DOI:10.7544/issn1000-1239.2017.20170086
54(6):1337-1347,2017

机器学习算法可近似性的量化评估分析

江树浩[1,2]　鄢贵海[1]　李家军[1,2]　卢文岩[1,2]　李晓维[1]

[1](计算机体系结构国家重点实验室(中国科学院计算技术研究所)　北京　100190)

[2](中国科学院大学　北京　100049)

(jiangshuhao@ict.ac.cn)

A Quantitative Analysis on the "Approximatability" of Machine Learning Algorithms

Jiang Shuhao[1,2], Yan Guihai[1], Li Jiajun[1,2], Lu Wenyan[1,2], and Li Xiaowei[1]

[1](*State Key Laboratory of Computer Architecture* (*Institute of Computing Technology*, *Chinese Academy of Sciences*), *Beijing* 100190)

[2](*University of Chinese Academy of Sciences*, *Beijing* 100049)

Abstract　Recently, Machine learning algorithms, such as neural network, have made a great progress and are widely used in image recognition, data searching and finance analysis field. The energy consumption of machine learning algorithms becomes critical with more complex issues and higher data dimensionality. Because of the inherent error-resilience of machine learning algorithms, approximate computing techniques, which trade the accuracy of results for energy savings, are applied to save energy consumption of these algorithms by many researchers. We observe that most works are dedicated to leverage the error-resilience of certain algorithm while they ignore the difference of error-resilience among different algorithms. Understanding the difference on "approximatability" of different algorithms is very essential because when the approximate computing techniques are applied, approximatability can help the classification tasks choose the best algorithms to achieve the most energy savings. Therefore, we choose 3 common supervised learning algorithms, that is, SVM, random forest (RF) and neural network (NN), and evaluate their approximatability targeted to different kinds of energy consumption. Meanwhile, we also design several metrics such as memory storage contamination sensitivity, memory access contamination sensitivity and energy diversity to quantify the difference on approximatability of learning algorithms. The conclusion from evaluation will assist in choosing the appropriate learning algorithms when the classification applications apply approximate computing techniques.

Key words　supervised machine learning algorithm; approximate computing; approximatability; energy consumption optimization; quantitative model

摘　要　近年来,以神经网络为代表的机器学习算法发展迅速并被广泛应用在图像识别、数据搜索乃至金融趋势分析等领域.而随着问题规模的扩大和数据维度的增长,算法能耗问题日益突出,由于机器学

收稿日期:2017-02-22;**修回日期**:2017-04-13

基金项目:国家自然科学基金项目(61572470,61532017,61522406,61432017,61376043,61521092);中国科学院青年创新促进会项目(404441000)

This work was supported by the National Natural Science Foundation of China (61572470, 61532017, 61522406, 61432017, 61376043, 61521092) and the Youth Innovation Promotion Association, CAS (404441000).

通信作者:李晓维(lxw@ict.ac.cn)

习算法自身拥有的近似特性,近似计算这种牺牲结果的少量精确度降低能耗的技术,被许多研究者用来解决学习算法的能耗问题.我们发现,目前的工作大多专注于利用特定算法的近似特性而忽视了不同算法近似特性的差别对能耗优化带来的影响,而为了分类任务使用近似计算时能够做出能耗最优的选择,了解算法"可近似性"上的差异对近似计算优化能耗至关重要.因此,选取了支持向量机(SVM)、随机森林(RF)和神经网络(NN) 3 类常用的监督型机器学习算法,评估了针对不同类型能耗时不同算法的可近似性,并建立了存储污染敏感度、访存污染敏感度和能耗差异度等指标来表征算法可近似性的差距,评估得到的结论将有助于机器学习算法在使用近似计算技术时达到最优化能耗的目的.

关键词 监督机器学习算法;近似计算;可近似性;能耗优化

中图法分类号 TP391

机器学习算法被广泛应用在识别、数据分析和搜索等领域的分类问题上.由于这类问题往往涉及大量高维度数据、复杂的训练及预测方法,算法的执行过程会消耗大量的能量.因此如何降低监督学习算法的能耗已经成为计算机领域研究热点之一.

近似计算作为一种非常有潜力的技术被用来优化机器学习算法的能耗.如文献[1]通过多种近似方法使得 KNN 算法的能耗降低了 55%,文献[2]通过对人工神经网的近似获得了 43%的能耗节省.这些近似计算工作的主要思路是利用算法训练或预测的过程中结果对误差的容忍性,通过近似数据或省略部分非关键计算,只引入少量的输出上的误差而极大地降低算法的能耗.

我们发现,目前的工作大多专注于利用特定算法的近似特性而忽视了不同算法近似特性的差别对能耗优化带来的影响.我们定义"可近似性"(approximatability, *Xbility*)为近似计算所节省的能耗 E_{savings} 与输出结果质量下降 Q_{loss} 的比值:

$$Xbility \triangleq \frac{E_{\text{savings}}}{Q_{\text{loss}}}, \tag{1}$$

其中,对 Q_{loss} 的度量需要根据具体的问题确定,如在图像处理应用中,Q_{loss} 通常以近似前后图像峰值信噪比(PSNR)损失值为度量标准,在本文中 Q_{loss} 为算法近似前后分类准确度的损失值.可近似性表征了单位结果质量的下降所能换取的能耗节省.了解不同算法"可近似性"的差异对近似计算优化能耗有着非常重要的作用,对算法"可近似性"的评估可以保证应用使用近似计算时选出能耗最优的算法.

对算法可近似性的评估首先需要确定要优化的能耗类型.机器学习算法处理分类问题时分为 2 个阶段:训练阶段和分类阶段,相比于分类阶段,训练阶段中影响能耗因素较多,如学习率、batch 大小和惩罚因子等,这些因素使得在训练阶段中可近似性评估的公平性难以被保证;其次对应用来说,训练过程的主要目的是确定分类所需的参数值,可以通过离线执行预先设定好,而算法分类阶段是解决问题的过程,对应用的执行不可或缺,因此本文主要关注分类阶段的能耗.根据计算系统层次和耗能因素,我们将分类阶段的能耗分解为

$$E = E_{\text{SMS}} + E_{\text{SMA}} + E_{\text{MMA}} + E_{\text{MCP}}, \tag{2}$$

其中,E_{SMS} 代表样本数据的存储能耗,本文中指样本数据存储在 DRAM 存储单元中消耗的能量;E_{SMA} 代表样本访存的能耗.式(2)等号右边前 2 项的能耗值主要由样本数据量所决定而与算法类型无关,因此对不同算法,这 2 项的近似前能耗值相等,近似后的能耗差异由算法可近似性差异所决定.而式(2)的最后 2 项为算法相关能耗,其近似前能耗值与算法类型相关,近似后的能耗差异受算法可近似性和近似前能耗差异 2 方面的影响,其中 E_{MMA} 代表访存算法参数所消耗的能量,E_{MCP} 代表算法计算所消耗的能量.

