大模型联邦定制：方法、实验与洞见

Yuchuan Ye、Ming Ding、Youjia Chen、Peng Cheng 和 Dusit Niyato

arXiv:2601.00526v1 [cs.LG] 2 Jan 2026

摘要——本文探讨大模型的联邦定制，并着重介绍其在联邦学习框架中面临的关键挑战。我们回顾了若干流行的大模型定制技术，包括全量微调、高效微调、提示工程、前缀调优、知识蒸馏和检索增强生成。随后，我们讨论如何在联邦学习框架中实现这些技术。此外，我们针对联邦前缀调优开展了实验。据我们所知，这是首次尝试在联邦学习设置中应用前缀调优。实验验证了该方法的可行性，其性能接近集中式方法。与另外三种联邦定制方法的进一步比较表明，该方法具有较强的竞争力、令人满意的效率和稳定的鲁棒性。

索引词——联邦学习、模型定制、大模型。

I. 引言

近年来，大模型（语言模型）在自然语言处理和计算机视觉领域展现出卓越能力。凭借数十亿个参数，它们能够捕捉复杂模式和细致表征。典型例子包括 OpenAI 的 GPT-3（175B）和 Google 的 PaLM（540B），二者都是基于 Transformer 的模型，在语言理解和推理方面树立了基准。视觉Transformer（ViT）在图像识别方面表现出色，而 CLIP 等多模态模型则能够整合文本与图像。从头开始训练此类语言模型需要极高的资源和大量时间，并且依赖海量数据集：GPT-3 使用 570 GB 的文本（约 3000 亿个词元）进行训练，而 PaLM 使用了超过 7800 亿个词元。

尽管通用语言模型展现出相当强的能力，但将其应用于专业任务时往往仍有不足。一种更可行、更有效的策略是以预训练基础语言模型为起点，随后通过任务或领域特定的定制对其进行适配，即采用一种高效的适配流程，在不从头重新训练整个模型的情况下，针对特定下游任务优化预训练模型。

常见的定制方法通常包括对所有参数进行全量微调、对部分参数进行高效微调 [1], [2], 通过提示工程优化输入提示 [3], 整合外部知识的检索增强生成（RAG） [4], 以及知识蒸馏，将知识迁移从语言模型迁移到更小的模型 [5], 以及添加输入前缀的前缀调优 [6]。

有效的语言模型定制通常依赖特定任务的敏感数据，而出于隐私和监管限制，这些数据无法集中存储。例如，多家医院可能希望利用各自的本地临床记录协作调整基础语言模型，但医疗保健法规禁止数据共享。企业尝试使用私人文档定制语言模型时，也会遇到类似挑战。除了合规方面的顾虑，将如此庞大的分布式数据集汇集到中央服务器还会产生高昂的通信开销。在此背景下，语言模型的联邦定制提供了一种切实可行且必要的解决方案。

联邦定制将联邦学习（FL）的范式应用于语言模型，使各方能够在敏感数据保持分散存储的同时开展协作训练。客户端无需集中数据，而是在本地微调特定任务或领域的组件，并仅将更新发送给中央服务器，由其汇总生成全局模型。这种方法既能满足隐私和监管限制，又能通过仅更新必要组件来降低通信和计算开销，从而为在各类机构和企业中部署语言模型提供一种可扩展、保护隐私且高效的解决方案。

本文首先讨论使用联邦学习定制基础语言模型所面临的挑战与局限。随后，我们回顾常见的定制技术，包括提示工程、全量/高效微调、前缀调优、知识蒸馏和检索增强生成，并探讨如何在联邦学习框架中有效地应用这些技术。此外，我们还开展了联邦前缀调优实验，据我们所知，这是首次尝试。最后，我们概述了推动和改进联邦语言模型定制的潜在未来研究方向。

II. 联邦语言模型的两个基本方面

本节将评估联邦学习应用于语言模型的可行性，概述其面临的挑战，并指出可能不适用的情形。

A. 基础语言模型的联邦训练

在联邦学习框架中训练基础语言模型，可以在无需数据集中化的情况下访问多个客户端的私有数据，并通过分布式学习减轻中央服务器的训练负担。然而在实践中，这面临着巨大挑战，并存在以下缺点。

Yuchuan Ye 和 Youjia Chen（通讯作者）供职于中国福州的福州大学物理与信息工程学院。Ming Ding 供职于澳大利亚新南威尔士州的 CSIRO Data61。Peng Cheng 供职于澳大利亚维多利亚州墨尔本的拉筹伯大学计算机科学与信息技术系。Dusit Niyato 供职于新加坡南洋理工大学计算与数据科学学院。

1）训练数据要求：GPT-3 等基础模型需要海量训练数据——通常多达数千亿至数万亿个词元——以学习其背后的复杂模式和含义。目前，只有大型企业或机构能够管理如此庞大且多样的数据集，因此很难通过分散的各个客户端收集到规模和种类如此丰富的数据。

此外，基础模型需要能够适用于不同情境的多样化数据集，而联邦学习通常使用各客户端特有的特定隐私数据。联邦学习中缺乏广泛且多样的数据，限制了模型学习普遍适用模式的能力，因此不适合用于训练基础模型。

2）成本和复杂性更高：语言模型庞大的参数量需要训练设备具备极强的计算能力和大容量内存。与集中式架构不同，在集中式架构中，更新可以在单个数据中心内高效管理；联邦学习要求每个客户端都训练一份完整的基础语言模型副本，给所有客户端带来很高的内存需求。此外，联邦学习要求客户端与中央服务器之间频繁进行模型聚合与分发，这会带来额外的通信成本。因此，联邦学习实际上比集中式训练成本更高。

