计算机研究与发展
Journal of Computer Research and Development
DOI:10.7544/issn1000-1239.2016.20150147
53(6):1202-1210,2016

DLPF:基于异构体系结构的并行深度学习编程框架

王岳青[1] 窦 勇[1] 吕 启[1] 李宝峰[2] 李 腾[1]
[1](国防科学技术大学并行与分布处理国防重点实验室 长沙 410073)
[2](国防科学技术大学计算机学院 长沙 410073)
(yqwang2013@163.com)

DLPF: A Parallel Deep Learning Programming Framework Based on Heterogeneous Architecture

Wang Yueqing[1], Dou Yong[1], Lü Qi[1], Li Baofeng[2], and Li Teng[1]
[1](*Science and Technology on Parallel and Distributed Processing Laboratory, National University of Defense Technology, Changsha* 410073)
[2](*College of Computer, National University of Defense Technology, Changsha* 410073)

Abstract Deep learning plays an important role in machine learning field, and it has been widely used in various applications. The prospect of research and applications of deep learning are huge. However, deep learning also faces several challenges. Firstly, there are many tools in deep learning field, but these tools are not convenient to use for non-expert users because the installation and usage of them are really complex. Secondly, the diversity of deep learning is limited because the flexibility of existing deep learning models is not enough. Furthermore, the training time of deep learning is so long that the optimal hyper-parameters combination cannot be found in a short time. To solve these problems, we design a deep learning programming framework based on heterogeneous architecture in this paper. The programming framework establishes a unified module library which can be used to build a deep model through the visual interface conveniently. Besides, the framework also accelerates the basic modules on heterogeneous platform, and makes the speed of searching optimal hyper-parameters combination be faster. Experimental results show that the programming framework can construct deep models flexibly, and more importantly, it can achieve comparative classification results and better timing performance for a variety of applications. In addition, the framework can search optimal hyper-parameters efficiently and make us infer the relationship of all hyper-parameters.

Key words deep learning; programming framework; visualization; heterogeneous architecture; accelerate

摘 要 深度学习在机器学习领域扮演着十分重要的角色,已被广泛应用于各种领域,具有十分巨大的研究和应用前景.然而,深度学习也面临3方面的挑战:1)现有深度学习工具使用便捷性不高,尽管深度学习领域工具越来越多,然而大多使用过程过于繁杂,不便使用;2)深度学习模型灵活性不高,限制了深度学习模型发展的多样性;3)深度学习训练时间较长,超参数搜索空间大,从而导致超参数寻优比较困难.针对这些挑战,设计了一种基于深度学习的并行编程框架,该框架设计了统一的模块库,能可视化地进行深度学习模型构建,提高了编程便捷性;同时在异构平台对算法模块进行加速优化,较大程度减少

收稿日期:2015-02-15;**修回日期**:2015-06-23
基金项目:国家自然科学基金项目(61125201,U1435219)
This work was supported by the National Natural Science Foundation of China (61125201,U1435219).

训练时间,进而提高超参数寻优效率.实验结果表明,该编程框架可以灵活构建多种模型,并且对多种应用取得了较高的分类精度.通过超参数寻优实验,可以便捷地获得最优超参数组合,从而推断各种超参数与不同应用的联系.

关键词 深度学习;编程框架;可视化;异构平台;加速

中图法分类号 TP183

信息时代,大数据已经渗透到众多行业和业务领域,成为重要的生产因素,但其原始数据往往量大而价值密度低,传统的数据分析技术已经无法有效挖掘数据的功能和价值.因此,必须采取更有效的方法提取大数据的主要特征和内在关联.2006 年,Hinton 和 Salakhutdinov[1] 在国际顶尖学术期刊 Science 发表文章,提出利用深度信念网络(deep belief network, DBN)实现大数据的降维及分析,引发了深度学习的浪潮.

典型的深度学习模型结构如图 1 所示,由多层构成,每层通过编码产生隐层输出,通过解码将隐层输出重构输入数据,并依据重构数据与原始数据的差异更新输入层与隐含层的连接权值.这种层次化的训练过程是模仿大脑的层次信息处理机制,组合低层特征形成高层特征从而发现数据的分布式特征表示,进而解释图像、声音和文本等数据的潜在关联.

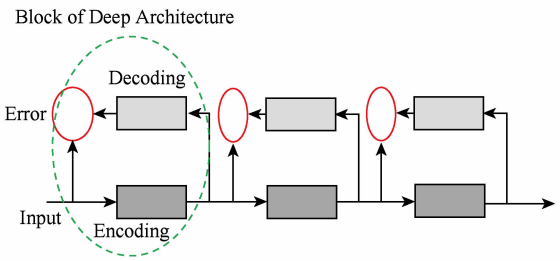


Fig. 1 Structure of deep learning model.

图 1 深度学习模型结构

与浅层神经网络相比,深度学习可以有效防止网络规模剧烈增加.总之,深度学习已经发展为机器学习领域一个十分重要的分支,具有十分大的研究潜力和应用价值,已成为各个领域智能化信息处理的热点.

深度学习的研究充满了机遇,但同时面临着诸多挑战.

1) 深度学习工具使用便捷性不高

深度学习的不断发展必然促进开源工具的增加,然而大多数开源工具的使用需要一系列复杂配置,容易出错;同时,深度学习算法对硬件平台和软件环境要求较高,很多研究者并不具备相应的研究条件;此外,已有的深度学习工具种类繁多,不易选择.以上因素会给深度学习的使用带来诸多不便,无疑会限制深度学习在新领域的推广和使用.

