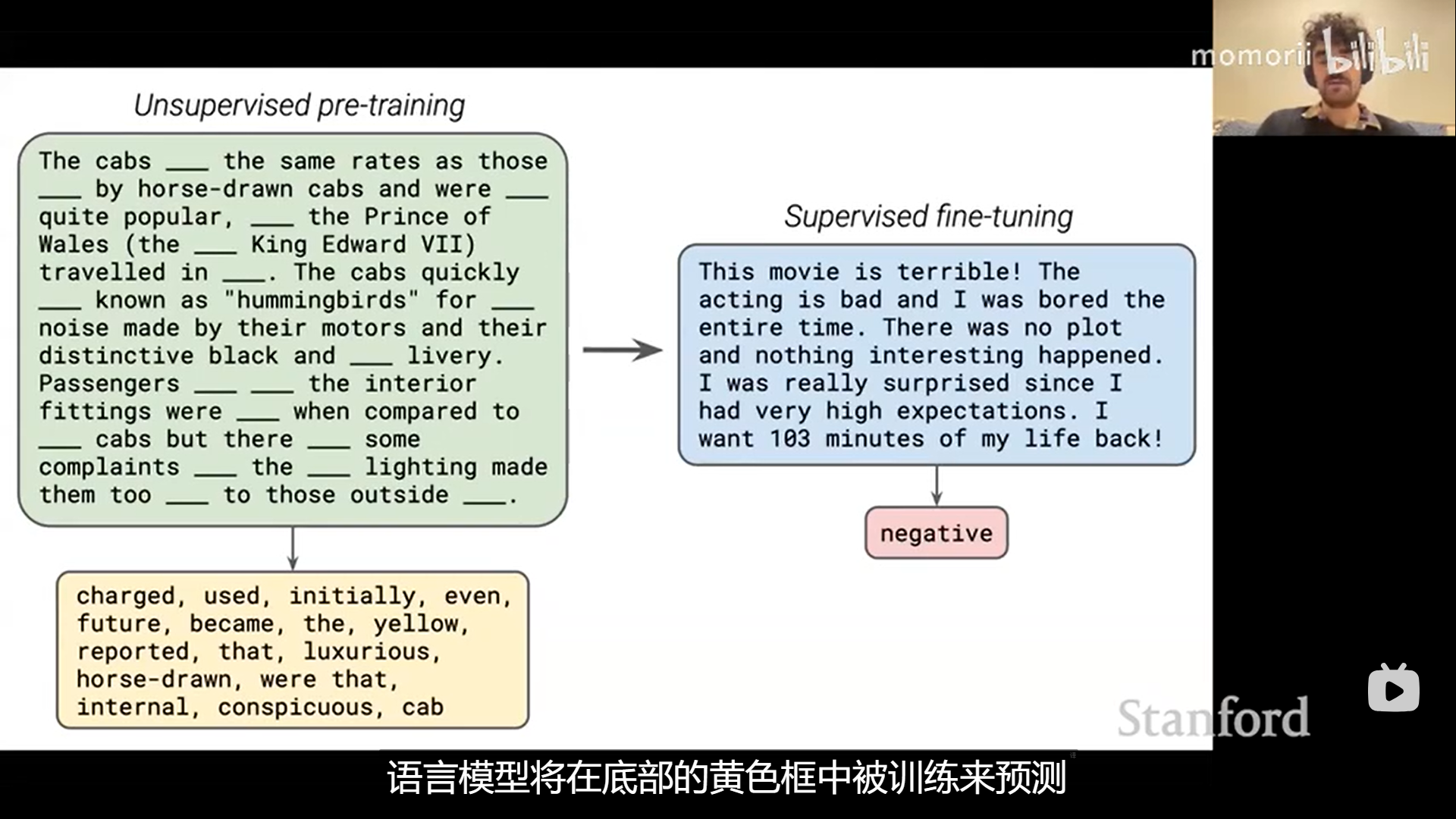
、

这张PPT展示了两种深度学习中的训练过程：**非监督预训练**（Unsupervised Pre-training）和**监督微调**（Supervised Fine-tuning）。