B. 简单任务中使用联邦学习与语言模型并无必要

虽然联邦学习是一种颇具吸引力的隐私保护和去中心化训练框架，但将其应用于语言模型并非总是必要的，尤其是在简单的分类或回归任务中。语言模型旨在处理语言理解和推理等复杂任务，因此计算成本高昂。相比之下，轻量级模型通常就能有效解决简单任务，因此在这类场景中使用联邦学习与语言模型显得过度且效率低下。

例如，在分类或回归任务中，目标是对输入数据进行分类或预测数值。规模较小但针对特定任务的模型，无需语言模型带来的额外复杂性，也能有效地完成这些任务。在 [7], 中，作者发现，尽管 GPT-4 等语言模型在具有挑战性的回归数据集上能够胜过传统方法，但性能提升幅度有限。例如，在 Liver Dis- orders 数据集上，GPT-4 的平均绝对误差为 2.55，略优于梯度提升的 2.57。考虑到语言模型的成本，传统方法是此类任务更实用的选择。

此外，在 [8], 中，作者提出了一种新颖的两步技术：先使用预训练语言模型，再接入轻量级分类器。与微调语言模型相比，他们的方法在文本分类上取得了显著的性能提升，在 SST-2 数据集和 TREC 数据集上的准确率分别提高了 0.273 和 0.07。他们得出结论，语言模型微调无需进行参数更新，因为轻量级下游模型已足以完成分类任务。

III. 语言模型定制技术

本节介绍语言模型定制的关键技术，分为 i）解冻方法（全面微调和高效微调）和 ii）冻结模型方法（提示、前缀调优、检索增强生成、蒸馏）。

A. 全量微调

全量微调会更新整个语言模型，以提升其在特定任务上的表现。也就是说，在微调过程中，预训练语言模型（如 GPT-3 或 BERT）的所有参数都会通过与目标任务相关的数据集重新训练。这种方法全面调整了模型参数，因此无疑能带来出色的性能，通常优于其他语言模型定制方法 [9]。

然而，全量微调的资源消耗最高。例如，GPT-3 包含 1750 亿个参数，而新语言模型往往拥有更多参数，以提升性能。因此，微调需要大量计算资源，包括充足的图形处理单元（GPU）内存、处理能力和大型数据集。通常需要采用多GPU配置或 TPU 等专用硬件，才能高效处理这类模型的规模。

B. 高效微调

与全量微调不同，全量微调会更新语言模型的所有参数，而高效微调会冻结语言模型的大部分参数，仅专注于更新或添加模型的一小部分，通常涉及少于 5% 的参数。选择调优参数的一种直接方法是使用稀疏矩阵，仅选取模型参数中的一小部分进行微调。

另一种方法是基于适配器的调优，即在预训练语言模型 Transformer 的各层之间插入轻量级神经网络层（称为适配器） [1]。实验结果表明，尤其是在训练数据有限的情况下，这种方法能够达到与全量微调相当的性能。

[2] 提出的低秩适配（LoRA）在预训练语言模型的部分稠密层中引入小型、特定于任务的矩阵。具体而言，它将某些层的权重更新分解为两个秩更低的矩阵，从而减少可调参数的数量。这些低秩矩阵被添加到关键权重矩阵中，例如 Transformer架构自注意力层中的权重矩阵，以优化内存占用和计算开销。

较低的计算成本使其成为更易于采用的模型定制方案。即使是消费级硬件，例如高端笔记本电脑或标准云实例，也足以进行高效微调。然而，对于高度复杂的任务，其性能仍可能不及全量微调。要平衡效率与准确性，通常需要谨慎选择具体策略，例如微调参数的比例、分解矩阵的秩等。

C. 提示工程

提示工程通过精心设计的输入提示，引导预训练模型的输出。通常，输入

提示会为语言模型提供上下文指令或详细示例，旨在影响其生成能力。

一种典型方法是上下文学习/少样本学习 [3],，即在提示中加入少量与任务相关的示例，以展示期望的结果，使模型能够高效地适应新任务。例如，在文本翻译任务中，提供几个从源语言到目标语言的句子对，让模型推断翻译模式。

提示工程具有很高的灵活性，可在文本生成/翻译和问答等多种应用中快速进行实验。最重要的是，它无需进行模型训练，因此不需要额外的计算资源。

然而，提示工程受限于语言模型的词元限制，即单个提示中上下文的长度。此外，它也受模型自身能力的限制。更重要的是，在词元限制内构建有效提示并非易事——这需要深入理解任务本身以及语言模型的行为。提示的质量会直接影响语言模型的输出的相关性和准确性。

D. 前缀调优

前缀调优 [6], 是一种参数高效方法，会在输入序列中引入一个额外的可学习向量，称为“前缀”。这些前缀充当任务特定指令，在推理过程中引导语言模型的行为。

通常，这些前缀由一个相对较小、使用任务特定数据训练的神经网络进行优化。在 [6], 中，前缀优化网络首先对输入词元进行嵌入，然后应用多层感知机（MLP）等简单网络，生成与 Transformer 的键矩阵和值矩阵维度相匹配的向量。前缀长度是可调的超参数，用于控制任务特定的适配。对于 GPT 这样的语言模型，优化后的前缀向量会直接插入 Transformer 的键矩阵和值矩阵，从而调整注意力机制以适应特定任务。

前缀调优的效率很高，但其性能受前缀优化网络的规模和优化后前缀的长度限制。前缀向量的设计是影响任务性能的关键，而对于复杂任务而言，这并非易事。

E. 检索增强生成

检索增强生成（RAG） [4] 结合了基于检索的方法与生成模型的优势，以提升语言模型在特定任务上的表现。它通过从知识库中检索外部的任务专属信息，并将其补充到语言模型的输入中。通过融入检索到的相关上下文，RAG 提高了语言模型生成内容的质量、相关性和准确性，尤其是在知识密集型任务中。

