软件学报 ISSN 1000-9825, CODEN RUXUEW
2025,36(9):4110−4133 [doi: 10.13328/j.cnki.jos.007259] [CSTR: 32375.14.jos.007259]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

基于联邦学习的 BERT 模型高效训练框架*

王鑫澳, 陈 珂, 寿黎但, 骆歆远, 陈 刚

(区块链与数据安全全国重点实验室 (浙江大学), 浙江 杭州 310027)

通信作者: 陈珂, E-mail: chenk@zju.edu.cn

摘 要: 高质量的训练数据对于预训练语言模型 (PLM) 至关重要, 但许多专业领域的数据因隐私问题而无法集中收集用于模型训练. 借助联邦学习, 可以在保护数据隐私的前提下完成模型训练. 然而, 联邦学习的客户端通常资源有限, 无法完成预训练语言模型的训练. 针对这一问题进行深入研究. 首先, 明确定义在资源有限前提下完成模型训练的问题, 通过调整计算开销与通信开销来优化模型的训练效果. 其次, 介绍一种适用于联邦学习环境下的 BERT 模型高效训练框架——FedBT. 该框架旨在实现 BERT 模型在联邦学习客户端上的训练, 涵盖进一步预训练和下游任务微调两种场景. FedBT 适应不同的应用场景, 在客户端针对 BERT 模型的关键参数进行训练, 并仅将更新的参数上传至服务器进行聚合. 这种方法显著减少模型训练过程中的计算和通信成本. 最后, 在多个专业领域的数据集上进行充分的实验对比, 进一步预训练场景下, FedBT 框架可以降低客户端的训练开销与通信开销至原来的 34.31% 和 7.04%, 下游任务微调场景下, FedBT 框架可以降低客户端的训练开销与通信开销至原来的 48.26% 和 20.19%, 并且均实现同传统联邦学习训练完整模型接近的精确度.

关键词: 联邦学习; 预训练语言模型; 进一步预训练; 下游任务微调

中图法分类号: TP18

中文引用格式: 王鑫澳, 陈珂, 寿黎但, 骆歆远, 陈刚. 基于联邦学习的BERT模型高效训练框架. 软件学报, 2025, 36(9): 4110–4133. http://www.jos.org.cn/1000-9825/7259.htm

英文引用格式: Wang XA, Chen K, Shou LD, Luo XY, Chen G. Efficient Framework for BERT Model Training Based on Federated Learning. Ruan Jian Xue Bao/Journal of Software, 2025, 36(9): 4110–4133 (in Chinese). http://www.jos.org.cn/1000-9825/7259.htm

Efficient Framework for BERT Model Training Based on Federated Learning

WANG Xin-Ao, CHEN Ke, SHOU Li-Dan, LUO Xin-Yuan, CHEN Gang

(State Key Laboratory of Blockchain and Data Security (Zhejiang University), Hangzhou 310027, China)

Abstract: High-quality training data is instrumental in pre-trained language models (PLMs), yet privacy concerns often preclude the centralized collection of data from many professional domains. Federated learning offers a solution by enabling model training while safeguarding data privacy. However, the limited resources of federated learning clients pose a challenge to the training of pre-trained language models. This study addresses this issue through several steps. Firstly, it defines the problem of completing model training with limited resources and explores strategies to balance computational and communication costs for optimizing training efficiency. Secondly, it introduces an efficient federated learning framework for BERT further pre-training and fine-tuning (FedBT). FedBT facilitates the training of the BERT model on federated learning clients, encompassing both further pre-training and downstream task fine-tuning. Depending on the application context, FedBT selectively trains key parameters of the BERT model at the clients, uploading only the updated parameters to the server for aggregation. This approach significantly reduces both computational and communication overhead during training. Finally, extensive experiments are conducted on datasets from multiple professional domains. Results demonstrate that FedBT reduces client-side computational costs to 34.31% and communication costs to 7.04% during further pre-training. In downstream task fine-tuning, it reduces client-side computational costs to 48.26% and communication costs to 20.19%. The accuracy achieved in both pre-training and downstream

* 基金项目: 浙江省“尖兵”计划 (2024C01021)
收稿时间: 2024-03-20; 修改时间: 2024-05-05; 采用时间: 2024-07-25; jos 在线出版时间: 2025-01-24
CNKI 网络首发时间: 2025-01-26

task fine-tuning is comparable to traditional federated learning methods that train the entire model.

Key words: federated learning; pre-trained language model (PLM); further pre-training; downstream fine-tuning

近年来, 随着人工智能技术的迅猛发展, 人工智能模型的性能和复杂度不断提升. 从最初的简单线性模型到深度学习神经网络, 再到最近的大型预训练语言模型 (PLM), 模型的能力得到了显著增强, 可广泛应用于图像识别、自然语言处理和语音识别等多个领域[1,2]. 然而, 这些先进的人工智能模型的成功很大程度上依赖于大量高质量的数据. 数据成为训练模型的基石, 其中包含丰富信息, 使模型能够学习到复杂的模式和规律. 随着模型的不断发展, 对可靠、丰富数据的需求也随之增加. 在这一背景下, 数据的重要性愈发凸显, 保证数据的质量和数量, 同时保护其中的隐私信息, 成为人工智能领域急待解决的问题[3,4].

为了解决数据隐私性问题, 联邦学习[5]等新兴技术崭露头角, 这些技术的目标是在保护个人隐私的前提下进行模型训练. 与传统的集中式训练不同, 联邦学习将模型的训练过程分布在多个设备 (客户端) 上. 各客户端利用本地数据进行模型训练, 但不将数据传输至中央服务器 (服务端). 相反, 它们将模型的更新信息 (如参数或梯度) 发送至服务端, 服务端对这些信息进行聚合, 从而更新全局模型. 联邦学习的引入有效解决了数据隐私保护和数据中心化的问题, 同时充分利用多方数据, 提高了模型的泛化性和准确率. 在联邦学习的框架下, 数据隐私得到了有效保护, 大大增加了参与方数据共享合作的意愿和能力. 联邦学习可广泛应用于医疗、金融、交通、工业等多个领域, 为各行业带来更优越的数据管理和应用体验. 联邦学习可以帮助企业和机构更好地利用数据, 为人类社会的发展提供更有效的服务.

然而, 在当前的联邦学习训练方法中, 通常需要高昂的通信和计算资源. 客户端设备受限于有限的硬件资源和网络带宽, 一般只能提供有限的通信和计算能力. 这使得在联邦学习框架内训练复杂模型变得异常困难, 尤其是训练那些在自然语言处理任务中取得显著进展的预训练语言模型 (PLM), 如 ELMo[6], GPT[7], BERT[8]和 RoBERTa[9]等, 通常需要上万秒的预训练时间和每轮数百 MB 的通信开销. 这些模型在海量通用领域语料上进行了充分的预训练, 可以在专业领域语料上对模型进行进一步的预训练来增强专业任务上的表现[10], 或者在具体下游任务数据上微调模型来适配不同任务[11]. 然而, 这些数据通常涉及隐私信息, 无法集中收集用于模型训练. 因此, 急需一种技术将联邦学习与大型预训练语言模型的训练相结合.

传统的联邦学习任务中, 主要集中在较小的数据集上, 并使用相对较小的网络模型进行训练, 典型的例子是在 CIFAR 数据集[12]上使用 ResNet[13]网络架构进行训练. 相比之下, 基于 Transformer[14]架构模型的研究较为有限. 为了使更复杂的模型能够适应联邦学习客户端有限的硬件资源和通信带宽, 先前的研究大致可以划分为以下 3 类[15].

(1) 信息压缩: 信息压缩会使用更少的比特位来表示客户端需要传输给服务端进行聚合的梯度或者参数, 从而达到节约通信开销的目的, 经典的例子是量化技术[16−18]和稀疏化技术[19,20].

(2) 模型剪枝: 通过模型剪枝技术, 可以获得更加紧凑、轻量化的模型, 适用于在资源有限的设备上进行推断. 模型剪枝的基本思想是通过去除冗余的连接或参数来减小模型的规模, 从而降低客户端上的计算成本[21].

(3) 模型蒸馏: 模型蒸馏[22]是一种用于压缩和精简深度神经网络的技术, 旨在将大型、复杂的模型转化为小型、高效的模型, 同时保持其性能. 这一技术的背后理念是通过在训练中传递一个模型的知识 (教师模型) 给另一个模型 (学生模型), 来实现模型的压缩, 然后使用这个小模型来进行推断, 从而降低客户端上的计算成本.

尽管信息压缩在减少通信成本方面发挥了作用, 但并未解决联邦学习中的主要瓶颈, 即客户端有限的计算资源无法支持对复杂模型的训练. 同时, 虽然模型剪枝和知识蒸馏技术能够创建小型而准确的模型以加速推断过程, 但在训练大型模型方面效果相对有限. 因此, 将复杂的网络模型, 特别是基于 Transformer 架构的大型模型, 部署在联邦学习场景下进行训练仍然面临着许多需要解决的问题.

为了解决在联邦学习场景下无法高效训练大型预训练模型的问题, 本文提出了一种联邦学习场景下 BERT 模型的高效训练框架——FedBT (efficient framework for BERT model training based on federated learning). 该框架基于渐进式训练、采样映射和循环递减训练等算法, 能够根据任务分类在联邦学习的客户端上训练 BERT 模型的不同部分参数, 并仅将所更新的部分参数上传到服务端进行聚合. 这一方法有助于降低客户端的计算开销和通信开

销, 在保护用户数据隐私的前提下完成对大型预训练语言模型的训练. 同传统的信息压缩方法相比, FedBT 在进行参数传递聚合时, 通过减少需要传递的参数量来节约通信开销, 而不是对原始信息进行压缩. 这样不会丢失参数梯度的原始信息, 从而保证模型的训练效果. 同模型剪枝和蒸馏技术相比, FedBT 更关注模型的训练过程而不是推断过程. FedBT 通过创建更小的模型来辅助原始大模型的训练, 在推断阶段仍然使用原始模型进行推断, 从而最大限度保证模型的表现效果. 本文的前期探索成果工作发表在 IJCAI 2023[23]上, 研究了 FedBT 在进一步预训练场景下的算法框架. 本文在文献 [23] 的基础上, 进一步深入研究, 并提出了新技术和内容: 联邦学习全链路训练过程的数学建模与理论分析、模型在联邦学习场景进行高效微调的循环采样算法和对应框架设计、ViT 图像模型在联邦学习场景中进一步预训练的策略以及各类训练策略的对比分析实验.

本文的主要贡献如下.

(1) 本文针对联邦学习场景中资源受限的客户端难以训练大型预训练模型的问题, 设计了一种适用于联邦学习环境的高效 BERT 模型训练框架 FedBT. 该框架能够根据任务需求灵活选择 BERT 模型的不同参数进行训练, 并将更新的那部分参数发送至服务端进行聚合, 从而显著降低客户端的计算与通信成本. 在保护用户数据隐私的同时, 成功实现了对大型预训练语言模型的高效训练.

(2) 本文引入了渐进式训练算法和深层采样映射算法, 协助 FedBT 在客户端上进行进一步预训练. 同时, 循环递减训练算法则辅助 FedBT 在客户端上执行下游任务的微调. 在进一步预训练场景下, FedBT 在客户端上构建规模较小的模型, 并结合算法对服务端全局模型浅层 Transformer 的高效训练. 在下游任务微调场景下, FedBT 在客户端上结合算法, 实现对服务端全局模型的深层 Transformer 的高效训练. FedBT 在聚合过程仅将所更新参数上传到服务端进行聚合, 从而显著降低通信开销.

(3) 本文对客户端的计算和通信开销进行了理论分析, 并在预训练阶段和下游任务微调阶段都进行了实验, 验证所提出的框架 FedBT 的有效性. 在预训练阶段, FedBT 将客户端的计算和通信开销分别降低至原来的 34.31% 和 7.04%. 在下游任务微调阶段, FedBT 将客户端的计算和通信开销分别降低至原来的 48.26% 和 20.19%. 并且两种场景下的模型精度都接近传统联邦学习的水平.

本文第 1 节介绍本文提及方法所需的背景知识和主要概念. 第 2 节定义在资源有限场景下的模型训练问题. 第 3 节对本文提出的 FedBT 在进一步预训练场景下的算法框架进行详细的描述, 这部分的相关工作内容我们曾发表在 IJCAI 2023 上[23]. 第 4 节对本文提出的 FedBT 在下游任务微调场景下的算法框架进行详细的描述. 第 5 节通过实验验证本文方法的有效性. 最后第 6 节总结全文并进行展望.

1 背景知识

本文探讨的是联邦学习场景下, 在资源有限的客户端上对 BERT 模型的进一步预训练和下游任务微调, 进一步预训练是指将预训练语言模型在专业领域的语料上再次进行无标注的预训练, 增强通用模型在专业领域任务上的性能; 下游任务微调是指对预训练语言模型调整不同的分类层并进行训练, 来使模型可以解决具体的下游任务. 下面对这些相关概念和相关基本知识进行介绍.

1.1 联邦学习

联邦学习 (federated learning) 是一种分布式机器学习方法, 最早由 Google 于 2016 年提出, 并于 2017 年正式发表相关论文[5]. 作为一种分布式学习方法, 联邦学习在多个本地设备 (客户端) 上进行模型训练, 分布式更新模型的参数, 并在中央服务器上通过参数聚合获得一个全局模型. 这一方法克服了集中式学习中的数据隐私和安全问题, 使得模型能够在分散的数据集上进行训练, 而无需将数据集中存储. 联邦学习的具体训练过程如下: 在第 t 轮, 联邦服务器将上一轮的全局模型参数 w^{t-1} 广播发送给各个客户端, 每个客户端在获取初始模型参数后会在本地使用其持有的私有数据集训练模型并得到相应的本地模型参数 w_i^t, 接着将本地模型参数而不是原始训练数据发送给联邦服务器. 在从各个客户端接收到本地模型参数后, 联邦服务器对这些参数进行聚合操作得到 w^t 来更新全局模型, 再将更新后的全局模型参数广播发送给各个客户端. 上述这一过程会多轮次重复进行直到满足要求即停止, 如

达到期望的模型测试准确率或达到规定的训练轮次等. 在联邦学习中, 服务器不需要将所有本地数据收集起来合为一个数据集来进行模型训练, 而是数据持有者作为联邦学习的参与者联合训练一个机器学习模型, 和传统的模型训练方法对比, 充分保护了数据的隐私安全.

