Q2.大模型为什么需要微调?
大模型如何进行微调?

占位符

通用理解	机器学习领域	大模型领域
学模型调用方法	安装sklearn 学习sklearn中各模型API	学习OpenAI各模型API 部署开源大模型，并掌握API使用方法
学模型优化方法	学习算法超参数优化器 对模型进行超参数优化	学习目前主流大模型微调框架 掌握大模型微调方法
学习完整的应用流程	学习特征工程、模型融合方法 掌握pipeline构建机器学习流	掌握基于LangChain的AI应用开发范式

占位符

要理解如何微调？用什么方法微调？

需要补充介绍大语言模型深度学习背景知识

LLM大语言模型
一般训练过程

占位符

Step 1.预训练阶段

大模型首先在大量的无标签数据上进行训练，预训练的最终目的是让模型学习到语言的统计规律和一般知识。在这个过程中模型能够学习到词语的语义、句子的语法结构、以及文本的一般知识和上下文信息。需要注意的是，预训练本质上是一个无监督学习过程；

Step 2.微调阶段

预训练好的模型然后在特定任务的数据上进行进一步的训练。这个过程通常涉及对模型的权重进行微小的调整，以使其更好地适应特定的任务；

LLM大语言模型
一般训练过程

占位符

Step 1.预训练阶段

得到预训练模型（Pretrained Model），也被称为基座模型（Base Model），模型具备通用的预测能力。如GLM-130B模型、OpenAI的A、B、C、D四大模型，都是基座模型；

Step 2.微调阶段

得到最终能力各异的模型，例如gpt code系列、gpt text系列、ChatGLM-6B等模型；

什么是大模型微调?

占位符

- **感性理解：**大模型微调指的是“喂”给模型更多信息，对模型的特定功能进行“调教”，即通过输入特定领域的数据集，让其学习这个领域的知识，从而让大模型能够更好的完成特定领域的NLP任务，例如情感分析、命名实体识别、文本分类、对话聊天等；

为什么需要微调?

占位符

- 核心原因还是在于需要“赋予”大模型更加定制化的功能，例如结合本地知识库进行检索、围绕特定领域问题进行问答等；
- 例如，VisualGLM是通用多模态大模型，但应用于医学影像判别领域，则需要代入医学影像领域的数据集来进行大模型微调，从而使得模型能够更好的围绕医学影像图片进行识别；
- 就像机器学习模型的超参数优化，只有调整了超参数，才能让模型更佳适用于当前的数据集；
- 同时，大模型是可以多次进行微调，每次微调都是一次能力的调整，即我们可以在现有的、已经具备某些特定能力的大模型基础上进一步进行微调；

占位符

补充介绍大模型预训练和微调两个核心阶段

涉及到的深度学习基础概念

预训练模型的核心模型特性：

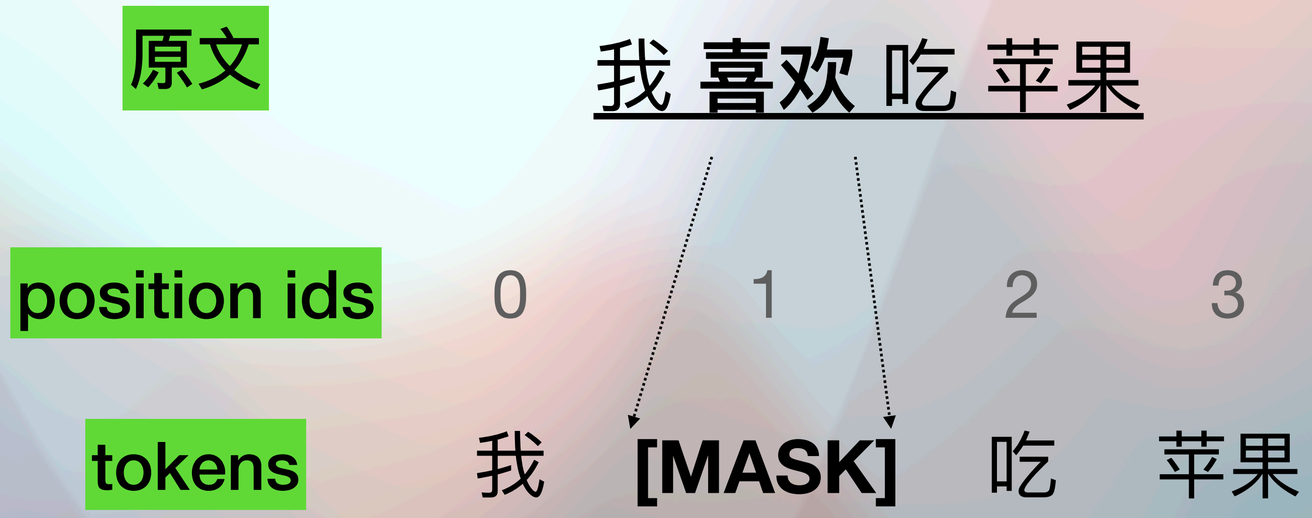
自回归与生成式

占位符

- 自回归概念，大模型的预训练过程是采用了一种名为自回归（Autoregressive）的方法，自回归模型是一种序列模型，它在预测下一个输出时，会将之前的所有输出作为输入，然后根据统计规律、结合已经输入的样本，预测下个位置各单词出现的概率，然后输出概率最大的单词，类似于完形填空；

预训练模型的核心模型特性：

自回归与生成式



- 自回归概念，大模型的预训练过程是采用了一种名为自回归（Autoregressive）的方法，自回归模型是一种序列模型，它在预测下一个输出时，会将之前的所有输出作为输入，然后根据统计规律、结合已经输入的样本，预测下个位置各单词出现的概率，然后输出概率最大的单词，类似于完形填空；

