软件学报 ISSN 1000-9825,CODEN RUXUEW
Journal of Software,[doi:10.13328/j.cnki.jos.007278]

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

FBO: 基于联邦学习的云数据库旋钮调优技术 *

燕钰, 戴志宇, 吕泽楷, 王宏志

(哈尔滨工业大学 计算学部,黑龙江 哈尔滨 150001)

通信作者:王宏志, Email:wangzh@hit.edu.cn

摘 要: 近年来,随着软硬件的发展,数据库上云已经成为了新兴发展趋势,并且能够降低中小型企业和个人用户的数据库运维成本.进一步地,云数据库的发展带来了庞大的运维市场需求,研究者们提出了诸多数据库自调优技术来支持数据库旋钮自动优化.为了提高调优效率,现有技术从仅仅关注调优问题本身,到开始关注如何复用历史经验来为当前数据库实例找到最佳参数配置.然而,随着云数据库的发展,用户逐渐提高了对隐私保护的要求,期望在拥有高效数据存取效率的同时避免隐私泄露.现有方法并未考虑到保护用户的历史调优经验隐私,可能会使得用户负载特征被感知,带来经济损失.本文详细分析了云数据库调优任务的特点,有机结合服务端和用户端,提出了一种基于联邦学习的云数据库旋钮调优技术.首先,为了解决联邦学习中数据异构的问题,本文提出了基于元特征匹配的经验筛选方法来提前将数据分布差异较大的历史经验剔除,以提高联邦学习的效率.为了实现保护用户隐私,本文有机结合了云数据库服务特性,提出了以节点端为训练中心的联邦贝叶斯调优算法,通过随机傅里叶特征来完成保证调优经验不失真的前提下保护用户隐私.在多个公开 benchmark 上的结果表明本文的方法可以达到与现有调优方法相当的调优结果,并且由于复用了历史经验,可以大大提高调优效率.

关键词: 云数据库;联邦学习;旋钮调优

中图法分类号: TP311

中文引用格式: 燕钰, 戴志宇, 吕泽楷, 王宏志. FBO: 基于联邦学习的云数据库旋钮调优技术. 软件学报. http://www.jos.org.cn/1000-9825/7278.htm

英文引用格式: Yan Y, Dai ZY, Lü ZK, Wang HZ. • FBO: The Cloud Database Knob-tuning Method Based on Federated learning. Ruan Jian Xue Bao/Journal of Software (in Chinese). http://www.jos.org.cn/1000-9825/7278.htm

- **FBO: The Cloud Database Knob-tuning Method Based on Federated learning**

YAN Yu, DAI Zhi-Yu, LÜ Ze-Kai, WANG Hong-Zhi

(Faculty of Computer, Harbin Institute of Technology, Harbin 150001, China)

Abstract: In recent years, with the development of software and hardware, the cloud database has become an emerging development trend and can reduce database operation and maintenance costs for small and medium-sized enterprises and individual users. Furthermore, the development of cloud databases has brought huge market demands for database tunings. Researchers have proposed many database self-tuning technologies to support automatic optimization of database knobs. To improve tuning efficiency, existing technologies have shifted from focusing solely on the tuning problem itself to focusing on how to reuse historical experience to find the optimal parameter configuration for the current database instance. However, with the development of cloud databases, users have gradually increased their requirements for privacy protection, hoping to avoid privacy leaks while having efficient data access efficiency. Existing methods do not consider protecting the privacy of users' historical tuning experience, which may cause user load characteristics to be perceived, causing economic losses. This article analyzes the characteristics of cloud database tuning tasks in detail, organically combines the server and the user, and proposes a cloud database knob-tuning technology based on federated learning. First, in order to solve the problem of data heterogeneity in federated learning, this paper proposes an experience screening method based on meta-feature matching to eliminate historical experiences with large differences in data distribution in advance to improve the efficiency of federated learning. In order to protect user privacy, this paper organically combines the characteristics of cloud database services and proposes a federated Bayesian optimization algorithm with the node as the training center. We utilize random Fourier features to complete the protection without distorting the tuning

* 基金项目:国家自然科学基金(编号:62232005,62202126,92267203)
收稿时间: 2024-05-27; 修改时间: 2024-07-16, 2024-08-19; 采用时间: 2024-08-29; jos 在线出版时间: 2024-09-13

experience. The evaluation results in extensive benchmarks present that our method could achieve competitive tuning performance compared with the existing centralized tuning methods.

Key words: Cloud Database; Federated Learning; Knob Tuning

随着数据库系统的日益复杂,自动化旋钮调优[1]已经成为优化数据库效率的重点研究.目前,研究者们提出了诸多基于贝叶斯优化[2]和强化学习[3]的智能方法来解决自动化旋钮调优问题.如 Cai 等人[3]提出了一种个性化配置的在线云数据库调优方法,首先使用机器学习方法削减旋钮范围,而后使用遗传算法增强强化学习旋钮调优器.Shen 等人[4]提出了一种基于广义迁移学习的旋钮调优方法来优化数据处理效率.

随着云数据库的发展,旋钮调优的一个主流优化方向是有效地重用历史经验[5],使得云数据库的各个用户之间可以互相增强,实现更加高效的旋钮优化.具体来说,如果云数据库厂商已经为 n 个用户场景进行了细致地自动化旋钮调优,那么对于某个与历史用户相似的新用户(例如都是银行信用卡业务),历史调优经验可以大幅度提高新用户的数据库旋钮优化效率.然而,历史经验重用在提高优化效率的同时,也带来了新的隐私保护问题[6].云数据库用户之间可能存在潜在的商业竞争关系,特别是拥有比较相似业务的用户,用户不能完全信任其他用户.如果不对历史调优经验进行隐私保护,在新用户利用这些经验来优化数据库性能时,很可能会泄露历史用户的重要业务特征,导致商业损失.此外,即使有商业保密协议的约束,用户对云厂商的完全信任也难以保证.随着大数据的发展,云厂商可能通过用户调优经验数据中发现商业机会,从而间接泄露用户隐私,带来商业损失的风险.因此,云数据库调优领域需要构建隐私保护机制,防止重用经验时用户之间以及用户与云厂商之间可能存在的隐私泄露问题,从而确保调优过程的安全性和对商业机密的保护.

通过分析云数据库场景,本文总结了设计基于联邦学习的旋钮调优技术存在的三个核心挑战:(1) 不同于传统联邦学习场景,云数据库的各个用户之间数据分布差异性大,导致模型拟合效果差.比如一个云数据库厂商可能同时服务多种类型的业务,银行金融类业务,工业数据类业务,政府数据管理业务等.这些业务之间,数据体量,数据结构,工作负载类型都是千差万别的,难以直接应用联邦学习来加速调优.(2) 与传统隐私保护学习场景不同,旋钮调优任务的模型参数更新只发生在当前用户节点端而不是云厂商服务端,并且不会反向传播更新其他用户节点.直接传递模型权重或者加密模型参数的方法不能直接适配到用户节点模型更新.(3) 云数据库厂商和数据库用户之间是可信任的,出于商业协议,厂商通常会细致地了解用户数据特征信息,为其提供高效的数据库服务.

为了针对上述特点,解决基于历史信息的云数据库调优中隐私保护问题,本文提出了一种创新性的联邦学习方法来处理旋钮调优过程中经验回放保护,能够有机结合云数据库服务端和节点端的优势,在保证数据库调优效率的前提下,有效提高用户数据安全性.本文的主要贡献如下:

（1） 为了解决联邦学习中数据异构的问题,本文提出了基于元特征匹配的经验筛选方法来提前将数据分布差异较大的历史经验剔除,以提高联邦学习的效率.

（2） 为了实现保护用户隐私,本文有机结合了云数据库服务特性,提出了以节点端为训练中心的联邦贝叶斯调优算法,通过随机傅里叶特征来完成保证调优经验不失真的前提下保护用户隐私.

（3） 本文在公开测试集上对比了联邦贝叶斯调优方法和现有调优方法的性能,发现本文的方法在多个工作负载下均表现出显著优势.例如,在 YCSB 负载中,相较于 DDPG,本文的 FBO 方法能将调优时间节省为原来的 17%,并且还能有效提高调优后负载执行性能.

本文的组织架构如下:第 1 节介绍云数据库旋钮调优和联邦学习的相关工作;第 2 节介绍本文提出的基于联邦学习云数据库旋钮调优方法的整体流程;第 3 节介绍基于元特征匹配的经验筛选方法来解决用户业务异构问题;第 4 节介绍联邦贝叶斯优化算法来解决用户隐私下调优模型更新问题;第 5 节介绍在两个开源 benchmark 下本文方法与现有方法的对比实验,实验结果表明本文的方法可以在保护用户隐私的情况下达到与现有调优方法相当的数据库性能;最后总结全文并展望未来工作.

1 相关工作

调优数据库系统旋钮以在给定工作负载上实现最佳性能是数据库领域的一个热门方向.随着云数据库的发展,复用已有的调优经验指导新场景调优成为了新的发展方向,以提高调优的时间效率.本文从两个方面介绍一下相关工作,包括数据库自调优技术和联邦学习技术.

1.1 数据库自调优技术

近年来,云数据库的发展使得中小型公司和个人用户也可以低成本建立自己的数据库服务.然而,大量的数据库调优运维工作给云数据库厂商带来了成本和效率的挑战.为了提高数据库调优效率,研究者们提出了诸多智能化方法[5][7][8]来进行自动化调优.从优化方向上来看,现有自调优技术的进展主要可以分为两个阶段,第一个阶段是单纯考虑优化旋钮配置带来的负载性能提升和资源配置优化;第二个阶段是考虑复用历史经验来尽可能节省调优资源,提高调优效率.具体来说,第一阶段的优化方法主要可以分为四种:启发式方法、贝叶斯优化方法、深度学习方法和强化学习方法.启发式方法在数据库自调优技术发展的初始阶段提出,以 David G Sullivan 等人 2004 年的研究[9]为代表,这种方法通过随机采样少量配置进行自动化探索.贝叶斯优化通过构建概率模型来预测和评估不同配置的性能.例如,OnlineTune[10]技术,这种方法通过的迭代采样和评估,能有效优化大规模的数据库系统配置.在处理更大的配置空间时,深度学习方法通过复杂的神经网络来构建概率模型进行优化,例如 D.Van Aken 等人提出的 DNN 方法[11]和 J.Tan 等人提出的 ibtune[12],这些方法通过替代传统的高斯过程,大幅提升了调优任务的处理能力和预测准确性.强化学习则通过代理和环境的交互来逐步优化数据库性能,例如 J.Zhang 等人提出的 CDBTune[1]和 G.Li 等人提出的 Qtune 技术[13],基于强化学习的调优方法不依赖历史数据,而是通过实时调整策略来适应新环境,能够有效应对变化的负载.第二个阶段研究者们考虑使用调优经验来提高调优效率.比较有代表性的工作是 X.Zhang 等人提出的 Restune[5],该方法使用文本嵌入来编码负载特征,而后基于元学习实现历史经验复用.更多地,rover[4]构建了调优经验树来优化贝叶斯调优过程中的模型初始化、模型探索指导等.但是以上方法均未考虑对用户历史调优经验进行保护,可能会带来隐私泄露,造成商业损失.

小结: 现有的数据库旋钮调优技术,包括基于启发式搜索、基于贝叶斯优化,基于强化学习的方法已经能够很好地处理静态单一负载下的旋钮优化.更多地,研究者们还尝试编码历史负载的特征来通过元学习或者集成学习等方法复用历史调优经验,以提高调优的时间效率.然而,随着云数据库的发展,更多的用户将数据库服务搬到云端,用户越来越重视数据的隐私保护,现有方法未能实现隐私保护下的旋钮调优,在经验复用时可能会带来潜在的数据泄露风险.