联邦服务器在更新全局模型时使用的聚合算法在联邦学习中起到了至关重要的作用, 采用不同的聚合算法得到的全局模型不同, 目前主流使用的聚合算法为联邦平均算法 FedAvg[5]. FedAvg 的思想是基于各客户端的训练数据量对其本地模型参数进行加权计算, 如公式 (1):

$$W^i = \sum_{k=1}^{N} \frac{n_i}{n} W_k^i \tag{1}$$

其中, n_k 为客户端 k 的本地训练数据量, n 为所有客户端的本地训练数据量之和, 通过数据量进行评估是联邦学习客户端贡献的经典评估方式[24], i 为第 i 轮迭代, W_k^i 是客户端 k 在第 i 轮迭代中的参数, W^i 是第 i 轮迭代的全局参数.

1.2 预训练语言模型

在自然语言处理领域, 预训练语言模型的发展取得了令人瞩目的进步, 为文本理解和语言生成任务提供了强大的工具. 传统的语言模型主要通过有监督学习从标注数据中学习, 然而在大规模数据上的性能受到一定限制. 随着时代的发展, 研究者们开始关注如何更充分地利用大规模文本数据, 并提出了一系列创新性的预训练方法. 早期的工作包括基于神经网络的语言模型, 例如 Word2Vec[25]和 GloVe[26]. 这些模型通过在大量文本数据上进行训练, 将单词映射到连续向量空间, 提高了对语言的表示能力. 然而, 它们仍然受制于上下文的限制, 难以充分捕捉句子或文档中的复杂关系. 在这一背景下, 预训练语言模型应运而生, BERT (bidirectional encoder representations from Transformers)[8]作为一个里程碑式的模型, 采用 Transformer 架构, 通过无监督学习从大规模文本数据中学到了深层次的语言表示. 与传统模型不同, BERT 在训练过程中同时考虑了上下文中的双向信息, 使其能够更好地捕捉语境中的语义关系. BERT 的成功标志着预训练语言模型迎来了崭新的时代, 为自然语言处理领域引入了全新的范式. 其创新性的双向编码方法和预训练目标极大地提升了模型在各种任务上的性能. 此后, 基于 BERT 的变体和扩展不断涌现, 进一步推动了预训练语言模型领域的研究和发展. 这一系列的进步为自然语言处理任务带来了巨大的推动力, 并在学术界和工业界引起了广泛的关注.

一些研究工作, 例如 SciBERT[27]、BioBERT[28]、FinBERT[29]已经证明使用专业领域特定的语料库, 如计算机科学、生物、医药或者金融等领域的语料对 BERT 进行进一步的预训练 (further pre-training), 可以提高模型在这些领域任务中的性能. 然而, 专业的领域数据通常存储在不同的专业机构, 由于隐私问题一般情况下无法进行集中式的收集并用于模型的训练. 联邦学习在隐私保护上具有天然的优势, 因此可以使用联邦学习来使用专业领域数据对模型进行训练[30], 然而 BERT 等大型模型通常具有巨大的参数量, 如果在联邦学习场景下训练 BERT 模型, 需要客户端拥有足够的计算资源和通信资源来更新模型参数和聚合全局模型. 然而联邦学习客户端的资源通常有限, 无法直接完成 BERT 之类预训练语言模型的训练.

1.3 轻量化模型

BERT 模型以其在自然语言处理任务中取得的显著成就而备受瞩目. 然而, 由于其庞大的参数量和高计算复杂度, 限制了在资源受限环境中的部署, 特别是在联邦学习场景下. 为了克服这一挑战, 研究人员提出了一系列 BERT 的轻量化模型, 最著名的包括 DistilBERT[31]和 TinyBERT[32]. 这些模型在保持 BERT 主要语义特征的同时, 对模型结构、参数量和计算效率等方面进行巧妙优化, 实现了在实际应用中更加灵活地平衡性能和资源消耗. 引入这些轻量化模型为 BERT 模型在更广泛的场景中的应用提供了更多可能性.

DistilBERT 是一种由 Hugging Face 公司提出的轻量化的 BERT 模型, 其灵感来源于知识蒸馏 (knowledge distillation)[22]的思想. DistilBERT 采用了知识蒸馏的技术, 将来自原始 BERT 的“教师模型”的知识传递给轻量化的“学生模型”. 在这个过程中, 学生模型被设计为一个更简化的网络结构, 以减小模型的参数数量. 通过使用教师模型的软标签 (logits), 学生模型被引导学习原始 BERT 模型的知识, 从而实现对语义信息的保留. DistilBERT 通过去除原始 BERT 中的一些复杂结构, 例如 Transformer 中的部分层和注意力头, 以实现模型结构的精简. 这种结构

精简不仅减少了模型的参数数量, 还加速了模型的训练和推理过程. 为了引导学生模型学习教师模型的知识, DistilBERT 采用了一种特殊的蒸馏损失函数, 该损失函数不仅确保了模型收敛到正确的类别, 还考虑了教师模型的输出. 这有助于学生模型更好地模拟教师模型的决策过程, 从而提高性能. DistilBERT 还引入了层融合的概念, 将原始 BERT 中的多个层合并为一个层. 这一层融合的过程通过学习来自不同层的信息, 使得模型在减小参数数量的同时仍然能够捕获输入序列的丰富语义表示.

TinyBERT 是由华为 Noah's Ark 实验室提出的一种轻量化的 BERT 模型, 旨在在保持高性能的同时减小模型的体积和计算复杂度, 采用了一系列创新性的设计来实现轻量化. 和 DistilBERT 类似, TinyBERT 也使用了知识蒸馏的方法, 它通过在一个较小的"学生模型"上学习一个大型的"教师模型"产生的软标签, 来传递原始 BERT 模型的知识. 为了减小计算复杂度, TinyBERT 引入了动态掩码长度的概念. 在训练过程中, TinyBERT 根据每个实例的输入长度动态地选择掩码的长度, 而不是使用固定的掩码长度, 这有助于降低计算量, 并使模型更适合处理不同长度的输入. TinyBERT 引入了 intra-layer 交叉注意力, 即在同一层内的不同位置之间进行注意力交互, 这有助于提高模型在保持语义表示的同时减小参数数量, 从而实现轻量化. TinyBERT 采用了隐式层交互的策略, 即通过将层间的信息隐式地传递给同一层, 从而减小了模型的深度和参数数量, 这一设计有助于实现轻量级模型结构. TinyBERT 还引入了阈值分层的概念, 即通过使用不同的阈值来分层选择要保留的注意力头, 这有助于进一步减小模型的规模, 使得模型更适用于资源受限的环境.

然而这些轻量化的模型通常以牺牲模型的精度为代价, 并且能够减少的模型规模有限, 在联邦学习的场景下进行部署仍然有一定的限制.

1.4 联邦学习中的模型高效训练

在联邦学习场景进行模型的高效训练主要可以分为使用轻量化模型和使用高效训练算法两部分, 训练预训练语言模型时的轻量化模型包括上文介绍的 DistilBERT 和 TinyBERT 等模型.

渐进式训练[33]是一种深度学习模型训练的策略, 其核心思想是通过逐步增加训练难度或复杂度来提高模型性能. 这种方法与传统的一次性训练所有任务的方式有所不同, 更注重阶段性学习和逐步模型优化. 在渐进式训练中, 模型首先在较为简单或具体的任务上进行初始化和训练, 然后逐步迭代地引入更复杂、抽象或广泛的任务, 使模型逐渐适应和学习更高层次的知识表示. 这种逐步增加任务的方式有助于模型更好地利用已学到的知识, 在新任务上表现更为出色. 在渐进式训练中, 任务被组织成层次结构, 每一层次解决一个相对简单或具体的问题. 模型首先在低层次任务上进行初始化和训练, 然后逐步迭代地引入更高层次的任务. 模型在低层次任务上学到的知识可以迁移到更高层次的任务中, 从而提高整体性能. 这种知识迁移的机制使得模型能够更好地利用已学到的信息, 减少在新任务上的学习负担.

渐进式训练最初提出是为了稳定训练过程, 如今已广泛应用于各种计算机视觉任务. ProgFed[15]尝试将渐进式训练引入联邦学习的场景下, 提出了一种名为 ProgFed 的新型联邦学习框架. 该框架通过采用渐进式训练来减少计算和通信成本, 同时保持强大的性能. 该框架采用分阶段的方式逐步增加模型的复杂度, 从而降低了每个客户端需要传输的数据量和计算量, 并提高了模型的收敛速度和性能. 此外, 该框架还引入了一种对称更新策略, 可更好地处理模型参数不均匀分布的情况. 然而, ProgFed 的框架目前仅适用于普通的神经网络模型, 并未在基于 Transformer 的模型结构上进行实验.

FedSplitBERT[33]提出了一种新型的框架, 旨在在联邦学习场景下对 BERT 模型进行下游隐私数据任务的微调, 并辅助解决数据异质性的问题. FedSplitBERT 通过将 BERT 编码器层分割成本地部分和全局部分来降低通信成本. 本地部分的参数仅由本地客户端训练, 而全局部分的参数则通过多个客户端的梯度聚合进行训练. 考虑到 BERT 的规模庞大, FedSplitBERT 还研究了一种量化方法, 以进一步减少通信成本, 并同时最小化性能损失. 此外, 它与许多现有的联邦学习算法兼容, 包括 FedAvg、FedProx 和 FedAdam. 然而, FedSplitBERT 仅适用于下游任务的微调, 而在预训练任务方面的适配性有限. 此外, 虽然该框架成功地降低了客户端与服务端之间的通信开销, 但并没有有效地减少计算开销. 对于计算资源受限的设备, FedSplitBERT 可能无法适配.

类似的, FedBERT[34]也提出了相关的工作. FedBERT 将 BERT 模型分为 Embedding 层、Transformer 层和 Head 层, 其中 Transformer 层部署在服务器上, 其他层在客户端上训练. 并且提出了基于 FedAvg 的联邦学习方法和基于分割学习的并行和顺序训练策略. 虽然 FedBERT 成功降低了客户端部分的计算开销, 但是需要客户端和服务器之间传输大量的梯度信息, 导致通信成本较高. 并且根据实验结果, 使用 FedBERT 进行训练的模型相比于集中式训练, 性能有一定的下降.

之前的研究工作指出, 神经网络模型在训练过程中呈现逐步从浅层到深层稳定的特性[35], 这一特性在以 BERT 为代表的预训练语言模型中也得到了验证. 在 BERT 的 Transformer 层中, 包含了丰富的语义信息. 具体而言, 底层 Transformer 捕捉表层特征, 中间层 Transformer 捕捉句法特征, 而顶层 Transformer 捕捉语义特征. 这表明在 BERT 模型中, 浅层 Transformer 在预训练阶段更加重要, 而深层 Transformer 在下游任务阶段更为关键[36−38]. 本文也通过实验验证了这一点, 根据这一特性, 在联邦学习的客户端上, 可以针对不同的训练任务, 选择性地在客户端训练全局模型的部分 Transformer 参数, 进而完成对全局模型的训练. 具体而言, 可以在预训练任务上重点训练模型的浅层 Transformer, 在下游任务上着眼于训练模型的深层 Transformer. 通过逐渐调整所训练的层, 在节约计算和通信开销的同时, 高效而安全地完成全局模型在隐私数据上的训练.

2 联邦模型高效训练问题

2.1 问题定义

本节形式化定义了联邦学习场景下 BERT 模型高效训练的问题, 旨在解决以下两个核心挑战: 一是在进一步预训练场景中, 充分利用资源受限客户端上的专业领域无标注语料, 对 BERT 模型进行深度优化, 以提升其在专业领域任务中的适应能力, 并通过在服务端对优化后的模型进行微调, 解决专业领域的下游任务需求; 二是在下游任务微调场景中, 借助资源受限客户端上的专业领域标注数据, 对 BERT 模型进行针对性的参数调整, 以应对服务端的特定下游任务.

在进一步预训练场景下, 训练的初始阶段, 服务端拥有一个在通用语料上经过充分预训练的 BERT 模型 M_G, 初始参数为 W_G^0; 同时拥有某个专业领域的下游任务数据 D_G, 用于微调并评估模型在专业领域的表现. 参与联邦学习的有 N 个客户端 C_k, 每个客户端上拥有存储在本地的同样专业领域无标注语料 D_k. 服务端需要使用各个资源受限客户端上的语料 D_k 对模型 M_G 进行进一步的预训练, 增强模型 M_G 在 D_G 上的表现.

在下游任务微调场景下, 训练的初始阶段, 服务端拥有一个在通用语料上经过充分预训练得到的 BERT 模型, 初始参数为 W_G^0, 同时拥有某个领域的下游任务测试数据 D_G, 仅用于评估模型在具体下游任务的表现. 参与联邦学习的有 N 个客户端 C_k, 每个客户端上拥有存储在本地的同样领域下游任务标注数据 D_k. 服务端需要使用各个资源受限客户端上的下游任务数据 D_k 对模型 M_G 进行微调, 让模型 M_G 适配服务端的具体任务, 并测试模型在数据集 D_G 上的表现.

因此, 本文所提出的 FedBT 整体的训练目标是通过客户端本地的数据对全局 BERT 模型进行训练, 增强全局 BERT 模型在特定任务上的表现效果, 同时尽量减小模型在客户端本地训练的计算开销, 以及联邦聚合过程中的通信开销.