1. **非监督预训练**：
   * 在左边的绿色框中，你看到的是一种叫做“遮掩语言模型”（Masked Language Model）的训练方法。这种方法的核心思想是在一段文本中随机掩盖（或遮掩）一些单词（在例子中，掩盖的单词用空白表示），然后让模型根据上下文来预测这些被掩盖的单词。
   * 举例来说，句子“the cabs were quite popular”中的某些词可能被掩盖，变成“the cabs \_\_\_ quite popular”，然后模型的任务就是预测这个空白处应该填什么词。
   * 这种方法是不依赖标注数据的，也就是非监督的，因为不需要知道句子中的单词之前是什么，模型通过不断尝试去填空，从而学习词汇与句子结构之间的关系。
2. **监督微调**：
   * 在非监督预训练之后，模型已经掌握了一些关于语言的基本知识（如词汇、语法和语境等）。接下来就是通过“监督学习”来进行微调，以适应特定任务。
   * 在右边的蓝色框中，展示了一个具体的应用任务：情感分析（Sentiment Analysis）。模型被给定一段电影评论，并且需要判断这段评论的情感（如“negative”或“positive”）。
   * 通过大量带有情感标签的文本数据进行监督学习，模型会逐渐学会识别文本中表达的情感。因为模型已经通过非监督预训练学习了丰富的语言知识，所以在进行监督微调时可以更快更好地完成任务。

**总结与原因**：

* 非监督预训练让模型在没有标注数据的情况下，先学习语言的基本结构和语义，这使得模型具备了很强的语言理解能力。
* 监督微调则利用标注数据，将模型在特定任务上的性能进一步优化。这种先非监督预训练再监督微调的方法可以充分利用无监督数据的大量优势，同时也利用了有监督数据的精准性，从而在很多自然语言处理任务中都能取得优异的效果。

这样的方法被广泛应用在诸如BERT、GPT等现代自然语言处理模型中，是提升模型表现的关键技术之一。

“指令微调”（Instruction Tuning）和“微调”（Fine-tuning）都是在预训练模型的基础上进行进一步优化的技术，但它们在目标、方法和应用场景上有所不同。

**1. 微调（Fine-tuning）**

**定义**： 微调指的是在一个已经经过预训练的模型（通常是一个大型语言模型，如BERT、GPT等）上，通过使用某个特定任务的数据集对模型进行进一步训练，从而使模型在该特定任务上表现更好。

**特点**：

* **目标**：专门优化模型在某个具体任务（如情感分析、命名实体识别、机器翻译等）上的表现。
* **方法**：模型会通过标注好的数据集（带有明确的输入和期望的输出）进行训练。调整模型的参数，使其能够更好地完成这个任务。
* **应用场景**：通常应用于有明确任务的场景，如医疗文本分类、法律文档处理、客服系统等。

**示例**：

* 一个预训练好的BERT模型可以通过情感分析数据集进行微调，使得它在情感分类任务中表现更佳。

**2. 指令微调（Instruction Tuning）**

**定义**： 指令微调是一种通过提供具体的指令或提示来引导模型生成期望输出的技术。它的目的是使模型更好地理解和执行用户的指令。

**特点**：

* **目标**：提高模型对自然语言指令的理解和执行能力，使得模型能够在广泛的任务中响应不同形式的指令。
* **方法**：使用大量的指令数据集（包括多种任务和格式）来训练模型，使其能够理解和执行复杂的、多样化的指令。这些指令可能包括问题解答、文本生成、摘要、翻译等多种类型。
* **应用场景**：主要应用于对话系统、智能助手和需要多任务执行的场景，如ChatGPT、Siri、Google Assistant等。