检索增强生成分为两个阶段：检索和生成。在检索阶段，查询会发送到检索工具/模型，由其在外部数据库（例如，任务特定文档集合或知识图谱）中搜索相关信息。在生成阶段，检索到的信息会与查询拼接后输入生成式语言模型（如 GPT 或 BERT），以生成面向任务的输出。

RAG 使语言模型能够访问超出其参数中存储内容的更广泛知识库，从而有效增强语言模型回答复杂问题或特定领域问题的能力。另一方面，RAG 系统的性能在很大程度上取决于检索文档的质量和相关性。因此，选择合适的知识库并优化检索模型，对于获得高质量输出至关重要。

F. 知识蒸馏

知识蒸馏旨在将知识从大型预训练模型（教师模型）迁移到更小、更高效的模型（学生模型）。在这一过程中，教师模型会生成软标签（概率输出）或中间特征（隐藏层激活值），通过最小化结合学生模型自身输出与教师模型生成输出的损失函数，指导学生模型的训练。

例如，在 [5], 句子对任务中，作者将 BERT 中的知识蒸馏出来，生成一个单层双向长短期记忆网络。该双向长短期记忆网络通过最小化自身逻辑值与 BERT 逻辑值之间的均方误差进行学习，以更少的参数（少 100 倍）和更短的推理时间（快 15 倍）取得了有竞争力的结果，达到与 ELMo（另一种广泛使用的深度上下文化词表示模型）相当的性能。

知识蒸馏在语言模型定制中具有显著优势，因为它直接利用语言模型已学习的知识，从而降低资源成本。知识蒸馏面临的主要挑战是对学生模型进行精细设计，使其能够尽可能缩小教师模型与学生模型之间的性能差距。

总而言之，全量微调性能强劲，但资源消耗大。高效微调可以降低成本，但在复杂任务上的表现可能不佳。提示工程和前缀调优无需重新训练即可快速适配，不过会受到词元长度或前缀设计的限制。检索增强生成利用外部知识提升输出质量，但效果取决于检索内容的相关性。知识蒸馏能够生成更小、更高效的模型，但可能需要以一定的性能为代价。应根据任务复杂度、资源限制和性能目标来选择合适的方法。

IV. 语言模型的联邦定制

本节将探讨如何将六种语言模型定制技术整合到联邦学习框架中，并着重介绍其潜力与局限性。

A. 联邦全量微调

如图1(a)所示，联邦全量微调是一种直接的方法，它在采用全量微调的多个客户端上应用标准联邦学习技术。在每次联邦学习迭代中，每个客户端都使用其本地数据对整个语言模型进行微调，并定期将更新后的模型参数发送至中央服务器。服务器按照某种

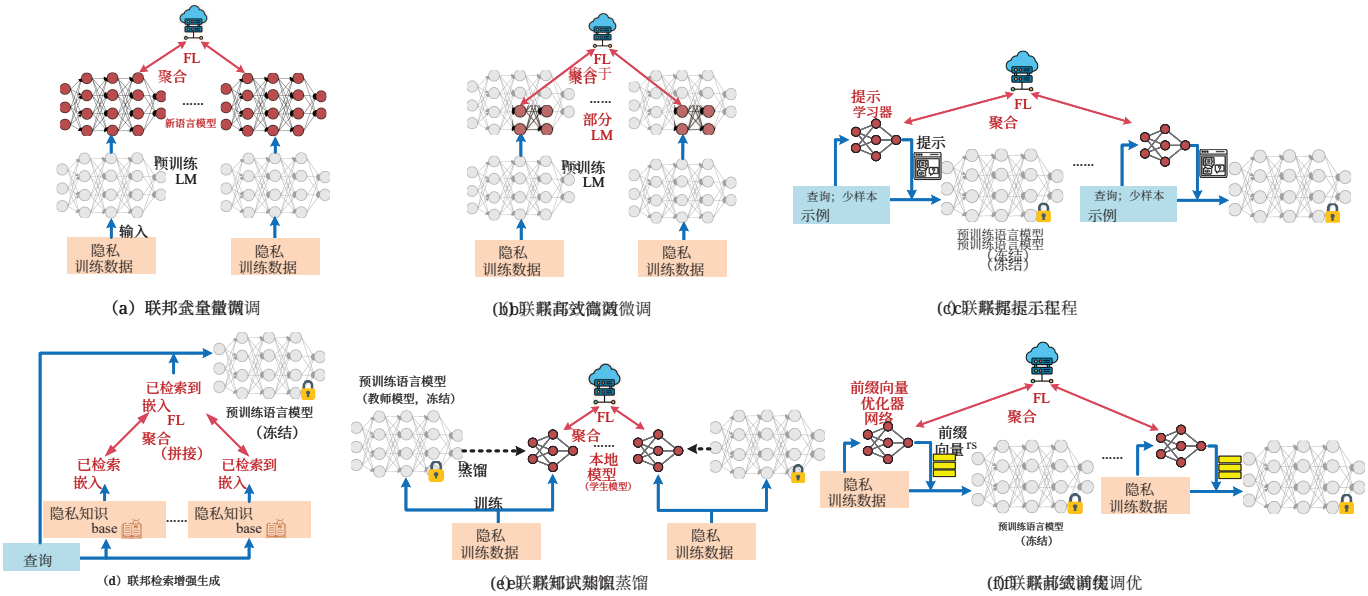


图 1. 不同联邦语言模型定制方法的框架。

策略聚合这些参数，然后将聚合模型分发给所有客户端，以便继续训练。

联邦全量微调会产生较高的计算和通信成本，因为每个客户端都必须重新训练整个语言模型，并交换大量参数更新。这种方法需要大量硬件资源，并带来沉重的通信开销，使其不适用于大多数边缘设备。此外，传输完整的模型梯度还会带来隐私风险，因为这些梯度可能泄露敏感的客户端数据。

