامل در شبکه‌های عصبی پیچشی است که اهمیت یا ارتباط کمتری با خروجی دارند. این استراتژی همچنین حافظه و تأخیر را کاهش می‌دهد، اگرچه ممکن است برای حفظ عملکرد نیاز به بازآموزی یا تنظیم دقیق داشته باشد [۷۶].

## شش - ۵- یک: زمان مناسب برای هرس مدل‌های هوش مصنوعی

هرس مدل‌های هوش مصنوعی در مراحل مختلف توسعه و چرخه پیاده‌سازی مدل، با توجه به تکنیک و هدف انتخابی، انجام می‌شود.

۱. **هرس پیش از آموزش**: از دانش قبلی یا اکتشافی برای تعیین ساختار بهینه شبکه قبل از شروع آموزش استفاده می‌کند. این رویکرد می‌تواند در زمان و منابع آموزش صرفه‌جویی کند، اما ممکن است نیاز به طراحی و آزمایش دقیق برای یافتن بهترین پیکربندی داشته باشد.

۲. **هرس پس از آموزش**: از معیارها یا اصولی برای ارزیابی اهمیت یا تأثیر هر مولفه شبکه پس از آموزش استفاده می‌کند. این روش به حفظ عملکرد مدل کمک می‌کند، اما ممکن است نیاز به اعتبارسنجی و آزمایش اضافی برای اطمینان از کیفیت و استحکام داشته باشد.

۳. **هرس پویا**: ساختار شبکه را در زمان استنتاج یا اجرای لحظه‌ای، بر اساس بازخورد یا سیگنال‌ها تنظیم می‌کند. این رویکرد می‌تواند مدل را برای سناریوهای یا وظایف مختلف بهینه‌سازی کند، اما ممکن است نیاز به سربار محاسباتی و پیچیدگی بیشتری برای پیاده‌سازی و اجرا داشته باشد.

### **شش - ۵ - دو: مزایای هرس**

۱. کاهش اندازه و پیچیدگی: هرس اندازه و پیچیدگی مدل را کاهش می‌دهد، که منجر به کاهش مصرف حافظه و زمان اجرا می‌شود.
۲. بهبود کارایی: مدل‌های هرس شده به صورت بهینه‌تری اجرا می‌شوند و در محیط‌های با منابع محدود، مانند دستگاه‌های موبایل یا سیستم‌های تعییه‌شده، قابل استفاده هستند.
۳. حفظ عملکرد: اگر هرس به درستی انجام شود، می‌تواند عملکرد مدل را حفظ کرده و حتی در برخی موارد بهبود دهد.

### **شش - ۵ - سه: چالش‌های هرس**

۱. توازن بین کاهش اندازه و عملکرد: دستیابی به تعادل بین کاهش اندازه و پیچیدگی مدل و حفظ عملکرد آن، چالش‌برانگیز است؛ هرس بیش از حد یا ناکافی می‌تواند به کاهش کیفیت و کارایی مدل منجر شود.
۲. انتخاب تکنیک‌های مناسب: انتخاب تکنیک، معیار و هدف مناسب هرس برای نوع و ساختار خاص شبکه عصبی بسیار مهم است، زیرا روش‌های مختلف می‌توانند تأثیرات و نتایج متفاوتی داشته باشند.
۳. ارزیابی و اعتبارسنجی: مدل‌های هرس شده نیاز به ارزیابی و اعتبارسنجی دقیق دارند تا اطمینان حاصل شود که فرآیند هرس باعث ایجاد خطاهای سوگیری‌ها یا آسیب‌پذیری‌هایی که می‌توانند بر عملکرد و استحکام مدل تأثیر بگذارند، نشده است.

## فصل هفتم

### مرحله پنجم: ارزیابی و اعتبارسنجی

#### هفت – یک: مراحل ارزیابی و اعتبارسنجی مدل‌های تنظیم شده

۱. تنظیم معیارهای ارزیابی: معیارهای ارزیابی مناسب مانند کراس‌انتروپی<sup>۱۲۲</sup> را انتخاب کنید تا تفاوت بین توزیع‌های پیش‌بینی‌شده و واقعی داده‌ها اندازه‌گیری شود.
۲. تفسیر منحنی خطای آموزشی: منحنی خطای آموزشی را به‌طور مداوم پایش و تحلیل کنید تا از یادگیری موثر مدل اطمینان حاصل کنید و از الگوهای زیرآموزش یا بیش‌آموزی جلوگیری شود.
۳. اجرای حلقه‌های اعتبارسنجی: پس از هر ایپاک آموزشی، مدل را روی مجموعه اعتبارسنجی ارزیابی کنید تا معیارهای عملکرد مربوطه را محاسبه کرده و توانایی تعمیم مدل را دنبال کنید.
۴. پایش و تفسیر نتایج: به طور پیوسته رابطه بین معیارهای آموزشی و اعتبارسنجی را بررسی کنید تا از عملکرد پایدار و موثر مدل اطمینان حاصل شود.
۵. تنظیم و بهینه‌سازی هایپرپارامترها: هایپرپارامترهای کلیدی مانند نرخ یادگیری، اندازه بچ، و تعداد ایپاک‌های آموزشی را تنظیم کنید تا عملکرد مدل بهینه شده و از بیش‌آموزی جلوگیری شود.

#### هفت – دو: تنظیم معیارهای ارزیابی

کراس‌انتروپی یک معیار کلیدی برای ارزیابی مدل‌های زبانی بزرگ (LLMs) در حین آموزش یا تنظیم است. این معیار، که از نظریه اطلاعات سرچشمه می‌گیرد، تفاوت بین دو توزیع احتمالی را محاسبه می‌کند.

**هفت – دو – یک: اهمیت کراس‌انتروپی برای آموزش و ارزیابی مدل‌های زبانی بزرگ**  
کراس‌انتروپی در آموزش و تنظیم مدل‌های زبانی بزرگ اهمیت بالایی دارد. این معیار به عنوان تابع خطا عمل می‌کند و با کاهش اختلاف بین داده‌های پیش‌بینی‌شده و واقعی، مدل را به سمت تولید پیش‌بینی‌های باکیفیت هدایت می‌کند. در مدل‌های زبانی بزرگ، هر کلمه بالقوه به عنوان یک کلاس مجزا عمل می‌کند و وظیفه مدل،

<sup>122</sup> Cross-Entropy

پیش‌بینی کلمه بعدی با توجه به متن است. این کار به خودی خود پیچیده است و نیاز به درک عمیق از نحو، معناشناسی و بستر متن دارد.

### هفت – دو – فراتر از کراس‌انتروپی: معیارهای پیشرفته برای ارزیابی مدل‌های زبانی بزرگ

در حالی که کراس‌انتروپی به عنوان معیار اساسی باقی می‌ماند، ارزیابی موثر مدل‌های زبانی بزرگ نیازمند معیارهای بیشتری است که به جنبه‌های مختلف عملکرد مدل بپردازند. برخی از این معیارهای پیشرفته عبارتند از:

#### پیچیدگی (Perplexity)

پیچیدگی اندازه می‌گیرد که مدل یا توزیع احتمالاتی چقدر می‌تواند یک نمونه را به خوبی پیش‌بینی کند. در مدل‌های زبانی بزرگ، پیچیدگی، میزان اطمینان مدل نسبت به کلمه بعدی در یک دنباله را ارزیابی می‌کند. پیچیدگی کمتر نشان‌دهنده عملکرد بهتر است، چراکه مدل اطمینان بیشتری در پیش‌بینی‌های خود دارد.

#### دقت اطلاعات (Factuality)

دقت اطلاعات به صحت اطلاعات تولید شده توسط مدل اشاره دارد. این معیار به ویژه در کاربردهایی که اطلاعات نادرست می‌تواند پیامدهای جدی داشته باشد، حائز اهمیت است. امتیاز دقت اطلاعات بالاتر، با کیفیت بالاتر خروجی ارتباط دارد.

#### عدم اطمینان مدل (LLM Uncertainty)

عدم اطمینان مدل با استفاده از احتمال لگاریتمی اندازه‌گیری می‌شود و به شناسایی تولیدات کم کیفیت کمک می‌کند. عدم اطمینان کمتر نشان‌دهنده کیفیت خروجی بالاتر است. این معیار از احتمال لگاریتمی هر توکن تولیدی استفاده کرده و بینش‌هایی از اطمینان مدل نسبت به پاسخ‌هایش ارائه می‌دهد.

#### پیچیدگی درخواست (Prompt Perplexity)

این معیار میزان درک مدل از درخواست ورودی را ارزیابی می‌کند. پیچیدگی درخواست پایین‌تر نشان می‌دهد که درخواست روش‌ن و قابل درک است و این می‌تواند منجر به عملکرد بهتر مدل شود.

#### ارتباط با متن (Context Relevance)

در سیستم‌های تولید بازیابی-تقویت شده<sup>۱۲۳</sup> (RAG)، ارتباط با متن میزان ارتباط متن بازیابی شده با پرسش کاربر را می‌سنجد. ارتباط بیشتر با متن، کیفیت پاسخ‌های تولیدی را بهبود می‌بخشد زیرا مدل از اطلاعات مرتبط‌تر استفاده می‌کند.

<sup>123</sup> Retrieval-Augmented Generation (RAG)

## کامل بودن (Completeness)

کامل بودن ارزیابی می‌کند که آیا پاسخ مدل به‌طور کامل پرسش را بر اساس متن موجود پوشش می‌دهد. کامل بودن بالا تضمین می‌کند که تمامی اطلاعات مرتبط در پاسخ گنجانده شده‌اند و دقت و کاربردپذیری آن را بهبود می‌بخشد.

## نسبت و بهره‌وری از تکه‌های اطلاعاتی (Chunk Attribution and Utilization)

این معیارها ارزیابی می‌کنند که مدل تا چه اندازه تکه‌های بازیابی شده از اطلاعات را به طور موثر در پاسخ نهایی به کار می‌گیرد. امتیاز بالاتر در این معیارها نشان می‌دهد که مدل به شکل کارآمد از متن موجود برای تولید پاسخ‌های دقیق و مرتبط استفاده می‌کند.

## پتانسیل خطای داده (Data Error Potential)

این معیار دشواری مدل را در یادگیری از داده‌های آموزشی کمیت‌گذاری می‌کند. کیفیت بالاتر داده منجر به کاهش پتانسیل خطای داده و بهبود عملکرد مدل را در پی دارد.

## معیارهای ایمنی (Safety Metrics)

معیارهای ایمنی تضمین می‌کنند که خروجی‌های مدل‌های زبانی بزرگ مناسب و غیرمضر هستند. این معیارها در بخش‌های نهایی فصل مورد بررسی قرار گرفته‌اند. ادغام این معیارهای پیشرفته، دید جامعی از عملکرد مدل‌های زبانی بزرگ فراهم می‌کند و به توسعه‌دهندگان اجازه می‌دهد تا به شکلی موثر مدل‌ها را تنظیم و بهینه‌سازی کنند. با استفاده از رویکردهای مبتنی بر معیارهای ایمنی اطمینان از تولید خروجی‌های دقیق و با کیفیت بالا توسط مدل‌های زبانی بزرگ فراهم می‌شود و آنها می‌توانند به طور پیوسته و مطمئن در کاربردهای مختلف به کار گرفته شوند.

## هفت – سه: درگ منحنی خطای آموزشی

منحنی خطای آموزشی، مقدار خطای ایپاک‌های آموزشی ترسیم می‌کند و برای پایش عملکرد مدل سیار حیاتی است.

### هفت – سه – یک: تفسیر منحنی‌های خطای

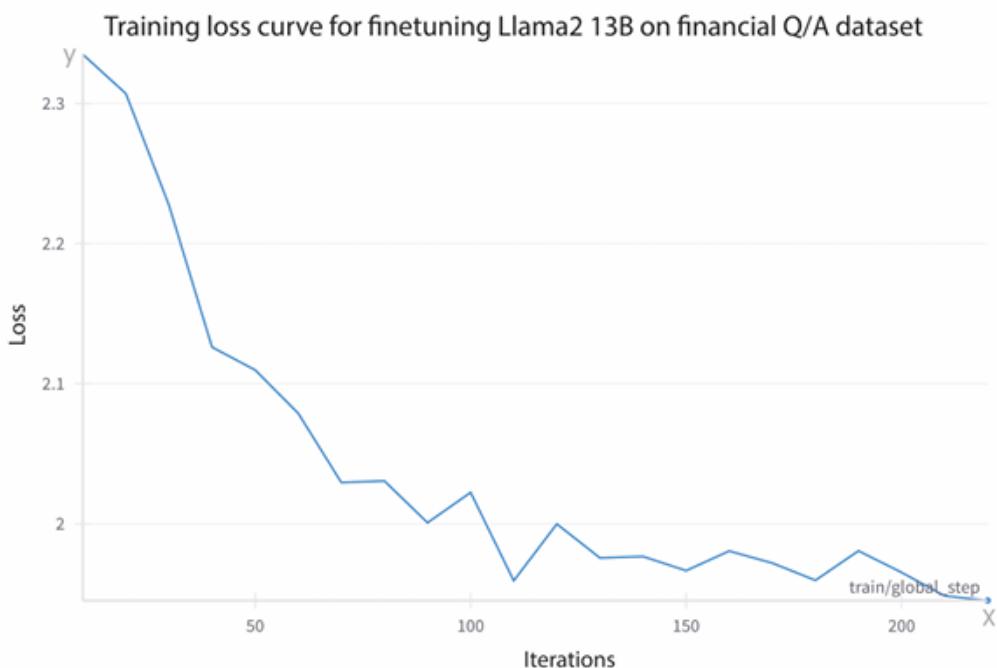
یک منحنی خطای آموزشی ایده‌آل نشان‌دهنده کاهش سریع خطای در مراحل اولیه است، که پس از آن کاهش تدریجی و در نهایت یک سکو را تجربه می‌کند. الگوهای خاصی که باید به دنبال آنها باشید عبارتند از:

۱. **نقص یادگیری (Underfitting):** مقدار خطای بالا که به‌طور قابل توجهی در طول زمان کاهش نمی‌یابد، نشان‌دهنده این است که مدل نمی‌تواند داده‌ها را یاد بگیرد. به عبارت دیگر Underfitting به وضعیتی اشاره دارد که در آن یک مدل یادگیری ماشین نمی‌تواند الگوهای موجود در داده‌های آموزشی را به درستی شناسایی

کند. این وضعیت معمولاً زمانی رخ می‌دهد که مدل خیلی ساده باشد و نتواند پیچیدگی‌های داده‌ها را یاد بگیرد.

۲. بیشآموزی (**Overfitting**): کاهش خطای آموزشی با افزایش خطای اعتبارسنجی، که نشان‌دهنده این است که مدل داده‌های آموزشی را حفظ کرده و قادر به تعمیم به داده‌های جدید نیست.

۳. نوسانات (**Fluctuations**): تغییرات قابل توجه ممکن است نشان‌دهنده نرخ یادگیری بالا یا گرادیان‌های نویزی باشد.



شکل ۷.۱: این شکل مثال منحنی خطای آموزشی را نشان می‌دهد که کاهش خطای آموزشی در طول تکرارها در حین فاین‌تاپینینگ Llama2 13B بر روی یک مجموعه داده سوال و جواب مالی ترسیم می‌کند. این منحنی اثربخشی فرآیند فاین‌تاپینینگ را در کاهش خطای آموزشی نشان می‌دهد.

## هفت - سه - دو: اجتناب از بیشآموزی

تکنیک‌هایی برای جلوگیری از بیشآموزی شامل موارد زیر است:

۱. منظم‌سازی (**Regularisation**): اضافه کردن یک عبارت جریمه بهتابع خطای آموزش زمانی که عملکرد اعتبارسنجی دیگر بهبود کوچک‌تر.

۲. توقف زودهنگام (**Early Stopping**): متوقف کردن آموزش زمانی که عملکرد اعتبارسنجی دیگر بهبود نمی‌یابد.

۳. دراپ آوت (**Dropout**): غیرفعال کردن تصادفی نورون‌ها در حین آموزش برای کاهش حساسیت به نویز.
۴. اعتبارسنجی متقابل (**Cross-Validation**): تقسیم داده‌ها به چندین زیرمجموعه برای آموزش و اعتبارسنجی به منظور ارزیابی تعمیم‌پذیری مدل.
۵. نرمالیزاسیون دسته‌ای (**Batch Normalisation**): نرمال‌سازی ورودی‌ها به هر لایه در حین آموزش برای تثبیت فرآیند یادگیری.
۶. مجموعه داده‌های بزرگتر و اندازه‌های دسته‌ای (**Larger Datasets and Batch Sizes**): کاهش بیش‌آموزی با افزایش میزان داده‌های متنوع و اندازه‌های دسته‌ای.

### **هفت – سه – سه: منابع گرادیان‌های نویزی (Sources of Noisy Gradients)**

گرادیان‌های نویزی در حین آموزش مدل‌های یادگیری ماشین، از جمله LLM‌ها، شایع هستند. آنها ناشی از تغییرپذیری در تخمین‌های گرادیان به دلیل نزول تصادفی گرادیان و انواع آن هستند. استراتژی‌هایی برای مدیریت گرادیان‌های نویزی شامل موارد زیر است:

۱. برنامه‌ریزی نرخ یادگیری (**Learning Rate Scheduling**): کاهش تدریجی نرخ یادگیری در طول آموزش می‌تواند تأثیر گرادیان‌های نویزی را کاهش دهد.
۲. کلیپ کردن گرادیان (**Gradient Clipping**): تعیین یک آستانه برای مقادیر گرادیان از بهروزرسانی‌های بزرگ جلوگیری می‌کند که می‌تواند آموزش را ناپایدار کند.

### **هفت – چهار: اجرای حلقه‌های اعتبارسنجی**

حلقه‌های اعتبارسنجی ارزیابی بدون تعصب از عملکرد مدل را فراهم می‌کنند. مراحل معمول شامل موارد زیر است:

۱. تقسیم داده (**Split Data**): تقسیم مجموعه داده به مجموعه‌های آموزشی و اعتبارسنجی.
۲. آغاز اعتبارسنجی (**Initialize Validation**): ارزیابی مدل بر روی مجموعه اعتبارسنجی در انتهای هر ایپاک.
۳. محاسبه متريک‌ها (**Calculate Metrics**): محاسبه متريک‌های عملکرد مرتبط، مانند خطای متقاطع.
۴. ضبط نتایج (**Record Results**): ثبت متريک‌های اعتبارسنجی برای هر ایپاک.
۵. توقف زودهنگام (**Early Stopping**): در صورت تمایل، آموزش را متوقف کنید اگر خطای اعتبارسنجی به مدت تعداد معينی از ایپاک‌ها بهبود نیابد.

## هفت – پنج: پایش و تفسیر نتایج

پایش نتایج اعتبارسنجی شامل تحلیل روندها در متريک‌های اعتبارسنجی در طول ایپاک‌ها است. جنبه‌های کلیدی شامل موارد زیر است:

۱. بهبود مداوم (**Consistent Improvement**): نشان‌دهنده تعميم‌پذيری خوب مدل است اگر هر دو متريک آموزشی و اعتبارسنجی بهبود یافته و به يك سکو برسند.
۲. انحراف (**Divergence**): اگر متريک‌های آموزشی بهبود یابند در حالی که متريک‌های اعتبارسنجی بدتر شوند، اين موضوع نشان‌دهنده بيش‌آموزی است.
۳. ثبات (**Stability**): اطمینان حاصل کنيد که متريک‌های اعتبارسنجی به طور قابل توجهی نوسان نکنند که نشان‌دهنده آموزش پایدار است.

## هفت – شش: تنظيم‌های هايپرپارامتر و ساير تغييرات

تنظيم دقیق شامل تنظيم هايپرپارامترهای کلیدی برای دستیابی به عملکرد بهینه است. هايپرپارامترهای مهم شامل موارد زیر است:

۱. نرخ يادگيري (**Learning Rate**): اندازه گام برای بهروزرسانی وزن‌های مدل را تعیین می‌کند. يك نقطه شروع خوب ( $2e-4$ ) است، اما این مقدار می‌تواند متفاوت باشد.
۲. اندازه دسته (**Batch Size**): اندازه‌های بزرگ‌تر دسته بهروزرسانی‌های پایدارتر ایجاد می‌کنند اما به حافظه بیشتری نیاز دارند.
۳. تعداد ایپاک‌های آموزشی (**Number of Training Epochs**): تعادل در تعداد ایپاک‌ها اطمینان حاصل می‌کند که مدل به اندازه کافی یاد بگیرد بدون اينکه بيش‌آموزی یا زير‌آموزی رخ دهد.
۴. بهينه‌ساز (**Optimizer**): بهينه‌سازهای مانند Paged ADAM استفاده از حافظه را بهينه می‌کنند که برای مدل‌های بزرگ مزیت دارد. پارامترهای قابل تنظيم دیگر شامل نرخ دروپ‌آوت، کاهش وزن و مراحل گرم‌کردن است.

## هفت – شش – يك: اندازه و كيفيت داده

كارابي LLM‌ها به طور مستقيم تحت تأثير كيفيت داده‌های آموزشی آنها قرار دارد. اطمینان از اينکه مجموعه داده‌ها تميز، مرتبط و کافي هستند، بسيار مهم است. تميزی داده به عدم وجود نویز، خطاهای ناهمانگی‌ها در داده‌های برچسب‌گذاری شده اشاره دارد. به عنوان مثال، داشتن عبارتی مانند "این مقاله پيشنهاد می‌کند..." چندين

بار در داده‌های آموزشی می‌تواند پاسخ LLM‌ها را خراب کرده و بایاس به استفاده از این عبارت خاص را بیشتر و در موقعیت‌های نامناسب‌تر ایجاد کند.

## هفت – هفت: ارزیابی LLM‌های دقیق تنظیم شده

LLM‌های مدرن با استفاده از معیارهای استاندارد مانند HellaSwag، SuperGLUE، GLUE، MMLU و TruthfulQA ارزیابی می‌شوند (به جدول ۷.۱ مراجعه کنید). این معیارها توانایی‌های مختلف را ارزیابی کرده و نمای کلی از عملکرد LLM را ارائه می‌دهند.

معیار	توصیف	منبع
GLUE	مجموعه‌های استاندارد از وظایف متنوع NLP برای ارزیابی اثربخشی مدل‌های زبانی مختلف	<a href="https://gluebenchmark.com/">https://gluebenchmark.com/</a>
SuperGLUE	مقایسه وظایف چالش‌برانگیزتر و متنوع‌تر با GLUE، همراه با پایگاه‌های انسانی جامع	<a href="https://super.gluebenchmark.com">https://super.gluebenchmark.com</a>
HellaSwag	ارزیابی توانایی یک مدل زبانی بزرگ (LLM) در کامل کردن جملات	<a href="https://rowanzellers.com/hellaswag">https://rowanzellers.com/hellaswag</a>
TruthfulQA	اندازه‌گیری صداقت پاسخ‌های مدل	<a href="https://github.com/hendrycks/test">https://github.com/hendrycks/test</a>
MMLU	ارزیابی توانایی LLM در انجام چندوظیفه‌ای	<a href="https://github.com/hendrycks/test">https://github.com/hendrycks/test</a>
IFEval	آزمایش توانایی مدل در پیروی از دستورالعمل‌های صریح، با تمرکز بر رعایت قالب‌بندی	<a href="https://github.com/google-research/google-research/tree/master/instruction_following_eval">https://github.com/google-research/google-research/tree/master/instruction_following_eval</a>
BBH (Big Bench Hard)	23 وظیفه چالش‌برانگیز از مجموعه داده BigBench برای ارزیابی LLM‌ها با استفاده از معیارهای عینی	<a href="https://github.com/suzgunmirac/BIG-Bench-Hard">https://github.com/suzgunmirac/BIG-Bench-Hard</a>
MATH	مجموعه‌ای از مسائل مسابقات دبیرستانی که با استفاده از Asymptote و LaTeX فرمتبندی شده‌اند	<a href="https://github.com/hendrycks/apps">https://github.com/hendrycks/apps</a>
GPQA	مجموعه داده‌ای چالش‌برانگیز با سؤالاتی که توسط کارشناسان دکتری در زمینه‌های مختلف طراحی شده‌اند	<a href="https://github.com/idavidrein/gpqa">https://github.com/idavidrein/gpqa</a>
MuSR	مجموعه داده‌ای با مسائل پیچیده که نیازمند ادغام استدلال با تحلیل متن در مقیاس بلند است	<a href="https://github.com/Zayne-Sprague/MuSR">https://github.com/Zayne-Sprague/MuSR</a>
MMLU-PRO	نسخه بهبودیافته MMLU با کیفیت بالاتر و سؤالات چندگزینه‌ای چالش‌برانگیزتر	<a href="https://github.com/TIGER-AI-Lab/MMLU-Pro">https://github.com/TIGER-AI-Lab/MMLU-Pro</a>

<a href="https://allenai.org/data/arc">https://allenai.org/data/arc</a>	اندازه‌گیری استدلال ماشینی با استفاده از مجموعه داده‌ای از سوالات علمی مدرسه‌ای	<b>ARC</b>
<a href="https://stanfordnlp.github.io/coqa">https://stanfordnlp.github.io/coqa</a>	مجموعه داده‌ای برای ساخت سیستم‌های پاسخ‌گویی به سوالات گفتگویی	<b>COQA</b>
<a href="https://github.com/allenai/allennlp-reading-comprehension/blob/master/allennlp_rc/eval/drop_eval.py">https://github.com/allenai/allennlp-reading-comprehension/blob/master/allennlp_rc/eval/drop_eval.py</a>	ارزیابی توانایی انجام استدلال‌های گستره بر روی پاراگراف‌های متن	<b>DROP</b>
<a href="https://rajpurkar.github.io/SQuAD-explorer/">https://rajpurkar.github.io/SQuAD-explorer/</a>	مجموعه داده‌ای برای درک مطلب که توانایی مدل‌ها را در پاسخ‌گویی به سوالات بر اساس متون ارزیابی می‌کند	<b>SQuAD</b>
<a href="https://trec.nist.gov">https://trec.nist.gov</a>	معیاری برای ارزیابی روش‌های بازیابی متن	<b>TREC</b>
<a href="https://www.statmt.org/wmt20/">https://www.statmt.org/wmt20/</a>	مجموعه داده و معیاری برای ارزیابی مدل‌های ترجمه ماشینی	<b>WMT</b>
<a href="https://cims.nyu.edu/~sbowman/xnli/">https://cims.nyu.edu/~sbowman/xnli/</a>	مجموعه داده‌ای برای ارزیابی درک زبان بین‌زبانی	<b>XNLI</b>
<a href="https://github.com/ybisk/ybisk.github.io/tree/master/piqa">https://github.com/ybisk/ybisk.github.io/tree/master/piqa</a>	مجموعه داده‌ای برای ارزیابی درک مدل‌ها از تعاملات فیزیکی	<b>PiQA</b>
<a href="https://allenai.org/projects/winogrande">https://allenai.org/projects/winogrande</a>	مجموعه داده‌ای با مقیاس بزرگ برای ارزیابی استدلال منطقی	<b>Winogrande</b>

جدول ۱.۷: بررسی دقیق مجموعه داده‌های معیار برای ارزیابی عملکرد مدل‌های زبانی

با پیشرفت LLM‌ها، معیارهای جدیدی مانند BigCodeBench چالش‌هایی برای معیارهای فعلی ایجاد کرده و استانداردهای جدیدی را در این حوزه تعیین می‌کند. با توجه به ماهیت متنوع LLM‌ها و کارهایی که می‌توانند انجام دهند، انتخاب معیارها به وظایف خاصی که LLM انتظار می‌رود انجام دهد، بستگی دارد. برای کاربردهای عمومی، باید از معیارهای مختلف برای برنامه‌های پایین‌دستی و استدلال‌های مختلف استفاده شود. برای LLM‌های خاص به دامنه یا وظیفه، ارزیابی می‌تواند محدود به معیارهای مرتبط مانند BigCodeBench برای برنامه‌نویسی باشد.

## هفت – هشت: ارزیابی LLM‌های دقیق تنظیم شده در معیار ایمنی

جنبه‌های ایمنی مدل‌های زبانی بزرگ (LLM‌ها) به طور فزاینده‌ای مورد بررسی قرار می‌گیرد زیرا این مدل‌ها قادر به تولید محتواهای مضر هستند وقتی که تحت تأثیر پرامپت‌های جیلبریک قرار می‌گیرند. این پرامپت‌ها می‌توانند دستورالعمل‌های ایمنی و اخلاقی داخلی مدل‌ها را دور بزنند، مشابه تکنیک‌های تزریق کد که در امنیت رایانه‌ای سنتی برای دور زدن پروتکل‌های ایمنی استفاده می‌شود. بهویژه، مدل‌هایی مانند GPT-3، ChatGPT و

InstructGPT در برابر این نوع دستکاری‌ها آسیب‌پذیر هستند که محدودیت‌های تولید محتوا را حذف می‌کنند و ممکن است قوانین OpenAI را نقض کنند. این امر ضرورت وجود ایمنی‌های قوی را برای اطمینان از پایبندی خروجی‌های LLM به استانداردهای اخلاقی و ایمنی نشان می‌دهد.

[۷۷] ارزیابی جامعی از قابلیت اعتماد LLM‌ها ارائه می‌دهد و بهویژه GPT-4 را با DecodingTrust ۳.۵ (ChatGPT) مقایسه می‌کند. این ارزیابی شامل چندین حوزه حیاتی است:

۱. **سمی بودن**: الگوریتم‌های بهینه‌سازی و مدل‌های تولیدی برای ایجاد پرامپت‌های چالش‌برانگیز که توانایی مدل را در جلوگیری از تولید محتوای مضر آزمایش می‌کند، به کار می‌روند.

۲. **تعصب کلیشه‌ای**: مجموعه‌ای از گروه‌های دموگرافیک و موضوعات کلیشه‌ای برای ارزیابی تعصب مدل استفاده می‌شود که به درک و کاهش پاسخ‌های پیش‌داوری کمک می‌کند.

۳. **استحکام در برابر حملات**: پایداری مدل‌ها در برابر حملات خصم‌مانه با به چالش کشیدن آنها با الگوریتم‌های پیچیده‌ای که برای فربیب یا گمراه کردن طراحی شده‌اند، آزمایش می‌شود.

۴. **استحکام در برابر ورودی‌های خارج از توزیع (OOD)**: مدل‌ها بر اساس توانایی خود در مدیریت ورودی‌هایی که به‌طور قابل توجهی با داده‌های آموزشی آنها متفاوت است، ارزیابی می‌شوند، مانند سبک‌های شعری یا شکسپیری.

۵. **استحکام در برابر نمایش‌های خصم‌مانه**: نمایش‌هایی که حاوی اطلاعات گمراه‌کننده هستند برای آزمایش استحکام مدل در وظایف مختلف استفاده می‌شوند.

۶. **حریم خصوصی**: سطوح مختلف ارزیابی حریم خصوصی بررسی می‌کند که چگونه مدل‌ها اطلاعات حساس را در حین تعاملات ایمن نگه می‌دارند و زمینه‌های مرتبط با حریم خصوصی را درک می‌کنند.

۷. **شناسایی توهمند**: شناسایی مواردی که در آن مدل اطلاعاتی تولید می‌کند که در زمینه یا داده‌های واقعی ارائه شده مستقر نیست. نرخ‌های پایین‌تر توهمند به بهبود قابلیت اطمینان و اعتبار خروجی‌های LLM کمک می‌کند.

۸. **مناسب بودن لحن**: ارزیابی می‌کند که آیا خروجی مدل لحن مناسبی برای زمینه مورد نظر حفظ می‌کند یا خیر. این امر بهویژه برای برنامه‌های خدمات مشتری، مراقبت‌های بهداشتی و دیگر زمینه‌های حساس مهم است.

۹. **اخلاق ماشین**: ارزیابی‌های اخلاقی شامل آزمایش مدل‌ها با سناریوهایی است که نیاز به قضاوت‌های اخلاقی دارند و از مجموعه‌های داده‌ای مانند ETHICS و Jiminy Cricket استفاده می‌کند.

۱۰. **انصاف**: انصاف مدل‌ها با تولید وظایفی که ویژگی‌های محافظت‌شده را تغییر می‌دهند، ارزیابی می‌شود و پاسخ‌های منصفانه‌ای را در میان گروه‌های دموگرافیک مختلف تضمین می‌کند.

مجموعه داده‌ای که برای ارزیابی هشت بعد ایمنی فوق‌الذکر استفاده می‌شود در اینجا موجود است. به عنوان همکاری با HuggingFace، جدول رتبه‌بندی ایمنی LLM از چارچوب DecodingTrust برای ارائه یک پلتفرم

ارزیابی یکپارچه برای اینمی LLM استفاده می‌کند. این امکان را برای محققان و کارشناسان فراهم می‌آورد تا بهتر درک کنند که قابلیت‌ها، محدودیت‌ها و خطرات مرتبط با LLM‌ها چیست. از کاربران دعوت می‌شود که مدل‌های خود را برای ارزیابی به HuggingFace ارسال کنند و اطمینان حاصل کنند که با استانداردهای در حال تحول اینمی و قابلیت اطمینان در این حوزه مطابقت دارند.

## هفت – نه: ارزیابی اینمی LLM‌های دقیق تنظیم شده با استفاده از مدل‌های هوش مصنوعی

### هفت – نه – یک: Llama Guard

[۷۸] Llama Guard 2 یک مدل حفاظتی است که بر روی LLM‌ها برای مدیریت ریسک‌ها در برنامه‌های هوش مصنوعی مکالمه‌ای ساخته شده است. این مدل به طور مؤثری پرامپت‌های ورودی و پاسخ‌ها از عوامل هوش مصنوعی را با استفاده از یک طبقه‌بندی دقیق ریسک اینمی که به شناسایی ریسک‌های قانونی و سیاستی بالقوه در تعاملات هوش مصنوعی کمک می‌کند، دسته‌بندی می‌کند. این طبقه‌بندی به مؤثر بودن شناسایی در حوزه‌هایی مانند:

- خشونت و نفرت: بررسی محتوایی که می‌تواند اقدامات خشونت‌آمیز یا تبعیض‌آمیز را تحریک کند.
  - محتوای جنسی: هدف‌گذاری محتوای یا رفتارهای جنسی صریح، به‌ویژه مربوط به افراد زیر سن قانونی.
  - سلاح‌ها و تسلیحات غیرقانونی: مربوط به ترویج یا آموزش تسلیحات غیرقانونی.
  - مواد تنظیم شده یا کنترل شده: شامل مواد مخدر غیرقانونی و سایر مواد کنترل شده.
  - خودکشی و خودآزاری: هدف‌گذاری محتوایی که می‌تواند رفتارهای خودتخیری را تشویق کند.
  - برنامه‌ریزی جنایی: محتوایی که می‌تواند به برنامه‌ریزی یا اجرای فعالیت‌های جنایی کمک کند.
- هسته 2 Llama Guard، چارچوب قوی آن است که امکان دسته‌بندی پرامپت و پاسخ را فراهم می‌کند و از یک مجموعه داده با کیفیت بالا که قابلیت نظارت بر تبادلات مکالمه‌ای را بهبود می‌بخشد، پشتیبانی می‌کند. با کار بر روی مدل Llama Guard 2-7b برای ارائه عملکرد قوی در معیارهایی مانند مجموعه داده ارزیابی مدیریت OpenAI و ToxicChat تنظیم شده است، جایی که توانایی‌های آن با ابزارهای موجود مدیریت محتوا برابری یا فراتر می‌رود.