2) 深度学习模型灵活性不高

大脑分层进行认知和学习,这是深度学习思想的来源.但大脑各个区域的结构和功能并不相同,因而很可能在学习过程的各个阶段使用的是不同的学习算法.为了更好地模拟大脑的学习过程,需要在深度学习的不同层次使用不同算法,这要求深度学习模型具有很高的灵活性,能够自由组合各种算法模块.深度学习包括具有不同结构和功能的算法模块,图 2 给出了基本的算法模块,如稀疏编码(SC)、自动编码机(AE)、受限玻尔兹曼机(RBM)、深度信念机(DBM)、递归神经网(RNN)、高斯混合模型(GMM)和卷积神经网络(CNN)等.然而,现有的深度学习

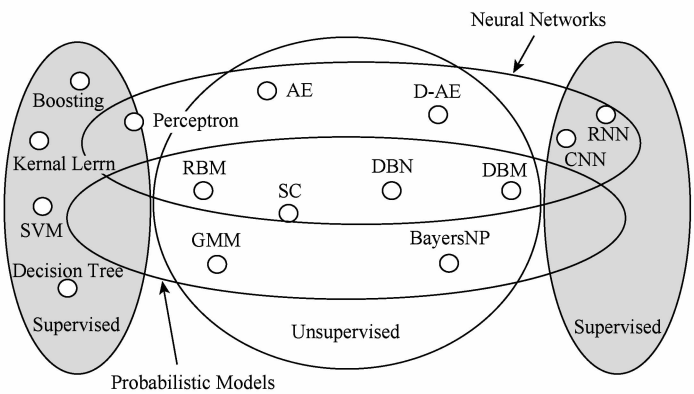


Fig. 2 The module library of deep learning.

图 2 深度学习的算法模块库

模型的各层均由相同的算法模块构成,这限制了模型的灵活性和多样性,对深度学习模型研究带来了不便.

3) 深度学习中超参数寻优比较困难

深度学习算法在训练过程中主要计算网络中层与层之间连接的强度参数.此外,深度学习算法训练得到的特征好坏还与超参数相关.所谓超参数,是指需要人为调整的参数,例如学习率(反映了调整权值参数的快慢程度)、隐层节点数、动量参数等,这些参数对最终的特征提取有着至关重要的作用.目前在深度学习领域并没有完善的理论指导求解最优超参数组合,只能凭借经验配置超参数,但大多数人并没有相关经验,因此更常用的是网格搜索(grid search)法,即将超参数的各种组合穷举并依据交叉验证的平均准确率得到最优组合.由于超参数数量和可能取值较多,从而导致搜索空间大、搜索耗时长.这对深度学习的模型选择带来了极高的难度.

综上所述,深度学习研究已经成为热点,但同样面临着诸多挑战.本文结合深度学习算法的特点,使用可视化界面简化深度学习模型的构建过程,通过统一深度学习算法模块的接口增加模型灵活性,通过自动搜索最优超参数减小超参数配置难度,在此基础之上研究深度学习并行编程框架.

1 相关工作

目前,深度学习领域已有诸多较为成熟的开源工具和平台.

2012 年,基于 C++/CUDA 的深度学习代码包 cuda-convnet[2] 发布,它主要采用反向传播算法的深度卷积神经网络[3]实现,基于其训练的深度卷积网模型在 ImageNet 数据集中取得了 Top5 的成绩[4].Caffe 是由 Jia[5]开发的深度学习架构,训练速度快,使用单块 NVIDIA K40 训练深度卷积网络(CNN)每天可以处理超过 4 000 万幅图像,使用 Titan GPU 训练时速度达到 5 000 fps. Theano[6] 提供了在深度学习数学计算方面的 Python 库,整合了 NumPy 矩阵计算库,具有良好的算法扩展性,同时也提供 GPU 支持. Kaldi[7] 是一个基于 C++ 和 CUDA 的语音识别工具集,提供给语音识别的研究人员使用. MLPACK[8] 和 Torch7[9] 是为机器学习算法提供广泛支持的科学计算框架.

上面介绍的开源工具尽管应用广泛,训练速度较快;但使用便捷性不高,配置较为复杂,对机器配置要求较高,使用过程容易出错,同时不支持具有混合算法模块的深度学习模型,灵活性不高,此外,大多数工具并不能自动搜索最优超参数组合.

深度学习除在学术界有较大发展之外,还被广泛应用于工业界,各大公司都成立了深度学习研究小组,也实现了自己的软件框架.

Google 的 DistBelief [10] 系统是 CPU 集群实现的数据并行和模型并行框架,集群内使用上万个 CPU 核来训练参数数目多达 10 亿的深度神经网络模型. Facebook 实现了多 GPU 训练深度卷积神经网络的并行框架[11],结合数据并行和模型并行的方式来训练 CNN 模型.百度搭建了 PADDLE(parallel asynchronous distributed deep learning)[12] 多机 GPU 训练平台,将数据分布到不同机器,通过参数服务器协调各机器训练,该架构支持数据并行和模型并行.腾讯的深度学习平台(Mariana)[13] 是为加速深度学习模型训练而开发的并行化平台,包括深度神经网络的多 GPU 数据并行框架、深度卷积神经网络的多 GPU 模型并行和数据并行框架,以及深度神经网络的 CPU 集群框架.

工业界的深度学习框架面向特定的应用领域能够达到较好的效果,但后果就是其模型结构特别复杂,无法兼顾使用的便捷性和模型的灵活性.

