软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2021,32(3):604–621 [doi: 10.13328/j.cnki.jos.006182]

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

支撑机器学习的数据管理技术综述*

崔建伟[1,2], 赵 哲[1,2], 杜小勇[1,2]

[1](数据工程与知识工程教育部重点实验室(中国人民大学),北京 100872)

[2](中国人民大学 信息学院,北京 100872)

通讯作者: 杜小勇, E-mail: duyong@ruc.edu.cn

摘 要: 应用驱动创新,数据库技术就是在支持主流应用的提质降本增效中发展起来的.从 OLTP、OLAP 到今天的在线机器学习建模无不如此.机器学习是当前人工智能技术落地的主要途径,通过对数据进行建模而提取知识、实现预测分析.从数据管理的视角对机器学习训练过程进行解构和建模,从数据选择、数据存储、数据存取、自动优化和系统实现等方面,综述了数据管理技术的应用及优缺点,在此基础上,提出支持在线机器学习的数据管理技术的若干关键技术挑战.

关键词: 人工智能;机器学习;数据管理

中图法分类号: TP311

中文引用格式: 崔建伟,赵哲,杜小勇.支撑机器学习的数据管理技术综述.软件学报,2021,32(3):604–621. http://www.jos.org.cn/1000-9825/6182.htm

英文引用格式: Cui JW, Zhao Z, Du XY. Survey on data management technology for machine learning. Ruan Jian Xue Bao/ Journal of Software, 2021,32(3):604–621 (in Chinese). http://www.jos.org.cn/1000-9825/6182.htm

Survey on Data Management Technology for Machine Learning

CUI Jian-Wei[1,2], ZHAO Zhe[1,2], DU Xiao-Yong[1,2]

[1](Key Laboratory of Data Engineering and Knowledge Engineering, MOE (Renmin University of China), Beijing 100872, China)

[2](School of Information, Renmin University of China, Beijing 100872, China)

Abstract: Applications drive innovation. The advance of database technology is achieved in support of development of mainstream applications effectively and efficiently. OLTP, OLAP, and online machine learning modeling today all follow this trend. Machine learning extracts knowledge and realizes predictive analysis by modeling data, is the main approach of artificial intelligence technology. This work studies the training process of machine learning from the perspective of data management, summarizes data management technology through data selection, data storage, data access, automatic optimization, and system implementation, and analyzes the advantages and disadvantages of these techniques. Based on the analysis, this study proposes key challenges of data management technology for online machine learning.

Key words: artificial intelligence; machine learning; data management

人工智能技术的目标是让机器在某些方面具备人类的智能能力,从而辅助人类完成各项复杂任务.数据、算力和模型是公认的人工智能三要素.近年来,数据规模的快速增长,GPU 等计算设备迭代迅速,计算能力快速提升;机器学习模型、尤其是基于神经网络的深度学习技术不断演进.人工智能在图像识别[1]、语音识别[2]以及智能问答[3]等领域的多项任务效果已经超过真实人类水平;与此同时,多家科技企业也将人工智能作为公司重

* 基金项目: 国家自然科学基金(62072458)
Foundation item: National Natural Science Foundation of China (62072458)
本文由“支撑人工智能的数据管理与分析技术”专刊特约编辑陈雷教授、王宏志教授、童咏昕教授、高宏教授推荐.
收稿时间: 2020-07-20; 修改时间: 2020-09-03; 采用时间: 2020-11-06; jos 在线出版时间: 2021-01-21

要战略.

机器学习是目前实现人工智能的主要途径,是通过对数据进行建模而提取知识的过程.传统的数据管理技术主要支持结构化数据交易型应用 OLTP(online transaction processing)、结构化和半结构化数据的分析型应用 OLAP(online analytics processing)等;非结化数据因为 4V[4]的特点,目前尚未形成统一的数据管理方案.目前,通过机器学习技术对非结构化数据建模而构建具备预测能力的模型,是非结构化数据分析的重要手段,因而,机器学习技术可以理解为是 OLTP、OLAP 等之后的自然延伸,思考支持机器学习的数据库技术正当其时.

另一方面,机器学习任务的一次训练过程包括数据选择、特征抽取、算法选择、超参数调优、效果评测等多个子过程;而在训练结束获得效果评测后,通常需要人工对模型效果进行分析,挖掘模型效果与数据、特征和算法之间关联,并基于数据分析或者人工经验对训练子过程进行调整,重新训练以提升模型效果;这个过程通常会多次循环迭代进行,直到模型效果达到应用要求.显然,相比与数据库系统的查询与分析任务,机器学习任务要复杂得多.由于机器学习训练子过程和迭代调整次数较多,且许多子过程需要人工参与,机器学习的训练过程目前仍然采取以任务为中心的做法,根据任务的特点对训练子过程进行定制化优化.这种做法人工参与成本较高,且无法在多任务之间实现数据、特征、模型等资源的复用,因而存在成本高、效率低、能耗大[5]的问题.如何降低机器学习过程的成本,提高机器学习建模的效率,就成为一个重要的需求.

数据管理技术,尤其是数据库和数据仓库系统经过多年的发展,形成了一套独特的方法论,包括以数据模型为核心、以层次化结构实现数据与应用之间的独立性、提供面向任务的描述性语言、通过查询优化技术提升任务执行效率等.从数据管理的角度看,机器学习各个子过程均涉及不同类型的数据读写、转化和计算,具有显著的数据管理与分析需求.将数据库和数据仓库的关键技术和系统构建经验应用于机器学习训练过程,有助于对机器学习形成系统的管理方案,从而提升整体效率.本文对支撑机器学习的主要数据管理技术进行整理和归纳,并提出若干研究课题.

本文第 1 节概述人工智能与机器学习的发展历史、机器学习主要类别.第 2 节从数据管理和系统构建的视角对机器学习训练过程进行解构,提出研究框架.第 3 节~第 7 节将具体介绍支撑机器学习的数据管理关键技术,包括数据选择、数据存储、数据存取、自动优化、以及系统构建的若干模式.最后,展望支撑机器学习的数据库系统的挑战与未来研究方向.

1 机器学习的建模过程

1.1 机器学习发展历史

人工智能的历史可以回溯到上世纪 60 年代,早期实现人工智能主要是基于规则的方法,对特定领域建立专家系统.比如:ELIZIA[6]通过情绪疏导领域的对话规则总结,最早实现了人机对话;早期的机器翻译系统也主要是语言学专家通过总结不同语言之间的映射规则而建立.这一时期,人工智能技术尚缺乏主流的应用,主要有两方面的原因:一是基于规则的方法通常只能处理已出现的请求,不具备通用预测能力,对新请求的预测效果较差;二是基于规则的方法需要人工总结大量规则,其构建成本高且难以维护.

进入 20 世纪 90 年代,随着数据不断积累以及统计模型持续改进,机器学习,也就是通过从已有的数据和经验中构建具备预测能力的模型,成为实现人工智能的主要途径,并在许多领域逐渐实现了效果突破.比较有代表性的工作包括基于统计模型的语音识别系统的识别错误率显著降低、基于统计机器翻译的效果有了显著提升.相对于基于规则的方法,机器学习方法优势显著:一方面,机器学习模型的通用预测能力较强,能够从数据中学习到隐藏的模式,更好地响应未出现的新请求;另一方面,机器学习通过从数据中直接学习知识,相对于人工总结规则,维护成本大大降低;通过不断的积累和清洗数据,通常能持续维护和提升机器学习模型的效果.这一时期,数据逐渐成为决定机器学习和人工智能效果的核心要素,在数据资源丰富的领域,语音识别和机器翻译已经逐渐实用.

进入 20 世纪,互联网快速发展,搜索引擎、社交网络等互联网应用快速推广,文本、图片、语音、用户行为等数据快速积累;与此同时,机器学习模型也进一步改进,两者相互促进,推动机器学习成为搜索、社交网络等重

要应用的核心技术.以文本理解为例,SVM[7],CRF[8],LDA[9]等模型成为互联网内容理解的主要工具,帮助用户快速找到需要的文档.这一时期,支撑机器的训练数据已经极大丰富,从数据中抽取和选择能够表达数据特性的特征,通常称为特征抽取或特征工程,成为决定机器学习效果的另一重要要素.以搜索引擎排序为例,通常需要从原数据抽取千级别以上的特征数量,并通过 Learning to rank[10]等方法决定不同特征对于排序任务的重要程度.对于文本数据,tfidf,ngram 等特征能够一定程度表达文本潜在语义,对各类下游任务,如分类、聚类等任务都能起到较好的数据区分作用;而对于语音、图片等类型的数据,当时特征抽取的方法还不足够表达数据的潜在语义,因而语音识别和图片理解任务效果尚未进一步取得显著突破.

2010 年以来,随着计算密集型硬件 GPU,FPGA 等的快速迭代,基于神经网络的深度学习技术在人机对弈、语音、图像处理等领域效果取得了显著提升.以图像识别为例,通过在海量图片数据 ImageNet[1]上进行学习,深度卷积网络在某些图片分类的效果已经超越人类水平.深度学习通过神经网络对数据进行端到端的特征抽取和建模,将两者统一到神经网络的计算中,通过神经网络逐层计算,形成对原数据不同粒度的特征抽象.实际表现证明:这种端到端的建模方式在许多任务上都能带来效果提升,尤其是对语音、图片等原始数据输入的任务,人工设计特征通常难以表达潜在的语义,深度学习的方法会带来显著提升.深度学习算法因为表达能力更强、自动进行特征抽取和选择,最近几年推动了人工智能技术广泛应用.通常来讲,深度学习算法依赖的训练数据规模更大,训练周期更长.比如近期快速发展的预训练技术[3,11],预训练阶段可能需要在 T 级别的数据上训练,即使在多台 GPU 机器上也需要数周的训练时间.