1.2 联邦学习隐私保护技术

联邦学习旨在保护隐私的同时联合多个参与方的数据构建机器学习模型.现有的理论研究中,大部分工作专注于基础模型,例如线性模型[14]、树模型[15][16]和深度神经网络(DNN)[17]的构建上,通过交换模型的中间参数来实现协作训练.此外,隐私保护技术[6]是联邦学习问题的另一个重点,研究者们针对中间参数交换的训练过程,提出了多种隐私攻击手段和隐私保护方法,典型的隐私攻击包括利用训练中的中间参数或请求查询来重建训练数据信息的重建攻击[18][19],以及推测特定记录或属性归属的推断攻击[20].为了应对这些隐私威胁,研究人员开发了基于中间参数过程的加密和信息模糊化的差分隐私保护方法.前者基于密码学方法,在加密后的数据上直接进行训练过程,从而避免了明文原始数据的泄露;后者通过数据处理,在任意单个数据随机化的同时,保证输出在统计上只有不可分辨的变化,从而模糊化具体原始数据保护隐私.例如,Aono Y 等人设计了一种高效的两方安全计算协议[21],用于实施联邦梯度下降算法训练回归模型;Wu Y 等人则结合使用同态加密、秘密共享和差分隐私等多种隐私保护技术,开发了一种通用的树模型纵向联邦学习系统[15].然而,对于一些常见的机器学习任务,如 DNN 超参数调整,由于缺少对梯度的中间参数访问,传统的联邦学习方法难以应用于这类协同训练.

为此,Dai 等人[22]提出联邦汤普森采样(FTS),通过整合来自其他智能体的信息来提高其 BO 任务的效率,把联邦学习扩展为联邦贝叶斯优化.尽管如此,当前针对数据库旋钮调优任务,利用联邦学习实现历史经验重用的研究仍然较少.

小结: 随着云数据库的发展,云数据库的各个用户积累的大量的数据可供训练,联邦学习提供了有效地重用这些历史经验的思路,用户与服务器之间交换信息互相增强优化器模型,同时现有联邦学习研究可以为云数据库经验重用的隐私保护提供方法指导,实现更加高效和安全的旋钮优化.但是传统联邦学习基于交换模型的中间参数,无法应用于超参数调整等无梯度的黑盒优化的任务,本文提出了完整的联邦学习云数据库旋钮调优框架,对单机训练的基于贝叶斯优化的旋钮调优模型进行了拓展以支持联邦学习的中间参数传递,将联邦学习扩展到无梯度的任务领域.本文的方法着重于对传递高斯过程模型的加密保护,与差分隐私的原始数据处理保护不冲突,拓展后的模型可以基于差分隐私实现用户隐私的信息模糊保护.

2 问题定义与框架

2.1 基本概念

本小节首先定义和本文密切相关的概念.

定义 1(旋钮调优). 数据库中的旋钮负责为多样化工作负载提供最佳的数据库配置,以大大提高存取效率.具体来说,数据库旋钮涉及数据库的多个模块,包括内存分配,查询并发控制,缓存配置,日志模式,写入配置,基数估计等.本文定义数据库旋钮空间为 $Knob=\{K_1,K_2,\ldots,K_n\}$,其中 n 是旋钮个数,X_i 是第 i 个旋钮的域值.在此基础上,旋钮调优的优化目标可以定义为寻找使得当前工作负载性能最佳的配置 $k^*:k^*=argmax_{k\in Knob}metric(k)$,其中 $metric(k)$ 是工作负载在配置 k 下的性能指标,可以是工作负载的时间代价或者吞吐量.

定义 2(联邦学习调优). 联邦学习是指使用多个机器上的数据来训练一个模型,并且能够保证这些机器上的数据隐私,进一步确保用户数据安全.本文定义历史云数据库客户端集合为 $C=\{C_1,C_2,\ldots,C_n\}$,每个客户端 C_i 有一系列调优经验 $D_i=\{(k_{i1},m_{i1}),(k_{i2},m_{i2}),\cdots\}$,其中 k_{i1} 是客户端 C_i 的旋钮配置,m_{i1} 是对应旋钮配置的性能指标.本文联邦学习调优的目标是在一个可靠服务端 S 的协调下,安全地复用历史客户端经验为新客户端 C_{n+1} 提供调优服务,找到最佳的旋钮配置.

定义 3(随机傅里叶特征). 随机傅里叶特征(Random Fourier Features ,RFF) [23]方法是一种核近似技术,特别适用于满足平移不变的核(稳定核),并被认为是加速机器学习核方法的有效方式之一[24][25][26].核近似技术的目标是通过一个低维映射函数 $\varphi(x)$,将原始的 D 维数据映射到 d 维随机特征空间中,在随机特征空间中,两个特征的内积约等于对应的核函数,即 $k(x,x')\approx\varphi(x)^T\varphi(x')$.在这种核近似技术的基础上,本文实现了调优经验的降维转换与重构,实现在经验传递时的隐私保护.

2.2 问题定义

目前广泛应用的联邦学习(Federated Learning,FL)采用客户-服务器架构.此架构涉及一个中央服务器 S 和多个数据持有者客户端 $C_1,\ldots,C_N$.假设系统中有 $N+1$ 参与方,其中服务端 S 可信任,每个客户端 C_k 拥有自己的训练数据集 $\{D_1,\cdots,D_N\}$.基于 $D=D_1\cup\cdots\cup D_N$ 联邦学习的调优问题可以表述为:

输入 各客户端的历史训练数据集 $\{D_k\}_{k=1}^N$ 和新的工作负载.

学习目标 在不共享任何原始数据的前提下,学习一个机器学习模型 M.该模型应能自动搜索该工作负载下,性能表现最佳的旋钮配置 k*.同时,需要构建隐私保护机制,以防止经验重用时的发生隐私泄露.

优化目标 假设传统的集中式模型通过将所有数据集合并作为训练数据集,进而训练得到模型 M_{SUM} 的精度为 V_{SUM}.联邦学习模型 M_{FED} 的精度为 V_{FED}.若存在非负实数 δ 使得:

$$|V_{FED}-V_{SUM}|<\delta \tag{1}$$

则可认为该联邦学习模型的精度损失在 δ 范围内.理想情况下,分布式协同训练的模型精度应接近集中训练模型的精度.

2.3 基于联邦学习云数据库旋钮调优方法框架

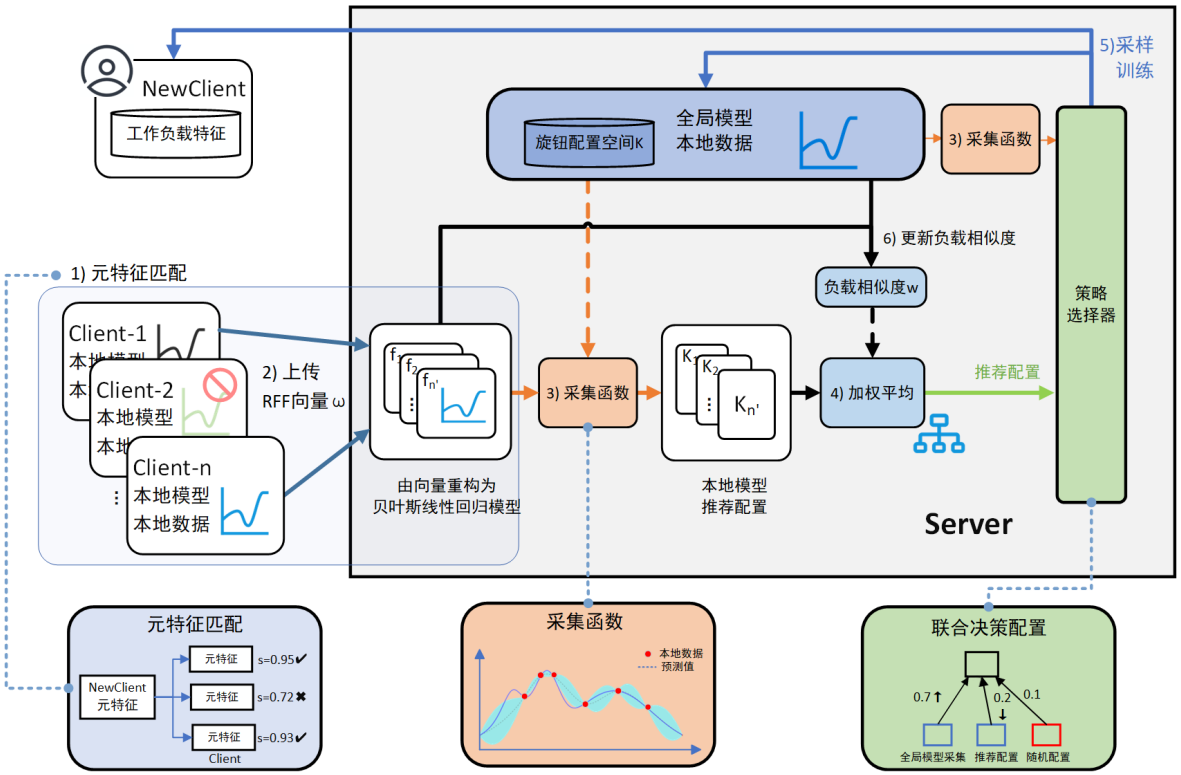


图 1 基于联邦学习云数据库旋钮调优过程

本小节介绍基于联邦学习云数据库旋钮调优技术的框架.为了实现保护用户隐私下的历史经验重用,本文设计了基于元特征匹配的经验筛选方法来完成历史模型的高斯核近似,而后在近似向量的基础上实现了联邦贝叶斯调优,为新用户联邦地完成旋钮调优推荐.

根据上文讨论,图 1 显示了基于联邦学习的贝叶斯优化系统框架,该框架包括四个主要组成部分:经验筛选,RFF 收集器,代理模型和采集函数.这几部分的具体功能如下:

1) 经验筛选:为了解决联邦学习的数据异构问题,本文通过考虑工作负载元特征,对分布差异过大的客户端经验进行过滤,以减小数据异构性带来的训练困难(详见第 3 章); 2) RFF 收集器:为了实现联邦学习数据的隐私保护,收集器与符合要求的客户端平台交互,向客户端发出包含 RFF 参数的训练请求.客户端收到请求后,利用参数对描述本地数据的模型进行隐私保护处理.再由收集器收集用户代理的返回的 RFF 向量信息,用以恢复客户端近似模型(隐私保护处理方法将在第 3 章具体介绍).

在经验筛选和 RFF 收集的基础上,本文的框架开始学习联邦调优器.概括来说, 3) 代理模型模块(详见第 4 章):本文的调优算法采用顺序式的、基于模型的优化方法,通过自适应采样搜索空间进行调优.代理模型旨在拟合配置与性能之间的关系,为搜索提供优化依据,模型分为客户端近似代理模型与全局代理模型两部分,前者由 RFF 收集器得到的 RFF 向量信息重建,而后者协调各近似模型学习新工作负载的知识进行更新; 4) 采集函

数(详见第 4 章):联邦学习中为了权衡全局模型与近似模型的协调训练,在迭代期间采集函数需要与策略决策器联合决策,选择适合的采样策略,为调优任务推荐有希望的配置,从而指导全局模型与当前任务用户的调优过程.

