doi:10.3969/j.issn.1001-4616.2019.03.002

基于动态指导的深度学习模型稀疏化执行方法

孙茹君,张鲁飞

(数学工程与先进计算国家重点实验室,江苏 无锡 214125)

[**摘要**] 以深度学习为代表的人工智能技术迅速发展,庞大的数据、模型,更大的计算量和更复杂的计算都对模型的执行提出了挑战. 在实际应用中,资源和应用的动态特征以及用户的动态需求,需要模型执行的动态性来保证. 而稀疏化是在资源受限、用户需求调整情况下动态模型的执行重要手段. 目前主流的稀疏化技术主要是针对特定问题的稀疏化,且针对推理的多,针对训练的少,缺乏在训练执行阶段进行动态调整和稀疏化的手段. 本文在对深度学习领域的基本计算单元进行可稀疏性分析的基础上,进一步分析了模型执行的不同层面、不同组成部分的稀疏化能力;经过对动态需求、模型稀疏化策略的建模后,提出了基于动态指导的深度学习模型稀疏化执行方法,并进行了基本实验;最后从量化建模与量化实验的角度对今后的研究工作提出了展望.

[**关键词**] 深度学习,稀疏化方法,资源受限,动态调度

[**中图分类号**]TP311　[**文献标志码**]A　[**文章编号**]1001-4616(2019)03-0011-09

Dynamic Sparse Method for Deep Learning Execution

Sun Rujun,Zhang Lufei

(State Key Laboratory of Mathematical Engineering and Advanced Computing,Wuxi 214125,China)

Abstract:Artificial intelligence,especially deep learning,is developing rapidly. Large data,large models,complex control flow,and more computations have challenged model execution in both training and inference stage. Practically,resources and user demands show dynamic characteristics,and need to be guaranteed by executing model dynamically. Finding sparsity is an important means of dynamically model execution under resource constraints and the changing user demand. At present,the main sparse method is focused on specific problems. Most of them are used in inference. We are in urgent need of dynamic sparse method in training. In this paper,we firstly analyzed the chance of sparsicy in basic deep learning models and further analyzed the sparseization ability of different layers and different components of models. After modeling the dynamic requirements and sparseness strategy,we proposed a sparse execution method of deep learning based on dynamic execution,and did some basic experiments. Finally,from the perspective of quantitative modeling and quantitative experiments,the future research work is prospected.

Key words:deep learning,sparse method,resource limitation,dynamic scheduling

虽然面向人工智能的体系结构在近年来发展迅速,但人工智能应用的发展更快. 庞大的数据、模型,更大的计算量和更复杂的计算都对模型的执行提出了挑战. 在资源受限的条件下,如何有效地执行应用?如何利用模型的动态特征? 在分析模型的各个组成部分时,如何合理地判断不同数据、参数、逻辑关系的重要程度?

稀疏性是很多科学计算得以高效执行的重要保证,而能否利用好更高的稀疏程度与应用执行时的动态能力密切相关. 虽然在已有的以深度学习卷积神经网络为代表的人工智能模型中,大量的计算都是稠密矩阵乘法,但随着应用的发展和对执行的限制条件越来越苛刻,研究人工智能模型的稀疏性能够有效指导未来人工智能应用的高效训练执行.

本文将从人工智能模型中具有稀疏特征的结构、稀疏性与稀疏化方法、人工智能基本算子的稀疏性分

收稿日期:2019-07-05.

基金项目:国家重点研发计划(2016YFB1000505)、国家重点研发计划(2017YFB0202001)、国家自然科学基金项目(9143020017).

通讯联系人:孙茹君,博士研究生,研究方向:人工智能运行环境. E-mail:sun.rujun@meac-skl.cn

析入手,讨论模型的稀疏特征,并提出基于动态指导的模型稀疏化方法.

1　背景和相关工作

人工智能应用中数据的稀疏性有多方面的研究. 基于稀疏表示的研究最为典型,稀疏表示指的是“为普通稠密表达的样本找到合适的字典,将样本转化为合适的稀疏表达形式,从而使学习任务得以简化,模型复杂度得以降低[1]”. 即使没有稀疏表示,很多数据本身也体现出一定的稀疏性. 例如,天文学中的稀疏数据[2]会作为机器学习算法的输入从而得到星系自动分类的结果. 文本文档通常表示为高维稀疏向量(典型的维度是几千,而典型的稀疏度是 90%到 95%)[3]. 医学诊疗数据由于其有限性,也有稀疏的特征,如视网膜数据[4]、光声层析成像数据[5]等. 连最常见的手写数字也体现出一定的稀疏特征[6].

随着应用的复杂和目标的提高,越来越多的极深网络开始出现,例如最初针对图像识别设计的 VGG[7]、针对语音识别的多语言极深卷积网路[8]、针对人脸识别应用的 DeepID3 网络[9]、文字识别的序列化深度卷积网络 SVDCNN[10]. 这些深层的网络参数量极大,计算量极大,对于应用设计和运行来说都是挑战.