**示例**：

* 一个经过指令微调的模型可能被训练得足够灵活，可以回答用户提出的各种问题，如“总结这篇文章的主要观点”或“翻译这段文字成法语”等。

**区别总结**

* **适用范围**：微调通常专注于特定任务，而指令微调旨在让模型理解并处理广泛的任务和指令。
* **数据集类型**：微调通常使用特定任务的数据集，而指令微调使用多样化的指令集。
* **灵活性**：微调后的模型一般在特定任务上表现出色，但可能在其他任务上表现不佳。指令微调的模型则更注重通用性，能够理解并执行多种类型的指令。

总之，指令微调是一种更通用的微调方法，旨在增强模型的多任务能力，而传统的微调则是针对特定任务进行优化。

**指令微调**（Instruction Tuning）是一种通过让模型学习如何理解和执行各种自然语言指令的技术。它的目标是使模型能够处理多种任务，而不需要为每个任务单独进行微调。

**详细讲解**

**1. 传统微调的局限性**

在传统微调中，我们通常会将一个预训练好的模型拿来针对特定任务进行微调。例如，如果你想让模型能够进行情感分析，你需要使用情感分析的数据集对模型进行训练。经过微调后，模型在情感分析任务上表现很好，但如果你想让它做别的事情（例如翻译），就需要重新微调模型，使用翻译的数据集。

**2. 指令微调的出现**

指令微调解决了这个问题。它通过一种更通用的方法来训练模型，使得模型能够理解各种不同的指令，并根据这些指令产生正确的输出。

指令微调通常使用多样化的指令数据集，这些数据集涵盖了不同类型的任务。例如：

* **翻译**：将“Translate the following sentence to French: 'I love programming.'”转化为“J'aime programmer.”
* **摘要**：将“Summarize the following article: ...”转化为简短的文章摘要。
* **问题解答**：将“Who is the president of the United States?”转化为具体的答案。
* **文本生成**：将“Write a short story about a talking cat”转化为一个简短的故事。

**3. 指令微调的过程**

在指令微调中，模型会接触到大量的多样化任务的指令和相应的输出。这使得模型不仅学习了如何执行特定的任务，还学习了如何理解和处理各种不同的自然语言指令。这种广泛的训练让模型能够适应更多的任务，并根据用户提供的自然语言指令做出准确的回应。

**通俗易懂的例子**

**想象你是一个厨师，而模型是一个学徒。**

* **传统微调**：你教你的学徒如何做一道特定的菜，比如说“煎蛋”。你给他很多关于煎蛋的训练（如何打鸡蛋、怎么煎、什么时候翻面等等）。最后，他会煎蛋了，但如果你突然让他去做“炒饭”，他就不知道该怎么办了。
* **指令微调**：相反，你教这个学徒如何理解和执行不同的指令。你教他说，如果有人说“煎个蛋”，他就按煎蛋的步骤来做；如果有人说“做炒饭”，他就根据你教的炒饭步骤去做。这个学徒学会了如何根据不同的指令来做出不同的菜，而不是只会煎蛋。

指令微调就像是教会模型如何“做各种菜”，而不是只会做一种菜。当你给它一个新指令时，它可以根据指令的内容选择合适的策略，灵活地执行任务。

**应用场景**

* **智能助手**：如Siri、Alexa、Google Assistant，它们可以处理用户的各种指令，例如播放音乐、设置提醒、回答问题等。它们就是通过指令微调来理解和响应不同指令的。
* **多任务对话系统**：如ChatGPT，它能够处理从编程问题到日常聊天的各种任务，正是因为它经过了指令微调。

**总结**

指令微调让模型变得更加智能和通用，它能够根据自然语言指令处理广泛的任务，不需要为每个任务都单独训练，这大大提高了模型的实用性和灵活性。

**1. 全量微调（Full Fine-Tuning）**

**定义**：

* 全量微调指的是对预训练模型中的所有参数进行调整，使其在特定任务上表现更佳。

**方法**：

* 在全量微调中，所有的模型参数（通常有数亿甚至数百亿个参数）都被重新训练。具体地，使用任务特定的数据集（如情感分析数据集）对模型进行训练，更新模型的所有权重，使其能够更好地完成该任务。