占位符

大模型预训练过程的核心概念：

自回归与生成式模型

占位符

- 生成式概念：与之类似的还有一个名为生成式模型的概念，也就是GPT中的G（Generative），所谓生成式模型的预测过程和自回归模型类似，都是根据统计规律预测下个单词的概率，所不同的是，生成式模型可以根据之前的样本的概率分布生成下一个词，生成式模型预测时会存在一定的随机性；

大模型预训练过程的核心概念：

自回归与生成式模型

占位符

- 不过自回归和生成式模型并不冲突，例如对于GPT来说，就是一个自回归生成式模型；
- 一个自回归生成式模型在进行预测的时候，会首先根据自回归模型，在参考到目前为止已经生成的词的情况下确定下一个词的概率分布，然后再根据生成式的方式来根据这个分布生成下一个词；
- 从最终的表现上来看，GPT这类自回归生成式模型在完成预训练之后，就获得了所谓的“Zero-Shot”或“Few-Shot”学习。即使没有针对新任务的训练数据，预训练模型也可以在新任务上有一定的表现；

自回归模型预测过程的极简示例

占位符

预测任务

单词的概率分布

The cat is on the ___

floor	20%
bed	14%
mat	35%

- 大模型能够创建语意连贯的句子，但其本身并不理解句子的含义；
- 正是因为GPT的生成性的特性，所以很多时候会“编造”一些事实；
- 预训练的过程实际上就是“大力出奇迹”的过程，模型参数越多、输入的数据越多，就越能够“读书百遍、其义自现”

- **补充概念一：**

双向自回归与自回归

主要的区别就在于自回归模型只看前文，而双向自回归模型会同时考虑前文和后文。

- **自回归预训练**：在这种方法中，模型在预测下一个词时，只会考虑已经看过的词（即前文）。例如，如果你正在阅读这个句子：“我喜欢吃…”，自回归模型会根据“我喜欢吃”来预测下一个词。这种方式类似于我们人类阅读文本的方式，我们通常是从左到右，一次读一个词，然后根据已读的内容预测下一个词。GPT系列模型就是使用自回归预训练的；
- **双向自回归预训练**：在这种方法中，模型在预测某个词时，会同时考虑该词的前文和后文。例如，如果你正在阅读这个句子：“我喜欢吃…苹果”，即使"苹果"这个词是被预测的目标，双向自回归模型仍会考虑“我喜欢吃”（前文）和“苹果”（后文）来预测这个词。这种方式类似于你同时读到了整个句子，然后再去理解每个词的含义。BERT和GLM模型就是采用的双向自回归进行预训练；

- **补充概念二：**

生成式模型与判别模型

很长一段时间，生成模型效果一直不如判别模型，直至“大模型”涌现能力出现

占位符

- 生成式模型：判别式模型直接学习输入到输出的映射，判别模型试图找到输入和输出之间的直接关系，而不是试图理解输入数据或输出数据的整体分布，因此通常在小规模数据和模型中能得到更好的性能。此外，判别式模型的训练过程通常比生成式模型更加稳定和直接；
- 判别类模型：试图学习数据的整体分布，这通常需要更大规模的模型和数据。在小规模模型中，学习这种复杂的分布可能会非常困难。因此，在深度学习早期，生成式模型的应用确实相对有限。然而，随着计算能力的提升和大规模数据集的出现，生成式模型的潜力逐渐得到展现；

占位符

一个经过预训练的大语言模型，就具备“一定程度的”通用能力，而只有经过微调，才能够让模型具备解决某项具体任务的能力

深度学习模型（大语言模型）

微调算法基本背景介绍

占位符

- 微调并不是大模型领域独有的概念，而是伴随着深度学习技术发展，自然诞生的一个技术分支，旨在能够有针对性的调整深度学习模型的参数（或者模型结构），从而能够使得其更佳高效的执行某些特定任务，而不用重复训练模型；
- 伴随着大模型技术的蓬勃发展，微调技术一跃成为大模型工程师必须要掌握的核心技术。并且，伴随着大模型技术的蓬勃发展，越来越多的微调技术也在不断涌现；

深度学习模型（大语言模型）

微调算法核心概念介绍

占位符

- **微调数据**：不同于大模型预训练过程可以代入无标签样本，深度学习模型微调过程需要代入有标签的样本来进行训练，微调的本质是一个有监督学习过程；
- **微调的本质**：相比海量数据在超大规模模型上进行预训练，微调只需要一部分有标签数据、并对模型的部分参数进行修改即可使得模型获得某种特殊能力；因此，从所需数据量和算力消耗来看，相比于训练微调的门槛和成本都要低很多；
- **例如**：对于一个经过预训练的大模型，如果我们代入诸多标有“垃圾邮件”和“非垃圾邮件”的标签文本进行训练，则可训练其具备垃圾邮件分类这一能力；

深度学习模型（大语言模型）

微调算法核心概念介绍

占位符

- **从模型本身角度而言：**微调阶段相当于是进一步进行训练，该过程会修改模型参数，并最终使模型“记住”了这些额外信息；**让大模型永久记住信息的唯一方法就是修改参数；**
- **有监督微调：supervised fine-tuning，简称SFT；**
- **数据标注：**高质量的有标签数据集在微调过程中必不可少，数据标注工作则是用于创建这些有标签的数据集；伴随着大模型发展，人们也在尝试使用大模型来完成很多数据标注工作；

- **概念补充：**

提示工程也能
让模型暂时记住一些信息

占位符

- 除了微调外，提示工程也能通过让大模型“暂时记住”某些信息，从而影响或优化模型输出结果；
- 从算法原理层面来说，当一个模型接收到一个输入时，它会在内部创建一个表示这个输入的数字向量，这被称为模型的”隐藏状态"。然后，模型使用这个隐藏状态来生成一个回答。模型的隐藏状态是基于其接收到的所有输入生成的，因此它包含了模型当前所"知道"的所有信息。这也是为什么模型的输出可能会受到输入的所有部分（包括上下文）的影响。
- 不过，大模型的隐藏状态是有接受信息的上限的，当超过这个上限，模型就会“遗忘”最开始的信息。这也就是为什么ChatGPT在进行对话的时候会有Token上限。也正是因为存在上限，提示工程在让模型“记住”信息方面，作用有限；