本文选取了 3 类常用的机器学习算法:支持向量机(SVM)、随机森林(RF)和神经网络(NN)算法(其中 NN 算法包括卷积神经网(CNN)和多层感知机(MLP)),在保证初始分类准确度一致的前提下,评估了针对上述不同类型的能耗不同算法可近似性的差异.

评估结果表明:算法可近似性的差异确实会对能耗优化产生重要影响,并且针对不同能耗,算法的可近似性并不固定.首先,针对样本存储能耗,对近似产生的突变噪声更加鲁棒的 RF 算法的可近似性最高,与 NN 算法相比,可多节省 14%的存储能耗.其次,针对样本访存能耗,NN 算法由于训练过程复杂易陷入局部最优等不稳定的特点使得在降低样本数据精度时分类准确度下降明显,而 SVM 算法和 RF 算法对访存近似更加鲁棒,平均节省能耗比 NN 算法分别多 16%和 11%.最后,针对与算法有关的参数访存能耗和计算能耗,由于不同算法近似前的

能耗差异很大,算法的可近似性差异对能耗优化的影响将十分有限,分类应用选择算法的依据将更多取决于参数访存能耗和计算能耗的差异度.

为了量化不同算法针对不同能耗类型的可近似性,我们建立了样本存储污染敏感度、样本访存污染敏感度和能耗差异度等评估指标,这些指标可以为分类应用选择能耗最优的算法提供依据.

1 相关工作

近似计算技术利用了算法对误差的容忍特性,其基本原理是通过误差注入等静态分析手段,得到并控制近似技术的近似程度,使得近似后的算法仍然满足某些指标(如 PSNR、MSE、分类准确度等)的要求.但目前的工作大多专注于利用特定算法,本文在此基础上评估了不同算法的可近似性差异,并分析了这种差异对算法能耗优化带来的影响.

针对多种类型的能耗,近似计算技术提出了多种提高效能的方法,常用的方法包括降低存储单元刷新率、电压调节、使用非精确逻辑电路、省略非关键计算、比特位截断等[1-12].下面根据能耗类型(对应式(2))介绍近似计算技术的相关工作.

1) 针对存储能耗.文献[13-14]等工作发现大多数 DRAM 存储单元数据可保持时间高于目前设定的刷新时间,因此可以适当延长 DRAM 刷新时间,节省大量的存储能耗,其中文献[13]的方法能够降低 74.6%的刷新频率、16.1%的 DRAM 能耗和 8.6%的系统整体能耗.电压调节技术也是降低存储能耗的手段之一,文献[15-17]分别通过适当地降低 cache、寄存器和功能性单元的供电电压来节省存储能耗.

2) 针对访存能耗.位截断技术是通过近似数据来降低访存能耗的重要方法之一,文献[18]通过比特位截断将 k 近邻算法和高斯混合聚类算法的能耗平均降低了 50%.文献[19]通过省略非重要节点的访存操作节省数据访存能耗.此外,对数据进行有损压缩也是降低访存能耗的方法之一,实验表明通过对 GPGPU 和片外存储间传输的浮点数进行压缩可以减少 41%的 GPGPU 负载[20].

3) 针对计算能耗.文献[21-23]等使用近似重用技术优化计算能耗,通过对输入相近程度的有效度量和条件分支结果的有效预测,相似的输入结果只需被计算一次,实验结果表明近似重用技术可获得 3.1 倍的能耗节省.文献[24-25]等工作通过设计近似加法器、乘法器等近似电路代替精确电路降低计算能耗,文献[2,26]则根据神经网络中各个节点对分类结果的影响力,近似对结果影响较小的节点计算达到优化网络能耗的目的.

2 样本存储能耗的算法可近似性评估

样本在主存中的存储能耗 E_{SMS} 是分类应用整体能耗的重要部分,根据文献[15,27]的能耗模型,表 1 评估了在不同应用中式(2)中各部分能耗所占比例,其中样本存储能耗占整体能耗的比例平均为 25%左右.需要指出的是,各部分能耗比例是随具体的参数可变的,如存储时长、样本数目等,本文列出的能耗分布比例是通过评估 NN 算法分类单样本的过程获得.

Table 1 Distribution of Energy Consumption on Different Datasets

表 1 不同类型能耗在不同数据集上的能耗分布 %

Type	HAR	Iris	MNIST	Adult	Average
E_{SMS}	36.14	27.68	4.73	31.64	25.05
E_{SMA}	27.10	20.76	3.55	23.73	18.79
E_{MMA}	27.63	46.71	30.57	35.59	35.13
E_{MCP}	9.13	4.84	61.14	9.04	21.04

节省存储能耗是近似计算优化能耗的一个重要方面,而随着 DRAM 密度越来越大,单位时间内需要刷新的存储单元越来越多,刷新操作的能耗开销越来越大[25].降低 DRAM 刷新率已成为降低存储能耗最常用的存储近似手段[13-14].

DRAM 刷新率的降低会造成一部分存储单元的数据丢失,对于分类任务的样本数据,这种近似手段将导致部分数据产生较大偏差,这相当于这些数据点受到噪声干扰而导致数值突变.DRAM 刷新率越低,受到干扰的数据点比例越高,算法分类精度受到的影响越大.为分析 3 种算法对降低刷新率方法的可近似性,我们评估了不同比例的噪声数据下分类准确度的下降程度.具体做法为:选择 4 个常用数据集,分别为 HAR,Iris,MNIST 和 Adult 数据集,在每个数据集上,通过适当的训练保证 3 类算法的初始分类性能分类准确度差异小于 1%(NN 算法在不同数据集上使用的结构略有差异,在 MNIST 数据集上,算法为 LeNet[28] 的 CNN 结构,而在其他 3 个数据集上算法使用多层感知机(MLP)结构),最后设定不同数值的噪声点比例,比较其对算法分类精度造成的影响,为使噪声点数据和原数据有足够

大的差异,我们将噪声点的值设定为原数据的相反数.