این مدل از طبقه‌بندی چندکلاسه پشتیبانی می‌کند و امتیازهای تصمیم‌گیری باینری تولید می‌کند. تنظیم دقیق دستورالعمل‌های آن امکان سفارشی‌سازی گسترده و ظایف و تطبیق فرمتهای خروجی را فراهم می‌کند. این ویژگی به کاربران اجازه می‌دهد که دسته‌های طبقه‌بندی را برای انطباق با موارد استفاده خاص اصلاح کنند و از قابلیت‌های درخواست انعطاف‌پذیر، از جمله کاربردهای بدون آموزش و با آموزش چندگانه، پشتیبانی می‌کند. انطباق‌پذیری و کارایی Llama Guard آن را به منبعی حیاتی برای توسعه‌دهندگان و محققان تبدیل می‌کند. با

در دسترس قرار دادن وزن‌های مدل به صورت عمومی، Llama Guard 2 توسعه و سفارشی‌سازی مداوم برای برآورده کردن نیازهای در حال تحول اینمی‌هوش مصنوعی در جامعه را تشویق می‌کند.

Llama Guard 3 پیشرفت جدیدی بر 2 Llama Guard است که بر روی مدل Llama 3 8b دقيق تنظیم شده است. تفاوت کلیدی بین این دو نسخه این است که Llama Guard 3 با معرفی سه دسته جدید: افترا، انتخابات و سوءاستفاده از مفسر کد، قابلیت‌های Llama Guard 2 را گسترش می‌دهد.

کتابخانه پایتون: 3 Hugging Face در AutoModelForCausalLM قابل دسترسی است. یک آموزش دقیق از [اینجا](#)<sup>۱۲۴</sup> در دسترس است.

لطفاً توجه داشته باشید که برای دسترسی به مدل، نیاز است که درخواست دسترسی را همراه با اطلاعات کاربری به Hugging Face ارسال کنید. علاوه بر این، وزن‌های مدل را می‌توانید با ارائه اطلاعات کاربری از پلتفرم Meta دانلود کنید و لینک آن در [اینجا](#)<sup>۱۲۵</sup> موجود است.

فرمت پرامپتها برای این دو مدل نیز متفاوت است؛ فرمتهای خاص برای 2 Llama Guard در [اینجا](#)<sup>۱۲۶</sup> و برای 3 Llama Guard در [اینجا](#)<sup>۱۲۷</sup> قابل دسترسی هستند.

## هفت - نه - دو: شیلد جما (Shield Gemma)

شیلد جما [۷۹] یک مدل پیشرفتی مدیریت محتوای ساخته شده بر روی پلتفرم Gemma2 است که برای افزایش اینمی و قابلیت اعتماد تعاملات بین LLMها و کاربران طراحی شده است. این مدل به طور موثری ورودی‌های کاربر و خروجی‌های مدل را فیلتر می‌کند تا از انواع آسیب‌های کلیدی، از جمله زبان توهین‌آمیز، سخنان نفرت‌انگیز، اطلاعات نادرست و محتوای صریح جلوگیری کند. مقیاس‌پذیری این مدل، با گزینه‌هایی از B27 پارامتر، اجازه می‌دهد که کاربردهای خاصی که نیازهای خاصی را برآورده می‌کنند، مانند کاهش تأخیر در برنامه‌های اینمی آنلاین یا افزایش عملکرد در وظایف تصمیم‌گیری پیچیده، سفارشی شوند.

یکی از ویژگی‌های متمایز شیلد جما رویکرد نوآورانه آن به گردآوری داده‌ها است. این مدل از تکنیک‌های تولید داده‌های مصنوعی برای ایجاد مجموعه‌های داده با کیفیت بالا استفاده می‌کند که در برابر پرامپتهای خصم‌مانه مقاوم و در بین گروه‌های هویتی مختلف عادلانه هستند. این رویکرد نیاز به حاشیه‌نویسی انسانی گستردگی را کاهش می‌دهد و فرآیند آماده‌سازی داده‌ها را ساده می‌کند و در عین حال اثربخشی مدل را تضمین می‌کند. در مقایسه با ابزارهای موجود مدیریت محتوا مانند WildGuard و LlamaGuard که معمولاً مدل‌های با اندازه ثابت و سفارشی‌سازی محدود ارائه می‌دهند، معماری منعطف شیلد جما و قابلیت‌های پیشرفتی مدیریت داده‌ها یک راه حل سازگارتر و کارآمدتر ارائه می‌دهد. این نوآوری‌ها شیلد جما را به عنوان یک پیشرفت قابل توجه در

<sup>124</sup> <https://huggingface.co/meta-llama/Llama-Guard-3-8B>

<sup>125</sup> <https://www.llama.com/llama-downloads/>

<sup>126</sup> <https://www.llama.com/docs/model-cards-and-prompt-formats/meta-llama-guard-2/>

<sup>127</sup> <https://llama.meta.com/docs/model-cards-and-prompt-formats/llama-guard-3>

مدیریت محتوای مبتنی بر LLM معرفی می‌کند و ابزاری چندمنظوره را برای توسعه‌دهندگان و محققان فراهم می‌آورد که تعاملات ایمن‌تر و قابل اعتمادتر هوش مصنوعی را در سرتاسر پلتفرم‌های مختلف ترویج می‌کند.

کتابخانه پایتون: مجموعه شیلد جما از طریق AutoModelForCausalLM در HuggingFace در دسترس است. می‌توانید به مدل‌ها از [اینجا<sup>128</sup>](#) دسترسی پیدا کنید. یک راهنمای آموزشی برای اجرای شیلد جما در Google Colab در [اینجا<sup>129</sup>](#) موجود است. مشابه مجموعه Llama Guard، مجموعه شیلد جما نیز دستورالعمل‌هایی برای پرامپت‌نویسی دارد که می‌توانید آن را در [اینجا<sup>130</sup>](#) پیدا کنید.

## هفت - نه - سه: وايلدگارد (WILDGUARD)

وايلدگارد [۸۰] یک ابزار نوآورانه متن‌باز است که برای افزایش ایمنی تعاملات با مدل‌های زبانی بزرگ (LLM‌ها) توسعه یافته است. این ابزار به سه وظیفه کلیدی مدیریت محتوا می‌پردازد: شناسایی نیتهاي مضر در پرامپت‌های کاربر، شناسایی خطرات ایمنی در پاسخ‌های مدل و تعیین زمان مناسب برای رد درخواست‌های نایمن. در مرکز توسعه آن، مجموعه داده‌ای به نام WILDGUARD MIX<sup>131</sup> وجود دارد که شامل ۹۲,۰۰۰ مثال برچسب‌گذاری شده است که شامل پرامپت‌های بی‌خطر و تلاش‌های خصمانه برای دور زدن تدبیر ایمنی است. این مجموعه داده به WILDGUARD TRAIN تقسیم می‌شود که برای آموزش مدل استفاده می‌شود و WILDGUARD TEST که شامل مثال‌های باکیفیت حاشیه‌نویسی شده توسط انسان برای ارزیابی است.

مدل وايلدگارد به‌طور خاص بر روی مدل زبانی Mistral-7B با استفاده از مجموعه داده WILDGUARD TRAIN دقیق تنظیم شده است و این امکان را به آن می‌دهد تا تمام سه وظیفه مدیریت محتوا را به صورت یکپارچه و چندوظیفه‌ای انجام دهد. نتایج نشان می‌دهد که وايلدگارد در مقایسه با ابزارهای موجود مدیریت محتوا در متن‌باز، در کارایی برتری دارد و به‌ویژه در برخورد با پرامپت‌های خصمانه و شناسایی دقیق رد درخواست‌های مدل موفق عمل می‌کند. در بسیاری از معیارها، عملکرد وايلدگارد در سطح یا فراتر از GPT-4، که یک مدل بسته و بسیار بزرگ‌تر است، قرار دارد.

در این گیتهاب راهنمای سریع و اطلاعات اضافی درباره وايلدگارد در دسترس است و می‌توانید به آن در [اینجا<sup>132</sup>](#) مراجعه کنید.

<sup>128</sup> <https://huggingface.co/google>

<sup>129</sup> [https://colab.research.google.com/github/google/generative-ai-docs/blob/main/site/en/responsible/docs/safeguards/shieldgemma\\_on\\_keras.ipynb](https://colab.research.google.com/github/google/generative-ai-docs/blob/main/site/en/responsible/docs/safeguards/shieldgemma_on_keras.ipynb)

<sup>130</sup> [https://ai.google.dev/gemma/docs/shieldgemma/model\\_card](https://ai.google.dev/gemma/docs/shieldgemma/model_card)

<sup>131</sup> <https://huggingface.co/datasets/allenai/wildguardmix>

<sup>132</sup> <https://github.com/allenai/wildguard>

## فصل هشتم

### مرحله ششم: استقرار

#### هشت - یک: مراحل مورد نیاز برای استقرار مدل تنظیم شده

۱. خروجی مدل: مدل تنظیم شده را برای استقرار در یک فرمت مناسب (مانند TensorFlow، ONNX، PyTorch، یا SavedModel) ذخیره کنید.
۲. راهاندازی زیرساخت: محیط استقرار را آماده کنید که شامل سخت افزارهای لازم، خدمات ابری و ابزارهای کانتینر سازی باشد.
۳. توسعه API: API‌هایی ایجاد کنید تا برنامه‌ها بتوانند با مدل تعامل داشته باشند و درخواست‌ها و پاسخ‌های پیش‌بینی را تسهیل کنند.
۴. استقرار: مدل را به محیط تولید استقرار دهید و آن را برای کاربران نهایی یا برنامه‌ها قابل دسترسی کنید.

#### هشت - دو: ارائه دهنده‌گان مبتنی بر ابر برای استقرار LLM

استفاده از استنتاج مدل‌های زبانی بزرگ (LLM) مبتنی بر ابر<sup>۱۳۳</sup>، معمولاً از یک مدل قیمت‌گذاری بر اساس تعداد توکن‌های پردازش شده استفاده می‌کند. کاربران بر اساس حجم متنی که مدل تجزیه و تحلیل یا تولید می‌کند، هزینه پرداخت می‌کنند. در حالی که این ساختار قیمت‌گذاری می‌تواند برای استفاده‌های مقطعی یا کوچک مقیاس مقرن به صرفه باشد، اما ممکن است برای بارهای کاری بزرگ‌تر یا مستمر همیشه اقتصادی نباشد. در برخی سناریوهای میزبانی یک راه حل LLM به صورت داخلی ممکن است صرفه‌جویی هزینه بیشتری در طولانی‌مدت ارائه دهد، به ویژه اگر استفاده مداوم یا با حجم بالا وجود داشته باشد. مدیریت زیرساخت خودتان کنترل بیشتری بر تخصیص منابع فراهم می‌کند و بهینه‌سازی هزینه را بر اساس نیازهای خاص امکان‌پذیر می‌سازد. علاوه بر این، میزبانی خود مزایایی از نظر حریم خصوصی و امنیت داده‌ها به همراه دارد، زیرا اطلاعات حساس در محیط خود شما باقی می‌ماند.

<sup>133</sup> Cloud-Based Large Language Model

با این حال، ارزیابی دقیق هزینه کل مالکیت هزینه مقایسه راه حل های مبتنی بر ابر با گزینه های میزبانی خود ضروری است. این ارزیابی باید عواملی مانند هزینه های سخت افزاری، نگهداری و هزینه های عملیاتی را در نظر بگیرد. در نهایت، تصمیم گیری باید بر اساس یک تحلیل جامع هزینه - فایده باشد که هم به قابلیت پرداخت کوتاه مدت و هم به پایداری بلند مدت توجه کند. چندین شرکت خدمات استقرار برای مدل های زبانی بزرگ (LLM) ارائه می دهند که ابزارها و پلتفرم های مختلفی برای پیاده سازی و مدیریت کارآمد این مدل ها فراهم می کنند. در اینجا فهرستی مفصل از برخی ارائه دهنده اگان برجسته و خدمات آن ها آورده شده است:

#### - خدمات وب آمازون (AWS)

**Amazon Bedrock** – این سرویس مجموعه ای از مدل های بنیادی شامل Amazon Titan را ارائه می دهد که از وظایف مختلف NLP مانند خلاصه سازی و تولید متن پشتیبانی می کند. به صورت یکپارچه با سایر خدمات AWS برای استقرار مقیاس پذیر و ایمن ادغام می شود.

**Amazon SageMaker** – یک سرویس یادگیری ماشین جامع را ارائه می دهد که شامل ابزارهایی برای ساخت، آموزش و استقرار LLM ها است. SageMaker JumpStart مدل های پیش آموزش دیده و راهنمایی های گام به گام برای ساده سازی فرآیند استقرار ارائه می دهد.

– آموزش: این آموزش<sup>۱۳۴</sup> استقرار LLM Agents را بر روی Amazon Bedrock توضیح می دهد. آموزش<sup>۱۳۵</sup> دیگری است که تنظیم دقیق و استقرار LLM ها را با Amazon Sagemaker Canvas و Amazon Bedrock به طور کامل توضیح می دهد. دستورالعمل های عمومی Amazon Bedrock برای کاربران LLM را می توان در اینجا<sup>۱۳۶</sup> پیدا کرد.

#### - مایکروسافت آژور

**OpenAI آژور**: این سرویس دسترسی به مدل های قدرتمند OpenAI ۳.۵ و GPT-۳.۵ را ارائه می دهد. این خدمات قابلیت های جاسازی<sup>۱۳۷</sup>، تولید تصویر با DALL-E و تبدیل گفتار به متن با Whisper را فراهم می کند. ادغام آژور با مدل های OpenAI گزینه های استقرار قوی را برای برنامه های مختلف تضمین می کند.

– یادگیری ماشین آژور: از استقرار مدل های سفارشی و پیش آموزش دیده پشتیبانی می کند و ابزارهایی برای مدیریت مدل، استقرار و نظارت ارائه می دهد. این سرویس با اکوسیستم وسیع تر آژور برای عملیات مقیاس پذیر و ایمن ML ادغام می شود.

<sup>134</sup> <https://docs.aws.amazon.com/bedrock/latest/userguide/agents-deploy.html>

<sup>135</sup> <https://aws.amazon.com/blogs/machine-learning/fine-tune-and-deploy-language-models-with-amazon-sagemaker-canvas-and-amazon-bedrock/>

<sup>136</sup> <https://docs.aws.amazon.com/bedrock/latest/userguide/what-is-bedrock.html>

<sup>137</sup> Embedding

- آموزش: در این [لینک](#)<sup>۱۳۸</sup> راهنمای آموزشی برای ایجاد و استقرار یک خدمات OpenAI آژور در پلتفرم مایکروسافت آژور است.

#### - پلتفرم ابری گوگل (GCP)

**Vertex AI** – این پلتفرم امکان استقرار مدل‌های زبانی بزرگ را با ابزارهایی برای آموزش، تنظیم و سرویس‌دهی به مدل‌ها فراهم می‌کند. Vertex AI از مدل‌هایی مانند BERT و GPT-3 پشتیبانی می‌کند و قابلیت‌های گسترده MLOps برای مدیریت انتها به انتهای را ارائه می‌دهد.

**API** – **هوش مصنوعی ابری API**: API‌هایی برای وظایف NLP مانند ترجمه، تحلیل احساسات و شناسایی موجودیت‌ها ارائه می‌دهد. این API‌ها با زیرساخت قدرتمند گوگل پشتیبانی می‌شوند و عملکرد و قابلیت اطمینان بالایی را تضمین می‌کنند.

- آموزش: در [اینجا](#)<sup>۱۳۹</sup> می‌توانید داکیومنتی شامل یک آموزش برای آموزش و استقرار یک LLM در GCP است مشاهده کنید.

#### Hugging Face –

- **Inference API**<sup>۱۴۰</sup> – این سرویس به کاربران اجازه می‌دهد تا LLM‌ها را که بر روی زیرساخت هاگینگ فیس میزبانی می‌شوند، استقرار و مدیریت کنند. این سرویس از مدل‌های مختلف کتابخانه Transformers پشتیبانی می‌کند و یک API آسان برای استفاده برای ادغام این مدل‌ها در برنامه‌ها فراهم می‌کند.

- **فضاهای**: یک محیط تعاملی است که کاربران می‌توانند مدل‌ها را با استفاده از پلتفرم میزبانی هاگینگ فیس استقرار و به اشتراک بگذارند. این محیط از استقرار مدل‌های سفارشی و دموهای تعاملی پشتیبانی می‌کند.

- آموزش: این [داکیومنت](#)<sup>۱۴۱</sup> شامل یک آموزش برای آموزش و استقرار یک LLM با استفاده از API استنتاج هاگینگ فیس است.

#### - پلتفرم‌های دیگر

**OpenLLM** – راه حل‌های استقرار را در این [لینک](#)<sup>۱۴۲</sup> می‌توانید بیابید.

**Deepseed** – راه حل‌های استقرار را در این [لینک](#)<sup>۱۴۳</sup> می‌توانید بیابید.

<sup>138</sup> <https://learn.microsoft.com/en-us/azure/ai-services/openai/how-to/create-resource?pivot=web-portal>

<sup>139</sup> <https://cloud.google.com/vertex-ai/docs/tutorials/tabular-bq-prediction/train-and-deploy-model>

Inference API<sup>۱۴۰</sup> به سرویسی اشاره دارد که مدل را برای پیش‌بینی یا استنتاج (Inference) در اختیار کاربران قرار می‌دهد. این API به کاربران امکان می‌دهد داده‌های جدید را به مدل بفرستند و نتایج پیش‌بینی شده توسط مدل را دریافت کنند، بدون نیاز به اجرای کل فرآیند آموزش مدل.

<sup>141</sup> <https://huggingface.co/blog/inference-endpoints-llm>

<sup>142</sup> <https://github.com/bentoml/OpenLLM?ref=content.whylabs.ai>

<sup>143</sup> <https://github.com/microsoft/DeepSpeed?ref=content.whylabs.ai>

## هشت - سه: تکنیک‌های بهینه‌سازی عملکرد مدل در طول استنتاج

بهینه‌سازی عملکرد مدل در طول استنتاج برای استقرار کارآمد مدل‌های زبان بزرگ (LLM) بسیار حیاتی است. تکنیک‌های پیشرفته زیر استراتژی‌های مختلفی را برای بهبود عملکرد، کاهش تأخیر و مدیریت مؤثر منابع محاسباتی ارائه می‌دهند.

### هشت - سه - یک: استقرار سنتی مبتنی بر GPU در محل<sup>۱۴۴</sup>

این رویکرد سنتی برای استقرار مدل‌های زبان بزرگ (LLM) شامل استفاده از واحدهای پردازش گرافیکی (GPU) به دلیل قابلیت‌های پردازش موازی آن‌ها است که امکان استنتاج سریع و کارآمد را فراهم می‌کند. با این حال، این روش نیاز به سرمایه‌گذاری اولیه در سخت‌افزار دارد و ممکن است برای برنامه‌هایی با تقاضای متغیر یا بودجه محدود مناسب نباشد. استقرار مبتنی بر GPU با چندین چالش مواجه است:

۱. استفاده از منابع ممکن است در دوره‌های تقاضای کم به دلیل وجود سوررهای غیر فعال تحت تأثیر قرار گیرد.

۲. افزایش یا کاهش مقیاس معمولاً نیاز به تغییرات فیزیکی در سخت‌افزار دارد که می‌تواند زمان بر باشد.

۳. سوررهای متمرکز می‌توانند نقاط شکست واحد و محدودیت‌های مقیاس‌پذیری را معرفی کنند.

برای کاهش این مشکلات، استراتژی‌هایی مانند توزیع بار بین چندین GPU، مسیر یابی پشتیبان، موازی‌سازی مدل و موازی‌سازی داده می‌توانند برای دستیابی به نتایج بهتر مورد استفاده قرار گیرند. تکنیک‌های بهینه‌سازی مانند استنتاج توزیع شده با استفاده از Accelerate از PartialState می‌توانند به افزایش کارایی کمک کنند.

#### مثال مورد استفاده: برنامه NLP مقیاس بزرگ

به عنوان مثال، یک پلتفرم بزرگ تجارت الکترونیک استقرار مبتنی بر GPU سنتی را برای مدیریت میلیون‌ها درخواست مشتری در روز پیاده‌سازی کرد. با استفاده از توزیع بار و موازی‌سازی مدل، آن‌ها موفق به کاهش قابل توجه تأخیر و بهبود رضایت مشتری شدند.

### هشت - سه - دو: LLM توزیع شده: استقرار به سبک تورنت و عبور موازی جلو<sup>۱۴۵</sup>

استراتژی نوآورانه برای استقرار مدل‌های زبان بزرگ (LLM) شامل توزیع آن‌ها در چندین GPU به صورت غیرمتمرکز و به سبک تورنت است. کتابخانه‌هایی مانند Petals<sup>146</sup> می‌توانند این کار را انجام دهند. Petals به عنوان یک فرایند غیرمتمرکز طراحی شده است که برای استنتاج سریع شبکه‌های عصبی با تقسیم مدل به بلوک‌ها یا لایه‌های متمایز که در چندین سرور جغرافیایی توزیع می‌شوند، عمل می‌کند. کاربران می‌توانند GPU‌های خود

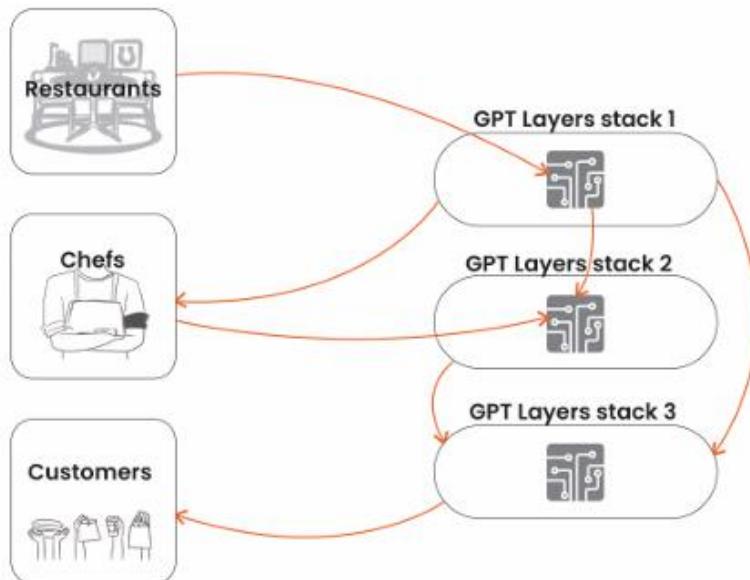
<sup>144</sup> Traditional On-Premises GPU-Based Deployments

<sup>145</sup> Torrent-Style Deployment and Parallel Forward Passes

<sup>146</sup> <https://github.com/bigscience-workshop/petals>

را به این شبکه متصل کنند و به عنوان همکاران و مشتریانی عمل کنند که می‌توانند به مدل دسترسی پیدا کرده و آن را به داده‌های خود اعمال کنند.

هنگامی که یک درخواست مشتری دریافت می‌شود، شبکه آن را از طریق مجموعه‌ای از سرورهای بهینه‌سازی شده برای به حداقل رساندن زمان کل عبور جلو مسیریابی می‌کند. هر سرور به طور پویا بهترین مجموعه بلوك‌ها را انتخاب می‌کند و به گلوگاه‌های فعلی در مسیر سازگار می‌شود. این چارچوب از اصول غیرمت مرکزسازی برای توزیع بار محاسباتی در مناطق مختلف بهره می‌برد و منابع محاسباتی و GPU‌ها را به گونه‌ای به اشتراک می‌گذارد که بار مالی بر روی سازمان‌های فردی کاهش یابد. این رویکرد تعاملی نه تنها بهینه‌سازی استفاده از منابع را تسهیل می‌کند بلکه جامعه جهانی را برای دستیابی به اهداف مشترک هوش مصنوعی تقویت می‌کند.



شکل ۸.۱: نمای مفهومی استقرار توزیع شده LLM با استفاده از رویکرد به سبک تورنت. این شکل استقرار توزیع شده یک مدل زبان بزرگ (LLM) را با استفاده از رویکرد به سبک تورنت نشان می‌دهد که در آن لایه‌های متعدد مدل GPT (پشت‌های) در نقاط مختلف (که به عنوان سرآشپزها نمایش داده شده‌اند) توزیع می‌شوند و عبورهای موازی جلو را انجام می‌دهند. این فرایند جریان سفارشات از مشتریان (داده‌های ورودی) از طریق رستوران‌ها (لایه‌های پردازش میانی) به سرآشپزها (لایه‌های مدل) را شبیه‌سازی می‌کند و کارآمدی پردازش موازی و محاسبات توزیع شده در مدیریت مدل‌های زبان مقیاس بزرگ را به تصویر می‌کشد. این رویکرد برای کاهش تأخیر استنتاج و بهبود مقیاس پذیری LLM‌ها در محیط‌های محاسباتی متتنوع ضروری است. (منبع: [۸۱])

### مثال مورد استفاده: همکاری جهانی تحقیقاتی

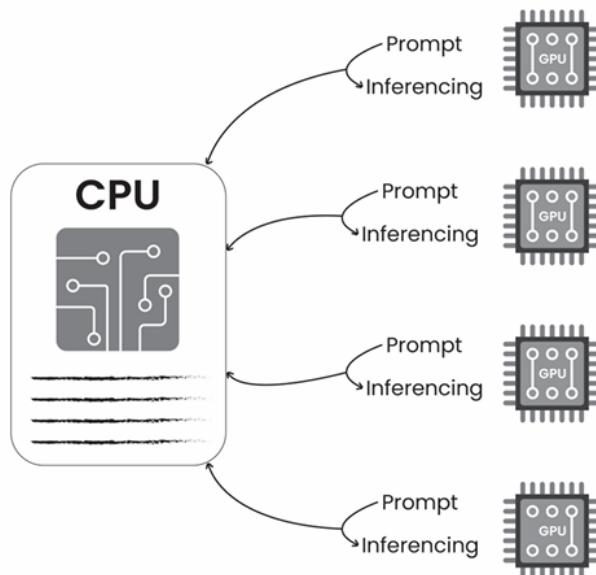
یک کنسرسیوم از مؤسسات تحقیقاتی از فریمورک Petals برای پیاده‌سازی یک LLM توزیع شده به منظور تحلیل داده‌های بزرگ در قاره‌های مختلف استفاده کرد. با بهره‌برداری از ماهیت غیرمت مرکز Petals، آن‌ها به کارایی بالایی در پردازش و توسعه مدل‌های تعاملی دست یافتند.

## هشت - سه - استقرار LLM مبتنی بر WebGPU

این گزینه استقرار برای مدل‌های زبان بزرگ (LLM) شامل استفاده از WebGPU، یک استاندارد وب است که رابطی سطح پایین برای برنامه‌های گرافیکی و محاسباتی در پلتفرم وب فراهم می‌کند. با WebGPU، سازمان‌ها می‌توانند قدرت GPU‌ها را به طور مستقیم در مرورگرهای وب خود بهره‌برداری کنند و استنتاج کارآمدی را برای LLM‌ها در برنامه‌های مبتنی بر وب ممکن سازند. WebGPU محاسبات با عملکرد بالا و رندرینگ گرافیک را به طور مستقیم در مرورگر وب مشتری امکان‌پذیر می‌کند. این فناوری به توسعه‌دهندگان اجازه می‌دهد از GPU مشتری برای کارهایی مانند رندرینگ گرافیک، شتاب دادن به بارهای محاسباتی و انجام پردازش موازی استفاده کنند، بدون نیاز به پلاگین‌ها یا نصب‌های نرم‌افزاری اضافی. این قابلیت اجازه می‌دهد محاسبات پیچیده به طور مؤثر بر روی دستگاه مشتری انجام شود که منجر به وب‌اپلیکیشن‌های سریع‌تر و پاسخگو‌تر می‌شود.

## هشت - سه - چهار: LLM بر روی WebGPU با استفاده از WebLLM

مشتریان می‌توانند به مدل‌های زبان بزرگ و ربات‌های چت قدرتمند به طور مستقیم در مرورگر خود دسترسی پیدا کنند و از شتاب‌دهی WebGPU بهره‌برداری کنند. این رویکرد وابستگی به سرورها را از بین می‌برد و عملکرد فوق العاده و حریم خصوصی بهبودیافته‌ای را برای کاربران فراهم می‌کند. WebLLM استفاده از مدل‌های زبان بزرگ را به طور مستقیم در مرورگر مشتری تسهیل می‌کند تا کارهایی مانند فیلتر کردن اطلاعات شناسایی شخصی (PII) یا شناسایی موجودیت‌های نامدار (NER) بر روی داده‌ها بدون ارسال آن‌ها از طریق شبکه انجام شود. این امر حریم خصوصی و امنیت بهبود یافته‌ای را با حفظ اطلاعات حساس در سمت مشتری تضمین می‌کند.



شکل ۸.۲: استقرار LLM مبتنی بر WebGPU این نمودار معماری استقرار یک مدل زبان بزرگ (LLM) را با استفاده از فناوری WebGPU نشان می‌دهد. CPU وظیفه مدیریت توزیع وظایف استنتاج prompt را به چندین GPU بر عهده دارد که سپس این prompt‌ها را به صورت موازی پردازش می‌کنند و کارایی و مقیاس‌پذیری را در استقرار LLM‌ها در پلتفرم‌های وب افزایش می‌دهند. (منبع [81])

## موارد استفاده اضافی برای WebLLM

۱. ترجمه زبان: امکان ترجمه همزمان متن به طور مستقیم در مرورگر را فراهم می‌کند و به کاربران اجازه می‌دهد تا بدون انتقال پیام‌های خود به شبکه، از موانع زبانی عبور کنند.
۲. تکمیل خودکار کد: توسعه ویرایشگرهای کدی که پیشنهادات تکمیل خودکار هوشمند را بر اساس زمینه ارائه می‌دهند و از WebLLM برای درک و پیش‌بینی قطعات کد استفاده می‌کنند.
۳. چت‌بات‌های پشتیبانی مشتری: پیاده‌سازی چت‌بات‌ها در وبسایت‌ها برای ارائه پشتیبانی فوری به مشتریان و پاسخ به سوالات متداول بدون وابستگی به سرورهای خارجی.
۴. تحلیل و تجسم داده: ایجاد ابزارهای مبتنی بر مرورگر برای تحلیل و تجسم داده‌ها، که WebLLM در پردازش داده، تفسیر و تولید بینش‌ها کمک می‌کند.
۵. توصیه‌های شخصی‌سازی شده: توسعه موثرهای توصیه که پیشنهادات محصول شخصی‌سازی شده، پیشنهادات محتوا یا توصیه‌های فیلم/موسیقی را بر اساس ترجیحات و رفتار کاربران ارائه می‌دهند.
۶. تحلیل‌های حفظ حریم خصوصی: توسعه پلتفرم‌های تحلیلی که تجزیه و تحلیل داده را به طور مستقیم در مرورگر انجام می‌دهند و اطمینان حاصل می‌کنند که اطلاعات حساس در سمت مشتری باقی می‌ماند و خطر نقض داده‌ها کاهش می‌یابد.

**مثال مورد استفاده: اپلیکیشن وب با تمرکز بر حفظ حریم خصوصی**

یک استارت‌آپ بهداشت و درمان LLM را با استفاده از WebLLM برای پردازش اطلاعات بیماران به طور مستقیم در مرورگر مستقر کرد و از حفظ حریم خصوصی داده‌ها و رعایت مقررات بهداشتی اطمینان حاصل کرد. این رویکرد به طور قابل توجهی خطر نقض داده‌ها را کاهش داد و اعتماد کاربران را بهبود بخشید.

## هشت - سه - پنج: LLM‌های کمی‌شده

کاهش اندازه مدل‌های AI با نمایندگی پارامترهای آن‌ها با بیت‌های کمتری، تکنیکی به نام کمی‌سازی است. در مدل‌های یادگیری ماشین سنتی، هر پارامتر (مانند وزن‌ها و بایاس‌ها در شبکه‌های عصبی) معمولاً به عنوان یک عدد اعشاری ۳۲ بیتی ذخیره می‌شود که نیاز به منابع قابل توجهی از حافظه و محاسبات دارد، بهویژه برای مدل‌های بزرگ. کمی‌سازی هدف دارد تا با کاهش دقت این پارامترها این نیاز را کاهش دهد. برای مثال، مانند اعداد ذخیره هر پارامتر به عنوان یک عدد اعشاری ۳۲ بیتی، ممکن است با استفاده از بیت‌های کمتری، مانند اعداد صحیح ۸ بیتی، نمایندگی شوند. این فشرده‌سازی اندازه حافظه مدل را کاهش می‌دهد و استقرار و اجرای آن را بهویژه در محیط‌های محدود از نظر منابع دستگاه‌های موبایل یا دستگاه‌های لبه‌ای کارآمدتر می‌کند. QLoRA یک نمونه محبوب از این کمی‌سازی برای LLM‌ها است و می‌تواند برای استقرار LLM‌ها به طور محلی یا میزبانی آن‌ها بر روی سرورهای خارجی استفاده شود.

## مثال مورد استفاده: استقرار در دستگاه لبه‌ای<sup>۱۴۷</sup>

یک شرکت فناوری از LLM های کمی‌شده برای استقرار مدل‌های NLP پیشرفته بر روی دستگاه‌های موبایل استفاده کرد و امکان عملکرد آفلاین را برای برنامه‌هایی مانند شناسایی صدا و ترجمه فراهم کرد. این استقرار به‌طور قابل توجهی عملکرد برنامه و تجربه کاربر را با کاهش تأخیر و وابستگی به اتصال اینترنت بهبود بخشید.

## هشت - سه - شش: vLLM‌ها

سیستم vLLM<sup>148</sup> با استفاده از روش مدیریت حافظه به صورت بلوکی و زمان‌بندی پیش‌گیرانه درخواست‌ها به‌طور مؤثری به درخواست‌ها پاسخ می‌دهد. این سیستم از الگوریتم PagedAttention [۸۲] برای مدیریت حافظه کلید-مقدار (KV) استفاده می‌کند و بدین ترتیب اتلاف و تکه‌تکه شدن حافظه را کاهش می‌دهد. با دسته‌بندی درخواست‌ها و اشتراک گذاری بلوک‌های فیزیکی در بین چندین نمونه، vLLM مصرف حافظه را بهینه کرده و از توان عملیاتی بیشتری برخوردار می‌شود. آزمایش‌های عملکرد نشان می‌دهند که vLLM در سناریوهای مختلف رمزگشایی، از سایر سیستم‌ها پیشی می‌گیرد. برای مثال، در نظر بگیرید که یک مدل مبتنی بر ترانسفورمر موظف است یک کتاب طولانی را خلاصه کند. ترانسفورمرهای سنتی کل کتاب را به‌طور همزمان پردازش می‌کنند که می‌تواند هم از نظر محاسباتی و هم از نظر حافظه‌بر باشد، بهویژه برای متون طولانی. با استفاده از PagedAttention، کتاب به بخش‌های کوچک‌تر یا صفحات تقسیم می‌شود. سپس مدل بر روی خلاصه‌سازی یک صفحه در هر بار تمرکز می‌کند، نه اینکه کل کتاب را به‌طور همزمان خلاصه کند. این رویکرد پیچیدگی محاسباتی و نیازهای حافظه را کاهش می‌دهد و پردازش و خلاصه‌سازی متون طولانی را به‌طور مؤثری ممکن می‌سازد.

## مثال مورد استفاده: تولید محتوا با حجم بالا

یک آژانس بازاریابی محتوا از vLLM ها برای تولید حجم زیادی از محتوای بهینه‌شده برای SEO استفاده کرد. با بهره‌برداری از مدیریت حافظه مؤثر vLLM ها، آن‌ها قادر به مدیریت چندین درخواست همزمان بودند و به‌طور قابل توجهی نرخ تولید محتوا خود را در حالی که کیفیت را حفظ می‌کردند، افزایش دادند.