总之,机器学习训练是通过对已有数据进行抽象,构建具有预测能力模型的过程.数据、特征、算法是影响训练效果的重要因素.从发展趋势上看,机器学习算法复杂度不断增加,依赖的数据规模不断增大,提升效率、降低成本,将成为决定机器学习广泛应用的关键因素.

1.2 机器学习训练过程

通常来讲,可以将机器学习算法分为监督学习、非监督学习、强化学习等主要类别,机器学习训练过程可以表达为基于训练数据最小化损失函数的过程.对于监督学习,训练数据同时包含输入数据和数据标签,通过学习输入数据和数据标签之间的映射关系来优化目标;对于非监督学习,训练数据不包含数据标签,通常是通过学习数据潜在结构来优化训练目标;强化学习则通过数据构建代理与模拟环境,通过代理与环境的交互获得反馈来实现目标优化.

图 1 抽象了主要机器学习类别通用训练过程.对于特定任务和给定原始训练数据情况下,训练过程通常首先进行数据选择和特征抽取,选择出任务相关的数据、并抽取具备区分能力的特征.接下来,由于同一任务可以使用多种算法,需要根据任务、数据和特征选择适合的算法.之后开始模型训练,许多算法需要在数据上多次迭代逐渐优化目标函数.在得到训练模型后,需要做效果评测,监督学习通常在对训练不可见的测试数据上进行,计算出准确率、召回率等指标;非监督学习可以通过先验指标或人工的方式评测.上述整个过程可以称为一次训练尝试,之后分析评测发现的问题,通常会调整数据选择、特征抽取、算法选择等步骤,进行下一次尝试,如此循环,直到评测效果满足要求.

Fig.1 Training process of machine learning

图 1 机器学习训练过程

2 支撑机器学习的数据管理技术研究框架

首先,从数据的视角分析机器学习训练过程,讨论子过程的数据存取与计算,梳理数据管理与分析需求;其次,将机器学习任务看作训练数据上的一类查询请求,讨论在任务与数据之间构建机器学习系统的必要性,以及

系统需要具备的功能;最后,基于上述分析,从数据管理、自动优化和系统设计与实现这 3 个方面归纳支撑机器学习的数据管理与技术,提出本文研究框架.

2.1 从数据的视角对机器学习训练解构

基于图 1 所示的机器学习训练过程,图 2 给出了各个子过程的数据处理流程.从各个子过程看,数据选择对于同一份原始数据可以针对任务形成不同的选择结果,输出多份中间数据;数据选择之后,特征抽取需要抽取各种类型的候选特征供后续算法尝试;算法选择需要基于任务需求、数据和特征的统计信息选择算法并确定超参数,比如对输入文本抽样分析确定聚类算法和聚类个数等超参数;模型训练在迭代优化过程中会产生多个中间结果模型以及最终模型;效果评测则需要依赖最终模型和评测数据,产出准确率、召回率等测试结果;而问题发现与分析除了参考测试结果,也需要分析每个子过程输入输出数据之间的关联.可以看出,机器学习过程涉及多种类型、多个步骤的数据读写和分析,具有显著的数据管理与分析需求.

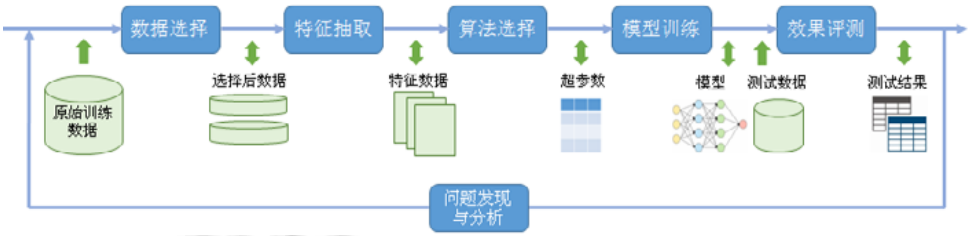

Fig.2 Data flow for training process of machine learning

图 2 机器学习训练数据处理

图 2 所示的训练过程需要多轮尝试,以获得优化的模型效果.通常来说,下轮训练尝试会基于当前模型效果对相关步骤进行一些调整,比如根据评测结果新选择一些相关的训练数据、增加或减少一些特征,之后进入新一轮训练.这个过程会进行多次,直到模型效果达到要求.同一任务的多轮训练,为数据复用提供了潜在机会.比如在当前特征集合,相对于上一轮如果只减少某个特征,那么可以较大比例地复用上一轮特征,避免重复特征抽取;比如当前轮只调整了超参数,则可以完全复用上一轮的数据和特征.

总结而言,从数据的视角看,机器学习可以看作是一个循环进行的多步数据处理与分析过程,具有显著的数据管理与分析需求和数据复用潜力.因此,数据管理与分析技术在机器学习中有重要应用.

2.2 从系统的视角对机器训练解构

从系统的视角看,机器学习训练可以看作是输入数据上的一类查询操作.如图 3 所示:训练请求的描述作为硬件环境和训练数据之上的查询条件,模型作为查询结果.与数据库统计查询相比,机器学习通过算法挖掘数据潜在模式并形成预测功能,因而查询的计算逻辑更加复杂.训练请求的描述通常是算法的逻辑表达,映射为训练数据和硬件环境上的物理执行需要较多的转化和优化工作;比如,需要考虑多种类型的数据存储和读取方式、以及考虑硬件的计算能力和内存限制合理调度计算.人工进行上述转化和优化需要较多的系统优化经验,工作量较大且优化的方法难以在不同任务间复用.因此,在训练任务和数据与硬件环境之间构建通用的机器学习系统,对各类机器学习算法效率进行整体优化,有利于提升模型构建和训练的效率.

基于机器学习训练过程的计算特点,机器学习系统需要包含 3 项重要功能.

- 语言处理功能.使用机器学习领域描述性语言进行建模,比如矩阵运算等;描述性语言建模过程较快、新算法支持灵活,构建过程只需要关注任务和算法本身而忽略执行细节;系统需要完成描述性语言向实际数据的读写和硬件上计算的转换;
- 查询优化功能.训练环境通常存在异构硬件,描述语言指定的训练过程通常有多种物理实现方式;系统通过自动优化技术,为任务自动生成优化的物理实现,减少人工参与优化过程,整体提升训练效率,也能够实现上层语言构建的模型在不同硬件环境执行;

- 数据存储与存取功能.基于数据自身结构特点,实现任务无关的存储和读取优化,整体优化存储成本的存取效率;同时,通过对多任务数据的系统管理,挖掘数据复用机会.

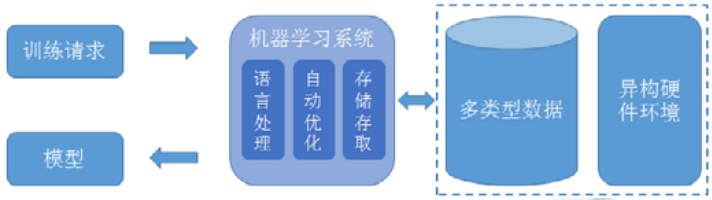

Fig.3 Machine learning as query request and system support

图 3 机器学习查询请求与系统支持

总结而言,从系统角度看,机器学习训练可以看作一类复杂分析的数据查询请求.通过系统支撑训练过程,能够提升建模速度、挖掘多任务之间的语义关联,通过任务间数据复用提升整体存储和计算效率.

2.3 研究框架

当前,机器学习训练过程以任务为中心的方式进行,为特定任务进行数据存储和计算优化,尚未形成整体的数据管理和系统性优化方案.基于前文对机器学习训练过程的数据管理和系统支撑需求分析,提出如图 4 所示的研究框架,对支持人工智能的数据管理与分析技术进行归纳与总结,梳理技术现状,提出优化方向.

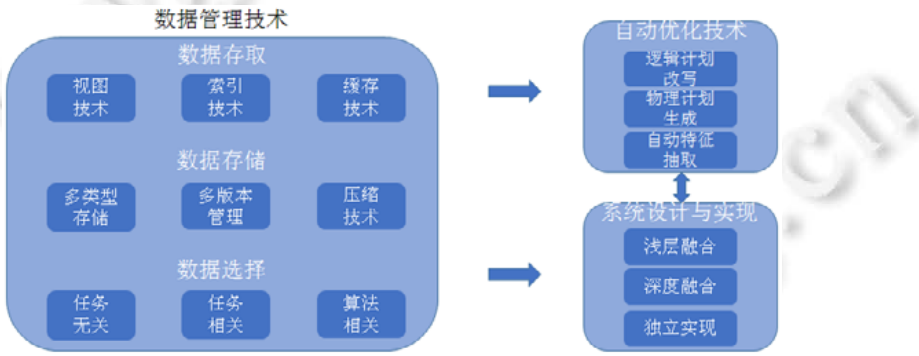

Fig.4 Research framework of data management technologies for maching learning

图 4 支撑机器学习的数据管理技术研究框架

数据管理技术以训练过程中各类数据作为对象,从数据选择、存储和存取方面总结和梳理现有技术在选择任务相关的训练数据、存储和读写效率优化方面的工作,并提出机器学习数据管理新研究方向.自动优化技术将借鉴数据库查询优化经验,梳理对描述性语言进行逻辑计划改写、物理计划生成和自动模型选择方面的工作;机器学习计算通常更为复杂,自动优化技术还有较大发展空间,提出后续优化方向.系统设计与实现将讨论实现机器学习系统的不同方法,包括基于数据库系统并进行浅层或深度的融合方法以及基于数据库设计思想在数据库之外独立实现,以最大限度复用系统已有的功能和实现经验.未来,随着机器学习技术广泛应用,在线机器学习需求显著,提出数据管理与分析技术的挑战和研究新方向.