评估结果如图 1 所示,横坐标代表噪声数据点占所有样本数据的比例,纵坐标代表算法的分类准确度.从图 1 中我们可以看到,不同算法的可近似性有明显的差异,其中 RF 算法对噪声的干扰有最好的可近似性,MLP 算法的可近似性最差,SVM 算法的可近似性介于二者之间.如在 HAR 数据集上,当噪声点比例为 10%时,RF 算法的分类准确度仍能达到 90%以上,而 MLP 算法的分类精度仅为 53%左右.

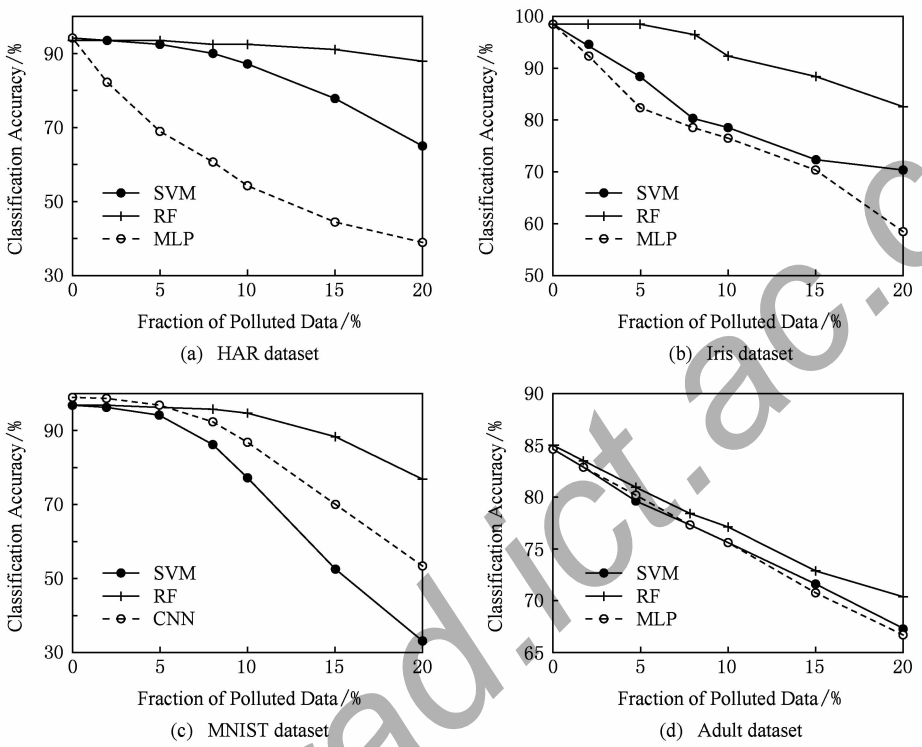

Fig. 1 Comparison on memory storage (MS) approximatability of different algorithms

图 1 不同算法的存储可近似性比较

需要指出的是,这种差异性的大小也会受到应用的影响,对于分类结果对输入噪声不敏感的应用,算法间的可近似性差异较小.如在 Adult 数据集中,SVM,RF 和 MLP 三种算法的曲线趋势非常接近.另一个有趣的发现为:在 MNIST 数据集(图 1(c))上,CNN 的可近似性要高于 SVM 算法,而在其他数据集上,相似结构的 MLP 算法可近似性却低于 SVM 算法,我们认为这是由于卷积神经网的池化层对特征的下采样过程具有去除部分噪声点的能力,因而增强了卷积神经网的抗突变噪声的能力.

为了以数值化的形式表征算法的可近似性的差异,我们对上述曲线进行线性拟合,拟合斜率的绝对值我们定义为存储污染敏感度 S_{MS},S_{MS} 表征了单位比例的噪声数据所造成的分类精度的平均下降程度,S_{MS} 越小,说明算法对存储近似的敏感度越低,能获得的存储能耗节省越大.表 2 列出了 3 种算法各自的存储污染敏感度,根据算法存储污染敏感度的数值可以得出:噪声数据比例每增加 1%,SVM 算法的分类精度平均下降 1.78%,RF 算法平均下降 0.7%,NN 算法平均下降 1.95%.

Table 2 S_{MS} of Three Algorithms

表 2 3 种算法存储污染敏感度

Algorithm	HAR	Iris	MNIST	Adult	Average
SVM	1.43	1.46	3.36	0.87	1.78
RF	0.26	0.84	0.94	0.75	0.70
NN	2.71	1.85	2.34	0.91	1.95

根据文献[13],DRAM 存储单元错误率和刷新时间呈对数关系,而存储能耗随刷新时间的增加线性递减[14],根据这 2 种关系,我们可以得到在满足应用对分类精度最低要求的情况下不同算法的能耗节省值.如图 2 所示,横坐标轴括号内数值代表了应用对分类精度的最低要求,其决定了算法能容忍的存储单元错误率;纵坐标代表了通过降低刷新率延长刷新时间所能节省的能耗,初始刷新时间的设定参照了文献[13]中的结果.总体上,RF 算法在 4 个

数据集中都能节省最多的存储能耗,RF 算法的平均节省能耗值比 SVM 算法多 10%以上,比 NN 算法可节省能耗多 14%以上.而在 HAR 数据集上,RF 算法的节省能耗比 NN 算法可节省能耗多 25%.

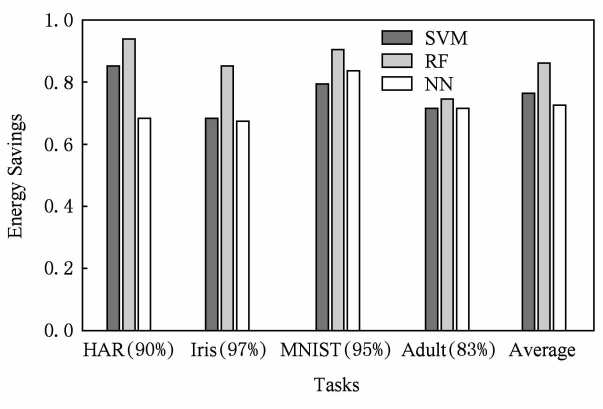

Fig. 2 The comparison on MS energy savings of different algorithms

图 2 3 种算法的可节省存储能耗比较

上述分析说明,针对样本存储能耗 E_{SMS},相比于其他 2 种算法,RF 算法由于对突变噪声的容忍程度最高,因而具有最低的存储污染敏感度,在使用降低刷新率技术时能够节省更多的存储能耗.