为了更好了说明基于联邦学习云数据库旋钮调优框架,本小节可以通过一个具体的例子来理解工作流程.假设一个名为 NewClient 的新用户对服务端请求调优,服务端会先基于该新负载的元特征匹配相似的历史调优经验.筛选出相似客户端后,服务端向这些客户端发出请求,各客户端对本地数据进行隐私处理后得到的 RFF 向量信息传会服务端.服务端接收到这些信息后,将其重构为近似模型 $f_1,\ldots,f_n$.接下来联邦算法开始调优,如果联合决策选择使用历史经验推荐配置,各近似模型会通过最大化采集函数给出本地模型推荐配置,然后根据与全局模型的相似度加权得到历史经验的综合推荐配置;否则,策略选择器会使用全局模型的采集函数给出推荐配置,或者随机生成配置.最终,得到的配置将传递给 NewClient 和全局模型,在真实数据库上测试性能,并加入调优历史,进入下一轮迭代.

3 基于元特征匹配的经验筛选

传统联邦学习中,不同客户端数据的异构性是一个重要问题,它可能导致训练过程中出现冗余和错误.为了解决这一问题,本文设计了一种基于元特征匹配的经验筛选方法.该方法旨在通过筛选掉与当前客户端存在潜在分布差异的历史客户端,进而利用与当前负载相似的历史调优经验进行联邦学习调优.全局代理 S 在开始调优任务之前,根据如读写比例和负载中算子比例的工作负载的元特征,对客户端相似度进行评估,排除差异过大的参与者.

元特征匹配本身是一个复杂且具有挑战性的问题,尤其是在处理动态负载变化时.现有的方法,如 ResTune[8],基于使用查询中保留字的 TF-IDF 值构建一个特征向量,在细粒度特征上进行比对,代价高且匹配速度较慢;而 Rover 方法[4]使用 17 个运行时指标进行特征向量化作为元特征,在匹配相似度前需要测量 SparkSQL 的运行时指标,也需要更多时间.结合云数据库调优的背景,本文中的元特征匹配需要兼顾匹配速度和时间效率,因此基于工作负载的元特征实现,注重负载变化,匹配速度快且适用于不同的工作负载.在调优过程中,特征并不是平等被使用,而是根据观测值的实时更新分配不同权重,以弥补在相似任务上的匹配误差,从而获得更好的效果.初始权重会随着数据集的实时变化进行更新,即使初筛不准确,也可以通过实时调整权重来优化匹配结果.

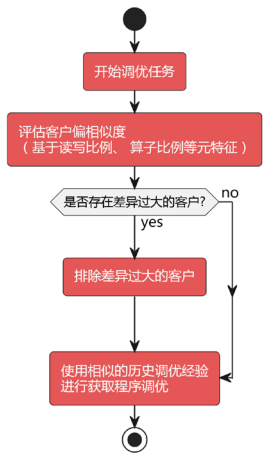


图 2 元特征经验筛选

随后,由收集器向其他代理 $C_1,\cdots,C_n$ 发送携带 RFF 近似所需参数的训练请求,希望代理返回本地的调优经验 D_i.但是因为调优经验可能涉及到用户隐私,在联邦学习中,为了降低隐私泄露的风险,客户端保留原始数据而不进行传输.本文中客户端的代理需要进行两次转换措施处理本地调优经验.首先,使用高斯过程模型去学习 D_i 经验数据,拟合出的模型能够描述旋钮配置与性能之间的关系,传统联邦学习通过传递中间参数信息,相比传递原始信息降低了隐私泄露的风险,但是在无梯度的贝叶斯优化任务中,由于高斯过程模型的更新不涉及中间梯度参数,只能传递模型本身,而模型本身隐含了许多原始数据信息,存在隐私风险.随后,利用随机傅里叶特征方法通过对本地高斯过程模型降维和重构进行加密,将原本传递的后验分布信息转换为一个向量,进一步保护用户隐私.

RFF 的核心思想是通过随机采样特征构建近似来避免对大型内核直接计算,在对内核降维的同时 RFF 还兼顾对精度的保证.Bochner 定理[2]指出,任何连续稳定核 k 可以表示为频谱密度的傅立叶积分 $p(w)$.而本文使用的高斯过程模型,其高斯核 $k(\Delta)=exp\left(-\frac{\|\Delta\|_2^2}{2}\right)$ 就是一种移位不变的稳定核,对应的频谱密度 $p(w)=(2\pi)^{-\frac{D}{2}}\exp\left(-\frac{\|w\|_2^2}{2}\right)$.对给定移位不变核函数 $k(x,y)=k(x-y)$,进行傅里叶逆变换[27]:

$$\begin{aligned} k(\boldsymbol{x},\boldsymbol{y}) &= k(\boldsymbol{x}-\boldsymbol{y}) \\ &= \int_{R^d} p(\boldsymbol{w}) e^{j\boldsymbol{w}^T(x-y)}\mathrm{d}\boldsymbol{w} \\ &= \boldsymbol{E}_{\boldsymbol{w}}\left[e^{j\boldsymbol{w}^T(x-y)}\right] \\ &= \boldsymbol{E}_{\boldsymbol{w}}\left[\varphi_{\boldsymbol{w}}(\boldsymbol{x})\varphi_{\boldsymbol{w}}(\boldsymbol{y})\right] \\ &\approx \varphi(\boldsymbol{x})^T\varphi(\boldsymbol{y}) \end{aligned} \tag{2}$$

其中 $\varphi_w(x)=\sqrt{2}\cos\left(w^Tx+b\right), w\sim p(w), b\sim \mathrm{Uniform}(0,2\pi)$.对 $\varphi_w(x)$ 使用蒙特卡洛方法抽样 D 次取均值,可以逼近期望得到 $\varphi(x)=\left[\varphi_{w1}(x),\cdots,\varphi_{wD}(x)\right]^T$,从而为所有本地数据 $x\in X$ 构建 D 维随机特征 $\varphi(x)$,其内积近似核值: $k(\boldsymbol{x},\boldsymbol{y})\approx\varphi(\boldsymbol{x})^T\varphi(\boldsymbol{y})$.

这种核近似方法在理论上具有高概率的近似质量保证.误差上界[23]: $\sup_{x,x'\in X}\left|k(x,x')-\varphi(x)^T\varphi(x')\right|\le\varepsilon$,其中 $\varepsilon=\mathcal{O}\left(D^{-1/2}\right)$,即通过增加随机特征数量 D 有效降低误差 ε 并提高近似质量,能够保证本文设计的联邦框架在传递调优经验时不失真.

具体来说,假设有一个历史客户端的调优经验 D_i,其中包括一系列调优经验 (k_{ij},m_{ij}),并假设该客户端收到随机特征近似的参数向量为 w 和 b.首先,客户端使用高斯过程模型对 D_i 中的数据进行拟合,这样可以模拟出该客户端的工作负载下,不同配置的性能表现.如图 3 左上,为一个基于 YCSB 负载客户端的部分调优经验,经过高斯过程拟合得到一个连续的函数映射关系 $f:R^n\to R$,该高斯过程在每个输入维度上的后验分布均为高斯分布,图左下为在 backend_flush_after 维度的结果可视化.这个函数描述了旋钮配置与性能之间的关系.之后,客户端代理利用 RFF 方法近似本地模型,计算得到特征映射函数 $\varphi(x)=\sqrt{2}\cos\left(w^Tx+b\right)$,将本地经验数据输入映射函数,得到一个输出向量,即为随机傅里叶特征 RFF.为了统一各客户端的向量维度,并且避免直接传输 RFF,客户端可以用 RFF 计算得到近似高斯核 $k(x,x')\approx\varphi(x)^T\varphi(x')$ 的后验分布 $\mathcal{N}\left(\nu_t,\sigma^2\Sigma_t^{-1}\right)$(见公式 5),从中进行 M 次抽样得到一个表示这个分布 M 维样本 ω(如图右上)并传回服务端代理.服务端将这个向量 ω 存储在数据知识库中,用于在必要时还原近似模型,获得工作负载相关信息.

此时,服务端利用同样的参数向量也可以构建出特征映射函数 $\varphi(x)$,用特征映射函数内积来近似高斯核

$k(x,x')\approx\varphi(x)^T\varphi(x')$,那么可以将高斯过程模型近似解释为线性贝叶斯模型 $f(x)=\varphi(x)^T\omega$.而在没有获得参数向量的情况下,恶意参与者无法构建出特征映射函数 $\varphi(x)$,也就无法利用客户端 RFF 中的信息,从而保证了数据的安全性.

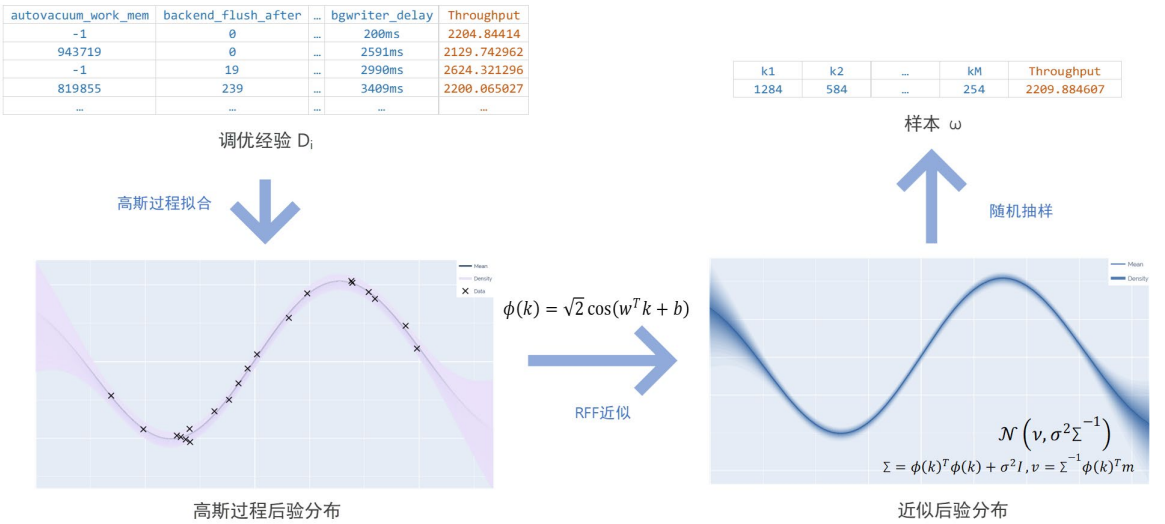


图 3 基于 YCSB 的客户端调优经验隐私保护示例

服务端 F 在接收到各代理消息后,开始运行联邦学习优化算法.在优化的早期阶段,全局模型还未充分训练,策略选择器更倾向于使用基于历史经验的推荐配置,以充分利用已有知识加速优化过程,快速达到一个较优的性能水平.随着迭代的进行,全局模型经过多轮更新后,开始更精确地表达出旋钮配置与性能之间的关系.在这个阶段,策略选择器更倾向选择更具优势的全局模型采集函数.为了避免局部收敛,策略选择器还会以一定概率随机选择配置,确保配置空间被广泛探索,从而发现可能被忽略的高性能配置.策略选择器一旦确定配置,将其传递给全局模型与用户,根据反馈数据更新代理模型并开始下一轮迭代.