B. 联邦高效微调

如图 1(b)所示，与需要聚合整个模型的联邦全量微调相比，联邦高效微调方法只需聚合经过微调的少量参数。因此，联邦学习的聚合成本和通信成本都显著降低。

联邦适配器微调的性能在 [10] 中进行了测试。以大型 ViT-B 模型为例，联邦适配器微调的通信成本从全量微调的 2.56 GBytes 降至 7.02 MBytes。在 CIFAR-100数据集上，联邦全量微调的准确率达到 92.09%，而联邦适配器微调可达到 88.05%。在 [11], 中，作者比较了联邦全量微调与联邦LoRA，结果表明，联邦LoRA仅需 5% 的通信成本，同时能够达到相近的准确率。

总之，联邦高效微调方法在实现与联邦全量微调相当的性能的同时，大幅降低了通信和计算成本。这些优势使联邦高效微调更适用于资源受限场景下的语言模型定制。

C. 联邦提示工程

如前所述，提示工程通过构建针对特定任务或数据上下文定制的提示来定制模型。然而，直接聚合这些提示以实现联邦提示工程并不合适。如图 1(c) 所示，联邦提示工程通常会在每个客户端部署一个提示学习器模块，然后聚合提示学习器，而非提示本身，例如 [12] 中的 PromptFL。

在 PromptFL 中，每个客户端都使用提示学习器来微调连续软提示。这些软提示是由可调向量表示的可学习嵌入。在这种方法中，每个客户端的提示学习器模型更新都会传输到中央服务器，在那里进行聚合并重新分发，以用于下一轮训练，直到收敛。性能评估表明，PromptFL 所需的通信轮次显著更少：传输 600 MB 仅需 1.4 分钟，而联邦学习传输 40 GB 则需九小时。

D. 联邦检索增强生成

如图 1(d) 所示，在联邦检索增强生成系统中，数据保留在其来源处，仅共享必要的信息，例如嵌入向量或模型更新。

一个具有代表性的实现是 C-FedRAG [13], , 它利用机密计算，在联邦检索增强生成中实现去中心化检索和嵌入聚合。每个数据提供方都会对其私有语料库进行编码，并根据用户查询返回相关嵌入向量或文本片段。中央协调器会对这些内容进行聚合和重新排序，以构建用于语言模型推理的增强查询。这种设计无需集中化数据，即可安全整合分布式知识。

另一个潜在方向是在每个客户端上训练本地检索器，并以联邦式方式聚合其参数。尽管这种方法已在

表 I 联邦语言模型定制方法对比。

方法	特征	通信成本	客户端计算成本	服务器端计算成本
联邦式 全量微调	重新训练与聚合 整个语言模型	High	High	中等
联邦式 高效微调	重新训练与聚合 语言模型的一部分	低到中等 （取决于该部分）	中等	Low
联邦式 提示工程	训练与聚合 提示学习器	Low	Low	Low
联邦式 前缀调优	训练与聚合 前缀优化器模型	Low	Low	Low
联邦式 **RAG**	聚合嵌入向量； 训练与聚合检索器	低至中等 （取决于嵌入大小）	Low	High
联邦式 知识蒸馏	训练学生模型 使用服务器端语言模型	低至中等 （取决于学生模型大小）	低至中等	Low

稠密检索的上下文中得到探索，但将此类机制整合到联邦检索增强生成系统中仍是一个开放的研究领域。

E. 联邦知识蒸馏

如图 1(e)所示，在联邦知识蒸馏中，每个客户端都部署一个本地教师模型（预训练语言模型）和一个学生模型（较小的模型）。教师模型在本地将其知识蒸馏至学生模型，随后将学生模型的更新上传至服务器。服务器聚合这些更新，生成全局学生模型。

在 [14], 中，每个客户端使用任务损失（交叉熵）和两种自适应蒸馏损失，从本地教师模型向学生模型蒸馏知识：互蒸馏（对齐软标签）和隐藏损失（对齐中间特征与注意力）。完成本地训练后，学生模型的更新会被聚合为全局模型并重新分发。在 MIND 数据集上，他们采用四层学生模型的方法达到了 71.0% 的曲线下面积，优于将压缩版 BERT 进行 FedAvg 微调所达到的 69.7%。

联邦知识蒸馏通过仅训练和传输规模较小的学生模型，降低通信和计算成本。然而，在数据异质性较强的情况下，其性能可能会受到影响，因为多样化的本地数据集会使知识整合变得复杂。此外，学生模型架构的差异可能会妨碍直接聚合，因此需要进行输出级对齐。

F. 联邦前缀调优

据我们所知，目前尚无文献专门探讨联邦前缀调优。因此，我们首次探索这一方法，并为其设计了自己的方案。

如图 1(f) 所示，为了实现联邦前缀调优，我们不直接聚合参与客户端学到的前缀向量。由于这些向量是高维且无结构的潜在参数，简单平均缺乏理论依据。因此，我们聚合前缀优化器网络的参数。该网络定义了一个结构化参数空间，从而能够在联邦学习框架中实现更有效、更稳定的聚合。所提出的联邦前缀调优在每一轮中包含以下步骤：

- 前缀向量优化器的本地训练：保持语言模型冻结，每个客户端使用私有数据来优化与前缀相关的小型神经网络，例如三层多层感知机。随后将其参数发送到中央服务器。
- 模型聚合与分发：服务器通过 FedAvg 等聚合算法聚合收到的参数，从而创建能够捕捉所有客户端数据特征的全局模型。随后，聚合后的全局模型会发送回各客户端，用于下一轮本地训练。