## هشت - چهار: ملاحظات کلیدی برای استقرار LLM ها

استقرار مؤثر مدل‌های زبان بزرگ (LLM) ها نیازمند برنامه‌ریزی دقیق و در نظر گرفتن عوامل مختلف برای اطمینان از عملکرد بهینه، کارایی هزینه و امنیت است. ملاحظات کلیدی شامل موارد زیر است:

<sup>147</sup> Edge Device Deployment

<sup>148</sup> <https://docs.vllm.ai/en/stable/>

## - نیازمندی‌های زیرساخت:

**منابع محاسباتی:** اطمینان از وجود منابع CPU/GPU کافی برای پاسخگویی به نیازهای محاسباتی مدل. معمولاً برای استنتاج و آموزش مؤثر، نیاز به GPU های با عملکرد بالا است.

**حافظه:** LLMها، بهویژه آن‌هایی که میلیارد‌ها پارامتر دارند، به حافظه قابل توجهی نیاز دارند. تکنیک‌های مدیریت حافظه مانند کمی‌سازی و موازی‌سازی مدل می‌تواند برای بهینه‌سازی استفاده از حافظه به کار رود.

## - مقیاس‌پذیری:

**مقیاس‌گذاری افقی:** برنامه‌ریزی برای مقیاس‌گذاری افقی به منظور توزیع بار بر روی چندین سرور که می‌تواند عملکرد را بهبود بخشد و به افزایش تقاضا پاسخ دهد.

**توزیع بار:** پیاده‌سازی استراتژی‌های توزیع بار برای اطمینان از توزیع یکنواخت درخواست‌ها و جلوگیری از هر نقطه‌ای از شکست.

## - مدیریت هزینه:

**قیمت‌گذاری مبتنی بر توکن:** درک عواقب هزینه‌ای مدل‌های قیمت‌گذاری مبتنی بر توکن که توسط ارائه‌دهندگان ابری پیشنهاد می‌شود. این مدل بر اساس تعداد توکن‌های پردازش شده هزینه می‌گیرد که با افزایش استفاده می‌تواند هزینه‌بر شود.

**میزبانی خود:** ارزیابی هزینه‌ها و مزایای میزبانی خود در برابر میزبانی ابری. میزبانی خود ممکن است برای استفاده‌های مداوم و با حجم بالا در طولانی‌مدت صرفه‌جویی کند، اما نیاز به سرمایه‌گذاری اولیه قابل توجهی در سخت‌افزار و نگهداری مداوم دارد.

## - بهینه‌سازی عملکرد:

**تاخیر:** کاهش تاخیر به منظور اطمینان از عملکرد همزمان، بهویژه برای برنامه‌هایی که نیاز به پاسخ‌های فوری مانند چت‌بات‌ها و دستیارهای مجازی دارند.

**توان عملیاتی:** حداکثر کردن توان عملیاتی برای مدیریت کارآمد حجم بالای درخواست‌ها. تکنیک‌هایی مانند دسته‌بندی و مدیریت مؤثر حافظه مانند (PagedAttention) می‌توانند کمک کنند.

## - امنیت و حریم خصوصی:

**امنیت داده:** تدبیر امنیتی قوی برای حفاظت از داده‌های حساس پیاده‌سازی کنید، از جمله رمزگذاری و کنترل‌های دسترسی امن.

**حریم خصوصی:** اطمینان حاصل کنید که با مقررات حریم خصوصی داده‌ها رعایت شده و در صورت میزبانی خود، داده‌های حساس در محیط شما باقی بمانند یا اطمینان حاصل کنید که ارائه‌دهندگان خدمات ابری با استانداردهای حریم خصوصی مربوطه سازگار هستند.

## - نگهداری و بهروزرسانی‌ها:

بهروزرسانی مدل: مدل را به طور منظم بهروزرسانی کنید تا داده‌های جدید را گنجانده و عملکرد را بهبود ببخشید. در صورت امکان، این فرایند را خودکار کنید تا تلاش‌های دستی کاهش یابد.

نگهداری سیستم: برنامه‌ریزی برای نگهداری منظم زیرساخت به منظور جلوگیری از زمان‌های غیرفعال و اطمینان از عملکرد روان.

## - انعطاف‌پذیری و شخصی‌سازی:

تنظیم دقیق: اجازه دهید مدل تنظیم دقیق شود تا LLM به موارد استفاده خاص و مجموعه‌های داده خاص سازگار شود. تنظیم دقیق می‌تواند دقت و ارتباط پاسخ‌ها را بهبود بخشد.

یکپارچه‌سازی API: اطمینان حاصل کنید که پلتفرم استقرار از یکپارچه‌سازی آسان با سیستم‌ها و جریان‌های کاری موجود از طریق API‌ها و SDK‌ها پشتیبانی می‌کند.

## - مدیریت کاربران:

کنترل دسترسی: کنترل دسترسی مبتنی بر نقش را پیاده‌سازی کنید تا مدیریت کنید که چه کسانی می‌توانند LLM را استقرار، استفاده و نگهداری کنند.

نظرارت و ثبت‌گذاری: نظرارت و ثبت‌گذاری جامع را راهاندازی کنید تا استفاده، عملکرد و مشکلات احتمالی را پیگیری کنید. این به رفع اشکال و بهینه‌سازی پیشگیرانه کمک می‌کند.

## - رعایت قوانین:

رعایت مقررات: اطمینان حاصل کنید که استقرار با تمام الزامات قانونی و مقرراتی مربوطه، از جمله قوانین حفاظت از داده‌ها مانند GDPR، HIPAA و غیره سازگار است.

ملاحظات اخلاقی: راهنمایی‌های اخلاقی را پیاده‌سازی کنید تا از بروز تعصبات جلوگیری کرده و استفاده مسئولانه از LLM‌ها را تضمین کنید.

## - پشتیبانی و مستندات:

پشتیبانی فنی: پلتفرم استقراری را انتخاب کنید که پشتیبانی فنی و منابع قوی ارائه دهد.

مستندسازی: مستندات جامعی برای توسعه‌دهندگان و کاربران ارائه دهید تا استقرار و استفاده روان را تسهیل کنید.

فصل نهم

## مرحله هفتم: نظارت و نگهداری

#### نه - پیک: مراحل مربوط به نظارت و نگهداری LLM‌های تنظیم شده

نظرات و نگهداری مداوم LLM‌های تنظیم شده برای اطمینان از عملکرد بهینه، دقت و امنیت آن‌ها در طول زمان ضروری است. در زیر مراحل کلیدی مربوط به این فرآیند آمده است:

۱. ایجاد مبنای اولیه: مبنای عملکرد اولیه را با ارزیابی مدل بر روی یک مجموعه داده تست جامع تعیین کنید. معیارهایی مانند دقت، تأخیر، توان عملیاتی و نرخ خطأ را ضبط کنید تا به عنوان نقاط مرجع برای نظارت آینده استفاده شوند.

۲. نظارت بر عملکرد: سیستم‌هایی را پیاده‌سازی کنید تا به‌طور مداوم معیارهای کلیدی عملکرد مانند زمان پاسخ، بار سرور و استفاده از توکن‌ها را پیگیری کنند. این معیارها را به‌طور منظم با مبنای تعیین شده مقایسه کنید تا هر گونه انحرافی شناسایی شود.

۳. نظارت بر دقت: پیش‌بینی‌های مدل را به طور مداوم در مقایسه با یک مجموعه داده حقیقت زمین ارزیابی کنید. از معیارهایی مانند دقت، فراخوان، امتیاز  $F_1$  و از دست دادن متقاطع برای اطمینان از حفظ سطوح بالای دقت مدل استفاده کنید.

۴. نظارت بر خطای از جمله خطاهای زمان اجرا و خطاهای پیش‌بینی را پیگیری و تحلیل کنید. مکانیزم‌های ثبت‌گذاری را پیاده‌سازی کنید تا اطلاعات دقیقی درباره هر خطای برای عیب‌یابی و بهبود جمع‌آوری شود.

۵. تحلیل لاغ: لاغ‌های جامع برای هر درخواست و پاسخ پیش‌بینی، شامل داده‌های ورودی، پیش‌بینی‌های خروجی، زمان‌های پاسخ و خطاهای مواجه شده حفظ کنید. به‌طور منظم لاغ‌ها را مرور کنید تا الگوهای منتهه‌های بهبود شناسایی شوند.

۶. مکانیسم‌های هشدار: سیستم‌های هشدار خودکار را برای اطلاع‌رسانی به ذینفعان در مورد هر گونه ناهنجاری یا انحراف از معیارهای عملکرد پیش‌بینی شده راهاندازی کنید. هشدارها را با ابزارهای ارتباطی مانند PagerDuty، Slack یا ایمیل، برای یاسخهای به موقع ادغام کنید.

۷. **چرخه بازخورد:** یک چرخه بازخورد با کاربران نهایی ایجاد کنید تا بینش‌هایی درباره عملکرد مدل و رضایت کاربر جمع‌آوری کنید. از این بازخورد برای بهبود و تنظیم مدل به‌طور مداوم استفاده کنید.
۸. **ناظارت بر امنیت:** تدابیر امنیتی قوی را برای ناظارت بر تهدیدات، از جمله دسترسی غیرمجاز، نقض داده‌ها و حملات خصم‌مانه پیاده‌سازی کنید. از رمزگذاری، کنترل دسترسی و ممیزی‌های امنیتی منظم برای حفاظت از مدل و داده‌ها استفاده کنید.
۹. **شناسایی انحراف:** به‌طور مداوم برای انحراف داده و مفهوم با استفاده از آزمون‌های آماری و شناسایی‌کننده‌های انحراف ناظارت کنید. به‌طور منظم مدل را بر روی مجموعه داده‌های نگه‌داری شده ارزیابی کنید تا تغییرات در توزیع داده ورودی یا عملکرد مدل شناسایی شود.
۱۰. **نسخه‌بندی مدل:** کنترل نسخه را برای نسخه‌های مختلف مدل حفظ کنید. معیارهای عملکرد هر نسخه را پیگیری کنید تا اطمینان حاصل شود که بهترین مدل در حال تولید است.
۱۱. **مستندسازی و گزارش‌گیری:** مستندات دقیقی از رویه‌های ناظارتی، معیارها و یافته‌ها نگه‌داری کنید. گزارش‌های منظم تولید کنید تا بینش‌هایی درباره عملکرد مدل و فعالیت‌های نگه‌داری به ذینفعان ارائه دهید.
۱۲. **بازبینی و بهروزرسانی دوره‌ای:** به‌طور منظم فرآیندهای ناظارتی را ارزیابی و بهروزرسانی کنید تا تکنیک‌ها، ابزارها و بهترین شیوه‌های جدید را در بر بگیرد و اطمینان حاصل کنید که سیستم ناظارتی مؤثر و به‌روز باقی می‌ماند.

## نه – دو: ناظارت مداوم بر عملکرد مدل

در حالی که برنامه‌های کاربردی مدل‌های زبانی بزرگ (LLM) تحت ارزیابی‌هایی قرار می‌گیرند، ناظارت مداوم در بیشتر موارد به‌طور کافی پیاده‌سازی نشده است. این بخش اجزای لازم برای ایجاد یک برنامه ناظارتی مؤثر که هدف آن محافظت از کاربران و حفظ یکپارچگی برنده است را شرح می‌دهد.

### نه – دو – یک: ناظارت عملکردی

در ابتدا، ناظارت مداوم بر معیارهای اساسی بسیار مهم است. این شامل پیگیری معیارهایی مانند حجم درخواست، زمان پاسخ، استفاده از توکن، هزینه‌های متحمل شده و نرخ خطاهای است.

### نه – دو – دو: ناظارت بر درخواست‌ها

پس از ناظارت بر معیارهای عملکردی، باید توجه به ناظارت بر درخواست‌ها یا ورودی‌های تولیدشده توسط کاربر معطوف شود. معیارهایی مانند خوانایی می‌توانند بینش‌های ارزشمندی ارائه دهند. ارزیابهای LLM باید برای شناسایی پتانسیل سمی بودن در پاسخ‌ها به کار گرفته شوند. علاوه بر این، معیارهایی مانند فاصله‌های تعبیه شده از درخواست‌های مرجع، اطمینان حاصل می‌کنند که مدل به تعاملات مختلف کاربر در طول زمان انطباق

دارد. معرفی یک دسته ارزیابی جدید شامل شناسایی تلاش‌های خصم‌مانه یا تزریق درخواست‌های مخرب است که غالباً در ارزیابی‌های اولیه نادیده گرفته می‌شوند. مقایسه با مجموعه‌های مرجع از درخواست‌های خصم‌مانه شناخته شده به شناسایی و علامت‌گذاری فعالیت‌های مخرب کمک می‌کند. LLM‌های ارزیابی‌کننده نقش حیاتی در طبقه‌بندی درخواست‌ها به عنوان بی‌خطر یا مخرب ایفا می‌کنند.

## نه – دو – سه: نظارت بر پاسخ‌ها

نظارت بر پاسخ‌ها شامل چندین بررسی حیاتی است تا اطمینان حاصل شود که با نتایج مورد انتظار هم راستا هستند. پارامترهایی مانند ارتباط، انسجام (halosineness<sup>۱۴۹</sup>)، تطابق موضوعی، احساس و تکامل آن‌ها در طول زمان بسیار مهم هستند. معیارهای مرتبط با سمی بودن و خروجی‌های مضر به دلیل تأثیرات بحرانی آن‌ها، نیاز به نظارت مکرر دارند. نشت درخواست<sup>۱۵۰</sup> نمایانگر یک تاکتیک خصم‌مانه است که در آن اطلاعات حساس درخواست به‌طور غیرمجاز از داده‌های ذخیره‌شده در برنامه استخراج می‌شود. نظارت بر پاسخ‌ها و مقایسه آن‌ها با پایگاه داده‌ای از دستورالعمل‌های درخواست می‌تواند به شناسایی این نوع نقض‌ها کمک کند. معیارهای فاصله تعییه‌شده به‌طور خاص در این زمینه مؤثر هستند. آزمایش‌های منظم در برابر مجموعه داده‌های ارزیابی نقاط مرجع برای دقیق فراهم می‌کند و هر گونه انحراف عملکردی را در طول زمان نشان می‌دهد. ابزارهایی که قادر به مدیریت تعییه‌ها هستند، اجازه می‌دهند مجموعه داده‌های خروجی با عملکرد ضعیف به منظور بهبودهای هدفمند صادر شوند.

## نه – دو – چهار: مکانیسم‌ها و آستانه‌های هشدار

نظارت مؤثر نیاز به آستانه‌های هشدار به‌خوبی تنظیم شده دارد تا از هشدارهای کاذب زیاد جلوگیری شود. پیاده‌سازی تشخیص انحراف چندمتغیره و مکانیسم‌های هشدار می‌تواند دقیق را افزایش دهد. در نظر گرفتن نرخ هشدارهای کاذب و بهترین شیوه‌ها برای تنظیم آستانه‌ها برای طراحی مؤثر سیستم نظارت بسیار حائز اهمیت است. ویژگی‌های هشدار باید شامل ادغام با ابزارهای ارتباطی مانند Slack و PagerDuty باشد. برخی از سیستم‌ها در صورت بروز هشدارهای ناشی از درخواست‌های مشکل‌زا، قابلیت مسدودسازی خودکار پاسخ‌ها را ارائه می‌دهند. مکانیسم‌های مشابه می‌توانند برای اسکرین کردن پاسخ‌ها به منظور اطلاعات شناسایی شخصی (PII)، سمی بودن و سایر معیارهای کیفیت قبل از تحويل به کاربران به کار گرفته شوند. معیارهای سفارشی متناسب با ظرایف خاص برنامه یا بینش‌های نوآورانه از دانشمندان داده می‌توانند به‌طور قابل توجهی اثربخشی نظارت را افزایش دهند. انعطاف‌پذیری برای گنجاندن چنین معیارهایی برای انطباق با نیازهای نظارتی در حال تحول و پیشرفت‌ها در این زمینه ضروری است.

<sup>۱۴۹</sup> Hallucination

<sup>۱۵۰</sup> Prompt Leakage

## نه - دو - پنج: نظارت بر رابط کاربری (UI)

رابط کاربری سیستم نظارت حیاتی است و معمولاً شامل نمودارهای زمان‌سرب از معیارهای نظارت شده است. رابطهای کاربری تفکیک شده امکان تحلیل عمیق‌تری از روندهای هشدار را فراهم می‌آورند و به تحلیل ریشه‌ای کمک می‌کنند. قابلیت‌های پیشرفت‌های رابط کاربری ممکن است شامل تجسم‌های فضاهای تعییه شده از طریق خوشبندی و پیش‌بینی‌ها باشد که بینش‌هایی در مورد الگوها و روابط داده ارائه می‌دهد. سیستم‌های نظارتی پیشرفت‌های داده‌ها را بر اساس کاربران، پژوهش‌ها و تیم‌ها دسته‌بندی می‌کنند و اطمینان حاصل می‌کنند که کنترل دسترسی مبتنی بر نقش (RBAC<sup>151</sup>) برای حفاظت از اطلاعات حساس وجود دارد. بهینه‌سازی تحلیل هشدار در رابط کاربری همچنان حوزه‌ای است که بهبودها می‌توانند به‌طور قابل توجهی نرخ هشدارهای کاذب را کاهش دهند و کارایی عملیاتی را افزایش دهند.

## نه - سه: به روزرسانی دانش LLM

برای بهبود پایگاه دانش یک LLM، پیش‌آموزش مداوم به کار می‌رود تا به LLM کمک کند با جدیدترین دانش و اطلاعات سازگار شود. جهان و زبان به‌طور مداوم در حال تحول هستند. اطلاعات جدید به وجود می‌آید، روندها تغییر می‌کنند و ارجاعات فرهنگی تغییر می‌یابند. LLM‌هایی که بر روی داده‌های ایستا آموزش دیده‌اند ممکن است به روز نباشند که منجر به موارد زیر می‌شود:

- خطاهای واقعی: اطلاعات قدیمی می‌تواند باعث شود که LLM‌ها پاسخ‌های نادرست ارائه دهند.
- بی‌ارتباطی: مدل‌ها ممکن است زمینه رویدادهای جاری را از دست بدهند یا از ارجاعات قدیمی استفاده کنند.
- تداوم تعصبات<sup>152</sup>: تعصبات موجود در داده‌های آموزشی می‌توانند اگر از طریق به روزرسانی‌ها به آن‌ها رسیدگی نشود، ریشه‌دار شوند.

## نه - سه - یک: روش‌های بازآموزی

- بازآموزی دوره‌ای<sup>153</sup>: این شامل به روزرسانی پایگاه دانش مدل در فواصل منظم (هفتگی، ماهانه، سالانه) با داده‌های جدید است. این یک روش ساده است اما نیاز به جریان مداوم داده‌های با کیفیت و بدون تعصب دارد.

<sup>151</sup> Role-Based Access Control (RBAC)

<sup>152</sup> Bias Perpetuation

<sup>153</sup> Periodic Retraining

- بازآموزی مبتنی بر رویداد<sup>۱۵۴</sup>: این رویکرد عملکرد LLM را تحت نظارت قرار می‌دهد. هنگامی که معیارهایی مانند دقت یا ارتباط به زیر آستانه‌ای خاص سقوط کند، فرایнд مجدد آموزش آغاز می‌شود. این روش پویاتر است اما نیاز به سیستم‌های نظارتی قوی و معیارهای عملکرد روشن دارد.

#### نه - سه - دو: روش‌های اضافی

- تنظیم دقیق: LLMها می‌توانند برای وظایف خاص با آموزش آن‌ها بر روی مجموعه داده‌های خاص حوزه‌ای تنظیم دقیق شوند. این امکان تخصصی‌سازی بدون نیاز به آموزش کامل مجدد را فراهم می‌آورد.
- یادگیری فعال (Active Learning): این رویکرد شامل پرسش انتخابی از LLM برای شناسایی حوزه‌هایی است که در آن‌ها از دانش کافی برخوردار نیست. اطلاعات به دست‌آمده سپس برای به روزرسانی مدل استفاده می‌شود.

#### نه - سه - سه: ملاحظات کلیدی

- کیفیت داده و تعصب: داده‌های آموزشی جدید باید با دقت انتخاب شوند تا کیفیت آن‌ها تضمین و تعصب<sup>۱۵۵</sup> کاهش یابد. تکنیک‌هایی مانند حاشیه‌نویسی انسانی و بررسی‌های عدالت اهمیت زیادی دارند.
- هزینه محاسباتی: مجدد آموزش LLMها می‌تواند از نظر محاسباتی پرهزینه باشد و به منابع قابل توجهی نیاز دارد. بهینه‌سازی‌هایی مانند یادگیری انتقالی (استفاده از مدل‌های پیش‌آموزش‌دیده به عنوان نقطه شروع) می‌تواند به کاهش هزینه‌ها کمک کند.
- زمان خاموشی: مجدد آموزش معمولاً زمان بر است و منجر به خاموشی LLM می‌شود. استراتژی‌هایی مانند به روزرسانی‌های تدریجی یا استقرار چندین مدل می‌تواند اختلالات خدمات را به حداقل برساند.
- کنترل نسخه: پیگیری نسخه‌های مختلف LLM و داده‌های آموزشی آن‌ها برای بازگشت به نسخه‌های قبلی در صورت بروز مشکلات عملکردی ضروری است.

#### نه - چهار: آینده به روزرسانی‌های LLM

تحقیقات در حال انجام است تا استراتژی‌های به روزرسانی LLM را کارآمدتر و مؤثرتر توسعه دهد. یکی از زمینه‌های امیدوارکننده، یادگیری مداوم است، جایی که LLMها می‌توانند به طور مداوم از جریان‌های داده جدید یاد بگیرند و خود را تطبیق دهند بدون اینکه نیاز به آموزش از ابتدا داشته باشند. هدف از یادگیری مداوم کاهش نیاز به مجدد آموزش کامل در فواصل زمانی کوتاه است، با این امکان که مدل‌ها بتوانند به طور تدریجی با اطلاعات

<sup>154</sup> Trigger-Based Retraining

<sup>155</sup> Bias

جدید به روزرسانی شوند. این رویکرد می‌تواند به طور چشمگیری توانایی مدل را برای به روز ماندن با دانش و استفاده از زبان در حال تحول افزایش دهد و عملکرد و ارتباط آن را در بلندمدت بهبود بخشد.

نوآوری‌ها در یادگیری انتقالی و یادگیری متأخر به پیشرفت‌های به روزرسانی LLM کمک می‌کنند. این تکنیک‌ها به مدل‌ها اجازه می‌دهند تا از دانش پیشین استفاده کرده و به سرعت به وظایف یا حوزه‌های جدید با حداقل آموزش اضافی سازگار شوند. با ادغام این روش‌های یادگیری پیشرفته، LLM‌های آینده می‌توانند در پردازش و درک اطلاعات جدید، سازگارتر و کارآمدتر شوند. علاوه بر این، بهبودهای مداوم در سخت‌افزار و منابع محاسباتی، به روزرسانی‌های بیشتر و کارآمدتر را پشتیبانی خواهد کرد. با افزایش قدرت پردازش و در دسترس بودن آن، بار محاسباتی به روزرسانی مدل‌های بزرگ کاهش می‌یابد و امکان به روزرسانی‌های منظم و جامع‌تر را فراهم می‌کند. همچنین، همکاری بین دانشگاه‌ها و صنعت برای پیشبرد این پیشرفت‌ها حیاتی است. با به اشتراک گذاشتن یافته‌های تحقیقاتی و بهترین شیوه‌ها، این حوزه می‌تواند به طور جمی به سمت روش‌های به روزرسانی LLM قوی‌تر و کارآمدتر حرکت کند و اطمینان حاصل کند که مدل‌ها در طول زمان دقیق، مرتبط و ارزشمند باقی بمانند.

## فصل دهم

# پلتفرم‌ها و فریمورک‌های صنعتی برای تنظیم دقیق مدل‌های زبانی بزرگ (LLMs)

پیشرفت تکنیک‌های تنظیم دقیق توسط شرکت‌های بزرگ فناوری و پلتفرم‌هایی صورت گرفته است که فریمورک‌ها و خدمات نوآورانه‌ای را معرفی کرده‌اند. شرکت‌هایی مانند Amazon Web Services (AWS)، Microsoft Azure، OpenAI و HuggingFace ابزارها و پلتفرم‌هایی توسعه داده‌اند که فرآیند تنظیم دقیق را ساده و فرآگیر می‌کنند. این پیشرفت‌ها نه تنها ورود به دنیای مدل‌های پیشرفت‌هه هوش مصنوعی را آسان‌تر کرده، بلکه امکان کاربردهای وسیعی در صنایع مختلف از جمله بهداشت و درمان، امور مالی، خدمات مشتری و تولید محظوا را فراهم ساخته است. هر یک از این پلتفرم‌ها قابلیت‌های منحصر‌به‌فردی ارائه می‌دهند که به نیازهای مختلف پاسخ می‌دهد؛ از جمله، گردش کارهای خودکار تنظیم دقیق، محیط‌های آموزشی مبتنی بر ابر و مقیاس‌پذیر و واسطه‌ای کاربرپسند API برای استقرار مدل‌های سفارشی.

برای مثال، HuggingFace با کتابخانه‌ی [Transformers<sup>156</sup>](https://huggingface.co/docs/transformers/en/index) و ابزارهایی مانند SetFit<sup>157</sup> و Autotrain<sup>158</sup> پیشرفت‌های چشمگیری داشته است که به کاربران امکان می‌دهد مدل‌ها را با حداقل کدنویسی و داده تنظیم دقیق کنند. پلتفرم آنها زیرساخت قدرتمندی فراهم می‌کند که هم از جامعه‌ی تحقیقاتی و هم از کاربران صنعتی پشتیبانی می‌کند و توسعه و استقرار سریع راه حل‌های هوش مصنوعی سفارشی را ممکن می‌سازد. به طور مشابه، AWS با SetFit<sup>159</sup> و SageMaker<sup>158</sup> مجموعه‌ای گسترده از خدمات را ارائه می‌دهد که تمام چرخه‌ی یادگیری ماشین را پوشش می‌دهد، از آماده‌سازی داده‌ها و آموزش تا استقرار و بهینه‌سازی مدل، که آن را به یک راه حل جامع برای برنامه‌های سطح سازمانی تبدیل می‌کند.

از سوی دیگر، Microsoft Azure قابلیت‌های تنظیم دقیق خود را با ابزارها و خدمات در سطح سازمانی یکپارچه کرده و راه حل‌هایی مانند Azure OpenAI Service و Azure Machine Learning را ارائه می‌دهد که به سازمان‌های بزرگ کمک می‌کند تا هوش مصنوعی پیشرفت‌ه را در عملیات خود وارد کنند. مرکز Azure

<sup>156</sup> <https://huggingface.co/docs/transformers/en/index>

<sup>157</sup> <https://huggingface.co/autotrain>

<sup>158</sup> <https://aws.amazon.com/sagemaker/>

<sup>159</sup> <https://platform.openai.com/docs/guides/fine-tuning/fine-tuning-integrations>

بر روی MLOps و یکپارچگی بی‌درنگ با سایر خدمات Azure باعث می‌شود که مدل‌های دقیق تنظیم شده به‌طور کارآمد در محیط‌های تولید مستقر و نگهداری شوند. در همین حال، OpenAI با ارائه‌ی مفهوم "تنظیم دقیق به عنوان یک سرویس"<sup>160</sup> به کسب‌وکارها این امکان را می‌دهد تا از مدل‌های قدرتمند خود مانند GPT-4 از طریق یک API<sup>160</sup> کاربرپسند استفاده کنند و مدل‌های سفارشی را بدون نیاز به تخصص داخلی در حوزه‌ی هوش مصنوعی یا زیرساخت‌های پیچیده، سازگار کنند.

تلاش‌های جمعی این شرکت‌های فناوری نه تنها کارایی و مقیاس‌پذیری تنظیم دقیق را بهبود بخشدید، بلکه دسترسی به ابزارهای پیشرفته هوش مصنوعی را فرآگیر و همگانی کرده‌اند. با کاهش موانع فنی و ارائه‌ی پلتفرم‌های جامع و کاربرپسند، این نوآوری‌ها به طیف وسیعی از صنایع امکان داده‌اند تا مدل‌های پیشرفته‌ی هوش مصنوعی را با توجه به نیازهای خاص خود مستقر کنند. جدول‌های ۱۰.۱ و ۱۰.۲ مقایسه‌ی سریعی از ابزارها و فریمورک‌های تنظیم دقیق مدل‌های زبانی بزرگ از تأمین‌کنندگان مختلف ارائه می‌دهند.

Hugging Face TrainerAPI	AWS SageMaker JumpStart	Amazon Bedrock	Hugging Face AutoTrain	NVIDIA NeMo	پارامتر
تنظیم دقیق دستی LLM‌ها با کنترل جزئی بر فرآیند آموزش	ساده‌سازی تنظیم دقیق و پیاده‌سازی در اکوسیستم AWS	تنظیم دقیق و پیاده‌سازی LLM‌ها در AWS زیرساخت	تنظیم دقیق و پیاده‌سازی LLM‌ها با کمترین نیاز به کدنویسی	تنظیم دقیق مدل‌های LLM با استفاده از GPU‌های پیشرفته NVIDIA	مورد استفاده اصلی
پشتیبانی از طیف گسترده‌ای از مدل‌ها از مخزن Hugging Face	مدل‌های پیش‌آموزش یافته از AWS و شرکاء؛ امکان ادغام با مدل‌های سفارشی	پشتیبانی از مدل‌های Amazon مانند Titan و مدل‌های شخص ثالث	پشتیبانی از طیف گسترده‌ای از مدل‌های پیش‌آموزش یافته در مخزن	پشتیبانی از انواع مدل‌های بزرگ از جمله سری Megatron	پشتیبانی از مدل‌ها
کاربران به صورت دستی داده‌ها را پیش‌پردازش و مرحل آموزش را مدیریت می‌کنند	بارگذاری و پردازش داده در AWS محیط؛ پشتیبانی از انواع فرمتهای داده	داده‌ها در محیط AWS؛ مدیریت می‌شوند؛ با خدمات داده AWS ادغام می‌شود	بارگذاری داده‌ها از طریق رابط ساده؛ AutoTrain پیش‌پردازش و آموزش مدل را انجام می‌دهد	کاربران داده‌های خود را برای تنظیم دقیق فراهم می‌کنند که توسط زیرساخت NVIDIA پردازش می‌شود	مدیریت داده

<sup>160</sup> <https://platform.openai.com/docs/guides/fine-tuning/fine-tuning-integrations>

بسیار بالا؛ کنترل دقیق بر هر جنبه از تنظیم دقیق	متوسط؛ تنظیمات پیش‌پیکربندی شده با امکان سفارشی‌سازی محدود	بالا؛ پیکربندی دقیق و ادغام با خدمات AWS	متوسط؛ فرآیند خودکار با گزینه‌های سفارشی‌سازی محدود	بالا؛ کنترل گسترده بر فرآیند تنظیم دقیق و پارامترهای مدل	سطح سفارشی سازی
بالا؛ مقیاس‌پذیری واپسیت به زیرساخت مورد استفاده (مثلًا محلی یا ابری)	بسیار بالا؛ اکوسیستم ابری AWS	بسیار بالا؛ مقیاس‌پذیری در سطح وسیع زیرساخت ابری AWS	بالا؛ مقیاس‌پذیر از طریق زیرساخت ابری Hugging Face	بالا؛ استفاده از قابلیت‌های GPU برای NVIDIA مقیاس‌پذیری کارآمد	مقیاس‌پذیری
قابلیت پیاده‌سازی محلی، ابری یا خروجی‌گیری برای پلتفرم‌های دیگر	پیاده‌سازی ابری AWS؛ امکان ادغام با سایر خدمات AWS	ادغام در خدمات AWS و امکان پیاده‌سازی در سطح جهانی AWS	پیاده‌سازی از طریق Hugging Face یا قابلیت Face برای NVIDIA پیاده‌سازی محلی	پیاده‌سازی در محل یا از طریق زیرساخت ابری NVIDIA	گزینه‌های پیاده‌سازی
ادغام با اکوسیستم Hugging Face و سایر ابزارهای مبتنی بر Python	ادغام قوی با خدمات AWS؛ ارتباط آسان با خطوط داده و تحلیل‌ها	ادغام یکپارچه با خدمات AWS (مانند S3، Lambda، SageMaker)	ادغام خوب با اکوسیستم Hugging Face و سایر ابزارهای ML (مانند TensorRT) و جریان‌های کاری GPU مبتنی بر	ادغام عمیق با ابزارهای NVIDIA (مانند) و جریان‌های کاری GPU مبتنی بر	ادغام با اکوسیستم
به‌عهده کاربر است؛ بستگی به محل میزبانی داده‌ها و مدل‌ها دارد	تدابیر امنیتی و حریم خصوصی AWS؛ همخوانی با استانداردهای صنعتی	تمرکز بالا بر حریم خصوصی داده در محیط AWS؛ همخوانی با استانداردهای مختلف	داده‌ها در محیط Hugging Face پردازش می‌شوند؛ حریم خصوصی وابسته به نحوه مدیریت داده‌ها است	کاربران باید از رعایت حریم خصوصی داده اطمینان حاصل کنند؛ NVIDIA پردازش داده‌ها را بر عهده دارد	حریم خصوصی داده
پژوهشگران، توسعه‌دهندگان و مهندسین ML که به کنترل دقیق بر فرآیند آموزش نیاز دارند	شرکت‌ها و توسعه‌دهندگانی که به دنبال راهکارهای ساده AI/ML در حوزه AWS هستند	کسب‌وکارها و توسعه‌دهندگانی که در AWS یکپارچه شده‌اند یا خدمات ابری AWS را دارند	توسعه‌دهندگان و کسب‌وکارهایی که به دنبال راهکارهای ساده و خودکار قصد بهره‌گیری از تنظیم دقیق LLM هستند	شرکت‌ها و توسعه‌دهندگانی که به تنظیم دقیق و سفارشی‌سازی پیشرفته و عملکرد بالا در تنظیم دقیق LLM دارند	کاربران هدف

نیاز به تخصص فنی؛ راهاندازی و مدیریت پیچیده‌تر	محدود به خدمات AWS؛ گزینه‌های پیش‌پیکربندی شده ممکن است سفارشی‌سازی عمیق را محدود کند	وابستگی به AWS؛ احتمال قفل شدگی به فروشنده، پیچیدگی مدیریت هزینه‌ها	کنترل کمتر بر جزئیات تنظیم دقیق؛ مبتنی بر ابر، ممکن است برای نیازهای در محل مناسب نباشد	نیاز به منابع زیاد و هزینه‌های احتمالی؛ وابستگی به اکوسیستم NVIDIA	محدودیت‌ها
--	---	---	---	--	------------

جدول ۱۰.۱: مقایسه‌ی جامع پلتفرم‌های تنظیم مدل‌های زبانی بزرگ (قسمت اول). این جدول مقایسه‌ای جامع از ابزارهای مختلف تنظیم دقیق برای مدل‌های زبانی بزرگ (LLMs) ارائه می‌دهد که شامل HuggingFace AutoTrain، NVIDIA NeMo API، HuggingFace Trainer API و AWS SageMaker JumpStart، Amazon Bedrock API متعددی از جمله کاربرد اصلی، پشتیبانی از مدل‌ها، مدیریت داده‌ها، سطح سفارشی‌سازی، مقیاس‌پذیری، گزینه‌های استقرار، یکپارچگی با اکوسیستم، حفظ حریم خصوصی داده‌ها، کاربران هدف و محدودیت‌های هر ابزار را پوشش می‌دهد.