3 样本访存能耗的算法可近似性评估

样本访存能耗 E_{SMA},即从主存向缓存传输样本数据的能耗,样本访存能耗是分类应用总能耗中不可忽略的一部分,而且随样本数据量增大而增多.根据文献[29],从主存传输数据到缓存的能耗为片内数据传输能耗的 19 倍,表 1 显示单样本分类情况下 E_{SMA} 占整体能耗的平均比例为 18.79%,因此通过近似计算技术优化样本访存能耗十分必要.

位截断是最直观和最常用的降低访存能耗的近似计算方法,图 3 表示了针对浮点数的比特位截断技术的方式,该方法的近似对象是浮点数的尾数部分,通过舍弃尾数部分的部分低位而只保留数据的高位部分降低访存能耗.

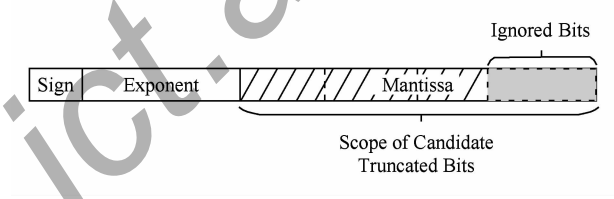

Fig. 3 Overview of precision scaling

图 3 位截断技术示意图

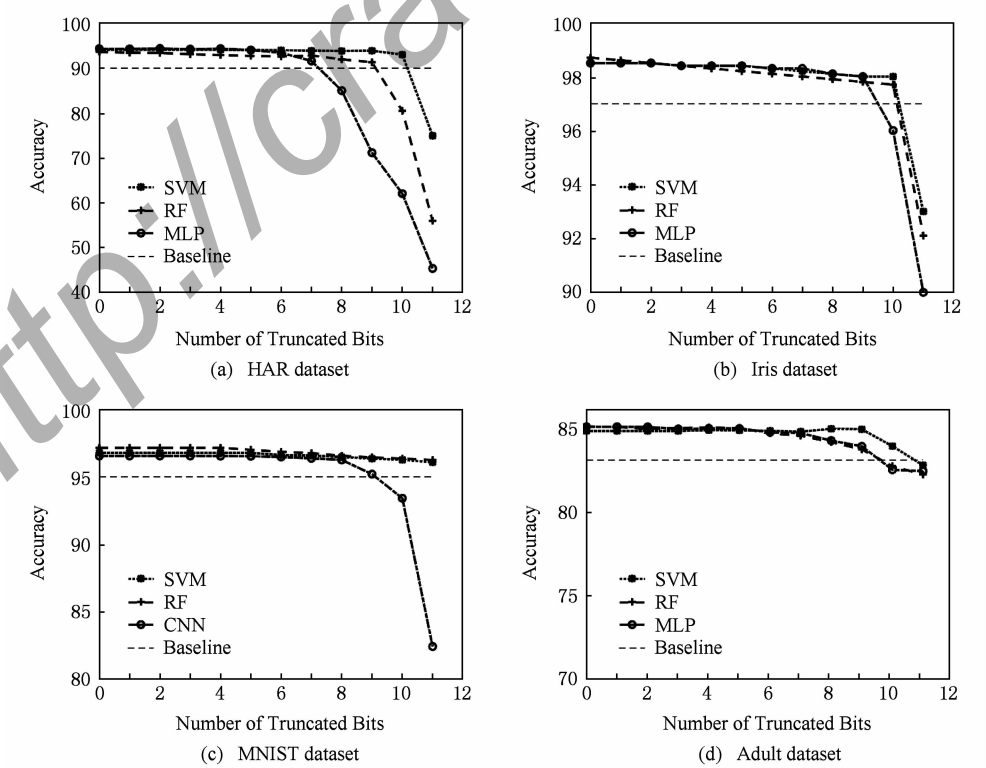

Fig. 4 Comparison on memory access (MA) approximatability of different algorithms

图 4 3 种算法的访存可近似性比较

位截断技术会造成数据的精度下降,原本数值相近的数据差异变得更小,甚至可能会被表示为同一数值.随着截断位数的增多,样本数据将更加"模糊",更不容易被正确分类.对算法而言,算法对模糊数据的区分度越好,算法可近似性就会越高,可节省更多的访存能耗.

为了评估算法针对样本访存能耗的可近似性,我们在不同数据集上分别用3种机器学习算法分类,在3种算法的初始分类精度一致(<1%)的前提下,比较不同程度的比特位近似下算法分类精度受到的影响.实验结果如图4所示,横轴和纵轴分别代表被舍弃的低位位数和分类精度,虚线代表应用对分类精度的最低要求.

我们从图4中发现,不同于算法的存储可近似性,使用位截断技术时,随着截断比特位数的增加,算法分类准确度的下降趋势具有明显的阶段性.在截断位数较少时(阶段1),截断位数的增加只会造成分类精度的微量的降低;而当截断尾数增加到某一临界位后(阶段2),分类准确度明显下降,截断位数的增加对分类准确度的影响十分显著.

其次,图4的结果表明在近似访存能耗时,不同算法的可近似性有比较明显的差异.如在HAR数据集上,SVM算法的临界值为10位,而NN算法的临界值仅为7位.此外,算法针对访存能耗的可近似性差异也受到应用的影响.在Iris数据集中,SVM算法和RF算法的临界值均为10,而NN的算法的临界值仅比其他2种算法少一位.总体来说,SVM算法和RF算法在使用比特位截断技术时可近似性较好,而NN算法的可近似性较差.

我们通过访存污染敏感度S_{MA}来量化针对访存能耗的算法可近似性,由于阶段2的算法分类准确度过低,而阶段1的分类准确度下降缓慢基本能够满足应用要求,因此以临界位定义访存污染敏感度S_{MA}:

$$S_{MA}=\frac{TruncatedScope}{Threshold}, \tag{3}$$

其中,分子代表尾数范围(本文中该值为11),分母为临界位的大小.S_{MA}的范围为$1\sim+\infty$,1代表算法的临界值达到了尾数的最大范围,正无穷代表临界值为0,即算法不能忍受对样本数据的近似,S_{MA}越小,说明算法对比特位截断的敏感度越低,可节省的访存能耗越多.

通过式(3),我们得到3种算法在不同应用的S_{MA}值,如表3所示,根据平均访存污染敏感度值,SVM算法的可近似性最高,NN算法的可近似性最低,RF算法介于两者之间.