本文提出一种基于异构平台的深度学习编程架构,将深度学习模型构建可视化,使得深度学习编程更为简单,从而使得更多应用领域的学者方便地使用深度学习模型进行研究;同时设计统一接口模式,使得模型构建更加灵活,增加模型的多样性.此外,我们的编程框架可以并行搜索最优超参数组合,代替手动调参的繁琐过程.

2 深度学习并行编程框架

深度学习并行编程框架包括 4 层:Web 层、调度层、计算层和加速层. Web 层面向用户,提供可视化编程界面;调度层动态调度计算资源和计算任务;计算层接受不同的应用数据训练和测试模型;加速层使用协处理器加速训练和测试过程.如图 3 所示,各层之间通过数据传递紧密耦合,形成完整架构.

2.1 Web 层

Web 层目标是提供简洁的可视化编程界面,提高深度学习使用的便捷性.该层由 3 个界面组成:模型管理界面、数据管理界面和任务管理界面.模型管理界面将深度学习算法以模块形式显示,支持通过添加算法模块动态创建深度学习模型.

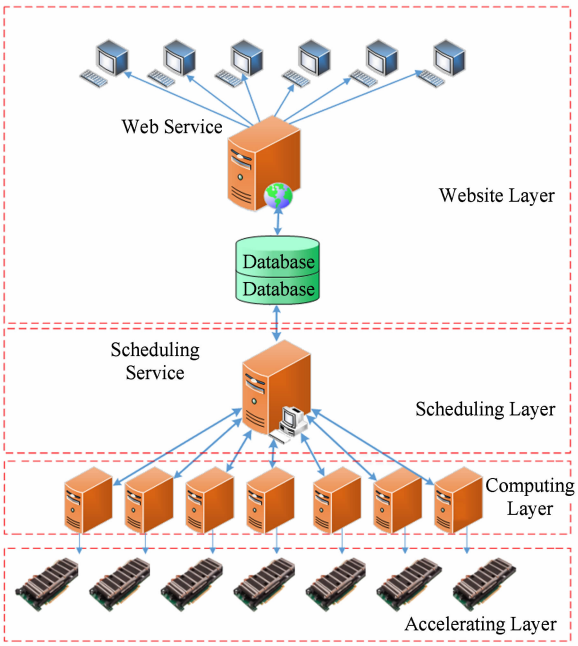


Fig. 3 Hardware architecture of DLPF.

图 3 深度学习编程架构硬件映射示意图

图 4 是 Web 层的可视化模型构建界面,左侧部分提供多种算法模块,右侧是根据左侧算法模块创建的深度信念网(DBN)模型. 在此界面中,用户无需编程,只需要点击相应算法模块并加入到模型中即可,操作简单,无需任何平台和软件配置.

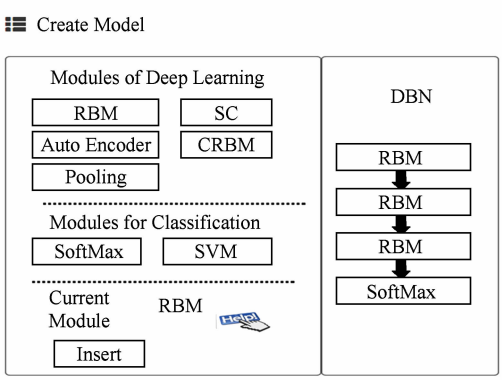


Fig. 4 Visual programming interface.

图 4 可视化模型构建界面

模型构建完成后,用户需要点击模型的各个模块进行超参数设置和选择,超参数设置有 3 种选项,分别对应设置特定值、设置搜索范围和步长以及自动搜索. 其中,设置特定值针对具备调参经验的用户;而后 2 种选项主要针对其他用户,采用后台并行搜索的方法搜索最优组合. 用户使用该界面构建好模型之后,进入数据管理界面. 数据管理界面提供数据上传和数据管理服务,用户可以选择已有的数据集或者上传新的训练和测试数据,如果数据集过大,也可以直接提供网址以供后台自动下载数据集. 当模型和数据都准备完成后,用户可以使用任务管理界面为模型分配数据从而创建计算任务,此外,该界面还提供任务管理功能,用户可以方便地新建、暂停、停止和删除已有任务. 当计算任务完成后,DLPF 会将结果发送到 Web 服务器和用户的注册邮箱,并在任务管理界面显示.

2.2 调度层

调度层目标是为空闲节点动态分配计算任务. 它接受 Web 层传递的任务,通过任务解析器将任务转换为模型文件,并将模型文件放置在任务池中等待调度,当有空闲节点时,从任务池中选择优先级较高的任务进行计算. 调度过程必须保证公平性和响应时间,而深度学习训练过程复杂、所需时间较长,如果任务耗时较长,将会急剧增加其他任务的响应时间,因此采用时间片轮转算法进行任务调度,这可以保证每个用户都能公平地使用资源,同时减少响应时间. 通过时间片轮转算法调度之后,各个计算节点能够尽量均衡地接受任务,从而展开进一步计算.