- **概念补充：**

大模型时代
AI应用开发的新范式

占位符

- **NLP“小模型”时代开发流程：**一个（类）场景训练一个模型。在传统的自然语言处理（NLP）中，通常会针对每个具体任务（如情感分析、命名实体识别、文本分类等）训练一个专门的模型。这些模型通常需要大量的标注数据，并且通常很难泛化到不同的任务或不同类型的数据；
- **通用大模型时代：**一个通用的大模型能够在很多领域完成Zero-Shot。伴随着大模型的诞生，这些经过海量数据预训练的大模型已经能够在预训练阶段学习到了丰富的语言知识，包括语法、词汇、一般常识和一些任务相关的模式。因此，这些模型在许多通用的自然语言处理（NLP）任务上，即使没有接受过特定任务的训练，也能够达到很好的效果。

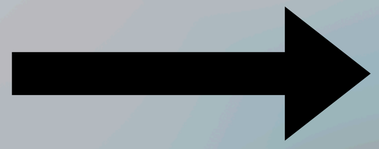
- 概念补充：

大模型时代
AI应用开发的新范式

占位符

- **大模型时代的开发流程：**由于通用大模型在很多通用NLP任务上都有非常好的表现，因此针对某项特定的任务，我们只需要选择一个大模型，然后带入特定领域的专业数据集对其进行微调，即可非常高效的完成对应任务；

一个（类）任务
单独训练一个模型



选择一个大模型
围绕具体任务进行微调

占位符

大模型微调重要性不言而喻，
那么到底有哪些微调方法呢?

大模型微调重要性不言而喻，

那么到底有哪些微调方法呢?

占位符

第一类方法：借助OpenAI提供的在线微调工具进行微调；

第二类方法：借助开源微调框架进行微调；

OpenAI系列模型微调关系

官网说明地址： https://platform.openai.com/docs/model-index-for-researchers

Models referred to as "GPT 3.5"

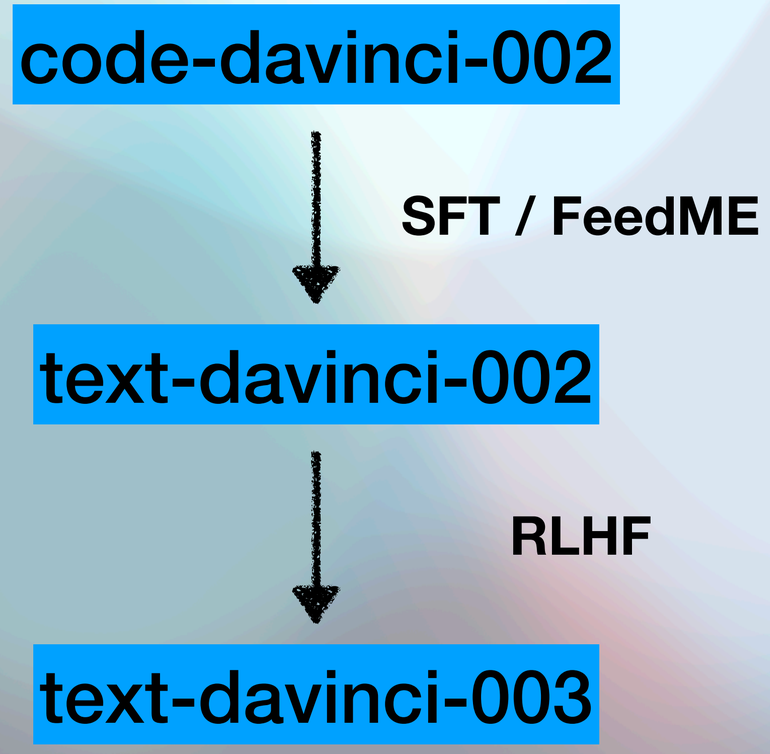
GPT-3.5 series is a series of models that was trained on a blend of text and code from before Q4 2021. The following models are in the GPT-3.5 series:

1. `code-davinci-002` is a base model, so good for pure code-completion tasks
2. `text-davinci-002` is an InstructGPT model based on `code-davinci-002`
3. `text-davinci-003` is an improvement on `text-davinci-002`
4. `gpt-3.5-turbo-0301` is an improvement on `text-davinci-003`, optimized for chat

InstructGPT models

We offer variants of InstructGPT models trained in 3 different ways:

TRAINING METHOD	MODELS
SFT Supervised fine-tuning on human demonstrations	`davinci-instruct-beta`[1]
FeedME Supervised fine-tuning on human-written demonstrations and on model samples rated 7/7 by human labelers on an overall quality score	`text-davinci-001`, `text-davinci-002`, `text-curie-001`, `text-babbage-001`
PPO Reinforcement learning with reward models trained from comparisons by humans	`text-davinci-003`



OpenAI大模型微调API

API地址

https://platform.openai.com/docs/guides/fine-tuning

- OpenAI提供了GPT在线大模型的微调API：Fine-tuning，用于在线微调在线大模型；
- 目前支持A、B、C、D四大模型在线微调；
- 根据格式要求在线提交数据集并支付算力费用，即可在线进行模型微调；
- 微调后会生成单独模型的API供使用；
- 一个模型可以多次微调；OpenAI在线微调是黑箱算法；

占位符

OpenAI API在线微调流程

API地址

https://platform.openai.com/docs/guides/fine-tuning

占位符

OpenAI提供的“傻瓜式”微调流程

At a high level, fine-tuning involves the following steps:

1. Prepare and upload training data
2. Train a new fine-tuned model
3. Use your fine-tuned model

Visit our pricing page to learn more about how fine-tuned model training and usage are billed.

- 按照格式要求，准备并上传数据集；
- 排队、支付费用并等待微调模型训练完成；
- 赋予微调模型API单独编号，调用API即可使用；

大模型微调重要性不言而喻，
那么到底有哪些微调方法呢?