Table 3 S_{MA} of Three Algorithms

表3 3种算法访存污染敏感度

Algorithm	HAR	Iris	MNIST	Adult	Average
SVM	1.10	1.10	1.00	1.10	1.07
RF	1.22	1.10	1.00	1.22	1.14
NN	1.57	1.22	1.22	1.22	1.31

节省访存能耗的多少和截断位数成正比,图5表示了在满足应用对分类精度最低要求的情况下不同算法的能耗节省值.从图5中可以看出,在HAR数据集上,SVM算法可节省的访存能耗分别比NN算法和RF算法多37%和10%;而对于平均可节省能耗,SVM算法可比NN算法平均多节省16%的访存能耗.

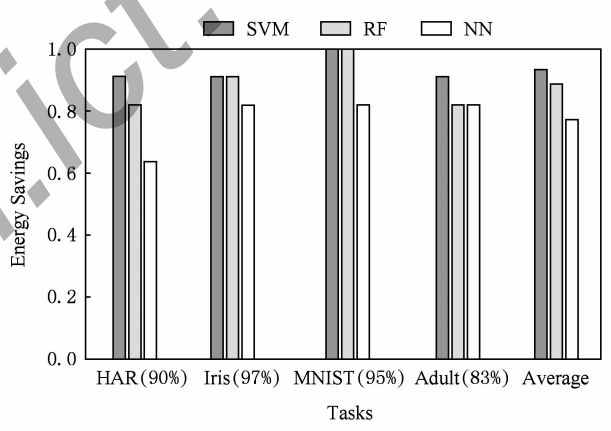

Fig. 5 The comparison on MA energy savings of different algorithms

图5 3种算法的可节省访存能耗比较

上述分析说明,针对样本存储能耗E_{SMA},相比于其他2种算法,SVM算法由于对突变噪声的容忍程度最高,因而具有最低的访存污染敏感度,在使用比特位截断技术时能够节省更多的存储能耗.

4 算法参数访存能耗和计算能耗的评估方法

参数访存能耗E_{MMA}和计算能耗E_{MCP}(这里统称为分类能耗)均与算法类型有关,即在应用近似计算技术前,不同算法的E_{MMA}和E_{MCP}是有差别的,不同算法近似后的能耗对比不仅取决于通过近似计算所能节省的能耗,还与算法的近似前能耗的差异有关.因此在评估不同算法针对这部分能耗的可近似性时,需要首先考虑算法近似前的初始能耗.

需要指出的是,参数访存能耗 E_{MMA} 的评估不可省略,因为对于某些算法,其参数数据量很大,而且在样本数量较少时,计算能耗所占比例并不高.

本节着重分析单样本分类任务时的初始参数访存能耗与计算能耗的评估方法,首先我们将建立参数访存/计算能耗模型,随后我们在算法指令层分别介绍不同算法的参数访存负载和计算负载组成.

4.1 参数访存/计算能耗模型

正如引言所提到的,单样本分类能耗(E_{MODEL})由 2 部分组成:算法参数的访存能耗 E_{MMA} 和计算能耗 E_{MCP},以 F_{MMA} 代表算法参数的访存负载(访存操作的数目),E_{MCP} 代表算法计算负载(主要 ALU 操作的数目),则分类能耗模型为

$$E_{MODEL}=\lambda\mu\times F_{MMA}+\mu\times E_{MCP}, \tag{4}$$

其中 μ 表示一个浮点数运算操作的平均能耗,$\lambda\mu$ 为一次访存操作的能耗. 根据文献[27],一个浮点数访存操作所消耗的能量是一个浮点数运算能耗的 15 倍,即 $\lambda=15$,下面的分析我们将沿用这一设定.

4.2 SVM 算法的负载分析

支持向量机(SVM)算法的基本原理是通过训练数据寻找一个满足分类要求的最优分类超平面,算法中这个超平面被表达为一组支持向量并作为分类的依据,针对一个已训练好的二分类支持向量机,样本的分类公式为

$$f(\boldsymbol{x})=\sum_{i=1}^{m}\alpha_i y_i\kappa(\boldsymbol{x},x_i)+b, \tag{5}$$

其中,$\boldsymbol{x}$ 代表输入样本向量,x_i 表示第 i 个支持向量,y_i 和 α_i 分别代表该支持向量对应的类别和拉格朗日乘子,支持向量的总数目为 m,b 代表偏移项,$\kappa(\cdot)$为核函数.

对多分类问题,常被采用的策略有一对一方式和一对多等方式. 本文采用一对一方式,对于一个 c 分类问题,该方式将产生 $c(c-1)/2$ 个上述二分类分类器,最终分类结果由这些分类器的投票结果产生.

SVM 算法的访存负载主要来自于对支持向量、拉格朗日乘子等必要的模型参数的访存操作,假设样本和支持向量的数据维度为 d,支持向量的数据量取决于数据维度 d 和向量数目 m,而拉格朗日乘子的数据量则正比于类别数 c 和向量数目 m,因此对单样本的 c 分类任务,SVM 的参数访存负载(即浮点数访存操作的数量)为

$$F_{MMA}^{SVM}=(d+c)\times m. \tag{6}$$

SVM 算法的计算负载主要分为 2 部分:1)核函数 κ 包含的计算操作,虽然核函数 κ 可以有很多种形式,但在大多数核函数中最频繁的计算操作是点乘操作(即乘法和加法操作),该部分的计算负载与样本维度 d 以及支持向量的数目 m 成正比;2)核函数结果与拉格朗日乘子的乘积操作,其计算量正比于 m,因此单样本 c 分类任务的计算负载可表示为

$$F_{MCP}^{SVM}=\sum_{i=1}^{c-1}\sum_{j=i+1}^{c}[2d(m_i+m_j)+(m_i+m_j)]\approx\sum_{i=1}^{c-1}\sum_{j=i+1}^{c}\left(2d\times\frac{2m}{c}+\frac{2m}{c}\right)=(2d+1)m(c-1). \tag{7}$$

4.3 RF 算法的负载分析

随机森林算法(RF)是一种集成学习方法,它以决策树为基学习器,通过多个基学习器的分类结果投票的方式决定最终的分类类别.

RF 算法的参数访存负载是所有决策树的节点上包含的参数的总和:

$$F_{MMA}^{RF}=\rho\times l\times t, \tag{8}$$

其中,ρ 是每个节点需存储的数据量,l 是每棵树的平均节点数,t 是 RF 算法包含的决策树的数目.