网络稀疏化是一种有效的简化计算方法:通过减少神经元之间的连接来减少参数、计算甚至通信. Thom[11]研究了基于稀疏表示的快速分类、多层感知器的稀疏连接和在线编码器的稀疏化,与传统的非稀疏模型相比,稀疏性可以作为正则化器,并且可以获得明显更好的分类结果;训练分类器的计算复杂度可降低大约一个数量级. 针对专用领域的特定稀疏神经网络研究繁多:卷积神经网络的稀疏化可以通过在卷积层之前添加 bias 层对参数初步处理,使后续的卷积核参数降低 90%,且针对 ILSRVR2012 的实验证明对于精确度的影响少于 1%[12];循环神经网络的稀疏化采用一般的预训练、剪枝、再训练的方法,百度的实验[13]可将网络参数量减少到 1/8 且几乎不影响精确度;语音识别和增强领域的深度神经网络的稀疏化,可以通过整流线性单元(ReLU)实现,在隐藏层之间建立线性连接,Xu[14]证明在对稀疏网络进行修剪和重新训练之后,可以在不降低性能的情况下大大减少计算和存储需求. 在计算资源受限的 FPGA 上,通过 L1 正则化在卷积层预处理数据不仅可以提高系统的准确性,而且还可以进一步减少后续全连接层的计算量[15]. 此外还有自动编码器的稀疏化[16-17],循环神经网络的稀疏化[18]. 除了应用领域的稀疏化分析,在体系结构设计上也有针对稀疏化的硬件结构,如 Cambricon-X[19].

2　人工智能模型的稀疏特征分析

直观地理解,“稀疏特征”与模型中“0”的比例关系密切. 而人工智能模型包括数据、模型结构等多个方面.

稀疏度:单位计算结构中,0 作为参数所占总参数的比例. 在人工智能模型中,数据稀疏度包含输入数据、模型参数等数据类别的稀疏度,结构稀疏度通过参数设 0,使之不发挥作用.

为了对人工智能模型中的稀疏度有初步的理解,首先分析各种典型模型的结构和参数. 这里用 ONNX 模型集合[20]中的模型作为基本的分析对象,该模型集合中包含预训练的模型参数,可为典型的参数分布分析提供参考.

模型的参数稀疏度来自两个方面:人工智能领域的模型设计者所设计的稀疏模型,运行时系统对于模型进行的稀疏化改动. 前者需要专业的知识,而后者是我们研究的重点. 本节将在没有应用知识的条件下,让运行时系统尽可能地对模型进行合理的稀疏化改动和执行.

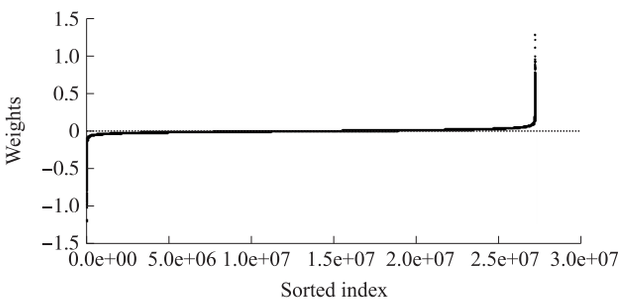

图 1　GoogleNet Inception V3 中参数分布情况

Fig. 1　Parameter distribution in GoogleNet Inception V3

图 1 是 GoogleNet Inception V3 预训练网络所有参数的分布情况. 从图中可以看出,绝大多数参数在 0 附近,只有极少数参数的绝对值较大. 实际上只有 0.4%的参数绝对值大于 0.1(即最大参数的 10%),44.3%的参数绝对值小于 0.01(即最大参数的 1%).

图 2(a)~(f)分别刻画了典型预训练模型中卷积和归一化算子的参数分布.