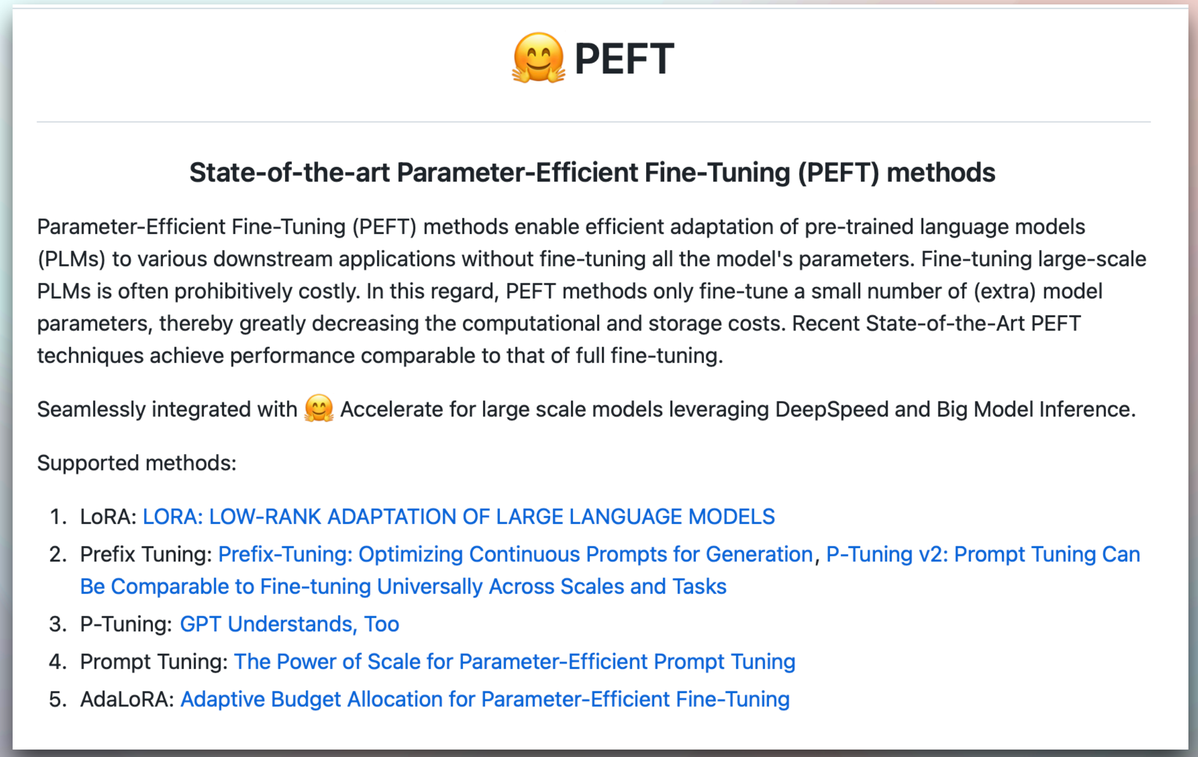
占位符

第一类方法：借助OpenAI提供的在线微调工具进行微调；

第二类方法：借助开源微调框架进行微调；

高效微调技术方法概述

Hugging Face PEFT项目地址
https://github.com/huggingface/peft



🤗 PEFT

State-of-the-art Parameter-Efficient Fine-Tuning (PEFT) methods

Parameter-Efficient Fine-Tuning (PEFT) methods enable efficient adaptation of pre-trained language models (PLMs) to various downstream applications without fine-tuning all the model's parameters. Fine-tuning large-scale PLMs is often prohibitively costly. In this regard, PEFT methods only fine-tune a small number of (extra) model parameters, thereby greatly decreasing the computational and storage costs. Recent State-of-the-Art PEFT techniques achieve performance comparable to that of full fine-tuning.

Seamlessly integrated with 🤗 Accelerate for large scale models leveraging DeepSpeed and Big Model Inference.

Supported methods:

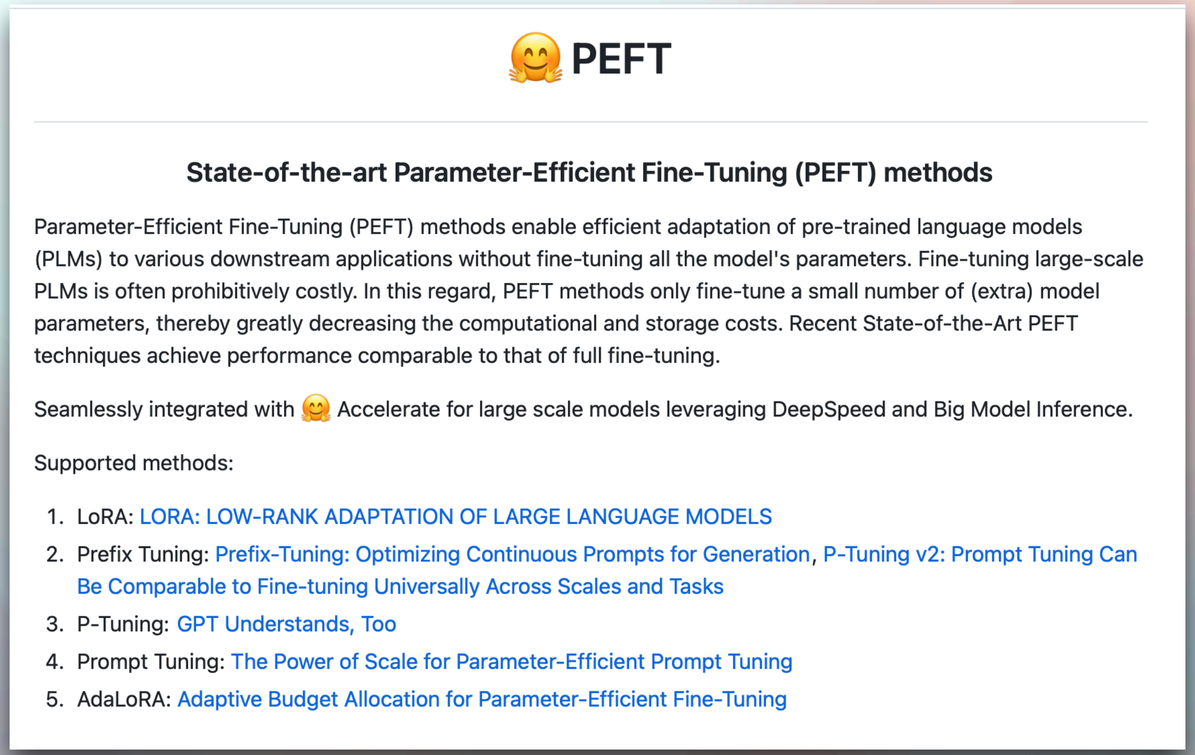
1. LoRA: LORA: LOW-RANK ADAPTATION OF LARGE LANGUAGE MODELS
2. Prefix Tuning: Prefix-Tuning: Optimizing Continuous Prompts for Generation, P-Tuning v2: Prompt Tuning Can Be Comparable to Fine-tuning Universally Across Scales and Tasks
3. P-Tuning: GPT Understands, Too
4. Prompt Tuning: The Power of Scale for Parameter-Efficient Prompt Tuning
5. AdaLoRA: Adaptive Budget Allocation for Parameter-Efficient Fine-Tuning