LangChain	Microsoft Azure AI Studio	Google Vertex AI Studio	OpenAI Fine-Tuning API	پارامتر
ساخت برنامه‌ها با استفاده از LLM ها با گردش‌کارهای ماژولار و قابل سفارشی‌سازی.	توسعه، آموزش و استقرار مدل‌های هوش مصنوعی در Azure.	توسعه و استقرار مدل‌های یادگیری ماشین از ابتدا تا انتها در Google Cloud	آموزش مدل‌های OpenAI با استفاده از داده‌های سفارشی و از طریق API.	مورد استفاده اصلی
پشتیبانی از ادغام با انواع LLM ها و ابزارهای هوش مصنوعی مانند GPT-4، OpenAI Cohere.	پشتیبانی از مدل‌های مایکروسافت و مدل‌های سفارشی در Azure.	پشتیبانی از مدل‌های از پیش آموزش‌دیده گوگل و مدل‌های سفارشی‌سازی شده.	محدود به مدل‌های OpenAI مانند GPT-4 و GPT-3	پشتیبانی از مدل
مدیریت داده انعطاف‌پذیر بوده و به LLM و ادغام خاص استفاده شده بستگی دارد.	داده‌ها در محیط Azure مدیریت شده و از فرمتهای فرمتهای مختلف پشتیبانی می‌کند.	داده‌ها در Google Cloud می‌شوند و از فرمتهای مختلف پشتیبانی می‌کنند.	کاربران داده‌ها را از طریق API بارگذاری می‌کنند؛ OpenAI مسئول پردازش و آموزش است.	مدیریت داده
بسیار بالا؛ امکان سفارشی‌سازی جزئیات گردش‌کارها، مدل‌ها و پردازش داده.	بالا؛ گزینه‌های گسترده برای سفارشی‌سازی از طریق ابزارهای هوش مصنوعی Azure.	بالا؛ ارائه آموزش و استقرار مدل سفارشی با تنظیمات دقیق.	متوسط؛ تمرکز بر سهولت استفاده با سفارشی‌سازی محدود.	سطح سفارشی‌سازی

بالا؛ مقیاس پذیری به زیرساخت و مدل‌های مورد استفاده بستگی دارد.	بسیار بالا؛ مقیاس پذیری در زیرساخت جهانی Azure.	بسیار بالا؛ بهره‌برداری از Google Zیرساخت Cloud برای مقیاس پذیری.	بسیار بالا؛ طریق زیرساخت ابری OpenAI.	مقیاس پذیری
استقرار در زیرساخت سفارشی؛ ادغام با سرویس‌های ابری و محلی مختلف.	استقرار در Azure؛ ادغام با مجموعه خدمات Azure.	استقرار در Google Cloud، یکپارچگی با سایر خدمات GCP.	استقرار از طریق API و ادغام در برنامه‌ها با استفاده از زیرساخت ابری OpenAI.	گزینه‌های استقرار
ادغام انعطاف‌پذیر با ابزارها، API‌ها و منابع داده مختلف.	یکپارچگی عمیق با خدمات Azure مانند Data Factory و Power BI.	ادغام بینه‌نیص با خدمات Google Cloud BigQuery و AutoML.	محدود به اکوسیستم OpenAI؛ ادغام خوب با اپ‌ها از طریق API.	یکپارچگی با اکوسیستم
وابسته به ادغام‌ها و زیرساخت‌های استفاده شده؛ مدیریت حریم خصوصی بر عهده کاربر.	امنیت و حریم خصوصی قوی در محیط Azure.	امنیت و حریم خصوصی قوی در محیط Google Cloud.	مدیریت شده توسط OpenAI؛ کاربران باید به انتقال داده و حریم خصوصی توجه داشته باشند.	حریم خصوصی داده
توسعه‌دهندگانی که نیاز به ساخت برنامه‌های LLM پیچیده مبتنی بر با گردش کارهای سفارشی دارند.	توسعه‌دهندگانی که نیاز به ساخت برنامه‌های LLM پیچیده مبتنی بر با گردش کارهای سفارشی دارند.	توسعه‌دهندگان و کسب‌وکارهای یکپارچه در Azure یا به دنبال استفاده از ابزارهای هوش مصنوعی Azure.	توسعه‌دهندگان و دنبال آموزش LLM با استفاده آسان از طریق API هستند.	کاربران هدف
پیچیدگی در زنجیره کردن مدل‌ها و منابع داده متعدد؛ نیاز به تنظیمات بیشتر.	محدود به اکوسیستم Azure؛ امکان هزینه و قفل شدن در پلتفرم.	محدود به اکوسیستم Google Cloud؛ امکان هزینه و قفل شدن در پلتفرم.	محدودیت در سفارشی‌سازی؛ وابستگی به زیرساخت AI؛ هزینه بالقوه.	محدودیت‌ها

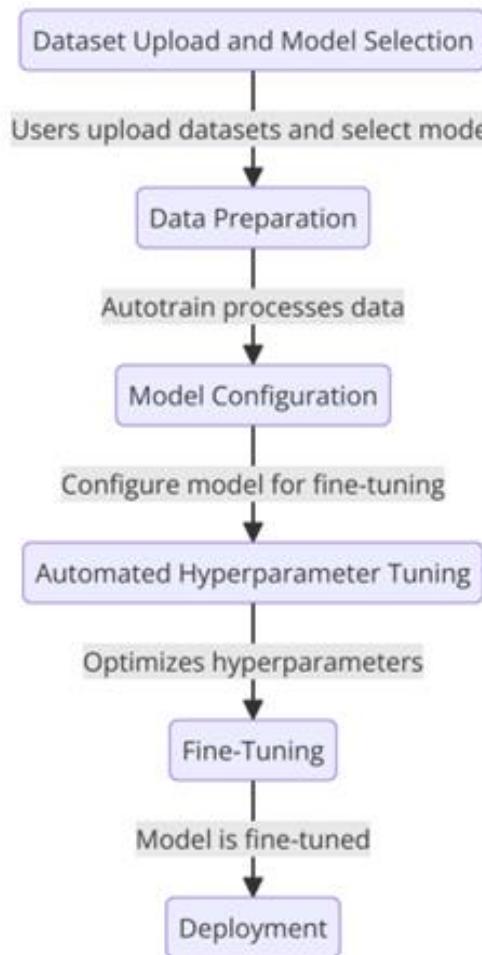
جدول ۱۰.۲: مقایسه‌ی جامع پلتفرم‌های تنظیم مدل‌های زبانی بزرگ (قسمت دوم). این جدول به مقایسه ابزارهای تنظیم دقیق مدل‌های زبانی بزرگ ادامه می‌دهد و بر Microsoft .Google Vertex AI Studio ،OpenAI Fine-Tuning API و LangChain و Azure AI Studio تمرکز دارد. این ابزارها براساس کاربرد اصلی، پشتیبانی از مدل، مدیریت داده، سطح سفارشی‌سازی، مقیاس پذیری، گزینه‌های استقرار، یکپارچگی با اکوسیستم، حریم خصوصی داده‌ها، کاربران هدف و محدودیت‌ها ارزیابی می‌شوند و دیدگاهی کامل از قابلیت‌ها و محدودیت‌های آنها ارائه می‌دهند.

## ۵- یک: Autotrain

پلتفرم نوآورانه HuggingFace است که فرآیند تنظیم دقیق مدل‌های زبانی بزرگ را به صورت خودکار انجام می‌دهد و این فرآیند را حتی برای افرادی با دانش محدود در یادگیری ماشین قابل دسترس می‌کند. پیچیدگی و نیازهای منابع در تنظیم دقیق مدل‌های زبانی بزرگ می‌تواند چالش‌برانگیز باشد، اما Autotrain با مدیریت جنبه‌های سخت این فرآیند، مانند آماده‌سازی داده‌ها، پیکربندی مدل و بهینه‌سازی ابرپارامترها، آن را ساده می‌کند. این خودکارسازی به ویژه برای تیم‌های کوچک یا توسعه‌دهندگان فردی که نیاز به استقرار سریع و کارآمد مدل‌های زبانی بزرگ سفارشی دارند، ارزشمند است.

### ۵- یک: مراحل تنظیم دقیق با استفاده از Autotrain

مراحل زیر مربوط به تنظیم دقیق مدل‌های زبانی بزرگ با استفاده از Autotrain است. شکل ۱۰.۱ جریان کاری تصویری را نشان می‌دهد.  
بارگذاری دیتابست و انتخاب مدل:



شکل ۱۰.۱: نمای کلی جریان کاری Autotrain. این نمودار فرآیند گام به گام سیستم Autotrain را نشان می‌دهد که با بارگذاری دیتاستها و انتخاب مدل توسط کاربران آغاز می‌شود. سپس جریان کاری به آماده‌سازی داده‌ها و پیکربندی مدل منتقل می‌شود و به دنبال آن تنظیم خودکار ابرپارامترها برای بهینه‌سازی عملکرد مدل انجام می‌گیرد. مرحله تنظیم دقیق، مدل را بر اساس دیتاست‌های ارائه شده تنظیم می‌کند و در نهایت، مدل کاملاً دقیق تنظیم شده برای استفاده عملی آماده است.

- کاربران با بارگذاری دیتاست‌ها روی پلتفرم Autotrain شروع می‌کنند.
- سپس یک مدل پیش‌آموزش دیده را از Model Hub گسترده HuggingFace انتخاب می‌کنند.

#### آماده‌سازی داده‌ها:

- Autotrain به صورت خودکار داده‌های بارگذاری شده را پردازش می‌کند که شامل وظایفی مانند توکنیزاسیون است تا متن را به فرمت قابل درک برای مدل زبانی بزرگ تبدیل کند.

#### پیکربندی مدل:

- این پلتفرم مدل را برای تنظیم دقیق پیکربندی می‌کند و محیط آموزش و پارامترهای لازم را تنظیم می‌نماید.

#### تنظیم خودکار ابرپارامترها:

- Autotrain تنظیمات مختلف ابرپارامترها (مانند نرخ یادگیری، اندازه دسته و طول توالی) را بررسی می‌کند و بهترین تنظیمات عملکردی را انتخاب می‌کند.

#### تنظیم دقیق:

- مدل با داده‌های آماده شده و ابرپارامترهای بهینه، دقیق تنظیم می‌شود.

#### استقرار:

- پس از اتمام تنظیم دقیق، مدل آماده است تا در کاربردهای مختلف پردازش زبان طبیعی (NLP) مانند تولید متن، تکمیل و ترجمه زبان به کار رود.

### ۵- یک - دو: بهترین شیوه‌های استفاده از Autotrain

- کیفیت داده‌ها: از داده‌های با کیفیت و دارای برچسب‌های دقیق استفاده کنید تا عملکرد مدل بهتر شود.
- انتخاب مدل: مدل‌های پیش‌آموزش دیده‌ای را انتخاب کنید که به خوبی با وظیفه‌ی خاص شما سازگار باشند تا نیاز به تنظیم دقیق کمتری داشته باشید.
- بهینه‌سازی ابرپارامترها: از تنظیم خودکار ابرپارامترهای Autotrain استفاده کنید تا بدون نیاز به مداخله دستی، عملکرد بهینه به دست آورید.

## ۵- یک - سه: چالش‌های استفاده از Autotrain

- حریم خصوصی داده‌ها: اطمینان از حفظ حریم خصوصی و امنیت داده‌های حساس در طول فرآیند تنظیم دقیق.
- محدودیت منابع: مدیریت مؤثر منابع محاسباتی، بهویژه در محیط‌هایی که دسترسی به سخت‌افزارهای قدرتمند محدود است.
- بیش آموزی<sup>۱۶۱</sup> مدل: جلوگیری از بیش آموزی مدل با استفاده از داده‌های متنوع و نماینده و همچنین بهره‌گیری از تکنیک‌های مناسب regularization.

## ۵- چهار - یک: موارد استفاده از Autotrain

۱. کمبود تخصص فنی عمیق: مناسب برای افراد یا تیم‌های کوچک بدون دانش گسترده در یادگیری ماشین یا مدل‌های زبانی بزرگ که نیاز به تنظیم دقیق سریع و کارآمد دارند.
  ۲. پروتوتایپ‌سازی سریع و استقرار: برای چرخه‌های توسعه سریع که زمان حیاتی است، مانند پروژه‌های اثبات مفهوم یا MVP‌ها مناسب است.
  ۳. محیط‌های با محدودیت منابع: در سناریوهایی با منابع محاسباتی محدود یا زمانی که نیاز به گردش سریع کار است، کاربردی است.
- به‌طور خلاصه، Autotrain ابزاری عالی برای تنظیم دقیق سریع و کاربرپسند مدل‌های زبانی بزرگ برای وظایف استاندارد NLP است، بهویژه در محیط‌هایی با منابع یا تخصص محدود. با این حال، ممکن است برای کاربردهای بسیار تخصصی یا آن‌هایی که نیاز به سفارشی‌سازی و مقیاس‌پذیری بالا دارند، مناسب نباشد.

## ۵- یک - پنج: آموزش‌ها

۱. نحوه ساخت مدل‌های سفارشی HuggingFace با استفاده از Autotrain<sup>۱۶۲</sup>.
۲. دقیق تنظیم کردن مدل‌ها با Autotrain<sup>۱۶۳</sup> HuggingFace

## ۵- دو: کتابخانه API و Transformers مربوط به Trainer

کتابخانه‌ی Transformers از HuggingFace به عنوان یک ابزار کلیدی برای تنظیم دقیق مدل‌های زبانی بزرگ (LLMs) مانند BERT، GPT-3 و GPT-4 شناخته می‌شود. این کتابخانه‌ی جامع مجموعه‌ای گسترده از مدل‌های پیش‌آموزش‌دیده برای وظایف مختلف LLM را فراهم می‌کند و به کاربران اجازه می‌دهد تا این مدل‌ها

<sup>161</sup> Overfitting

<sup>162</sup> <https://cobusgreiling.medium.com/how-to-create-huggingface-custom-ai-models-using-autotrain-72d75484b82b>

<sup>163</sup> <https://www.kdnuggets.com/how-to-finetune-mistral-ai-7b-llm-with-hugging-face-autotrain>

را بهآسانی و با تلاش کم برای نیازهای خاص خود تطبیق دهنند. چه در حال تنظیم دقیق برای وظایفی مانند تحلیل احساسات، دسته‌بندی متنی، یا ایجاد پاسخ‌های پشتیبانی مشتری باشد، این کتابخانه فرایند را با امکان انتخاب آسان مدل از Model Hub و سفارشی‌سازی ساده از طریق API‌های سطح بالا ساده می‌کند.

API اصلی در فرآیند تنظیم دقیق کتابخانه‌ی Trainer، Transformers مربوط به API است. این API شامل کلاس Trainer است که پیچیدگی‌های تنظیم دقیق مدل‌های زبانی بزرگ را مدیریت و خودکارسازی می‌کند. پس از اتمام پیش‌پردازش داده‌ها، کلاس Trainer فرایند آماده‌سازی آموزش مدل را تسهیل می‌کند، شامل مدیریت داده‌ها، بهینه‌سازی و ارزیابی. کاربران تنها نیاز دارند چند پارامتر، مانند نرخ یادگیری و اندازه دسته، را تنظیم کنند و API بقیه‌ی کارها را انجام می‌دهد. با این حال، توجه به این نکته ضروری است که اجرای Trainer.train() ممکن است روی CPU زمان‌بر و منابع‌بر باشد. برای آموزش کارآمدتر، استفاده از GPU یا TPU توصیه می‌شود. پلتفرم‌هایی مانند Google Colab دسترسی رایگان به این منابع را فراهم می‌کند و امکان تنظیم دقیق مدل‌ها را برای کاربرانی که سخت‌افزارهای قدرتمند ندارند، ممکن می‌سازد.

API مربوط به Trainer همچنین از ویژگی‌های پیشرفته‌ای مانند آموزش توزیع شده و آموزش با دقت مختلط پشتیبانی می‌کند که برای پردازش‌های مقیاس بزرگ مورد نیاز مدل‌های زبانی مدرن بسیار ضروری است. آموزش توزیع شده اجازه می‌دهد فرآیند تنظیم دقیق در چندین GPU یا نود مقیاس شود و به‌طور قابل توجهی زمان آموزش را کاهش می‌دهد. آموزش با دقت مختلط، از طرف دیگر، با استفاده از محاسبات با دقت پایین‌تر، مصرف حافظه و سرعت محاسبات را بهینه می‌کند، بدون این که عملکرد مدل کاهش یابد. تعهد HuggingFace به دسترسی آسان به فناوری، در مستندات گسترده و پشتیبانی جامعه کاربری مشهود است که امکان تنظیم دقیق مدل‌های زبانی بزرگ را برای کاربران با سطوح مختلف تخصص فراهم می‌کند. این دموکراتیزه‌سازی فناوری پیشرفته NLP، توسعه‌دهندگان و محققان را قادر می‌سازد تا مدل‌های پیچیده و دقیق تنظیم شده‌ای را برای طیف گسترده‌ای از کاربردها، از درک زبان تخصصی گرفته تا پردازش داده‌های مقیاس بزرگ، به کار بگیرند.

## ۵-۲-یک: محدودیت‌های کتابخانه Trainer API و Transformers

- محدودیت سفارشی‌سازی برای کاربران پیشرفته: در حالی که API مربوط به Trainer بسیاری از جنبه‌های آموزش را ساده می‌کند، ممکن است سفارشی‌سازی عمیقی که کاربران پیشرفته یا محققان برای کاربردهای خاص و پیچیده به آن نیاز دارند را فراهم نکند.

- منحنی یادگیری: با وجود API ساده، یادگیری و استفاده مؤثر از کتابخانه Trainer و Transformers API هنوز نیاز به یادگیری و درک دارد، به ویژه برای کسانی که در NLP و مدل‌های زبانی بزرگ تازه کار هستند.

- محدودیت‌های یکپارچه‌سازی: یکپارچگی و سهولت استفاده اغلب با اکوسیستم HuggingFace مرتبط است و ممکن است با همه‌ی جریان‌های کاری یا پلتفرم‌های خارج از این محیط سازگار نباشد.

به طور خلاصه، کتابخانه Trainer و API مربوط به Transformers راه حل های قدرتمند و مقیاس پذیری برای تنظیم دقیق مدل های زبانی بزرگ در طیف گسترده ای از کاربردها ارائه می دهند، با قابلیت استفاده آسان و آموزش کارآمد. با این حال، کاربران باید از نیازهای منابع و محدودیت های احتمالی در سفارشی سازی و مدیریت پیچیدگی آگاه باشند.

## ۵-۴: افزایش بهره وری در استقرار مدل های زبانی بزرگ

ابزاری از HuggingFace Optimum است که برای بهینه سازی استقرار مدل های زبانی بزرگ (LLMs) طراحی شده و هدف آن افزایش کارایی این مدل ها در پلتفرم های سخت افزاری مختلف است. با رشد اندازه و پیچیدگی مدل های زبانی بزرگ، استقرار آن ها به صورت مقرن به صرفه و کارآمد چالش برانگیزتر می شود. Optimum به این چالش ها پاسخ می دهد و با استفاده از بهینه سازی های ویژه سخت افزاری مانند کوانتاپیشن، پرونینگ و دیستیلیشن مدل، اندازه مدل را کاهش داده و سرعت استنتاج را بهبود می بخشد، بدون این که دقت مدل به طور قابل توجهی کاهش یابد. تکنیک های اصلی که توسط Optimum پشتیبانی می شوند عبارتند از:

- کوانتاپیشن: یکی از تکنیک های کلیدی در Optimum کوانتاپیشن است. این فرآیند شامل تبدیل وزن های مدل از اعداد با دقت بالا (مانند اعداد اعشاری شناور) به فرمت های دقت پایین تر (مانند int8 یا float16) است. کاهش دقت، استفاده از حافظه و نیازهای محاسباتی مدل را کاهش می دهد و موجب اجرای سریع تر و کاهش مصرف انرژی می شود، به ویژه در دستگاه های لبه ای (Edge) و پلتفرم های موبایل. Optimum فرآیند کوانتاپیشن را خود کار کرده و آن را در دسترس کاربرانی قرار می دهد که شاید در بهینه سازی سخت افزار در سطوح پایین تجربه ای نداشته باشند.

- پرونینگ: پرونینگ یکی دیگر از استراتژی های مهم بهینه سازی ارائه شده توسط Optimum است. این فرآیند شامل شناسایی و حذف وزن های کم اهمیت مدل می شود که پیچیدگی و اندازه کلی مدل را کاهش می دهد. این کاهش در پیچیدگی به زمان های استنتاج سریع تر و نیازهای ذخیره سازی کمتر منجر می شود که به ویژه برای استقرار مدل ها در محیط هایی با منابع محاسباتی محدود بسیار مفید است. الگوریتم های پرونینگ Optimum به دقت این وزن های زائد را حذف می کنند و در عین حال عملکرد مدل را حفظ می کنند، به طوری که حتی پس از بهینه سازی نیز همچنان نتایج با کیفیت بالایی ارائه می دهد.

- دیستیلیشن مدل: Optimum همچنین از دیستیلیشن مدل پشتیبانی می کند. دیستیلیشن فرآیندی است که در آن یک مدل کوچکتر و کارآمدتر آموزش می بیند تا رفتار مدل بزرگتر و پیچیده تر را تقلید کند. این مدل دیستیل شده، بخش زیادی از دانش و قابلیت های مدل اصلی را در حالی حفظ می کند که به طور قابل توجهی سبک تر و سریع تر است. Optimum ابزارهایی را برای تسهیل فرآیند دیستیلیشن فراهم می کند که به کاربران اجازه می دهد مدل های زبانی فشرده ای ایجاد کنند که برای برنامه های کاربردی زمان حقیقی (Real-time) مناسب باشند. با ارائه مجموعه ای جامع از ابزارهای بهینه سازی، Optimum اطمینان می دهد

که مدل‌های زبانی بزرگ HuggingFace می‌توانند به طور مؤثری در محیط‌های مختلف، از سرورهای ابری قدرتمند گرفته تا دستگاه‌های لبه‌ای با محدودیت منابع، مستقر شوند.

#### ۵-۴-یک: بهترین روش‌ها برای استفاده از Optimum

- شناخت نیازهای سخت‌افزاری: محیط استقرار مورد نظر (مثل دستگاه‌های لبه‌ای یا سرورهای ابری<sup>۱۶۴</sup>) را ارزیابی کنید تا تنظیمات مدل را به طور مناسب بهینه کنید.
- بهینه‌سازی تکرارشونده: با آزمایش تکنیک‌های مختلف بهینه‌سازی (سطح کوانتایزیشن، آستانه‌های پرونینگ)، تعادل بهینه‌ای بین اندازه مدل، سرعت و دقت پیدا کنید.
- اعتبارسنجی و آزمایش: مدل‌های بهینه‌سازی شده را به دقت اعتبارسنجی کنید تا اطمینان حاصل شود که در شرایط مختلف عملکرد و دقت مورد نظر را ارائه می‌دهند.
- مراجعه به مستندات و پشتیبانی: از منابع HuggingFace برای راهنمایی دقیق در استفاده موثر از ابزارهای Optimum استفاده کنید و از پشتیبانی جامعه برای رفع مشکلات و به اشتراک‌گذاری بهترین روش‌ها بهره ببرید.
- پایش مداوم: مدل‌های مستقر شده را پس از بهینه‌سازی برای شناسایی هرگونه کاهش عملکرد پایش کنید و استراتژی‌های بهینه‌سازی را در صورت نیاز تنظیم کنید تا عملکرد بهینه به مرور زمان حفظ شود.

#### ۵-۴-دو: آموزش

مقدمه‌ای بر استفاده از Hugging Face و Transformers.

#### ۵-چهار: Amazon SageMaker JumpStart

Amazon SageMaker JumpStart یکی از قابلیت‌های موجود در اکوسیستم SageMaker است که با هدف ساده‌سازی و تسريع فرآیند تنظیم دقیق مدل‌های زبانی بزرگ (LLMs) طراحی شده است. این ابزار یک کتابخانه گسترده از مدل‌های از پیش ساخته شده و راه حل‌های آماده را ارائه می‌دهد که به سرعت می‌توان آن‌ها را برای استفاده‌های مختلف شخصی‌سازی کرد. این ابزار به ویژه برای سازمان‌هایی که به دنبال استقرار راهکارهای پردازش زبان طبیعی (NLP) به صورت کارآمد و بدون نیاز به تخصص عمیق در یادگیری ماشین یا منابع محاسباتی گسترده برای آموزش مدل‌های بزرگ از صفر هستند، بسیار مفید است. شکل ۱۰.۲ معماری یک پایپ‌لاین جامع را برای تنظیم دقیق و استقرار مدل‌های زبانی بزرگ با استفاده از خدمات AWS نشان می‌دهد.

<sup>۱۶۴</sup> Edge Devices or Cloud Servers

<sup>۱۶۵</sup> <https://www.datacamp.com/tutorial/an-introduction-to-using-transformers-and-hugging-face>

## ۵- چهار-یک: مراحل استفاده از JumpStart

### - آماده‌سازی و پیش‌پردازش داده:

- ذخیره داده‌ها: ابتدا مجموعه داده‌های خام را به طور ایمن در Amazon S3، سرویس ذخیره‌سازی اشیاء مقیاس‌پذیر AWS، ذخیره کنید.

- پیش‌پردازش: از چارچوب Apache Spark همراه با EMR Serverless بهصورت کارآمد استفاده کنید. این مرحله داده‌های خام را برای مراحل بعدی آموزش و ارزیابی مدل پالایش و آماده می‌کند.

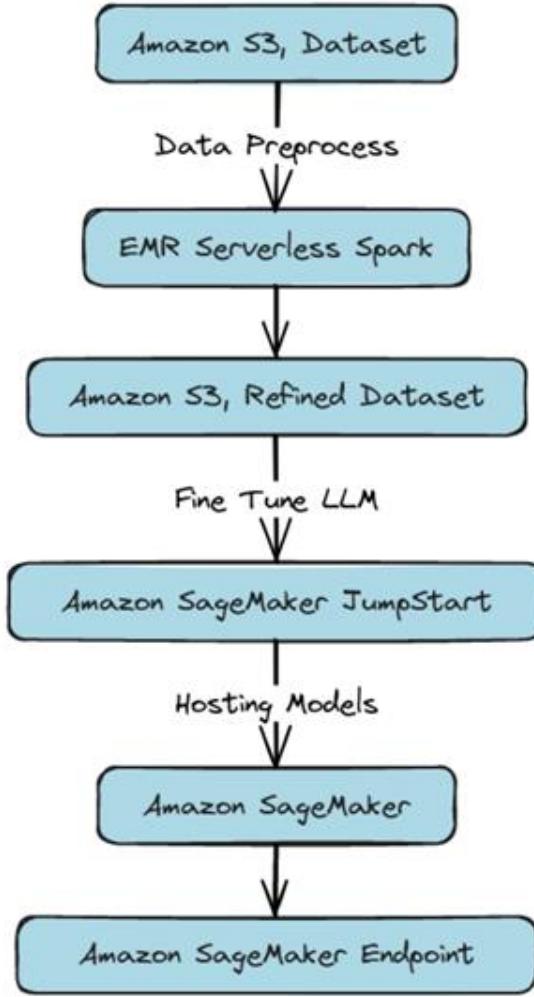
- تصفیه داده‌ها: پس از پیش‌پردازش، مجموعه داده‌های پردازش شده را مجدداً به Amazon S3 بازگردانید تا برای مراحل بعدی در دسترس و آماده باشند.

### - تنظیم دقیق مدل با SageMaker JumpStart

- انتخاب مدل: از میان مدل‌ها و راه حل‌های متنوع موجود در کتابخانه گسترده SageMaker JumpStart مدلی را مناسب با وظایف خاصی مانند تحلیل احساسات، تولید متن یا خودکارسازی پشتیبانی مشتری انتخاب کنید.

- اجرای تنظیم دقیق: از قابلیت‌های Amazon SageMaker که با SageMaker JumpStart یکپارچه شده است، برای تنظیم دقیق مدل انتخاب شده استفاده کنید. این مرحله شامل تنظیم پارامترها و پیکربندی‌ها برای بهینه‌سازی عملکرد مدل برای کاربردهای خاص می‌شود.

- ساده‌سازی جریان کاری: از الگوریتم‌ها و قالب‌های مدل از پیش ساخته شده‌ای که توسط SageMaker JumpStart ارائه می‌شود، برای ساده‌سازی جریان کاری تنظیم دقیق بهره ببرید و زمان و تلاش مورد نیاز برای استقرار را کاهش دهید.



شکل ۱۰.۲: جریان کاری گام به گام که فرآیند Amazon SageMaker JumpStart را نشان می‌دهد، از پیش‌پردازش داده‌ها با استفاده از EMR Serverless Spark شروع شده، به تنظیم دقیق مدل‌های زبانی بزرگ (LLM) می‌رسد و با استقرار مدل در نقاط پایانی (Endpoints) Amazon SageMaker (اقتباس از [۸۳])

#### - استقرار و میزبانی مدل

- راهاندازی استقرار: مدل دقیق تنظیم شده را با استفاده از قابلیت‌های استقرار نقطه پایانی Amazon SageMaker مستقر کنید. این تنظیمات تضمین می‌کند که مدل در محیطی مقیاس‌پذیر و آماده برای ارائه پیش‌بینی‌های بلاذرنگ به صورت کارآمد میزبانی می‌شود.

- مقیاس‌پذیری: از مقیاس‌پذیری زیرساخت AWS بهره‌مند شوید که به شما امکان می‌دهد منابع را به صورت پیوسته و هماهنگ با تغییرات بار کاری و نیازهای عملیاتی تنظیم کنید.

- کارآمدی و دسترسی پذیری: اطمینان حاصل کنید که مدل مستقر شده از طریق نقاط پایانی قابل دسترسی است، و این امکان را می‌دهد که به طور کارآمد در برنامه‌های عملیاتی برای وظایف پیش‌بینی بلاذرنگ ادغام شود.

#### ۵- چهار - دو: بهترین روش‌های استفاده از JumpStart

- مدیریت قوی داده‌ها: روش‌های ذخیره‌سازی داده‌های امن و سازمان‌یافته را در Amazon S3 پیاده‌سازی کنید تا به دسترسی و مدیریت کارآمد داده‌ها در طول پایپ‌لاین کمک کند.
- پردازش مقرن‌به‌صرفه: از چارچوب‌های محاسبات بدون سرور مانند Apache EMR Serverless با Spark استفاده کنید تا پیش‌پردازش داده‌ها به صورت مقیاس‌پذیر و مقرن‌به‌صرفه انجام شود.
- تنظیم دقیق بهینه‌شده: از مدل‌ها و الگوریتم‌های از پیش ساخته‌شده در SageMaker JumpStart بهره ببرید تا فرآیند تنظیم دقیق را سرعت بخشیده و بهینه کنید و عملکرد مدل را بدون نیاز به تنظیمات دستی گسترده تضمین کنید.
- پایش و بهینه‌سازی مستمر: پس از استقرار، مکانیزم‌های پایش قوی برای پیگیری معیارهای عملکرد مدل پیاده‌سازی کنید. این کار امکان بهینه‌سازی و تنظیمات به موقع را فراهم می‌کند تا دقت و کارایی مدل در طول زمان حفظ شود.
- یکپارچگی با خدمات AWS: از مجموعه جامع خدمات AWS و قابلیت‌های یکپارچه‌سازی آن استفاده کنید تا پایپ‌لاین‌های انتهای ایجاد کنید و استقرار مدل‌های زبانی بزرگ را در محیط‌های عملیاتی متنوع به صورت قابل اعتماد و مقیاس‌پذیر تضمین کنید.

#### ۶- چهار ت سه: محدودیت‌های استفاده از JumpStart

- سفارشی‌سازی محدود: با وجود آنکه JumpStart فرآیند را برای موارد استفاده رایج ساده می‌کند، ممکن است انعطاف‌پذیری کمتری برای کاربردهای بسیار تخصصی یا پیچیده که به شخصی‌سازی‌های قابل توجهی فراتر از الگوها و جریان‌های کاری ارائه شده نیاز دارند، داشته باشد.
- وابستگی به اکوسیستم AWS: JumpStart با خدمات AWS به طور تنگاتنگی یکپارچه شده است که ممکن است برای کاربرانی که ترجیح می‌دهند یا نیاز دارند در محیط‌های چندابری کار کنند یا زیرساخت‌های موجود خارج از AWS دارند، چالش‌برانگیز باشد.
- هزینه‌های منابع: استفاده از منابع مقیاس‌پذیر SageMaker برای تنظیم دقیق مدل‌های زبانی بزرگ، به ویژه مدل‌های حجمی، ممکن است هزینه‌های قابل توجهی به همراه داشته باشد که می‌تواند برای سازمان‌های کوچک‌تر یا با بودجه محدود مانع ایجاد کند.

## ۵- چهار- چهار: آموزش‌ها

۱. تنظیم دقیق LLaMA 2 با Amazon SageMaker JumpStart<sup>166</sup>
۲. ایجاد نمایندگان LLM با استفاده از مدل‌های پایه AWS SageMaker JumpStart<sup>167</sup>

## ۵- پنج: Amazon Bedrock

یک سرویس کاملاً مدیریت شده است که با هدف ساده‌سازی دسترسی به مدل‌های Amazon Bedrock<sup>168</sup> پایه‌ی (FMs)<sup>169</sup> با کارایی بالا از برترین نوآوران هوش مصنوعی، مانند Cohere، Anthropic، AI21 Labs، Stability AI، Mistral AI، Meta و Amazon Bedrock طراحی شده است. این سرویس یک API یکپارچه فراهم می‌کند که این مدل‌ها را ادغام کرده و قابلیت‌های گسترده‌ای برای توسعه برنامه‌های مولد هوش مصنوعی امن، خصوصی و مسئولانه ارائه می‌دهد. با استفاده از Amazon Bedrock، کاربران می‌توانند به راحتی مدل‌های پایه پیشرو را بر اساس نیازهای خاص خود آزمایش و ارزیابی کنند. این سرویس از شخصی‌سازی خصوصی مدل‌ها از طریق تنظیم دقیق و تولید بازیابی افزوده (RAG) پشتیبانی می‌کند و امکان ایجاد عوامل هوشمند را فراهم می‌سازد که از داده‌ها و سیستم‌های سازمانی بهره می‌برند. معماری بدون سرور Amazon Bedrock به استقرار سریع، ادغام آسان و سفارشی‌سازی امن مدل‌های پایه بدون نیاز به مدیریت زیرساخت‌ها کمک می‌کند و با استفاده از ابزارهای AWS، این مدل‌ها را به صورت کارآمد و امن در برنامه‌ها مستقر می‌کند.

## ۵- پنج - یک: مراحل استفاده از Amazon Bedrock

یک جریان کاری ساده برای استقرار و تنظیم دقیق مدل‌های زبانی بزرگ (LLMs) Amazon Bedrock ارائه می‌دهد و آن را به گزینه‌ای ایده‌آل برای کسب‌وکارهایی تبدیل می‌کند که به دنبال ادغام سریع قابلیت‌های پیشرفته هوش مصنوعی در عملیات خود هستند. در اینجا مروری کلی بر نحوه عملکرد Bedrock آمده است:

- انتخاب مدل: کاربران از میان مجموعه‌ای انتخاب شده از مدل‌های پایه که از طریق Bedrock در دسترس هستند، انتخاب می‌کنند. این مدل‌ها شامل مدل‌های AWS Titan (مانند Amazon Titan) و ارائه‌دهندگان شخص ثالث (مانند Stability AI و Anthropic Claude) می‌شوند.