分析客户端模型 M_k 本地训练的计算开销时, 假设训练的数据集规模相同, 硬件环境等外界因素相同, 则模型的计算开销与模型的规模, 模型需要更新的参数有关. 模型的计算开销可以形式化表示为公式 (2), 其中, $Train()$ 代表模型训练函数, 进一步预训练阶段为 MLM 训练, 下游任务微调阶段为 fine-tune 微调; L_k 为客户端模型的 Transformer 层数, 用来代表客户端的模型规模; L_k^u 代表客户端模型需要更新的 Transformer 层数量, 会影响模型的计算开销; D_k 代表客户端本地的数据集; π 代表客户端的具体参数训练策略, 不同的参数训练策略会影响模型的计算开销. 进一步预训练场景和下游任务微调场景下的训练策略分别在第 3 节和第 4 节进行详细介绍.

$$Cost_{\text{comp}} = \sum_{k=1}^{N} Train(M_k(L_k, L_k^u), D_k, \pi) \tag{2}$$

客户端进行联邦聚合过程中会产生通信开销, 客户端需要把本地更新的参数上传到服务端进行聚合, 并且下载服务端聚合后的参数用于新一轮的训练. 模型的通信开销可以形式化的表示为公式 (3), 主要由客户端模型需要更新的 Transformer 层数量影响.

$$Cost_{\text{comm}} = \sum_{k=1}^{N} Trans(M_k(L_k^u)) \tag{3}$$

训练完成之后, 服务端全局模型 M_G 通过映射使用客户端聚合得到的参数 W', 之后在服务端的数据集上测试模型的表现 (进一步预训练场景下需要先在服务端的下游任务 D_G 上进行微调, 然后在测试数据集 $D_{\text{G-test}}$ 上测试). 模型最终的表现效果可以表示为公式 (4).

$$Acc = \begin{cases} Fine\_Tune(M_G \cdot Map(W_G^0, W'), D_G) \cdot Test(D_{\text{G-test}}) \\ Test(M_G \cdot Map(W_G^0, W') \cdot D_G) \end{cases} \tag{4}$$

FedBT 框架整体的目标是想要使用客户端的数据达到尽可能好的表现效果, 同时尽量减小客户端的计算开销与通信开销, 因此可以将框架的目标表示为公式 (5), 开销的上角标 $Cost^n$ 代表将开销归一化为和模型表现效果同样的维度.

$$Target = \max(Acc - Cost_{\text{comp}}^n - Cost_{\text{comm}}^n) = \max F(L_k, L_k^u, \pi) \tag{5}$$

在联邦学习的情境下, 客户端的计算资源和通常资源通常是有限的, 因此在 FedBT 框架中, 客户端的训练开销与通信开销至少需要控制在传统联邦学习的一半以下 (传统联邦学习指的是在客户端使用完整的模型, 同时训练、更新和传输模型的全部参数). 综上所述, FedBT 框架整体需要解决的问题为: 通过调整客户端的模型规模 L_k、更新的参数数量 L_k^u 和参数训练策略 π, 减少模型在联邦学习场景下的计算开销与通信开销, 并保证模型的训练效果, 可以表示为公式 (6):

$$\begin{cases} \max F(L_k, L_k^u, \pi) \\ \text{s.t.} \begin{cases} Cost_{\text{comp}} \leqslant \dfrac{Cost_{\text{comp}}^{\text{trad}}}{2} \\ Cost_{\text{comm}} \leqslant \dfrac{Cost_{\text{comm}}^{\text{trad}}}{2} \end{cases} \end{cases} \tag{6}$$

2.2 开销分析

联邦学习场景下, 在资源受限的客户端上进行模型训练时, 需要特别关注模型的计算开销和参与联邦聚合过程中的通信开销. 本节将通过理论分析探讨 FedBT 框架下模型的计算开销与通信开销, 为了更好地进行理论分析, 本文在分析过程中设立如下假设.

(1) 为了准确评估在联邦学习迭代过程中本地模型对全局模型的影响, 本文假设所有客户端使用相同的学习率、批处理大小、句子长度以及其他超参数, 并在客户端对本地模型进行相同轮次的迭代 (epoch) 训练.

(2) 为了准确评估对客户端本地模型调整对整体开销的影响, 本文对某一个具体的客户端进行分析, 排除本地数据集的规模以及外部因素 (如硬件规格) 的影响.

在本文的研究场景下, 主要是对 BERT 模型进行专业领域语料的进一步预训练和下游任务上的微调. BERT 模型的结构主要由 Embedding 层, Transformer 层, 和输出分类层构成. 在本节的分析中, 假设 V 是词汇表大小, S 是句子长度, H 是词向量维度大小, C 是分类数, N 为多头注意力机制的数量, L 为 Transformer 层的数量. 一个本地 BERT 模型与全局 BERT 模型唯一的区别在于 Transformer 层的数量 L, 本节在分析过程中使用 L_G 和 L_k 来区分全局模型和第 k 个本地模型中 Transformer 层的数量.

根据之前的相关研究[13]可知 BERT 模型的各个部分的计算时间复杂度为:

(1) Embedding 时间复杂度为: $O((V+S)\cdot H)$;

(2) Self-attention 时间复杂度为: $O(L\cdot N\cdot S^2\cdot H^2)$;

(3) Feed forward 时间复杂度为: $O(L\cdot S^2\cdot H^2)$;

(4) Add&Norm 时间复杂度为: $O(L\cdot S\cdot H)$.

在 BERT 模型的训练过程中, 包含有前向传播和反向传播两个部分, 前向传播用于得到模型的计算结果, 反向传播用于更新模型的参数. 可以分析得到 BERT 模型计算过程中的整体时间复杂度如下.

(1) Embedding 层计算: 时间复杂度为 $O((V+S)\cdot H)$;

(2) 前向传播计算: 包括 Self-attention 计算, feed forward 计算和 Add&Norm 计算, 时间复杂度为 $O(Lf\cdot(S^2\cdot H^2+S^2\cdot H^2+S\cdot H))\approx O(Lf\cdot S^2\cdot H^2)$, 其中 Lf 为前向传播计算过程中参与计算的 Transformer 层数目;

(3) 反向更新计算: 时间复杂度同前向传播计算类似, 时间复杂度为 $O(Lb\cdot S^2\cdot H^2)$, 其中 Lb 为反向传播更新过程中参与计算的 Transformer 层数目.

因此 BERT 模型的训练更新时间复杂度可以如公式 (7) 所示, 并且根据 BERT 模型的具体数据可知 $(V+S)\cdot H\ll(Lf+Lb)\cdot S^2\cdot H^2$.

$$O((V+S)\cdot H+Lf\cdot S^2\cdot H^2+Lb\cdot S^2\cdot H^2) \tag{7}$$

在 FedBT 框架中, 服务端全局模型为 BERT-base-uncased 模型, 其中 $Lf=Lb=L_G=12$. 进一步预训练场景下, 在客户端构建的本地模型中, 可以设置模型拥有更少的 Transformer 层参数, 比如 6 层 Transformer, 因此 $Lf=L_k=6$, 同时由于每次只更新一层 Transformer 层参数, 可知 $Lb=1$. 代入公式 (7) 可知, FedBT 在使用客户端的隐私数据对模型进行进一步的预训练时, 节约了大量的计算资源. 在下游任务微调场景下, 客户端模型的规模同全局模型的规模一致, 即 $Lf=L_k=L_G=12$. 但下游微调场景下仅训练更新深层的 Transformer 参数, 相比于浅层 Transformer 参数离输出层更近, 因此反向传播需要的开销更低. 综上所述, FedBT 框架可以有效节约联邦学习场景下模型训练的计算资源.

同计算开销分析类似, 根据之前的研究[8,13], 可以得到 BERT 模型中不同部分的参数的空间复杂度如下.

(1) Embedding: $O((V+S)\cdot H)$;

(2) Transformer: $O(L\cdot(3\cdot H^2+4H\cdot(2H+1)))$;

(3) Output: $O(V+S)$.

在传统联邦学习每一轮的迭代过程中, 需要将模型更新的所有参数传输到服务端进行聚合, 将模型聚合过程中的空间复杂度表示为公式 (8), 其中 Lb 表示参与反向传播更新的 Transformer 层的数量, 因为在联邦学习过程中, 只有进行本地训练更新的参数需要上传到服务端进行聚合.

$$O((V+S)\cdot H+Lb\cdot(3\cdot H^2+4H\cdot(2H+1))+H\cdot C) \tag{8}$$

在本文提出的 FedBT 框架中, 当模型使用客户端本地的隐私数据进行进一步预训练时, 只需将 1 层 Transformer 层和输出层的参数上传到服务端进行参数更新. 下游任务微调场景下, 也仅需要将训练的深层 Transformer 参数和输出层参数上传到服务端. 相较之下, 在传统的联邦学习中, 需要上传 12 层的 Transformer 层参数, 以及 Embedding 层参数和输出层参数. 因此, 代入公式 (8) 可知, 本文的 FedBT 方法能够大幅节约联邦学习场景下的通信开销. 本文将在第 5 节的实验部分进行充分的实验来进行验证, 在同样的训练轮次下, FedBT 能够消耗更少的计算资源和通信资源达到同传统联邦学习接近的训练效果.

3 高效进一步预训练框架

3.1 框架总览

BERT 模型利用大规模通用语料库 (例如维基百科等) 进行了充分地预训练, 获得了广泛的通用领域知识, 在通用领域任务中表现卓越. 然而, 在处理专业领域任务时, BERT 模型的性能相对较一般. 一些先前的研究, 如 SciBERT[27]、BioBERT[28]、FinBERT[29]等已经证明, 使用专业领域语料库 (例如计算机科学、生物学、医药、金融等领域的语料), 对 BERT 模型进行进一步的预训练, 可以提高 BERT 模型在这些领域任务中的表现. 然而, 由于这些专业语料通常分布在各个独立的设备上, 因隐私问题无法集中收集, 这就制约了模型进行进一步的预训练.

本节提出的框架针对以下问题展开研究: 在联邦学习场景中, 充分挖掘资源受限客户端上的专业领域语料,

对 BERT 模型进行进一步预训练, 以提升其在专业领域任务中的表现, 以满足服务端专业领域下游任务的需求, 同时尽可能降低客户端的计算和通信成本.

为了解决此问题, FedBT 在进一步预训练阶段提出了如下解决方案.

(1) 在进一步预训练阶段, 客户端模型仅对浅层 Transformer 进行更新, 因为浅层 Transformer 在预训练过程中起到更为关键的作用, 这一点在第 1.4 节中已作详细探讨.

(2) 客户端构建规模更小的模型用来辅助浅层 Transformer 训练, 并且每次只更新一层 Transformer 参数和输出层参数来节约计算开销, 即 $L_k \leqslant L_G$, $L_k^u = 1$. 这是因为, 在预训练过程中语料充分, 预训练过程为无监督训练, 并且浅层 Transformer 离输出层较远, 只要最关键的参数能够得到有效更新, 就能够实现同训练完整参数接近的表现效果.

(3) 客户端在联邦聚合过程中仅传输和下载一层 Transformer 参数和输出层参数, 通过传输参数量的减少来降低通信成本.

(4) 客户端的训练策略 π 为: 通过渐进式训练算法将有限的计算资源尽可能多地用于训练更重要的浅层 Transformer, 同时兼顾深层 Transformer 的更新; 通过深层采样算法帮助浅层 Transformer 参数训练时充分获得服务端全局模型的深层知识. FedBT 框架通过两个算法的结合来保证模型训练效果.

FedBT 在进一步预训练阶段的整体训练框架如图 1 所示, 图中的 T-Layer 表示模型的 Transformer 参数, 整体的框架流程如下.

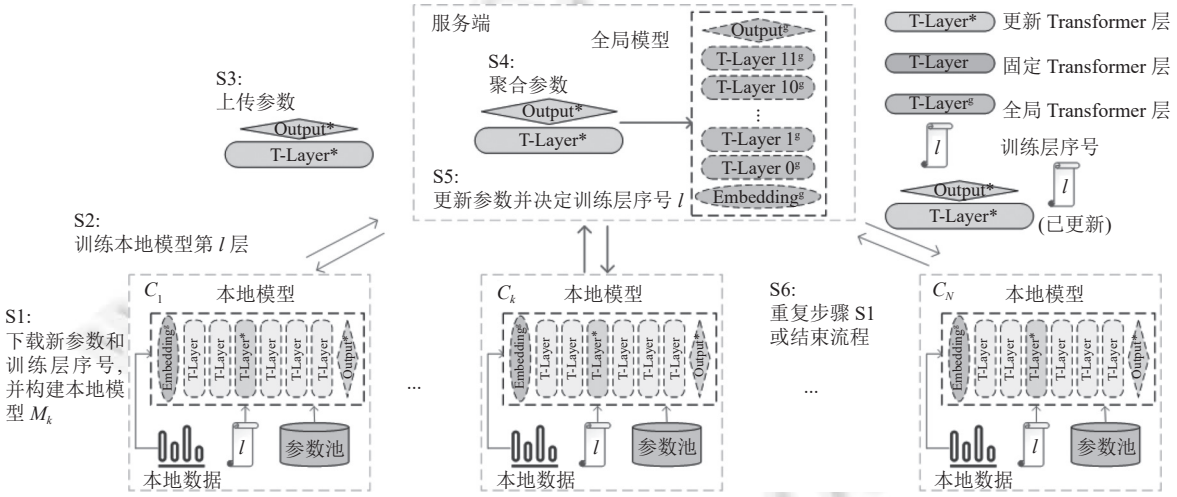


图 1 FedBT 进一步预训练框架

● S0: 联邦学习初始阶段, 客户端 C_k 从服务器获取全局模型的参数, 并将获取的参数保存在本地参数池 P_k 中.

● S1: 客户端 C_k 从服务器接收到一个索引号 l, 该索引号表示在客户端 C_k 的本地模型中要训练和更新的 Transformer 层索引号. 如果不是首轮联邦学习, 客户端还将接收到服务端下发的聚合后参数, 客户端使用此参数更新本地参数池 P_k. 之后客户端 C_k 根据本地的参数池 P_k 构建本地模型, 关于层索引号 l 确定和如何构建本地模型将在第 3.2 节详细说明.