算法 1: 计算客户端经验数据的 RFF 向量信息

输入 客户端上的调优经验数据集 (k,m), 随机特征数量 D
输出 经验数据集的 RFF 信息
1 // 假设服务端与客户端约定使用 w、b 向量加密
2 $x\leftarrow GP(k,m)$ // 先用高斯过程模型拟合经验, 再从拟合模型中采样
3 $\varphi_w(x)\leftarrow\sqrt{2}\cos\left(w^Tx+b\right)$
4 $RFF\leftarrow\left[\varphi_{w1}(x),\cdots,\varphi_{wD}(x)\right]^T$ // 蒙特卡洛方法抽样
5 **return** RFF

4 基于联邦学习的贝叶斯调优算法

本节提出一种基于联邦学习的贝叶斯调优算法,它基于联邦学习并结合贝叶斯优化的思想,通过协作多客户端学习一个贝叶斯优化模型,同时在各客户端上对训练数据保密.

贝叶斯优化是一种通过自适应采样搜索空间,以尽可能少的样本数量来找到未知函数 f 的最优值的方法.

它包括两个关键组成部分:代理模型 M 和采集函数 $acq(k;M)$.代理模型用于捕捉目标函数 f 的行为特征,而采集函数则用于指导选择下一次样本的位置.贝叶斯优化遵循基于模型的顺序优化框架,在反复迭代中通过采集函数选择样本,并不断更新其代理模型以确定函数 f 的极值.

基于联邦学习的贝叶斯优化框架将这一思想扩展到分布式环境中.在该框架下,服务端协调多个客户端参与训练过程.

算法 2: 基于联邦学习的贝叶斯调优算法

输入 现有的调优经验数据集 $D=\left\{\left(k^{(1)},m^{(1)}\right),\ldots,\left(k^{(n)},m^{(n)}\right)\right\}$ 及对应元特征 $\{meta_1,meta_2,\ldots,meta_n\}$,新工作负载的性能度量 $f(k)$ 及元特征 $meta$,元特征筛选阈值 γ,迭代轮数 n

输出 最大吞吐量 m_{max} 和最佳配置 k_{best}

```
// Step1. 元特征匹配客户端,客户端计算 RFF 特征向量,传递给服务端
ω ← ∅
For i = 1 to n do                          // 请求每个客户端
    s = sim(meta, meta_i)                  // 计算与新工作负载元特征的相似度
    If s > γ do
        ω ← ω ∪ RFF(k^(i), m^(i))
    End if
End for

// Step2. 服务端初始化全局模型和客户端模型
global_model ← init_model()
client_models ← init_client_models(ω)

// Step3. 服务端开始协作训练贝叶斯优化模型
k_max ← k_default
m_max ← f(k_default)
iter ← 0, history ← ∅
While iter < n do
    global_model ← udpate_model(history)
    k ← advise_configuration(global_model, client_models, iter)
    // 使用推荐配置测量数据库性能,更新调优历史信息
    m ← f(k)
    iter ← iter + 1, history ← history ∪ (k, m)
    If m > m_max do
        m_max ← m
        k_best ← k
    End If
End while
return m_max, k_best
```

下面结合算法 2,详细介绍本章结合联邦学习框架的贝叶斯旋钮调优算法:

首先,使用新场景的负载元特征与历史调优的负载元特征进行匹配,根据阈值 γ 筛选掉存在潜在分布差异的客户端经验.符合要求的客户端使用第 3 节的隐私保护方式对本地调优数据加密处理,将 RFF 向量传递给服务端.(算法 2 的 1-8 行)

然后,服务端初始化代理模型,包括初始化全局贝叶斯优化模型,以及利用接收到的 RFF 向量和约定的 w、b 参数还原出客户端的近似模型(算法 2 的 10-12 行).

算法 3: RFF 向量还原为客户端模型 *init_client_models*()

输入 **客户端经验 RFF 向量信息** ω

输出 **客户端近似模型** *client_models*

```
1  global_model ← init_model()
2  x ← global_model_sample()          //从全局高斯过程中采样
3  φ(x) ← √2 cos(w^T x + b)           // 计算全局模型核函数的 RFF 近似
4  client_models ← ∅
5  For ω_i in ω do
6    // 客户端模型初始化为具有 φ(x) 作为特征的贝叶斯线性回归模型 φ(x)^T ω_i
7    client_models ← client_models ∪ {φ(x)^T ω_i}
8  End for
9  return client_models
```

其中,算法对代理模型的具体定义如下:

代理模型. 高斯过程(Gaussian Process, GP)模型是一个随机过程,其特点是任何有限子集的随机变量都遵循多元高斯分布,由于其强大的表达能力和对不确定性的精确估计,GP 模型在贝叶斯优化中被广泛应用.它不仅能够支持噪声观测,还能提供预测的置信区间,这对于导向优化算法的决策尤为重要.

给定一组观测值 $D_t=\{(x_1,y_1),\cdots,(x_t,y_t)\}$,GP 模型利用这些数据生成后验边缘高斯分布的预测.其预测均值 $\mu_t(x)$ 和协方差 $\sigma_t^2(x,x')$ 可以表示如下,

$$\mu_t(x)=k_t(x)^T\left(K_t+\sigma^2 I\right)^{-1}y_t \tag{3}$$

$$\sigma_t^2(x,x')=k(x,x')-k_t(x)^T\left(K_t+\sigma^2 I\right)^{-1}k_t(x') \tag{4}$$

其中,σ^2 是噪声水平,$k(x,x')$ 是协方差核函数,有对应的协方差矩阵 $K_t=\left[k(x_i,x_j)\right]_{i,j=1,\cdots,t}$ 和协方差向量 $k_t(x)=\left[k(x,x_i)\right]_{i=1,\cdots,t}^T$.

客户端代理模型. 利用客户端传回的向量,服务端可以在不知道原始数据的情况下重构出近似的客户端 GP 模型.首先,将具有 RFF 近似核函数($k(x,x')\approx\varphi(x)^T\varphi(x')$)代回 GP 代理模型,可以解释 GP 为具有 $\varphi(x)$ 作为特征的贝叶斯线性回归模型: $f(x)=\varphi(x)^T\omega$.利用一个标准正态分布 $\omega\sim N(0,1)$,在观测集 D_t 上可以推导出 ω 的后验概率分布:

$$P(\omega\mid D_t)=N\left(v_t,\sigma^2\Sigma_t^{-1}\right) \tag{5}$$

其中,$\Sigma_t=\Phi(X_t)^T\Phi(X_t)+\sigma^2 I$, $v_t=\Phi(X_t)^T\Sigma_t^{-1}y_t$, $\Phi(X_t)=\left[\varphi(x_1),\cdots,\varphi(x_t)\right]^T$.

因此,可以用对后验分布 $P(\omega\mid D_t)$ 抽样 $\tilde{\omega}$(也就是客户端传回的向量),构建出 RFF 近似的后验 GP 模型 $\tilde{f}(x)=\varphi(x)^T\tilde{\omega}$.此外,对于这个近似模型,它的近似后验边缘高斯分布,预测均值和协方差可以表示如下:

$$\mu_t(x)=\varphi(x)^T v_t \tag{6}$$

$$\sigma_t^2(x)=\sigma^2\varphi(x)^T\Sigma_t^{-1}\varphi(x) \tag{7}$$

在观测集 D_t 上,通过计算近似模型 $\tilde{f}(x)$ 和服务端全局模型的肯德尔相关系数(Kendall's Tau),可以评估调优轨迹上的相似性(算法 4 的 21 行).

最后算法 2 的 14-28 行,初始化全局模型和客户端模型后,服务端开始旋钮调优.大致过程分为如下三步:

1. 基于现有的观测数据 $history=\{(k_1,m_1),\ldots,(k_{n-1},m_{n-1})\}$ 拟合全局概率代理模型 $global\_model$,其中 k_i 是第 i 次迭代中评估的配置, m_i 是对应观察到的性能;
2. 使用策略选择器选择采集方式来评估最有希望的配置 $k_n=advise\_configuration()$ （算法 4）.在调优过程中,采用了三种可选评估方式,并通过一个递增的概率数组 pt 来确保随着调优迭代的进行,选择器会逐渐倾向于使用全局模型推荐配置:
   a) 最大化全局模型的采集函数 $k_n=argmax_{x\in X}acq(k;model)$ 来选择最有希望的配置(算法 4 的 6-9 行);
   b) 综合各客户端近似模型的采集函数,将近似模型推荐的旋钮配置以近似模型与当前全局模型的相似度赋予权重,加权求和得到历史经验推荐的配置(算法 4 的 9-16 行);
   c) 为了避免模型局部最优,选择器将按固定概率随机选择配置,使算法保持最低的探索能力(算法 4 的 3-5 行).
3. 使用所选的配置 k_n 评估真实数据库性能度量 m_n ,并将结果加入观测集 $history=history\cup\{(k_n,m_n)\}$ 扩展该集合,进入下一轮调优.

算法 4: 推荐旋钮配置 *advise_configuration*()

输入 全局模型 *global_model* , 客户端模型 *client_models* , 当前调优轮次 *iter*
输出 推荐配置 *k*

```
1   pt ← 0.9×(1−exp(array(n)×(−1/c)))
2   decider ← random(0,1)
3   // 选择随机配置
4   If decider > 0.9 do
5     k ← generate_random_configuration()
6   // 使用全局模型推荐配置
7   Else if decider < pt[iter] do
8     k ← argmax acq(global_model)
9   // 通过客户端模型推荐配置
10  Else do
11    τ ← ∅, configs ← ∅
12    For model ∈ client_models do
13      τ ← τ ∪ {kendall(global_model, model, history)}
14      configs ← configs ∪ {argmax acg(model)}
15    End for
16    k ← Σ_i τ_i · configs_i
17  End If
18  return k
```

算法 4 中关于模型的采集函数定义如下:

采集函数. 采集函数确定在下一次迭代中评估的最有希望的配置.本文定义采样 EI(Expected Improvement)函数,该函数用于衡量某配置相对于当前最佳观测性能的预期改进量,其公式为,

$$EI(x)=\int_{-\infty}^{\infty}\max\left(y^{*}-y,0\right)p_{M}(y\mid x)dy \tag{8}$$

其中 y^{*} 是目前观察到的最佳性能, $EI(x)$ 就是在配置 x 时性能 y 提升的期望.下一次迭代中最有希望的配置则是使得 EI 达到最大值的 $x_{n}=argmax_{x}EI(x)$.

在公式中, $p(y\mid x)$ 是先验数据 x 下的后验概率,当模型 M 为服务端的全局模型时,本文使用 GP 模型的后验分布来计算 EI,以决定下一步的最优配置;当模型 M 被选择为客户端的本地模型时,则使用重构得到的客户端代理模型依次计算每一个 EI,并将下一步的配置按与当前任务的相似度进行加权综合.

5 实验

为了验证本文提出的联邦学习调优框架的可行性, 本文进行了实验, 以评估联邦学习算法的调优性能. 首先, 在 5.1 节介绍了实验环境设置, 实验数据集和工作负载, 对比实验 baselines, 实验测评指标. 为了展示联邦调优框架的实用性和有效性,本文设计了三方面对比实验将:1) 联邦学习调优框架的效率提升:在第 5.2 节,本节将基于联邦学习的贝叶斯调优算法与其他先进的调优算法进行比较,评估其在实际应用中的改进效果;2) 联邦学习框架调优的鲁棒性:在 5.3 节,本节将分析当云端提供历史调优数据的客户端数量变化时对调优效率的影响;3) 调优经验权重动态变化:在最后的 5.4 节,本节将研究基于联邦学习的贝叶斯调优算法框架中,各调优经验的权重随算法迭代的变化,以验证基于元特征匹配的经验筛选对客户端异构性问题的处理效果.