<sup>166</sup> <https://www.linkedin.com/pulse/fine-tuning-llama-2-amazon-sagemaker-jumpstart-elhousieny-phd%E1%B4%AC%E1%B4%AE%E1%B4%BO-8zp9c/>

<sup>167</sup> <https://aws.amazon.com/blogs/machine-learning/learn-how-to-build-and-deploy-tool-using-lm-agents-using-aws-sagemaker-jumpstart-foundation-models/>

<sup>168</sup> <https://aws.amazon.com/bedrock/>

<sup>169</sup> Foundation Models

#### - تنظیم دقیق:

- پس از انتخاب مدل، کاربران می‌توانند آن را برای برآورده کردن نیازهای خاص خود دقیق تنظیم کنند. این کار با ارائه داده‌های خاص به مدل یا دستورالعمل‌های خاص به منظور تطبیق خروجی‌های مدل با حوزه مورد نظر انجام می‌شود.

- فرآیند تنظیم دقیق از طریق فراخوانی API ساده انجام می‌شود و نیازی به تنظیمات پیچیده یا پیکربندی‌های دقیق ندارد. کاربران داده‌های سفارشی خود را ارائه می‌دهند و **Bedrock** فرآیند آموزش را در پشت‌صحنه مدیریت می‌کند.

#### - استقرار:

- پس از تنظیم دقیق، **Bedrock** به‌طور کارآمد و مقیاس‌پذیر مدل را مستقر می‌کند. این بدان معناست که کاربران می‌توانند به سرعت مدل دقیق تنظیم شده را در برنامه‌ها یا سرویس‌های خود ادغام کنند.

- اطمینان می‌دهد که مدل بر اساس تقاضا مقیاس می‌شود و بهینه‌سازی عملکرد را مدیریت می‌کند، که تجربه‌ای بدون مشکل را برای کاربران فراهم می‌آورد.

#### - یکپارچه‌سازی و پایش:

- **Bedrock** به‌طور یکپارچه با سایر خدمات AWS ادغام می‌شود، به کاربران امکان می‌دهد قابلیت‌های هوش مصنوعی را مستقیماً در اکوسیستم AWS موجود خود تعییه کنند.

- کاربران می‌توانند از طریق ابزارهای پایش جامع AWS، عملکرد مدل‌های مستقر شده را پایش و مدیریت کنند تا اطمینان حاصل شود که مدل‌ها به صورت بهینه به کار خود ادامه می‌دهند.

### ۵- پنج - دو: محدودیت‌های استفاده از Amazon Bedrock

با وجود آنکه Amazon Bedrock مجموعه‌ای قوی از ابزارها و خدمات برای حل برخی چالش‌های هوش مصنوعی ارائه می‌دهد، برای تمام نیازهای هوش مصنوعی یک راه حل جامع نیست. یکی از محدودیت‌های کلیدی این است که این سرویس نیاز به تخصص انسانی را به‌طور کامل حذف نمی‌کند. سازمان‌ها همچنان به کارشناسانی با دانش کامل از پیچیدگی‌های فناوری هوش مصنوعی نیاز دارند تا بتوانند مدل‌های ارائه شده توسط **Bedrock** را به‌طور موثر توسعه، دقیق تنظیم و بهینه‌سازی کنند.

علاوه بر این، Amazon Bedrock به‌گونه‌ای طراحی نشده است که به صورت یک سرویس مستقل عمل کند. این سرویس برای بهره‌گیری کامل، به یکپارچگی با سایر خدمات AWS مانند Amazon S3 برای ذخیره‌سازی داده‌ها، AWS Lambda برای محاسبات بدون سرور و Amazon SageMaker برای توسعه مدل‌های یادگیری ماشین متکی است. بنابراین، کسب‌وکارهایی که از Amazon Bedrock استفاده می‌کنند، نیاز به استفاده از این خدمات مکمل AWS نیز خواهند داشت تا از تمام قابلیت‌های آن بهره‌مند شوند. این وابستگی به معنای آن است که در حالی که Amazon Bedrock قابلیت‌های هوش مصنوعی را در اکوسیستم

AWS افزایش می‌دهد، ممکن است برای کسانی که با AWS آشنایی ندارند، نیاز به آموزش بیشتر و مدیریت زیرساخت‌ها داشته باشد.

## ۵- پنج - سه: آموزش‌ها

۱. تنظیم دقیق مدل‌های زبانی بزرگ (LLMs) در [Amazon Bedrock<sup>۱۷۰</sup>](#)
۲. استفاده از [Amazon Bedrock](#) برای هوش مصنوعی مولد<sup>۱۷۱</sup>.

## ۶- شش: API تنظیم دقیق OpenAI

API تنظیم دقیق OpenAI یک پلتفرم جامع است که به کاربران امکان می‌دهد مدل‌های زبانی بزرگ (LLM) از پیش آموزش‌دیده‌ی OpenAI را برای انجام وظایف و حوزه‌های خاص سفارشی کنند. این سرویس با هدف سهولت استفاده طراحی شده است تا مجموعه وسیعی از کاربران، از کسب‌وکارها گرفته تا توسعه‌دهندگان فردی، بتوانند بدون پیچیدگی‌های معمول مرتبط با آموزش و استقرار مدل‌ها از قدرت هوش مصنوعی پیشرفته بهره‌مند شوند.

### ۶- شش - یک: مراحل استفاده از API تنظیم دقیق OpenAI

- انتخاب مدل:

- انتخاب یک مدل از پیش آموزش‌دیده: کاربران با انتخاب یک مدل پایه از مجموعه مدل‌های OpenAI شروع می‌کنند. این مدل‌ها شامل مدل‌های قدرتمندی مانند GPT-4 هستند که نقطه شروعی قوی برای بسیاری از وظایف پردازش زبان به شمار می‌روند.
- پایه قابل سفارشی‌سازی: این مدل‌ها با حجم وسیعی از داده‌ها از پیش آموزش‌دیده شده‌اند و پایه‌ای محکم فراهم می‌کنند که می‌تواند برای برآورده کردن نیازهای خاص، بهبود یابد.
- آماده‌سازی و بارگذاری داده:

- گردآوری داده‌های مرتبط: کاربران باید یک مجموعه داده تهیه و آماده کنند که بازتاب‌دهنده وظیفه یا حوزه خاصی است که قصد تنظیم دقیق مدل برای آن را دارند. این داده‌ها برای آموزش مدل در جهت انجام بهتر وظیفه موردنظر ضروری است.

- بارگذاری داده در API: API تنظیم دقیق امکان بارگذاری آسان داده‌ها را فراهم می‌کند. کاربران می‌توانند مجموعه داده‌های آماده‌شده خود را با دستورات ساده‌ای در API بارگذاری کنند، که این فرآیند حتی برای افراد با دانش فنی محدود نیز قابل دسترس است.

<sup>۱۷۰</sup> <https://medium.com/@abdullahiolaoye4/finetuning-langs-on-amazon-bedrock-887ebc547adc>

<sup>۱۷۱</sup> <https://cloudnature.net/blog/the-complete-guide-to-amazon-bedrock-for-generative-ai>

## - شروع تنظیم دقیق:

- فرآیند خودکار: پس از بارگذاری داده‌ها، زیرساخت OpenAI فرآیند تنظیم دقیق را به عهده می‌گیرد. API پارامترهای مدل را بر اساس داده‌های جدید تنظیم می‌کند تا عملکرد آن در وظایف مشخص شده بهبود یابد.

## - استقرار مدل دقیق تنظیم شده:

- یکپارچگی API: مدل دقیق تنظیم شده می‌تواند از طریق API OpenAI مورد دسترسی و استقرار قرار گیرد. این امر امکان ادغام آسان مدل در برنامه‌های مختلف، از جمله چتبات‌ها، ابزارهای تولید محتوای خودکار یا سیستم‌های تخصصی خدمات مشتری را فراهم می‌کند.

## ۵- شش - دو: محدودیت‌های API تنظیم دقیق

- مدل‌های هزینه دار: تنظیم دقیق و استفاده از مدل‌های API از طریق API OpenAI می‌تواند هزینه‌بر باشد، به‌ویژه برای استقرارهای گسترده یا استفاده مداوم. این موضوع می‌تواند برای سازمان‌های کوچک‌تر یا پروژه‌های با بودجه محدود چالشی اساسی باشد.

- حریم خصوصی و امنیت داده‌ها: کاربران باید داده‌های خود را برای فرآیند تنظیم دقیق به سرورهای OpenAI بارگذاری کنند. این موضوع ممکن است نگرانی‌هایی درباره حریم خصوصی داده‌ها و امنیت اطلاعات حساس یا اختصاصی ایجاد کند.

- وابستگی به زیرساخت API: وابستگی به زیرساخت API OpenAI برای میزبانی مدل و دسترسی API می‌تواند منجر به وابستگی به یک فروشنده خاص شود و انعطاف‌پذیری و کنترل بر محیط استقرار را محدود کند.

- کنترل محدود بر فرآیند آموزش: فرآیند تنظیم دقیق به‌طور عمده خودکار و توسط API مدیریت می‌شود، که کنترل و شفافیت محدودی بر تنظیمات خاص اعمال شده بر مدل را به کاربران ارائه می‌دهد.

## ۵- شش - سه: آموزش‌ها

### ۱. تنظیم دقیق GPT-3 با استفاده از <sup>172</sup> API OpenAI

## ۵- هفت: سفارشی‌سازی NeMo NVIDIA

NVIDIA NeMo Customizer<sup>173</sup> بخشی از چارچوب NeMo است، مجموعه‌ای از ابزارها و مدل‌ها که توسط NVIDIA طراحی شده‌اند تا توسعه و تنظیم دقیق مدل‌های زبانی بزرگ (LLM) را تسهیل کنند. این

<sup>172</sup> <https://www.datacamp.com/tutorial/fine-tuning-gpt-3-using-the-open-ai-api-and-python>

<sup>173</sup> <https://developer.nvidia.com/blog/fine-tune-and-align-langs-easily-with-nvidia-nemo-customizer/>

سرویس به ویژه بر روی آسان تر کردن تنظیم دقیق مدل های زبانی بزرگ برای وظایف و حوزه های خاص تمرکز دارد. مانند سایر ابزارهای تنظیم دقیق، NeMo Customizer به کاربران کمک می کند تا مدل های از پیش آموزش دیده را برای برنامه های خاص، مانند هوش مصنوعی مکالمه ای، ترجمه، یا تولید متن های مرتبط با یک حوزه خاص سفارشی کنند.

این پلتفرم با ارائه دقت بالا در گردآوری داده ها، گزینه های سفارشی سازی گسترشده، تولید بازیابی افزوده (RAG) و ویژگی های بهبود عملکرد، مدل های آماده استفاده در سطح سازمانی ارائه می دهد. این پلتفرم از آموزش و استقرار مدل های مولد هوش مصنوعی در محیط های متعدد، از جمله فضای ابری، مراکز داده و مکان های مرزی پشتیبانی می کند. NeMo Customizer پکیج کاملی را همراه با پشتیبانی، امنیت و API های قابل اعتماد به عنوان بخشی از NVIDIA AI Enterprise فراهم می کند.

## ۵- هفت - یک: ویژگی های کلیدی NVIDIA NeMo

- **NVIDIA NeMo** برای بهبود پروژه های هوش مصنوعی با چندین ویژگی برجسته طراحی شده است. **تکنیک های آموزش پیشرفته:** NeMo از ابزارهای شتاب دهنده GPU مانند NeMo Curator برای تهییه مجموعه داده های با کیفیت بالا و در مقیاس بزرگ استفاده می کند. این ابزارها با بهره گیری از هزاران هسته محاسباتی، فرآیند آموزش پیشین مدل های مولد هوش مصنوعی را تسهیل می کنند و زمان آموزش را به طور قابل توجهی کاهش می دهند و دقت مدل های زبانی بزرگ (LLM) را افزایش می دهند.

- **سفارشی سازی پیشرفته برای LLM ها:** میکروسرویس NeMo Customiser امکان تنظیم دقیق و هم راستایی دقیق LLM ها را برای حوزه های خاص فراهم می کند. این سرویس با استفاده از موازی سازی مدل، سرعت آموزش را افزایش می دهد و از مقیاس پذیری در چندین GPU و گره پشتیبانی می کند، که این امر امکان تنظیم دقیق مدل های بزرگ تر را فراهم می کند.

- **بهینه سازی استنتاج AI با NVIDIA Triton:** NVIDIA Triton Server شامل NVIDIA Triton Inference Server است که استنتاج هوش مصنوعی را در مقیاس وسیع تسهیل می کند. این یک پارچگی، استنتاج مولد هوش مصنوعی را تسريع می کند و اطمینان می دهد که برنامه های هوش مصنوعی به طور مطمئن هم در محل و هم در ابر استقرار یابند.

- **ابزارهای کاربر پسند برای هوش مصنوعی مولد:** NeMo دارای معماری ماژولار و قابل استفاده مجدد است که توسعه مدل های هوش مصنوعی مکالمه ای را ساده می کند. این پلتفرم از فرآیندهای جامع از پردازش داده تا استقرار پشتیبانی می کند و شامل مدل های از پیش آموزش دیده برای شناسایی گفتار خودکار (ASR)، پردازش زبان طبیعی (NLP) و تبدیل متن به گفتار (TTS) است که می توان آن ها را دقیق تنظیم کرد یا به صورت آماده استفاده کرد.

- مدل‌های از پیش آموزش دیده با بهترین کیفیت: مجموعه‌های NeMo تنوعی از مدل‌های از پیش آموزش دیده و اسکریپت‌های آموزشی را ارائه می‌دهند که توسعه سریع برنامه‌ها یا تنظیم دقیق برای وظایف خاص را تسهیل می‌کند. در حال حاضر، NeMo از مدل‌هایی مانند 2 Llama، Stable Diffusion و خانواده Nemotron-3 8B NVIDIA پشتیبانی می‌کند.

- تولید با بازیابی بهینه‌شده: NeMoRetriever قابلیت‌های بازیابی اطلاعات با عملکرد بالا و تأخیر کم را فراهم می‌کند و برنامه‌های مولد هوش مصنوعی را با قابلیت‌های تولید بازیابی افزوده (RAG) در سطح سازمانی تقویت می‌کند. این ویژگی از بینش‌های تجاری در زمان واقعی و استفاده از داده‌ها پشتیبانی می‌کند.

## ۵- هفت - دو: اجزای NVIDIA NeMo

- **NeMo Core**: عناصر ضروری مانند کارخانه مژول‌های عصبی برای آموزش و استنتاج را فراهم می‌کند و توسعه مدل‌های هوش مصنوعی مکالمه‌ای را تسهیل می‌کند.

- **مجموعه‌های NeMo**: مژول‌ها و مدل‌های تخصصی برای NLP، ASR و TTS را شامل می‌شود، از جمله مدل‌های از پیش آموزش دیده و اسکریپت‌های آموزشی که این پلتفرم را متنوع می‌سازد.

- **مژول‌های عصبی**: به عنوان اجزای سازنده NeMo عمل می‌کنند و اجزای قابل آموزش مانند کدگذارها و کدگشاه را تعریف می‌کنند که می‌توانند به هم متصل شوند تا مدل‌های جامع‌تری ایجاد کنند.

- **اسکریپت‌های کاربردی**: استقرار مدل‌های هوش مصنوعی مکالمه‌ای را با اسکریپت‌های آماده برای استفاده ساده می‌کنند و امکان آموزش سریع یا تنظیم دقیق بر روی مجموعه داده‌های خاص برای برنامه‌های مختلف هوش مصنوعی را فراهم می‌آورند.

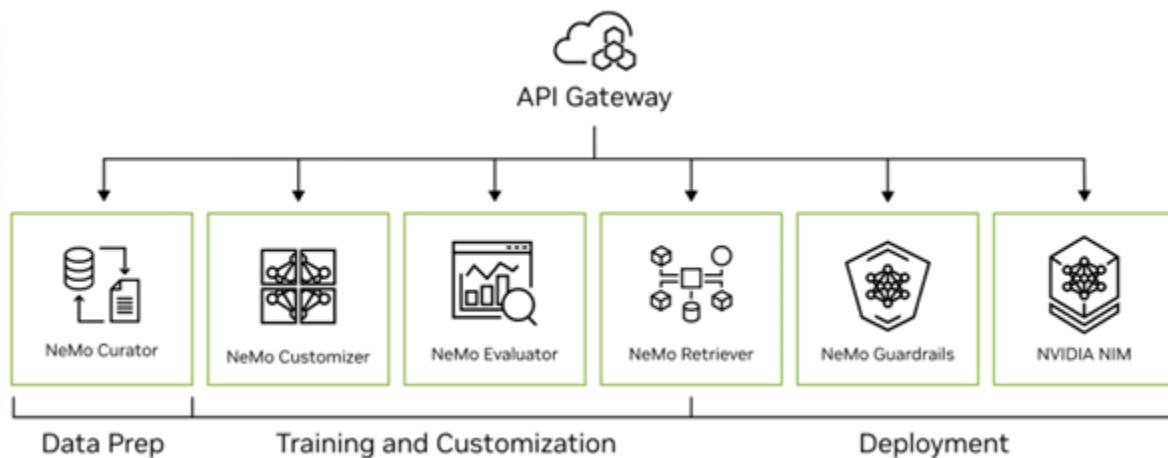
## ۵- هفت - سه: سفارشی‌سازی مدل‌های زبانی بزرگ (LLM‌ها)

در حالی که LLM‌های عمومی که با مهندسی پرامپت یا تنظیم دقیق سبک بهبود یافته‌اند، به سازمان‌ها این امکان را داده‌اند که پروژه‌های موفق نمونه‌سازی (Proof of Concept) را به انجام برسانند، انتقال به محیط تولید چالش‌های اضافی را به همراه دارد. شکل ۱۰.۳ چرخه سفارشی‌سازی دقیق LLM NVIDIA را نشان می‌دهد و راهنمایی‌های ارزشمندی برای سازمان‌هایی که در حال آماده‌سازی برای استقرار مدل‌های سفارشی در یک محیط تولیدی هستند، ارائه می‌دهد.

### ۱. انتخاب یا توسعه مدل

NVIDIA مجموعه‌ای از مدل‌های از پیش آموزش دیده، از B۸ تا B۴۳ پارامتر، ارائه می‌دهد و از ادغام سایر مدل‌های متن‌باز به هر اندازه‌ای پشتیبانی می‌کند. به علاوه، کاربران می‌توانند مدل‌های خود را توسعه دهند، که با گرداوری داده‌ها آغاز می‌شود. این شامل انتخاب، برچسب‌گذاری، پاکسازی، اعتبارسنجی و یکپارچه‌سازی داده‌ها می‌شود. این فرآیند که بهتر است مهندسی داده نامیده شود، شامل تحلیل‌های اضافی، طراحی ذخیره‌سازی، ارزیابی نتایج آموزش مدل و گنجاندن یادگیری تقویتی با بازخورد انسانی (RLHF) است.

در حالی که ساخت یک مدل پایه سفارشی معمولاً پرهزینه، پیچیده و زمانبر است، اکثر شرکت‌ها ترجیح می‌دهند از یک مدل از پیش آموزش‌دیده شروع کنند و بر روی سفارشی‌سازی تمرکز کنند.



شکل ۱۰.۳: چارچوب Nvidia NeMo برای سفارشی‌سازی و استقرار مدل‌های زبانی بزرگ (LLM‌ها). چارچوب NeMo برای سفارشی‌سازی و استقرار تمام عیار مدل‌های زبانی بزرگ (LLM‌ها) طراحی شده است. این نمودار فرآیند را از گردآوری داده‌ها و آموزش توزیع شده مدل‌های پایه، از طریق سفارشی‌سازی مدل، تا استنتاج شتاب‌زده با کنترل‌های لازم نشان می‌دهد. این پلتفرم به توسعه‌دهندگان هوش مصنوعی امکان می‌دهد که پاسخ‌های ایمن و ارجاعی در زمینه خاص را در برنامه‌های سازمانی ادغام کنند و اطمینان حاصل کنند که LLM‌ها به طور مؤثر برای وظایف و صنایع خاص تنظیم شده‌اند. چارچوب NeMo، که با پشتیبانی OpenAI GPT شرکت Nvidia AI Enterprise همراه است، همچنین از مدل‌های پایه از پیش آموزش‌دیده مختلف مانند خانواده GPT دارد. به خوبی پشتیبانی می‌کند و اطمینان حاصل می‌کند که در استقرارهای هوش مصنوعی مقیاس‌پذیری و قابلیت اطمینان وجود دارد. (منبع: [۸۵])

## ۲. سفارشی‌سازی مدل

سفارشی‌سازی مدل شامل بهینه‌سازی عملکرد با استفاده از مجموعه‌داده‌های خاص وظیفه و تنظیم وزن‌های مدل است. NeMo دستورالعمل‌هایی برای سفارشی‌سازی ارائه می‌دهد و شرکت‌ها می‌توانند مدل‌هایی را انتخاب کنند که از پیش برای وظایف خاص تنظیم شده‌اند و سپس آن‌ها را با داده‌های اختصاصی دقیق تنظیم کنند.

## ۳. استنتاج

استنتاج به اجرای مدل‌ها بر اساس پرسش‌های کاربر اشاره دارد. این مرحله شامل در نظر گرفتن عوامل سخت‌افزاری، معماری و عملکرد است که تأثیر قابل توجهی بر قابلیت استفاده و هزینه در تولید دارند.

## ۴. کنترل‌های لازم

NVIDIA از کنترل‌ها به عنوان خدمات میانجی بین مدل‌ها و برنامه‌ها استفاده می‌کند. این خدمات ورودی‌های ارسال شده را برای انطباق با سیاست‌ها بررسی می‌کنند، مراحل داوری یا هماهنگ سازی را اجرا می‌کنند و

اطمینان حاصل می‌کنند که پاسخ‌های مدل مطابق با سیاست‌ها هستند. کنترل‌ها کمک می‌کنند تا ارتباط، دقیق، ایمنی، حریم خصوصی و امنیت حفظ شود.

#### ۵. برنامه‌ها

چارچوب NVIDIA برنامه‌های سازمانی را به عنوان آماده برای LLM ارائه می‌دهد، اگرچه همیشه این‌گونه نیست. برنامه‌های موجود ممکن است به LLM‌ها متصل شوند تا ویژگی‌های جدیدی را فعال کنند. با این حال، ایجاد دستیارهایی برای دسترسی به دانش یا اجرای وظایف معمولاً شامل طراحی برنامه‌های جدید به‌طور خاص برای رابطه‌های زبان طبیعی است.

### ۵- هفت - جهار: آموزش‌ها

۱. مقدمه‌ای بر NVIDIA NeMo - آموزش و مثال<sup>۱۷۴</sup>.

۲. نحوه تنظیم دقیق یک مدل دو زبانه NMT Riva با استفاده از [Nvidia NeMo<sup>175</sup>](#)

---

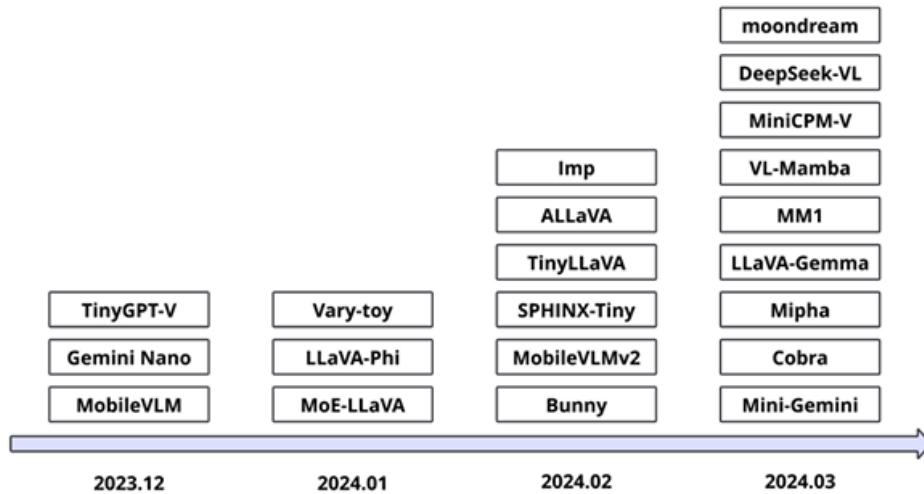
<sup>174</sup> <https://medium.com/@khang.pham.exxact/introduction-to-nvidia-nemo-tutorial-example-478f6ba6b160>

<sup>175</sup> <https://docs.nvidia.com/deeplearning/riva/user-guide/docs/tutorials/nmt-python-advanced-finetune-nmt-model-with-nemo.html>

## فصل یازدهم

### مدل‌های چندرسانه‌ای و تنظیم دقیق آن‌ها

یک مدل چندرسانه‌ای، مدلی در یادگیری ماشین است که می‌تواند اطلاعات را از چندین نوع رسانه، مانند تصاویر، ویدئوها و متن، پردازش کند. به عنوان مثال، مدل چندرسانه‌ای گوگل به نام Gemini [۸۶] می‌تواند عکسی از یک بشقاب کلوچه را تحلیل کرده و در پاسخ یک دستور تهیه کتبی تولید کند و همچنین می‌تواند به صورت معکوس این کار را نیز انجام دهد. تفاوت بین هوش مصنوعی مولد و هوش مصنوعی چندرسانه‌ای این است که هوش مصنوعی مولد به استفاده از مدل‌های یادگیری ماشین برای ایجاد محتوای جدید، مانند متن، تصویر، موسیقی، صدا و ویدیو، معمولاً از یک نوع ورودی اشاره دارد. هوش مصنوعی چندرسانه‌ای این قابلیت‌های مولد را با پردازش اطلاعات از چندین نوع رسانه، شامل تصاویر، ویدئوها و متن، گسترش می‌دهد. این امر به هوش مصنوعی اجازه می‌دهد که حالت‌های حسی مختلف را درک و تفسیر کند و به کاربران این امکان را می‌دهد که انواع مختلفی از داده‌ها را وارد کرده و در عوض، طیف متنوعی از انواع محتوا را دریافت کنند.



شکل ۱۱.۱: زمانبندی توسعه مدل‌های چندرسانه‌ای. این شکل پیشرفت مدل‌های چندرسانه‌ای مهم را نشان می‌دهد و انتشارهای کلیدی از شرکت‌های فناوری بزرگ و مؤسسات تحقیقاتی را از دسامبر ۲۰۲۳ تا مارس ۲۰۲۴ هایلایت می‌کند. این زمانبندی مدل‌هایی مانند TinyGPT-V و Gemini Nano از گوگل را به همراه نوآوری‌های دیگری نظیر DeepSeek-VL، MoE-LLAVA و LLAVA Gemma به نمایش می‌گذارد و پیشرفت سریع در فناوری‌های هوش مصنوعی چندرسانه‌ای را نشان می‌دهد (منبع: [۸۷]).

## یازده - یک: مدل‌های زبان بصری (VLMs)

مدل‌های زبان بصری شامل مدل‌های چندرسانه‌ای هستند که قادر به یادگیری از ورودی‌های تصویری و متنی می‌باشند. این مدل‌ها به دسته‌ای از مدل‌های مولد تعلق دارند که از داده‌های تصویری و متنی برای تولید خروجی‌های متنی استفاده می‌کنند. این مدل‌ها، به ویژه در مقیاس‌های بزرگ‌تر، قابلیت‌های قوی در یادگیری بدون نمونه (zero-shot) را نشان می‌دهند، در وظایف مختلف به خوبی تعمیم می‌یابند و انواع مختلف داده‌های بصری مانند اسناد و صفحات وب را به طور مؤثری مدیریت می‌کنند. کاربردهای معمول شامل تعاملات گفتگویی با تصاویر، تفسیر تصاویر بر اساس دستورالعمل‌های متنی، پاسخ‌گویی به سوالات مربوط به محتویات بصری، درک اسناد، تولید زیرنویس برای تصاویر و موارد دیگر است. برخی از مدل‌های زبان بصری پیشرفته نیز می‌توانند ویژگی‌های فضایی موجود در تصاویر را درک کنند. آن‌ها می‌توانند به درخواست، جعبه‌های محدود کننده (bounding boxes) یا ماسک‌های تقسیم‌بندی (segmentation masks) تولید کنند تا موضوعات خاص را شناسایی یا جداسازی کنند، یا به سوالات مربوط به موقعیت‌های نسبی یا مطلق آن‌ها پاسخ دهند. چشم‌انداز مدل‌های بزرگ زبان بصری به خاطر تنوع قابل توجهی در داده‌های آموزشی، تکنیک‌های رمزگذاری تصویر و در نتیجه قابلیت‌های عملکردی آن‌ها مشخص می‌شود.

### یازده - یک - یک - معماری

مدل‌های زبان-بصری به خوبی اطلاعات بصری و متنی را ادغام می‌کنند و از سه جزء اساسی بهره می‌برند:

- **رمزگذار تصویر (Image Encoder):** این جزء داده‌های بصری (تصاویر) را به فرمی تبدیل می‌کند که مدل بتواند آن را پردازش کند.

- **رمزگذار متن (Text Encoder):** مشابه رمزگذار تصویر، این جزء داده‌های متنی (کلمات و جملات) را به فرمی تبدیل می‌کند که مدل بتواند آن را درک کند.

- **استراتژی ادغام (Fusion Strategy):** این جزء اطلاعات را از هر دو رمزگذار تصویر و متن ترکیب می‌کند و دو نوع داده را به یک نمایش یکپارچه تبدیل می‌کند.

این عناصر به طور مشترک کار می‌کنند و فرآیند یادگیری مدل (توابع زیان) به طور خاص برای معماری و استراتژی یادگیری به کار رفته طراحی شده است. اگرچه مفهوم مدل‌های زبان-بصری جدید نیست، اما ساخت آن‌ها به طور قابل توجهی تکامل یافته است. مدل‌های اولیه از توصیف‌های تصویر به طور دستی و بردارهای کلمات از پیش آموزش‌دیده استفاده می‌کردند. اما مدل‌های مدرن از ترانسفورمرها - یک معماری شبکه عصبی پیشرفته - برای رمزگذاری هر دو تصویر و متن استفاده می‌کنند. این رمزگذاران می‌توانند ویژگی‌ها را به صورت مستقل یا مشترک یاد بگیرند.

یک جنبه حیاتی این مدل‌ها، پیش‌آموزش (pre-training) است. قبل از اینکه به وظایف خاصی اعمال شوند، مدل‌ها بر روی مجموعه‌های داده گستردگی با استفاده از اهداف بهدقت انتخاب شده آموزش می‌بینند. این پیش‌آموزش آن‌ها را با دانش بنیادی لازم برای برتری در برنامه‌های مختلف پایین‌دست مجهز می‌کند. در ادامه یکی از معماری‌های نمونه مدل‌های زبان بصری ارائه شده است.

### یازده – یک – دو: یادگیری متضاد (Contrastive Learning)

یادگیری متضاد یک تکنیک است که بر درک تفاوت‌ها بین نقاط داده مرکز است. این تکنیک یک نمره شباهت بین نمونه‌ها محاسبه کرده و هدف آن حداقل کردن زیان متضاد (contrastive loss) است، که به ویژه در یادگیری نیمه‌نظرارت (semi-supervised learning) مفید است، جایی که تعداد محدودی از نمونه‌های برچسب‌گذاری شده، فرآیند بهینه‌سازی را برای طبقه‌بندی نقاط داده دیده‌نشده هدایت می‌کند.

#### چگونه کار می‌کند؟

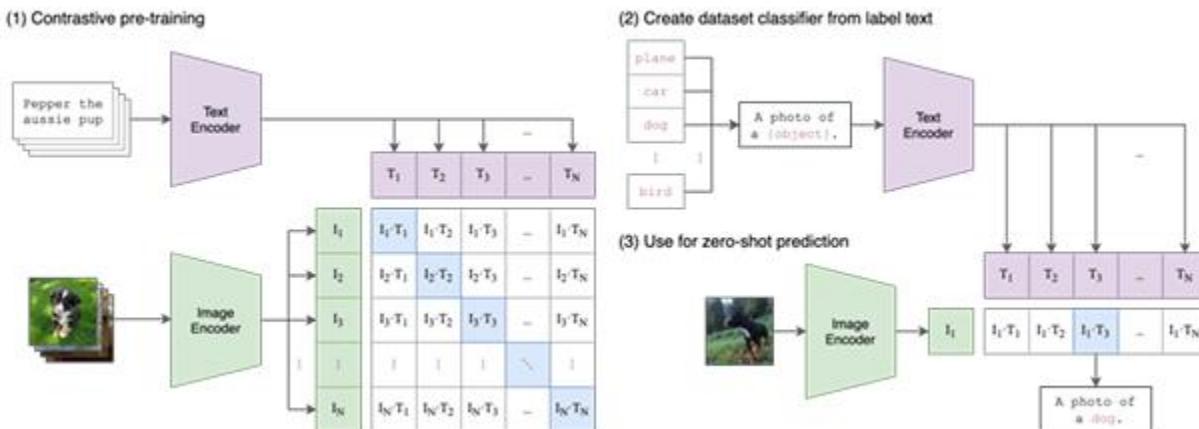
به عنوان مثال، برای شناسایی یک گربه، یادگیری متضاد تصویر یک گربه را با یک تصویر مشابه گربه و یک تصویر سگ مقایسه می‌کند. مدل یاد می‌گیرد که بین گربه و سگ تمایز قائل شود، با شناسایی ویژگی‌هایی مانند ساختار صورت، اندازه بدن و نوع پشم. با تعیین اینکه کدام تصویر به تصویر "لنگر" (anchor) نزدیک‌تر است، مدل کلاس آن را پیش‌بینی می‌کند.

مدل CLIP از یادگیری متضاد برای محاسبه شباهت بین رمزگذاری‌های متن و تصویر از طریق رمزگذاران متنی و بصری استفاده می‌کند. این مدل یک فرآیند سه مرحله‌ای برای پیش‌بینی بدون نمونه (zero-shot) دنبال می‌کند:

- پیش‌آموزش (Pre-training): آموزش یک رمزگذار متن و تصویر برای یادگیری جفت‌های تصویر-متن.

- تبدیل زیرنویس (Caption Conversion): تبدیل کلاس‌های مجموعه داده‌های آموزشی به زیرنویس‌ها.

- پیش‌بینی بدون نمونه (Zero-Shot Prediction): برآورد بهترین زیرنویس برای یک تصویر ورودی خاص بر اساس شباهت‌های یادگرفته شده.



شکل ۱۱.۲: فرآیند پیش‌آموزش کنتراست برای مدل‌های چندرسانه‌ای. این شکل فرآیند پیش‌آموزش کنتراست را نشان می‌دهد که در آن رمزگذارهای متن و تصویر آموزش می‌بینند تا نمایش‌های مربوط به هر دو نوع داده را هم راستا کنند. مرحله ۱ شامل پیش‌آموزش کنتراست با جفت‌سازی داده‌های متنی و تصویری است، در حالی که مرحله ۲ ایجاد یک طبقه‌بند مجموعه‌داده با استفاده از متن برچسب‌گذاری شده توسط رمزگذار متن را نشان می‌دهد. مرحله ۳ کاربرد مدل برای پیش‌بینی بدون آموزش را با بهره‌گیری از رمزگذارهای متن و تصویر پیش‌آموزش دیده نمایش می‌دهد. این روش به مدل امکان می‌دهد تا در وظایف مختلف بدون نیاز به تنظیمات خاص وظیفه، عمومی‌سازی کند (به نقل از [۸۸]).