- 深度学习微调方法非常多，主流方法包括LoRA、Prefix Tuning、P- Tuning、Promt Tuning、AdaLoRA等；
- 目前这些方法的实现均已集成至Hugging Face项目的库中，我们可以通过安装和调用Hugging Face的PEFT（高效微调）库，来快速使用这些方法；

占位符

高效微调技术方法概述

Hugging Face PEFT项目地址
https://github.com/huggingface/peft



🤗 PEFT

State-of-the-art Parameter-Efficient Fine-Tuning (PEFT) methods

Parameter-Efficient Fine-Tuning (PEFT) methods enable efficient adaptation of pre-trained language models (PLMs) to various downstream applications without fine-tuning all the model's parameters. Fine-tuning large-scale PLMs is often prohibitively costly. In this regard, PEFT methods only fine-tune a small number of (extra) model parameters, thereby greatly decreasing the computational and storage costs. Recent State-of-the-Art PEFT techniques achieve performance comparable to that of full fine-tuning.

Seamlessly integrated with 🤗 Accelerate for large scale models leveraging DeepSpeed and Big Model Inference.

Supported methods:

1. LoRA: LORA: LOW-RANK ADAPTATION OF LARGE LANGUAGE MODELS
2. Prefix Tuning: Prefix-Tuning: Optimizing Continuous Prompts for Generation, P-Tuning v2: Prompt Tuning Can Be Comparable to Fine-tuning Universally Across Scales and Tasks
3. P-Tuning: GPT Understands, Too
4. Prompt Tuning: The Power of Scale for Parameter-Efficient Prompt Tuning
5. AdaLoRA: Adaptive Budget Allocation for Parameter-Efficient Fine-Tuning

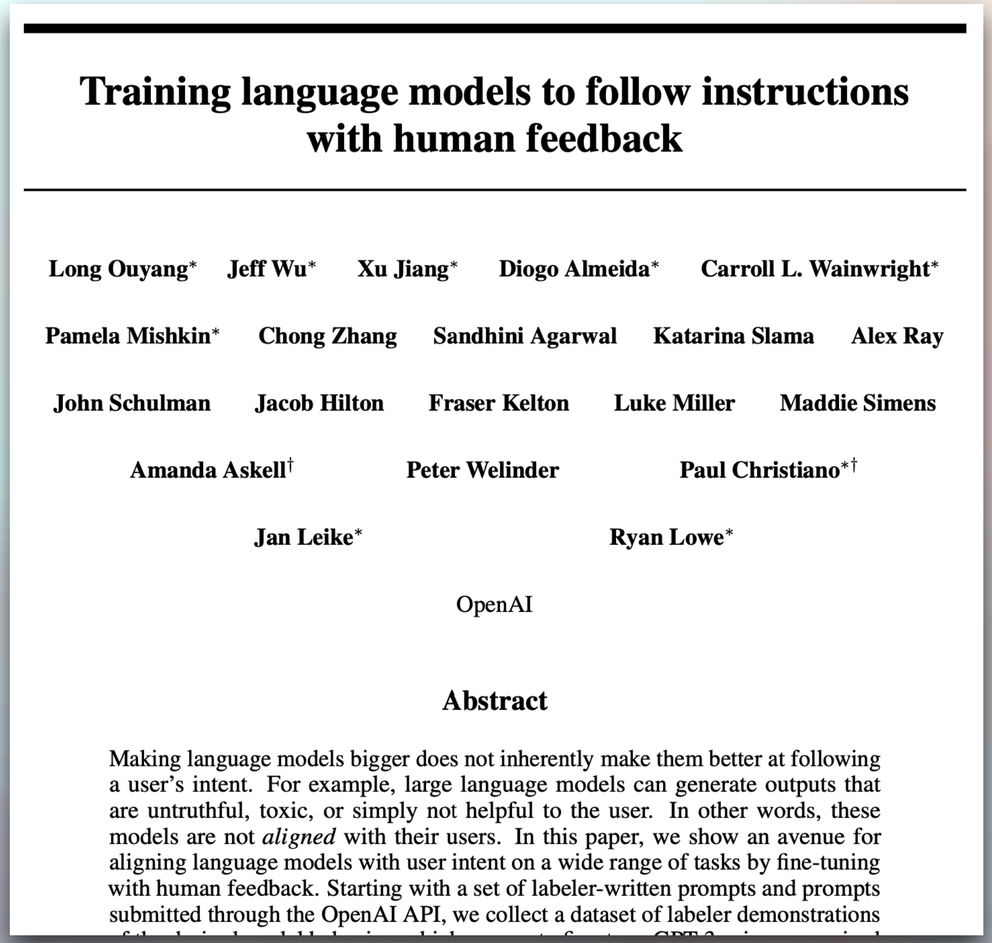
- Hugging Face 是一家专注于自然语言处理（NLP）技术的公司，同时也开发并维护了多个广受欢迎的自然语言处理的开源库和工具，如Transformers 库、ChatGLM-6B库等；
- 高效微调，State-of-the-art Parameter-Efficient Fine-Tuning (SOTA PEFT)，与之对应的更早期的简单的全量参数训练微调的方法（Fine- Tuning），高效微调则是一类算力功耗比更高的方法；