RF 算法的主要分类过程是决策树的遍历,从而得到每个基学习器的分类结果,主要的计算操作是样本属性值与节点值的比较操作,需要注意的是,因为分类时每层只需要比较一个节点,比较操作的数目正比于树的深度而不是树的节点数目,因此 RF 算法对单样本的计算负载为

$$F_{MCP}^{RF}=t\times\mathrm{lb}\,l. \tag{9}$$

4.4 NN 算法的负载分析

神经网络(NN)算法是模拟生物神经系统的结构,通过将多个神经元节点按照一定的层次结构连接起来的机器学习算法.

一个已训练好的 NN 算法模型的主要参数是神经元之间连接的权重和常数项,其数量与神经网络的层数相关. 因此,神经网络的参数访存负载表示为

$$F_{MMA}^{NN}=\sum_{i\in layers}w_i+b_i. \tag{10}$$

NN 算法的分类方式是前向传播算法,即从输入层开始逐层计算每层的神经元的数值,直至到达输出层,主要的计算操作是神经元节点值与连接权重的点乘操作,因此 NN 算法单样本的计算负载可表示为

$$F_{MCP}^{NN}=\sum_{i\in layers}s_i. \tag{11}$$

需要指出的是,由于多层感知机(MLP)和卷积神经网(CNN)结构上的差异,2 种神经网络的访存负载和计算负载有一定差异. 多层感知机为全互连

结构,而卷积神经网为权值共享的连接方式,因此在相同层数和节点数的情况下,卷积神经网的模型参数更少,访存负载更低,而在计算负载上,卷积神经网的点乘操作被限制在"感受野"范围内,因此计算负载也小于多层感知机.

5 算法参数访存能耗和计算能耗的评估结果

通过第 4 节分析,我们得到 3 类算法的参数访存负载和计算负载的估算公式.我们在 4 个不同的数据集上训练得到的算法参数如表 4 所示.

由表 4 的参数数据以及 4.1 节的分类能耗模型可以得到 3 类算法的分类能耗,如图 6 所示.图 6(a)表示分类能耗,图 6(b)(c)分别代表参数访存能耗和计算能耗,图 6(d)代表参数访存能耗和计算能耗的差异度.根据评估结果,我们针对参数访存能耗 E_{MMA} 和计算能耗 E_{MCP} 可以得出 3 点结论,详见 5.1～5.3 节所述.

Table 4 The Value of Model Parameters

表 4 3 种算法参数设置

Algorithm	HAR	Iris	MNIST	Adult
SVM	$d=561$ $m=3\,231$ $c=6$	$d=4$ $m=37$ $c=3$	$d=784$ $m=13\,884$ $c=10$	$d=14$ $m=12\,297$ $c=2$
RF	$\rho=5$ $l=465$ $t=20$	$\rho=5$ $l=13$ $t=1$	$\rho=5$ $l=9\,829$ $t=200$	$\rho=5$ $l=6\,679$ $t=20$
NN*	561-5-6 (MLP)	4-2-3 (MLP)	LeNet (CNN)	14-5-2 (MLP)

*: As for NN algorithm, we use the classical LeNet network to classify MNIST dataset and use MLP network to other datasets. The numeral value of MLP represents the number of nodes in each layer.

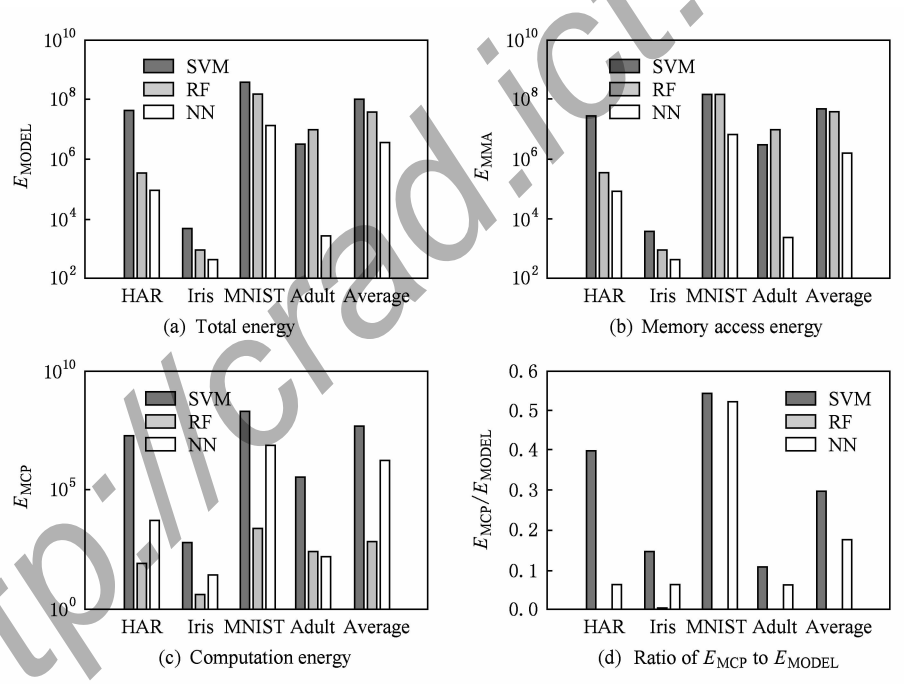

Fig. 6 Comparison on classification energy of different algorithms

图 6 3 种算法分类能耗比较

5.1 E_{MMA} 和 E_{MCP} 的算法可近似性分析

结论 1. 针对参数访存能耗 E_{MMA} 和计算能耗 E_{MCP},算法可近似性的差异对能耗优化的影响有限,算法的选择主要决定于算法初始能耗的差别.从图 6(a)中可以看出,不同算法的初始分类能耗差异很大,SVM 算法由于参数访存负载和计算负载均正比于支持向量的数目,而在复杂应用中这一数目往往很大,因此分类能耗最高;RF 算法由于节点数量多,参数数据量大,访存能耗很高,几乎与 SVM 算法的访存能耗相当,分类能耗略低于 SVM 算法;而 NN 算法由于其参数数量和神经网络层数均较少,分类能耗在 3 类算法中最低.对比平均分裂能耗,SVM 算法平均分类能耗约为 NN 算法分类能耗的 29 倍,RF 算法的平均分类能耗是 NN 算法的 12 倍.因此这样初始能耗的巨大差异,使得算法可近似性的差异对能耗的影响十分有限.