● S2: 客户端之间并行地进行本地训练, 对本地模型的第 l 层 Transformer 和输出层参数使用本地数据 D_k 进行掩码语言模型 (MLM) 训练, 其他层 (包括嵌入层) 保持冻结, 仅用于辅助计算, 不进行更新.

● S3: 客户端 C_k 上传当前训练的第 l 层 Transformer 和输出层参数到服务端.

● S4: 服务端将所有客户端上传的参数进行聚合, 本应用场景下, 使用典型的 FedAVG 算法来得到聚合的参数, 如公式 (1) 所示, 根据客户端的数据量来决定参数聚合过程中的权重, 可以有效层处理客户端数据分布不均匀的场景.

● S5: 服务端使用聚合参数更新全局模型, 并基于第 3.2 节所述的浅层渐进式训练策略决定之后训练的 Transformer 层索引号 l.

● S6: 如果联邦学习轮次还在继续, 则重复步骤 S1, 客户端下载新的参数, 更新本地参数池, 构建新的模型用于训练, 否则结束流程.

FedBT 在进一步预训练场景下的算法如算法 1 所示.

算法 1. FedBT 进一步预训练算法.

输入: 数据集 $\{D_1,\ldots,D_N\}$, 全局模型 M_G, 初始参数为 W_G^0, 联邦学习迭代轮次 I, 本地 MLM 训练轮次 e_k, 当前训练层编号 l;

输出: 最终全局模型 M_G 的参数 W_G^I.

初始化: 每个客户端 C_k 的本地参数池 P_k; 初始训练 Transformer 层编号 $l=0$

1. //执行 I 轮联邦学习迭代
2. **for** $i=0$ **to** $I-1$ **do**
3. //各个客户端并行执行
4. **for** each client C_k **do**
5. **if** $i>0$ **then**
6. 使用服务端的 $W_G^{(i-1)}$ 更新 M_k
7. **end if**
8. //在客户端使用本地模型训练第 l 层 Transformer 层和输出层参数
9. $W_k^i \leftarrow MLM\_Train(M_k, D_k, P_k, l, e_k)$
10. 上传 W_k^i 到服务端
11. **end for**
12. //服务端聚合客户端所上传参数
13. $W_G^i \leftarrow Merge(W_1^i,\ldots,W_N^i)$
14. //根据浅层渐进式训练策略决定接下来训练的 Transformer 层编号 l
15. $l \leftarrow Next(l, i, r, I)$
16. 发送 W_G^i 到每个客户端
17. **end for**

3.2 训练策略的选择

联邦学习场景下, 本地客户端需要使用比服务端全局模型更小的模型, 并且在训练过程中需要更高效的更新模型参数. 这要求客户端本地模型在使用隐私数据进行进一步预训练时, 需要选择对服务端全局模型更重要的部分参数进行训练. 之前的工作[35−38]已经指出, 在 BERT 模型中, 浅层的 Transformer 模型会捕获语法层面的相关知识, 这些参数在预训练环节会比深层的 Transformer 参数发挥更大的作用.

本文所提出的 FedBT 在使用客户端隐私数据进行进一步预训练时, 会更关注模型的浅层 Transformer 参数, 并且只上传所更新的参数到服务端聚合. 具体来说, 在每一轮联邦学习的迭代过程中, FedBT 只会训练并传输一层 Transformer 参数和输出层参数, 其余层的参数则仅用于辅助计算, 保持不变. 受渐进式训练[39]思想的启发, 在进一步预训练阶段, FedBT 会从浅层的 Transformer 参数中逐层递增的选择参数用于训练, 由浅至深的逐步训练 Transformer 参数. 在训练过程中, 如何确定训练的 Transformer 层以及构建客户端的本地模型, 从而高效地完成模型参数的更新, 变得至关重要.

因此, 在使用专业领域的隐私数据对模型进行进一步预训练时, 需要尽可能地将训练资源用于训练浅层的

Transformer 参数. 基于此原则, FedBT 在进一步预训练阶段, 提出了一套渐进式的训练流程: 如果当前需要进行 I 轮联邦学习的迭代训练, FedBT 会将前半部分的联邦学习轮次用于训练第 0 层的 Transformer 参数, 也就是前 $\lceil I/2 \rceil$ 轮迭代时, $l=0$; 剩余联邦学习轮次 $(I-\lceil I/2 \rceil)$ 的一半用于训练第 1 层的 Transformer 参数, 也就是 $l=1$; 依次类推, 每完成剩余联邦轮次的一半时, 当前所训练的 Transformer 层索引号 l 会递增加 1, 并且 l 会比客户端本地所用的 Transformer 层数量 L_k 更小. 假如 $I=10$, 则 l 的取值为 [0, 0, 0, 0, 0, 1, 1, 1], 详细的流程如图 2 所示, 描述了如何确定训练的 Transformer 层.

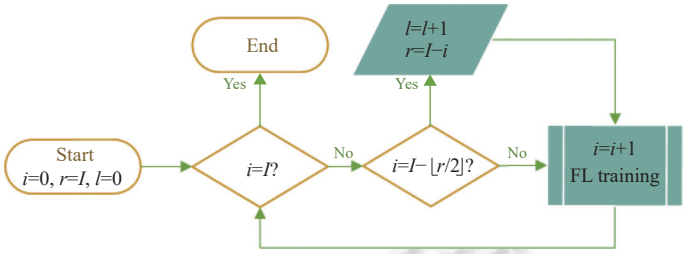


图 2 浅层渐进式训练算法流程

确定客户端 C_k 要训练的 Transformer 层索引号 l 之后, 根据 l 与本地参数池 P_k 构建客户端的本地模型 M_k. 本地模型的网络结构同全局模型保持一致, 由 Embedding 层, Transformer 层和输出层参数构成. 在构建本地模型的过程中, 假设本地模型共拥有 L_k 层 Transformer 层, 那么本地模型 Transformer 层参数的构建可以分为第 $[0,l]$ 层的 Transformer 层参数, 和第 $[l+1, L_k)$ 层的 Transformer 层参数两部分来进行构建. 本地模型的其余参数, 即 Embedding 层参数和输出层参数, 同全局模型的对应参数保持一致. 由于在每一轮联邦学习中, 本地参数池 P_k 都会根据服务端下发的最新的聚合参数进行更新, 所以本地模型的 Embedding 层参数和输出层参数同本地参数池 P_k 中的对应参数保持一致即可.

如上文所述, 在进一步预训练阶段, FedBT 会从第 0 层的 Transformer 参数, 以逐步递增的方式训练深层的 Transformer 参数. 因此, 对于本地模型 M_k 中第 $[0,L_k]$ 层的 Transformer 层参数, 都是在之前的联邦学习轮次中进行更新过的参数, 同本地参数池 P_k 中的对应参数保持一致即可, 其中第 l 层 Transformer 参数会进行训练更新. 本地模型第 $[l+1, L_k)$ 层的 Transformer 层参数, 从本地参数池 P_k 的第 $[l+1, L_G)$ 层的 Transformer 层参数中进行随机采样得到, 其中 L_k 和 L_G 分别是本地模型和全局模型的 Transformer 层数量, 如图 3 所示, 描述了如何构建客户端的本地模型.

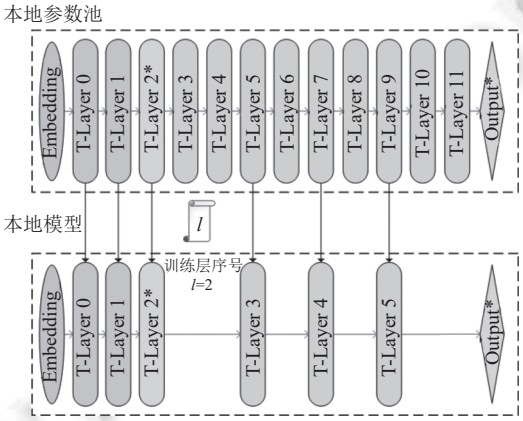


图 3 深层采样映射算法

假设当前训练的 Transformer 层索引号 $l=2$, 本地模型的 Transformer 层数量 $L_k=6$, 全局模型的 Transformer 层数量 $L_G=12$. 本地模型第 $[0,2]$ 层的 Transformer 层参数, 同本地参数池 P_k 中的对应参数保持一致. 本地模型第

[3, 6) 层的 Transformer 层参数, 从本地参数池 P_k 中的第 [3, 12) 层的 Transformer 层参数进行采样映射得到. 比如采样序列可以是 [5,7,9], 或者 [7,7,11]. 对采样序列进行升序排序之后, 根据本地参数池 P_k 中对应的 Transformer 参数构建本地模型的深层 Transformer 参数. 深层采样算法可以帮助浅层 Transformer 参数训练时充分获得服务端全局模型的深层知识, 从而保证浅层 Transformer 参数的更新符合全局模型, 达到更好的训练效果. 本文将在第 5 节实验部分详细验证这一点.

4 高效微调训练框架

4.1 框架总览

对于 BERT 模型, 除了需要使用各个客户端上的专业语料进行进一步的预训练, 还存在一种常见情境, 即使用各个客户端上的有标注数据进行下游任务的微调. 服务端拥有某个领域的少量测试数据, 需要使用 BERT 模型对这些数据进行文本分类 (CLS) 或命名实体识别 (NER) 等任务. 各个客户端同样拥有相同领域的有标注数据, 但出于隐私问题, 无法将客户端上的数据上传到服务端进行集中训练. 因此, 可以利用联邦学习将 BERT 模型在各个客户端的专业数据上进行微调, 从而解决服务端的下游任务.

本节描述的框架旨在解决以下问题: 在联邦学习场景下, 利用各个资源受限客户端上存在的专业领域标注数据, 对 BERT 模型进行微调, 来解决服务端上的具体下游任务, 同时尽量减小客户端的计算开销和通信开销. 为了解决此问题, FedBT 在下游任务微调阶段提出了如下解决方案.

(1) 客户端模型在下游任务微调阶段仅更新深层 Transformer, 这是因为深层 Transformer 在下游任务微调阶段更加重要, 本文在第 1.4 节进行了相应的探讨.

(2) 客户端在本地使用完整的模型计算更新部分深层 Transformer 参数, 通过固定其余参数不参与更新的方式来节约计算开销, 即 $L_k = L_G, L_k^u \leqslant 6$.

这是因为在下游任务微调阶段, FedBT 框架需要更新的是深层 Transformer 参数, 浅层 Transformer 参数仅用于前向传播, 无需参与反向传播, 可以更有效地节约计算资源. 而在进一步预训练阶段, 更新浅层 Transformer 参数时, 深层 Transformer 参数不仅需要前向传播, 还需进行辅助反向传播 (但并不保存梯度和更新参数). 因此客户端本地模型的 Transformer 层数 L_k 更多地影响模型表现效果, 而非计算开销. 综合考虑模型表现效果与计算资源之间的关系后, FedBT 在下游任务微调阶段选择 $L_k = L_G = 12$ 来平衡模型的表现效果与训练开销, 让模型得到更充分的训练.

同时在下游任务微调过程中, 客户端本地数据不够充分, 微调训练过程为有监督训练, 深层 Transformer 离输出层更近, 模型对数据更加敏感. 需要让模型的多层深层 Transformer 参数协同训练, 才能确保重要参数都能够得到有效更新. 本文将在第 5 节实验章节对所提出的策略进行详细的实验验证.

(3) 客户端在联邦聚合过程中仅传输和下载所训练的 Transformer 参数和输出层参数, 通过减少传输参数量来减少通信开销.

(4) 客户端的训练策略 π 为: 通过循环递减训练算法, 让 FedBT 框架下游任务微调阶段尽可能地节约计算开销和通信开销, 同时协同多层深层 Transformer 参数进行高效训练, 保证模型的训练效果.

FedBT 在高效微调阶段的整体训练框架如图 4 所示, 图中的 T-Layer 指代的是模型的 Transformer 参数, 整体的框架流程如下.

● S0: 联邦学习初始阶段, 每个客户端 C_k 从服务器获取全局模型的参数, 并将获取的参数保存在本地模型 M_k 中.

● S1: 客户端 C_k 从服务器接收到一个索引号 l, 该索引号表示在客户端 C_k 的本地模型中要训练和更新的最浅层 Transformer 层索引号; 如果不是首轮联邦学习, 客户端还将接收到服务端下发的聚合之后的参数, 客户端使用此参数更新本地模型 M_k, 关于层索引号 l 将在第 4.2 节详细说明.

● S2: 客户端之间在并行的条件下进行本地训练, 对本地模型的第 $[l, L_k)$ 层的 Transformer 参数和输出层参数

使用本地数据 D_k 进行下游任务训练, 其他层 (包括嵌入层) 保持冻结, 仅用于辅助计算, 不进行更新.

● S3: 每个客户端 C_k 上传当前训练的第 $[l, L_k)$ 层 Transformer 参数和输出层参数到服务端.

● S4: 服务端会把所有客户端上传的参数进行聚合, 本应用场景下, 使用典型的 FedAVG 算法来得到聚合的参数, 如公式 (1) 所示, 由客户端 C_k 的训练数据量决定参数聚合的权重, 用于处理客户端数据分布不均匀的场景.

● S5: 服务端使用聚合后的参数更新全局模型, 并基于第 4.2 节所述的循环递减训练策略决定之后训练的 Transformer 层最浅层索引号 l.

● S6: 如果需要继续训练, 则重复步骤 S1, 客户端下载新的参数, 否则结束当前联邦学习流程.