占位符

基于强化学习的进阶微调方法

RLHF方法

论文地址：https://arxiv.org/abs/2203.02155



Training language models to follow instructions with human feedback

Long Ouyang* **Jeff Wu*** **Xu Jiang*** **Diogo Almeida*** **Carroll L. Wainwright***

Pamela Mishkin* **Chong Zhang** **Sandhini Agarwal** **Katarina Slama** **Alex Ray**

John Schulman **Jacob Hilton** **Fraser Kelton** **Luke Miller** **Maddie Simens**

Amanda Askell† **Peter Welinder** **Paul Christiano*†**

Jan Leike* **Ryan Lowe***

OpenAI

Abstract

Making language models bigger does not inherently make them better at following a user's intent. For example, large language models can generate outputs that are untruthful, toxic, or simply not helpful to the user. In other words, these models are not *aligned* with their users. In this paper, we show an avenue for aligning language models with user intent on a wide range of tasks by fine-tuning with human feedback. Starting with a set of labeler-written prompts and prompts submitted through the OpenAI API, we collect a dataset of labeler demonstrations

- RLHF：Reinforcement Learning from Human Feedback，即基于人工反馈机制的强化学习。最早与2022年4月，由OpenAI研究团队系统总结并提出，并在GPT模型的对话类任务微调中大放异彩，被称为ChatGPT“背后的功臣”；
- RLHF也是目前为止常用的、最为复杂的基于强化学习的大语言模型微调方法，目前最好的端到端RLHF实现是DeepSpeedChat库，由微软开源并维护；

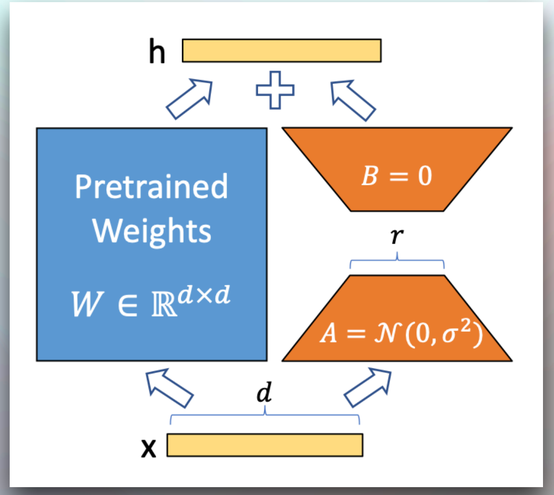
占位符

高效微调方法一：LoRA

- Github地址： https://github.com/microsoft/LoRA
- 论文地址： https://arxiv.org/abs/2106.09685

占位符

LoRA：LOW-RANK ADAPTATION OF LARGE LANGUAGE MODELS（2021）
基于低阶自适应的大语言模型微调方法



- 原理简述：基于大模型的内在低秩特性，增加旁路矩阵来模拟全参数微调；
- LoRA最早由微软研究院发布的一项微调技术；
- 简而言之，是通过修改模型结构进行微调，是一种四两拨千斤的微调方法，是目前最通用、同时也是效果最好的微调方法之一；
- 课程将详细介绍LoRA微调实践方法；

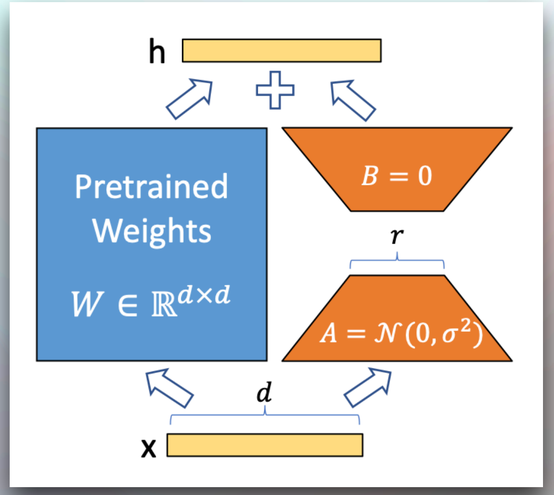
高效微调方法一：LoRA

- Github地址： https://github.com/microsoft/LoRA
- 论文地址： https://arxiv.org/abs/2106.09685

占位符

LoRA：LOW-RANK ADAPTATION OF LARGE LANGUAGE MODELS（2021）

基于低阶自适应的大语言模型微调方法



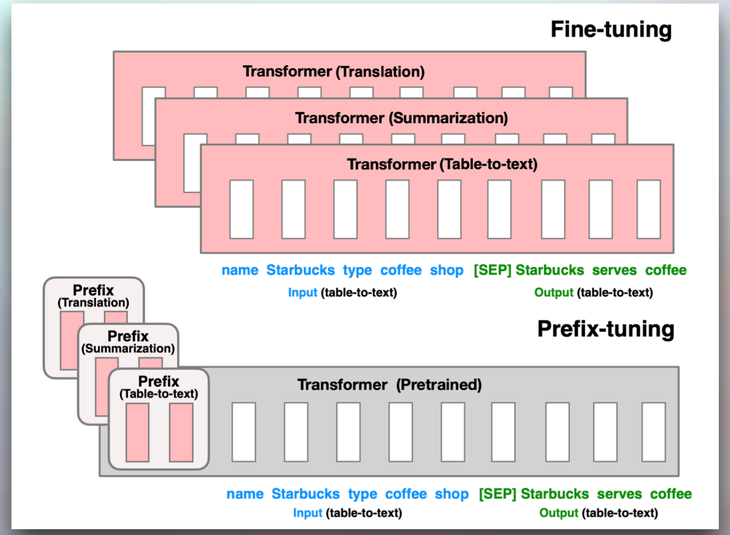
- 概念辨析：大模型微调LoRA与通信技术LoRa，二者相差一个字母的大小写，是完全两种不同的技术；
- LoRA除了可以用于微调大语言模型（LLM）外，目前还有一个非常火爆的应用场景：围绕diffusion models（扩散模型）进行微调，并在图片生成任务中表现惊艳；