5.1 实验环境设置及数据集

本文在 Windows 10 操作系统上进行实验,开发环境为 Python 3.10 和 PostgreSQL 14.4.实验主机配备 Intel Core i7-7700H 处理器,32GB 内存和 1.5TB 磁盘.为了隔离数据库系统,本文将其部署在运行 Ubuntu 操作系统的 Linux 虚拟机上,该虚拟机配置了 4 核心的 CPU,16G 内存和 512G 磁盘.

测试数据集. 如表 1 所示,本文采用了 YCSB、TPCC 和 TPCH 三个基准测试工具来评估数据库系统的性能.具体来说,在 TPCC 负载中,本文设置参数 warehouse=10,loadWorkers=1 生成占用 1GB 空间的数据集.此外,为了验证本文模型在大规模数据集上调优的有效性,实验进一步进行了扩展,将机器内存限制在 4GB,并通过设置 TPCC 参数 warehouse=80,loadWorkers=1 生成占用 8GB 空间的数据集以进行调优测试.在 TPCH 负载中本文基于 19 个复杂查询模板,按比例生成含有 100 个查询的事务,并生成 8 个表总共 1GB 的数据集进行查询.本文排除了第 2、第 17 和第 20 模板,因为基于这三个模板生成的查询执行时间比其他模板生成的查询要长得多,如果不排除这三个模板将主导整个负载的成本.YCSB 负载使用以下工作负载类型生成数据集(每个数据集占用 1GB 空间):workloada(读写均衡型,50%/50% Reads/Writes)、workloadb(偏读型,95%/5% Reads/Writes)和 workloadc(纯读型,100%/0% Reads/Writes),每个工作负载使用参数 recordcount=900,000 生成数据集.

TPCC、TPCH 和 YCSB 提供了多种工作负载模板,这些基准测试封装了数据集和工作负载,有效地模拟了真实世界的数据库系统.在实验过程中,本文根据实际生产环境的需求调整了各种参数,通过细致处理和分析实验数据,得出了可靠的结果和结论.

对比方法. 现有工作使用的旋钮优化方法可以分为三类:(i) 基于搜索的方法,(ii) 基于强化学习(Reinforcement Learning,RL)的方法,以及 (iii)基于贝叶斯优化(Bayesian Optimization,BO)的方法.在基于搜索的方法中,BestConfig[28]是一种常用的示例.它将搜索空间划分为多个部分,通过分支定界策略探索.虽然该方法可以快速选择要评估的下一个点,但由于不适用先验知识库,其性能受到了限制[29].

实验中,本文将基于联邦学习的贝叶斯调优算法与以下调优算法进行比较:(1) DDPG(Deep Deterministic

Policy Gradient)[1]:利用深度确定性策略梯度算法,Actor 基于 DBMS 内部指标训练深度神经网络,并提出平衡探索和利用的配置.(2) BO(Bayesian Optimization)[29]:贝叶斯调优算法,基于普通贝叶斯优化来寻找数据库最佳配置.(3) CBO(Centralized Bayesian Optimization):集中式贝叶斯调优算法,在单机上利用多个贝叶斯优化提供的历史经验进行调优,用于验证基于联邦学习的贝叶斯调优算法的精度损失.

表 1 实验数据集

No.	Workload	Scale	Transaction Ratios
1	TPCC	1GB	[45, 43, 4, 4, 4]
2	YCSB-A	1GB	[0.50, 0.50, 0, 0]
3	YCSB-B	1GB	[0.95, 0.05, 0, 0]
4	YCSB-C	1GB	[1, 0, 0, 0]
5	TPCH	1GB	[3, 0, 5, 6, 5, 5, 5, 3, 5, 3, 8, 4, 5, 5, 8, 5, 0, 5, 5, 0, 5, 10]
6	TPCC-8G	8GB	[45, 43, 4, 4, 4]

测试指标. 在 DBMS 优化的情景下,主要目标是是最大化性能,例如吞吐量和延迟(如 95% 延迟),这些指标旨在衡量 DBMS 有效处理大量工作负载或查询的能力.实验中为了全面深入地检验本文设计的联邦学习调优框架的实际表现,本文制定了一系列关键的实验指标.核心指标是**吞吐量(Throughput)**和**调优用时(Runtime)**.在本实验中,吞吐量具体指数据库在处理特定工作负载时,每秒钟能够执行的操作数量,这是衡量旋钮性能的重要标准;而调优用时则表示达到最优吞吐量所消耗的时间,反映调优的收敛速度.通过设定这些指标,本章能够全面深入地评估不同旋钮调优方法,从而准确地揭示其调优效果的优劣.

5.2 针对不同方法调优效率的对比实验

本章在三种不同的数据库基准测试(TPCC 负载、YCSB 负载和 TPCH 负载)上训练三种旋钮调优模型:DDPG,BO,FBO.表 2 展示了三种调优方法在实验中的效率,包括 optimal-throughout(最优吞吐量)和 optimal-runtime(达到最优吞吐量所消耗的时间).具体来说,本文设计的 FBO 方法能够大大提高云数据库场景下旋钮调优的效率,能在更短的时间开销下,达到更高的负载吞吐量.例如,在 YCSB-A 负载中,FBO 方法仅仅消耗 145.46s 就能够达到 1714.32 ops/s 的吞吐量,而基于普通 BO 和 DDPG 的调优模型需要消耗更多的时间来完成调优.

在表 2 中,TPCH 负载上的调优过程相比其他基准测试更为缓慢,主要原因在于 TPCH 负载的查询执行耗时更长.该负载的查询复杂性对每个查询进行细致的分析和优化提出了更高的要求,并且部分极其耗时的查询造成了长尾效应.首先,TPCH 包含 22 个高度复杂的查询,涵盖了各种类型的复杂查询,包括聚合查询、连接查询、嵌套子查询等.例如,Q1 和 Q10 中涉及大量的分组和聚合运算,而 Q20 和 Q22 则包含多层相关的嵌套子查询.这种复杂性使得数据库对旋钮参数的设置极为敏感,不同的旋钮配置可能对不同的查询产生显著的影响.其次,TPCH 基准测试还表现出明显的长尾效应,即少数几个查询占用了大部分的执行时间.优化这些长尾查询通常涉及更多的旋钮,需要更多的时间和资源,导致调优周期延长.不同查询的资源需求各异,有些查询可能是 CPU 密集型,而另一些则可能是 I/O 密集型,导致难以实现统一的旋钮优化策略.这不仅增加了优化的难度,耗费更多资源也可能对其他查询的性能产生负面影响.

表 2 三种调优方法的调优效率对比实验

Benchmark	Default Knobs	DDPG		BO		FBO	
	Throughput (ops/s)	Throughput (ops/s)	Runtime (s)	Throughput (ops/s)	Runtime (s)	Throughput (ops/s)	Runtime (s)
TPCC	2462.27	2479.02	155.09	2602.25	174.38	2566.45	161.76

YCSB-A	1609.50	1708.02	851.38	1727.71	499.33	1714.32	145.46
YCSB-B	2763.59	2914.97	841.44	3002.22	323.76	2999.53	294.56
YCSB-C	3274.39	3511.31	402.03	3522.98	445.61	3525.31	385.98
TPCH	0.106	7.068	6240.38	7.096	1168.55	7.144	2267.91

在图 4 中展示了 TPCC 负载中采用不同调优方法所得到的最大吞吐量以及达到最优吞吐量的用时.实验结果可以看到,采用 FBO 模型进行调优后,最终吞吐量上达到了 2566.45 ops/s,相比默认旋钮配置的 2462.27 ops/s 有约 4.2%的提升,这使得其性能达到了 BO 模型 98.6%.因为 TPCC 上的负载相对复杂,在联邦客户端数量有限的情况下,FBO 可能需要更多的迭代才能达到更接近普通调优方法的性能.相比之下,DDPG 模型提升较小,吞吐量仅从 2462.27 ops/s 略增至 2479.02 ops/s,性能提升仅 0.68%,这可能由于所使用的神经网络的复杂性,基于 RL 的方法还需要更多的迭代.

在迭代开始阶段,观察到 FBO 选择的配置要优于 BO 的情况.这是由于 FBO 实现基于历史调优经验的学习,能够在短时间内利用经验获得较佳的性能配置,启动后无需多次探索,而普通的 BO 方法还需要在从先验数据开始拟合不太准确的模型,然后进行迭代优化探索,因此其初始配置性能较差.

在时间效率上,本文设计的 FBO 模型在达到最大吞吐量所需的时间相比于单机上 BO 的 174.38s 减少了约 7.23%,这说明联邦学习的历史经验有在不调优的迭代过程实现加速.

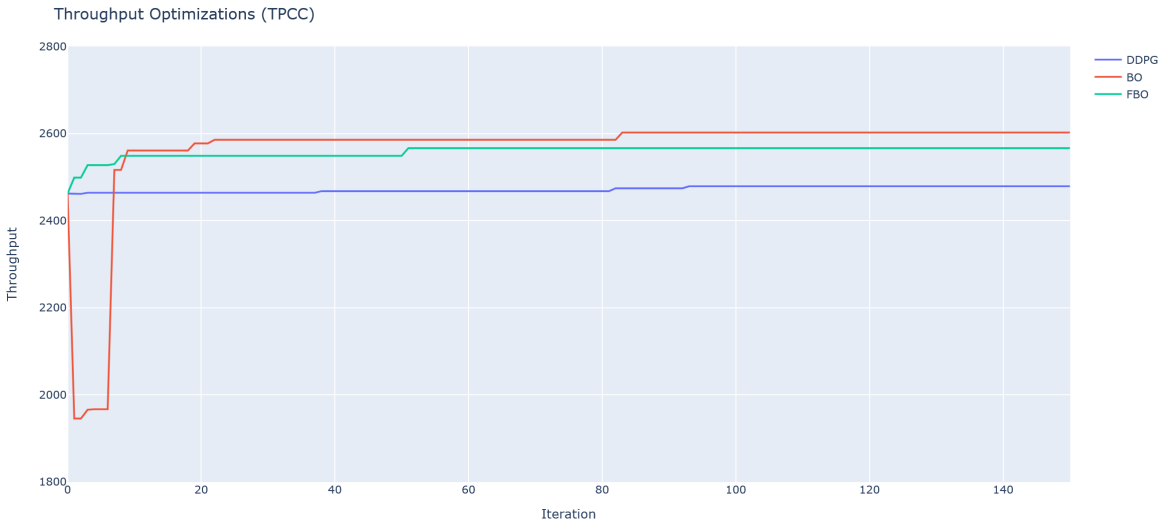


图 4 TPCC 负载上的旋钮调优过程中吞吐量随迭代变化

如图 5 到图 7 所示,它们展示了各种 YCSB 负载上各种旋钮调优方法的最大吞吐量和所需时间.以 YCSB-A 为例,可以看出 FBO 模型的最终调优效果在吞吐量上达到了 1714.32 ops/s,对于默认配置的 1609.50 ops/s 提升约 6.5%的性能,接近单机上 BO 的 1727.71 ops/s.与 TPCC 类似,由于 FBO 可以综合历史调优经验,因此在开始迭代时就可以探索达到相近于 BO 最终性能的效果,并能保持稳定收敛.此外,DDPG 也有 1708.02 ops/s 的结果,提升约 6.1%,不过在有限的迭代次数内,效果仍然未达到基于贝叶斯优化调优方法的性能.时间效率上,FBO 用时 145.46s,相比 BO 的 499.33s 有明显提升,这表明其在迭代中能够快速达到较好的收敛状态.同时 BO 模型在收敛方面表现都比 DDPG 更为出色,后者达到最大吞吐量需要 851.38s,也印证了采用贝叶斯模型可以避免神经网络复杂导致收敛缓慢.