3 个维度的关系是:数据管理技术和自动优化技术为在训练过程可以独立运用的局部优化,也是机器学习系统的核心功能;机器学习系统的实现则是从全局的角度实现各类技术的合理集成,一方面为机器学习训练提供完整的系统支持;另一方面,也为各项优化技术持续改进提供平台支撑.

3 数据选择技术

数据选择是从原始输入数据中选择与机器学习任务相关训练数据的过程.原始输入数据通常规模较大,包

含重复或者与训练任务不相关的数据,直接在原始输入数据上训练可能面临训练周期长、任务效果不够优化的问题.因此,自动选择与任务相关的语料将有助于提升训练效果与效率.数据选择的主要挑战是建立训练数据之间、训练数据与训练任务和算法之间的关联性,进而选择出数据规模适中对训练任务和算法效果帮助最大的训练数据子集.本节将首先介绍任务无关的方法,通过少量数据能够覆盖全量数据的语义空间;其次介绍任务相关的方法,从通用的训练数据中选择与任务领域相关的数据;最后介绍算法相关的方法,自动选择对算法效果影响较大的训练数据.

3.1 任务无关的数据选择

数据通常可以表达为高维空间中的向量,训练数据集可以看成是高维空间中的区域.任务无关的方法通常基于区域覆盖,假设原始数据包含重复、相似或相关的数据,可以选择一组规模较小且相互独立的数据来覆盖原数据在高维空间中的区域.因此,在选择后的数据上的训练效果不会明显降低,但效率可能显著提升.常用的任务无关的数据选择方法包括采样算法和基于相似度计算.

随机采样是常用的采样算法,随机抽取一定数量的数据作为采样集用于后续训练.当数据在高维空间中分布较均匀时,采样集通常能够较好地覆盖原始数据区域.但对于分布不均匀的情况,比如分类问题中不同类别训练数据量差异较大,随机采样可能会漏掉部分类别的训练数据,或者需要较多次采样才能覆盖.Coresets[12]是一种更复杂的采样技术,首先通过数据向量表达计算之间的相关性,得到每条数据被选择的概率以反映所在高维空间的分布,之后按照选择概率抽样.Coresets 所需的采样次数通常与原始数据向量表达的维度有关,与原始数据量无关;对于数据量大但维度较小的情况,能以很少的采样数量覆盖原始数据区域.

基于相似度计算的方法通过定义和数据间相似度计算,选择互相之间相似度较低的子集来表达完整的原始数据集.通常来讲,数据可以表达为向量,相似度可以通过余弦计算.比如对于文本数据,可以通过文档向量模型表达数据.近年来,深度神经网络被广泛用于抽取数据的潜在语义的向量表达,比如:可以使用词向量[13]、句向量[14]来表达文本,可以通过卷积神经网络的隐层向量来表达图片,通过递归神经网络的隐层向量来表达语音.获得相似度后,可以增量添加的进行数据选择.比如:先随机选择一条数据,之后优先选择并添加与所有已选择数据均不相似的数据;也可以通过聚类的方法,比如采用 *k*-means 等算法将数据进行分类,之后从个聚类中选择一定量的数据形成对原始数据的覆盖.

3.2 任务相关的数据选择

对于监督学习任务,通常会面临缺乏任务领域相关的训练数据的挑战:一方面,某些领域的训练数据本身比较稀缺,如低资源语言的机器翻译任务;另一方面,在实际产品中,机器学习模型的领域数据通常依赖于用户对产品的请求,在产品被广泛使用前,通常缺乏领域数据.任务相关的数据选择技术方法基于少量的领域相关的训练数据,从海量通用数据中选择与任务领域相关的数据扩充训练数据,通常能够显著提升数据稀缺的训练任务.本节将介绍两类基于领域的数据选择方法——基于生成的方法和基于判别的方法.

- 基于生成的方法首先基于少量领域数据学习领域相关生成式模型,如语言模型、AutoEncoder[15]等,学习领域相关数据的潜在结构;之后,利用生成式模型计算通用数据的生成概率,选择生成概率较大的数据用于后续训练[16,17].也可以在领域数据和通用数据上各训练一个生成式模型,如果数据在两个生成式模型的生成概率类似,则将通用数据用于后续训练.以自然语言处理任务为例,可以使用领域相关的数据训练神经语言模型,之后对通用领域的数据进行解码,并基于解码得到的困惑度(perplexity)[18]来表示生成概率,进而选择困惑度较低的数据加入训练;
- 基于判别的方法将领域相关的数据作为正样本,从通用数据中随机采样数据作为负样本,基于正负样本构建领域数据分类器,学习领域相关和领域无关数据之间分布上的差异,通过分类选择正向分类概率较高的数据输入后续训练使用[19].由于领域相关的数据规模较小,分类器的效果可能较差,可以通过融入半监督特征增强分类器的效果.比如对于文本数据,可以先在海量非监督语料上训练语言模型,获得任务无关的通用语言序列和结构表达能力,之后将通用语言模型在正负样本的隐藏层输出作为特

征,与正负样本一起训练领域数据分类器,通常能够获得更好的分类效果.

3.3 算法相关的数据选择

算法相关的方法选择对具体算法效果提升最大的数据用于训练.与任务相关的数据选择方法相比,算法相关的方法更能体现特定算法对训练数据的区分度差异.比如:对于同一分类任务,通常有逻辑回归、支持向量积等不同算法,可能会将原始数据映射到不同的高维空间,因此,优化不同算法依赖的数据分布可能不同,需要针对性的选择.算法相关的方法总体基于主动学习[20]的思路,基于模型当前预测能力计算数据对当前模型的效果提升,选择提升最大的数据用于后续模型训练.本节将分别从基于增量选择的方法和基于数据加权的方法两个方面介绍.

- 增量选择的方法首先基于少量的领域相关数据训练初始模型,之后使用初始模型预测通用数据,选择最容易预测错误的数据加入训练.如此循环,直到模型效果符合预期[21].判断预测是否容易发生错误,可以基于模型对预测结果的置信度.对于分类任务,置信度可以通过概率最高的类与次高的类之间的差异来评估,差异越小,则置信度越低;也可以根据多个类别上概率的交叉熵来评估,交叉熵越高,置信度越低.对于生成类任务,置信度可以基于模型生成的结果与真实结果的差异来评估.增量选择方法只在选择后的数据上训练,通常效率更高,但可能遗漏部分对模型效果有提升的数据;
- 基于加权的方法首先在全量通用数据上进行一轮训练得到初始模型,后续迭代过程中,逐步增加当前轮模型容易预测错误的数据权重,以快速提升模型在整体数据上的预测效果[22].目标函数的优化通常会在数据上多次迭代进行,比如采用随机梯度下降等方法,模型每次迭代都学习到训练数据的部分知识.可以使用当前迭代模型对数据进行预测,如果置信度较高,则说明已经学习到该条数据的知识,后续训练可以降低该条数据的权重;反之则可以增加该条数据的权重.置信度的计算与任务相关的方法中的介绍类似.基于加权的方法由于首先在全量数据上进行训练,通过逐渐增加重要训练数据的权重而避免直接丢弃部分训练数据,通常可以保留训练数据的整体知识,模型鲁棒性更强;但相对于基于增量的方法,训练周期可能更长.

3.4 小 结

数据是影响训练效果的源头性因素,数据选择是模型训练效果和效率重要决定因素.上述 3 类数据选择方法总体是基于单任务的一轮训练过程,将来优化方向包括:1) 在单任务多轮训练训练场景下,如何复用前轮的数据选择结果优化当前轮数据选择的效率;2) 在多任务训练的场景下,如何复用相关任务的数据选择结果优化当前任务数据选择的效率.

4 数据存储技术

机器学习训练过程会读取或产生多种类型的数据,对各类数据进行合理存储,是后续数据存取与分析的基础.面向机器学习的数据存储技术面临许多挑战:首先,训练过程中需要输入和产生的多种类型的数据,包括结构化数据、半结构和非结构化数据,需要依赖多类型存储系统形成统一的存储方案;其次,建模和训练过程通常需要多次迭代优化模型效果,同一数据会生成多个版本,需要兼顾多版本存储成本和读取效率;最后,随着数据规模增大,压缩技术能够提升数据读写效率,需要基于数据之间的关联,探索压缩比更高且计算开销更友好的压缩算法.本节将针对上述挑战讨论和总结相关工作.

4.1 多类型数据存储

机器学习训练过程主要涉及的数据类型为训练数据、特征和模型,同时也涉及训练日志、评测结果等相关数据.首先介绍主要数据类型的常用存储方法,其次介绍相关数据存储,体现数据间关联、帮助问题分析.

对于结构化的训练数据,可以采用关系数据库表存储;半结构化训练数据可以存储在 KV 系统中,或者采用 JSON,XML 等可以灵活扩展属性的格式存储在文件中.TFRecord[23]是一种灵活和紧凑的文件格式,采用 Protobuf[24]可以自解析地表达任意类型数据,内部采用二进制存储,方便分块压缩以及机器间网络传输;对于大

规模训练数据,采用 TFRecord 格式存储将有利于进行分布式处理.特征数据通常表达为矩阵:行表示数据,列表示不同维度的特征.列较少的特征矩阵可以存储在关系表中,以表属性存储列;较大规模的特征矩阵可以采用 TFRecord 等文件格式分块或者整体存储.模型一般表达为参数矩阵之间的有向无环图,可以用通用的格式 ONNX[25]来存储,不同类型机器学习框架的模型存储格式可能不同.

除了数据、特征和模型,训练过程还包括任务描述、超参数、损失函数变化等多类信息,需要形成统一存储方案来体现数据之间的关联.ModelDB[26]以模型为中心,关联存储依赖的训练数据、特征、超参数、评测结等数据,支持查看和追踪训练过程数据流.同时,为方便训练过程问题发现与分析,可以基于数据沿袭[27]技术存储数据选择、特征抽取等过程细粒度数据变换的映射关系,方便正向和反向的溯源查找.