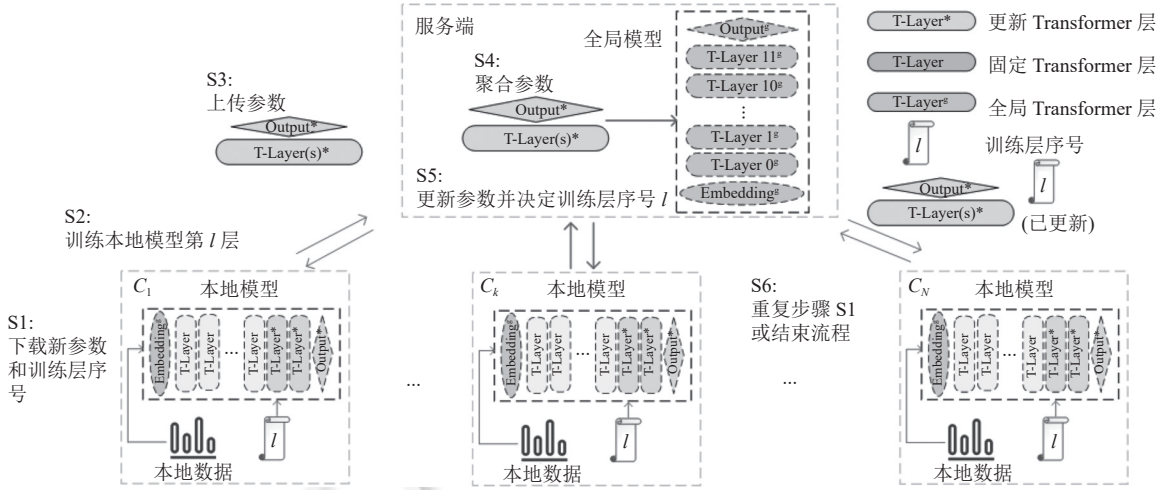


图 4 FedBT 下游任务微调框架

FedBT 在下游任务微调场景下的算法如算法 2 所示. 整体框架同算法 1 类似, 为联邦学习框架. 具体区别在于每次联邦学习训练过程中更新的参数不同以及本地模型构造的不同, 包括算法 2 中的初始化、第 8, 9 行和第 14, 15 行. 在第 4.2 节会进行详细的介绍.

算法 2. FedBT 下游任务微调算法.

输入: 数据集 $\{D_1,\ldots,D_N\}$, 全局模型 M_G, 初始参数为 W_G^0, 联邦学习迭代轮次 I, 本地下游任务训练轮次 e_k, 当前训练层编号 l, 训练循环因子 c;

输出: 最终全局模型 M_G 的参数 W_G^I.

初始化: 每个客户端 C_k 的本地参数池 P_k; 初始训练 Transformer 层编号 $l = L_G - 1$

1. //执行 I 轮联邦学习迭代
2. **for** $i = 0$ **to** $I - 1$ do
3. //各个客户端并行执行
4. **for** each client C_k do
5. **if** $i > 0$ **then**
6. 使用服务端的 W_G^{i-1} 更新 M_k
7. **end if**
8. //在客户端使用本地模型训练第 $[l, L_k)$ 层 Transformer 层和输出层参数
9. $W_k^i \leftarrow Fine\_Tune(M_k, D_k, P_k, l, L_k, e_k)$
10. 上传 W_k^i 到服务端
11. **end for**

12. //服务端聚合客户端所上传参数
13. $W_G^i \leftarrow Merge(W_1^i,\ldots,W_N^i)$
14. //根据循环递减训练策略决定接下来训练的 Transformer 层编号 l
15. $l \leftarrow Next(l, i, c, L_G)$
16. 发送 W_G^i 到每个客户端
17. **end for**

4.2 训练策略的选择

BERT 模型的深层 Transformer 参数在下游任务微调时会发挥更加重要的作用, 因此当 FedBT 框架用于下游任务时, 可以着重训练模型的深层 Transformer 参数. 同 FedBT 框架在进一步预训练场景下采用的渐进式采样映射算法不同, 在下游任务微调阶段, FedBT 框架使用的是循环递减训练算法, 如图 5 所示, 描述了如何确定训练的 Transformer 层.

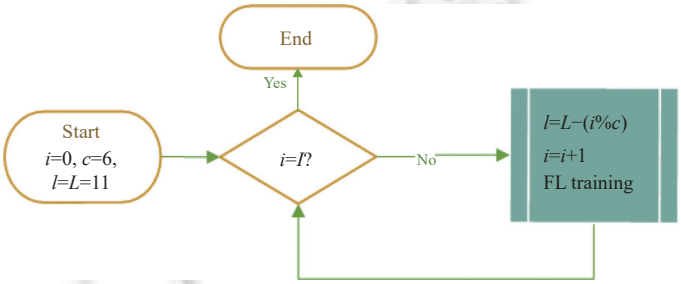


图 5 循环递减训练算法流程

图 5 中的 i 为当前训练的联邦学习迭代轮次索引号, I 为联邦学习迭代总轮数, c 为循环因子. 在微调阶段, BERT 模型的高 6 层 Transformer 要比低 6 层的 Transformer 更加重要, 因此设置循环因子 $c=6$. 初始阶段, 需要训练的最浅层 Transformer 层索引号 l 同模型最深层 Transformer 层索引号 L_G-1 一致, 为 11. 随着联邦学习的进行, 每一轮训练的最浅层 Transformer 层索引号 l 都会逐步减 1, 直到达到循环因子设置的边缘, 也就是 $l=6$ 之后, l 重置为最深层索引号 L_G-1 继续训练. 假如 $I=10$, 则 l 的取值为 $[11,10,9,8,7,6,11,10,9,8]$.

在训练过程中, 第 $[l, L_k)$ 层的 Transformer 参数和输出层参数会使用本地数据 D_k 进行下游任务训练, 其他层 (包括嵌入层) 保持冻结, 仅用于辅助计算, 不进行更新, 从而在训练过程中减小训练开销; 并且只传输所更新参数到服务端进行聚合来节约通信开销, 从而在资源有限的客户端上面完成模型的微调. 通过循环递减训练算法, 可以有效节约计算开销与通信开销, 同时协同多层深层 Transformer 参数进行高效训练, 保证模型的训练效果.

同第 3 节中介绍的一样, FedBT 框架在下游阶段同样可以使用参数池和更小的模型来微调全局模型的重要参数. 如果客户端模型 M_k 的 Transformer 层数 L_k 比全局模型的 Transformer 层数 L_G 更少, 比如 $L_k=8$, 则二者的参数之间的对应关系如图 6 所示, 描述了如何构建客户端的本地模型.

如果全局模型的 Transformer 层数 $L_G=12$, 客户端模型的 Transformer 层数 $L_k=8$, 则当服务端需要训练的最浅层 Transformer 层索引号为 $l=10$ 时, 则客户端需要训练的最浅层索引号为 $l_k=l-(L_G-L_k)=6$. 本地模型第 $[l_k, L_k)$ 层的 Transformer 参数同全局模型第 $[l, L_G)$ 层的 Transformer 参数一一对应; 本地模型第 $[0, l_k)$ 层的 Transformer 参数同全局模型第 $[0, l_k)$ 层的 Transformer 参数一一对应; 其余 Embedding 层参数和 Output 层参数一一对应. 在模型使用本地数据进行微调时, 将更新本地模型的第 $[l_k, L_k)$ 层的 Transformer 参数和 Output 层参数, 并上传服务端进行聚合. 但客户端使用更小的模型, 并不能很好的平衡模型的训练开销与表现效果, 本节将在后续实验中验证这一点.

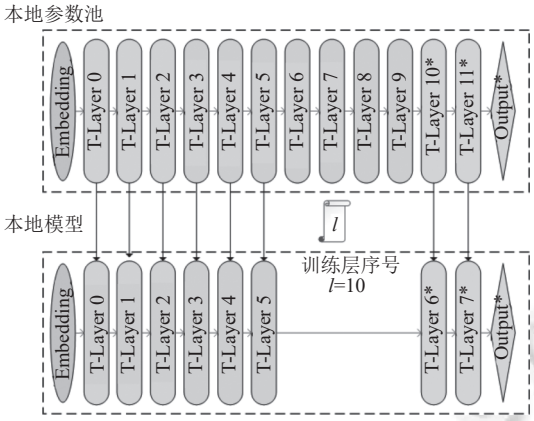


图 6 浅层对应映射

5 实验分析

5.1 实验设置

(1) 数据集

进一步预训练场景下构建客户端专业领域的语料时, 使用的是 S2ORC 数据集[40], 它是一个用于自然语言处理 (NLP) 和文本挖掘研究的通用语料库, 涵盖各个专业科学领域的论文, 包括生物、化学、计算机科学、经济、医药、物理学等多个领域的专业论文. 通过对这些专业领域论文的处理, 可以得到各个专业领域的无标注语料. 使用这些语料对预训练语言模型进行进一步的预训练, 可以增强模型在专业领域任务上的表现效果.

使用专业领域的语料对 BERT 模型进行进一步的预训练之后, 对得到的 BERT 模型在同样专业领域的下游任务上进行微调, 来衡量模型的表现. 本章所使用的下游任务为命名实体识别任务 (NER)[41]与文本分类任务 (CLS)[42−44], 具体为: 生物领域 (biology) 的命名实体识别任务 JNLPBA 数据集[45], 计算机科学领域 (CS) 的命名实体识别任务 SciERC 数据集[46], 医药领域 (medicine) 的文本分类任务 Rct-20k 数据集[10]. 其中 Rct-20k 数据集数据量过大, 本文使用了它的一个 2% 大小的子集用于下游任务的微调.

在下游任务微调场景下, 选择 BC5CDR[47], NCBI disease[48]两项命名实体识别任务, 以及 ChemProt[49], citation intent[50]两项文本分类任务进行评估模型.

(2) 联邦学习设置

为了在服务器环境下模拟联邦学习场景, 本文在服务器上创建了 6 个模拟客户端, 并为每个客户端分别构建了本地数据集. 为了模拟客户端的隐私数据, 将一定量的无标注语料或者某个具体的下游任务训练数据均匀划分成 6 个子集, 分别作为 6 个客户端本地拥有的隐私数据. 为了进行对比实验, 本文将这些数据同样进行集中式的收集, 用于在集中式场景下进行模型微调, 并将其与 FedBT 的效果进行比较.

(3) 评估指标

由于是在联邦学习资源受限的客户端上进行训练, 因此需要控制模型的训练开销与计算开销. 本文使用模型在固定大小的数据集上, 使用相同的超参数完成一定轮次的 MLM 训练或者下游任务微调的时间作为模型的计算开销, 使用模型需要上传到服务端进行更新的参数所占的存储空间大小作为模型的通信开销.

不论是进一步预训练场景还是下游任务微调场景, 最后都是在下游任务上衡量模型的表现效果. 在进行模型评估时, 使用 $F1$ 分数衡量模型在命名实体识别任务上的表现, 使用准确率衡量模型在文本分类任务上的表现.

5.2 实验结果与分析

表 1 给出了不同的模型在进一步预训练场景下的计算开销和通信开销, 以及进一步预训练结束之后在同样专

业领域的下游任务上进行微调之后的模型表现效果, 模型在微调过程中随训练轮次而变化的 $F1$ 分数或准确率如图 7 所示, 其中命名实体识别任务用 $F1$ 分数衡量, 文本分类任务用准确率衡量. 对比了一些常见的基线模型与训练方法, 在本文的实验中, 最好的实验结果会用加粗进行标识, 次好的实验结果会用下划线进行标识. 主要对比的基线模型如下.

表 1 FedBT 进一步预训练场景

模型	计算开销 (s)	通信开销 (MB)	JNLPBA ($F1$)	SciERC ($F1$)	Rct-20k (Accuracy)
BERT	N/A	N/A	0.718 1	0.622 3	0.824 8
BERT-C	13 642.88 (−)	N/A	**0.724 4**	**0.649 9**	0.812 1
BERT-FL	13 642.88 (−)	417.83 (−)	0.722 4	0.633 0	0.818 5
DistilBERT	7 553.50 (55.37%)	267.98 (64.14%)	0.711 1	0.627 8	0.828 0
TinyBERT-6	6 715.92 (49.23%)	255.57 (61.17%)	0.708 5	0.608 9	0.807 3
TinyBERT-4	**2 482.41 (18.20%)**	54.88 (13.13%)	0.673 2	0.566 4	0.799 4
FedBT (ours)	4 680.99 (34.31%)	**29.42 (7.04%)**	0.719 8	0.642 1	**0.831 2**

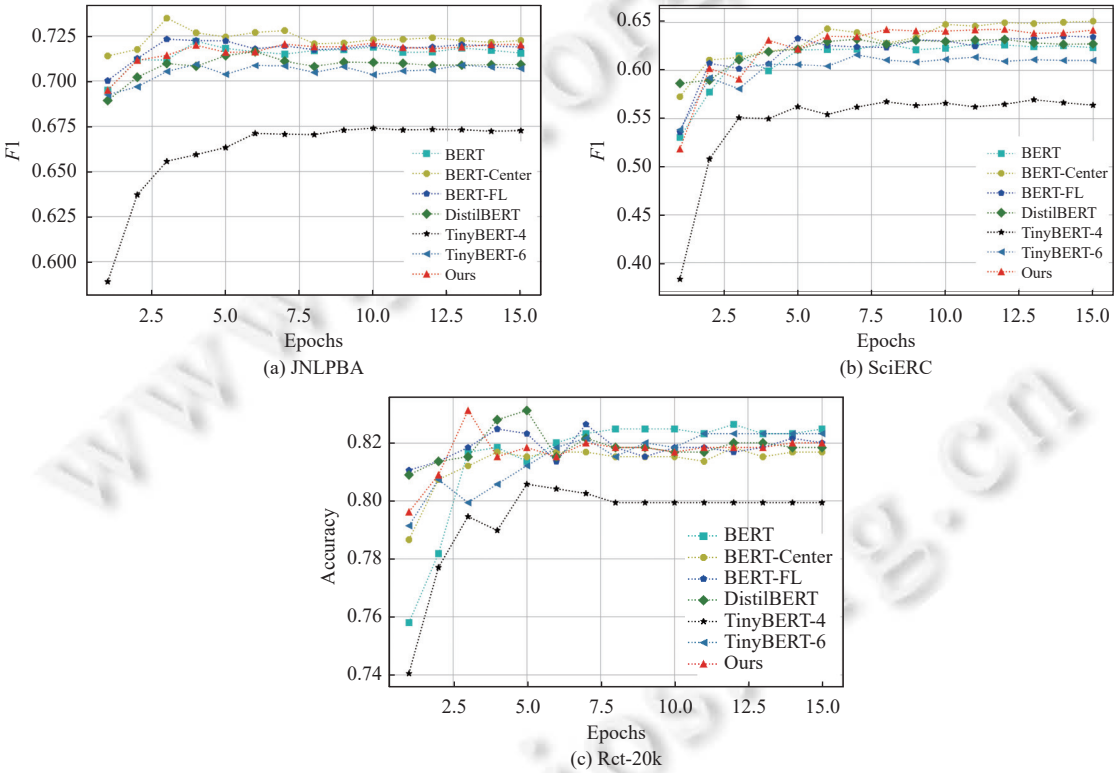


图 7 模型进一步预训练之后在下游任务上的表现

● BERT: 指通用领域语料上训练得到的 BERT-base-uncased 模型, 没有经过进一步预训练, 直接在专业领域的下游任务上进行微调.