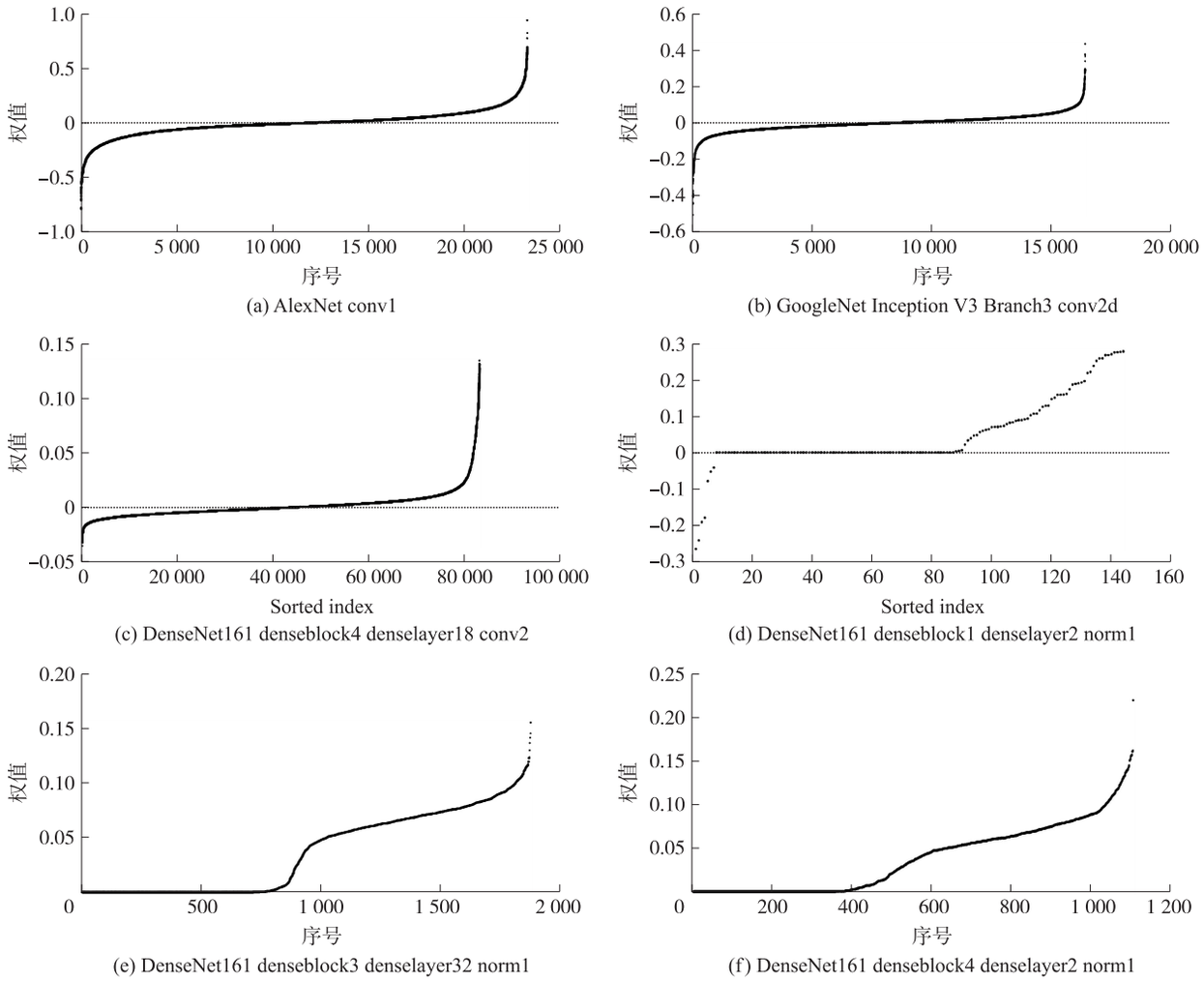

图 2　典型模型中算子的参数分布

Fig. 2　Parameter distribution of typical operators in some models

针对 AlexNet 网络 5 个卷积单元,GoogleNet Inception V3 网络 98 个卷积单元和 DenseNet161 网络 160 个卷积单元的参数分析:大部分参数都在 0 附近,只有少部分绝对值较大的参数;绝对值较大的参数在算子参数中的比例在 5%到 25%,因此接近“0”的参数比例在 75%以上.

归一化单元只在部分模型中存在,其中参数有 30%或以上是 0,其余数据呈近似线性分布.

从以上例子可以看出,人工智能模型中的确有一定比例的数据等于 0 或接近 0. 等于 0 的数据自然可以不做计算,而接近 0 的数据能否直接近似成 0,下文将进行进一步实验.

3　基本算子的稀疏性分析

在深度学习模型中,基本算子一般分为 4 类:稠密计算(如卷积)、按元素计算(如池化、偏移、sigmoid 等)、遍历计算(如归一化)、数据排布(如数组合并).

稠密计算的计算密度较高,常见的计算密度数值为 $O(n)$. 它的实际数值与单位计算单元的体系结构参数(例如内存大小)有关.

按元素计算的计算特征与元素本身的计算有关,与数据量和体系结构几乎无关. 例如 exp 需要对于元素或者向量中的每个元素求 e 的幂,power 需要对于元素或者向量中的每个元素求幂,二者的计算密度与 exp 和 power 函数本身的计算复杂度和实现有关.

遍历计算除了按照元素进行简单的基础操作之外,还需要附加一定的整体操作,例如比大小、求极值、求误差等. 其基础操作往往简单,整体操作靠累加或者维护一个具体的值实现,主要的计算过程是遍历,

其计算密度是常数.

数据排布是对数据的基本操作,包括添加外围 0 元素、拆分成块、合并等. 其基本操作是访存和数据转移,计算密度低,也可以通过前后层(算子)整合来优化隐藏这部分开销.

以上类别的基本算子中,除数据排布之外,其余的算子都涉及到计算过程. 稀疏性也在计算过程中体现. 本节将从两个方面分析稀疏性:计算所导致的(输出)数据稀疏性和算子参数稀疏性. 如图 3 所示.

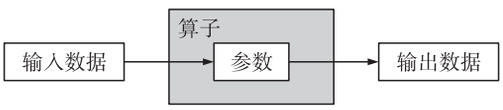

图 3　算子执行中的数据

Fig. 3　Data in operator execution

表 1 列举了稠密、按元素计算、遍历计算 3 类基本算子的参数特征、参数可稀疏化能力、输出数据特征、输出数据稀疏性. 只有参数较多的算子(例如 conv, bias, lstm, BN)才有参数稀疏化的意义. dropout, relu 计算本身可以使输出数据稀疏化. 对于输入数据为 0 的情况,按元素计算的所有基本算子都能保持输出为 0,稠密计算和遍历计算由于涉及到相邻数据,还需要考虑邻居数据的影响.