而在 YCSB-B 和 YCSB-C 工作负载下,FBO 模型的最终调优效果在吞吐量上也分别达到了 2999.53 ops/s、3515.31 ops/s,相比默认配置的 2763.59 ops/s、3274.39 ops/s 对应有提升约 8.5%、6.9%的性能.同时,消耗时间均略低于 BO,表明在这两个工作负载上,利用各历史调优经验的 FBO 可以实现更快的稳定收敛.

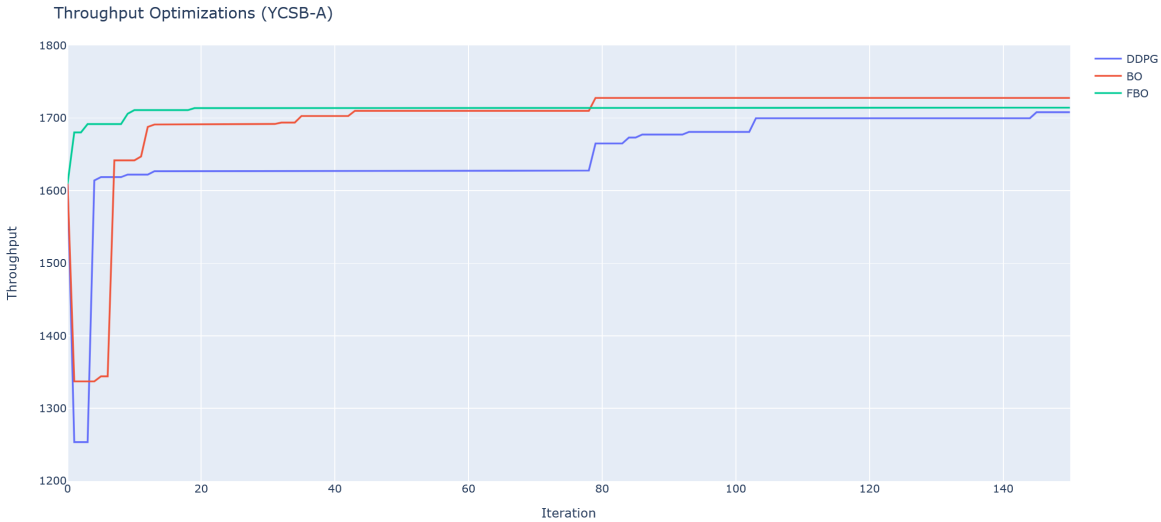


图 5 YCSB-A 负载上的旋钮调优过程中吞吐量随迭代变化

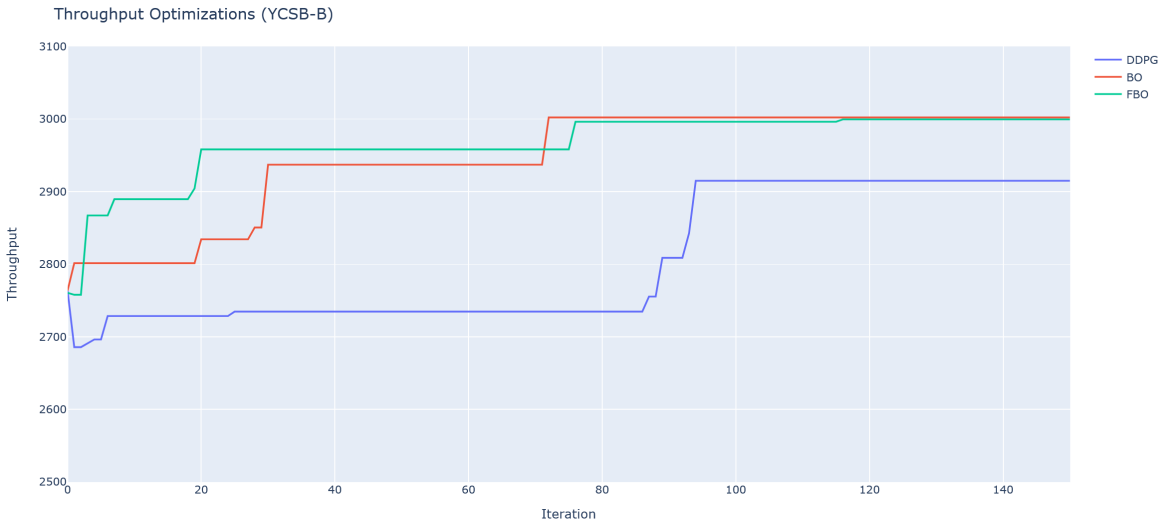


图 6 YCSB-B 负载上的旋钮调优过程中吞吐量随迭代变化

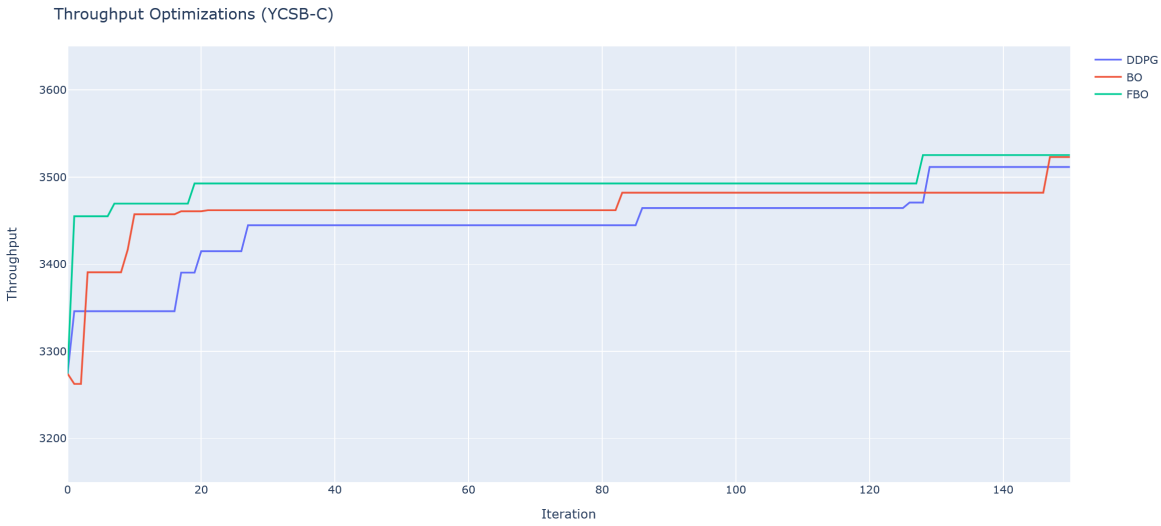


图 7 YCSB-C 负载上的旋钮调优过程中吞吐量随迭代变化

在图 8 中,本文进一步分析了 TPCH 负载中四种旋钮调优方法的最大吞吐量和所需时间性能表现.从实验结果可以看出,由于 TPCH 上负载的复杂性,默认旋钮配置的性能并不是很好,而经过各调优方法的优化,性能结果相较默认配置都有显著提升.在吞吐量方面,本文的 FBO 模型在最终调优效果上实现了 7.144 ops/s 的吞吐量,优于单机贝叶斯优化 BO 的 7.096 ops/s 和强化学习 DDPG 的 7.068 ops/s,在复杂查询场景表现良好.

在时间效率上,FBO 模型的调优耗时为 2267.91s,明显少于 DDPG 的 6240.38s.虽然耗时长于 BO 模型,但是能观察到在迭代初期阶段,联邦学习框架能够在短时间内利用经验获得较佳的性能配置.启动后无需多次探索试错,FBO 即可达到与 BO 相近 20 次迭代后相近的调优效果.

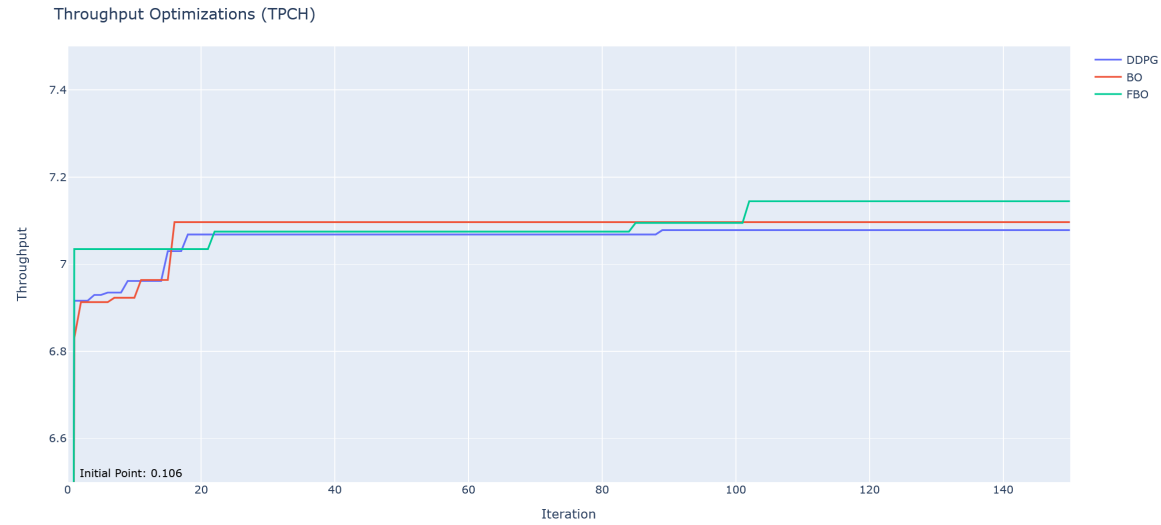


图 8 TPCH 负载上的旋钮调优过程中吞吐量随迭代变化

表 3 联邦学习调优与集中式调优的对比实验

Benchmark	Default Knobs	CBO		FBO	
	Throughput (ops/s)	Throughput (ops/s)	Runtime (s)	Throughput (ops/s)	Runtime (s)
YCSB-A	1609.50	1729.68	530.82	1714.32	145.46
YCSB-B	2763.59	3000.69	264.13	2999.53	294.56
YCSB-C	3274.39	3538.77	401.77	3525.31	385.98
TPCH	0.106	7.207	1994.52	7.144	2267.91

更多地,本文还对基于联邦学习的贝叶斯调优算法(FBO)和集中式调优方法(CBO)进行了比较.联邦学习框架下的贝叶斯调优算法需要对数据进行隐私保护处理,并扩展代理模型以实现中间参数传递,集中式贝叶斯优化则可以视为在单机上不进行隐私保护,直接使用多个客户端历史经验数据的架构.

表 3 的实验数据表明,在各种负载情况下,FBO 相对于 CBO 模型的精度损失较小,能达到和集中式匹配的性能.以 YCSB-A 为例,FBO 模型的最终调优效果在吞吐量上达到了 1714.32 ops/s,相比默认配置的 1609.50 ops/s 提升约 6.5%的性能,与集中式的贝叶斯优化相比精度损失只有 0.89%.这表明,尽管联邦学习方法为了保护隐私可能会带来一定的数据失真,但其整体性能够与集中式方法保持非常接近的水平.