表 I 总结并比较了上述联邦语言模型定制方法，重点说明其计算和通信开销。

V. 实验结果

A. 实验设置

我们在 E2E 和 DART 数据集上评估表格到文本任务。E2E 数据集[1] 包含约 42K 个示例，每个示例由包含八个字段的结构化语义表示（例如“名称”“食物”“价格范围”）及其对应的自然语言描述组成。DART 数据集[2] 规模更大，约包含 82K 个示例，这些示例来源于多个数据集，包括 WikiSQL、WikiTableQuestions、E2E 和 DBpe- dia，并同时包含人工生成和自动生成的文本转换。评估时，我们采用五项官方指标——BLEU、NIST、METEOR、ROUGE-L 和 CIDEr——分数越高表示性能越好。按照 [6] 中的配置，前缀长度设为 10。

在联邦学习中，我们考虑 10 个客户端，每个客户端持有数据集的十分之一，并执行相同的下游任务（表格到文本生成）。我们同时评估独立同分布和非独立同分布的数据划分。除非另有说明，否则采用 GPT-2 Medium（GPT2-M）作为骨干语言模型，并配合一个包含 25M 个参数的多层感知机网络进行前缀优化。如果验证损失连续 3 个轮次未下降，则训练采用早停。在所有实验中，每次聚合前，每个客户端都在其私有数据上完成一个本地轮次；根据以往的理论研究 [15] 和我们的初步实验，这种设置能够确保本地更新有效地贡献于聚合模型，从而提升全局模型的性能。

[1]https://github.com/tuetschek/e2e-dataset [2] https://github.com/Yale-LILY/dart

表 II 不同模型规模下联邦前缀调优的性能，以及与集中式训练和单客户端训练的比较。

		E2E					DART				
	模型	**BLEU**	**NIST**	METEOR	ROUGE-L	CIDEr	**BLEU**	**NIST**	METEOR	ROUGE-L	CIDEr
FPT	GPT-2M (345M)	68.91±0.12	8.80±0.02	46.25±0.21	71.71±0.08	2.48±0.01	45.55±0.21	8.73±0.06	38.38±0.25	60.21±0.23	2.82±0.01
	GPT-2 L (774M)	69.98±0.15	8.82±0.13	46.43±0.14	71.45±0.19	2.51±0.11	46.48±0.19	8.86±0.26	38.79±0.18	60.92±0.23	2.89±0.03
	LLaMA-3.2 (1B)	65.29±0.10	8.38±0.08	44.67±0.17	67.96±0.15	2.32±0.02	40.24±0.21	8.18±0.09	37.37±0.18	55.82±0.22	2.48±0.02
FPT	GPT-2M (345M)	68.**91**±**0**.12	8.**80**±**0**.02	46.**25**±**0**.21	71.**71**±**0**.08	2.**48**±**0**.01	45.**55**±**0**.21	8.**73**±**0**.06	38.**38**±**0**.25	60.**21**±**0**.23	2.**82**±**0**.01
CPT	GPT-2M (345M)	69.**61**±**0**.22	8.**89**±**0**.03	47.**75**±**0**.11	72.**21**±**0**.18	2.**53**±**0**.01	46.**42**±**0**.33	8.**85**±**0**.07	38.**91**±**0**.42	61.**54**±**0**.07	2.**87**±**0**.02
仅客户端1	GPT-2M (345M)	64.09±0.51	8.18±0.01	44.15±0.06	68.21±0.12	2.28±0.01	40.78±0.36	8.06±0.12	36.45±0.50	56.82±0.18	2.47±0.02
仅客户端2	GPT-2M (345M)	66.24±0.24	8.44±0.09	45.**27**±**0**.27	68.80±0.56	2.35±0.01	41.27±0.14	8.**18**±**0**.07	36.68±0.11	56.97±0.31	2.49±0.01
仅客户端3	GPT-2M (345M)	65.79±0.12	8.49±0.08	43.43±0.59	67.68±0.51	2.22±0.03	41.28±0.26	8.11±0.05	36.51±0.28	56.76±0.03	2.47±0.01
仅客户端4	GPT-2M (345M)	63.93±0.11	8.29±0.07	43.63±0.36	67.68±0.54	2.18±0.04	40.72±0.32	7.84±0.18	35.99±0.55	57.**39**±**0**.54	2.44±0.03
仅客户端5	GPT-2M (345M)	65.65±0.09	8.47±0.07	43.86±0.55	68.19±0.85	2.23±0.01	41.17±0.09	8.02±0.07	36.19±0.45	56.94±0.08	2.47±0.01
仅客户端6	GPT-2M (345M)	63.61±0.32	8.12±0.06	42.58±0.19	68.01±0.16	2.10±0.02	40.92±0.47	7.89±0.14	36.03±0.05	57.03±0.59	2.46±0.03
仅客户端7	GPT-2M (345M)	66.21±0.53	8.45±0.07	44.36±0.38	68.51±0.05	2.31±0.01	40.39±0.39	7.75±0.06	35.91±0.36	56.47±0.54	2.41±0.01
仅客户端8	GPT-2M (345M)	67.**08**±**0**.47	8.**61**±**0**.04	44.32±0.45	69.**05**±**0**.02	2.**36**±**0**.02	39.67±0.62	7.85±0.13	35.87±0.11	56.11±0.05	2.44±0.01
仅客户端9	GPT-2M（345M）	64.76±0.45	8.32±0.13	43.41±0.11	68.11±0.08	2.25±0.01	41.35±0.25	8.04±0.09	36.**39**±**0**.09	57.22±0.64	2.**50**±**0**.01
仅客户端10	GPT-2M (345M)	65.34±0.09	8.51±0.03	44.47±0.26	67.35±0.54	2.24±0.01	41.**46**±**0**.84	8.15±0.05	36.25±0.66	57.13±0.74	2.50±0.01