**优点**：

* 最大程度地优化模型性能，适合对特定任务要求非常高的场景。

**缺点**：

* 计算和存储资源要求很高，尤其是对于大型模型。需要保存新的模型参数，这在资源有限的情况下可能不现实。
* 每个任务都需要一套新的权重，这对于多任务场景或部署在设备上的场景不够灵活。

**2. P-Tuning**

**定义**：

* P-Tuning是一种基于**软提示**（Soft Prompting）的方法，它通过在输入文本前或后添加可学习的“提示词”来微调模型，而不是直接调整模型的参数。

**方法**：

* 具体做法是在输入序列前面或后面添加一些特殊的可学习的标记（也称为“提示词”或“虚拟标记”）。这些标记的表示可以通过训练来优化，使模型在特定任务上表现更好。
* 这里，“提示词”是一个嵌入向量，而不是实际的文本单词。

**优点**：

* 只需要学习少量的参数（即提示词的嵌入向量），所以相比全量微调更为高效。
* 适合低资源环境，并且易于在不同任务之间切换。

**缺点**：

* 性能可能不如全量微调，尤其是在某些复杂任务上。

**3. Prompt Tuning**

**定义**：

* Prompt Tuning是基于P-Tuning的扩展，主要用于更大规模的预训练模型（如GPT-3）。它仅微调特定任务的提示词，而保持模型的其余部分不变。

**方法**：

* 与P-Tuning类似，Prompt Tuning通过引入可学习的提示词来调整模型的表现，但只微调提示部分，而不调整模型的其他参数。
* 这种方法主要针对非常大的语言模型，允许它们在不同任务上有更好的表现，而无需重新训练整个模型。

**优点**：

* 高效，仅需要调整很少的参数即可适应不同任务，尤其适合大型语言模型。
* 极大地减少了计算成本和存储需求。

**缺点**：

* 在某些任务中，效果可能不如全量微调，尤其是在模型规模较小时。

**4. LoRA（Low-Rank Adaptation）**

**定义**：

* LoRA是一种通过对模型权重矩阵进行低秩分解来实现参数高效微调的方法。

**方法**：

* LoRA的核心思想是在不改变原始模型权重矩阵的基础上，通过低秩分解添加一个低秩矩阵，使得模型可以适应新的任务。
* 原始模型的权重保持不变，LoRA只对新增的低秩矩阵进行训练。这意味着微调后的权重是原始权重与低秩矩阵的线性组合。

**优点**：

* 只需存储低秩矩阵，而不是整个模型参数，因此大大减少了存储和计算的需求。
* 能够在不影响原始模型性能的情况下进行任务特定的优化。

**缺点**：

* 需要一些额外的矩阵计算，可能会增加推理时的计算复杂度。
* 性能表现依赖于任务的复杂性和LoRA的实现细节。

**总结与对比**

* **全量微调**：对所有参数进行训练，性能最佳，但资源消耗大。
* **P-Tuning**：通过软提示词微调，效率较高，但性能稍逊。
* **Prompt Tuning**：进一步优化P-Tuning，专为大型模型设计，极大减少了计算成本。
* **LoRA**：通过低秩矩阵进行参数高效微调，存储和计算需求低，但需要额外的矩阵运算。

这些方法在微调预训练模型时各有优缺点，选择哪种方法取决于具体的应用场景、资源限制和性能需求。

**P-Tuning** 是一种通过在输入文本前或后添加可学习的“提示词”来微调模型的方法。与传统的微调方法不同，P-Tuning 不直接调整模型的所有参数，而是通过引入额外的可学习向量来引导模型在特定任务上的表现。

**详细讲解**

**1. 背景**

在传统的微调方法中，模型的所有参数都会根据特定任务的数据集进行调整。但这种方式需要大量计算资源，并且每个任务都需要存储一套新的模型参数。为了解决这个问题，P-Tuning 引入了一种更轻量级的方法：通过在输入数据上添加可学习的提示词来引导模型的行为，而不是直接修改模型内部的参数。