4.2 多版本数据存储

机器学习训练过程迭代进行,同一份数据可能产生多个版本.首先,同一任务的一轮训练过程通常需要在输入数据上迭代优化目标函数,每次迭代都会产生一个模型版本;其次,机器学习模型调优的过程通常需要多轮训练尝试,每轮训练可能会对数据、特征和模型进行调整,形成多个版本.本节介绍首先介绍多版本数据通用存储方法,接下来分别介绍针对训练数据和模型的优化存储方案.

对于多版本存储需求,数据库和文件系统已经有成熟的解决方案.HBase[28]等数据库系统以及 AWS S3[29]等文件系统均原生支持多版本存储,可以分别存储训练过程中的结构化、半结构化和非结构化数据.可以配置同一数据允许存储的最大版本数,避免存储过多版本;也可以通过设置最长过期时间(time-to-service)来清理长期未适用的版本.

对于训练数据,同一任务多次迭代训练形成的多个版本之间通常有较大比例的数据重复,可以采用全量存储和增量存储结合的方式优化存储成本.DataHub[30,31]借用 Github 的思想对连续改变的多版本数据进行管理,支持修改后添加新版本,从多个已有版本数据合并形成新版本等操作.对于某个版本的数据,可以全量存储,也可以增量存储相对某个已有版本的增量变化.通过构建数据版本图,以数据版本为节点,边表达版本之间增量存储的存储成本和重构成本,成本最优的存储方案可以通过基于数据版本图上边的存储成本来搜索最小生成树.实际应用中,在最小化存储成本的同时,也需要限制重构成本的上限,实现两者之间的平衡.

ModelHub[32]提出针对模型的多版本存储方案,由于模型主要由浮点类型参数矩阵构成,尚难有清晰的语义解释;即使是同一轮训练得到的不同版本模型,在整体参数值上也可能表现出较大的差异,不利于增量存储.因此,ModelHub 对模型进行更细粒度拆分,构建类似 DataHub 的数据版本图,将参数矩阵作为数据版本图的节点.对于同一轮训练产生的多个中间结果模型,其对应的参数矩阵数值通常变化不大,通过增量存储可以带来存储空间收益;而同一训练任务多轮训练产生多个模型,其浮点参数的尾数高位部分可能变化较小,可以截取高位尾数进行连续存储,降低该部分的存储空间.与 DataHub 不同的是,读取模型时需要同时读取其所包含的所有矩阵,因此在数据版本图搜索最小生成树时,需要考虑多节点共同读取的要求.

4.3 数据压缩

压缩技术通过避免多次存储数据中的重复模式,能够减小数据存储和读取成本.随着机器学习训练所需要的数据规模不断增加,压缩技术可以有效降低训练中数据读取规模,也能帮助更多常用数据常驻内存,提升计算效率.输入数据和特征通常都可以用矩阵表达,本节首先介绍通用数据压缩技术在矩阵存储上的应用,接下来介绍基于矩阵列簇的压缩技术,以及支持在非解压数据上直接进行训练的压缩技术.

稠密矩阵可以连续存储,如果矩阵规模较大,可以进行分块划分存储,进而应用块压缩技术进行压缩[33].需要考虑一些压缩选项:一是压缩算法的选择,训练过程本身计算开销较大,需要考虑解压算法的计算开销,可以根据需要选择如 Snappy[34]等轻量级压缩算法,可以根据每个分块的数据分布特点选择不同的压缩算法;二是如果需要对块内数据进行多次更新,比如多次更新训练数据或特征,可以将数据块在逻辑上划分成更小的分片,对分片数据进行分别压缩,避免更新时对整个块进行解压.

特征之间通常并会完全独立,因而特征矩阵的的列之间可能存在相关性.将相关性较强的行或者列进行组

成列簇,对列簇进行整体压缩可能带来更高的压缩比[35].具体而言,在矩阵写入时,可以基于数据抽样快速计算列之间相关性搜索出列簇划分.不同列簇可以选择不同的压缩算法,比如列簇内重复数字的位置如果较连续,可以采用行程长度压缩算法[36];而如果列簇内的数据重复度较低,则可以选择不进行压缩.通过对列簇的选择不同压缩算法,实现更优的局部压缩,从而在整体矩阵上实现更高压缩比.

特征矩阵通常每行表示一条训练数据,列表示不同维度的特征值.块压缩技术忽略矩阵行之间的边界,训练读入时必须先对整块解压,分离出行数据后才能进行计算.可以用字典记录每行数据的边界位置,压缩算法避免跨行压缩,进而支持在压缩数据上直接进行训练[37].比如:如果使用前缀树压缩算法,读取到行边界时可以停止继续查找更长前缀,多行共同具有的重复模式只存储一次,实现压缩效果;同时,行内的数据压缩可以保留训练算法所需要的列索引信息,这样对压缩数据进行顺序读取时,可以获得数据的行索引和列索引,算法训练可以直接在压缩后的数据上计算[38].

4.4 小 结

目前,机器学习数据存储仍主要以任务为中心为主,尚未形成统一的存储模式.将来,从不同训练任务中抽象出统一数据存储模式、挖掘任务间的数据语义关联,将成为从数据的视角管理机器学习训练过程的基础.以此为基础实现任务间数据共享与复用,能够显著降低工业级机器学习应用的资源成本.

5 数据存取技术

机器学习训练过程需要对训练数据、特征、模型等数据进行变换和多次读写,数据存取技术,如视图、索引和缓存等技术,对于提升训练效率至关重要.面向机器学习的数据存取技术将面临许多挑战:首先,数据和特征的变换可能使用较复杂的线性变换,对变换得到的中间结果进行高效增量维护将面临挑战;其次,机器学习通常需要处理非结构化数据以及以通常将各类数据表达为矩阵,需要探索针对非结构化数据和矩阵的索引方法以提升数据读取效率;最后,可以缓存高频读取数据以优化效率.由于不同数据的重构计算成本不同,缓存方案需要同时考虑缓存开销和重构计算成本.本节将介绍针对机器学习训练过程的视图、索引和缓存技术.

5.1 视图技术

在数据库领域,对于同一关系表,不同应用可能具有相同的数据选择或转化需求.数据库通过视图技术来管理通用的数据选择或转化的逻辑与结果,实现数据与上层应用的逻辑独立,并通过复用中间结果提升整体读取效率.视图通常分为 Layzer 和 Eager 两种模式:Layzer 并不实际存储中间结果,在数据读取发生时,实时从原数据表读取数据并计算,实现了数据选择和转化逻辑的复用;Eager 模式则实际存储和维护中间结果,直接从视图中读取数据,实现了数据选择和转化结果的复用.本节首先介绍基于视图技术管理数据转换和特征抽取中间结果的管理,其次介绍针对线性代数变换的增量视图维护方法.

机器学习训练通常需要多种类型、多个步骤的数据转化处理.对于文本为输入,通常需要进行大小写归一化、词根提取等操作;对于图片输入,通常需要对图片进行旋转、分辨率变化等操作.特征抽取和选择可以认为是机器学习特有的数据转换,同一任务多轮训练尝试使用特征集合通常存在重复,同一原数据上不同任务的特征集合通常也不是完全独立.因此,通过视图存储公共的数据处理和特征抽取中间数据,可以在多训练任务之间实现数据复用,提升读取效率.具体来讲,可以统计每个任务特征集合的访问频率,对不同特征集合采取不同的视图存储模式(Layzer 或 Eager),整体平衡多任务的特征存储成本和存取效率.COLUMBUS[39]针对相关任务特征集合之间存在重复,提出对特征集合的并集矩阵进行 QR 分解,Q 是正交矩阵,R 是上三角矩阵,可以看作多任务特征的公共中间表达,后续数据变换基于 Q 和 R 进行计算会更高效,因而可以采用 Eager 模式视图对 Q 和 R 进行存储.

机器学习训练可能会对数据进行一些复杂的变换,比如插值和线性变换等.原始输入数据在迭代过程中通常会增量更新,增量更新在这类复杂变换中的传播模式相对复杂,使用视图维护变换后的结果并增量维护将面临挑战.对于插值计算变换,增量数据只影响局部数据的插值结果,MauveDB[40]提出通过线段树来索引原始数

据,进而快速查找增量数据影响的局部原始数据,进而实现增量更新;对于回归计算变换,线性变换基函数计算结果通常可以增量更新,MauveDB 提出对基函数变换结果进行物化存储而实现增量更新.线性代数计算通常可以表达为矩阵乘法,局部增量会通过乘法传播至全矩阵,使得增量计算的时间复杂度较高;LINVIEW[41]提出:增量矩阵的行和列之间通常不是线性独立的,可以将增量矩阵分解为两个低维度的矩阵,降低矩阵乘法增量计算的时间复杂.

5.2 索引技术

在数据系统中,索引是提升数据读取效率的关键技术.常用的索引技术包括 Hash 索引、B 树索引等:Hash 索引用于快速执行索引项上的随机查找,B 树索引同时支持索引项上的随机查找和区间查找.对于训练过程中的结构化数据,可以直接使用数据库索引提升读取速度.本节将首先介绍非结构化数据的索引方法;由于训练过程主要数据类型可以抽象为矩阵,接下来介绍矩阵索引技术.

ZOMBIE[42]提出对数据按照潜在结构划分成任务无关的不同分组并建立索引,对于特定机器学习任务,可以评估每个分组的数据对模型效果的影响,选择影响较大的分组数据进行训练.具体而言:训练数据可以表达为向量,之后使用聚类算法划分成 k 个独立的分组;训练过程中,每次从当前对任务效果影响最大的索引组读取数据加入训练,索引组对效果的影响可以通过组中已加入训练的数据来评估.数据聚类通常能够反映数据潜在特性,有利于模型训练更快提升效果.比如:如果文档长度是某个分类任务的重要区分特征,而长文档可能包含一些共同的词语,因此,聚类算法可能将长文本聚为一类,训练过程从长文本所在类中读取数据可能收敛更快.由于非结构化数据缺乏语义解释,这种基于数据潜在结构和特征的聚类的分组索引是快速读取任务相关数据的一种可行方式.