表III
E2E 和 DART 数据集上不同联邦定制方法的资源比较

数据集	方法	**BLEU**	**NIST**	METEOR	ROUGE-L	CIDEr	可训练参数（百万）	**Peak** 内存（GB）	轮次停止
E2E	FPT	68.91±0.12	8.80±0.02	46.25±0.21	71.71±0.08	2.48±0.01	25	4.8	17
	FFFT	67.67±0.14	8.56±0.19	45.81±0.18	70.53±0.17	2.44±0.02	345	7.6	5
	FAT	68.23±0.16	8.63±0.19	45.95±0.14	71.90±0.17	2.44±0.02	25	4.8	6
	FKD	68.55±0.18	8.71±0.23	45.29±0.01	70.60±0.23	2.40±0.03	38.3	5.9	18
DART	FPT	45.55±0.21	8.73±0.06	38.38±0.25	60.21±0.23	2.82±0.01	25	4.8	21
	FFFT	34.19±0.28	6.50±0.25	38.70±0.19	55.00±0.17	1.95±0.01	345	7.6	6
	FAT	32.24±0.18	6.11±0.22	38.42±0.21	53.98±0.15	1.88±0.01	25	4.8	7
	FKD	31.13±0.20	5.60±0.12	26.83±0.15	43.75±0.18	1.44±0.02	38.3	5.9	20

我们在实验中比较了四种具有代表性的联邦定制方法：联邦全量微调（FFFT）、联邦适配器微调（FAT）、联邦知识蒸馏（FKD）和联邦前缀调优（FPT）。所有方法均采用经典的 FedAvg 方法聚合客户端更新。

B. 实验结果

1) 联邦前缀调优的可行性研究：为评估 FPT 的有效性，我们将其性能与以下方案进行比较：i) 在不同模型规模下采用 FPT，评估其在 GPT-2 M、GPT-2 Large（GPT-2 L）和 LLaMA-3.2 1B 上的可扩展性；ii) 使用整个数据集进行集中式前缀微调（CPT）；以及独立的仅客户端训练，即每个模型仅使用单个客户端的本地训练数据进行训练，并在共享 IID 测试集上评估。

表 II 展示了在 E2E 和 DART 数据集上的实验结果。首先，结果表明，使用 GPT-2 Large 的 FPT 在两个数据集上的得分均略高于 GPT-2 Medium，这说明在 GPT-2 系列中，模型规模增大时，该方法的表现会更好。相比之下，LLaMA-3.2 1B 的表现不如这两个 GPT-2 模型，尤其是在 DART 数据集上。这可能是因为它的预训练数据主要由自由形式自然文本构成，而非结构化的表格到文本配对数据，因此与基准任务的契合度较低。其次，FPT 的表现始终优于单独的客户端训练，证明了其有效性，能够提升整体表现。此外，FPT 与 CPT 之间的性能差距很小，进一步凸显了 FPT 的有效性。

2) 不同联邦定制方法的比较：我们比较四种联邦定制方法的性能和资源开销：FPT、FFFT、FAT 和 FKD。如前所述，FPT 训练一个拥有 25M 参数的轻量级前缀优化器网络。FFFT 更新 GPT-2 Medium 的全部 345M 参数。FAT 插入具有 25 M 可训练参数的适配器，而 FKD 则在 GPT-2 Medium 的指导下训练一个拥有 38.3M 参数的紧凑型学生模型。

如表III所示，在 E2E 数据集上，所有方法的表现相当，FPT 略胜一筹。在更具挑战性的 DART 数据集上，FPT 明显领先，FFFT 表现中等，FAT 的表现有所下降，而 FKD 的表现最差。

在效率方面，尽管 FFFT 收敛最快，但由于参数数量庞大，需要大量资源。FAT 的计算开销与 FPT 相近，同时收敛速度较快。由于采用了新的学生模型，FKD 的内存占用更高，且收敛速度较慢。FPT 的计算和通信开销适中。因此，应根据任务的具体要求和可用的设备资源来选择方法，兼顾性能与效率。

3）客户端数量的影响：我们评估了客户端数量为 20、30 和 50 时的性能（见表 IV），同时保持

表 IV 不同客户端数量下不同联邦定制方法的性能。

客户端数量	方法	E2E					DART				
		BLEU	**NIST**	METEOR	ROUGE-L	CIDEr	**BLEU**	**NIST**	METEOR	ROUGE-L	CIDEr
20	FPT	68.50±0.03	8.72±0.01	45.88±0.06	71.28±0.09	2.45±0.01	44.64±0.35	8.56±0.06	38.07±0.18	59.74±0.27	2.76±0.03
	FAT	67.82±0.21	8.58±0.12	45.73±0.17	70.85±0.11	2.43±0.02	31.78±0.21	6.11±0.22	38.39±0.18	53.63±0.18	1.72±0.02
	FKD	53.52±0.30	7.52±0.08	35.36±0.25	58.25±0.19	1.20±0.03	22.69±0.41	4.51±0.18	21.97±0.26	35.09±0.23	0.79±0.02
30	FPT	68.12±0.32	8.68±0.01	45.59±0.13	70.72±0.22	2.41±0.01	44.41±0.05	8.50±0.01	37.86±0.06	59.59±0.11	2.74±0.01
	FAT	67.46±0.22	8.53±0.09	45.57±0.20	70.07±0.13	2.40±0.02	32.14±0.20	6.24±0.16	38.31±0.22	53.85±0.21	1.72±0.02
	FKD	53.09±0.29	7.40±0.07	34.42±0.27	57.66±0.20	1.31±0.03	19.05±0.42	4.17±0.17	19.21±0.25	33.64±0.22	0.61±0.02
50	FPT	67.53±0.21	8.58±0.02	44.63±0.13	69.36±0.14	2.35±0.01	42.73±0.03	8.13±0.64	36.76±0.03	58.31±0.10	2.58±0.01
	FAT	66.56±0.22	8.43±0.06	45.19±0.16	69.48±0.18	2.37±0.02	29.87±0.26	5.89±0.21	37.56±0.22	48.93±0.16	1.62±0.02
	FKD	52.24±0.31	7.42±0.08	35.12±0.26	58.07±0.21	1.14±0.03	17.46±0.44	3.79±0.19	17.74±0.24	31.95±0.21	0.50±0.02