表 1　基本算子的稀疏性和可稀疏化分析

Table 1　Sparse analysis of basic operators

基本算子	稠密	按元素计算											遍历计算			
	conv	absval	bias	dropout	eltwise	exp	relu	power	scale	sigmoid	lstm	tanh	BN	lrn	loss	pooling
参数	有	无	少	无	无	无	无	无	少	无	有	无	有	无	无	无
稀疏化	可	—	否	—	—	—	—	—	—	—	可	—	否	—	—	—
输出数据	有	有	有	有	有	有	有	有	有	有	有	有	多	少	少	少
稀疏性	无	无	无	有	无	无	有	无	无	无	无	无	无	—	—	—

4　模型稀疏化策略

稀疏性表示计算单元本身的稀疏度,而通过一些稀疏化方法,可以增加计算单元的稀疏度从而减少数据量、计算量或模型连接关系.

在传统的机器学习领域,稀疏化方法包括 L1 正则化、稀疏贝叶斯方法等. 这些方法也是以深度神经网络为代表的人工智能模型稀疏化的重要指导.

4.1　数据稀疏化

L1 正则化:一般的 L1 正则化的主要操作是求权值向量中的各元素绝对值之和. 用在损失函数的计算中,原始参数需要与带系数的 L1 正则化项求和,例如 $X'=X+\lambda\Sigma|w_i|$. 因此,后者用在损失函数项,以对参数进行一些限制. 在传统的机器学习中,L1 正则化可以产生稀疏模型(即稀疏的权值系数),因此可以用来特征选择.

稀疏贝叶斯方法:稀疏贝叶斯方法作为实现参数的简约表示用在回归和分类中,意图只保留很少非零权重作为参数模型. 稀疏贝叶斯学习中,假设所要求解的对象符合参数化的高斯分布,通常会设置某个阈值,将趋近于 0 的参数置为 0(稀疏化). 这样,在各已知条件和对解的先验分布进行假设的基础上,利用贝叶斯规则可求得后验高斯分布的参数,所求的解由后验高斯分布的平均值给出.

掩码法:通过设置掩码(mask),将数据中符合掩码约束的部分数据过滤出来进行计算,从而忽略其他数据. 掩码法还有包括特征提取等不同的计算方式和变形.

参数量化:通过用更少的比特数表示参数,可以实现更小的内存开销.

4.2　结构稀疏化

人工智能模型中,结构也由参数来表示. 结构稀疏化指:将靠近 0 的参数修改为 0,以删除在神经元意义下的该神经元.

结构稀疏化是压缩神经网络的重要方法,但经过结构变化的神经网络往往会面临精确度下降的问题,一个解决方案是将训练数据重新输入压缩网络训练,优化参数.

另外一种结构稀疏化的方法是直接训练具有稀疏连接特征的神经网络(稀疏进化训练):在训练的初始阶段,确定好神经元数量之后随机设定一个稀疏连接;每一轮训练之后删除权值最小的连接,再随机加入一条新连接;直到某一结束条件. 此方法易扩展,可适应任意大的模型[21].

此外,还可以将大数据上广泛使用的关联数组代数(Associative Array Algebra)应用于深度神经网络,以构建更大规模的稀疏神经网络[22].

4.3 执行稀疏化

人工智能模型的训练执行有多种迭代方法,例如 SGD、Adagrad、Adadelta、Adam、Adamax、Nadam 等. 在执行阶段,稀疏化有多种方法. 其一是惰性参数更新,例如并行执行的模型参数可以经过多轮之后再更新. 此外,在训练期间执行动态稀疏图[23],每次迭代中仅选择少量具有高选择性的神经元,可以显著节约内存并减少操作数量.

对于机器学习领域的算法,如决策树或者 EM 等,可以从离散数据中提取单向或双向计数,以稀疏执行;或用朴素贝叶斯对推断聚类,以稀疏执行[24].

5 基于动态指导的模型稀疏化方法

在实际应用中,常常会遇到资源受限、时间受限、精度受限的人工智能训练需求,且这些需求可能是实时产生、动态变化的. 利用稀疏化的方法,可以为实际应用执行时复杂多变的情况提供高效的应对策略.

5.1 动态需求建模

分别将对于资源(计算力 c,存储空间 m)、时间(t)、精度(p)的需求精确描述. 用实际运行中正在使用的模型中的上述 4 种因素参数与用户期待参数相比得到改进参数 $r_{\mathrm{c}},r_{\mathrm{m}},r_{\mathrm{t}},r_{\mathrm{p}}$:

$$r_{\mathrm{c}}=\frac{c_{\mathrm{sparsemodel}}}{c_{\mathrm{currentmodel}}},\ r_{\mathrm{m}}=\frac{m_{\mathrm{sparsemodel}}}{m_{\mathrm{currentmodel}}},\ r_{\mathrm{t}}=\frac{t_{\mathrm{sparsemodel}}}{t_{\mathrm{currentmodel}}},\ r_{\mathrm{p}}=\frac{p_{\mathrm{sparsemodel}}}{p_{\mathrm{currentmodel}}}$$

这些需求,以及未来可能出现的更多需求,为模型的改动提供了不同维度的参考信息.

5.2 模型稀疏化方法

模型执行的结构可以表示为表 2.