矩阵是机器学习重要数据类型,可能需要同时支持连续读写和随机访问.如果是稠密矩阵,通常连续存储的读取效率较高;对于稀疏矩阵,通常可以将矩阵线性变换为一维数组,根据非零数据的下标建立B树索引,叶节点存储对应数据值,以支持高效的随机读取和范围读取.与通用的 B 树索引不同的是,矩阵的不同区域可能具有不同的稠密程度.LABTree[43]设计了稀疏类型和稠密类型叶节点,稀疏叶节点存储每个数据的矩阵索引和数据值.稠密节点对一段连续的值只保留一个索引项,包含连续值的起始矩阵坐标以及值本身,存储格式更紧凑.LABTree 支持根据矩阵的局部稠密程度自动选择稀疏类型或者稠密类型的叶节点,比如:当一个稀疏类型的叶节点发生插入操作并触发叶节点分裂,如果检测到叶节点数据的连续程度较高,会自动转换了稠密类型叶节点.

5.3 缓存技术

机器学习训练通常需要在同一份数据上多次迭代,通过将需要多次读取的数据放入缓存,可以提升迭代过程中的训练效率.本节首先介绍基于代价估算的缓存策略,对单次训练生成访问效率提升最大的缓存策略;接下来介绍在多任务场景下模型缓存方案.

将单次训练所有依赖的所有数据作为节点,有计算依赖关系的节点之间添加有向边,可以构建训练数据流的有向无环图(direct acyclic graph,简称 DAG)[44].为 DAG 中每个节点定义执行耗时代价估算:如果节点放入缓存,计算耗时为 0,但消耗等数据大小的内存资源;否则,内存消耗为 0,节点执行耗时则需要综合考虑从原始输入数据节点到当前节点的计算时长,以及该节点被后续节点依赖的次数.基于执行耗时的代价估算,在训练环境限定内存资源条件下,可以搜索最优缓存策略,以最大程度缩短训练耗时,在内存资源与计算资源之间取得平衡.

在多任务训练环境中,对于新训练任务,可以搜索相关历史任务的训练结果,使用其进行初始化将有助于新模型快速训练收敛.因此,可以通过缓存被新任务依赖较多的模型来提升多任务训练整体效率.这里,计算新任务与历史任务相关性是关键.COLUMBUS[39]提出使用缓存中的模型对新任务的测试数据进行评测,取效果最好的模型作为新任务的初始化模型;之后,基于 LRU 算法,可以将不常访问的模型交换出内存.实际上,这是一种基于迁移学习[45]的训练方法,最新预训练技术[3]的进展验证了其有效性.结合模型缓存优化等技术,迁移学习是实现基于资源复用的实时机器学习的重要途径.

5.4 小 结

当前,机器学习训练的数据读取对象粒度较大,比如对非结构化训练数据、特征矩阵和模型进行整体读取.上述缓存技术主要针对整体数据对象的粒度优化存取效率,并考虑到多轮训练和多任务间数据复用的需求.未来需要进一步探索训练中数据对象的内部语义,提炼出适合复用的子结构,通过探索细粒度的存取技术来进一步优化数据读写整体效率.

6 自动优化技术

机器学习训练过程通常采用领域描述语言进行灵活而高效的建模,如 DSL(domain specific lanaguage)[46]或者 DML(declaritive machine learning language)[47])等.DSL 向上提供矩阵运算、统计函数等机器学习领域的逻辑计算操作,由于需要考虑内部数据表达、机器硬件环境等执行细节,无法直接映射为前述数据存取、存储技术提供的物理操作,需要使用自动优化技术将 DSL 描述的逻辑计划转化为可执行的物理计划,在多种可能的物理计划中自动决策并选择最优计划.本节首先介绍逻辑计划改写,由于建模过程通常表达为向量和矩阵之间的连续运算,需要考虑维度等信息,将运算序列改写为代价最优的等价序列;其次将讨论物理计划生成,需要考虑计算在异构硬件上(CPU、GPU 等)的调度方案;最后讨论自动模型选择,也即为训练任务自动生成效果较优的训练计划,需要在限定的训练时间内综合考虑特征选择、算法选择和超参数调优等多种因素的合理组合.

6.1 逻辑计划改写

将 DSL 的运算操作和输入数据作为节点,操作与数据之间的依赖作为边,DSL 代码可以转换为 DAG,称为逻辑计划.原始的 DAG 中可能包含重复计算子图、操作执行顺序可能不是最优,可以通过逻辑改写技术将原始 DAG 改写逻辑等价、执行效率更优的新计划.实际上,关系数据对用户 SQL 进行查询优化时,会广泛应用改写技术,比如选择下推、投影下推等,其思路是:让缩减数据规模的操作尽早的发生,这对机器学习逻辑计划改写也有很大的参考价值.本节将分别介绍静态改写和动态改写技术.

静态改写[46]针对编译时已确定的参数信息,如矩阵维度等,在 DAG 运行前进行改写.常用的方法包括公共子表达式消除、常量折叠以及分支移除等.DAG 中的公共子表达移除可以递归进行:首先,从叶子节点开始标识相同子表达式;之后从,父亲节点到根节点逐级递归标识和消除.分支移除可以帮助提前确定一些维度信息,使得运行前能更多进行静态改写.对于机器学习,DAG 中的主要操作是矩阵运算,矩阵运算操作和顺序可以基于启发式规则优化,比如两次矩阵转秩不需要计算($X^{TT}=X$)、二元操作 $X+X$ 可以改写为一元操作 $2\cdot X$ 而避免两次读取 X.矩阵乘法顺序对计算量有较大影响,比如计算 $X^T\cdot Y\cdot d$,d 为向量,如果按照$(X^T\cdot Y)\cdot d$ 的顺序,则会产生较大中间矩阵,时间复杂度是 $O(n^3)$;如果按照 $X^T\cdot(Y\cdot d)$进行,则中间结果为向量,时间复杂度为 $O(n^2)$.实际上,矩阵乘法顺序优化与数据库多表连接顺序优化类似,都具有最优子结构和无后效性的特征,可以采用动态规划方法求解最优乘法顺序;与多表连接不同的是,矩阵乘法计算代价需要额外矩阵稀疏性等信息.

动态改写[46]是在运行过程中进行,其原因是改写依赖的维度等信息在运行时才能完全确定.一个典型的例子是 DAG 中包含 if-else 分支,if 分支和 else 分支的输出数据的维度可能不一样,导致后续计算操作输入数据的维度不能在运算前确定.在确定完整的维度后,可以继续对一些运算顺序进行改写.比如对于$(-X^T)\cdot y$,如果向量 y 的维度小于 X 的行数,可以改写为$-(X^T)\cdot y$ 以较小负号运算计算量.

6.2 物理计划生成

机器学习的训练环境通常包括异构硬件,如 CPU、GPU、FPGA 以及计算集群等;逻辑改写后的 DAG 可以有多种物理计划,需要结合计算的资源开销和硬件环境生成最优物理计划.数据库领域通常采用基于代价估算的物理计划生成,比如根据读取数据规模的预估确定直接读取原数据表或者读取索引.机器学习 DAG 的操作通常是密集型计算,运行代价估算需要重点考虑计算复杂度和内存要求两方面的因素.

计算复杂度通常可以通过矩阵维度来预估,维度较小或者数据稀疏的矩阵运算,可以调度到 CPU 中运算;维度较大的稠密矩阵运算,可以调度到 GPU 并行计算;操作输出矩阵的维度信息可以直接由输入矩阵维度确

定.基于计算复杂度预估调度也需要考虑操作融合的影响,比如可以将连续的运算融合成一个算子,调度到同一硬件上执行,避免数据在内存间频繁拷贝,进而提升计算效率.

操作的内存需求也可以通过矩阵维度来估算[46].如果单机可以满足操作内存需求,操作计算可以调度到单机 CPU,GPU 上进行.对于内存超出单机限制的矩阵,可以使用多机分布式计算,比如采用分布式分块乘法.矩阵运算的物理执行可以基于矩阵内存估算进一步细化,比如对于矩阵乘法 $X \cdot Y$,如果 Y 较大无法放入内存,但 X 较小可以放入内存,则计算过程可以让 X 常驻内存,对 Y 进行一次外存扫描可以得出计算结果,这与数据库两表连接算法选择有相似的地方.同时,确定操作内存需求和机器内存限制后,可以对运算顺序进一步改写.比如计算 $X^T \cdot y$,如果向量 y 的维度小于 X 的行数,且 y 可以放入机器内存,计算顺序可以改写为 $(y^T \cdot X)^T$ 而避免对数据规模较大的 X 进行转秩操作.

6.3 自动模型选择

上述逻辑计划改写和物理计划生成优化技术主要对单次训练进行自动优化,避免人工参与.如前文所述,同一任务通常需要多轮训练尝试,过程中需要分析评测效果对特征、算法、超参数等进行调整优化训练效果,迭代尝试需要大量的人工分析工作.自动模型选择在用户不完整指定训练依赖的情况下,自动搜索出特征、算法等之间的合理组合,训练得到效果优化的模型.一方面可以减少多次尝试的人工参与工作量,另一方面也给系统更大的优化空间.

模型选择可以看作是特征选择、算法选择和超参数选择三者之间的组合,称为 MST(model selection triple)[48].对于设定的一组候选 MST,系统可以自动选择和训练出任务效果最优的模型.与上文单次训练优化技术不同的是,从候选 MST 自动选择最优模型可以基于面向候选集合的优化思路:一方面可以充分利用并行计算,将候选 MST 调度到多卡或者多机上并行训练;另一方面,候选模型之间通常具有较高的数据共享,可以通过复用来提升模型候选的整体执行效率.比如固定特征和算法,指定超参数的搜索条件,候选 MST 之间的特征可以完全复用;如果对特征设置了枚举条件,枚举得到的特征集合之间并不完全独立,采用前文所述的视图技术存储特征集合的公共表达,可以避免多任务重复计算特征.