高效微调方法二： Prefix Tuning

- 论文地址： https://aclanthology.org/2021.acl-long.353/

占位符

Prefix-Tuning: Optimizing Continuous Prompts for Generation（2021）
基于提示词前缀优化的微调方法



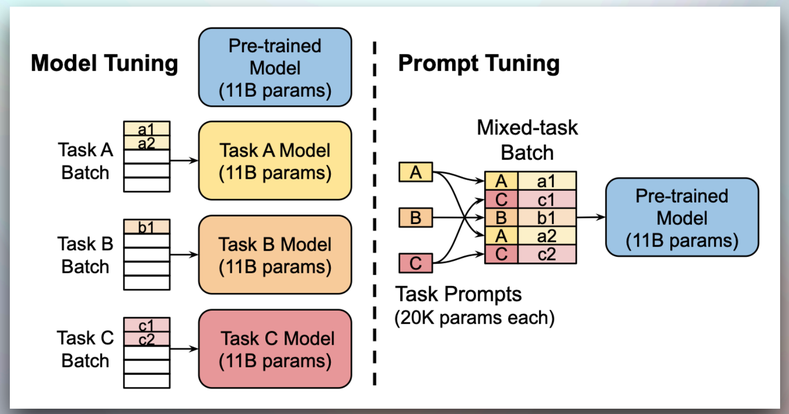
- 来源于斯坦福大学的一种高效微调方法；
- 原理简述：在原始模型基础上，增加一个可被训练的Embedding层，用于给提示词增加前缀，从而让模型更好的理解提示词意图，并在训练过程中不断优化这些参数；
- Prefix Tuning既能够在模型结构上增加一些新的灵活性，又能够在模型使用上提供一种自动的、能够改进模型表现的提示机制；

高效微调方法三： Prompt Tuning

- 论文地址： https://arxiv.org/abs/2104.08691

占位符

The Power of Scale for Parameter-Efficient Prompt Tuning （2021）



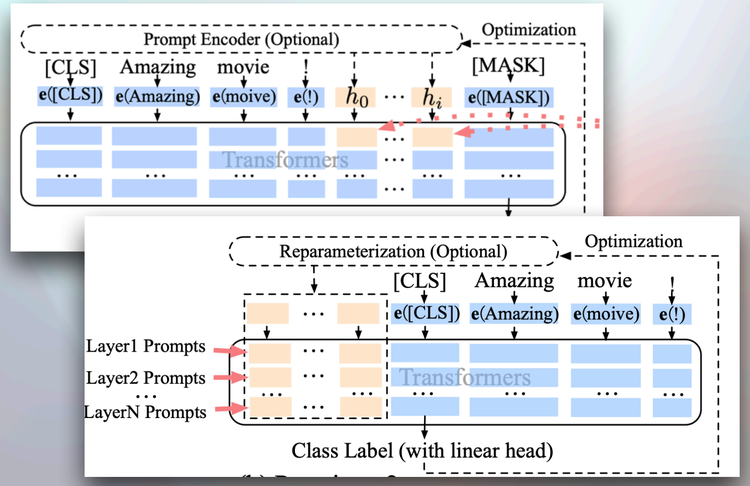
- 由谷歌提出的一种轻量级的优化方法；
- 原理简述：该方法相当于是Prefix Tuning的简化版本，即无需调整模型参数，而是在已有的参数中，选择一部分参数作为可学习参数，用于创建每个Prompt的前缀，从而帮助模型更好地理解和处理特定的任务；
- 不同于Prefix方法，Prompt Tuning训练得到的前缀是具备可解释性的，我们可以通过查看这些前缀，来查看模型是如何帮我们优化prompt的；
- 该方法在参数规模非常大的模型微调时效果很好，当参数规模达到100亿时和全量微调效果一致；

高效微调方法四：P-Tuning v2

- GitHub地址：https://github.com/THUDM/P-tuning-v2
- 论文地址： https://aclanthology.org/2021.acl-long.353/
- ChatGLM-6B+P-Tuning微调项目地址： https://github.com/THUDM/ChatGLM-6B/blob/main/ptuning/README.md

占位符

P-Tuning v2: Prompt Tuning Can Be Comparable to Fine-tuning Universally Across Scales and Tasks（2022）



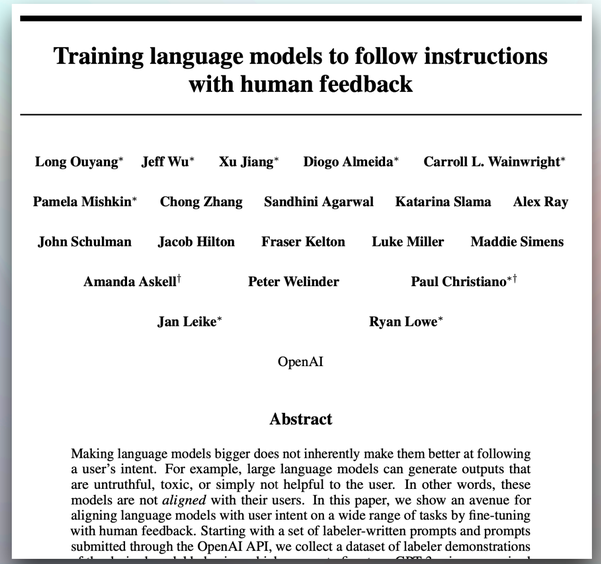
- 来源于清华大学团队提出的高效微调方法；
- 原理简述：可以理解为Prefix tuning的改进版本，即P-Tuning v2不仅在输入层添加了连续的prompts（可被训练的Embedding层），而且还在预训练模型的每一层都添加了连续的prompts；
- 这种深度的prompt tuning增加了连续prompts的容量，出于某些原因，P-Tuning v2会非常适合GLM这种双向预训练大模型微调；