表 2 模型执行结构与稀疏化建模

Table 2 Sparse execution model

模型执行结构	举例		稀疏化方法 f	稀疏化影响
外层执行方式	SGD		f_{iter}	$(r_{\mathrm{c}},r_{\mathrm{m}},r_{\mathrm{p}},r_{\mathrm{t}})_{\mathrm{iter}}$
内层计算流图	AlexNet		f_{graph}	$(r_{\mathrm{c}},r_{\mathrm{m}},r_{\mathrm{p}},r_{\mathrm{t}})_{\mathrm{graph}}$
基本算子	conv	输出数据	$f_{\mathrm{unit\_out}}$	$(r_{\mathrm{c}},r_{\mathrm{m}},r_{\mathrm{p}},r_{\mathrm{t}})_{\mathrm{unit\_out}}$
		结构参数	$f_{\mathrm{unit\_weight}}$	$(r_{\mathrm{c}},r_{\mathrm{m}},r_{\mathrm{p}},r_{\mathrm{t}})_{\mathrm{unit\_weight}}$

稀疏化方法 f 是由多层执行结构组成的一整套策略,每层结构都可以包含多个基本单元,每个单元可以使用 0 或多个稀疏化方法. 因此,一系列方法产生的影响需要综合评价.

在实际应用中,稀疏化需求的产生来源于多个方面,例如:运行环境本身的状态(资源使用率等),用户对于执行的新需求(希望实时调整模型结构、执行时间、执行精度、执行开销等). 这些需求往往可以通过系统资源环境接口或者用户反馈直接获得.

设计人工智能模型并运行是一个多次反馈、实时调整的过程. 动态指导体现在对于动态需求的实时响应.

综合动态需求和模型稀疏化策略两方面的模型,可以得到基于动态指导的模型稀疏化方法的表示:

$$\mathrm{argmin}\left\{\int_{t}\left|\ g_{\mathrm{demand}}(t)\ -g_{f}(t)\ \right|\right\}.$$

方法的目标是尽量满足实时需求. 其中可以表示资源、期望时间和精度随时间变化的函数,需求部分 g_{demand} 可通过参数 r 求得. 实际执行时,“实时需求”除了实时产生之外,还包括额外的输入需求,上式的积分过程难以实际计算. 为此,将模型稀疏方法表示分解到每一个需求的时刻 t_i 得到新的表示 $\mathrm{argmin}\{g_{\mathrm{demand}}(t_i)-g_f(t_i)\}$.

$f=\bigcup_j f_j$ 是采用的所有稀疏化方法的全集,有的方法可能嵌套在其他方法之中. 更进一步,该稀疏化方法集合可用函数表示,但难以准确求解,因此提出近似求解方式.

在近似求解过程中,需要综合比较不同稀疏方式的开销和效果,对照目标进行选择.

算法 1　稀疏化方法近似方式

输入:r_c, r_m, r_t, r_p
输出:U
1　计算 c, m, t, p 的实际需求;
2　初始化 U, U_c, U_m, U_t, U_p 为空;
3　for require in $\{r_c, r_m, r_t, r_p\}$ do
4　for f in $\{\{f_{unit\_weight}\}, \{f_{unit\_out}\}, \{f_{graph}\}, \{f_{iter}\}\}$ do
5　　寻找满足 require 条件的多种 f 并评价其影响,加入对应的集合 $U_{require}$
6　end for
7　end for
8　选择满足最多需求条件的 f 集合 $U \leftarrow U_c \cap U_m \cap U_t \cup U_p$,当后者为空时选取满足其中主要需求的方法集合作为 U;
9　评价 U 中各方法的综合影响,并酌情调整.

6　实验

6.1　实验环境

实验在 Intel Core i7,16 GB 1 867 MHz DDR3 上完成. C++编译器为 clang-700.1.81,原始框架 caffe 的版本为 128797eb.

目前的人工智能训练任务绝大多数都是数据并行,单机能够容纳一个完整的模型. 系统级的资源状态粒度较粗,而芯片级的处理器利用率、内存占用率等状态更能精细反映当前处理器上执行任务的实时需求. 此外,通过合理组合多个芯片级的任务调整,可以应对更大规模的系统级需求. 因此,在单机上完成本实验能说明本方法的效用.

6.2　结构参数稀疏化

实验 1 结构参数稀疏化采用简化的 AlexNet 网络,进行手写数字识别. 训练数据是简化版的 Mnist(包含 0~4 共 5 种手写数字). 网络共包含 41 700 个权重参数,其中 layer1 包含所有的卷积层共 32 000 个参数,layer2 包含所有的全连接层共 2 500 个参数.

定义稀疏度参数为 s,其取值范围是[0,1],用来稀疏化 $w = s \times |w_{max}|$ 以下的参数. 稀疏度 $p_s = c_{w.below\_threshold}/c_{w.all}$,设定权值绝对值在最大权值绝对值的 s 比例以下的参数为 0.

图 4(a)~(d)表示在设定稀疏度参数为 0.1、0.25、0.5、0.75 条件下,迭代过程中的稀疏度变化. 由于有 2 个 layer,每个 layer 的最大稀疏度为 1,因此总的比例在图中以和的形式呈现. 可以看出,保留绝对值较大的权值可以显著地改变模型参数的稀疏度,即使只删除 $s=0.25$ 的参数,总稀疏度也接近 0.999.