表 V 非独立同分布设置下不同联邦定制方法的性能。

设置	方法	E2E					DART				
		BLEU	**NIST**	METEOR	ROUGE-L	CIDEr	**BLEU**	**NIST**	METEOR	ROUGE-L	CIDEr
非独立同分布（80%）	FPT	65.79±0.40	8.41±0.06	43.35±0.96	68.67±0.57	2.25±0.07	44.36±0.34	8.49±0.08	37.87±0.16	59.59±0.25	2.74±0.02
	FAT	64.15±0.25	8.21±0.05	44.20±0.30	67.85±0.40	2.23±0.03	23.10±0.40	4.00±0.10	36.00±0.35	44.20±0.30	0.60±0.04
	FKD	67.55±0.28	8.56±0.05	42.96±0.34	68.64±0.38	2.16±0.03	29.08±0.40	5.19±0.15	25.99±0.31	42.37±0.29	1.30±0.04
非独立同分布（60%）	FPT	66.06±0.09	8.44±0.02	43.70±0.46	68.75±0.29	2.28±0.01	44.52±0.04	8.55±0.01	37.95±0.05	59.64±0.07	2.76±0.01
	FAT	64.80±0.20	8.32±0.04	44.00±0.28	68.20±0.32	2.25±0.02	24.60±0.35	5.00±0.08	36.80±0.30	46.80±0.30	0.62±0.03
	FKD	67.76±0.24	8.54±0.04	45.81±0.22	70.25±0.29	2.43±0.02	30.02±0.32	5.49±0.13	26.44±0.29	43.17±0.27	1.37±0.04
非独立同分布(40%)	FPT	67.29±0.39	8.55±0.01	44.02±0.64	69.43±0.25	2.32±0.08	44.81±0.05	8.57±0.14	38.04±0.04	59.83±0.05	2.77±0.01
	FAT	66.20±0.25	8.40±0.05	44.50±0.30	69.00±0.35	2.28±0.03	25.50±0.38	5.10±0.10	37.50±0.32	47.50±0.35	0.70±0.04
	FKD	68.27±0.22	8.64±0.05	45.98±0.20	70.75±0.27	2.42±0.02	30.74±0.41	5.54±0.15	26.06±0.30	43.26±0.32	1.48±0.05

数据集总量不变，并在客户端之间平均分配。随着客户端数量增加，每个客户端所分配的数据量减少，导致所有方法的性能均有所下降。

值得注意的是，FPT 在两个数据集上的性能下降幅度都是最小的，FAT 表现出中等程度的下降，而 FKD 的性能退化最为严重，尤其是在 DART 数据集上。这是由它们不同的底层设计所致。FPT 聚合前缀优化器的参数，同时保持预训练语言模型冻结，从而利用预训练语言模型强大的泛化能力来推动最终性能。相比之下，FKD 聚合的是一个全新的学生模型，而非冻结的骨干网络，因此对本地训练的质量更加敏感。

此外，先进的聚合算法或许有助于减少这种性能下降。我们使用 FedProx 算法对 FPT 进行的实验表明，在不同客户端数量下，与 FedAvg 相比，E2E 上的性能提升约为 0.2–0.8 个百分点，DART 上的性能提升约为 0.4–1.0 个百分点。

4) 非独立同分布数据的影响：在 E2E 数据集上，我们通过为客户端分配不同的饮食偏好来模拟非独立同分布数据，即 Client1 的数据中有 x% 是意大利菜，而 Client2 的数据中有 x % 是法国菜。对于 DART 数据集，我们按来源（WikiSQL、WikiTableQuestions、E2E、DBpe-dia）对数据进行分类，其中每个客户端的数据中有 x% 来自某个来源，其余则从其他来源抽样。我们设置 x = 80%、60%、40%，其中 x 越大，表示非独立同分布程度越强。所有评估结果均在共享 IID 测试集上报告，用于衡量全局聚合模型的性能。

根据表 V，在 E2E 数据集上，非独立同分布设置下的三种方法表现相近，其中 FAT 略逊一筹。在 DART 数据集上，非独立同分布设置下 FPT 的表现明显优于 FAT 和 FKD。比较这两个数据集上的表现后，我们发现 FPT 是最稳定的方法，而 FKD 对任务较为敏感。

此外，随着非独立同分布数据程度的增加，所有方法在两个数据集上的性能都会下降。在 E2E 上，当非独立同分布程度从 40% 上升至 80% 时，性能降幅均约为 3 个百分点。在 DART 上，FPT 受到的影响最小（0.6 个百分点），而 FAT 受到的影响显著更大（12 个百分点），FKD 则受到中等程度的影响（3 个百分点）。这些结果凸显了 FPT 相对较强的鲁棒性。

对于 FPT，性能下降主要源于异构数据下客户端之间前缀向量不对齐，这一点体现在本地与聚合前缀向量之间的余弦相似度上。在 DART 上，随着非独立同分布程度从 40% 上升到 80%，相似度略有下降（从 0.842 降至 0.834），这表明前缀向量仍基本保持对齐，也解释了 FPT 的稳健性。这种稳定性可归因于 LM 强大的泛化能力，以及前缀优化器捕捉客户端之间共享的任务级模式的能力。