● BERT-C: 指服务端使用集中收集的专业领域语料对 BERT-base-uncased 模型进行进一步预训练, 训练完成后在下游任务进行微调测试效果.

● BERT-FL, DistilBERT 和 TinyBERT: 指忽略客户端的资源限制, 分别使用完整的 BERT-base-uncased 模型, DistilBERT 和 TinyBERT 模型在联邦学习场景下进行进一步预训练, 并上传聚合更新模型的所有参数, 然后在训练完成后通过下游任务对模型进行微调并测试其性能, 其中 TinyBERT 分别测试了 TinyBERT-6 和 TinyBERT-4.

从表 1 可知, 在预训练阶段, FedBT 能够显著减少模型的计算开销和通信开销. 同时结合图 7 和表 1 中的模型表现效果可知, 在预训练阶段 FedBT 能够实现同传统联邦学习接近的表现效果. 相对于传统的 BERT 模型在联邦学习场景下的训练, FedBT 在客户端构建了更小的模型, 并且仅训练全局 BERT 模型的重要参数, 不会让所有的参数都进行反向传播更新, 因此 FedBT 可以有效地减少计算开销. 与轻量化的 BERT 模型在联邦学习场景下的训练相比, FedBT 在每次训练与聚合更新的过程中只训练并更新 1 层 Transformer, 而不是像传统方法一样更新所有的参数. 传统的 BERT 模型需要传输 Embedding 层参数和 12 层 Transformer 参数, 而轻量化的 BERT 模型, 除了 Embedding 层参数外, 也需要传输 4–6 层 Transformer 参数. 因此 FedBT 可以显著的降低联邦学校聚合过程中的通信开销.

因此, 与传统方法相比, FedBT 显著优化了模型的计算成本和通信成本. 尽管 TinyBERT-4 的计算成本更低, 但其模型精度却因此受到较大影响. 从表 1 可以看出, TinyBERT-4 在下游任务中的表现欠佳, 而 FedBT 在仅消耗 BERT-FL 方法 34.31% 的计算资源和 7.04% 的通信资源的情况下, 实现了与 BERT-FL 方法接近的精度.

本文对 FedBT 方法和 BERT-FL 方法进行了一致性 T 检验 (T-Test), 对两种方法分别在下游任务上进行了 10 个不同随机数种子的微调实验, 然后对二者的数据进行 T-Test 计算得到 p-value. 在 JNLPBA 数据集上的 p-value 值为 0.292 2, 在 SciERC 数据集上的 p-value 值为 0.417 3, 在 Rct-20k 数据集上的 p-value 值为 0.451 1, 均大于 0.05, 因此不拒绝零假设, 可以认为 FedBT 方法同 BERT-FL 方法的下游任务表现效果没有显著差异.

FedBT 在客户端的训练过程中构建了一个小模型, 用于对服务端全局模型的关键参数进行训练更新, 大大降低了客户端训练的计算成本; 在训练过程中, 仅更新并传输一层 Transformer 参数和输出层参数, 有效减少了联邦学习聚合阶段的通信开销; 通过采用渐进式算法, FedBT 帮助客户端将资源重点训练预训练阶段更为关键的浅层 Transformer 参数; 借助深层采样映射算法, FedBT 进一步促使客户端的浅层 Transformer 参数更充分地获取全局模型的深层 Transformer 知识, 以确保模型在进一步预训练阶段取得更好的效果. 通过这些策略, FedBT 能够在保持更好性能的同时, 消耗更少的资源, 同时相较于其他轻量化模型表现更出色.

为了进一步验证 FedBT 在不同预训练架构模型中的广泛适用性和有效性, 本文选择了 ViT (vision Transformer) 模型[51]进行了类似的实验. ViT 是一种用于计算机视觉任务的深度学习模型, 它由 Google Research 在 2020 年提出. ViT 模型的主要特点是它将 Transformer 架构成功应用于图像处理, 这是 Transformer 架构首次在自然语言处理领域之外取得显著成果. 本文将 ViT 模型在 Tiny ImageNet[52]数据集上进行进一步的预训练, 然后在 CIFAR100 数据集上测试模型的表现效果. Tiny ImageNet 是基于 ImageNet 数据集的一个子集, 但相比之下, 它的规模要小得多, 这使得它对于研究和教育目的更加实用, 因为 ImageNet 数据集非常庞大, 对于计算资源和时间的要求较高. 在进一步预训练阶段, 本文对比分析了 ViT、ViT-C、ViT-FL 和 FedBT 的训练策略, 策略设置同上文类似, FedBT 使用 ViT 模型进行进一步预训练的策略同第 3.2 节类似, 实验结果如表 2 所示. 由表 2 可知, FedBT 训练策略在使用 ViT 模型的情况下同样有效, 仅消耗传统联邦学习 37.23% 的计算开销和 9.08% 的通信开销, 就实现了同传统联邦学习接近的进一步预训练效果.

表 2 FedBT 使用 ViT 进一步预训练

模型	计算开销 (s)	通信开销 (MB)	CIFAR100 (Accuracy)
ViT	N/A	N/A	0.847 2
ViT-C	2 434.53 (−)	N/A	**0.856 1**
ViT-FL	2 434.53 (−)	346.32	0.855 8
FedBT (ours)	**906.26 (37.23%)**	**31.44 (9.08%)**	0.854 5

在下游任务微调场景下, 同上文进一步预训练场景所介绍基线模型类似, 选择 BERT-C、BERT-FL、DistilBERT、TinyBERT-6 和 TinyBERT-4 共 5 个基线模型进行比较. 在表 2 中展示了使用上述各种方法在某个具体客户端上, 对模型进行相同轮次微调的计算开销和通信开销, 以及下游任务微调之后的模型表现效果. 模型微调时随训练轮次而变化的 $F1$ 分数或准确率如图 8 所示.

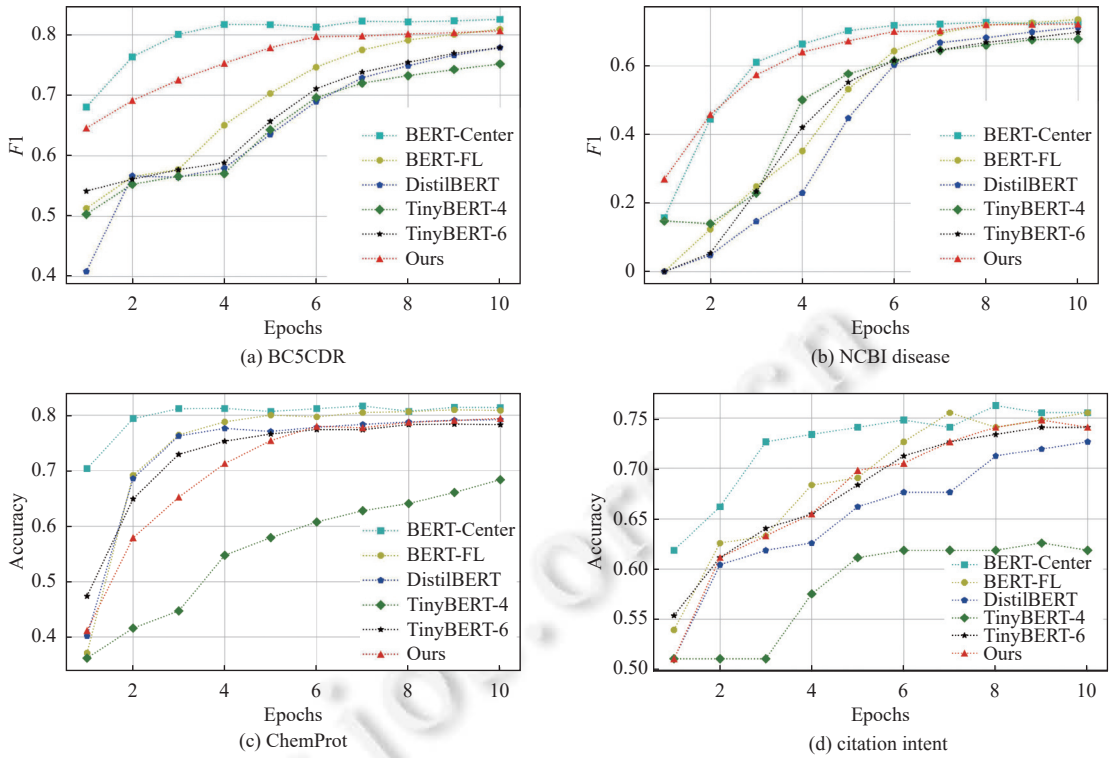


图 8 模型下游任务微调时的表现

从表 3 和图 8 可知, 在下游任务微调阶段, FedBT 显现出明显的优势, 大幅降低了模型的计算成本和通信成本. 与传统的 BERT 模型和轻量化模型在联邦学习场景下的训练方式相比, FedBT 仅在客户端训练全局 BERT 模型的部分关键参数, 并将这些参数上传至服务端进行联邦聚合, 而非上传所有参数. 因此, 以更加高效的方式完成了模型训练, 显著优化了计算和通信资源的消耗.

表 3 FedBT 下游任务微调场景

模型	计算开销 (s)	通信开销 (MB)	BC5CDR (*F*1)	NCBI disease (*F*1)	ChemProt (Accuracy)	citation intent (Accuracy)
BERT-C	347.91 (—)	N/A	**0.825 5**	0.726 6	0.812 3	0.733 8
BERT-FL	347.91 (—)	435.67 (—)	0.803 9	**0.735 1**	**0.818 1**	**0.755 4**
DistilBERT	180.63 (51.92%)	265.53 (60.95%)	0.778 9	0.711 1	0.791 9	0.726 6
TinyBERT-6	182.27 (52.39%)	265.50 (60.94%)	0.779 2	0.697 4	0.783 2	0.741 0
TinyBERT-4	**62.66 (18.01%)**	**57.05 (13.09%)**	0.752 2	0.678 1	0.683 5	0.618 7
FedBT (ours)	167.89 (48.26%)	87.95 (20.19%)	0.806 7	0.721 4	0.794 2	0.741 0

尽管 TinyBERT-4 的计算开销和通信开销较 FedBT 更为低廉, 但这是以损害模型精度为代价实现的, 从表 2 可知, TinyBERT-4 在下游任务上的表现并不理想, 相比之下, FedBT 取得了接近 BERT-FL 方法的精度, 且仅使用了 BERT-FL 方法 48.26% 的计算开销和 20.19% 的通信开销.

FedBT 在客户端的训练过程中, 专注于训练和更新模型的深层 Transformer 参数, 浅层的 Transformer 参数只参与了前向传播, 从而有效地节约了客户端训练中的计算资源. 每轮联邦学习聚合过程中, 只上传所更新部分 Transformer 参数和输出层参数, 从而节约了联邦学习聚合过程中的通信资源. 在训练过程中, FedBT 尽可能少的训练 Transformer 层参数, 并通过循环递减训练算法使模型在训练过程中能够协同深层 Transformer 参数参与训练, 保证重要参数都能够得到有效更新, 同时尽可能减少训练开销与通信开销. 通过这些策略, FedBT 能够在保持

良好性能的同时, 消耗更少的资源, 并相较于其他轻量化模型表现更出色.

5.3 参数影响分析

为了充分探讨本文提出的方法受各参数的影响状况, 分别在不同条件下进行实验来检验分析不同参数对 FedBT 模型训练效果的影响.

表 4 展示了在进一步预训练场景下的算法消融实验, 使用计算机科学与技术领域的语料对消融模型进行预训练之后, 再将消融模型在 SciERC 数据集上进行微调, 得到各种消融模型在下游任务上的 $F1$ 分数, 用于证明 FedBT 框架的有效性. 主要对比了以下几种策略.

表 4 FedBT 进一步预训练场景消融实验

模型	计算开销 (s)	通信开销 (MB)	$F1$
FedBT-All	6 753.05 (49.50%)	255.57 (61.17%)	0.639 3
FedBT\PL	4 806.06 (35.23%)	**29.42 (7.04%)**	0.640 7
FedBT\M	4 680.99 (34.31%)	**29.42 (7.04%)**	0.623 9
FedBT\S	4 680.99 (34.31%)	**29.42 (7.04%)**	0.603 8
FedBT\SP	**4 309.93 (31.59%)**	**29.42 (7.04%)**	0.641 8
FedBT (ours)	4 680.99 (34.31%)	**29.42 (7.04%)**	**0.642 1**

- FedBT-All, 此方法为在客户端的本地模型训练过程中会更新所有的参数;
- FedBT\PL, 此方法为在客户端的本地模型训练过程中只训练第 1 层的 Transformer 参数和输出层参数, 而不会渐进式的改变训练参数;
- FedBT\M, 此方法为在客户端构建本地模型的过程中, 只使用参数池中前 L_k 层的 Transformer 参数, 没有进行深层采样映射的过程;
- FedBT\S, 此方法为只有 l 改变时, 客户端才会进行深层采样映射并构建新的本地模型, 而 FedBT 会在每一轮联邦学习中都进行深层采样映射并构建新的本地模型;
- FedBT\SP, 此方法为联邦学习中, l 按照联邦学习迭代轮次进行增长, 而 FedBT 的 l 是折半递进的进行增长.