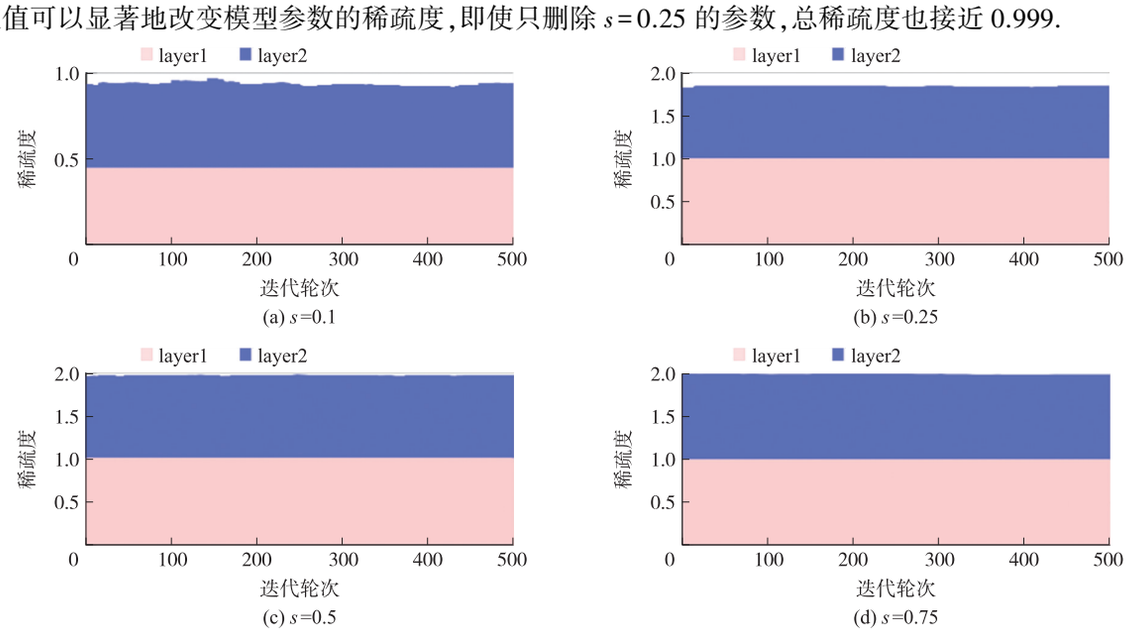

图 4　稀疏化实验迭代过程(模型参数权值)

Fig. 4　Sparsity in parameter sparse method

6.3 输出参数稀疏化

实验 2(输出参数稀疏化)采用 caffe 框架,对于 Lenet 网络进行 Mnist 手写数字识别. 实验中 base loss rate=0.01,momentum=0.9,weight decay=0.000 5,lr policy="inv",gamma=0.000 1,power=0.75. 将基本算子的输出部分稀疏化作为一个特殊的 layer 插入到原始网络中进行实验. 如图 5,在 Lenet 网络的卷积层、池化层、全连接层中进行输出数据稀疏化的实验,原始网络结构和实际网络结构分别为上下两部分.

实验的运行结果如图 6,设定基本算子输出部分绝对值分别在 0.1、0.01、0.000 01 以下的数据为 0,描述迭代过程 loss(判断收敛的条件)曲线. 可以看出,对于基本算子输出数据的微小调整基本不影响实验结果.

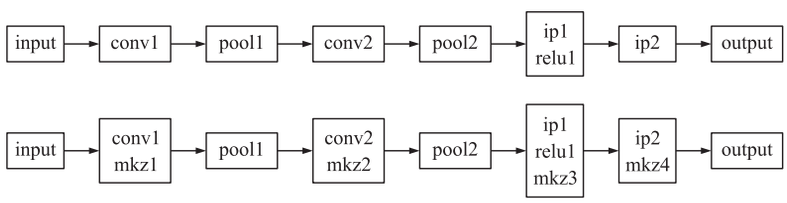

图 5 稀疏化实验模型结构变换,上:变换前;下:变换后

Fig. 5 Model structure transformation in sparse method

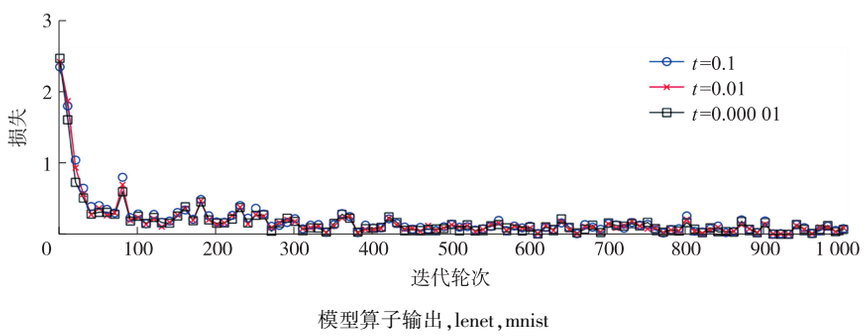

图 6 稀疏化实验迭代过程

Fig. 6 Convergence in operator output sparse method

6.4 基于动态指导的模型稀疏化

动态稀疏化需要动态运行环境的支持. 目前已有的人工智能框架大多是静态执行,其调度粒度为一次训练,虽然 DyNet 等动态框架支持动态计算图,但需要在每轮迭代之前重新构建计算图,引发的开销较大.