此外，先进的聚合算法也能发挥作用。在我们的测试中，在 80% 非独立同分布场景下，与 FedAvg 相比，FedProx 在 E2E 上平均提升 2%，在 DART 上平均提升 1.5%。

VI. 未来研究方向

A. 多任务或多模态场景下的联邦定制

语言模型能够支持多种任务和模态，这既带来了机遇，也为联邦定制带来了挑战。在此类场景中，模型必须适应每个客户端的特定任务和模态，同时全局模型还需在这些任务和模态上保持泛化能力。潜在策略包括基于任务的迁移学习、有选择地共享参数，以及协同设计客户端的特征适配模块与服务器的对齐模块。

B. 隐私增强型联邦定制

在语言模型的联邦定制中，确保隐私至关重要，因为客户端特定任务会用到敏感数据。可将差分隐私融入联邦检索增强生成、高效微调和提示工程等方法，通过向本地数据或更新添加校准噪声，限制单个数据点的影响并降低数据泄露风险。然而，这需要在隐私与模型准确性之间取得平衡，因为噪声可能会影响输出质量。

C. 语言模型的轻量级联邦定制

大多数现有的联邦定制方法都要求每个客户端存储完整模型，这对于移动电话等资源受限设备并不现实。一种有前景的方向是将参数高效定制与并行处理或卸载策略相结合，例如采用张量并行和流水线并行，或在边缘设备与云端之间进行拆分学习。

七、结论

总之，我们的研究表明，联邦定制是将语言模型部署于联邦学习中的一种可行用例。通过指出联邦基础语言模型训练的缺点，例如对数据和资源需求较高，我们强调了采用提示工程、全量/高效微调、前缀调优、知识蒸馏和检索增强生成等技术对语言模型进行联邦定制的有效性。在表格到文本生成任务上开展的实验验证了所提出的FPT框架的可行性。与另外三种联邦定制方法的进一步比较表明，该框架具有竞争力的性能、令人满意的效率和稳定的鲁棒性。未来研究可致力于提升联邦定制的可扩展性和实用性，并使其适用于更多样化的任务和部署场景。

参考文献

[1] R. He, L. Liu, H. Ye, Q. Tan, B. Ding, L. Cheng, J. Low, L. Bing, and L. Si, 《基于适配器的调优对预训练语言模型适配的有效性》，载于第59届计算语言学协会年会暨第11届自然语言处理国际联合会议论文集（ACL/IJCNLP），2021年，第2208–2222页。

[2] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen, 《LoRA：大语言模型的低秩适配》，载于第十届国际学习表征会议（ICLR），2022年，第1–26页。[3] Z. Shi, J. Wei, Z. Xu, and Y. Liang, 《为什么较大的语言模型会以不同方式进行上下文学习？》，载于第四十一届国际机器学习会议（ICML），2024年，第1–51页。[4] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, and et al., 《面向知识密集型自然语言处理任务的检索增强生成》，载于神经信息处理系统进展（NeurIPS 2020），2020年，第9459–9474页。[5] R. Tang, Y. Lu, L. Liu, L. Mou, O. Vechtomova, and J. Lin, 《将任务专属知识从BERT蒸馏到简单神经网络》，arXiv:1903.12136，2019年，访问日期：2025年5月30日。 [在线]。网址：https://arxiv.org/abs/1903 12136 。X. L. Li and P. Liang, 《前缀调优：优[6]化用于生成的连续提示》，载于第59届计算语言学协会年会暨第11届自然语言处理国际联合会议论文集（ACL/IJCNLP），2021年，第4582–4597页。[7] R. Vacareanu, V. A. Negru, V. Suciu, and M. Surdeanu, 《从词语到数字：提供上下文示例后，你的大语言模型其实暗中具备出色的回归能力》，载于第一届语言建模会议，2024年，第1–55页。[8] S. Pawar, N. Ramrakhiyani, A. Sinha, M. Apte, and G. Palshikar, 《既然可以判别，为何还要生成？一种使用语言模型进行文本分类的新方法》，载于计算语言学协会论文集：EACL 2024，2024年，第1099–1114页。[9] J. Liu, C. Sha, and X. Peng, 《预训练代码模型参数高效微调方法的实证研究》，载于2023年第38届IEEE/ACM自动化软件工程国际会议（ASE），2023年，第397–408页。[10] G. Sun, U. Khalid, M. Mendieta, P. Wang, and C. Chen, 《探索参数高效微调，以支持联邦学习中的基础模型》，载于2024年IEEE大数据国际会议（Big Data），2024年，第8015–8024页。[11] S. Babakniya, A. R. Elkordy, Y. H. Ezzeldin, Q. Liu, K. B. Song, M. El-Khamy, and S. Avestimehr, 《SLoRA：语言模型的联邦参数高效微调》，arXiv:2308.06522，2023年，访问日期：5月30, 2025 [ ] 2308 06522 。在线。[12]网址：https://arxiv.org/abs/。T. Guo, S. Guo, J. Wang, X. Tang, and W. Xu, 《PromptFL：让联邦参与方协同学习提示而非模型——基础模型时代的联邦学习》，《IEEE移动计算汇刊》，第23卷，第5期，第5179–5194页，2024年。[13] P. Addison, M.-T. H. Nguyen, T. Medan, J. Shah, M. T. Manzari et al., 《C-FedRAG：机密联邦检索增强生成系统》，arXiv:2412.13163，2024年，访问日期：2025年5月30日。 [在线]。网址：https://arxiv.org/abs/2412 13163 。C.[14]Wu, F. Wu, L. Lyu, Y. Huang, and X. Xie, 《通过知识蒸馏实现高通信效率的联邦学习》，《自然·通讯》，第13卷，第1期，文章编号2032，2022年。[15] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang, 《非独立同分布数据上的FedAvg收敛性》，载于第八届国际学习表征会议（ICLR），2020年，第1–26页。