在时间开销方面,FBO 与 CBO 在大多数负载情况下表现相当.特别是在 YCSB-A 和 YCSB-C 负载中,FBO 的时间开销甚至低于 CBO.虽然在 YCSB-B 和 TPCH 负载中,FBO 的时间开销略有增加,但总体上仍然在可接受范围内.尽管隐私保护带来了一定的数据失真,FBO 仍然能够在保持高吞吐量和较低时间开销的情况下,取得接近集中式调优的效果,说明了 FBO 框架在实际应用中具有较好的实用性和有效性.

为了验证基于联邦学习的贝叶斯调优算法在大规模数据集上的有效性,本文单独在 4G 内存系统上对 8G 数据量的 TPCC 负载进行调优实验.调优结果如表 4 所示,该负载下默认配置吞吐量为 3435.45 ops/s,使用 FBO 模型调优后,吞吐量提升至 3456.70 ops/s,优于 DDPG 模型的 3443.46 ops/s.此外,与集中式的 CBO 模型相比,FBO 模型的精度损失仅为 0.16%,基本到达集中式调优的水平,展现出较高的性能.

而在该数据集调优的时间开销上,实验结果仍然表明,在实验设置相同的调优迭代次数下,基于贝叶斯优化的方法优于基于强化学习的 DDPG 方法.尤其是利用历史经验的 CBO 和 FBO,调优速度更为显著.在图 9 的吞吐量变化曲线显示,CBO 和 FBO 均在前 10 轮迭代时就已达到与 BO 在 50 轮迭代时相近的吞吐量性能,并在 40 轮迭代左右达到了接近 BO 最终的吞吐量性能.相比之下,本文的 FBO 模型具有更高的效率.

表 4 不同调优方法在 TPCC-8G 数据集上的调优对比实验

Benchmark	Throughput (ops/s)	Runtime (s)
DDPG	3443.46	11263.06
BO	3452.58	8767.62
CBO	3451.03	6647.43
FBO	3456.70	5530.60

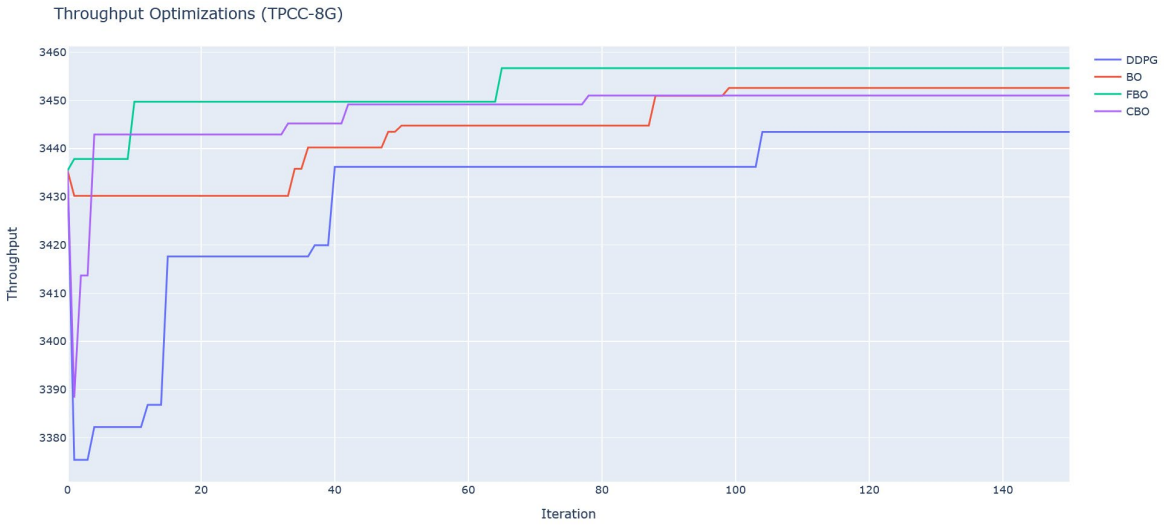


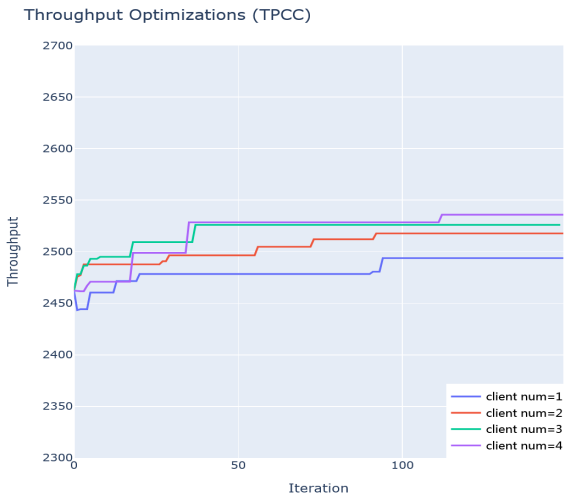
图 9 TPCC-8G 负载上的旋钮调优过程中吞吐量随迭代变化

5.3 针对基于联邦学习的贝叶斯调优算法鲁棒性的对比实验

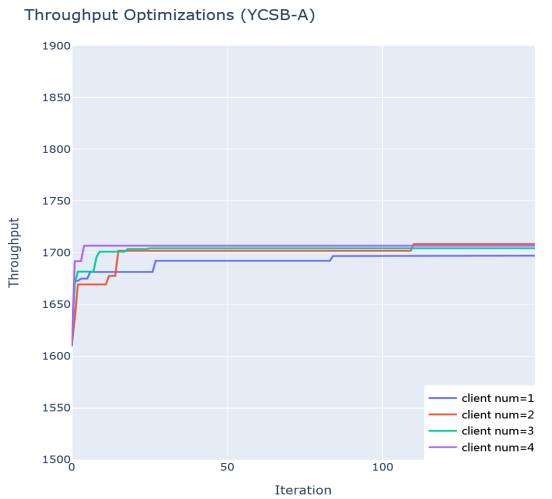
本章首先训练了四个客户端的调优数据,以探究不同经验数量对模型的性能调优能力影响.如图 10 中,在经过工作负载元特征匹配筛选后,FBO 可以使用不同数量的客户端经验进行旋钮调优任务.

在图 10 (a)中,本节研究了 TPCC 负载环境下,不同客户端数量对联邦贝叶斯优化调优性能的具体影响.实验结果详细展示了从单客户端到四客户端不同配置下的吞吐量变化情况:在单客户端环境下,吞吐量在 2440-2500 ops/s 范围内,由于数据历史经验有限,模型可能受到单个客户端的经验过多影响,导致无法有效拟合出准确的性能代理函数,需多次迭代以适应.引入第二个客户端后,利用历史经验得到的初始配置性能更好了,同时吞吐量迅速提升至约 2520 ops/s,显示出增加历史经验可以加速模型的早期学习.随着更多客户端的加入,吞吐量更快稳定在 2526 ops/s,模型能够利用更丰富的历史经验进行调优,提升调优效率并增强了迭代的稳定性.

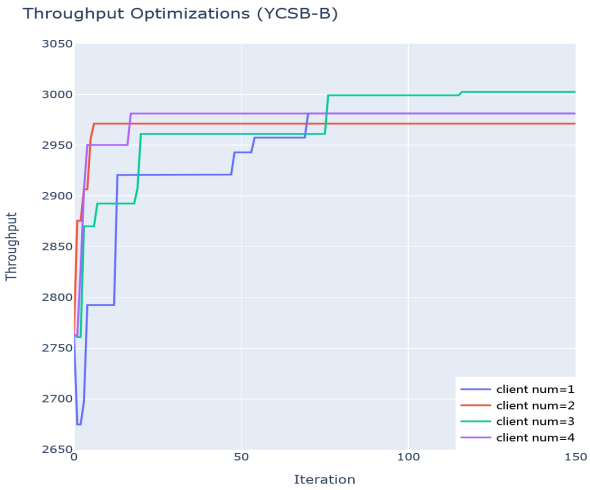
在 YCSB 负载中,不同客户端数量对 FBO 调优性能的影响如图(b)-(d)所示.相比 TPCC,可能是 YCSB 的负载测试直接根据读写比例生成的测试更单纯,使得客户端经验的异构性小.实验数据显示,YCSB-A 与 YCSB-B 在增加客户端经验数量时最终调优吞吐量没有明显增加,都稳定在约 1705 ops/s 和 2970 ops/s 附近.这与 5.3 节实验中 YCSB 负载下 FBO 调优性能已经接近 BO 效果的结论一致,说明此时以较少的客户端经验即可达到不错的调优结果.随着客户端数量增加,根据经验推荐的初始配置性能在提高,达到最终收敛的轮数也在减少,说明增加客户端数量仍然能带来调优效率提升.而对于纯读的 YCSB-C,由于本章使用的四个可选客户端异构性较大,可以观察到从一个客户端到四个客户端最终调优性能提升较为显著.



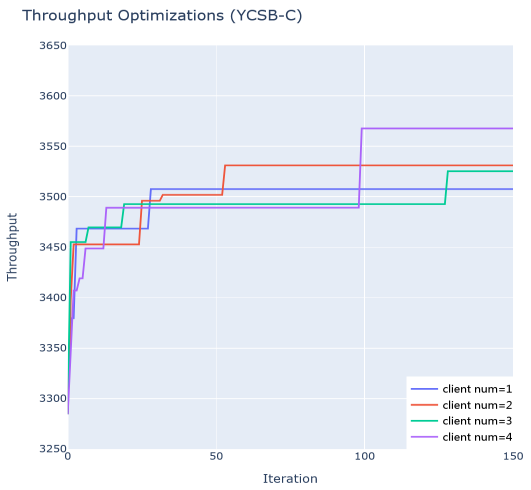
(a) TPCC 负载



(b) YCSB-A 负载



(c) YCSB-B 负载



(d) YCSB-C 负载

图 10 不同历史经验数量的调优过程吞吐量变化

5.4 动态权重分析

为了更清晰地展示联邦学习内部迭代决策过程,本节在图 11 呈现了优化 TPCC 时使用的四个客户端经验权重.结合图 10 (a)的 client_num=4,可以观察到客户端模型的权重变化如何影响整体吞吐量.在调优的初期阶段,到大约 25 轮迭代,各客户端的吞吐量显示出显著的波动,这反映了模型在尝试不同配置下的适应性.随着调优的深入,大约到 40 轮迭代左右,算法逐渐倾向于增加与当前调优目标负载更相似的客户端权重.这时整体模型到达一个新的最大吞吐量,同时由于策略选择器增加了全局模型的概率,客户端权重的变化趋于稳定.最终,各客户端的权重相差不大,这与本章选择客户端时采用相近但不同负载设置相符合.