2.3 计算层

计算层包括多个物理节点或者虚拟节点,目标是根据已有数据完成模型参数训练. 当调度层将任务分配给某个计算节点时,计算节点首先从相应存储设备读取数据,之后根据调度层传递的模型文件,通过已有深度学习模块库构建深度学习模型. 每个深度学习模型对应一个代价函数,而目标是调整网络参数使代价函数最小,当代价函数最小时模型达到最优. 寻找最小代价函数的过程即为优化过程,可以采用多种优化算法. 较为常用的优化算法包括对比散度算法(CD-K)、反向传播算法(BP)、共轭梯度法(CG)和随机梯度下降算法(SGD)等. 算法 1 是计算层 RBM 模块采用的 CD-K 算法. 在算法 1 中,首先根据已有数据 v 计算隐层节点输出 h,然后根据 h 重构输入数据 v',再由 v' 计算出重构数据的隐层输出 h',最后根据这 4 个变量更新权值和偏置参数.

算法 1. K 步对比散度算法(CD-K).

输入:RBM $[v_1,v_2,\cdots,v_n,h_1,h_2,\cdots,h_m]$,S;

输出:RBM $[\Delta w_{ij},\Delta b_j,\Delta c_i]$.

① 初始化 $\Delta w_{ij}=0,\Delta b_j=0,\Delta c_i=0$;

② $v^{(0)}\leftarrow S$;

③ for $t=0,1,\cdots,K-1$ do

④ for $i=1,2,\cdots,n$ do

⑤ 采样 $h_i^{(t)}\sim p(h_i|v^{(t)})$;

⑥ end for

⑦ for $j=1,2,\cdots,m$ do
⑧ 采样 $v_j^{(t)}\sim p(v_j|h^{(t)})$;
⑨ end for
⑩ for $i=1,2,\cdots,n$, $j=1,2,\cdots,m$ do
⑪ $\Delta w_{ij}\leftarrow\Delta w_{ij}+h_i^{(0)}v_j^{(0)}-h_i^{(K)}v_j^{(K)}$;
⑫ end for
⑬ for $i=1,2,\cdots,n$ do
⑭ $\Delta c_i\leftarrow\Delta c_i+h_i^{(0)}-h_i^{(K)}$;
⑮ end for
⑯ for $j=1,2,\cdots,m$ do
⑰ $\Delta b_j\leftarrow\Delta b_j+v_j^{(0)}-v_j^{(K)}$;
⑱ end for
⑲ end for

所有优化算法在寻优过程中需要不断计算代价函数值来判断是否找到最优解,而代价函数的计算过程以矩阵计算为主,当数据量较大时,这一过程成为瓶颈,为了减少矩阵运算时间,DLPF 选择一些速度较快的开源软件库用于矩阵运算. 例如 Armadillo 数学库[14],其底层调用 openBLAS[15],NVBLAS[16],LAPACK[17]等库函数对矩阵运算进行优化加速. 此外,由于随机梯度下降算法(SGD)可以用于除 RBM 和 CRBM 之外的其他模块,而 RBM 和 CRBM 都使用 CD-K 算法优化,因此我们选择了 RBM 模块和 SGD 算法进行 GPU 加速. 图 5 是 DLPF 框架的软件架构,核心在于中间的模块库(module library),模块库目前已经实现了 RBM、Auto-Encoder、sparse coding、极限学习机(ELM)[18]、卷积 RBM、卷积神经网络(CNN)以及其他一些变形算法,与核心模块库紧耦合的部分包括矩阵运算库和优化算法库.

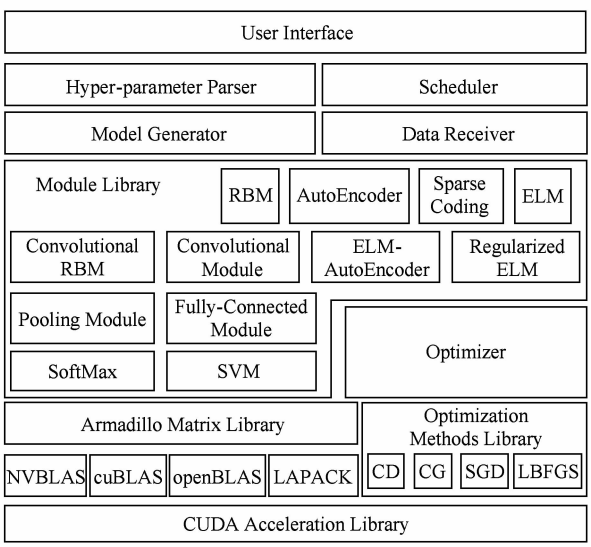


Fig. 5 Software architecture of DLPF.

图 5 编程框架软件架构示意图

2.4 加速层

除在计算层中调用优化数学库进行程序加速之外,我们还专门设计了高效的 GPU 并行算法库,使其在运算过程中可被灵活使用. 下面简单介绍加速层采用的并行算法.

深度学习一般采取分层的网络结构,采用逐层贪心算法进行训练. 根据层间的连接方式,可以将单层网络分为局部连接(如卷积神经网)与全连接(如深度信念网络)2 种: 1)对于全连接网络,设计增强局部并行性的训练方法,如图 6 所示. 单层网络分为多个局部计算单元,单元间计算并行,并通过通信方式传递跨单元计算信息. 在此条件下,每个单元存储整个单层网络的权值副本,并计算对其他单元中隐含层节点的贡献,每个隐含层节点接收各个单元的结果后规约更新. 2)对于局部连接的卷积网络,可以对其不同的卷积核展开并行学习,对于相同的卷积核,也可以利用卷积特性进行细粒度并行.

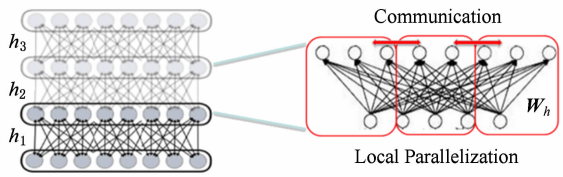


Fig. 6 Parallel algorithm for fully-connected network.