基于强化学习的进阶微调方法

占位符

RLHF：Reinforcement Learning from Human Feedback

基于人工反馈机制的强化学习方法



Training language models to follow instructions with human feedback

Long Ouyang* Jeff Wu* Xu Jiang* Diogo Almeida* Carroll L. Wainwright*

Pamela Mishkin* Chong Zhang Sandhini Agarwal Katarina Slama Alex Ray

John Schulman Jacob Hilton Fraser Kelton Luke Miller Maddie Simens

Amanda Askell† Peter Welinder Paul Christiano*†

Jan Leike* Ryan Lowe*

OpenAI

Abstract

Making language models bigger does not inherently make them better at following a user's intent. For example, large language models can generate outputs that are untruthful, toxic, or simply not helpful to the user. In other words, these models are not *aligned* with their users. In this paper, we show an avenue for aligning language models with user intent on a wide range of tasks by fine-tuning with human feedback. Starting with a set of labeler-written prompts and prompts submitted through the OpenAI API, we collect a dataset of labeler demonstrations

- 最早由OpenAI研究团队提出，并用于训练OpenAI的InstructGPT模型；
- 效果惊艳：根据OpenAI相关论文说明，基于RLHF训练的InstructGPT模型，在仅拥有1.3B参数量的情况下，输出效果已经开源和GPT-3 175B模型媲美。这充分说明了RLHF方法的实践效果；
- 目前Hugging Face、PyTorch和微软研究院都有提供RLHF实现方法，其中微软研究院的DeepSpeed Chat是目前最高效稳定的端到端RFHL训练系统；

RLHF训练流程

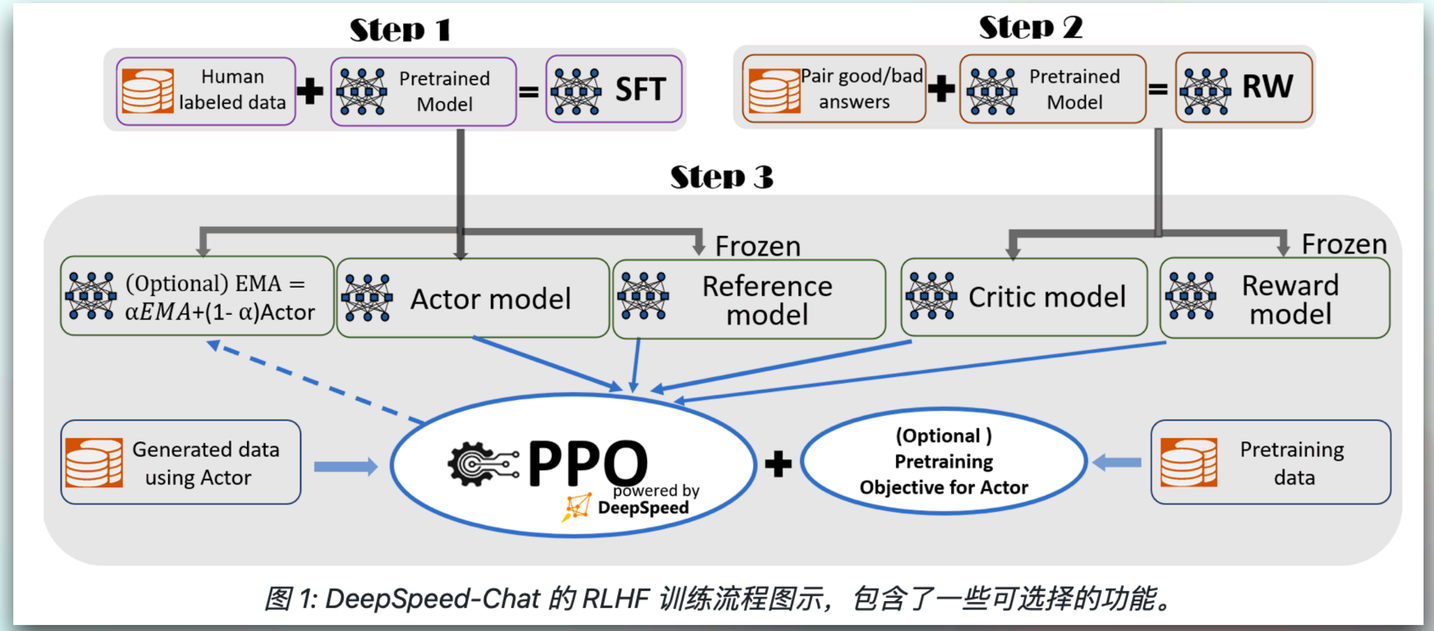


图 1: DeepSpeed-Chat 的 RLHF 训练流程图示，包含了一些可选择的功能。

- 项目地址： https://github.com/microsoft/DeepSpeed/

- **步骤1：监督微调（SFT）** —— 使用精选的人类回答来微调预训练的语言模型以应对各种查询；
- **步骤2：奖励模型微调** —— 使用一个包含人类对同一查询的多个答案打分的数据集来训练一个独立的（通常比 SFT 小的）奖励模型（RW）；
- **步骤3：RLHF 训练** —— 利用 Proximal Policy Optimization（PPO）算法，根据 RW 模型的奖励反馈进一步微调 SFT 模型。