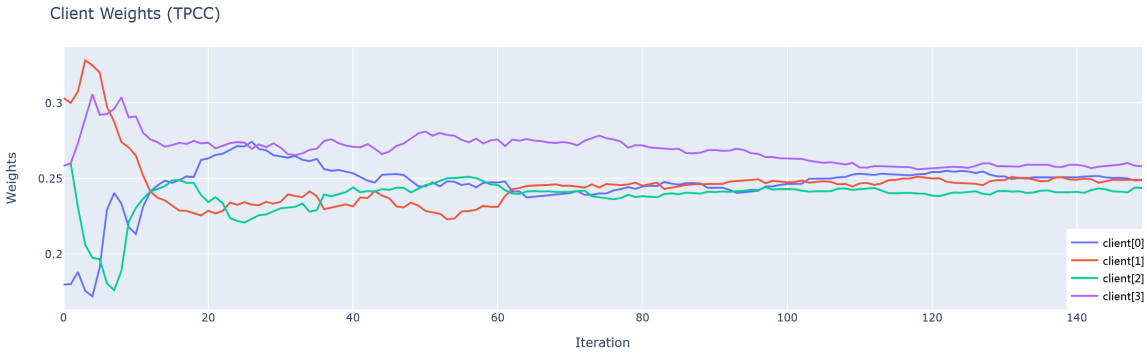
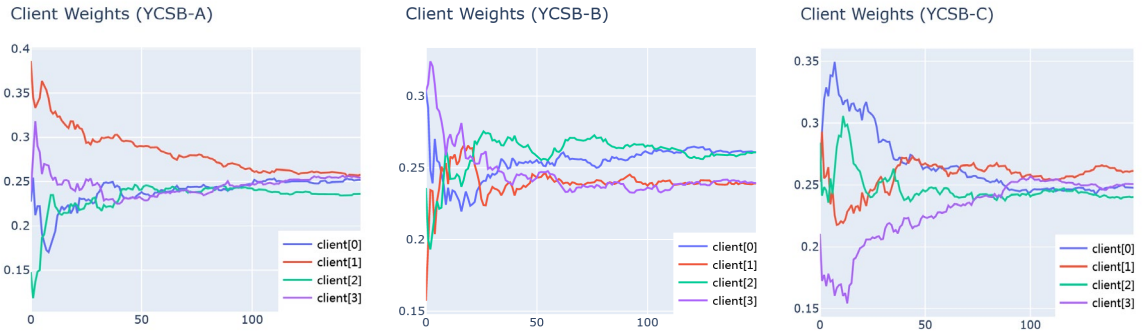


图 11 TPCC 负载上各客户端经验权重变化

如图 12,在 YCSB 工作负载上,各客户端经验权重动态变化趋势类似.随着迭代的进行,客户端经验之间的权重逐渐趋于稳定,说明模型逐渐找到了适应该负载的最佳配置.



(a) YCSB-A 负载 (b) YCSB-B 负载 (c) YCSB-C 负载

图 12 YCSB 负载上各客户端经验权重变化

6 总结与展望

本文针对云数据库旋钮调优中的隐私保护问题,提出了一种基于联邦学习的云数据库旋钮调优技术. 通过引入元特征匹配的经验筛选方法,有效解决了联邦学习中数据异构问题,并提高了学习效率. 同时,本文创新地结合云数据库服务特性,提出了以节点端为训练中心的联邦贝叶斯调优算法,利用随机傅里叶特征扩展联邦学习框架,实现了无梯度参数任务的联邦学习,并能在保护用户隐私的前提下保证经验不失真. 实验结果表明,该方法在多个工作负载下表现出显著优势. 未来本文计划在联邦学习贝叶斯调优算法的基础上,进一步探讨随机傅里叶特征在参数传递过程中的隐私保护能力与对应的优化技术,特别是其在多旋钮高维数据场景中的适用性与稳定性.同时,需要深入研究调优算法在隐私保护的调优效率直接的权衡,并探索自动化实现这种权衡的方法.具体而言,可以设计一种自适应机制,根据不同的工作负载元特征信息和隐私需求,动态调整算法的参数,以实现最佳的隐私保护和调优效率.此外,还可以考虑在更广泛和更真实的数据库环境中进行实验,以验证本文方法的普适性和实用性,从而推动基于联邦学习的云数据库调优技术的发展和应用.

References:

[1] Zhang J, Liu Y, Zhou K, Li G, Xiao Z, Cheng B, Xing J, Wang Y, Cheng T, Liu L, Ran M, Li Z. An end-to-end automatic cloud database tuning system using deep reinforcement learning. In: Proceedings of the 2019 International Conference on Management of Data. Amsterdam Netherlands: ACM, 2019: 415–432. [doi:10.1145/3299869.3300085]

[2] Rasmussen CE, Williams CKI. Gaussian processes for machine learning. The MIT Press, 2005. [doi:10.7551/MITPRESS/3206.001.0001]

[3] Cai B, Liu Y, Zhang C, Zhang G, Zhou K, Liu L, Li C, Cheng B, Yang J, Xing J. HUNTER: an online cloud database hybrid tuning system for personalized requirements. In: Proceedings of the 2022 International Conference on Management of Data. Philadelphia PA USA: ACM, 2022: 646–659. [doi:10.1145/3514221.3517882]

[4] Shen Y, Ren X, Lu Y, Jiang H, Xu H, Peng D, Li Y, Zhang W, Cui B. Rover: an online spark sql tuning service via generalized transfer learning. In: Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. Long Beach CA USA: ACM, 2023: 4800–4812. [doi:10.1145/3580305.3599953]

[5] Zhang X, Wu H, Chang Z, Jin S, Tan J, Li F, Zhang T, Cui B. ResTune: resource oriented tuning boosted by meta-learning for cloud databases. In: Proceedings of the 2021 International Conference on Management of Data. Virtual Event China: ACM, 2021: 2102–2114. [doi:10.1145/3448016.3457291]

[6] Liu YX, Chen H, Liu YH, Li CP. Privacy-preserving Techniques in Federated Learning. Ruan Jian Xue Bao/Journal of Software, 2022, 33(3): 1057−1092 (in Chinese). [doi:10.13328/j.cnki.jos.006446]

[7] Van Aken D, Pavlo A, Gordon GJ, Zhang B. Automatic database management system tuning through large-scale machine learning. In: Proceedings of the 2017 ACM International Conference on Management of Data. Chicago Illinois USA: ACM, 2017: 1009–1024. [doi:10.1145/3035918.3064029]

[8] Duan S, Thummala V, Babu S. Tuning database configuration parameters with ituned. In: Proceedings of the VLDB Endowment, 2009, 2(1): 1246–1257.[doi:10.14778/1687627.1687767]

[9] Sullivan DG, Seltzer MI, Pfeffer A. Using probabilistic reasoning to automate software tuning. ACM SIGMETRICS Performance Evaluation Review, 2004, 32(1): 404–405. [doi:10.1145/1012888.1005739]

[10] Zhang X, Wu H, Li Y, Tan J, Li F, Cui B. Towards dynamic and safe configuration tuning for cloud databases. In: Proceedings of the 2022 International Conference on Management of Data. 2022: 631–645. [doi:10.1145/3514221.3526176]

[11] Van Aken D, Yang D, Brillard S, Fiorino A, Zhang B, Bilien C, Pavlo A. An inquiry into machine learning-based automatic configuration tuning services on real-world database management systems. In: Proceedings of the VLDB Endowment, 2021, 14(7): 1241–1253. [doi:10.14778/3450980.3450992]

[12] Tan J, Zhang T, Li F, Chen J, Zheng Q, Zhang P, Qiao H, Shi Y, Cao W, Zhang R. IBTune: individualized buffer tuning for large-scale cloud databases. In: Proceedings of the VLDB Endowment, 2019, 12(10): 1221–1234. [doi:10.14778/3339490.3339503

[13] Li G, Zhou X, Li S, Gao B. QTune: a query-aware database tuning system with deep reinforcement learning. In: Proceedings of the VLDB Endowment, 2019, 12(12): 2118–2130. [doi:10.14778/3352063.3352129]

[14] Chaudhuri K, Monteleoni C. Privacy-preserving logistic regression. In: Proceedings of the 21st International Conference on Neural Information Processing Systems. Red Hook, NY, USA: Curran Associates Inc., 2008: 289–296. [doi:10.5555/2981780.2981817]

[15] Wu Y, Cai S, Xiao X, Chen G, Ooi BC. Privacy preserving vertical federated learning for tree-based models. In: Proceedings of the VLDB Endowment, 2020, 13(12): 2090–2103. [doi:10.14778/3407790.3407811]

[16] Abspoel M, Escudero D, Volgushev N. Secure training of decision trees with continuous attributes. In: Proceedings on Privacy Enhancing Technologies, 2021, 2021(1): 167–187. [doi:10.2478/popets-2021-0010]

[17] Abadi M, Chu A, Goodfellow I, McMahan HB, Mironov I, Talwar K, Zhang L. Deep learning with differential privacy. In: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security. 2016: 308–318. [doi:10.1145/2976749.2978318]

[18] Fredrikson M, Lantz E, Jha S, Lin S, Page D, Ristenpart T. Privacy in pharmacogenetics: an end-to-end case study of personalized warfarin dosing. In: Proceedings of the ... USENIX Security Symposium. UNIX Security Symposium, 2014, 2014: 17–32.

[19] Fredrikson M, Jha S, Ristenpart T. Model inversion attacks that exploit confidence information and basic countermeasures. In: Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security. Denver Colorado USA: ACM, 2015: 1322–1333. [doi:10.1145/2810103.2813677]

[20] Shokri R, Stronati M, Song C, Shmatikov V. Membership inference attacks against machine learning models. 2017 IEEE Symposium on Security and Privacy (SP). San Jose, CA, USA: IEEE, 2017: 3–18. [doi:10.1109/SP.2017.41]

[21] Aono Y, Hayashi T, Trieu Phong L, Wang L. Scalable and secure logistic regression via homomorphic encryption. In: Proceedings of the Sixth ACM Conference on Data and Application Security and Privacy. New Orleans Louisiana USA: ACM, 2016: 142–144. [doi:10.1145/2857705.2857731]

[22] Dai Z, Low BKH, Jaillet P. Federated bayesian optimization via thompson sampling. In: Proceedings of the 34th International Conference on Neural Information Processing Systems. Red Hook, NY, USA: Curran Associates Inc., 2020: 9687–9699. [doi:10.5555/3495724.3496536]

[23] Rahimi A, Recht B. Random features for large-scale kernel machines. In: Proceedings of the 20th International Conference on Neural Information Processing Systems. Red Hook, NY, USA: Curran Associates Inc., 2007: 1177–1184. [doi:10.5555/2981562.2981710]

[24] Le Q, Sarlós T, Smola A. Fastfood: approximating kernel expansions in loglinear time. In: Proceedings of the 30th International Conference on International Conference on Machine Learning - Volume 28. Atlanta, GA, USA: JMLR.org, 2013: III-244-III–252. [doi:10.5555/3042817.3042964]

[25] Dai B, Xie B, He N, Liang Y, Raj A, Balcan M-F, Song L. Scalable kernel methods via doubly stochastic gradients. In: Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2. Cambridge, MA, USA: MIT Press, 2014: 3041–3049. [doi:10.5555/2969033.2969166]

[26] Zhao J, Meng D. FastMMD: ensemble of circular discrepancy for efficient two-sample test. Neural Computation, 2015, 27(6): 1345–1372. [doi:10.1162/NECO_a_00732]

[27] Bracewell R, Kahn PB. The fourier transform and its applications. American Journal of Physics, 1966, 34(8): 712–712.[doi:10.1119/1.1973431]

[28] Zhu Y, Liu J, Guo M, Bao Y, Ma W, Liu Z, Song K, Yang Y. BestConfig: tapping the performance potential of systems via automatic configuration tuning. In: Proceedings of the 2017 Symposium on Cloud Computing. 2017: 338–350. [doi:10.1145/3127479.3128605]

[29] Cereda S, Valladares S, Cremonesi P, Doni S. CGPTuner: a contextual gaussian process bandit approach for the automatic tuning of it configurations under varying workload conditions. In: Proceedings of the VLDB Endowment, 2021, 14(8): 1401–1413. [doi:10.14778/3457390.3457404]

附中文参考文献:

[6] 刘艺璇,陈红,刘宇涵,李翠平.联邦学习中的隐私保护技术.软件学报,2022,33(3):1057-1092 [doi:10.13328/j.cnki.jos.006446]