5.2 E_{MMA}和E_{MCP}的算法能耗结构分析

结论 2. 不同算法的参数访存能耗和计算能耗的比例有明显的差异.通过图 6(b)和图 6(c)可以发现,SVM 算法的参数访存能耗和计算能耗大小处于同一数量级,其主要原因在于 SVM 算法的参数访存负载和计算负载均与支持向量的数目成正比,因此 2 类能耗的差异较小;RF 算法参数访存能耗很高但计算能耗却很低,其主要原因在于 RF 算法的节点数量很大,因此需要访存的参数很多,而计算负载与节点的对数成正比并且计算操作简单,因此计算能耗远远低于参数访存能耗;NN 算法参数访存能耗和计算能耗正比于网络层数,平均访存能耗约为计算能耗的 12 倍.

为了更好地表征算法在能耗结构上的差异性,我们将参数访存能耗与分类能耗的比值关系定义为算法的能耗结构差异度,简称能耗差异度(energy diversity),即

$$Energy\ Diversity=\frac{E_{MCP}}{E_{MODEL}}. \tag{12}$$

3 类算法能耗差异度的对比如图 6(d)所示,RF 算法由于计算能耗远远低于参数访存能耗,能耗差异度接近为 0,平均能耗差异度为 0.001;SVM 算法的平均能耗差异度最高,约为 0.3;NN 算法介于二者之间,能耗差异度为 0.17.

5.3 E_{MMA}和E_{MCP}的算法能耗差异度影响

结论 3. 我们将问题扩展到多样本分类情况,算法能耗差异度的差别可以帮助多样本分类过程选择能耗最优的算法.多样本分类能耗的计算需要考虑到分类应用的样本处理模式,我们将样本处理模式分为 2 种:1)批量处理模式,该模式对应在专用的机器学习加速器或大型的数据中心上的分类应用,这种模式下样本数据往往被批量处理,算法模型参数只需被访存一次,因此算法的参数访存能耗不受样本数量的影响,而计算能耗由于样本数量的增加而成为分类能耗的主要部分;2)非批量处理模式,对应于多任务平台如手机等移动设备的分类应用,由于其他任务对资源的抢占,该模式下的样本数据往往无法批量处理样本数据,算法参数由于资源受限而被频繁地换入换出,参数访存能耗和计算能耗均随样本数量上升而增加,此时参数访存能耗占分类能耗的主要部分.因此在分类多样本时,应用应该根据算法能耗差异度和样本处理模式选择合适的算法,具体地,在批量数据处理时,应选择算法能耗差异度低的算法如 RF 算法,这样可以充分利用这类算法计算能耗低的优势,而在非批量处理模式下,能耗差异度高的算法如 NN 算法等将有助于减少参数访存能耗带来的开销,从而降低整体分类能耗.

6 总 结

算法"可近似性"的差异对近似计算优化能耗问题有着重要的影响,本文评估了 3 种机器学习算法 SVM 算法、RF 算法和 NN 算法在优化样本存储能耗、样本访存能耗、参数访存能耗和计算能耗的可近似性差异,并提出了存储污染敏感度、访存污染敏感度和能耗差异度等评估指标以表征可近似性的差距.这些评估指标将有助于分类应用选择合适的机器学习算法实现能效最优.

评估结果表明:算法可近似性的差异确实对能耗优化具有重要的作用,并且针对不同类型的能耗,算法的可近似性并不固定.SVM 算法对比特位截断技术造成的数据模糊容忍能力更高,因此针对访存能耗的可近似性更好,访存污染敏感度最低,适合优化样本访存能耗时被选用;RF 算法对降低刷新率带来的样本突变噪声更加鲁棒,存储污染敏感度更低,适合优化样本存储能耗时被采用;而对于参数访存能耗和计算能耗,由于算法初始能耗的差异很大,可近似性的差异对能耗优化的影响将非常有限,经验性的结果表明针对单样本分类,NN 算法的参数访存能耗和计算能耗最低.此外,算法能耗结构上的差异也十分显著,有助于多样本分类过程的算法选择,具体地,RF 算法等能耗差异度低的算法适合数据批量处理模式来尽量降低计算能耗,而非批量处理模式为减少参数访存能耗更适合选择 NN 算法等能耗差异度高的算法.

参 考 文 献

[1] Chippa V K, Mohapatra D, Raghunathan A, et al. Scalable effort hardware design: Exploiting algorithmic resilience for energy efficiency [C] //Proc of the 47th ACM/IEEE Design Automation Conf (DAC 2010). Piscataway, NJ: IEEE, 2010: 555-560

[2] Venkataramani S, Ranjan A, Roy K, et al. AxNN: Energy-efficient neuromorphic systems using approximate computing [C] //Proc of the 2014 Int Symp on Low Power Electronics and Design. New York: ACM, 2014: 27-32

[3] Shafique M, Hafiz R, Rehman S, et al. Invited: Cross-layer approximate computing: From logic to architectures [C] // Proc of the 53rd ACM/EDAC/IEEE Design Automation Conf (DAC 2016). Piscataway, NJ: IEEE, 2016: 1-6

[4] Xu Qiang, Mytkowicz T, Kim N S. Approximate computing: A survey [J]. IEEE Design & Test, 2016, 33 (1): 8-22

[5] Moons B, Verhelst M. Dvas: Dynamic voltage accuracy scaling for increased energy-efficiency in approximate computing [C] //Proc of 2015 IEEE/ACM Int Symp on Low Power Electronics and Design (ISLPED). Piscataway, NJ: IEEE, 2015: 237-242

[6] Raha A, Venkataramani S, Raghunathan V, et al. Quality configurable reduce-and-rank for energy efficient approximate computing [C] //Proc of the 2015 Design, Automation & Test in Europe Conf & Exhibition (DATE 2015). Piscataway, NJ: IEEE, 2015: 665-670

[7] Wunderlich H J, Braun C, Schöll A. Pushing the limits: How fault tolerance extends the scope of approximate computing [C] //Proc of the 22nd IEEE Int Symp on On-Line Testing and Robust System Design (IOLTS 2016). Piscataway, NJ: IEEE, 2016: 133-136

[8] Mazahir S, Hasan O, Hafiz R, et al. An area-efficient consolidated configurable error correction for approximate hardware accelerators [C] //Proc of the 53rd ACM/EDAC/IEEE Design Automation Conf (DAC 2016). Piscataway, NJ: IEEE, 2016: 1-6