表 4 的结果充分证明了本文提出的 FedBT 框架在进一步预训练场景下的有效性, 所引入的渐进式采样映射算法能够高效地完成进一步预训练. 该算法在进一步预训练阶段更注重训练模型的浅层 Transformer 参数, 同时通过深层采样映射算法, 辅助浅层 Transformer 参数在训练过程中获取更多模型深层的知识. 在较小的模型规模下, 深层采样映射对模型训练至关重要, 可确保浅层 Transformer 参数在训练过程中获取到必要的模型整体信息, 从而提升训练效果. FedBT\SP 等策略虽然计算开销较小, 但是对于浅层 Transformer 的训练不够充分, 训练效果不够稳定同时由表可知, 当客户端训练的参数更多时, 并没有带来明显的效果提升, 但是却增加了计算开销与通信开销. 此外, 表 4 还验证了在每一轮联邦学习迭代中进行深层采样映射, 对于提升模型训练效果十分有效. 综上, 可知 FedBT 框架的每一步设置都是合理且有效的.

表 5 显示进一步预训练场景下, 不同客户端本地模型的中间 Transformer 层数 L_k 的影响, FedBT 的客户端的中间 Transformer 层数为 6, 当本地模型的 Transformer 层数为 4 层和 8 层时, 分别记作 FedBT-4 和 FedBT-8. 对 3 种策略在计算机科学与技术领域进行进一步的预训练, 然后在 SciERC 数据集上进行微调得到模型的 $F1$ 分数.

表 5 进一步预训练场景下的客户端模型规模对比实验

模型	计算开销 (s)	通信开销 (MB)	$F1$
FedBT-4	**4 309.93 (31.59%)**	**29.42 (7.04%)**	0.620 1
FedBT-8	5 616.39 (41.17%)	**29.42 (7.04%)**	0.626 7
FedBT-6 (ours)	4 680.99 (34.13%)	**29.42 (7.04%)**	**0.642 1**

由表 5 可知, 对于 FedBT 所选择的 $L_k = 6$, 在平衡整体模型开销和性能表现方面是有效的. 客户端本地模型层数较少或者较多都会对模型的训练效果产生影响, 而增加层数会显著地增加计算开销. 在进一步预训练阶段,

FedBT 选择了一个平衡的层数, 通过渐进式训练算法与深层采样映射算法, 帮助模型获得更好的表现, 同时兼顾了模型训练过程中的计算开销和联邦学习聚合过程中的通信开销.

表 6 展示了在下游任务微调场景下的消融实验, 使用 ChemProt 数据集, 在联邦场景中对消融模型进行微调, 并通过比较各种消融模型在下游任务中的准确率, 进一步验证 FedBT 框架的有效性. 主要对比了以下几种策略.

- FedBT-All, 此方法为在客户端的本地模型训练过程中会更新所有的参数.
- FedBT-Half, 此方法为在客户端的本地模型训练过程中会更新 [6, 11] 层 Transformer 和输出层参数.
- FedBT-Map, 此方法为在客户端的本地模型训练过程中会类比第 3.2 节所描述的策略进行浅层采样构建本地模型.
- FedBT\Circle, 此方法为在客户端的本地模型训练过程中只训练最后一层的 Transformer 参数和输出层参数, 而不会循环式地改变训练参数.
- FedBT\Add, 此方法为在客户端微调本地模型的过程中, 每次只训练一层 Transformer 参数和输出层参数, 而不会递增的训练更多参数.

表 6 FedBT 下游任务微调消融对比实验

模型	计算开销 (s)	通信开销 (MB)	准确率
FedBT-All	347.91 (—)	435.67 (—)	**0.8086**
FedBT-Half	219.75 (63.16%)	170.15 (39.05%)	0.806 6
FedBT-Map	167.89 (48.26%)	87.95 (20.19%)	0.764 5
FedBT\Circle	**128.83 (37.03%)**	**28.37 (6.51%)**	0.706 8
FedBT\Add	148.29 (42.62%)	**28.37 (6.51%)**	0.770 8
Prog	143.63 (41.28%)	51.05 (11.72%)	0.664 7
Porg\Add	137.38 (39.49%)	**28.37 (6.51%)**	0.721 8
Prog-Map	143.63 (41.28%)	51.05 (11.72%)	0.742 3
FedBT (ours)	167.89 (48.26%)	87.95 (20.19%)	0.794 2

除此之外, 还类比 FedBT 在进一步预训练场景下的渐进式采样映射算法, 在下游任务微调阶段使用类似的策略来决定训练的 Transformer 层序号 l 并构建本地模型. 具体来说, 如果需要进行 10 轮联邦学习, 那这 10 轮联邦学习中 l 的值为 [11, 11, 11, 11, 11, 10, 10, 10, 9, 8]. 主要对比的方法如下.

- Prog, 此方法除去 l 值的决定策略之外, 同 FedBT 一致, 会递增地训练更多的 Transformer 层;
- Prog\Add, 此方法为渐进式采样策略下, 在客户端微调本地模型的过程中, 每次只训练一层 Transformer 参数和输出层参数, 而不会递增的训练更多参数;
- Prog-Map, 此方法为渐进式采样策略下, 在客户端的本地模型训练过程中类比第 3.2 节所描述的策略进行浅层采样构建本地模型.

由表 6 可知, FedBT 框架在下游任务微调场景下的有效性, 引入的循环递减训练算法能够在有限的资源下高效地完成模型的微调. 该算法在微调模型时更加注重训练模型的深层 Transformer 参数, 并能够协同更多深层 Transformer 参数参与训练, 以确保重要参数都能够得到有效更新. 同时, 通过数据对比分析可知, 相较于其他消耗更少资源的训练策略, FedBT 框架不仅模型表现更佳, 而且增加的额外开销有限. 同表现更佳的策略相比, FedBT 消耗更少的资源并且模型表现效果接近. 此外, 如果本地模型已经是完整的 BERT 模型, 那么根据第 3.2 节所描述的策略进行浅层采样构建本地模型并不能提升模型的表现, 因此在下游任务微调阶段, FedBT 放弃了这一策略.

并且可知, 虽然整体渐进式训练算法在计算和通信成本上更具优势, 但模型整体性能却明显不及循环递减训练算法. 这是因为在下游任务微调阶段, 由于数据集规模较小且深层 Transformer 更贴近输出层, 模型对数据的敏感性更高, 因此需要对深层 Transformer 的所有参数进行协同训练才能确保关键参数能够充分调整, 循环递减训练算法有效达成了这一目标. 同时, 从表 6 中可以看出, 当本地模型已为完整的 BERT 模型时, 第 3.2 节所描述的进行浅层采样构建本地模型的策略并不能提升模型的表现.

表 7 展示了本地模型是其他尺寸的对比情况, FedBT 框架在下游任务微调场景下, 客户端模型 Transformer 层的数量 L_k, 更多的是影响模型的表现效果, 而不是模型计算资源, 本节将对比分析 $L_k = 6$, $L_k = 8$, $L_k = 10$ 这 3 种模型, 分别记作 FedBT-6, FedBT-8, FedBT-10. 针对每种模型, 对比分析 FedBT-Map, FedBT\Add 两种情况. 对以上模型使用 ChemProt 数据集, 在联邦场景下对上述各种模型进行微调之后, 可以得到各种策略模型在下游任务上的准确率, 本地计算开销与联邦聚合通信开销.

表 7 下游任务微调场景下的客户端模型规模对比实验

模型	计算开销 (s)	通信开销 (MB)	准确率
FedBT-6-Map	94.62 (27.20%)	87.95 (20.19%)	0.576 2
FedBT-6\Add	**93.88 (26.98%)**	**28.37 (6.51%)**	0.515 4
FedBT-6	94.62 (27.20%)	87.95 (20.19%)	0.601 9
FedBT-8-Map	129.86 (37.33%)	87.95 (20.19%)	0.695 0
FedBT-8\Add	111.86 (32.15%)	**28.37 (6.51%)**	0.593 5
FedBT-8	129.86 (37.33%)	87.95 (20.19%)	0.678 6
FedBT-10-Map	148.28 (42.62%)	87.95 (20.19%)	0.750 9
FedBT-10\Add	129.85 (37.32%)	**28.37 (6.51%)**	0.732 8
FedBT-10	148.28 (42.62%)	87.95 (20.19%)	0.767 4
FedBT-Map	167.89 (48.26%)	87.95 (20.19%)	0.764 5
FedBT\Add	148.29 (42.62%)	**28.37 (6.51%)**	0.770 8
FedBT (ours)	167.89 (48.26%)	87.95 (20.19%)	**0.794 2**

由表 7 可知, 客户端模型 Transformer 层的数量 L_k, 更多的是影响模型的表现效果, 而不是模型计算资源, 相比于减少客户端网络层数所节约的计算资源, 增加网络层数会更好地提升模型的表现效果. 在训练过程中使用浅层采样构建本地模型这一策略, 在模型本地模型层数较浅时对于提升模型表现效果有一定影响, 但是当本地模型的 Transformer 层数较多时, 原始的网络结构已经能很好地捕捉数据信息, 进行浅层采样映射会影响模型效果.

综上所述, FedBT 框架结合渐进式训练算法, 深层采样映射算法和循环递减训练算法, 能够在充分减少客户端的训练过程中的计算开销, 联邦聚合过程中的通信开销的前提下, 保证模型的训练效果, 高效地完成 BERT 模型的进一步预训练和下游任务微调, 实现同传统联邦学习训练完整模型接近的精确度.

6 总结与展望

本文针对联邦学习场景下客户端资源受限, 无法完成需要大量计算资源和通信资源的模型训练的问题, 提出了一种联邦学习场景下 BERT 模型的高效训练框架 FedBT. 此框架基于渐进式训练, 采样映射和循环递减训练等算法, 可以根据任务分类在联邦学习的客户端上训练 BERT 模型的不同部分参数, 从而节约客户端的通信开销和计算开销, 进而在保护用户隐私数据的前提下完成对大型预训练语言模型的训练. 然而本文所研究的 BERT 模型是一种自编码式 (autoencoder) 的预训练语言模型, 对于自回归式的预训练语言模型 (autoregressive), 如 GPT 之类的生成式语言模型, 在联邦学习场景下的应用没有进行深入研究, 这将是本文之后的一个研究方向.

References:

[1] Wang NY, Ye YX, Liu L, Feng LZ, Bao T, Peng T. Language models based on deep learning: A review. Ruan Jian Xue Bao/Journal of Software, 2021, 32(4): 1082–1115 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/6169.htm [doi: 10.13328/j.cnki.jos.006169]

[2] Song XM, Nie LQ, Shen HT, Tian Q, Huang H. Special topic: Research on multimodal learning integrated with pre-training techniques preface. Ruan Jian Xue Bao/Journal of Software, 2023, 34(5): 1997–1999 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/6776.htm [doi: 10.13328/j.cnki.jos.006776]

[3] Gu YH, Bai YB. Survey on security and privacy of federated learning models. Ruan Jian Xue Bao/Journal of Software, 2023, 34(6): 2833–2864 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/6658.htm [doi: 10.13328/j.cnki.jos.006658]

[4] Tan ZW, Zhang LF. Survey on privacy preserving techniques for machine learning. Ruan Jian Xue Bao/Journal of Software, 2020, 31(7):

2127–2156 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/6052.htm [doi: 10.13328/j.cnki.jos.006052]

[5] McMahan B, Moore E, Ramage D, Hampson S, Arcas BA. Communication-efficient learning of deep networks from decentralized data. In: Proc. of the 20th Int'l Conf. on Artificial Intelligence and Statistics. Fort Lauderdale: PMLR, 2017. 1273–1282.

[6] Peters M, Neumann M, Iyyer M, Gardner M, Zettlemoyer L. Deep contextualized word representations. In: Proc. of the 2018 Conf. of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Vol. 1 (Long Papers). New Orleans: Association for Computational Linguistics, 2018. 2227–2237. [doi: 10.18653/v1/N18-1202]

[7] Radford A, Narasimhan K, Salimans T, Sutskever I. Improving language understanding by generative pre-training. 2018. https://cdn.openai.com/research-covers/language-unsupervised/language_understanding_paper.pdf

[8] Devlin J, Chang MW, Lee K, Toutanova K. BERT: Pre-training of deep bidirectional Transformers for language understanding. In: Proc. of the 2019 Conf. of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Vol. 1 (Long and Short Papers). Minneapolis: Association for Computational Linguistics, 2019. 4171–4186. [doi: 10.18653/v1/N19-1423]

[9] Liu YH, Ott M, Goyal N, Du JF, Joshi M, Chen DQ, Levy O, Lewis M, Zettlemoyer L, Stoyanov V. RoBERTa: A robustly optimized BERT pretraining approach. arXiv:1907.11692, 2019.

[10] Sun C, Qiu XP, Xu YG, Huang XJ. How to fine-tune BERT for text classification? In: Proc. of the 18th China National Conf. on Chinese Computational Linguistics. Kunming: Springer, 2019. 194–206. [doi: 10.1007/978-3-030-32381-3_16]

[11] Krizhevsky A. Learning multiple layers of features from tiny images. Technical Report, Toronto: University of Toronto, 2009.

[12] He KM, Zhang XY, Ren SQ, Sun J. Deep residual learning for image recognition. In: Proc. of the 2016 IEEE Conf. on Computer Vision and Pattern Recognition. Las Vegas: IEEE, 2016. 770–778. [doi: 10.1109/CVPR.2016.90]

[13] Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser Ł, Polosukhin I. Attention is all you need. In: Proc. of the 31st Int'l Conf. on Neural Information Processing Systems. Long Beach: Curran Associates Inc., 2017. 6000–6010.

[14] Wang HP, Stich SU, He Y, Fritz M. Progfed: Effective, communication, and computation efficient federated learning by progressive training. In: Proc. of the 39th Int'l Conf. on Machine Learning. Baltimore: PMLR, 2022. 23034–23054.