## یازده – دو: تنظیم دقیق مدل‌های چندرسانه‌ای

برای تنظیم دقیق یک مدل زبان بزرگ چندرسانه‌ای (MLM)، تکنیک‌های PEFT مانند LoRA و QLoRA می‌توانند مورد استفاده قرار گیرند. فرآیند تنظیم دقیق برای برنامه‌های چندرسانه‌ای مشابه آنچه برای مدل‌های زبان بزرگ انجام می‌شود، است و تنها تفاوت اصلی در ماهیت داده‌های ورودی است. علاوه بر LoRA، که از تکنیک‌های تجزیه ماتریسی برای کاهش تعداد پارامترها استفاده می‌کند، ابزارهای دیگری مانند LLM-Adapters و Adapters<sup>۳</sup> [۸۹] نیز می‌توانند به‌طور مؤثری مورد استفاده قرار گیرند. LLM-Adapters مختلف آداتور را به معنایی مدل پیش‌آموزش دیده ادغام می‌کند، که این امر امکان تنظیم دقیق مؤثر پارامترها برای وظایف متعدد را با بهروزرسانی تنها پارامترهای آداتور در حالی که پارامترهای مدل پایه ثابت باقی می‌مانند، فراهم می‌آورد. آداتورهای تزریقی با مهار و تقویت فعالیت‌های داخلی، با یادگیری وکتورهایی برای وزن‌دهی پارامترهای مدل از طریق ضربهای فعالیت، عملکرد را بهبود می‌بخشد و از عملکرد robust few-shot و ترکیب وظایف بدون نیاز به تنظیمات دستی پشتیبانی می‌کند. علاوه بر این، تکنیک‌های تطبیق پویا Mانند DyLoRA [۹۰] اجازه می‌دهند که بلوک‌های انطباق با رتبه پایین در رتبه‌های مختلف آموزش داده شوند و فرآیند یادگیری را با مرتب‌سازی نمایش‌ها در حین آموزش بهینه کنند. LoRA-FA [۹۱]، که یک نوع از است، فرآیند تنظیم دقیق را با فریز کردن اولین ماتریس با رتبه پایین پس از اولیه‌سازی و استفاده از آن به عنوان

یک نمایش تصادفی در حالی که دیگر ماتریس آموزش داده می‌شود، بهینه می‌کند و بدین ترتیب تعداد پارامترها را بدون به خطر انداختن عملکرد نصف می‌کند.

ماژول Skip Attention مؤثر (EAS) [۹۲] یک روش تنظیم جدید کارآمد از نظر پارامتر و محاسبات برای MLLMs ارائه می‌دهد که هدف آن حفظ عملکرد بالا در حالی که هزینه‌های پارامتر و محاسبات برای وظایف پایین‌دستی را کاهش می‌دهد، می‌باشد. با این حال، MemVP [۹۳] به این رویکرد انتقاد می‌کند و یادآور می‌شود که هنوز طول ورودی مدل‌های زبانی را افزایش می‌دهد. برای حل این مسئله، MemVP پرامپ‌های بصری را با وزن‌های شبکه‌های پیش‌روندۀ ادغام می‌کند و بدین ترتیب دانش بصری را برای کاهش زمان آموزش و تأخیر استنتاج تزریق می‌کند که در نهایت از روش‌های قبلی PEFT پیشی می‌گیرد.

## یازده—دو—یک: نظمیم دقیق کامل پارامترها

روش‌هایی مانند آنچه توسط LOMO [۹۴] و MeZO [۹۵] معرفی شده‌اند، راه حل‌های جایگزینی را با تمرکز بر کارایی حافظه ارائه می‌دهند. LOMO از یک تکنیک بهینه‌سازی با حافظه پایین استفاده می‌کند که از نزولی تصادفی (SGD) مشتق شده و مصرف حافظه معمولاً مرتبط با بهینه‌ساز ADAM را کاهش می‌دهد. از سوی دیگر، MeZO یک بهینه‌ساز کارآمد از نظر حافظه را ارائه می‌دهد که تنها به دو گذر پیش‌رو برای محاسبه گرادیان نیاز دارد و به تنظیم دقیق جامع مدل‌های بزرگ با یک اثر حافظه معادل با استنتاج امکان می‌دهد [۸۷].

## یازده—دو—دو: مطالعه موردی تنظیم دقیق MLLMs برای حوزه پزشکی

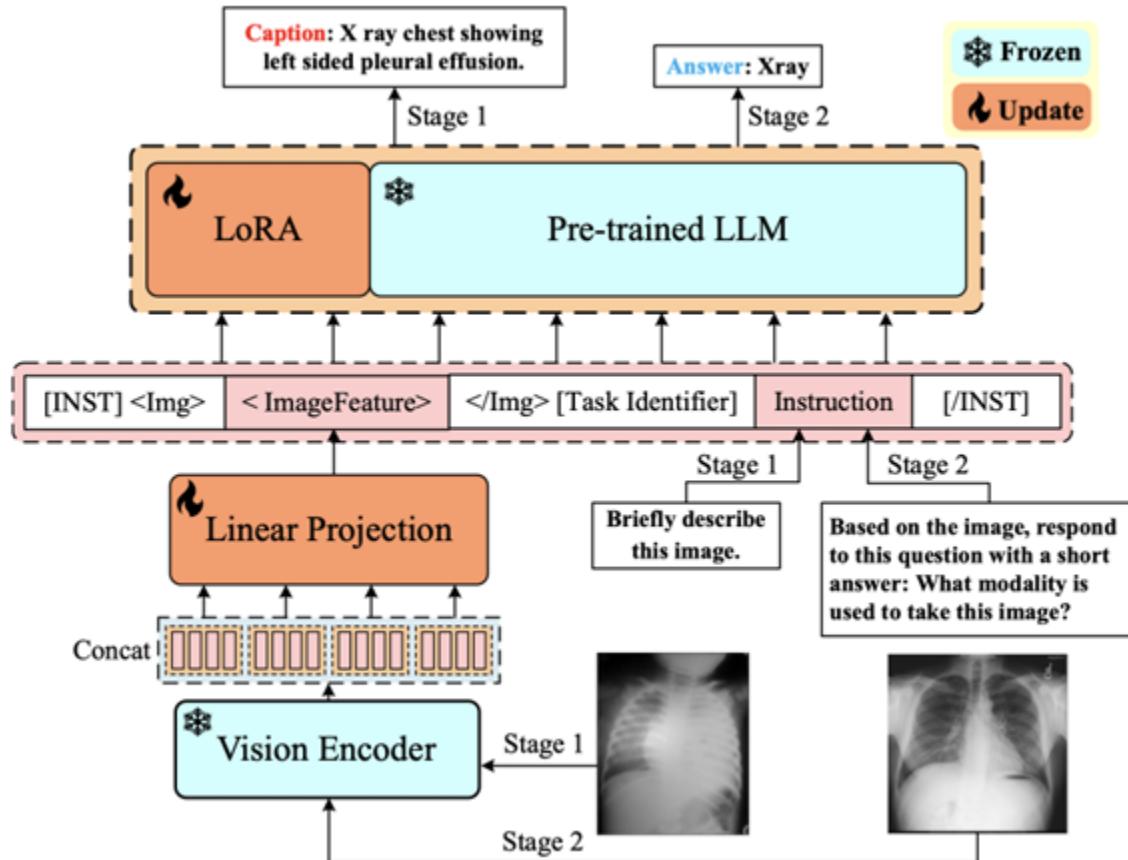
بخش زیر یک مطالعه موردی درباره تنظیم دقیق MLLMs برای وظیفه پاسخ‌گویی به سوالات بصری (VQA) را ارائه می‌دهد. در این مثال، یک PEFT برای تنظیم دقیق MLLM به‌طور خاص طراحی شده برای برنامه‌های MedVQA ارائه می‌شود. برای اطمینان از اندازه‌گیری دقیق عملکرد، ارزیابی‌های انسانی انجام شد که نشان می‌دهد مدل به دقت کلی ۸۱.۹٪ دست می‌یابد و با حاشیه قابل توجه ۲۶٪ در دقت مطلق بر مدل GPT-4v پیشی می‌گیرد.

این مدل از سه مولفه تشکیل شده است: رمزگذار بصری، یک مدل زبان بزرگ پیش‌آموزش‌دیده (LLM) برای مدیریت ورودی‌های چندرسانه‌ای و تولید پاسخ‌ها، و یک لایه خطی واحد برای پیش‌بینی توکن‌ها از فضای رمزگذاری بصری به فضای LLM، همانطور که در شکل ۱۱.۳ نشان داده شده است.

پایه‌ای از نوع ViT، EVA، Vision Transformer (ViT)، توکن‌های تصویری را به توکن‌های بصری رمزگذاری می‌کند و وزن‌های مدل در حین فرآیند تنظیم دقیق ثابت باقی می‌مانند. از تکنیک MiniGPT-v2 استفاده می‌شود که چهار توکن متوالی را به یک توکن بصری گروه‌بندی می‌کند تا مصرف منابع را به‌طور کارآمد کاهش دهد و با ادغام بر روی بعد رمزگذاری عمل می‌کند.

این توکن‌های بصری گروه‌بندی شده سپس از طریق لایه پیش‌بینی شده پردازش می‌شوند که منجر به تولید توکن‌ها (به طول ۴۰۹۶) در فضای LLM می‌شود. یک الگوی پرامپ چندرسانه‌ای اطلاعات بصری و سوالی را ادغام

می‌کند که به LLM پیش‌آموزش دیده، LLaMA2-chat(7B)، برای تولید پاسخ وارد می‌شود. تکنیک انتباق با رتبه پایین (LoRA) برای تنظیم دقیق مؤثر اعمال می‌شود و باقی‌مانده LLM در حین تنظیم دقیق پایین‌دستی ثابت باقی می‌ماند. از جستجوی پرتو با عرض ۱ استفاده می‌شود.



شکل ۱۱.۳: نمای کلی معماری LoRA Med VQA که مدل زبان پیش‌آموزش دیده (LLM) را با یک رمزگذار بصری برای وظایف پرسش و پاسخ بصری پزشکی ادغام می‌کند. این معماری شامل مراحل پردازش تصاویر و تولید پاسخ‌های مرتبه با زمینه است و ادغام مدل‌های بصری و زبانی را در یک زمینه پزشکی نشان می‌دهد (منبع [۹۶]).

پرسش چندسانه‌ای شامل تصاویر ورودی، سوالات و یک توکن خاص برای وظایف VQA است که از الگوی MiniGPT-v2 پیروی می‌کند. در شکل ۱۱.۳، ویژگی‌های تصویری به دست آمده از **projection** خطی به عنوان **ImageFeature** علامت‌گذاری شده‌اند، در حالی که سوالات مربوطه به عنوان دستورالعمل‌های متنی عمل می‌کنند. توکن ویژه [VQA] به عنوان شناسنده وظیفه استفاده می‌شود و الگوی آموزشی چندسانه‌ای کامل را تشکیل می‌دهد:

"[INST]<img><ImageFeature></img>[VQA] Instruction [/INST]".

## آموزش مدل

وزن‌های MiniGPT-v2 که بر روی داده‌های عمومی پیش‌آموزش دیده شده‌اند، در دو مرحله با استفاده از داده‌های چندرسانه‌ای پزشکی بیشتر آموزش دیده می‌شوند. تکنیک LoRA برای آموزش مؤثر استفاده می‌شود و تنها بخش کوچکی از کل مدل به روزرسانی می‌شود، همان‌طور که در زیر توضیح داده شده است:

- آموزش با استفاده از توصیف تصاویر: در این مرحله، مدل با استفاده از مجموعه داده ROCO (تصویر-توصیف پزشکی) آموزش داده می‌شود که شامل جفت‌های تصویر-توصیف پزشکی با طول‌های مختلف است. الگوی آموزشی استفاده شده

“<Img><ImageHere></Img>[caption] <instruction>”

است که در آن دستورالعمل به صورت تصادفی از میان چهار گزینه انتخاب می‌شود، مانند "این تصویر را به طور مختصر توصیف کنید." در حین آموزش، تنها لایه projection خطی و لایه LoRA در LLM به روزرسانی می‌شوند، در حالی که سایر قسمت‌های مدل ثابت باقی می‌مانند.

- آموزش در VQA: در مرحله دوم، مدل در مجموعه داده Med-VQA (VQA-RAD) که شامل سه‌تایی از تصاویر، سوالات و پاسخ‌ها است، آموزش داده می‌شود. با پیروی از الگوی آموزشی پیشنهادی در v2، الگوی استفاده شده به صورت زیر است:

“[INST] <img><ImageFeature></img>[VQA] Instruction [/INST]”

که در آن دستورالعمل به صورت "بر اساس تصویر، به این سوال با یک پاسخ کوتاه پاسخ دهید: سوال" است که در آن "سؤال" به سوال مربوط به تصویر پزشکی داده شده اشاره دارد. انگیزه برای تولید پاسخ‌های کوتاه این است که با داده‌های برچسب‌گذاری شده موجود در VQA-RAD مطابقت داشته باشد، جایی که پاسخ‌ها معمولاً در هر دو نوع QA باز و بسته کوتاه هستند. مشابه مرحله اول، رمزگذار بصری و LLM ثابت باقی می‌مانند و تنها لایه‌های projection خطی و LoRA در LLM به روزرسانی می‌شوند.

## یازده - سه: کاربردهای مدل‌های چندرسانه‌ای

۱. شناسایی حرکات: این مدل‌ها حرکات انسانی را تفسیر و شناسایی می‌کنند که برای ترجمه زبان اشاره حیاتی است. مدل‌های چندرسانه‌ای ارتباط فراگیر را با پردازش حرکات و تبدیل آن‌ها به متن یا گفتار تسهیل می‌کنند.

۲. خلاصه‌سازی ویدئو: مدل‌های چندرسانه‌ای می‌توانند ویدیوهای طولانی را با استخراج عناصر بصری و صوتی کلیدی خلاصه کنند. این قابلیت مصرف محتوا را ساده می‌کند، امکان مرور مؤثر محتوا را فراهم می‌آورد و مدیریت محتوای ویدیویی را بهبود می‌بخشد.

**۳. DALL-E**: مثال قابل توجهی از هوش مصنوعی چندرسانه‌ای است که تصاویر را از تصویف‌های متنی تولید می‌کند. این فناوری امکانات خلاقانه را در تولید محتوا و روایت بصری گسترش می‌دهد و در زمینه‌های هنر، طراحی، تبلیغات و موارد دیگر کاربرد دارد.

**۴. ابزارهای آموزشی**: مدل‌های چندرسانه‌ای تجربه‌های یادگیری را با ارائه محتوا آموزشی تعاملی که به نشانه‌های بصری و کلامی از دانش آموزان پاسخ می‌دهد، تقویت می‌کنند. آن‌ها جزء جدایی‌ناپذیر پلتفرم‌های یادگیری تطبیقی هستند که محتوا و دشواری را بر اساس عملکرد و بازخورد دانش آموزان تنظیم می‌کنند.

**۵. دستیارهای مجازی**: مدل‌های چندرسانه‌ای دستیارهای مجازی را با درک و پاسخ به دستورات صوتی در حالی که داده‌های بصری را برای تعامل جامع با کاربر پردازش می‌کنند، قدرت می‌بخشند. آن‌ها برای اتوماسیون خانه‌های هوشمند، دستگاه‌های کنترل شونده با صدا و دستیاران شخصی دیجیتال ضروری هستند.

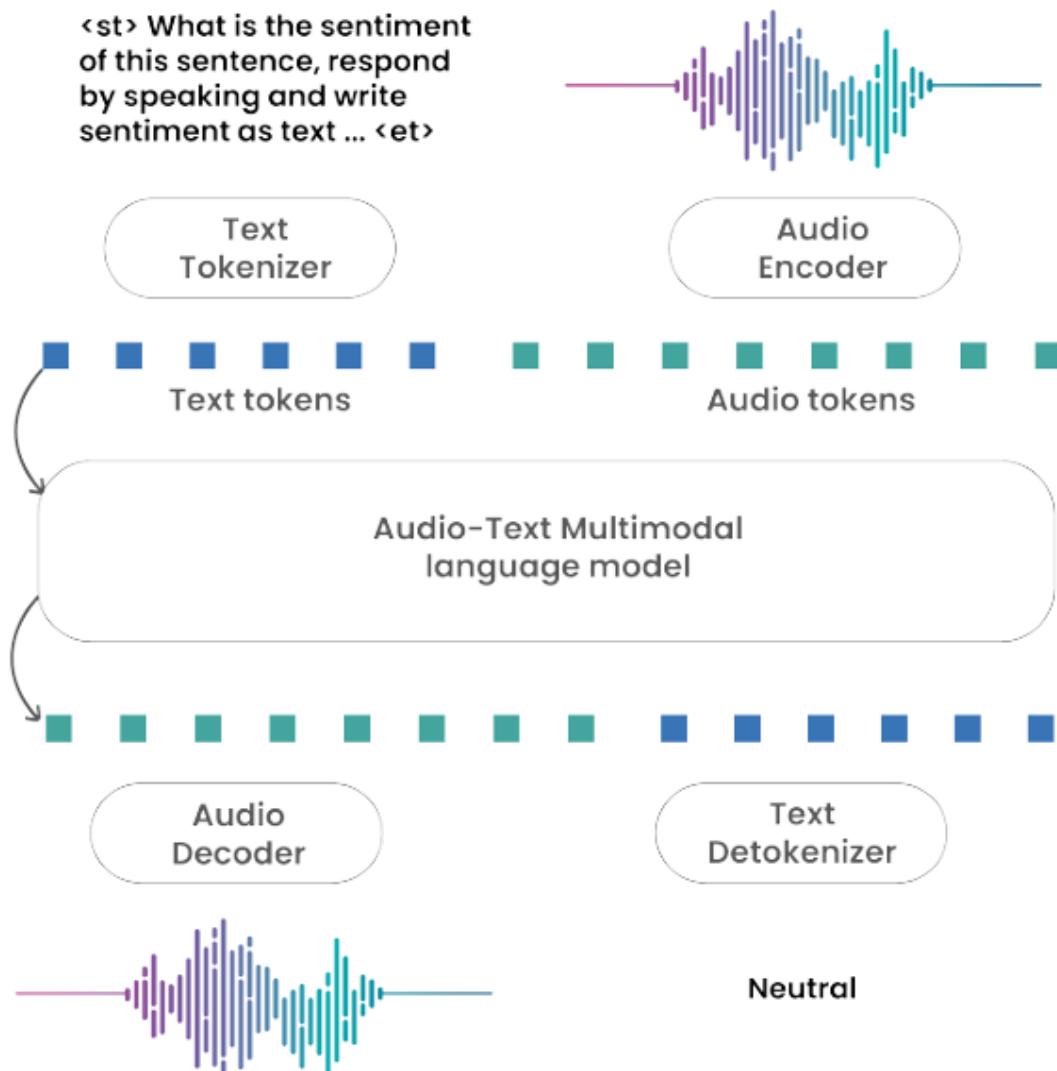
## یازده – چهار: مدل‌های زبان بزرگ صوتی یا گفتاری

مدل‌های صوتی یا گفتاری LLM‌ها به گونه‌ای طراحی شده‌اند که زبان انسانی را بر اساس ورودی‌های صوتی درک و تولید کنند. این مدل‌ها در کاربردهایی مانند شناسایی گفتار، تبدیل متن به گفتار و وظایف درک زبان طبیعی به کار می‌روند. این مدل‌ها عموماً بر روی مجموعه‌های داده بزرگ پیش‌آموزش داده می‌شوند تا الگوهای زبانی عمومی را یاد بگیرند و سپس بر روی وظایف یا حوزه‌های خاص برای بهبود عملکرد، به صورت دقیق‌تر آموزش می‌بینند.

مدل‌های بزرگ زبان صوتی و گفتاری (LLMs) پیشرفت قابل توجهی در ادغام پردازش زبان با سیگنال‌های صوتی را نشان می‌دهند. این مدل‌ها از یک مدل زبان بزرگ و robust به عنوان پایه اصلی استفاده می‌کنند که برای مدیریت داده‌های چندرسانه‌ای با افزودن توکن‌های صوتی سفارشی بهبود یافته است. این تحول به مدل‌ها اجازه می‌دهد تا در یک فضای چندرسانه‌ای مشترک یاد بگیرند و عمل کنند، جایی که هر دو سیگنال متنی و صوتی می‌توانند به طور مؤثر پردازش شوند.

برخلاف متن که ذاتاً گستته است، سیگنال‌های صوتی پیوسته‌اند و نیاز به گستته‌سازی به توکن‌های صوتی قابل مدیریت دارند. تکنیک‌هایی مانند HuBERT و wav2vec برای این منظور به کار می‌روند و صدا را به قالبی توکنیزه‌شده تبدیل می‌کنند که مدل زبان بزرگ (LLM) بتواند آن را در کنار متن پردازش کند. مدل که به طور معمول از نوع خودبازگشتی (autoregressive) و مبتنی بر دیکودر است، با ترکیبی از وظایف خودناظارتی (self-supervised) مانند پیش‌بینی توکن‌های ماسک‌شده در متن و صدای درهم‌آمیخته و تنظیم دقیق نظارت شده برای وظایف خاصی مانند رونویسی یا تحلیل احساسات، از پیش آموزش داده می‌شود. این قابلیت برای پردازش و تولید همزمان صوت و متن، دامنه گسترده‌ای از کاربردها، از جمله پاسخ به سوالات صوتی تا تشخیص احساسات مبتنی بر گفتار، را امکان‌پذیر می‌سازد و مدل‌های زبانی بزرگ صوتی و گفتاری را به ابزاری چندکاره در هوش مصنوعی چندرسانه‌ای تبدیل می‌کند. شکل ۱۱.۴ نمونه‌ای از معماری مدل زبانی چندرسانه‌ای صوتی را

نشان می‌دهد. در این ساختار، یک پرسش ورودی شامل دستورالعمل‌ها به هر دو صورت متن و صوت ارائه می‌شود. صوت با استفاده از یک توکن‌ساز صوتی (audio tokenizer) به توکن تبدیل می‌شود. سپس مدل چندسانه‌ای این توکن‌های متنی و صوتی را با هم ترکیب کرده و گفتار تولیدشده را از طریق یک ووکودر<sup>۱۷۶</sup> (که به عنوان دیکودر صوتی نیز شناخته می‌شود) ایجاد می‌کند.



شکل ۱۱.۴: معماری مدل زبان صوتی-متنی چندسانه‌ای که ورودی‌های متنی و صوتی را برای پردازش چندسانه‌ای پیشرفتهدغفام می‌کند. این معماری از توکن‌سازهای متنی و کدگذار/توکن‌سازهای صوتی برای تبدیل ورودی‌ها به توکن‌ها استفاده می‌کند که سپس توسط مدل زبان صوتی-متنی پردازش می‌شوند. این مدل از پردازش گفتار گسسته و پیوسته پشتیبانی می‌کند و امکان انجام وظایفی مانند تحلیل احساسات و تولید پاسخ‌ها به زبان طبیعی را فراهم می‌آورد. توکن‌های صوتی با استفاده از یک vocoder به‌طور دقیق‌تر پردازش می‌شوند، در حالی که توکن‌های متنی برای تولید خروجی‌های متنی منسجم، دوباره به حالت اولیه خود برمی‌گردند (منبع: [۹۹]).

<sup>۱۷۶</sup> Vocoder

مدل‌های صوتی و گفتاری LLM مانند [100] AudioLM، AudioPaLM [101] و انواع مختلفی از مدل‌ها مانند Whisper و LLaMA، قابلیت‌هایی برای درک و تولید داده‌های صوتی شامل تبدیل گفتار به متن (STT)، تبدیل متن به گفتار (TTS) و ترجمه گفتار به گفتار (STS) را ادغام می‌کنند. این مدل‌ها نشان داده‌اند که LLM‌ها که در ابتدا برای متن طراحی شده‌اند، می‌توانند به طور مؤثری برای وظایف صوتی از طریق تکنیک‌های پیچیده توکن‌سازی و آموزش دقیق سازگار شوند.

#### یازده - چهار - یک: توکن‌سازی و پیش‌پردازش

یک جنبه کلیدی در سازگاری LLM‌ها با صوت، توکن‌سازی داده‌های صوتی به نمایه‌های گسسته است که مدل می‌تواند پردازش کند. به عنوان مثال، AudioPaLM و AudioLM از ترکیبی از توکن‌های صوتی و معنایی استفاده می‌کنند. توکن‌های صوتی جنبه سنتز صوتی با کیفیت بالا را ضبط می‌کنند، در حالی که توکن‌های معنایی به حفظ انسجام ساختاری درازمدت در صوت تولید شده کمک می‌کنند. این رویکرد دو توکنی به مدل‌ها اجازه می‌دهد تا هم پیچیدگی‌های موج‌های صوتی و هم محتواهای معنایی گفتار را مدیریت کنند.

#### یازده - چهار - دو: تکنیک‌های تنظیم دقیق

- تنظیم دقیق مدل‌های زبان بزرگ (LLM) صوتی و گفتاری معمولاً شامل چندین استراتژی کلیدی است:
- **تنظیم دقیق کامل پارامترها:** در این روش، تمامی پارامترهای مدل در طی فرآیند تنظیم دقیق به روزرسانی می‌شوند. به عنوان مثال، مدل‌های LauraGPT و SpeechGPT همه پارامترها را برای تطبیق مدل‌های زبانی بزرگ متنی از پیش آموزش دیده با وظایف مختلف صوتی تنظیم می‌کنند، هرچند این کار از نظر محاسباتی هزینه‌بر است.
  - **تنظیم دقیق لاشه محور:** تکنیک‌هایی مانند LoRA (تطبیق کمرتبه) فقط لاشه‌ها یا مأذول‌های خاصی از مدل را به روزرسانی می‌کنند. این روش به طور قابل توجهی نیازهای محاسباتی را کاهش می‌دهد، در عین حال امکان تطبیق مؤثر را فراهم می‌کند. مدل‌هایی مانند Qwen-Audio از LoRA برای تنظیم دقیق اجزای از پیش آموزش دیده جهت بهبود عملکرد در وظایف تشخیص گفتار بهره می‌برند.
  - **تنظیم دقیق مبتنی بر اجزا:** مدل‌های اخیر، مانند مدل‌هایی که رمزگذار Whisper را یکپارچه می‌کنند، برخی از بخش‌های مدل (مانند رمزگذار گفتار) را ثابت نگه می‌دارند و تنها یک نگاشت خطی یا آداتورهای خاصی را تنظیم می‌کنند تا هماهنگی میان حالت‌های گفتار و متن را فراهم کنند. این روش، فرآیند آموزش را ساده‌تر کرده و بازدهی را افزایش می‌دهد [102].
  - **تنظیم دقیق چند مرحله‌ای:** مدل‌هایی مانند AudioPaLM تنظیم دقیق را در چند مرحله انجام می‌دهند و ابتدا با یک مرحله پیش‌آموزش مبتنی بر متن آغاز می‌کنند، سپس با ترکیبی از وظایف که شامل داده‌های متنی و صوتی است، تنظیم دقیق می‌شوند. این رویکرد مرحله‌ای از توانایی‌های مدل‌های متنی از پیش‌آموزش یافته استفاده کرده و آن‌ها را برای وظایف چند رسانه‌ای تطبیق می‌دهد.

## یازده - چهار - سه: تنظیم دقیق Whisper برای تشخیص خودکار گفتار (ASR)

Whisper<sup>177</sup> یک مدل پیشرفته تشخیص خودکار گفتار (ASR) است که توسط OpenAI توسعه یافته و برای تبدیل زبان گفتاری به متن طراحی شده است. این مدل که بر اساس معماری قدرتمند Transformer ساخته شده، در شناسایی و رونویسی الگوهای مختلف گفتاری در زبان‌ها و لهجه‌های متنوع عملکرد بسیار خوبی دارد. برخلاف مدل‌های ASR سنتی که نیاز به داده‌های برچسب‌گذاری شده گسترش دارند، Whisper از یک مجموعه داده وسیع و یادگیری خودناظارتی بهره می‌برد و به همین دلیل قادر است به طور قابل اعتمادی در محیط‌های پرسر و صدا عمل کرده و با انواع مختلف گفتار سازگار شود. تنوع پذیری و دقت بالای آن، این مدل را به گزینه‌ای ایده‌آل برای کاربردهایی مانند دستیارهای صوتی، خدمات رونویسی، و سیستم‌های چندزبانه تشخیص گفتار تبدیل کرده است.

### چرا تنظیم دقیق Whisper مهم است؟

تنظیم دقیق Whisper برای وظایف خاص تشخیص گفتار (ASR) می‌تواند عملکرد آن را در حوزه‌های تخصصی به طور قابل توجهی بهبود بخشد. اگرچه Whisper از پیش بر روی یک مجموعه داده بزرگ و متنوع آموزش دیده است، ممکن است نتواند به طور کامل جزئیات خاص واژگان یا لهجه‌های موجود در کاربردهای ویژه را درک کند. تنظیم دقیق این امکان را به Whisper می‌دهد که خود را با ویژگی‌های صوتی و اصطلاحات خاص تطبیق دهد و منجر به رونویسی‌های دقیق‌تر و قابل اعتمادتر شود. این فرآیند به ویژه در صنایعی با اصطلاحات تخصصی، مانند حوزه‌های پزشکی، حقوقی، یا فنی سودمند است، جایی که مدل عمومی ممکن است با واژگان خاص این حوزه‌ها دچار مشکل شود.

### مراحل تنظیم دقیق Whisper

- **گردآوری و آماده‌سازی داده‌ها:** یک مجموعه داده مناسب که با حوزه یا وظیفه مورد نظر همخوانی دارد جمع‌آوری کنید. اطمینان حاصل کنید که مجموعه داده شامل نمونه‌های متنوع با رونویسی‌های واضح باشد. فایل‌های صوتی و رونویسی‌ها را پاکسازی و پیش‌پردازش کنید تا در قالبی منسجم و هم‌راستا قرار بگیرند. ابزارهایی مانند FFmpeg<sup>178</sup> می‌توانند به استانداردسازی فرمتهای صوتی و نرخ‌های نمونه‌برداری کمک کنند.

- **افزایش داده‌ها:** برای بهبود استحکام، مجموعه داده را با تنوعاتی مانند سطوح مختلف نویز، لهجه‌ها یا سرعت‌ها افزایش دهید. تکنیک‌هایی مانند افزودن نویز پس‌زمینه، تغییر زیر و بمی صدا یا تغییر سرعت، به مدل کمک می‌کنند تا بهتر به شرایط واقعی تعیین یابد.

- **پیش‌پردازش:** فایل‌های صوتی را به قالبی مناسب برای Whisper تبدیل کنید، که معمولاً به طیف‌نگاشت مل (Mel Spectrograms) یا سایر نمایش‌های زمان-فرکانس تبدیل می‌شود. این تبدیل

<sup>177</sup> <https://openai.com/index/whisper/>

<sup>178</sup> <https://ffmpeg.org/ffmpeg.html>

بسیار مهم است، زیرا Whisper به چنین نمایش‌هایی برای یادگیری و رونویسی گفتار به طور مؤثر متکی است.

- پیکربندی مدل: مدل Whisper را با وزن‌های از پیش آموزش‌دیده مقداردهی اولیه کنید. مدل را برای پشتیبانی از زبان یا تنظیمات خاص حوزه هدف پیکربندی کنید. این شامل تنظیم ابرپارامترهای مناسب مانند نرخ یادگیری و اندازه دسته است که متناسب با اندازه و پیچیدگی مجموعه داده انتخاب می‌شوند.

- آموزش: مدل Whisper را با استفاده از چارچوب‌هایی مانند TensorFlow یا PyTorch بر روی مجموعه داده آماده‌شده تنظیم دقیق کنید. مطمئن شوید که عملکرد مدل را روی مجموعه اعتبارسنجی پایش می‌کنید تا از بیش‌برازش جلوگیری شود. تکنیک‌هایی مانند برش گرادیان، زمان‌بندی نرخ یادگیری، و توقف زودهنگام می‌توانند به پایداری و کارایی آموزش کمک کنند.

- ارزیابی و تست: پس از آموزش، عملکرد مدل را روی یک مجموعه تست جداگانه ارزیابی کنید تا دقت و تعمیم‌پذیری آن سنجیده شود. معیارهایی مانند نرخ خطای واژه (WER) یا نرخ خطای کاراکتر (CER) بینشی را فراهم می‌کنند که نشان می‌دهد مدل تا چه اندازه می‌تواند صوت را نسبت به رونویسی واقعی به درستی تبدیل کند.

#### یازده - چهار - چهار: مطالعه‌های موردی و کاربردها

۱. رونویسی پزشکی: آموزش دقیق LLM‌های گفتاری بر روی داده‌های پزشکی منجر به بهبودهای قابل توجهی در رونویسی تعاملات پزشک-بیمار شده است. مدل‌هایی مانند Whisper بر روی واژگان پزشکی آموزش دقیق شده‌اند و این امر منجر به رونویسی‌های دقیق‌تر و قابل اعتمادتر شده است.

۲. پردازش مستندات قانونی: شرکت‌های حقوقی از LLM‌های صوتی آموزش دقیق‌شده برای رونویسی جلسات دادگاه و بحث‌های قانونی استفاده کرده‌اند. آموزش دقیق مبتنی بر حوزه، توانایی مدل‌ها را در شناسایی و رونویسی دقیق اصطلاحات حقوقی بهبود بخشیده است.

۳. اتوماسیون خدمات مشتری: شرکت‌ها از مدل‌های گفتاری آموزش دقیق‌شده برای خودکارسازی تعاملات خدمات مشتری استفاده می‌کنند. این مدل‌ها بر روی داده‌های پشتیبانی مشتری آموزش دیده‌اند تا به‌طور مؤثرتری به سؤالات پاسخ دهند و تجربه کاربری بهتری ارائه دهند.

# فصل دوازدهم

## چالش‌های باز و جهت‌های تحقیق

### دوازده – یک: مسائل مقیاس‌پذیری

آموزش دقیق مدل‌های زبان بزرگ (LLMs) مانند GPT-4<sup>179</sup>، PaLM1<sup>180</sup> و T5 به یک حوزه تحقیقاتی حیاتی تبدیل شده است که چالش‌های قابل توجهی را به همراه دارد و فرصت‌های جدیدی برای اکتشاف فراهم می‌آورد، به ویژه در مقیاس‌بندی این فرآیندها به طور مؤثر. این بحث بر روی دو جنبه اصلی تمرکز دارد: چالش‌های مقیاس‌بندی فرآیندهای آموزش دقیق و جهت‌های تحقیقاتی بالقوه برای راه حل‌های مقیاس‌پذیر.

#### دوازده – یک – یک: چالش‌های مقیاس‌بندی فرآیندهای آموزش دقیق

۱. منابع محاسباتی: مدل‌های بزرگ‌مقیاس مانند GPT-3 و PaLM نیاز به منابع محاسباتی بسیار زیادی برای آموزش دقیق دارند. به عنوان مثال، آموزش دقیق یک مدل با ۱۷۵ میلیارد پارامتر مانند GPT-3 نیاز به GPU‌ها یا TPU‌های با عملکرد بالا دارد که قادر به پردازش حجم زیادی از داده‌ها و عملیات پیچیده باشند. حجم عظیم پارامترها به تقاضای محاسباتی گستره‌ای منجر می‌شود. حتی یک مدل نسبتاً کوچک‌تر، مانند BERT-large با ۳۴۰ میلیون پارامتر، می‌تواند در آموزش دقیق نیاز به محاسبات زیادی داشته باشد.

۲. نیازهای حافظه: حجم حافظه برای آموزش دقیق LLM‌ها بسیار زیاد است. هر پارامتر در مدل نیاز به فضای ذخیره‌سازی دارد و در طول آموزش، حافظه اضافی برای ذخیره محاسبات میانی، گرادیان‌ها و وضعیت‌های بهینه‌ساز مورد نیاز است. به عنوان مثال، بارگذاری یک مدل با ۷ میلیارد پارامتر (مانند LLaMA 2) در FP32 (۴ بایت برای هر پارامتر) حدود ۲۸ گیگابایت حافظه GPU نیاز دارد، در حالی که آموزش دقیق به حدود ۱۱۲ گیگابایت حافظه GPU نیاز دارد. این نیاز حافظه فراتر از ظرفیت بیشتر سخت‌افزارهای مصرف‌کننده است و دسترسی به آموزش دقیق را عمدتاً به سازمان‌ها یا مؤسسات تحقیقاتی با بودجه بالا محدود می‌کند.