图 7 和图 8 基准实验是需求被满足且无变化的模型执行过程,如"基本"一列. 模拟一个简化的真实

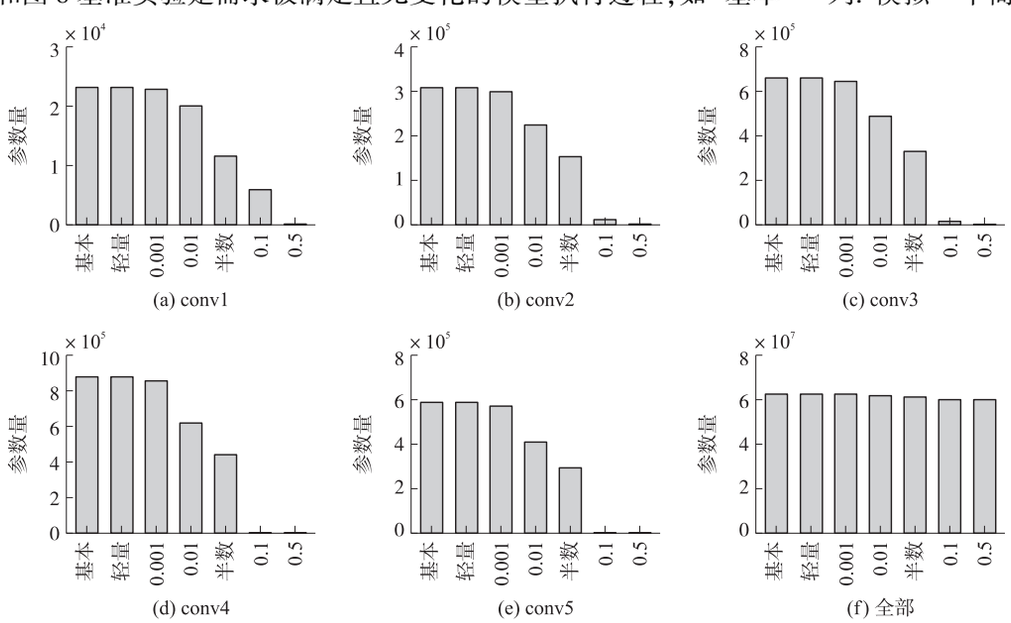

图 7 动态稀疏化实验 AlexNet 一次调整-参数量

Fig. 7 Quantitative parameters in dynamic sparse method of AlexNet

场景:在计算执行开始时,不作稀疏要求,在计算执行到特定轮数时,设置稀疏化调整需求分别为“轻量”(在不影响模型的条件下稀疏化)、0.001(稀疏阈值为 0.001,后同)、0.01、0.1、“半数”(稀疏度为 0.5)、0.5(稀疏阈值为 0.5). 子图(a)~(e)分别为不同的算子模块的情况,子图(f)为全模型的情况. 可以看出稀疏化基本可以按照指导要求进行调整.

只调整卷积算子对于总模型参数的影响很小(图 7(f)),因为绝大多数参数都在 FC 算子中;但对于总计算需求的影响符合稀疏需求(图 8(f)). 在训练过程中,稀疏化的目标主要是减少计算量,因此针对计算占比较高的算子稀疏化是有意义的.

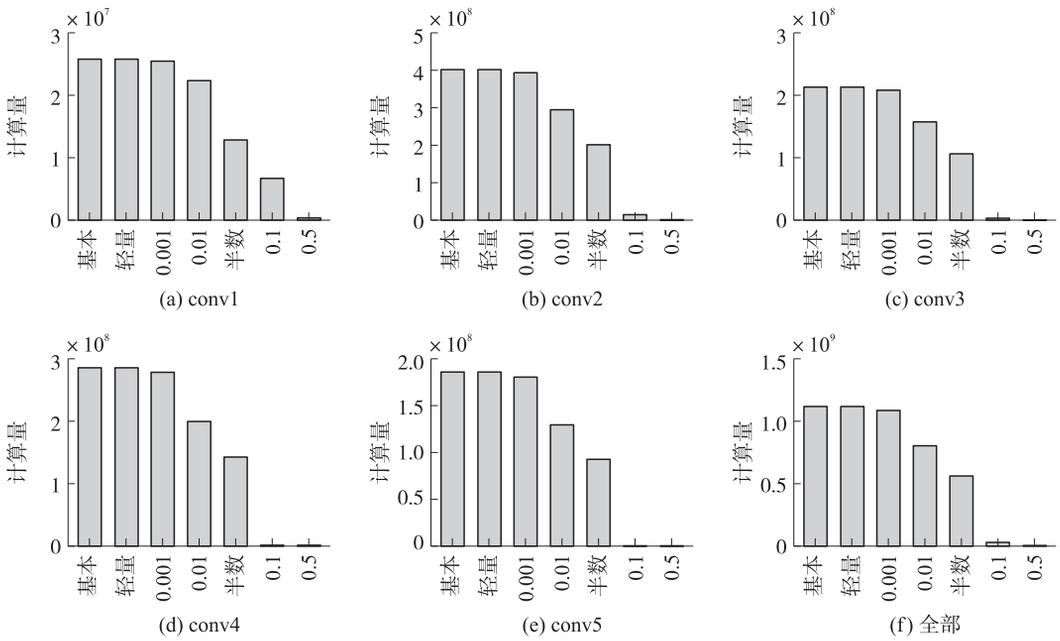

图 8　动态稀疏化实验 AlexNet 一次调整-计算量

Fig. 8　Quantitative computations in dynamic sparse method of AlexNet

因此实验采用 caffe 作为基础框架并修改,为了响应动态需求,在执行的每一轮(epoch)开始前加入指导检测.