图 6 全连接网络的并行算法

以 RBM 算法模块的 GPU 加速为例,图 7 给出了算法在 GPU 的 block 和 thread 层面的并行策略. 每个样本使用 1 个 GPU block 计算,而每个样本需要计算该样本与所有隐层节点的连接权值,这些连接权值的计算可以完全并行,使用 1 个线程计算 1 个连接权值. 这种分配策略能够更好地利用 GPU 的并行资源,也能更大地挖掘算法的并行性.

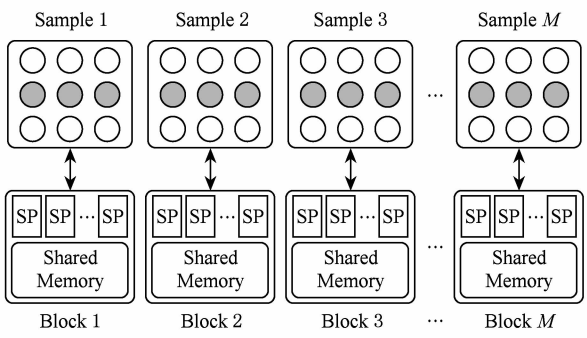


Fig. 7 Block and thread parallel strategies of RBM on GPU.

图 7 RBM 在 GPU 的 block 和 thread 层面的并行策略

统一的编程架构能够将编程和使用简单化,可以方便用户使用和改进机器学习算法.我们的工作是研究各类深度算法处理结构的共性及特异性,根据提取的算法要素提炼出表达简洁的算法结构,目标是使用户通过高效的输入界面、配置文件或者命令语言就能够准确描述所需要的深度学习算法,之后将算法结构翻译成为算法模块,并通过构建数据链紧密耦合各个模块,从而形成完整计算模型.

2.5 编程框架特点

1) 可视化编程界面

Web层提供简洁的可视化编程界面,将深度学习各种算法模块以图形方式展现,用户可以通过添加模块的形式轻松构建深度学习模型.图8给出了使用Web界面构建的2个深度学习模型结构的例子,包括有监督的CNN模型和无监督的DBN模型,可视化界面可以清晰地表达模型结构和层间连接关系.

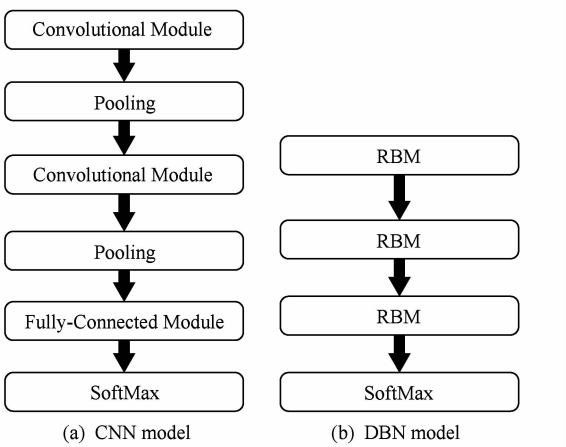


Fig. 8 CNN and DBN models made by the Web layer.

图8 DBN和CNN模型构建示意图

2) 多模型和多应用支持

DLPF编程框架支持多种应用,目前已经测试过的应用包括手写字符识别、遥感图像分类、三维物体分类和自然图像分类,而且DLPF框架在这些应用上都取得了较好的结果.同时,由于算法模块的灵活性和统一的接口,DLPF框架还支持构建不同模型,除传统的深度学习模型之外,还支持构建混合算法模型,例如RBM-AE模型:第1层使用RBM模块,第2层使用自动编码机模块.混合模型的构建是有实际意义的,在引言提到过,大脑在工作时信息会在不同区域接受不同的处理,传统的深度学习模型在各层使用同样的算法(DBN)或者每几层使用类似的算法(CNN),这实际是对大脑不同的区域使用相似的算法模拟,与实际情况并不符合.如果可以构建混合模型,应该能够更好地模拟大脑工作原理,从而进一步加深人对大脑工作原理的理解.

3) 超参数并行搜索

DLPF框架提供超参数并行搜索功能.用户在超参数设置时可以指定多个超参数值或者指定超参数范围和步长,后台程序会启动多个主机运行不同的超参数组合,进行网格搜索,通过交叉验证的方式判断最优超参数组合并将最优组合和模型返回给用户.一般模型的超参数个数较多,因此构成的搜索空间巨大,搜索较为耗时,我们采用粗粒度并行方式搜索最佳超参数组合.

3 实验结果与分析

实验平台由4台服务器和1台PC组成,PC机处理器型号为4核Intel i7-4790K,主频4 GHz,用于做加速对比实验.4台服务器中1台作为Web服务器,1台作为调度服务器,2台作为计算节点,每台有2个8核处理器,处理器型号为Intel Xeon E5-2650,主频为2 GHz,内存为64 GB.每台计算节点包括1块NVIDIA K40 GPU,核心频率为0.745 GHz,共有2 880个核.在多模型实验、多应用实验和时间性能测试过程中我们使用1台计算节点完成,超参数并行搜索实验采用2个计算节点完成.