对于深度学习,神经网络模型进行端到端的特征抽取和建模,神经网络架构搜索技术(NAS)[49−51]是自动模型选择的重要方法.NAS 通常可以以人工设计的网络作为搜索起点,将超参数、层数、连接方式等不同维度作为搜索空间,应用强化学习、进化算法、贝叶斯优化等搜索策略,根据效果评测的反馈来搜索新模型.深度神经网络通常具有可复用子结构[52],可以通过将已训练好的模型参数迁移到新模型来提升搜索效率.

6.4 小 结

自动优化是系统提升机器学习训练效率的关键技术.当前,自动优化的研究主要针对单任务训练,针对多任务整体训练效率的自动优化技术有很大的探索空间.其中,探索模型中可复用子结构、建立任务间的语义关联、自动识别相关任务模型,将是自动优化技术重要的研究方向.

7 系统实现

数据库是多种数据管理技术的系统集成,本节从数据库系统的视角讨论对机器学习训练任务的支持,探讨数据管理技术对提升机器学习训练效率的系统支持.由于数据库和机器学习系统在数据模型、管理对象与优化目标上存在差异,基于数据库已有的数据管理技术支撑机器学习训练需要在数据库功能复用和建模与训练灵活性方面进行合理适配,比如对数据模型、存储与存取功能、查询优化技术采取不同的复用和适配策略.本节将讨论 3 种实现方式:与数据库系统浅层融合、深层融合以及在数据库系统之外独立实现[53].

7.1 浅层融合

浅层融合背后的假设是机器学习算法和计算可以使用 SQL 语言来表达,实际上,大部分机器学习算法都属于线性代数运算,可以用矩阵间的计算来表达;如果将矩阵存储为关系表,矩阵间运算就可以通过表连接操作来计算.如果训练数据存储在数据库中,浅层融合方法可以在数据库内部完成训练而避免额外的数据移动,也可以

复用数据库的存储和读写功能以及优化技术[54−58].本节将分两方面介绍浅层融合的关键技术——各类矩阵运算的实现和主要机器学习算法的实现.

矩阵按行存储为关系表,行下标与表的行号对应,每行的向量可以用数据库支持的数组类型来存储.为方便后续计算,同一矩阵可以按行或者列为顺序分别存储为一个表[54,55].基于这个格式,矩阵之间的加减法可以通过表之间的等值连接来计算;矩阵乘法 $A\cdot B$ 可以表达为以按行存储的 A 矩阵表与按列存储的 B 矩阵表之间的全连接,并计算连接向量的点积.对于稀疏矩阵,可以按照行下标、列下标、值的三元组形式存储为关系表,乘法结果可以对相乘两个表的行下标和列下标进行 group by,再对列下标和行下标进行等值连接得到.对于更复杂的矩阵运算,比求矩阵求逆操作,如果数据可以放入内存,可以通过用户自定义函数(user define function,简称 UDF)来计算并扩充到 SQL 中.如果矩阵的规模较大,可以将矩阵划分为子矩阵存储到不同关系表中,乘法先在分块矩阵表中进行,之后进行聚合.

基于上述矩阵运算,机器学习算法可以分为一趟计算和多趟计算:一趟计算,如最小二乘法算法等,可以直接表达为矩阵运算的序列,因而使用多条 SQL 语句就可以实现;多趟计算则针对需要迭代优化的机器学习算法,比如逻辑回归、梯度下降、k-means 聚类算法等.多趟计算的主要挑战是,如何使用 SQL 表达迭代逻辑、存储中间结果和判断迭代终止.可以将迭代内的计算逻辑定义为一个视图,将视图与存储递增序列的虚表进行链接来表达迭代逻辑.更灵活的方式是将算法定义为一个 UDF,迭代内的逻辑通过 SQL 来实现,迭代生成的中间结果使用一个中间表进行存储,外层通过脚本语言来控制结束条件.这种方式将中间数据存储为关系表,其读写在数据库引擎内进行,访问效率较高;同时,通过将 UDF 集成到 SQL,不同的用户也可以复用算法的实现.

7.2 深层融合

机器学习训练和数据库的数据模型分别是矩阵和关系表:矩阵适合表达多行多列的并行计算,而关系表更适合表达行和列级别的数据读写和计算.使用关系表存储矩阵,会对后续计算带来限制.深度融合的方式通过对数据库内部数据存取和计算流程进行优化和适配,使得数据库能够对机器学习计算进行更原生的支持,进而提升模型构建和训练效率[59−63].本节将从两个方面介绍深度融合方法:一是对数据库内部功能进行扩充和适配,二是基于数据库底层功能构建机器学习系统.

SimSQL[59]提出将矩阵、向量作为数据库的内置数据类型,同时内置矩阵乘法等计算的实现.这样,矩阵可以作为元组的属性,矩阵间的运算表达为元组属性间的运算,使用 SQL 编写训练代码会大大简化.内置数据类型还有两点优势:一是存储的额外开销较小,数据页之内不再需要存储行的位移、长度等信息;二是方便实现并行计算优化,比如矩阵作为属性整体读取到内存后,矩阵乘法可以整体通过 Eigen[60]等 CPU 并行计算库实现.另一方面,内置数据类型可以将矩阵的统计信息直接暴露给数据库查询优化器,提升代价估算的准确度.比如:如果矩阵 A 的维度是[100,10000],矩阵 B 的维度是[10000,100],则 $A\cdot B$ 的维度是[100,100],数据规模较小;查询优化器可以基于维度信息让 A 与 B 的乘法尽早发生,使得中间数据的规模更小.

数据库系统的设计实现了数据管理与应用独立,具有逻辑独立的数据存取层.通常来讲,数据库数据存储层提供面向块的数据读写操作,可以直接在数据存取层之上构建机器学习系统.这样做有两方面优势:一是可以完全复用数据库成熟的数据存储和存取技术,比如数据压缩、缓存管理等;二是直接进行块读写的数据吞吐更大,数据访问效率更高,与后续并行计算更好地匹配.DAnA[61]提供机器学习领域的描述性语言(DSL),直接使用数据库块对训练数据进行存储;将 DSL 转换为 DAG 后,数据读取会直接转换为块的读取操作.DAnA 直接从数据库的缓存管理器中复制整块数据到并行计算硬件 FPGA 的缓存中,之后利用 FPGA 的多核能力对块进行并行解析提取数据字段,避免了在数据库内部将块解析为元组时,额外的计算和数据拷贝过程.这种方式实现了存储和计算在数据模型与硬件资源方面的解耦合,系统灵活性更高.

7.3 独立实现

数据库和机器学习除了数据模型不同,管理与优化的目标也有所不同:数据库主要管理结构化数据,而机器学习是非结构化数据分析的重要手段.使用数据库管理非结构化数据不够灵活,且较难无缝集成到非结构化数

据处理流程中.数据库的优化重点是存取效率,机器学习的优化重点是计算效率,两者依赖的具体优化技术存在差别.因此,借鉴数据库系统的设计理念和实现经验进行独立的设计和实现,也是机器学习系统构建的重要方式[64−70].本节首先介绍独立构建机器学习系统的典型工作,之后总结数据库设计思想对于系统构建的参考价值.

SciDB[33]是独立实现的机器学习系统.以多维数组作为数据模型,分 chunk 连续存储;支持面向数组的查询语言(AQL),方便对数组中的数据进行访问和计算表达,同时也兼容 R 语言;计算效率方面,数据组间的运算可以在对齐 chunk 划分边界后,直接在 chunk 间进行;由于存储和计算完全独立实现,SciDB 在效率优化方面具有很高的灵活性.SystemML[64,65]在分布式计算系统 MapReduce,Spark 的基础上实现机器学习系统;数据模型为矩阵,使用 HDFS 进行存储,矩阵运算可以转化 Spark 计算任务;对外提供机器学习领域描述性语言(DSL),转化为 DGA 后,通过自动优化技术转化为物理计划,并根据资源限制调度到单机或者 Spark 集群上运行;由于使用 Spark 作为计算引擎,任务训练可以无缝集成到 Spark 整体数据处理流程中.Cumulon[66]的数据模型也为矩阵,以分块形式存储在 HDFS 中,并对分块矩阵运算自主调度,与使用 MapReduce 进行矩阵计算相比效率更高;Cumulon 考虑了不同机型在内存资源和计算能力的差异,建立了使用不同机型在训练成本和时长的预估模型,因而可以在满足训练时长的要求下,自动选择和生成成本最低的物理计划.

上述系统虽然独立实现,但数据库系统的设计理念和实现经验对其有重要参考意义,总结为 3 点.

1) 面向领域问题的描述性语言:数据库系统成功的一个重要因素是提供描述性语言 SQL,支持方便的描述数据读写需求和较简单的统计计算,而不必关系执行细节.SciDB 支持面向数组的查询语言 AQL,SystemML 和 Cumulon 均支持机器学习领域的描述语言(DSL).描述性语言隐藏了训练过程的细节,可以灵活而高效地构建机器学习模型;
2) 分层设计独立优化:数据库的核心设计理念是保持语言处理层、查询优化层、数据存取层、数据存储层各层之间接口独立,可以灵活组合并分别优化.SystemML 可以对相同的训练任务选择调度到单机或者 Spark 集群,实现了语言处理层与计算存储层的独立,Spark 系统在计算效率方面的优化可以直接在 SystemML 中生效;
3) 自动优化技术:数据库查询优化技术对于任意的 SQL 请求生成效率优化的执行计划,是支撑数据库广泛应用的核心技术.对于机器学习而言,由于计算更加复杂,异构硬件类型更多,对自动优化技术依赖程度更高.SystemML 和 Cumulon 均实现了自动优化技术,将 DSL 编写的训练代码自动转化为高效执行的物理计划,能够系统地提升各类机器学习算法的训练效率.