**2. P-Tuning 的方法**

* **提示词的概念**：提示词（prompt）通常是一些额外的文本或标记，添加在输入文本的前面、后面或中间，用来引导模型的理解。例如，在问答系统中，提示词可以是“请回答以下问题：”，目的是让模型更好地理解接下来要回答的问题。
* **软提示词**：在 P-Tuning 中，提示词不是实际的自然语言文本，而是一些可学习的嵌入向量（soft prompts）。这些嵌入向量是模型在训练过程中优化出来的，它们被附加在输入序列之前或之后，作为额外的信息输入给模型。
* **训练过程**：在训练过程中，这些提示词的嵌入向量会被调整，以便它们能够更好地帮助模型在特定任务上进行推理。与传统的微调不同，模型本身的参数在这个过程中保持不变，只有提示词的嵌入向量会被优化。

**3. 通俗易懂的例子**

**想象你是一个司机，而模型是导航仪。**

* **传统微调**：假设你每天要跑不同的路线（如送快递、接送乘客），你每次都要对导航仪进行不同的设置（相当于模型的所有参数都要调整），这样你才能顺利完成任务。
* **P-Tuning**：现在，你只需要在每次出发前，给导航仪输入一个简短的提示，比如“今天跑市区”。导航仪会根据这个提示，自动调整自己的路线规划和导航方式，而不需要你去手动调整每一条路线的设置。

在这个例子中，“今天跑市区”就是提示词，它帮助导航仪（模型）理解当天的任务，而不需要重新设置所有的导航参数（模型参数）。提示词本身是可调整的，这样导航仪可以在多个不同的任务间灵活切换，而不需要对每个任务都进行完整的设置。

**P-Tuning 的优势**

* **高效**：相比全量微调，P-Tuning 只需要调整提示词的嵌入向量，而不需要重新训练整个模型。这使得 P-Tuning 变得更高效，适合资源有限的环境。
* **灵活**：通过引入不同的提示词，P-Tuning 可以很容易地适应多个任务，而不需要为每个任务都存储一整套模型参数。
* **可解释性**：提示词的引入可以为模型提供更多的上下文信息，使得模型在面对特定任务时表现得更符合预期。

**总结**

P-Tuning 通过引入可学习的提示词来调整模型在特定任务上的表现。这种方法避免了对整个模型进行微调，节省了大量的计算资源，并提高了模型的灵活性和适应性。通过P-Tuning，模型能够在多个任务之间快速切换，而无需为每个任务单独训练和存储模型参数。

这个方法特别适用于大型预训练模型（如GPT等），在多个任务或场景中进行微调时，P-Tuning 是一种高效且灵活的选择。

例如在问答系统中，提示词可以是“请回答以下问题：“，然后在 P-Tuning 中就会把提示词转化为一些可学习的嵌入向量（soft prompts），但是可能这些嵌入向量刚开始不太好（表现出来就是模型不太能理解问题的要求），因此训练过程中会对这部分嵌入向量进行优化，最终得到较为准确的嵌入向量，从而实际使用时我们输入类似的提示词，模型就能够很好的回答我们了，并且整个过程中模型的参数不会改变，只会改变处理提示词的参数

你提到的“处理提示词的参数”确实是模型的一部分，但在 P-Tuning 中，通常我们将这些参数视为“外部附加”的可学习嵌入向量，而不是模型核心架构中原有的参数。

**更清晰的解释：**

* **模型的原有参数**：这些是指预训练模型内部的所有权重和偏置项，它们在整个 P-Tuning 过程中保持不变。比如 GPT-3 的数百亿个参数。
* **提示词的嵌入向量（可学习参数）**：这些是指 P-Tuning 中引入的额外参数，主要用于表示提示词。虽然它们也是模型的一部分，但它们是作为外部附加的输入处理参数，不是模型原始参数的一部分。