<sup>179</sup> <https://ai.google/discover/palm/>

<sup>180</sup> [https://huggingface.co/docs/transformers/en/model\\_doc/t5](https://huggingface.co/docs/transformers/en/model_doc/t5)

**۳. حجم داده: LLMها معمولاً نیاز به حجم زیادی از داده‌های آموزشی دارند تا در طول آموزش دقیق عملکرد پیشرفته‌ای داشته باشند. این داده‌ها باید با سرعت بالا بارگذاری، پیش‌پردازش و به مدل وارد شوند تا آموزش مؤثری حفظ شود. مدیریت مجموعه داده‌های بزرگ می‌تواند یک گلوگاه شود، بهویژه اگر داده‌ها به طور توزیع شده در چندین سیستم ذخیره شوند یا اگر نیاز به دریافت از ذخیره‌سازی راه دور داشته باشند.**

**۴. بازدهی و نقاط تنگنا:** سرعت بالای پردازش برای استفاده کامل از GPUها یا TPUها ضروری است. با این حال، مسیرهای داده می‌توانند اگر به درستی بهینه‌سازی نشوند، گلوگاه شوند. به عنوان مثال، در هم‌آمیزی مجموعه داده‌های بزرگ یا بارگذاری آن‌ها در حافظه به اندازه کافی سریع برای همگام‌سازی با فرآیند آموزش ممکن است چالش‌برانگیز باشد. تکنیک‌هایی مانند بسته‌بندی داده‌ها، که در آن چند نمونه کوچک به دسته‌های بزرگ‌تر ترکیب می‌شوند، به بهبود سرعت پردازش کمک می‌کنند اما پیچیدگی‌هایی را به روتین‌های مدیریت داده اضافه می‌کنند.

**۵. استفاده مؤثر از منابع:** هزینه‌های مالی و زیست‌محیطی آموزش دقیق مدل‌های بزرگ قابل توجه است. آموزش دقیق بزرگ‌مقیاس فقط شامل هزینه مستقیم منابع محاسباتی نیست بلکه شامل هزینه‌های غیرمستقیم مربوط به مصرف انرژی و نگهداری زیرساخت نیز می‌شود. تکنیک‌هایی مانند آموزش با دقت مخلوط و چک‌پوینتینگ گرادیان می‌توانند این هزینه‌ها را با بهینه‌سازی حافظه و کارایی محاسباتی کاهش دهند.

چالش‌های مقیاس‌بندی فرآیندهای آموزش دقیق LLMها چندوجهی و پیچیده هستند و شامل محدودیت‌های قابل توجه محاسباتی، حافظه و مدیریت داده می‌شوند. نواوری‌ها در PEFT، بهینه‌سازی سرعت پردازش داده و روش‌های آموزش کارآمد منابع برای غلبه بر این چالش‌ها حیاتی هستند. با ادامه رشد LLMها در اندازه و قابلیت، رسیدگی به این چالش‌ها برای دسترسی و کارایی هوش مصنوعی پیشرفته برای دامنه وسیع‌تری از کاربردها ضروری خواهد بود.

## دوازده - یک - دو: جهت‌های تحقیق برای راه حل‌های مقیاس‌پذیر

### تکنیک‌های پیشرفته PEFT و آموزش دقیق پراکنده

پیشرفته‌های اخیر در تکنیک‌های PEFT، مانند LORA و واریانت آن، LORA کمی‌شده، مقیاس‌پذیری LLMها را دگرگون کرده است. LORA بار محاسباتی را با بهروزرسانی تنها یک تقریب با رتبه پایین از پارامترها کاهش می‌دهد و به طور قابل توجهی نیازهای حافظه و پردازش را کاهش می‌دهد. LORA کمی‌شده با اعمال کمی‌سازی به این ماتریس‌های با رتبه پایین، استفاده از منابع را بیشتر بهینه می‌کند و در عین حال عملکرد بالای مدل را حفظ می‌کند و نیاز به سخت‌افزار گسترده را به حداقل می‌رساند. این امر امکان آموزش دقیق کارآمد مدل‌های عظیم را فراهم کرده است، مانند پروژه LLaMA متأ، جایی که سازگاری با مجموعه کوچکی

از پارامترهای تأثیرگذار به مدل‌ها اجازه می‌دهد تا در وظایف مختلف با فشار محاسباتی کمتری به‌طور قابل اعتمادی عمل کنند.

تکنیک‌های آموزش دقیق پراکنده، مانند SpIEL، این تلاش‌ها را با بهروزرسانی انتخابی تنها پارامترهای مؤثر تکمیل می‌کنند. SpIEL با تغییر تنها یک بخش کوچک از پارامترها، که آن‌ها را با یک ایندکس پیگیری می‌کند، مدل‌ها را آموزش دقیق می‌کند. این فرآیند شامل بهروزرسانی پارامترها، حذف مهم‌ترین پارامترها و افزودن پارامترهای جدید بر اساس گردایان‌ها یا مومنتوم برآورده شده با استفاده از یک بهینه‌ساز کارآمد است.

### آموزش دقیق کارآمد داده<sup>۱۸۱</sup> (DEFT)

برای رسیدگی به چالش‌های مقیاس‌پذیری، اخیراً مفهوم DEFT ظهرور کرده است. این رویکرد نوین، هرس داده را به عنوان یک مکانیسم برای بهینه‌سازی فرآیند آموزش دقیق با تمرکز بر روی مهم‌ترین نمونه‌های داده معرفی می‌کند.

قصد دارد تا کارایی و اثربخشی آموزش دقیق LLM‌ها را با هرس انتخابی داده‌های آموزشی به منظور شناسایی نمونه‌های تأثیرگذار و نماینده افزایش دهد. این روش از اصول یادگیری با نمونه‌های محدود بهره می‌برد و به LLM‌ها این امکان را می‌دهد که با حداقل نمونه‌ها به داده‌های جدید سازگار شوند در حالی که عملکرد یا حتی سطوح بالاتری از عملکرد را که با مجموعه‌های کامل داده به دست آمده است، حفظ کنند.

### اجزای کلیدی DEFT

دقت بالا از طریق نمره تأثیر: DEFT مفهوم نمره تأثیر را معرفی می‌کند تا اهمیت هر نمونه داده را در زمینه آموزش دقیق LLM ارزیابی و رتبه‌بندی کند. نمره تأثیر تخمین می‌زند که حذف یک نمونه خاص چگونه بر عملکرد کلی مدل تأثیر می‌گذارد. این رویکرد اجازه می‌دهد تا یک زیرمجموعه کوچک از داده‌ها که بسیار نماینده و تأثیرگذار هستند، انتخاب شود و در نتیجه مدل قادر به حفظ دقت بالا با تعداد بسیار کمتری از نمونه‌ها باشد.

کارایی بالا از طریق نمره تلاش و مدل‌های جانشین: برای پرداختن به هزینه و پیچیدگی ارزیابی مجموعه داده‌های بزرگ، DEFT از یک مدل جانشین - یک مدل کوچکتر و با بار محاسباتی کمتر - برای تقریب‌زدن نمره‌های تأثیر استفاده می‌کند. این مدل جانشین به تخمین تأثیر هر نمونه بدون بار محاسباتی سنگین مربوط به استفاده مستقیم از LLM کمک می‌کند. علاوه بر این، DEFT نمره تلاشی را معرفی می‌کند تا نمونه‌های چالش‌برانگیزتر را شناسایی و اولویت‌بندی کند که ممکن است به توجه خاصی از LLM نیاز داشته باشند. این سیستم دو نمره‌ای اطمینان حاصل می‌کند که فرآیند آموزش دقیق هم کارآمد و هم مؤثر باقی بماند.

### پیامدهای عملی و موارد استفاده

- آموزش دقیق با نمونه‌های محدود برای سازگاری سریع: DEFT به‌ویژه برای کاربردهایی مفید است که در آن مدل‌ها نیاز به سازگاری سریع با داده‌های جدید با حداقل نمونه‌ها دارند. در سناریوهایی

<sup>۱۸۱</sup> Data Efficient Fine-Tuning

مانند پیشنهادات شخصی‌سازی شده یا سازگاری با تغییرات ناگهانی در رفتار کاربر، DEFT امکان آموزش دقیق سریع را فراهم می‌کند و عملکرد بالایی را با بخشی از داده‌های معمولی که نیاز است، حفظ می‌کند.

- کاهش هزینه‌های محاسباتی در پیاده‌سازی‌های بزرگ‌مقیاس: با تمرکز بر روی تأثیرگذارترین نمونه‌های داده و استفاده از مدل‌های جانشین، DEFT به طور قابل توجهی منابع محاسباتی مورد نیاز برای آموزش دقیق را کاهش می‌دهد. این امر امکان حفظ LLM‌های با عملکرد بالا را حتی در پیاده‌سازی‌های بزرگ‌مقیاس که حجم داده‌ها زیاد است، فراهم می‌کند.

### جهت‌های آینده

DEFT یک وظیفه هرس داده را برای آموزش دقیق مدل‌های زبان بزرگ (LLMs) معرفی می‌کند و زمینه را برای تحقیقات جدید در سیستم‌های توصیه‌گر مبتنی بر LLM و ارائه فرصت‌های متعددی برای اکتشافات آینده فراهم می‌آورد. زمینه‌های کلیدی برای تحقیق بیشتر شامل موارد زیر است:

- اعمال رویکرد پیشنهاد شده DEALRec به طیف وسیع‌تری از مدل‌های توصیه‌گر مبتنی بر LLM در داده‌های چند دامنه‌ای مختلف، به طوری که عملکرد آموزش دقیق در محدودیت‌های منابع تقویت شود.
- پرداختن به محدودیت پنجره زمینه LLM‌ها با تمرکز انتخابی بر روی تأثیرگذارترین اقلام در دنباله‌های تعامل کاربر برای اهداف آموزش دقیق.

## دوازده - یک - سه: طراحی همزمان سخت‌افزار و الگوریتم

طراحی همزمان سخت‌افزار و الگوریتم‌ها برای LLM‌ها می‌تواند بهبودهای قابل توجهی در کارایی فرآیندهای تنظیم دقیق ایجاد کند. شتاب‌دهنده‌های سخت‌افزاری سفارشی که برای کارهای خاص یا انواع محاسبات بهینه شده‌اند، می‌توانند نرژی و زمان مورد نیاز برای آموزش و تنظیم دقیق مدل را به طرز چشم‌گیری کاهش دهند.

- شتاب‌دهنده‌های سفارشی: توسعه شتاب‌دهنده‌های سخت‌افزاری به طور خاص برای محاسبات پرآکنده و با دقت پایین که معمولاً در تنظیم دقیق LLM‌ها استفاده می‌شود، می‌تواند عملکرد را بهبود بخشد. این شتاب‌دهنده‌ها به گونه‌ای طراحی شده‌اند که به طور مؤثر نیازهای منحصر به فرد LLM‌ها را، مانند پهنای باند بالای حافظه و ضربه‌ای ماتریسی وسیع که در معماری‌های ترنسفورمر وجود دارد، مدیریت کنند.

- بهینه‌سازی الگوریتمی: ترکیب نوآوری‌های سخت‌افزاری با تکنیک‌های بهینه‌سازی الگوریتمی، مانند تکنیک‌هایی که حرکت داده‌ها را حداقل می‌کنند یا از ویژگی‌های خاص سخت‌افزاری (مانند هسته‌های تنسور برای محاسبات با دقت مختلط<sup>۱۸۲</sup>) استفاده می‌کنند، می‌تواند کارایی فرآیندهای تنظیم دقیق را بیشتر افزایش دهد.

- مثال: NVIDIA از TensorRT3 از نمونه‌ای از طراحی همزمان سخت‌افزار و الگوریتم است که در عمل به کار می‌رود. این ابزار مدل‌های یادگیری عمیق را برای استنتاج بهینه‌سازی می‌کند و از قابلیت‌های GPU‌های

<sup>182</sup> Tensor Cores for Mixed-Precision Calculations

NVIDIA بهره می‌برد و در نتیجه فرآیند را به طور قابل توجیهی تسریع می‌کند و نیازهای منابع را کاهش می‌دهد. بهینه‌سازی‌های TensorRT<sup>183</sup> شامل پشتیبانی از عملیات‌های تنسور پراکنده و با دقت مختلف است که آن را برای تنظیم دقیق مدل‌های بزرگ بسیار مناسب می‌سازد.

با ادامه رشد مقیاس مدل‌های زبانی، پرداختن به چالش‌های تنظیم دقیق مؤثر آنها به طور فزاینده‌ای حائز اهمیت می‌شود. نوآوری‌ها در PEFT، تنظیم دقیق های پراکنده، مدیریت داده و یکپارچه‌سازی راه حل‌های پیشرفته سخت‌افزاری و الگوریتمی مسیرهای امیدبخشی برای تحقیقات آینده ارائه می‌دهند. این راه حل‌های مقیاس‌پذیر نه تنها برای قابل انجام کردن استقرار LLM‌ها در دامنه وسیع تری از کاربردها ضروری هستند، بلکه مرزهای آنچه که این مدل‌ها می‌توانند به آن دست یابند را نیز گسترش می‌دهند.

## دوازده – دو: ملاحظات اخلاقی در تنظیم دقیق LLM‌ها

### دوازده – دو – یک: سوگیری و انصاف

هنگام تنظیم دقیق LLM‌ها، هدف معمولاً بهینه‌سازی عملکرد آنها برای کارها یا مجموعه‌های داده خاص است. با این حال، این مجموعه‌های داده ممکن است به طور ذاتی سوگیری‌هایی داشته باشند که در طول فرآیند تنظیم دقیق به مدل منتقل می‌شوند. سوگیری‌ها می‌توانند از منابع مختلفی ناشی شوند، از جمله داده‌های تاریخی، نمونه‌های آموزشی نامتعادل و تعصبات فرهنگی نهفته در زبان. برای مثال، یک LLM که بر روی مجموعه داده‌ای که عمدتاً از کشورهای انگلیسی‌زبان به دست آمده است، دقیق تنظیم شده است، ممکن است زمانی که به متون از پس‌زمینه‌های زبانی یا فرهنگی دیگر اعمال می‌شود، عملکرد کمتری داشته باشد یا پیش‌بینی‌های سوگیری‌شده‌ای ارائه دهد.

ابزار Google AI Fairness Indicators<sup>184</sup> یک راه حل عملی است که به توسعه‌دهندگان اجازه می‌دهد انصاف مدل‌های خود را با تحلیل معیارهای عملکرد در گروه‌های دموگرافیک مختلف ارزیابی کنند. این ابزار می‌تواند در فرآیند تنظیم دقیق ادغام شود تا سوگیری را به طور بلاذرنگ زیر نظر گیرد و برطرف کند.

### پرداختن به سوگیری و انصاف

- **داده‌های متنوع و قابل شمول:** اطمینان از اینکه مجموعه‌های داده تنظیم دقیق متنوع و نماینده همه گروه‌های دموگرافیک کاربران هستند، می‌تواند به کاهش سوگیری کمک کند.

- **حدودیت‌های انصاف:** گنجاندن محدودیت‌های انصاف، همان‌طور که در فریمورک FairBERTa<sup>185</sup> پیشنهاد شده، اطمینان حاصل می‌کند که مدل‌های دقیق تنظیم شده عملکرد عادلانه‌ای در گروه‌های مختلف حفظ کنند.

<sup>183</sup> <https://docs.nvidia.com/tensorrt/index.html>

<sup>184</sup> <https://research.google/blog/fairness-indicators-scalable-infrastructure-for-fair-ml-systems/>

<sup>185</sup> <https://huggingface.co/facebook/FairBERTa>

- **مثال کاربرد:** در حوزه بهداشت و درمان، یک LLM که برای کمک در تشخیص بیماری‌ها دقیق تنظیم شده، ممکن است در ابتدا بر روی داده‌های مربوط به بیماران عمدتاً سفیدپوست آموزش ببیند. چنین مدلی می‌تواند تشخیص‌های کمتری برای بیماران از نژادهای دیگر ارائه دهد. با استفاده از تکنیک‌های تنظیم دقیق با آگاهی از انصاف، ارائه‌دهندگان خدمات بهداشتی می‌توانند مدل‌هایی توسعه دهند که در بین جمعیت‌های بیمار متنوع عملکرد عادلانه‌تری داشته باشند.

#### دوازده - دو - نگرانی‌های حریم خصوصی

تنظیم دقیق معمولاً شامل استفاده از مجموعه‌های داده حساس یا اختصاصی است که خطرات حریم خصوصی قابل توجهی را به همراه دارد. اگر به درستی مدیریت نشود، مدل‌های دقیق تنظیم شده ممکن است به طور ناخواسته اطلاعات خصوصی را از داده‌های آموزشی خود نش特 دهند. این موضوع به ویژه در حوزه‌هایی مانند بهداشت و درمان یا مالی که محramانگی داده‌ها بسیار حیاتی است، بسیار مهم است.

#### اطمینان از حریم خصوصی در حین تنظیم دقیق

- **حریم خصوصی تفاضلی<sup>۱۸۶</sup>:** اجرای تکنیک‌های حریم خصوصی تفاضلی در حین تنظیم دقیق می‌تواند از نشت اطلاعات حساس توسط مدل‌ها جلوگیری کند.

- **یادگیری غیر متتمرکز<sup>۱۸۷</sup>:** استفاده از چارچوب‌های یادگیری غیر متتمرکز به مدل‌ها اجازه می‌دهد که در داده‌های غیرمتتمرکز دقیق تنظیم شوند، که حریم خصوصی را با حفظ محلی سازی داده‌ها تقویت می‌کند.

- **مثال کاربرد:** در برنامه‌های خدمات مشتری، شرکت‌ها ممکن است LLM‌ها را با استفاده از داده‌های تعاملات مشتری دقیق تنظیم کنند. استفاده از حریم خصوصی تفاضلی اطمینان حاصل می‌کند که مدل از این تعاملات یاد می‌گیرد بدون اینکه اطلاعات شخصی را حفظ کند و به طور بالقوه نشست دهد، و بدین ترتیب محramانگی مشتری را حفظ می‌کند.

#### دوازده - دو - سه: خطرات امنیتی

- **آسیب‌پذیری‌های امنیتی در مدل‌های تنظیم شده:** مدل‌های LLM که به طور خاص تنظیم شده‌اند، در برابر آسیب‌پذیری‌های امنیتی، به ویژه از حملات خصم‌مانه، آسیب‌پذیر هستند. این حملات شامل ورودی‌هایی هستند که به منظور بهره‌برداری از نقاط ضعف مدل طراحی شده‌اند و باعث می‌شوند مدل خروجی‌هایی اشتباه یا مضر تولید کند. چنین آسیب‌پذیری‌هایی ممکن است در مدل‌های تنظیم شده به دلیل داده‌های آموزشی خاص آن‌ها که ممکن است تمامی سناریوهای ورودی ممکن را پوشش ندهد، بیشتر مشهود باشند.

<sup>186</sup> <https://privacytools.seas.harvard.edu/differential-privacy>

<sup>187</sup> <https://research.ibm.com/blog/what-is-federated-learning>

- تحقیقات و شیوه‌های اخیر: ماتریس تهدید ML خصمانه مایکروسافت یک چارچوب جامع برای شناسایی و کاهش تهدیدات خصمانه در طول توسعه و تنظیم مدل ارائه می‌دهد. این ماتریس به توسعه‌دهندگان کمک می‌کند تا از احتمال بروز حملات آگاه شوند و استراتژی‌های دفاعی مناسبی را پیاده‌سازی کنند.

- تقویت امنیت در تنظیم دقیق:

- آموزش خصمانه: قرار دادن مدل‌ها در معرض نمونه‌های خصمانه در طول تنظیم می‌تواند تابآوری آن‌ها را در برابر حملات افزایش دهد.

- بازرسی‌های امنیتی: انجام منظم بازرسی‌های امنیتی بر روی مدل‌های تنظیم شده می‌تواند به شناسایی و رفع آسیب‌پذیری‌های احتمالی کمک کند.

## دوازده – سه: پاسخگویی و شفافیت

### دوازده – سه – یک: نیاز به پاسخگویی و شفافیت

تنظیم دقیق مدل می‌تواند به طور قابل توجهی رفتار یک LLM را تغییر دهد، بنابراین مستندسازی و درک تغییرات و تأثیرات آن‌ها ضروری است. این شفافیت برای ذینفعان حیاتی است تا به خروجی‌های مدل اعتماد کنند و برای توسعه‌دهندگان اهمیت دارد تا در قبال عملکرد و پیامدهای اخلاقی آن پاسخگو باشند.

### دوازده – سه – دو: تحقیقات و شیوه‌های اخیر

چارچوب هوش مصنوعی مسئولانه متا<sup>۱۸۸</sup> بر اهمیت مستندسازی فرآیند تنظیم و تأثیرات آن بر رفتار مدل تأکید می‌کند. این شامل نگهداری سوابق دقیق از داده‌های مورد استفاده، تغییرات ایجاد شده در طول تنظیم و معیارهای ارزیابی به کار رفته است.

### دوازده – سه – سه: ترویج پاسخگویی و شفافیت

- مستندسازی جامع: ایجاد مستندات دقیق از فرآیند تنظیم و تأثیر آن بر عملکرد و رفتار مدل.

- گزارش‌دهی شفاف: استفاده از چارچوب‌هایی مانند کارت‌های مدل<sup>۱۸۹</sup> برای گزارش‌دهی در مورد ویژگی‌های اخلاقی و عملی مدل‌های تنظیم شده.

- نمونه کاربرد: در سیستم‌های مدیریت محتوا، مدل‌های LLM که برای شناسایی و فیلتر کردن محتوای مضر تنظیم شده‌اند، نیاز به مستندسازی و گزارش‌دهی واضح دارند. این اطمینان می‌دهد که کاربران پلتفرم و مقامات ناظر می‌فهمند که مدل چگونه کار می‌کند و می‌توانند به تصمیمات مدیریت محتوای آن اعتماد کنند.

<sup>188</sup> <https://huggingface.co/docs/hub/en/model-cards>

<sup>189</sup> <https://huggingface.co/docs/hub/en/model-cards>

## دوازه‌های چهار چوبها/ تکنیک‌های پیشنهادی برای تنظیم دقیق اخلاقی

### چارچوب‌های کاهش تعصب

چارچوب‌های تنظیم آگاه به تعصب هدف دارند تا انصاف را در فرآیند آموزش مدل گنجانده و اجرا کنند. FairBERTa، که توسط فیسبوک معرفی شده، یک نمونه از چنین چارچوبی است که محدودیت‌های انصاف را به طور مستقیم در تابع هدف مدل در طول تنظیم ادغام می‌کند. این رویکرد اطمینان حاصل می‌کند که عملکرد مدل به طور متوازن در میان گروه‌های جمعیتی مختلف است.

سازمان‌ها می‌توانند از چارچوب‌های آگاه به انصاف برای توسعه سیستم‌های هوش مصنوعی منصفانه‌تر استفاده کنند. به عنوان مثال، پلتفرم‌های رسانه‌های اجتماعی می‌توانند از این چارچوب‌ها برای تنظیم مدل‌هایی که سخنان نفرت‌انگیز را شناسایی و کاهش می‌دهند استفاده کنند، در حالی که از رفتار منصفانه در بین جمعیت‌های مختلف کاربران اطمینان حاصل می‌کنند.

### تکنیک‌های حفظ حریم خصوصی

حریم خصوصی به عنوان یک اصل کلیدی در فرآیند تنظیم دقیق مدل‌ها، با استفاده از تکنیک‌هایی مانند حریم خصوصی تفاضلی و یادگیری غیر متمرکز حفظ می‌شود.<sup>190</sup> TensorFlow Privacy که توسط گوگل توسعه یافته است، پشتیبانی داخلی از حریم خصوصی تفاضلی را فراهم می‌کند و به توسعه‌دهندگان این امکان را می‌دهد که مدل‌ها را به صورت ایمن تنظیم کنند بدون اینکه حریم خصوصی داده‌ها به خطر بیفتد.

مدل‌های LLM بسیار مؤثر هستند، اما در هنگام استفاده در زمینه‌های حساس که حفظ حریم خصوصی داده‌ها حیاتی است، با چالش‌هایی روبرو می‌شوند. برای مقابله با این موضوع، پژوهشگران بر بهبود مدل‌های زبان کوچک (SLMs) که به حوزه‌های خاصی اختصاص یافته‌اند، تمرکز می‌کنند. روش‌های موجود معمولاً از LLM‌ها برای تولید داده‌های اضافی یا انتقال دانش به SLM‌ها استفاده می‌کنند، اما این رویکردها به دلیل تفاوت‌های بین داده‌های تولید شده توسط LLM و داده‌های خصوصی مشتریان با مشکلاتی مواجه هستند. در پاسخ به این چالش، چارچوب جدیدی به نام انتقال دانش خاص فدرال (FDKT) معرفی شده است. FDKT با بهره‌گیری از LLM‌ها، نمونه‌های مصنوعی ایجاد می‌کند که توزیع داده‌های خصوصی مشتریان را با استفاده از حریم خصوصی تفاضلی شبیه‌سازی می‌کند. این رویکرد به طور قابل توجهی عملکرد SLM‌ها را حدود ۵ درصد بهبود می‌بخشد و در عین حال حریم خصوصی داده‌ها را با یک بودجه حریم خصوصی حداقلی حفظ می‌کند و از روش‌های سنتی که تنها به داده‌های خصوصی محلی متکی هستند، پیشی می‌گیرد.

در حوزه بهداشت و درمان، تنظیم دقیق غیر متمرکز می‌تواند به بیمارستان‌ها اجازه دهد که به طور مشترک مدل‌ها را بر روی داده‌های بیماران آموزش دهند بدون اینکه اطلاعات حساس را منتقل کنند. این رویکرد حریم خصوصی داده‌ها را حفظ کرده و در عین حال امکان توسعه سیستم‌های هوش مصنوعی مقاوم و قابل تعمیم را فراهم می‌کند.

<sup>190</sup> [https://www.tensorflow.org/responsible\\_ai/privacy/guide](https://www.tensorflow.org/responsible_ai/privacy/guide)

## چارچوب‌هایی برای تقویت امنیت

آموزش خصمانه و اقدامات امنیتی robust برای حفاظت از مدل‌های دقیق تنظیم شده در برابر حملات ضروری هستند. رویکرد آموزش خصمانه شامل آموزش مدل‌ها با استفاده از نمونه‌های خصمانه به منظور بهبود تاب‌آوری آن‌ها در برابر ورودی‌های مخرب است. ابزارهای آموزش خصمانه مایکروسافت آژور را حل‌های عملی برای ادغام این تکنیک‌ها در فرآیند تنظیم ارائه می‌دهند و به توسعه دهنده‌گان کمک می‌کنند تا مدل‌های ایمن و قابل اعتمادی ایجاد کنند.

در زمینه امنیت سایبری، LLM‌های تنظیم شده که برای شناسایی تهدیدات استفاده می‌شوند، می‌توانند از آموزش خصمانه بهره‌مند شوند تا توانایی خود را در شناسایی و پاسخ به حملات پیچیده بهبود بخشنند و بدین ترتیب امنیت سازمانی را افزایش دهند.

## چارچوب‌هایی برای تضمین شفافیت

چارچوب‌های شفافیت و پاسخگویی، مانند کارت‌های مدل و FactSheets<sup>191</sup> هوش مصنوعی، روش‌های ساختاری برای مستندسازی و گزارش‌دهی در مورد فرآیند تنظیم و رفتارهای مدل‌های بهدست‌آمده فراهم می‌کنند. این چارچوب‌ها با تعریف واضح قابلیت‌ها، محدودیت‌ها و ملاحظات اخلاقی مدل، درک و اعتماد را بین ذینفعان ترویج می‌کنند.

در کاربردهای دولتی، جایی که سیستم‌های هوش مصنوعی ممکن است برای تصمیم‌گیری یا خدمات عمومی استفاده شوند، حفظ مستندسازی شفاف از طریق چارچوب‌هایی مانند FactSheets هوش مصنوعی اطمینان می‌دهد که این سیستم‌ها پاسخگو هستند و تصمیمات آن‌ها می‌تواند مورد بررسی و اعتماد عموم قرار گیرد. تنظیم LLM‌ها چالش‌های اخلاقی متعددی را معرفی می‌کند، از جمله تعصب، خطرات حریم خصوصی، آسیب‌پذیری‌های امنیتی و نگرانی‌های مربوط به پاسخگویی. برای مقابله با این چالش‌ها، نیاز به یک رویکرد چندبعدی وجود دارد که شامل چارچوب‌های آگاه به عدالت، تکنیک‌های حفظ حریم خصوصی، اقدامات امنیتی قوی و مکانیسم‌های شفافیت و پاسخگویی باشد. با بهره‌گیری از پیشرفت‌های اخیر در این زمینه‌ها، پژوهشگران و کارشناسان می‌توانند LLM‌های توسعه و پیاده‌سازی کنند که نه تنها قدرتمند بلکه اخلاقی و قابل اعتماد نیز باشند.

## دوازده – چهار: یکپارچگی با فناوری‌های نوظهور

یکپارچه‌سازی LLM‌ها با فناوری‌های نوظهور مانند اینترنت اشیا (IoT) و محاسبات لبه‌ای<sup>192</sup> فرصت‌ها و چالش‌های متعددی را به همراه دارد و پیشرفت‌ها و بینش‌های حاصل از تحقیقات و توسعه‌های صنعتی اخیر را منعکس می‌کند.

<sup>191</sup> <https://aifs360.res.ibm.com/>

<sup>192</sup> Edge Computing

## دوازده - چهار - یک: فرصت‌ها

- تصمیم‌گیری و خودکارسازی بهبود یافته: LLM‌ها قابلیت تحلیل و استخراج بینش‌ها از حجم بالای داده‌های غیرساختاری که توسط دستگاه‌های IoT تولید می‌شوند، دارند. این داده‌ها می‌توانند از قرائت‌های حسگر در کارخانه‌های تولیدی تا داده‌های زیستمحیطی در شهرهای هوشمند متغیر باشند. با پردازش این داده‌ها به صورت بلادرنگ، LLM‌ها می‌توانند فرآیندهای تصمیم‌گیری را بهینه‌سازی کرده و وظایفی را که به‌طور سنتی نیاز به مداخله انسانی دارند، خودکار کنند. به عنوان مثال:
  - **کاربردهای صنعتی:** نگهداری قابل پیش‌بینی می‌تواند با تحلیل داده‌های حسگر توسط LLM‌ها بهبود یابد تا پیش‌بینی خرابی تجهیزات قبل از وقوع آن‌ها، بدین ترتیب زمان خرابی و هزینه‌های نگهداری کاهش یابد.
  - **شهرهای هوشمند:** LLM‌ها می‌توانند الگوهای ترافیکی و داده‌های زیستمحیطی را از حسگرهای IoT تحلیل کنند تا زیرساخت‌های شهری را بهینه‌سازی کرده و تصمیمات برنامه‌ریزی شهری را بهبود بخشنند.
  - **تجربیات شخصی‌سازی شده برای کاربران:** یکپارچه‌سازی با محاسبات لبه‌ای به LLM‌ها این امکان را می‌دهد که داده‌ها را به صورت محلی روی دستگاه‌ها پردازش کنند و به جای تکیه بر سرورهای ابری، به طور مستقیم از داده‌های محلی بهره‌برداری کنند. این به LLM‌ها اجازه می‌دهد خدمات بسیار شخصی‌سازی شده‌ای را بر اساس داده‌های بلادرنگ و ترجیحات کاربران ارائه دهند و تجربیات کاربری را در حوزه‌های مختلف بهبود بخشنند:
  - **بهداشت و درمان:** LLM‌ها می‌توانند با تحلیل داده‌های مربوط به دستگاه‌های پوشیدنی و ادغام آن‌ها با سوابق پزشکی که به‌طور ایمن بر روی دستگاه‌های لبه‌ای ذخیره می‌شوند، توصیه‌های بهداشتی شخصی‌سازی شده ارائه دهند.
  - **بهبود درک زبان طبیعی:** ادغام داده‌های IoT توانایی LLM‌ها را در درک زمینه و پاسخ‌دهی هوشمندانه‌تر به پرسش‌های زبان طبیعی غنی‌تر می‌کند. این می‌تواند به طور قابل توجهی تعاملات کاربران با محیط‌های هوشمند را بهبود بخشد.
  - **خانه‌های هوشمند:** LLM‌ها که با دستگاه‌های IoT یکپارچه شده‌اند، می‌توانند دستورات صوتی را با دقیق‌تری درک کرده و به آن‌ها پاسخ دهند و تنظیمات خانه‌های هوشمند را بر اساس داده‌های حسگر در زمان واقعی (مانند تنظیم نور و دما بر اساس اشغال و شرایط محیطی) تنظیم کنند.

## دوازده - چهار - دو: چالش‌ها

- پیچیدگی داده و ادغام: ادغام داده‌ها از دستگاه‌های مختلف IoT چالش‌هایی را در رابطه با کیفیت داده، قابلیت همکاری و مقیاس‌پذیری به همراه دارد. LLM‌ها باید این داده‌های ناهمگن را به طور مؤثر پردازش و تفسیر کنند تا بینش‌های معناداری استخراج کنند:
- ادغام داده: اطمینان از ادغام بی‌وقفه جریان‌های داده از پلتفرم‌ها و دستگاه‌های مختلف IoT بدون به خطر انداختن یکپارچگی یا عملکرد داده.
- پیش‌پردازش داده: پاکسازی و پیش‌پردازش داده‌های IoT برای اطمینان از سازگاری و قابلیت اعتماد قبل از تغذیه آن‌ها به LLM‌ها برای تحلیل.
- حریم خصوصی و امنیت: محاسبات لبه‌ای شامل پردازش داده‌های حساس به صورت محلی بر روی دستگاه‌هاست و نگرانی‌هایی را در مورد حریم خصوصی و امنیت داده‌ها ایجاد می‌کند:
- حریم خصوصی داده: پیاده‌سازی تکنیک‌های رمزگاری قوی و مکانیسم‌های کنترل دسترسی برای حفاظت از داده‌های حساس پردازش شده توسط LLM‌ها بر روی دستگاه‌های لبه‌ای.
- ارتباطات امن: اطمینان از وجود کانال‌های ارتباطی امن بین دستگاه‌های IoT و LLM‌ها برای جلوگیری از نقض داده‌ها یا دسترسی غیرمجاز.
- پردازش در زمان واقعی و قابلیت اطمینان: LLM‌های مستقر در محیط‌های محاسبات لبه‌ای باید با تأخیر کم و قابلیت اطمینان بالا عمل کنند تا از برنامه‌های زمان واقعی پشتیبانی کنند:
- تأخیر: بهینه‌سازی الگوریتم‌ها و قابلیت‌های پردازشی LLM‌ها برای مدیریت کارآمد جریان‌های داده در زمان واقعی بدون تأخیر.
- قابلیت اطمینان: اطمینان از دقیق و ثبات بینش‌های تولیدشده توسط LLM‌ها در محیط‌های دینامیک و غیرقابل پیش‌بینی IoT.