3.1 多模型支持

DLPF可以灵活构建多种模型,包括RBM,Auto-Encoder,Sparse Coding,Extreme Learning Machine (ELM),ELM-AutoEncoder等简单模型,也包括DBN,Stacked-AutoEncoder,Stacked-ELM-AE,CNN,Convolutional-ELM(卷积ELM)等深层结构模型,还包括混合算法模型,例如RBM-AE,SC-RBM,CRBM-RBM等.我们在表1给出各个模型针对MNIST手写字符识别数据库的最终预测准确率.除ELM相关模型之外的所有模型使用SoftMax模块进行分类.

通过实验可以得到3点结论:1)使用深度模型后准确率会增加,但训练时间也会有所增加;2)不同的模型预测的准确率不相同,这不仅跟算法有关,也跟初始化得到的参数矩阵相关;3)混合模型也能够得到较好的性能,这是因为在无监督网络中,每层实际是独立的,层与层之间通过数据耦合,各层接受输入数据并抽取数据特征,不需关注这些数据的来源.

Table 1 Testing Accuracy of Different Models on MNIST Dataset

表 1 MNIST 手写字符识别在各种不同模型的测试精度

Model Name	Model Structure	Testing Accuracy/%
RBM	784-400	97.55
AutoEncoder	784-400	97.64
CRBM	784-5_12	96.18
Sparse Coding	784-400	92.40
ELM	784-10000	97.61
DBN	784-400-200-100	98.54
Stacked Auto-Encoder	784-400-200-100	98.63
Stacked ELM-AEs	784-700-700-10000	99.00
CNN	784-5_6-5_12	98.69
CNN	784-5_12-5_12	98.74
RBM-SAE-RBM	784-400-200-100	98.61
CRBM -RBM	784-5_12-400	96.81
RBM-ELM	784-10000	97.91

3.2 多应用支持

基于 DLPF,可以便捷地设计不同模型支持多种应用.目前,DLPF 已经成功应用于手写字符识别、自然图像识别(CIFAR-10)、合成孔径雷达(SAR)遥感图像分类和三维物体识别等应用,并已有研究者愿意使用 DLPF 开展医疗图像疾病检测、驾驶员危险动作识别等应用的研究工作.

图 9 是使用 DLPF 构建的 DBN 模型对手写字符无监督提取的特征.从特征图可以看出,DBN 有效提取了字符的一些边缘信息和纹理特征,这些特征比原图的像素特征具备更好的区分度.

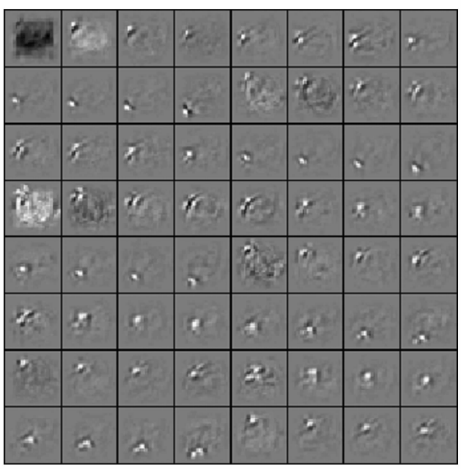


Fig. 9 Features of MNIST dataset selected by DBN model.

图 9 DBN 模型提取的手写字符特征

CIFAR 自然图像分类直接使用无监督网络的分类精度只有 46%,但使用有监督网络分类精度可达 55%左右,如果进行进一步调参并加入 RGB 三通道训练,分类精度将会更高.

文献[19]使用 DLPF 完成实验,将深度学习应用于遥感图像分类中发现,DBN 较其他 3 种算法 SVM[20], neural networks (NN), stochastic Expectation-Maximization (SEM)[21],取得了更好的分类效果,在总体分类精度上有较大提升,分类结果与真实的物类别更符合,如图 10 所示:

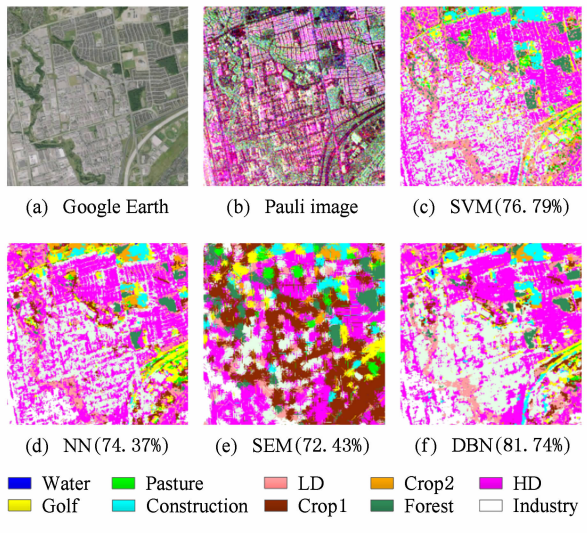


Fig. 10 Remote sensing image classification[21].

图 10 遥感图像分类[21]

DLPF 也可以被应用于三维物体识别和分类.通过 DLPF 设计的三维卷积极限学习机在三维物体分类的数据集上能够达到 87.8%的分类精度,比普林斯顿大学在同样数据集的分类精度(86%)更高[22]且训练速度更快.图 11 所示是得到的三维物体的特征.

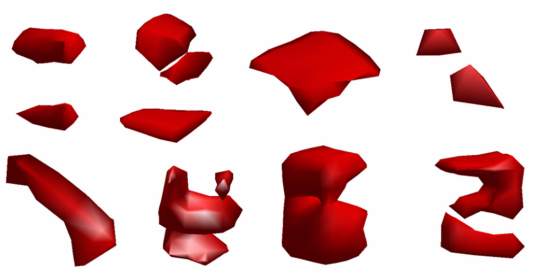


Fig. 11 Features of 3D object.