因此，在 P-Tuning 的上下文中，当我们说“模型的参数不变”时，我们指的是模型核心架构的预训练参数保持不变，而提示词的嵌入向量（尽管也属于模型的一部分）是可以调整的。

这样区分的主要原因是为了强调 P-Tuning 不需要像全量微调那样，对模型中大量原有参数进行调整，而只需要优化少量的提示词嵌入向量，从而更高效、更灵活。

**Prompt Tuning** 是一种针对大规模预训练模型的微调方法，与 P-Tuning 有些类似，但专门设计用于处理非常大的模型，如 GPT-3。Prompt Tuning 主要通过微调\*\*提示词（prompts）\*\*的嵌入向量，而不微调模型的核心参数。

**详细讲解**

**1. 背景**

随着预训练语言模型的规模越来越大（如 GPT-3，拥有数百亿个参数），全量微调变得越来越困难，因为微调如此庞大模型的所有参数既耗时又耗资源。Prompt Tuning 提供了一种更高效的方式，通过微调提示词，使模型能够在特定任务上表现更好，而不需要调整模型的所有参数。

**2. Prompt Tuning 的方法**

* **提示词的概念**：提示词是一段预设的文本或标记，可以帮助模型理解即将进行的任务。例如，在情感分析任务中，提示词可以是“这段文本的情感是：”，引导模型去判断接下来的文本是积极、消极还是中性的。
* **软提示词（Soft Prompts）**：在 Prompt Tuning 中，提示词被表示为一组可学习的嵌入向量（soft prompts）。这些嵌入向量会被附加在输入文本的前面、后面或中间，作为模型的输入的一部分。
* **微调过程**：在训练过程中，这些软提示词的嵌入向量会被微调，以便它们能够更好地帮助模型在特定任务上做出正确的推断。这里，模型的核心参数保持不变，只有提示词的嵌入向量被优化。

**3. 通俗易懂的例子**

**想象你在一家餐厅点餐，模型是厨师。**

* **传统微调**：如果你有特定的饮食偏好（如少盐或不吃辣），每次点餐时，厨师（模型）都需要调整菜谱（模型参数）来适应你的要求。这就像在全量微调中，每个新任务都需要调整模型的所有参数。
* **Prompt Tuning**：现在，餐厅提供了一个预先设置好的点餐模板（提示词），比如“请做一道不辣的菜”。你每次只需要选择合适的模板，厨师就会按照这个模板来制作菜品，而不需要每次都从头调整菜谱。这就像在 Prompt Tuning 中，你通过提示词来引导模型完成任务，而无需调整模型的所有参数。

在这个例子中，点餐模板（提示词）被微调，使得厨师能够准确理解你的需求，而菜谱（模型参数）则保持不变。

**Prompt Tuning 的优势**

* **高效**：仅需要调整提示词的嵌入向量，而不需要调整整个模型的参数，极大减少了计算和存储的需求。
* **灵活**：通过调整不同任务的提示词，Prompt Tuning 可以快速适应不同的任务场景，而不需要为每个任务单独存储一套完整的模型参数。
* **适合大模型**：Prompt Tuning 专门设计用于处理非常大的预训练模型，如 GPT-3，这些模型的参数太多，不适合全量微调。

**总结**

Prompt Tuning 是一种专为大型预训练模型设计的微调方法，通过微调提示词的嵌入向量，而非模型的所有参数，使得模型在特定任务上能够表现更好。这种方法既高效又灵活，非常适合在不同任务之间切换，而不需要对每个任务都进行完整的微调。

通俗来说，Prompt Tuning 就像是通过预设的“点餐模板”来引导模型做出正确的决策，而无需每次都调整模型的“菜谱”。

 **Prompt Tuning** 更适合那些非常大规模的预训练模型，使用简单的提示词来进行微调，并且调整的是输入序列前面的固定嵌入向量。

 **P-Tuning** 更加灵活，适用于不同规模的模型，并允许更复杂的提示词设计，可以在输入文本的不同位置插入嵌入向量。