7.4 小 结

本节从与数据库系统的关系视角对机器学习系统实现进行了分类归纳,前述数据管理技术在不同类别中具有不同应用方式.在数据库基础上,浅层或者深层融合的机器学习系统能够较大程度复用数据库已经提供的数据压缩、视图、缓存、索引以及自动查询优化等技术,并根据需要对数据库数据类型和查询优化技术进行扩展.数据库之外独立实现的机器学习系统通常提供 DSL 进行建模,需要独立实现自动优化技术;数据存储与存取技术可以独立实现,也可以复用文件系统已有功能.由于机器学习技术仍在快速发展过程中,前文讨论的数据选择、多版本存储、数据索引等技术尚未形成完整的系统集成方案,探索各类数据管理技术的统一系统集成方法是未来面临的重要挑战.

8 总结及展望

本文分别从数据和系统的视角对机器学习训练过程进行解构,从数据管理、自动优化、系统设计与实现方面梳理了支撑机器学习的数据管理技术现状.数据选择、数据存储和数据存取技术的优化思路从任务为中心逐渐转化到数据为中心,通过多版本存储、视图、缓存等技术初步实现跨任务数据共享,但尚未形成统一的跨任务数据管理方案.自动优化技术目前主要采取任务为中心的做法,从为训练过程自动优化效率逐渐发展到为训练任务自动生成效果优化的训练过程.自动模型选择需要搜索大量的候选,效率仍面临挑战.机器学习系统需要集成数据管理、自动优化等技术,可以复用已有数据库功能进行扩充与适配,也可以基于数据库的设计思想独

立实现.总结而言,数据库系统长期积累的数据管理技术已经广泛应用到机器学习训练的各个环节.随着更多面向机器学习的工业级应用的出现,构建以数据为中心、支持跨任务数据复用和在线实时建模与训练的新型数据库系统,是提升机器学习效率、降低成本的重要途径.未来的挑战和研究方向包括:

1) 数据库内支持机器学习完整过程.

数据库内支持机器学习能够避免训练数据在数据库和机器学习系统间移动,也能够复用数据完备的安全管理机制保护数据隐私.当前,数据库内支持机器学习主要关注建模和训练过程[54,55],对建模前数据处理和完成训练后在线推理关注相对较少.对于数据处理,结构化和半结构化数据通常抽象为关系表或 KV 表,适合采用 SQL 进行转换计算;非结构化数据尚未形成统一的数据模型,数据处理通常根据具体任务而采用不同的语言和模式;而模型构建通常采用多维矩阵作为通用数据表达,适合采用 DSL 进行建模.因此,需要探索无缝连接数据处理和模型构建的更有效方式,形成统一融合的数据模型和描述性语言[71],降低数据在两个阶段的转换成本.对于在线推理,其挑战包括需要探索模型计算自动优化方式以提升推理效率[72]以及请求量动态变化时,基于异构硬件自适应调度计算以满足推理延时要求[73]等.

2) 跨任务数据与模型的管理与复用.

当前,机器学习数据与模型主要以任务为中心的方式管理,难以实现全局存储与计算最优.未来,随着机器学习的广泛应用,将出现大量训练目标和过程类似的任务,建立跨任务数据与模型关联、实现任务间资源复用,是提升多任务整体训练效率的重要途径.这方面已经有初步研究[74,75],但仍面临许多挑战.

- 首先,需要探索通用的数据语义表示,形成跨任务数据表示基础.非结构化数据的语义表达仍面临挑战,基于深度神经网络非监督训练得到的潜在语义向量表达是有希望的方向,需要进一步探索与结构化数据表达方式的融合;
- 其次,需要探索任务间模型的复用方法.由于机器学习模型,尤其是神经网络模型尚缺乏可解释性,模型复用仍面临挑战.预训练是有希望的方向,通过海量非监督数据训练得到能被多任务复用的公共模型,但仍是整体模型的复用,需要进一步探索模型子结构以实现更细粒度的复用;
- 同时,在多用户、多任务协同的环境中,训练数据、中间结果和模型的安全访问机制,也是值得探索的方向.

3) 支持在线自动模型构建与训练.

自动模型构建与训练将进一步减少机器学习过程的人工参与,提升效率.当前,自动模型构建与训练主要考虑效果优化,对于效率与实时性方面考虑较少.支持在线自动模型构建与训练要求在更低资源开销下达到要求的训练效果,面临许多挑战:首先,需要探索训练数据、算法、超参数更高效的搜索与组合方法,更早过滤效果次优的训练方案;其次,需要更准确地预估不同方案的训练耗时,比如在训练代价估算中同时考虑数据和模型的元数据信息以及考虑数据和模型复用带来训练耗时降低.同时,自动模型构建与训练需要具备自学习特性,可以将人工建模的过程作为学习对象.随着人工建模过程和效果评测数据的积累,学习其中潜在的建模和训练优化策略,持续优化自动模型构建和训练的效果.

References:

[1] Deng J, Dong W, Socher R, Li LJ, Li K, Li FF. Imagenet: A large-scale hierarchical image database. In: Proc. of the 2009 IEEE Conf. on Computer Vision and Pattern Recognition. IEEE, 2009. 248–255.

[2] Lian Z, Li Y, Tao J, Huang J. Improving speech emotion recognition via transformer-based predictive coding through transfer learning. arXiv. Nov:arXiv-1811. 2018.

[3] Devlin J, Chang MW, Lee K, Toutanova K. Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805. 2018.

[4] Du XY, Lu W, Zhang F. History, present, and future of big data management systems. Ruan Jian Xue Bao/Journal of Software, 2019,30(1):127−141 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/5644.htm [doi: 10.13328/j.cnki.jos.005644]

[5] Strubell E, Ganesh A, McCallum A. Energy and policy considerations for deep learning in NLP. arXiv preprint arXiv:1906.02243. 2019.

[6] Weizenbaum J. ELIZA—A computer program for the study of natural language communication between man and machine. Communications of the ACM, 1966,9(1):36−45.

[7] SVM. 2020. https://en.wikipedia.org/wiki/Support_vector_machine

[8] CRF. 2020. https://en.wikipedia.org/wiki/Conditional_random_field

[9] LDA. 2020. https://en.wikipedia.org/wiki/Latent_Dirichlet_allocation

[10] Liu TY. Learning to Rank for Information Retrieval. Springer Science & Business Media, 2011.

[11] Brown TB, Mann B, Ryder N, Subbiah M, Kaplan J, Dhariwal P, Neelakantan A, Shyam P, Sastry G, Askell A, Agarwal S. Language models are few-shot learners. arXiv preprint arXiv:2005.14165. 2020.

[12] Boutsidis C, Drineas P, Magdon-Ismail M. Near-Optimal coresets for least-squares regression. IEEE Trans. on Information Theory, 2013,59(10):6880−6892.

[13] Mikolov T, Chen K, Corrado G, Dean J. Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781. 2013.

[14] Le Q, Mikolov T. Distributed representations of sentences and documents. In: Proc. of the Int'l Conf. on Machine Learning. 2014. 1188−1196.

[15] Pu Y, Gan Z, Henao R, Yuan X, Li C, Stevens A, Carin L. Variational autoencoder for deep learning of images, labels and captions. In: Proc. of the Advances in Neural Information Processing Systems. 2016. 2352−2360.

[16] Axelrod A, He X, Gao J. Domain adaptation via pseudo in-domain data selection. In: Proc. of the 2011 Conf. on Empirical Methods in Natural Language Processing. 2011. 355−362.

[17] Moore RC, Lewis W. Intelligent selection of language model training data.

[18] Perplexity. 2020. https://en.wikipedia.org/wiki/Perplexity

[19] Chen B, Huang F. Semi-Supervised convolutional networks for translation adaptation with tiny amount of in-domain data. In: Proc. of the 20th SIGNLL Conf. on Computational Natural Language Learning. 2016. 314−323.

[20] Active learning. 2020. https://en.wikipedia.org/wiki/Active_learning_(machine_learning)

[21] Wei K, Iyer R, Bilmes J. Submodularity in data subset selection and active learning. In: Proc. of the Int'l Conf. on Machine Learning. 2015. 1954−1963.

[22] Wang R, Utiyama M, Sumita E. Dynamic sentence sampling for efficient training of neural machine translation. arXiv preprint arXiv:1805.00178. 2018.

[23] TFRecord. 2020. https://www.tensorflow.org/tutorials/load_data/tfrecord

[24] Protobuf. 2020. https://developers.google.com/protocol-buffers

[25] ONNX. 2020. https://en.wikipedia.org/wiki/Open_Neural_Network_Exchange

[26] Vartak M, Subramanyam H, Lee WE, Viswanathan S, Husnoo S, Madden S, Zaharia M. ModelDB: A system for machine learning model management. In: Proc. of the Workshop on Human-in-the-loop Data Analytics. 2016. 1−3.

[27] Zhang Z, Sparks ER, Franklin MJ. Diagnosing machine learning pipelines with fine-grained lineage. In: Proc. of the 26th Int'l Symp. on High-Performance Parallel and Distributed Computing. 2017. 143−153.

[28] George L. HBase: The Definitive Guide: Random Access to Your Planet-Size Data. O'Reilly Media, Inc., 2011.

[29] AWS S3. 2020. https://aws.amazon.com/s3/

[30] Bhattacherjee S, Chavan A, Huang S, Deshpande A, Parameswaran A. Principles of dataset versioning: Exploring the recreation/storage tradeoff. Proc. of the VLDB Endowment, 2015,8(2):1346.