[15] Alistarh D, Grubic D, Li JZ, Tomioka R, Vojnovic M. QSGD: Communication-efficient SGD via gradient quantization and encoding. In: Proc. of the 31st Int'l Conf. on Neural Information Processing Systems. Long Beach: Curran Associates Inc., 2017. 1707–1718.

[16] Lin YJ, Han S, Mao HZ, Wang Y, Dally B. Deep gradient compression: Reducing the communication bandwidth for distributed training. In: Proc. of the 6th Int'l Conf. on Learning Representations. Vancouver: OpenReview.net, 2018.

[17] Fu FC, Hu YZ, He YH, Jiang JW, Shao YX, Zhang C, Cui B. Don't waste your bits! Squeeze activations and gradients for deep neural networks via TinyScript. In: Proc. of the 37th Int'l Conf. on Machine Learning. PMLR, 2020. 3304–3314.

[18] Stich SU, Cordonnier JB, Jaggi M. Sparsified SGD with memory. In: Proc. of the 32nd Int'l Conf. on Neural Information Processing Systems. Montreal: Curran Associates Inc., 2018. 4452–4463.

[19] Konečný J, McMahan HB, Yu FX, Richtárik P, Suresh AT, Bacon D. Federated learning: Strategies for improving communication efficiency. arXiv:1610.05492, 2016.

[20] Li DL, Wang JP. FedMD: Heterogenous federated learning via model distillation. arXiv:1910.03581, 2019.

[21] Lin T, Kong LJ, Stich SU, Jaggi M. Ensemble distillation for robust model fusion in federated learning. In: Proc. of the 34th Int'l Conf. on Neural Information Processing Systems. Vancouver: Curran Associates Inc., 2020. 198.

[22] Hinton G, Vinyals O, Dean J. Distilling the knowledge in a neural network. arXiv:1503.02531, 2015.

[23] Wang XA, Li H, Chen K, Shou LD. FEDBFPT: An efficient federated learning framework for BERT further pre-training. In: Proc. of the 32nd Int'l Joint Conf. on Artificial Intelligence. Macao: ijcai.org, 2023. 4344–4352. [doi: 10.24963/IJCAI.2023/483]

[24] Wang Y, Li GL, Li KY. Survey on contribution evaluation for federated learning. Ruan Jian Xue Bao/Journal of Software, 2023, 34(3): 1168–1192 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/6786.htm [doi: 10.13328/j.cnki.jos.006786]

[25] Rong X. Word2Vec parameter learning explained. arXiv:1411.2738, 2014.

[26] Pennington J, Socher R, Manning CD. GloVe: Global vectors for word representation. In: Proc. of the 2014 Conf. on Empirical Methods in Natural Language Processing (EMNLP). Doha: Association for Computational Linguistics, 2014. 1532–1543. [doi: 10.3115/v1/D14-1162]

[27] Beltagy I, Lo K, Cohan A. SciBERT: A pretrained language model for scientific text. In: Proc. of the 2019 Conf. on Empirical Methods in Natural Language Processing and the 9th Int'l Joint Conf. on Natural Language Processing (EMNLP-IJCNLP). Hong Kong: Association for Computational Linguistics, 2019. 3615–3620. [doi: 10.18653/v1/D19-1371]

[28] Lee J, Yoon W, Kim S, Kim D, Kim S, So CH, Kang J. BioBERT: A pre-trained biomedical language representation model for biomedical text mining. Bioinformatics, 2020, 36(4): 1234–1240. [doi: 10.1093/bioinformatics/btz682]

[29] Yang Y, Uy MCS, Huang A. FinBERT: A pretrained language model for financial communications. arXiv:2006.08097, 2020.

[30] Zhou XH, Shen J. Survey on federated learning for medical application scenarios. Information Technology and Informatization, 2023, (11): 135–141 (in Chinese). [doi: 10.3969/j.issn.1672-9528.2023.11.031]

[31] Sanh V, Debut L, Chaumond J, Wolf T. DistilBERT, a distilled version of BERT: Smaller, faster, cheaper and lighter. arXiv:1910.01108, 2019.

[32] Jiao XQ, Yin YC, Shang LF, Jiang X, Chen X, Li LL, Wang F, Liu Q. TinyBERT: Distilling BERT for natural language understanding. In: Proc. of the 2020 Findings of the Association for Computational Linguistics. Association for Computational Linguistics, 2020. 4163–4174. [doi: 10.18653/v1/2020.findings-emnlp.372]

[33] Lit Z, Sit S, Wang JZ, Xiao J. Federated split BERT for heterogeneous text classification. In: Proc. of the 2022 Int'l Joint Conf. on Neural Networks (IJCNN). Padua: IEEE, 2022. 1–8. [doi: 10.1109/IJCNN55064.2022.9892845]

[34] Tian YYS, Wan Y, Lyu LJ, Yao DZ, Jin H, Sun LC. FEDBERT: When federated learning meets pre-training. ACM Trans. on Intelligent Systems and Technology, 2022, 13(4): 66. [doi: 10.1145/3510033]

[35] Hao YR, Dong L, Wei FR, Xu K. Visualizing and understanding the effectiveness of BERT. In: Proc. of the 2019 Conf. on Empirical Methods in Natural Language Processing and the 9th Int'l Joint Conf. on Natural Language Processing (EMNLP-IJCNLP). Hong Kong: Association for Computational Linguistics, 2019. 4143–4152. [doi: 10.18653/v1/D19-1424]

[36] Jawahar G, Sagot B, Seddah D. What does BERT learn about the structure of language? In: Proc. of the 57th Annual Meeting of the Association for Computational Linguistics. Florence: Association for Computational Linguistics, 2019. 3651–3657. [doi: 10.18653/v1/P19-1356]

[37] Manginas N, Chalkidis I, Malakasiotis P. Layer-wise guided training for BERT: Learning incrementally refined document representations. In: Proc. of the 4th Workshop on Structured Prediction for NLP. Association for Computational Linguistics, 2020. 53–61. [doi: 10.18653/v1/2020.spnlp-1.7]

[38] Wang J, Chen K, Chen G, Shou LD, McAuley J. SkipBERT: Efficient inference with shallow layer skipping. In: Proc. of the 60th Annual Meeting of the Association for Computational Linguistics (Vol. 1: Long Papers). Dublin: Association for Computational Linguistics, 2022. 7287–7301. [doi: 10.18653/v1/2022.acl-long.503]

[39] Fayek HM, Cavedon L, Wu HR. Progressive learning: A deep learning framework for continual learning. Neural Networks, 2020, 128: 345–357. [doi: 10.1016/j.neunet.2020.05.011]

[40] Lo K, Wang LL, Neumann M, Kinney R, Weld D. S2ORC: The semantic scholar open research corpus. In: Proc. of the 58th Annual Meeting of the Association for Computational Linguistics. Association for Computational Linguistics, 2020. 4969–4983. [doi: 10.18653/v1/2020.acl-main.447]

[41] Liang Z, Wang HZ, Dai JJ, Shao XY, Ding XO, Mu TY. Interpretability of entity matching based on pre-trained language model. Ruan Jian Xue Bao/Journal of Software, 2023, 34(3): 1087–1108 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/6794.htm [doi: 10.13328/j.cnki.jos.006794]

[42] Sheng XC, Chen DW. Research on text classification model based on federated learning and differential privacy. Journal of Information Security Research, 2023, 9(12): 1145–1151 (in Chinese with English abstract). [doi: 10.12379/j.issn.2096-1057.2023.12.02]

[43] Li BH, Xiang YX, Feng D, He ZC, Wu JJ, Dai TL, Li J. Short text classification model combining knowledge aware and dual attention. Journal of Software, 2022, 33(10): 3565–3581 (in Chinese with English abstract). [doi: 10.13328/j.cnki.jos.006630]

[44] Zhao JS, Song MX, Gao X, Zhu QM. Research on text representation in natural language processing. Ruan Jian Xue Bao/Journal of Software, 2022, 33(1): 102–128 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/6304.htm [doi: 10.13328/j.cnki.jos.006304]

[45] Collier N, Ohta T, Tsuruoka Y, Tateisi Y, Kim JD. Introduction to the bio-entity recognition task at JNLPBA. In: Proc. of the 2004 Int'l Joint Workshop on Natural Language Processing in Biomedicine and Its Applications (NLPBA/BioNLP). Geneva: COLING, 2004. 73–78.

[46] Luan Y, He LH, Ostendorf M, Hajishirzi H. Multi-task identification of entities, relations, and coreference for scientific knowledge graph construction. In: Proc. of the 2018 Conf. on Empirical Methods in Natural Language Processing. Brussels: Association for Computational Linguistics, 2018. 3219–3232. [doi: 10.18653/v1/D18-1360]

[47] Li J, Sun YP, Johnson RJ, Sciaky D, Wei CH, Leaman R, Davis AP, Mattingly CJ, Wiegers TC, Lu ZY. BioCreative V CDR task corpus: A resource for chemical disease relation extraction. Database, 2016, 2016: baw068. [doi: 10.1093/database/baw068]

[48] Doğan RI, Leaman R, Lu ZY. NCBI disease corpus: A resource for disease name recognition and concept normalization. Journal of Biomedical Informatics, 2014, 47: 1–10. [doi: 10.1016/j.jbi.2013.12.006]

[49] Kringelum J, Kjaerulff SK, Brunak S, Lund O, Oprea TI, Taboureau O. ChemProt-3.0: A global chemical biology diseases mapping. Database, 2016, 2016: bav123. [doi: 10.1093/database/bav123]

[50] Cohan A, Ammar W, van Zuylen M, Cady F. Structural scaffolds for citation intent classification in scientific publications. In: Proc. of the 2019 Conf. of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Vol. 1 (Long and Short Papers). Minneapolis: Association for Computational Linguistics, 2019. 3586–3596. [doi: 10.18653/v1/N19-1361]

[51] Dosovitskiy A, Beyer L, Kolesnikov A, Weissenborn D, Zhai XH, Unterthiner T, Dehghani M, Minderer M, Heigold G, Gelly S, Uszkoreit J, Houlsby N. An image is worth 16x16 words: Transformers for image recognition at scale. In: Proc. of the 9th Int'l Conf. on Learning Representations. OpenReview.net, 2021.

[52] Le Y, Yang X. Tiny imagenet visual recognition challenge. 2015. http://vision.stanford.edu/teaching/cs231n/reports/2015/pdfs/yle_project.pdf

附中文参考文献:

[1] 王乃钰, 叶育鑫, 刘露, 凤丽洲, 包铁, 彭涛. 基于深度学习的语言模型研究进展. 软件学报, 2021, 32(4): 1082–1115. http://www.jos.org.cn/1000-9825/6169.htm [doi: 10.13328/j.cnki.jos.006169]

[2] 宋雪萌, 聂礼强, 申恒涛, 田奇, 黄华. 融合预训练技术的多模态学习研究专题前言. 软件学报, 2023, 34(5): 1997–1999. http://www.jos.org.cn/1000-9825/6776.htm [doi: 10.13328/j.cnki.jos.006776]

[3] 顾育豪, 白跃彬. 联邦学习模型安全与隐私研究进展. 软件学报, 2023, 34(6): 2833–2864. http://www.jos.org.cn/1000-9825/6658.htm [doi: 10.13328/j.cnki.jos.006658]

[4] 谭作文, 张连福. 机器学习隐私保护研究综述. 软件学报, 2020, 31(7): 2127–2156. http://www.jos.org.cn/1000-9825/6052.htm [doi: 10.13328/j.cnki.jos.006052]

[24] 王勇, 李国良, 李开宇. 联邦学习贡献评估综述. 软件学报, 2023, 34(3): 1168–1192. http://www.jos.org.cn/1000-9825/6786.htm [doi: 10.13328/j.cnki.jos.006786]

[30] 周新虹, 沈洁. 面向医疗应用场景的联邦学习综述. 信息技术与信息化, 2023, (11): 135–141. [doi: 10.3969/j.issn.1672-9528.2023.11.031]

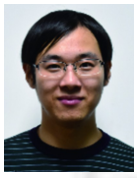
[41] 梁峥, 王宏志, 戴加佳, 邵心玥, 丁小欧, 穆添愉. 预训练语言模型实体匹配的可解释性. 软件学报, 2023, 34(3): 1087–1108. http://www.jos.org.cn/1000-9825/6794.htm [doi: 10.13328/j.cnki.jos.006794]

[42] 盛雪晨, 陈丹伟. 基于联邦学习和差分隐私的文本分类模型研究. 信息安全研究, 2023, 9(12): 1145–1151. [doi: 10.12379/j.issn.2096-1057.2023.12.02]

[43] 李博涵, 向宇轩, 封顶, 何志超, 吴佳骏, 戴天伦, 李静. 融合知识感知与双重注意力的短文本分类模型. 软件学报, 2022, 33(10): 3565–3581. [doi: 10.13328/j.cnki.jos.006630]

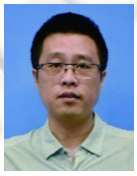
[44] 赵京胜, 宋梦雪, 高祥, 朱巧明. 自然语言处理中的文本表示研究. 软件学报, 2022, 33(1): 102–128. http://www.jos.org.cn/1000-9825/6304.htm [doi: 10.13328/j.cnki.jos.006304]

王鑫澳(2001－), 男, 硕士, 主要研究领域为联邦学习, 模型轻量化预训练.



骆歆远(1988－), 男, 博士, 助理研究员, 主要研究领域为大数据管理, 大数据智能计算, 信息检索.

陈珂(1977－), 女, 博士, 副研究员, CCF 专业会员, 主要研究领域为非结构化数据管理, 数据挖掘, 隐私保护.



陈刚(1973－), 男, 博士, 教授, 博士生导师, CCF 杰出会员, 主要研究领域为数据库, 大数据管理系统, 大数据智能计算.

寿黎但(1976－), 男, 博士, 教授, 博士生导师, CCF 高级会员, 主要研究领域为非结构化数据管理, 移动社会媒体数据管理, 多媒体挖掘.