图 11 三维物体特征

3.3 超参数并行搜索

采用 2 个计算节点进行超参数并行搜索实验,一方面证明并行搜索的有效性,另一方面证明不同超参数对最终预测精度的影响.本节的实验数据集是 MNIST 手写字符库,模型采用单层 RBM+SoftMax

分类器,并行搜索学习率(learning rate)和隐层节点数(number of hidden nodes)这 2 个超参数对模型分类准确率的影响,搜索过程中固定其他超参数.通过测试得到不同的学习率和隐层节点数对最终预测准确率的影响,超参数搜索可以找到分类准确率最高的超参数组合.与单节点相比,双节点测试时间减少了将近一半,表明并行搜索是有效的.图 12 展示了不同超参数组合的预测错误率搜索空间.

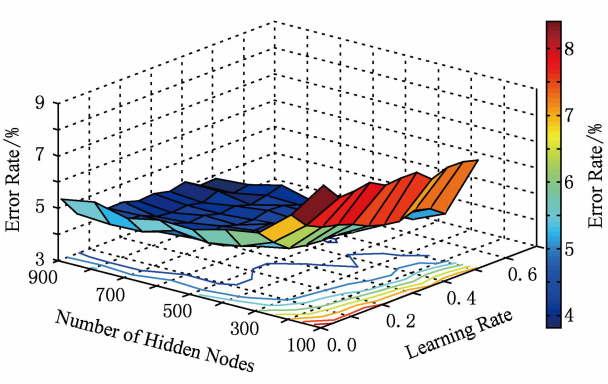


Fig. 12 The hyper-parameter search space.

图 12 超参数搜索空间

从图 12 可以得出 3 点结论:1)不同超参数组合对最终预测精度是有影响的,如果超参数选择不好,会直接导致预测错误率偏高;2)就手写字符数据集而言,隐层节点数目越多,对应的预测错误率越低,但训练时间也越长;3)选择好的学习率有助于快速达到最优解(可能是局部最优),但学习率相近时对性能影响不大.

3.4 时间性能

图 13 和图 14 分别展示了基于 CPU-GPU 异构平台对 RBM 模块和 SGD 算法模块的加速效果.

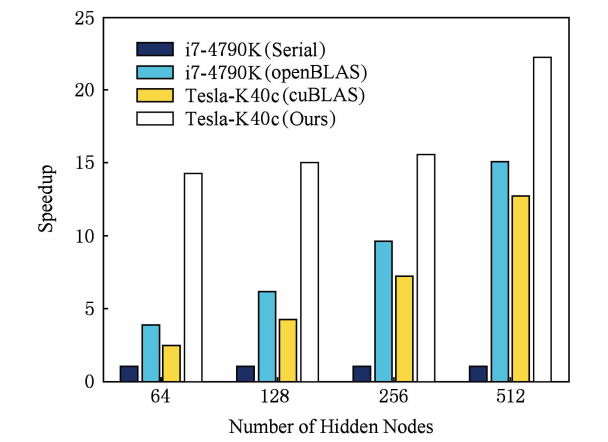


Fig. 13 Speedup comparison of RBM module.

图 13 RBM 模块加速效果对比

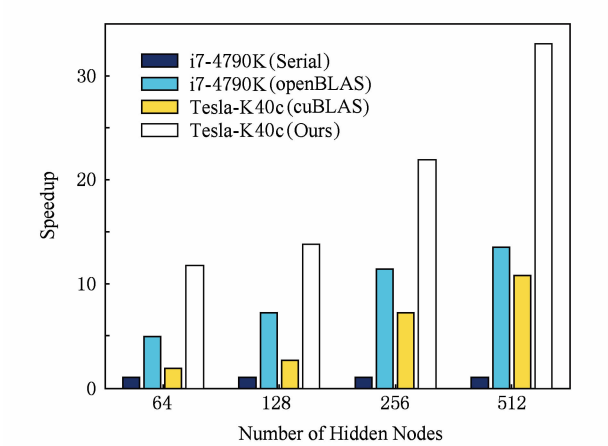


Fig. 14 GPU speedup comparison of SGD algorithm.

图 14 SGD 算法的 GPU 加速效果对比

RBM 模块和 SGD 算法的 GPU 加速效果与 CPU 相比,已经分别达到 20 多倍和 30 多倍.在同等硬件平台下,RBM 模块的加速效果与文献[23]相比性能更高,加速比达到 1.5 倍.

4 结束语

深度学习在机器学习领域扮演着十分重要的角色,已被广泛应用于各种领域,具有十分巨大的研究和应用前景.然而,深度学习也面临着诸多方面的挑战.本文针对这些挑战,设计了一种基于异构体系结构的深度学习编程框架.该框架建立了统一的模块库,能可视化地进行深度学习模型构建,从而提高了编程的便捷性;同时在异构平台对算法模块进行加速,从而减少训练时间,进而提高超参数寻优效率.实验结果表明,该编程框架可以灵活构建多种模型,并且对多种应用取得了较高的预测精度.通过超参数寻优实验,可以便捷地获得最优超参数组合,从而推断各种超参数与不同应用的联系.通过 GPU 加速实验可以看到,框架的某些并行算法模块(RBM,SGD)具有较快的计算速度,与 CPU 相比,加速比可达数十倍.