[9] Shim Y, Sengupta A, Roy K. Low-power approximate convolution computing unit with domain-wall motion based "Spin-Memristor" for image processing applications [C] // Proc of the 53rd ACM/EDAC/IEEE Design Automation Conf (DAC 2016). Piscataway, NJ: IEEE, 2016: 1-6

[10] Sarbishei O, Radecka K. Analysis of precision for scaling the intermediate variables in fixed-point arithmetic circuits [C] // Proc of the Int Conf on Computer-Aided Design. Piscataway, NJ: IEEE, 2010: 739-745

[11] Samadi M, Lee J, Jamshidi D A, et al. Sage: Self-tuning approximation for graphics engines [C] //Proc of the 46th Annual IEEE/ACM Int Symp on Microarchitecture. New York: ACM, 2013: 13-24

[12] Esmaeilzadeh H, Sampson A, Ceze L, et al. Neural acceleration for general-purpose approximate programs [C] // Proc of the 45th Annual IEEE/ACM Int Symp on Microarchitecture. Los Alamitos, CA: IEEE Computer Society, 2012: 449-460

[13] Liu J, Jaiyen B, Veras R, et al. RAIDR: Retention-aware intelligent DRAM refresh [C] //Proc of ACM SIGARCH Computer Architecture News. Los Alamitos, CA: IEEE Computer Society, 2012, 40(3): 1-12

[14] Liu S, Pattabiraman K, Moscibroda T, et al. Flikker: Saving DRAM refresh-power through critical data partitioning [C] //Proc of the 16th Int Conf on Architectural Support for Programming Languages and Operating Systems (ASPLOS XVI). New York: ACM, 2011: 213-224

[15] Sampson A, Dietl W, Fortuna E, et al. EnerJ: Approximate data types for safe and general low-power computation [C] // Proc of ACM SIGPLAN Notices. New York: ACM, 2011: 164-174

[16] Filippou F, Keramidas G, Mavropoulos M, et al. Recovery of performance degradation in defective branch target buffers [C] //Proc of the 22nd IEEE Int Symp on On-Line Testing and Robust System Design (IOLTS 2016). Piscataway, NJ: IEEE, 2016: 96-102

[17] Mohapatra D, Chippa V K, Raghunathan A, et al. Design of voltage-scalable meta-functions for approximate computing [C] //Proc of the 2011 Design, Automation & Test in Europe Conf & Exhibition (DATE 2011). Piscataway, NJ: IEEE, 2011: 1-6

[18] Tian Ye, Zhang Qian, Wang Ting, et al. ApproxMA: Approximate memory access for dynamic precision scaling [C] //Proc of the 25th Edition on Great Lakes Symp on VLSI. New York: ACM, 2015: 337-342

[19] Zhang Qian, Wang Ting, Tian Ye, et al. ApproxANN: An approximate computing framework for artificial neural network [C] //Proc of the 2015 Design, Automation & Test in Europe Conf & Exhibition. Piscataway, NJ: IEEE, 2015: 701-706

[20] Sathish V, Schulte M J, Kim N S. Lossless and lossy memory I/O link compression for improving performance of GPGPU workloads [C] //Proc of the 21st Int Conf on Parallel Architectures and Compilation Techniques. New York: ACM, 2012: 325-334

[21] He Xin, Lu Wenyan, Yan Guihai, et al. Exploiting the potential of computation reuse through approximate computing [J]. IEEE Trans on Multi-Scale Computing Systems, 2016, DOI: 10.1109/TMSCS.2016.2617343

[22] He Xin, Yan Guihai, Han Yinhe, et al. ACR: Enabling computation reuse for approximate computing [C] //Proc of the 21st Asia and South Pacific Design Automation Conf (ASP-DAC 2016). Piscataway, NJ: IEEE, 2016: 643-648

[23] Alvarez C, Corbal J, Valero M. Fuzzy memoization for floating-point multimedia applications [J]. IEEE Trans on Computers, 2005, 54(7): 922-927

[24] Kahng A B, Kang S. Accuracy-configurable adder for approximate arithmetic designs [C] //Proc of the 49th Annual Design Automation Conf. New York: ACM, 2012: 820-825

[25] Fang Shuangde, Du Zidong, Fang Yuntan, et al. Run-time technology for low-power oriented inexact heterogeneous multi-core [J]. Chinese High Technology Letters, 2014, 24 (8): 791-799 (in Chinese)
(房双德, 杜子东, 方运潭, 等. 面向低能耗的非精确异构多核上的运行时技术[J]. 高技术通讯, 2014 (8): 791-799)

[26] Venkataramani S, Raghunathan A, Liu Jie, et al. Scalable-effort classifiers for energy-efficient machine learning [C] // Proc of the 52nd Annual Design Automation Conf (DAC'15). New York: ACM, 2015: 61-67

[27] Düben P, Schlachter J, Parishkrati, et al. Opportunities for energy efficient computing: A study of inexact general purpose processors for high-performance and big-data applications [C] //Proc of the 2015 Design, Automation & Test in Europe Conf & Exhibition (DATE'15). Piscataway, NJ: IEEE, 2015: 764-769

[28] LeCun Y, Jackel L D, Bottou L, et al. Comparison of learning algorithms for handwritten digit recognition [C] // Proc of Int Conf on Artificial Neural Networks. Paris, France: EC2 & Cie, 1995: 53-60

[29] Arumugam G P, Srikanthan P, Augustine J, et al. Novel inexact memory aware algorithm co-design for energy efficient computation: Algorithmic principles [C] //Proc of the 2015 Design, Automation & Test in Europe Conf & Exhibition (DATE 2015). Piscataway, NJ: IEEE, 2015: 752-757

Jiang Shuhao, born in 1993. PhD, candidate. Student member of IEEE. His main research interests include machine learning and approximate computing.

Yan Guihai, born in 1981. PhD, professor. Member of IEEE, ACM, and CCF. His main research interests focus on energy-efficient and resilient architectures.

Li Jiajun, born in 1990. PhD candidate. Student member of IEEE/CCF. His main research interests include machine learning and heterogeneous computer architecture.

Lu Wenyan, born in 1988. PhD candidate. Student member of IEEE/CCF. His main research interests include heterogeneous computing and deep learning accelerator.

Li Xiaowei, born in 1964. PhD, professor, PhD supervisor. Senior member of IEEE. His main research interests include VLSI testing and design verification, dependable computing, wireless sensor networks.