[31] Bhardwaj A, Bhattacherjee S, Chavan A, Deshpande A, Elmore AJ, Madden S, Parameswaran AG. Datahub: Collaborative data science & dataset version management at scale. arXiv preprint arXiv:1409.0798. 2014.

[32] Miao H, Li A, Davis LS, Deshpande A. Modelhub: Towards unified data and lifecycle management for deep learning. arXiv preprint arXiv:1611.06224. 2016.

[33] Stonebraker M, Brown P, Poliakov A, Raman S. The architecture of SciDB. In: Proc. of the Int'l Conf. on Scientific and Statistical Database Management. Berlin, Heidelberg: Springer-Verlag, 2011. 1−16.

[34] Snappy. 2020. https://en.wikipedia.org/wiki/Snappy_(compression)

[35] Elgohary A, Boehm M, Haas PJ, Reiss FR, Reinwald B. Compressed linear algebra for large-scale machine learning. Proc. of the VLDB Endowment, 2016,9(12):960−971.

[36] Run-Length_Encoding. 2020. https://en.wikipedia.org/wiki/Run-length_encoding

[37] Li F, Chen L, Kumar A, Naughton JF, Patel JM, Wu X. When lempel-ziv-welch meets machine learning: A case study of accelerating machine learning using coding. arXiv preprint arXiv:1702.06943. 2017.

[38] Tabei Y, Saigo H, Yamanishi Y, Puglisi SJ. Scalable partial least squares regression on grammar-compressed data matrices. In: Proc. of the 22nd ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining. 2016. 1875−1884.

[39] Zhang C, Kumar A, Ré C. Materialization optimizations for feature selection workloads. ACM Trans. on Database Systems (TODS), 2016,41(1):1−32.

[40] Deshpande A, Madden S. MauveDB: Supporting model-based user views in database systems. In: Proc. of the 2006 ACM SIGMOD Int'l Conf. on Management of Data. 2006. 73−84.

[41] Nikolic M, ElSeidy M, Koch C. LINVIEW: Incremental view maintenance for complex analytical queries. In: Proc. of the 2014 ACM SIGMOD Int'l Conf. on Management of Data. 2014. 253−264.

[42] Anderson MR, Cafarella M. Input selection for fast feature engineering. In: Proc. of the 2016 IEEE 32nd Int'l Conf. on Data Engineering (ICDE). IEEE, 2016. 577−588.

[43] Zhang Y, Munagala K, Yang J. Storing matrices on disk: Theory and practice revisited. Proc. of the VLDB Endowment, 2011,4(11): 1075−1086.

[44] Sparks ER, Venkataraman S, Kaftan T, Franklin MJ, Recht B. Keystoneml: Optimizing pipelines for large-scale advanced analytics. In: Proc. of the 2017 IEEE 33rd Int'l Conf. on Data Engineering (ICDE). IEEE, 2017. 535−546.

[45] Transfer learning. 2020. https://en.wikipedia.org/wiki/Transfer_learning

[46] Boehm M, Burdick DR, Evfimievski AV, Reinwald B, Reiss FR, Sen P, Tatikonda S, Tian Y. SystemML's optimizer: Plan generation for large-scale machine learning programs. IEEE Data Engineering Bulletin, 2014,37(3):52−62.

[47] Sujeeth AK, Lee H, Brown KJ, Rompf T, Chafi H, Wu M, Atreya AR, Odersky M, Olukotun K. OptiML: An implicitly parallel domain-specific language for machine learning. In: Proc. of the ICML. 2011.

[48] Kumar A, McCann R, Naughton J, Patel JM. Model selection management systems: The next frontier of advanced analytics. ACM SIGMOD Record, 2016,44(4):17−22.

[49] Zoph B, Le QV. Neural architecture search with reinforcement learning. arXiv preprint arXiv:1611.01578. 2016.

[50] Xie L, Yuille A. Genetic CNN. In: Proc. of the IEEE Int'l Conf. on Computer Vision. 2017. 1379−1388.

[51] Baker B, Gupta O, Naik N, Raskar R. Designing neural network architectures using reinforcement learning. arXiv preprint arXiv: 1611.02167. 2016.

[52] Yosinski J, Clune J, Bengio Y, Lipson H. How transferable are features in deep neural networks? In: Proc. of the Advances in Neural Information Processing Systems. 2014. 3320−3328.

[53] Kumar A, Boehm M, Yang J. Data management in machine learning: Challenges, techniques, and systems. In: Proc. of the 2017 ACM Int'l Conf. on Management of Data. 2017. 1717−1722.

[54] Cohen J, Dolan B, Dunlap M, Hellerstein JM, Welton C. MAD skills: New analysis practices for big data. Proc. of the VLDB Endowment, 2009,2(2):1481−1492.

[55] Hellerstein J, Ré C, Schoppmann F, Wang DZ, Fratkin E, Gorajek A, Ng KS, Welton C, Feng X, Li K, Kumar A. The MADlib analytics library or MAD skills, the SQL. arXiv preprint arXiv:1208.4165. 2012.

[56] Feng X, Kumar A, Recht B, Ré C. Towards a unified architecture for in-RDBMS analytics. In: Proc. of the 2012 ACM SIGMOD Int'l Conf. on Management of Data. 2012. 325−336.

[57] Cheng Y, Qin C, Rusu F. GLADE: Big data analytics made easy. In: Proc. of the 2012 ACM SIGMOD Int'l Conf. on Management of Data. 2012. 697−700.

[58] Rusu F, Dobra A. GLADE: A scalable framework for efficient analytics. ACM SIGOPS Operating Systems Review, 2012,46(1): 12−18.

[59] Luo S, Gao ZJ, Gubanov M, Perez LL, Jermaine C. Scalable linear algebra on a relational database system. IEEE Trans. on Knowledge and Data Engineering, 2018,31(7):1224−1238.

[60] Eigen. 2020. http://eigen.tuxfamily.org/

[61] Mahajan D, Kim JK, Sacks J, Ardalan A, Kumar A, Esmaeilzadeh H. In-RDBMS hardware acceleration of advanced analytics. arXiv preprint arXiv:1801.06027. 2018.

[62] Cai Z, Vagena Z, Perez L, Arumugam S, Haas PJ, Jermaine C. Simulation of database-valued Markov chains using SimSQL. In: Proc. of the 2013 ACM SIGMOD Int'l Conf. on Management of Data. 2013. 637−648.

[63] Kara K, Eguro K, Zhang C, Alonso G. ColumnML: Column-store machine learning with on-the-fly data transformation. Proc. of the VLDB Endowment, 2018,12(4):348−361.

[64] Ghoting A, Krishnamurthy R, Pednault E, Reinwald B, Sindhwani V, Tatikonda S, Tian Y, Vaithyanathan S. SystemML: Declarative machine learning on MapReduce. In: Proc. of the 2011 IEEE 27th Int'l Conf. on Data Engineering. IEEE, 2011. 231−242.

[65] Boehm M, Dusenberry MW, Eriksson D, Evfimievski AV, Manshadi FM, Pansare N, Reinwald B, Reiss FR, Sen P, Surve AC, Tatikonda S. Systemml: Declarative machine learning on spark. Proc. of the VLDB Endowment, 2016,9(13):1425−1436.

[66] Huang B, Babu S, Yang J. Cumulon: Optimizing statistical data analysis in the cloud. In: Proc. of the 2013 ACM SIGMOD Int'l Conf. on Management of Data. 2013. 1−12.

[67] Brown PG. Overview of SciDB: Large scale array storage, processing and analysis. In: Proc. of the 2010 ACM SIGMOD Int'l Conf. on Management of Data. 2010. 963−968.

[68] Xin RS, Rosen J, Zaharia M, Franklin MJ, Shenker S, Stoica I. Shark: SQL and rich analytics at scale. In: Proc. of the 2013 ACM SIGMOD Int'l Conf. on Management of Data. 2013. 13−24.

[69] Stonebraker M, Madden S, Dubey P. Intel "big data" science and technology center vision and execution plan. ACM SIGMOD Record, 2013,42(1):44−49.

[70] Zhang Y, Zhang W, Yang J. I/O-Efficient statistical computing with RIOT. In: Proc. of the 2010 IEEE 26th Int'l Conf. on Data Engineering (ICDE 2010). IEEE, 2010. 1157−1160.

[71] Zhou X, Chai C, Li G, Sun J. Database meets artificial intelligence: A survey. IEEE Trans. on Knowledge and Data Engineering, 2020.

[72] Lee Y, Scolari A, Chun BG, Santambrogio MD, Weimer M, Interlandi M. {PRETZEL}: Opening the black box of machine learning prediction serving systems. In: Proc. of the 13th {USENIX} Symp. on Operating Systems Design and Implementation (OSDI 2018). 2018. 611−626.

[73] Wang W, Wang S, Gao J, Zhang M, Chen G, Ng TK, Ooi BC. Rafiki: Machine learning as an analytics service system. arXiv preprint arXiv:1804.06087. 2018.

[74] Smith MJ, Sala C, Kanter JM, Veeramachaneni K. The machine learning bazaar: Harnessing the ML ecosystem for effective system development. In: Proc. of the 2020 ACM SIGMOD Int'l Conf. on Management of Data. 2020. 785−800.

附中文参考文献:

[4] 杜小勇,卢卫,张峰.大数据管理系统的历史、现状与未来.软件学报,2019,30(1):127−141. http://www.jos.org.cn/1000-9825/5644.htm [doi: 10.13328/j.cnki.jos.005644]

崔建伟(1986－),男,博士生,CCF 学生会员,主要研究领域为深度学习,自然语言处理.

赵哲(1992－),男,博士生,主要研究领域为深度学习,自然语言处理.

杜小勇(1963－),男,博士,教授,博士生导师,CCF 会士,主要研究领域为数据库,大数据系统.