## دوازده - پنج: حوزه‌های تحقیقاتی آینده

- یادگیری توزیع شده و محاسبات لبه: بررسی تکنیک‌های یادگیری فدرال که در آن LLM‌ها می‌توانند به طور مشترک در دستگاه‌های اج<sup>۱۹۳</sup> آموزش بینند بدون جمع‌آوری داده‌های متمرکز. این رویکرد به مسائل حریم خصوصی پاسخ می‌دهد و با ارتباطی را کاهش می‌دهد.
- سیستم‌های پشتیبانی تصمیم‌گیری در زمان واقعی: توسعه سیستم‌های مبتنی بر LLM که قادر به تصمیم‌گیری در زمان واقعی از طریق ادغام با زیرساخت محاسبات لبه‌ای هستند. این شامل بهینه‌سازی الگوریتم‌ها برای پردازش با تأخیر کم و اطمینان از قابلیت اطمینان تحت شرایط محیطی دینامیک است.

<sup>193</sup> edge devices

- پیامدهای اخلاقی و مقرراتی: بررسی پیامدهای اخلاقی ادغام LLMها با IoT و محاسبات لبه‌ای، بهویژه در مورد مالکیت داده، شفافیت و انصاف. این حوزه نیازمند چارچوب‌هایی برای پیاده‌سازی و حاکمیت هوش مصنوعی اخلاقی است.

## واژه‌نامه

**LLM** (مدل زبانی بزرگ): نوعی مدل هوش مصنوعی که معمولاً دارای میلیارد‌ها پارامتر است و بر روی مقادیر زیادی از داده‌های متنی آموزش دیده تا بتواند متن‌هایی شبیه به انسان را درک و تولید کند. این مدل‌ها عمدها برای وظایف پردازش زبان طبیعی (NLP) طراحی شده‌اند.

**NLP** (پردازش زبان طبیعی): شاخه‌ای از هوش مصنوعی که بر تعامل بین کامپیوترها و انسان‌ها از طریق زبان طبیعی تمرکز دارد و شامل وظایفی مانند تولید زبان، ترجمه و تحلیل احساسات می‌شود.

**LoRA** (تنظیم تطبیق پایین رتبه): تکنیک تنظیم دقیق کارآمد از نظر پارامتر که تنها ماتریس‌های کوچک و پایین رتبه را برای انطباق مدل‌های از پیش آموزش دیده با وظایف خاص تنظیم می‌کند و به این ترتیب بیشتر پارامترهای مدل اصلی را حفظ می‌کند.

**DoRA** (تنظیم تطبیق پایین رتبه با وزن تفکیک شده): تکنیکی که وزن‌های مدل را به اجزای اندازه و جهت تقسیم می‌کند، که تسهیل‌کننده تنظیم دقیق در عین حفظ کارایی استنباط می‌باشد.

**QLoRA** (تنظیم تطبیق پایین رتبه کمی شده): نسخه‌ای از LoRA که به‌طور خاص برای مدل‌های کمی شده طراحی شده و اجازه می‌دهد تا تنظیم دقیق به‌طور کارآمد در محیط‌های محدود از نظر منابع انجام شود.

**PPO** (بهینه‌سازی سیاست مجاور): الگوریتم یادگیری تقویتی که سیاست‌ها را با تعادل بین کاوش اقدامات جدید و بهره‌برداری از پاداش‌های شناخته‌شده تنظیم می‌کند و برای ثبات و کارایی در آموزش طراحی شده است.

**DPO** (بهینه‌سازی ترجیحات مستقیم): روشی که به‌طور مستقیم مدل‌های زبانی را با ترجیحات انسانی هم‌راستا می‌کند از طریق بهینه‌سازی ترجیحات و دور زدن مدل‌های یادگیری تقویتی مانند PPO.

**MoE** (ترکیب کارشناسان): معماری مدلی که از چندین زیرشبکه تخصصی، به نام کارشناسان، استفاده می‌کند که به‌طور انتخابی بر اساس ورودی فعال می‌شوند تا عملکرد و کارایی مدل را بهبود بخشدند.

**MoA (ترکیب عوامل)**: چارچوب چند عاملهای که در آن چندین عامل در طول آموزش و استنباط همکاری می‌کنند و از نقاط قوت هر عامل برای بهبود عملکرد کلی مدل استفاده می‌شود.

**PEFT (تنظیم دقیق کارآمد از نظر پارامتر)**: رویکردی برای تنظیم دقیق مدل‌های بزرگ که تنها شامل تنظیم یک زیرمجموعه از پارامترهای مدل است و کارایی را در سناریوهای با منابع محاسباتی محدود بهبود می‌بخشد. این شامل تکنیک‌هایی مانند QLoRA، LoRA و آداتورها می‌شود.

**آداتورها**: ماثولهای کوچک و قابل آموزش که به لایه‌های مدل‌های زبانی از پیش آموزش دیده اضافه می‌شوند و اجازه می‌دهند تنظیم دقیق خاص وظیفه به طور کارآمدی بدون تغییر پارامترهای اصلی مدل انجام شود. تکنیک‌هایی مانند AdapterSoup و AdapterFusion در این دسته قرار می‌گیرند و ترکیب چندین آداتور را برای چند وظیفه‌ای‌های پیچیده تسهیل می‌کنند.

**تنظیم نرم‌افزار (SPT)**: تکنیک تنظیم دقیق که در آن مجموعه‌ای از توکن‌های قابل آموزش به دنباله ورودی اضافه می‌شود تا مدل از پیش آموزش دیده را به سمت عملکرد خاص وظیفه هدایت کند بدون اینکه وزن‌های داخلی مدل تغییر کنند.

**تنظیم پیشوند**: نسخه‌ای از تنظیم نرم‌افزار که در آن دنباله‌ای ثابت از بردارهای قابل آموزش به لایه ورودی در هر لایه از مدل اضافه می‌شود و سازگاری خاص وظیفه را افزایش می‌دهد.

**کوانتسازی**: فرآیند کاهش دقت وزن‌ها و فعالیت‌های مدل، اغلب از ۳۲ بیت به نمایش‌های با دقت پایین‌تر مانند ۸ بیت یا ۴ بیت، به منظور کاهش مصرف حافظه و بهبود کارایی محاسباتی.

**مدل‌های زبانی کمی‌شده (Quantised LLMs)**: مدل‌های زبانی بزرگ که تحت فرآیند کوانتسازی قرار گرفته‌اند، فرآیندی که دقت وزن‌ها و فعالیت‌های مدل را کاهش می‌دهد، معمولاً از ۳۲ بیت به ۸ بیت یا پایین‌تر، به منظور افزایش کارایی حافظه و محاسبات.

**حذفی (Pruning)**: تکنیک بهینه‌سازی مدل که پیچیدگی مدل‌های زبانی بزرگ را با حذف پارامترهای کم‌اهمیت کاهش می‌دهد و به این ترتیب استنباط سریع‌تر و مصرف حافظه کمتری را امکان‌پذیر می‌کند.

**تنظیم دقیق نیمه (Half Fine-Tuning - HFT)**: روشی برای تنظیم دقیق که در آن نیمی از پارامترهای مدل ثابت نگهداشته می‌شود و نیمی دیگر به روز می‌شود، که به حفظ دانش از پیش آموزش دیده کمک می‌کند در حالی که مدل را برای وظایف جدید تطبیق می‌دهد.

**ماسک‌گذاری ساختاری (Structured Masking)**: تکنیکی که کل لایه‌ها، سرها یا اجزای ساختاری دیگر یک مدل را ماسک می‌کند تا پیچیدگی را کاهش دهد در حین تنظیم دقیق برای وظایف خاص.

**ماسک‌گذاری غیرساختاری (Unstructured Masking)**: تکنیکی که در آن برخی پارامترهای مدل به طور تصادفی یا بر اساس یک الگو در حین تنظیم دقیق ماسک می‌شوند و این اجازه را می‌دهد تا مهم‌ترین وزن‌های مدل شناسایی شوند.

**GLUE (ارزیابی درک زبان عمومی)**: معیاری که برای ارزیابی عملکرد مدل‌های NLP در مجموعه‌ای از وظایف درک زبان، مانند تحلیل احساسات و استنباط زبان طبیعی استفاده می‌شود.

**SuperGLUE (ارزیابی درک زبان عمومی فوق العاده)**: نسخه‌ای چالش‌برانگیزتر از GLUE که شامل وظایف سخت‌تر است که برای آزمایش robustness و قابلیت انطباق مدل‌های NLP طراحی شده‌اند.

**TruthfulQA**: معیاری که برای اندازه‌گیری صحت خروجی یک مدل زبانی طراحی شده و بر دقت واقعی و مقاومت در برابر توهمنات تمرکز دارد.

**IFEval (ارزیابی پیروی از دستورالعمل)**: معیاری که توانایی یک مدل در پیروی از دستورالعمل‌های صریح در سراسر وظایف را ارزیابی می‌کند، معمولاً در زمینه تنظیم دقیق مدل‌های بزرگ برای پیروی از دستورالعمل‌های خاص.

**BBH (BigBench Hard)**: زیرمجموعه‌ای از مجموعه داده Big Bench که شامل وظایف به خصوص دشواری است که به ارزیابی توانایی‌های استدلال پیشرفته مدل‌های زبانی بزرگ می‌پردازد.

**MATH**: مجموعه داده‌ای که برای ارزیابی توانایی یک مدل در حل مسائل ریاضی سطح دبیرستان ایجاد شده است و در فرمتهای رسمی مانند LaTeX ارائه می‌شود.

**GPQA (پاسخ‌دهی به سوالات عمومی)**: مجموعه داده‌ای چالش‌برانگیز که شامل سوالات مبتنی بر دانش طراحی شده توسط کارشناسان برای ارزیابی استدلال عمیق و به خاطر سپردن واقعی است.

**MuSR** (استدلال ساختاری چندرسانه‌ای): مجموعه داده‌ای که شامل مسائل پیچیده‌ای است که نیاز به یکپارچه‌سازی استدلال در بین مдалیته‌ها دارد و معمولاً متن را با دیگر اشکال داده مانند تصاویر یا نمودارها ترکیب می‌کند.

**MMLU** (درک زبان چندوظیفه‌ای بزرگ): معیاری که توانایی یک مدل زبانی در انجام وظایف مختلف در دامنه‌های متنوع، مانند علوم انسانی، STEM، علوم اجتماعی و دیگران را ارزیابی می‌کند و معمولاً نیاز به استدلال سطح بالا دارد.

**MMLU-PRO**: نسخه‌ای اصلاح شده از مجموعه داده MMLU با تمرکز بر مسائل چند گزینه‌ای چالش‌برانگیزتر که معمولاً نیاز به تجزیه و تحلیل زمینه‌های دور دارند.

**ARC** (چالش استدلال AI2): معیاری برای ارزیابی توانایی‌های استدلال یک مدل زبانی با استفاده از مجموعه داده‌ای از سوالات علمی چند گزینه‌ای.

**COQA** (پاسخ‌دهی به سوالات مکالمه‌ای): معیاری که ارزیابی می‌کند یک مدل زبانی چقدر می‌تواند درک کند و در مکالمه‌های دوطرفه به‌ویژه در قالب سوال و جواب شرکت کند.

**DROP** (استدلال گستته بر روی پاراگراف‌ها): معیاری که توانایی یک مدل در انجام استدلال گستته بر روی متن را آزمایش می‌کند، به‌ویژه در سناریوهایی که نیاز به استدلال‌های ریاضی، مقایسه یا منطقی دارد.

**SQuAD** (مجموعه داده پاسخ‌دهی به سوالات استنفورد): مجموعه داده‌ای محبوب برای ارزیابی توانایی یک مدل در درک و پاسخ به سوالات مبتنی بر مقاطع متنی.

**TREC** (کنفرانس بازیابی متن): معیاری که مدل‌ها را در وظایف مختلف بازیابی متن ارزیابی می‌کند و معمولاً بر بازیابی اطلاعات و جستجوی مدارک تمرکز دارد.

**WMT** (کارگاه ترجمه ماشینی): مجموعه داده و معیاری برای ارزیابی عملکرد سیستم‌های ترجمه ماشینی در زبان‌های مختلف.

**XNLI** (استنتاج زبان طبیعی چندزبان): مجموعه داده‌ای که برای ارزیابی توانایی یک مدل در درک و استنتاج معنا در چندین زبان طراحی شده است.

**PiQA** (پاسخ‌دهی به سوالات تعاملات فیزیکی): مجموعه داده‌ای که توانایی یک مدل را در درک تعاملات فیزیکی و وظایف روزمره اندازه‌گیری می‌کند.

**Winogrande**: مجموعه داده‌ای در مقیاس بزرگ که هدف آن ارزیابی توانایی یک مدل زبانی در رسیدگی به استدلال‌های مبتنی بر عقل سلیم است، معمولاً از طریق وظایفی که شامل حل ابهام‌های ضمیری در جملات می‌شود.

**RLHF** (یادگیری تقویتی از بازخورد انسانی): روشی که در آن مدل‌های زبانی بر اساس بازخورد ارائه شده توسط انسان‌ها تنظیم دقیق می‌شوند و معمولاً برای هدایت مدل‌ها به سمت رفتارها یا خروجی‌های مطلوب استفاده می‌شود.

**RAFT** (تنظیم دقیق تقویت شده با بازیابی): روشی که تکنیک‌های بازیابی را با تنظیم دقیق ترکیب می‌کند تا عملکرد مدل‌های زبانی را با اجازه دادن به آن‌ها برای دسترسی به اطلاعات خارجی در طول آموزش یا استنباط، بهبود بخشد.

## منابع

- [1] N-gram language models. <https://web.stanford.edu/~jurafsky/slp3/3.pdf>. [Accessed 01-07 2024].
- [2] Anis Koubaa. Gpt-4 vs. gpt-3.5: A concise showdown, 04 2023.
- [3] Timo Kaufmann, Paul Weng, Viktor Bengs, and Eyke H̄ ullermeier. A survey of reinforcement learning from human feedback, 2024.
- [4] Yu-Chu Chang, Xu Wang, Jindong Wang, Yuanyi Wu, Kaijie Zhu, Hao Chen, Linyi Yang, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, Weirong Ye, Yue Zhang, Yi Chang, Philip S. Yu, Qian Yang, and Xingxu Xie. A survey on evaluation of large language models. ACM Transactions on Intelligent Systems and Technology, 15:1– 45, 2023.
- [5] Ahtsham Zafar, Venkatesh Balavadhani Parthasarathy, Chan Le Van, Saad Shahid, Aafaq Iqbal Khan, and Arsalan Shahid. Building trust in conversational ai: A review and solution architecture using large language models and knowledge graphs. Big Data and Cognitive Computing, 8(6):70, 2024.
- [6] Zhibo Chu, Shiwen Ni, Zichong Wang, Xi Feng, Min Yang, and Wenbin Zhang. History, development, and principles of large language models-an introductory survey, 2024.
- [7] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space, 2013.
- [8] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.
- [9] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019.
- [10] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Re won Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. Palm: Scaling language modeling with pathways, 2022.

- [11] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models, 2023.
- [12] The depth art of — fine-tuning linkedin.com. large language models, explained in <https://www.linkedin.com/pulse/art-fine-tuning-large-language-models-explained-depth-cherickal-giavc>. 01-07-2024]. [Accessed 106]
- [13] Humza Naveed, Asad Ullah Khan, Shi Qiu, Muhammad Saqib, Saeed Anwar, Muhammad Usman, Naveed Akhtar, Nick Barnes, and Ajmal Mian. A comprehensive overview of large language models, 2024.
- [14] Jeff Li, MBA, PMP on LinkedIn: Fine-tuning versus RAG in Generative AI Applications Architecture — linkedin.com. [https://www.linkedin.com/posts/xjeffli\\_fine-tuning-versus-rag-in-generative-ai-applications-activity-7189276988690382848--vxT](https://www.linkedin.com/posts/xjeffli_fine-tuning-versus-rag-in-generative-ai-applications-activity-7189276988690382848--vxT). [Accessed 01-08-2024].
- [15] Tingfeng Hui, Zhenyu Zhang, Shuohuan Wang, Weiran Xu, Yu Sun, and Hua Wu. Hft: Half fine-tuning for large language models. arXiv preprint arXiv:2404.18466, 2024.
- [16] Rion Snow, Brendan O’Connor, Dan Jurafsky, and Andrew Y Ng. Cheap and fast—but is it good? evaluating non-expert annotations for natural language tasks. In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 254–263, 2008.
- [17] Alexander Ratner, Stephen H Bach, Henry Ehrenberg, Jason Fries, Sen Wu, and Christopher Ré. Snorkel: Rapid training data creation with weak supervision. In Proceedings of the VLDB Endowment, volume 11, pages 269–282, 2017.
- [18] Liang Ding, Philipp Gentner, Artur Duda, Vaibhav Sangtani, Dominik Ziegler, Max Hennen, Siddharth Jain, and Roland Werthschützky. Automatic data labeling for supervised learning with applications to visual inspection of mixed-plastic waste. Journal of Cleaner Production, 234:1033–1044, 2019.
- [19] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. In Proceedings of the International Conference on Learning Representations (ICLR), 2013.
- [20] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 1532–1543, 2014.
- [21] Rico Sennrich, Barry Haddow, and Alexandra Birch. Improving neural machine translation models with monolingual data. Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 86–96, 2016.
- [22] Javid Ebrahimi, Anyi Rao, Daniel Lowd, and Dejing Dou. Hotflip: White-box adversarial examples for text classification. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), pages 31–36, 2017.

- [23] Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. arXiv preprint arXiv:2005.14165, 2020.
- [24] Tianyu Gao, Adam Fisch, and Danqi Chen. Making pre-trained language models better few shot learners. In Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pages 3816–3830, 2021.
- [25] Steven Feng, Varun Gangal, Jinjun Wei, Yashvardhan Chandrasekhar, Yichong Chen, Dani He, Shuyang Huang, Faisal Ladha, Jiao Lee, Xinyi Li, et al. A survey of data augmentation approaches for nlp. arXiv preprint arXiv:2106.07499, 2021.
- [26] Suchin Gururangan, Ana Marasović, Swabha Swamyamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A Smith. Don’t stop pretraining: Adapt language models to domains and tasks. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 8342–8360, 2020.
- [27] Emily M Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. On the dangers of stochastic parrots: Can language models be too big? Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency, pages 610–623, 2021. 107
- [28] Reuben Binns. Fairness in machine learning: Lessons from political philosophy. Proceedings of the 2018 Conference on Fairness, Accountability, and Transparency, pages 149–159, 2018.
- [29] Sebastian Ruder. The stanford natural language inference (snli) corpus. arXiv preprint arXiv:1807.03519, 2021.
- [30] Pradeep Rajan, Krishna Vyas, Rajiv Bansal, Ranjan Sharma, and Shubhranshu Mukherjee. Machine learning for data preprocessing. *Journal of Big Data*, 6(1):1–25, 2019.
- [31] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16:321–357, 2002.
- [32] Connor Shorten and Taghi M Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of Big Data*, 6(1):1–48, 2019.
- [33] Alexander Ratner, Henry Ehrenberg, Zeshan Hussain, Jared Dunnmon, and Christopher Ré. Snorkel: Rapid training data creation with weak supervision. *Proceedings of the VLDB Endowment*, 11(3):269–282, 2020.
- [34] Solon Barocas, Moritz Hardt, and Arvind Narayanan. Fairness in machine learning: Lessons from political philosophy. In Proceedings of the 2017 ACM on Conference on Fairness, Accountability, and Transparency, pages 149–159, 2017.
- [35] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierrick Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. Transformers: State-of-the-

art natural language processing. Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, pages 38–45, 2020.

[36] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. Advances in Neural Information Processing Systems, 32, 2019.

[37] Mart’ in Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, et al. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. arXiv preprint arXiv:1603.04467, 2015.

[38] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805, 2018.

[39] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. arXiv preprint arXiv:1907.11692, 2019.

[40] Sheng Shen, Zhewei Dong, Xiaocheng Ye, Linjian Ma, Zhewei Li, Zirui Wang, Samyam Rajbhan dari, Yuxiong Wang, and Zhen Yang. Q-bert: Hessian based ultra low precision quantization of bert. Proceedings of the AAAI Conference on Artificial Intelligence, 34(05):8815–8821, 2020.

[41] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. OpenAI Blog, 1(8):9, 2019.

[42] Timnit Gebru, Jamie Morgenstern, Briana Vecchione, Jennifer Wortman Vaughan, Hanna Wallach, Hal Daumé III, and Kate Crawford. Datasheets for datasets. Communications of the ACM, 64(12):86–92, 2021.

[43] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980, 2014.

[44] Norman P Jouppi, Cliff Young, Nishant Patil, David Patterson, Gaurav Agrawal, Raminder Bajwa, Sarah Bates, Suresh Bhatia, Nan Boden, Al Borchers, et al. In-datacenter performance analysis of a tensor processing unit. Proceedings of the 44th Annual International Symposium on Computer Architecture, pages 1–12, 2017. 108

[45] Mart’ in Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: A system for large-scale machine learning. 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16), pages 265–283, 2016.

[46] Mohammad Shoeybi, Mostofa Patwary, Raghavendra Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. Megatron-lm: Training multi-billion parameter language models using model parallelism. arXiv preprint arXiv:1909.08053, 2019.

- [47] Yang You, Jing Li, Sashank Reddi, Jonathan Hseu, Sanjiv Kumar, Srinadh Bhojanapalli, Xiaodan Song, James Demmel, Cho-Jui Hsieh, and Payal Yadollahpour. Large batch optimization for deep learning: Training bert in 76 minutes. arXiv preprint arXiv:1904.00962, 2019.
- [48] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. Deep learning. 2016.
- [49] James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. Journal of Machine Learning Research, 13(2):281–305, 2012.
- [50] Frank Hutter, Lars Kotthoff, and Joaquin Vanschoren. Automated Machine Learning: Methods, Systems, Challenges. Springer Nature, 2019.
- [51] Lutz Prechelt. Early stopping-but when? Neural Networks: Tricks of the trade, pages 55–69, 1998.
- [52] Alexander Sergeev and Mike Del Balso. Horovod: fast and easy distributed deep learning in tensorflow. arXiv preprint arXiv:1802.05799, 2018.
- [53] Samyam Rajbhandari, Jeff Rasley, Olatunji Ruwase, and Yuxiong He. Deepspeed: Extreme-scale model training for everyone. arXiv preprint arXiv:2007.04822, 2020.
- [54] Paulius Micikevicius, Sharan Narang, Jonah Alben, Gregory Diamos, Erich Elsen, David Garcia, Boris Ginsburg, Michael Houston, Oleksii Kuchaiev, Ganesh Venkatesh, et al. Mixed precision training. arXiv preprint arXiv:1710.03740, 2018.
- [55] Karan Singhal, Tao Tu, Juraj Gottweis, Rory Sayres, Ellery Wulczyn, Le Hou, Kevin Clark, Stephen Pfahl, Heather Cole-Lewis, Darlene Neal, Mike Schaekermann, Amy Wang, Mohamed Amin, Sami Lachgar, Philip Mansfield, Sushant Prakash, Bradley Green, Ewa Dominowska, Blaise Aguera y Arcas, Nenad Tomasev, Yun Liu, Renee Wong, Christopher Semturs, S. Sara Mahdavi, Joelle Barral, Dale Webster, Greg S. Corrado, Yossi Matias, Shekoofeh Azizi, Alan Karthikesalingam, and Vivek Natarajan. Towards expert-level medical question answering with large language models, 2023.
- [56] Hongyang Yang, Xiao-Yang Liu, and Christina Dan Wang. Fingpt: Open-source financial large language models, 2023.
- [57] Zhi Zhou, Jiang-Xin Shi, Peng-Xiao Song, Xiao-Wen Yang, Yi-Xuan Jin, Lan-Zhe Guo, and Yu Feng Li. Lawgpt: A chinese legal knowledge-enhanced large language model, 2024.
- [58] Linqing Chen, Weilei Wang, Zilong Bai, Peng Xu, Yan Fang, Jie Fang, Wentao Wu, Lizhi Zhou, Ruiji Zhang, Yubin Xia, Chaobo Xu, Ran Hu, Licong Xu, Qijun Cai, Haoran Hua, Jing Sun, Jin Liu, Tian Qiu, Haowen Liu, Meng Hu, Xiuwen Li, Fei Gao, Yufu Wang, Lin Tie, Chaochao Wang, Jianping Lu, Cheng Sun, Yixin Wang, Shengjie Yang, Yuancheng Li, Lu Jin, Lisha Zhang, Fu Bian, Zhongkai Ye, Lidong Pei, and Changyang Tu. Pharmagpt: Domain-specific large language models for bio-pharmaceutical and chemistry, 2024.
- [59] Writer Engineering team. Palmyra-Fin-70B-32k: a powerful LLM designed for Finance. <https://dev.writer.com>, 2024.

- [60] Zeyu Han, Chao Gao, Jinyang Liu, Jeff Zhang, and Sai Qian Zhang. Parameter-efficient fine-tuning for large models: A comprehensive survey, 2024.
- [61] Lin Tian, Xiuzhen Zhang, and Jey Han Lau. Metatroll: Few-shot detection of state-sponsored trolls with transformer adapters. In Proceedings of the ACM Web Conference 2023, WWW '23. ACM, April 2023. 109
- [62] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models, 2021.
- [63] PhD Sebastian Raschka. Practical Tips for Finetuning LLMs Using LoRA (Low-Rank Adaptation) — magazine.sebastianraschka.com. <https://magazine.sebastianraschka.com/p/practical-tips-for-finetuning-langs>. [Accessed 01-08-2024].
- [64] Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms, 2023.
- [65] What is QLoRa? — Analytics Vidhya — community.analyticsvidhya.com. <https://community.analyticsvidhya.com/c/generative-ai-tech-discussion/what-is-qlora>. [Accessed 01-08 2024].
- [66] Shih-Yang Liu, Chien-Yi Wang, Hongxu Yin, Pavlo Molchanov, Yu-Chiang Frank Wang, Kwang Ting Cheng, and Min-Hung Chen. Dora: Weight-decomposed low-rank adaptation, 2024.
- [67] Apple intelligence foundation language models, 2024.
- [68] Tingfeng Hui, Zhenyu Zhang, Shuohuan Wang, Weiran Xu, Yu Sun, and Hua Wu. Hft: Half fine-tuning for large language models, 2024.
- [69] Johnny Li, Saksham Consul, Eda Zhou, James Wong, Naila Farooqui, Yuxin Ye, Nithyashree Manohar, Zhuxiaona Wei, Tian Wu, Ben Echols, Sharon Zhou, and Gregory Diamos. Banishing llm hallucinations requires rethinking generalization, 2024.
- [70] Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, Gianna Lengyel, Guillaume Bour, Guillaume Lample, L' elio Renard Lavaud, Lucile Saulnier, Marie Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, Szymon Antoniak, Teven Le Scao, Th'eophile Gervet, Thibaut Lavril, Thomas Wang, Timoth' ee Lacroix, and William El Sayed. Mixtral of experts, 2024.
- [71] Applying cal Blog Mixture of Experts in — developer.nvidia.com. LLM Architectures — NVIDIA Techni <https://developer.nvidia.com/blog/applying-mixture-of-experts-in-lm-architectures/>. [Accessed 01-08-2024].
- [72] Junlin Wang, Jue Wang, Ben Athiwaratkun, Ce Zhang, and James Zou. Mixture-of-agents enhances large language model capabilities, 2024.
- [73] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017.

[74] Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model, 2024.

[75] Shusheng Xu, Wei Fu, Jiaxuan Gao, Wenjie Ye, Weilin Liu, Zhiyu Mei, Guangju Wang, Chao Yu, and Yi Wu. Is dpo superior to ppo for llm alignment? a comprehensive study, 2024.

[76] What are the most effective techniques for pruning ai models? — linkedin.com.  
<https://www.linkedin.com/advice/3/what-most-effective-techniques-pruning-0mlef>. [Accessed 05-07 2024].

[77] Boxin Wang, Weixin Chen, Hengzhi Pei, Chulin Xie, Mintong Kang, Chenhui Zhang, Chejian Xu, Zidi Xiong, Ritik Dutta, Rylan Schaeffer, Sang T. Truong, Simran Arora, Mantas Mazeika, Dan Hendrycks, Zinan Lin, Yu Cheng, Sanmi Koyejo, Dawn Song, and Bo Li. Decodingtrust: A comprehensive assessment of trustworthiness in gpt models, 2024.

[78] Hakan Inan, Kartikeya Upasani, Jianfeng Chi, Rashi Rungta, Krithika Iyer, Yuning Mao, Michael Tontchev, Qing Hu, Brian Fuller, Davide Testuggine, and Madian Khabsa. Llama guard: Llm based input-output safeguard for human-ai conversations, 2023. 110

[79] Wenjun Zeng, Yuchi Liu, Ryan Mullins, Ludovic Peran, Joe Fernandez, Hamza Harkous, Karthik Narasimhan, Drew Proud, Piyush Kumar, Bhaktipriya Radharapu, Olivia Sturman, and Oscar Wahltinez. Shieldgemma: Generative ai content moderation based on gemma, 2024.

[80] Seungju Han, Kavel Rao, Allyson Ettinger, Liwei Jiang, Bill Yuchen Lin, Nathan Lambert, Yejin Choi, and Nouha Dziri. Wildguard: Open one-stop moderation tools for safety risks, jailbreaks, and refusals of llms, 2024.

[81] Vishal Magic Mysore. , Its LLM Logic! Deployment — visrow. Strategies : Its not <https://medium.com/@visrow/llm-deployment-strategies-its-not-magic-its-logic-71d5f32ac2b4>. 2024]. [Accessed 07-08

[82] Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention, 2023.

[83] Preprocess and fine-tune llms quickly and cost-effectively using amazon emr serverless and amazon sagemaker — aws.amazon.com. <https://aws.amazon.com/blogs/big-data/preprocess-and-fine-tune-llms-quickly-and-cost-effectively-using-amazon-emr-serverless-and-amazon-sagemaker/> [Accessed 06-08-2024].

[84] Nvidia nemo build and customize your own llms (with tutorial) — run.ai. <https://www.run.ai/guides/ai-open-source-projects/nvidia-nemo>. [Accessed 07-08-2024].

[85] Nvidia. What is nvidia nemo? <https://www.nvidia.com/en-us/ai-data-science/products/nemo/>.

[86] Gemini Team and Rohan Anil et al. Gemini: A family of highly capable multimodal models, 2024.

- [87] Yizhang Jin, Jian Li, Yexin Liu, Tianjun Gu, Kai Wu, Zhengkai Jiang, Muyang He, Bo Zhao, Xin Tan, Zhenye Gan, Yabiao Wang, Chengjie Wang, and Lizhuang Ma. Efficient multimodal large language models: A survey, 2024.
- [88] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision, 2021.
- [89] Haokun Liu, Derek Tam, Mohammed Muqeeth, Jay Mohta, Tenghao Huang, Mohit Bansal, and Colin Raffel. Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning, 2022.
- [90] Mojtaba Valipour, Mehdi Rezagholizadeh, Ivan Kobyzev, and Ali Ghodsi. Dylora: Parameter efficient tuning of pre-trained models using dynamic search-free low-rank adaptation, 2023.
- [91] Longteng Zhang, Lin Zhang, Shaohuai Shi, Xiaowen Chu, and Bo Li. Lora-fa: Memory-efficient low-rank adaptation for large language models fine-tuning, 2023.
- [92] Qiong Wu, Weihao Ye, Yiyi Zhou, Xiaoshuai Sun, and Rongrong Ji. Not all attention is needed: Parameter and computation efficient transfer learning for multi-modal large language models, 2024.
- [93] Shibo Jie, Yehui Tang, Ning Ding, Zhi-Hong Deng, Kai Han, and Yunhe Wang. Memory-space visual prompting for efficient vision-language fine-tuning, 2024.
- [94] Kai Lv, Yuqing Yang, Tengxiao Liu, Qinghui Gao, Qipeng Guo, and Xipeng Qiu. Full parameter fine-tuning for large language models with limited resources, 2024.
- [95] Sadhika Malladi, Tianyu Gao, Eshaan Nichani, Alex Damian, Jason D. Lee, Danqi Chen, and Sanjeev Arora. Fine-tuning language models with just forward passes, 2024.
- [96] Gang Liu, Jinlong He, Pengfei Li, Genrong He, Zhaolin Chen, and Shenjun Zhong. Pefomed: Parameter efficient fine-tuning of multimodal large language models for medical imaging, 2024. 111
- [97] Wei-Ning Hsu, Benjamin Bolte, Yao-Hung Hubert Tsai, Kushal Lakhotia, Ruslan Salakhutdinov, and Abdelrahman Mohamed. Hubert: Self-supervised speech representation learning by masked prediction of hidden units, 2021.
- [98] Alexei Baevski, Henry Zhou, Abdelrahman Mohamed, and Michael Auli. wav2vec 2.0: A framework for self-supervised learning of speech representations, 2020.
- [99] Deepak chitecture Babu P R. Audio — prdeepak.babu. language models and multimodal ar <https://medium.com/@prdeepak.babu/audio-language-models-and-multimodal-architecture-1cdd90f46fac>. 2024]. [Accessed 19-07]
- [100] Paul K. Rubenstein, Chulayuth Asawaroengchai, Duc Dung Nguyen, Ankur Bapna, Zal'an Borsos, Félix de Chaumont Quiry, Peter Chen, Dalia El Badawy, Wei Han, Eugene Kharitonov, Hannah Muckenhirk, Dirk Padfield, James Qin, Danny Rozenberg, Tara Sainath, Johan Schalkwyk, Matt Sharifi, Michelle Tadmor Ramanovich, Marco Tagliasacchi, Alexandru Tudor, Mihajlo

Velimirović, Damien Vincent, Jiahui Yu, Yongqiang Wang, Vicky Zayats, Neil Zeghidour, Yu Zhang, Zhishuai Zhang, Lukas Zilka, and Christian Frank. Audiopalm: A large language model that can speak and listen, 2023.

[101] Zalán Borsos, Raphaël Marinier, Damien Vincent, Eugene Kharitonov, Olivier Pietquin, Matt Sharifi, Dominik Roblek, Olivier Teboul, David Grangier, Marco Tagliasacchi, and Neil Zeghidour. Audiolum: a language modeling approach to audio generation, 2023.

[102] Humza Naveed, Asad Ullah Khan, Shi Qiu, Muhammad Saqib, Saeed Anwar, Muhammad Usman, Naveed Akhtar, Nick Barnes, and Ajmal Mian. A comprehensive overview of large language models, 2024.

[103] Fine-tune Llama 2 with Lora: Customizing a large language model for question-answering — rocm.blogs.amd.com. <https://rocm.blogs.amd.com/artificial-intelligence/llama2-lora/README.html>. [Accessed 15-07-2024].

[104] Aayush Mittal. Towards Understanding LLM fine-tuning: Tailoring large language models to your unique requirements — linkedin.com. <https://www.unite.ai/understanding-llm-fine-tuning-tailoring-large-language-models-to-your-unique-requirements>. [Accessed 11-07-2024].

[105] Alan Ansell, Ivan Vulić, Hannah Sterz, Anna Korhonen, and Edoardo M. Ponti. Scaling sparse fine-tuning to large language models, 2024.

[106] Xinyu Lin, Wenjie Wang, Yongqi Li, Shuo Yang, Fuli Feng, Yinwei Wei, and Tat-Seng Chua. Data-efficient fine-tuning for LLM-based recommendation, 2024.

[107] Yue Liu, Shihao Zhu, Jun Xia, Yingwei Ma, Jian Ma, Wenliang Zhong, Xinwang Liu, Guannan Zhang, and Kejun Zhang. End-to-end learnable clustering for intent learning in recommendation, 2024.

[108] Haoran Li, Xinyuan Zhao, Dadi Guo, Hanlin Gu, Ziqian Zeng, Yuxing Han, Yangqiu Song, Lixin Fan, and Qiang Yang. Federated domain-specific knowledge transfer on large language models using synthetic data, 2024.

[109] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks, 2019.