参 考 文 献

[1] Hinton G E, Salakhutdinov R. Reducing the dimensionality of data with neural networks [J]. Science, 2006, 313 (5786): 504-507

[2] Krizhevsky A. Cuda-convnet: A fast C++/CUDA implementation of convolutional (or more generally, feed-forward) neural networks [OL]. [2014-11-12]. https://code.google.com/p/cuda-convnet

[3] LeCun Y, Jackel L D, Bottou L, et al. Learning algorithms for classification: A comparison on handwritten digit recognition [J]. Neural Networks, 1995, 2(3): 261-276

[4] Krizhevsky A, Sutskever I, Hinton G E. ImageNet classification with deep convolutional neural networks [C] // Proc of 2012 Neural Information Processing Systems (NIPS'12). Cambridge, MA: MIT Press, 2012: 1097-1106

[5] Jia Yangqing. Caffe: An open source convolutional architecture for fast feature embedding [OL]. [2014-12-12]. http://caffe.berkeleyvision.org

[6] Bergstra J, Breuleux O, Bastien F, et al. Theano: A CPU and GPU math expression compiler [C] //Proc of the Python for Scientific Computing Conf (SciPy). Austin, Texas: No Starch Press, 2010: 3-11

[7] Povey D, Ghoshal A, Boulianne G, et al. The kaldi speech recognition toolkit [C] //Proc of 2011 Automatic Speech Recognition and Understanding (ASRU 2011). Piscataway, NJ: IEEE, 2011

[8] Curtin R R, Cline J R, Slagle N P, et al. MLPACK: A scalable C++ machine learning library [J]. Journal of Machine Learning Research, 2013, 14(1): 801-805

[9] Ronan C, Clement F, Koray K, et al. Torch7 : A scientific computing framework with wide support for machine learning algorithms [OL]. [2015-01-04]. http://torch.ch

[10] Dean J, Corrado G S, Monga R, et al. Large scale distributed deep networks [C] //Proc of the Neural Information Processing Systems (NIPS'12). Cambridge, MA: MIT Press, 2012: 1223-1232

[11] Yadan O, Adams K, Taigman Y, et al. Multi-GPU training of convnets [OL]. [2015-01-12]. http://kr.nvidia.com/content/tesla/pdf/machine-learning

[12] Yu Kai. Large-scale deep learning at Baidu [C] //Proc of Int Conf on Information and Knowledge Management (CIKM 2013). New York: ACM, 2013: 2211-2212

[13] Zou Yongqiang, Jin Xing, Li Yi, et al. Mariana: Tencent deep learning platform and its applications [C] //Proc of the Int Conf on Very Large Data Bases (VLDB 2014). Amsterdam, the Netherlands: Elsevier, 2014: 1772-1777

[14] Conrad S, Ryan C. Armadillo: An open source C++ linear algebra library for fast prototyping and computationally intensive experiments [R]. Queensland, Australia: National Information Communications Technology Australia (NICTA), the University of Queensland, 2010

[15] Zhang Xianyi. OpenBLAS: An optimized BLAS library based on GotoBLAS2 1.13 BSD version [OL]. [2014-12-11]. http://www.openblas.net

[16] NVIDIA. NVBLAS library: A GPU-accelerated library that implements BLAS [OL]. [2014-10-11]. http://developer.nvidia.com

[17] University of Tennessee, University of California, Berkeley, etc. LAPACK (linear algebra package): A standard software library for numerical linear algebra [OL]. [2014-10-12]. http://www.netlib.org/lapack

[18] Huang Guangbin, Zhu Qinyu, Siew C K. Extreme learning machine: A new learning scheme of feedforward neural networks [J]. Neural Networks, 2004, 2(3): 985-990

[19] Lü Qi, Dou Yong, Niu Xin, et al. Remote sensing image classification based on DBN model [J]. Journal of Computer Research and Development, 2014, 51(9): 1911-1918 (in Chinese)
(吕启, 窦勇, 牛新, 等. 基于DBN模型的遥感图像分类 [J]. 计算机研究与发展, 2014, 51(9): 1911-1918)

[20] Chang C, Lin C. LIBSVM: A Library for support vector machines [OL]. [2014-11-20]. http://www.csie.ntu.edu.tw/~cjlin/libsvm

[21] Nielsen S F. The stochastic EM algorithm: Estimation and asymptotic results [J]. Bernoulli, 2000, 3(6): 457-489

[22] Wu Z, Song S, Khosla A, et al. 3D ShapeNets for 2.5D object recognition and next-best-view prediction [OL]. [2015-02-02]. http://arxiv.org/abs/1406.5670

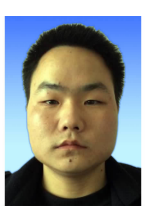
[23] Lopes N, Ribeiro B. Towards adaptive learning with improved convergence of deep belief networks on graphics processing units [J]. Pattern Recognition, 2014, 47(1): 114-127

Wang Yueqing, born in 1988. PhD candidate. His main research interests include computer architecture, machine learning and high performance computing.

Dou Yong, born in 1966. Professor and PhD supervisor. Senior member of China Computer Federation. His main research interests include computer architecture and reconfigurable computing.

Lü Qi, born in 1987. PhD candidate. Student member of China Computer Federation. His main research interests include computer architecture, machine learning and remote sensing image processing.

Li Baofeng, born in 1980. Assistant professor at National University of Defense Technology. His main research interests include design of supercomputer and high performance computer architecture.



Li Teng, born in 1991. Master candidate. His main research interests include computer architecture and high performance computing.