图 9 是 Lenet 和 Cifar10 quick 两个网络在 Mnist 和 Cifar10 两个数据集上的训练过程. 其中分别统计了经过卷积(conv)、正则化(relu)、归一化(ip)这几类模块数据的稀疏度变化曲线. 经过一定迭代轮数之后,若出现稀疏化需求则进行调整,图 9(a)中由于执行过程中资源需求较小,因此减少稀疏程度是可以接受的;图 9(b)中在拐点处出现了稀疏化需求,模型执行适当调整,增加了部分单元的稀疏度.

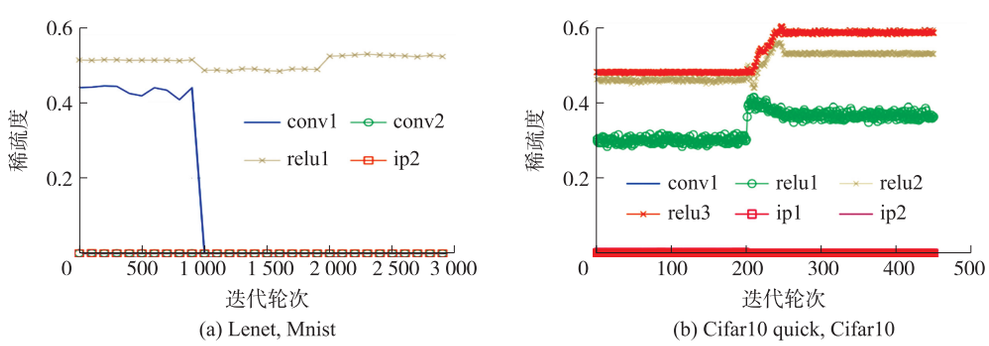

图 9　模型训练过程中的数据稀疏度变化

Fig. 9　Sparsity in training iterations

7　总结与未来工作

稀疏化是未来人工智能领域的一个重要发展趋势. 本文提出了基于动态指导的模型稀疏化方法,可以在保持模型基本功能的同时为复杂多变的应用执行情况提供高效的应对策略,以适应用户和体系结构

的需求或变化,为在不同环境和条件下的深度学习应用提供了自适应的调整方案.

当然,本文的工作仅是自适应模型稀疏化的一部分. 在未来的研究中,我们将会进一步充实并完全量化各种稀疏化方案,将此技术作为工具,融合进常见的人工智能运行环境中.

[参考文献]

[1] 周志华. 机器学习[M]. 北京:清华大学出版社,2016.

[2] JENKINSON J,GRIGORYAN A M,HAJINOROOZI M,et al. Machine learning and image processing in astronomy with sparse data sets[C]//2014 IEEE International Conference on Systems,Man,and Cybernetics,SMC 2014. San Diego,CA,USA,2014:200-203.

[3] DHILLON I S,MODHA D S. Concept decompositions for large sparse text data using clustering[J]. Machine learning,2001,42(1/2):143-175.

[4] YANG C H,LIU F,HUANG J,et al. Auto-classification of retinal diseases in the limit of sparse data using a two-streams machine learning model[J]. CoRR,2018,abs/1808.05754.

[5] ANTHOLZER S,HALTMEIER M,SCHWAB J. Deep learning for photoacoustic tomography from sparse data[J]. CoRR,2017,abs/1704.04587.

[6] HAN S,DALLY B. Efficient methods and hardware for deep learning[D]. Stanford:Stanford University,2017.

[7] SIMONYAN K,ZISSERMAN A. Very deep convolutional networks for large-scale image recognition[J]. CoRR,2014,abs/1409.1556.

[8] SERCU T,PUHRSCH C,KINGSBURY B,et al. Very deep multilingual convolutional neural networks for LVCSR[J]. CoRR,2015,abs/1509.08967.

[9] SUN Y,LIANG D,WANG X G,et al. Deepid3:face recognition with very deep neural networks[J]. CoRR,2015,abs/1502.00873.

[10] DUQUE A B,SANTOS L L J,MACêDO D,et al. Squeezed very deep convolutional neural networks for text classification[J]. CoRR,2019,abs/1901.09821.

[11] THOM M. Sparse neural networks[D]. Ulm:University of Ulm,2015.

[12] LIU B Y,WANG M,FOROOSH H,et al. Sparse convolutional neural networks[C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.[S.l.:s.n.]. Boston,MA,USA,2015:806-814.

[13] NARANG S,ELSEN E,DIAMOS G,et al. Exploring sparsity in recurrent neural networks[J]. arXiv preprint arXiv:1704.05119,2017.

[14] XU L,CHOY C,LI Y W. Deep sparse rectifier neural networks for speech denoising[C]//IEEE International Workshop on Acoustic Signal Enhancement,IWAENC 2016. Xi'an,China,2016:1-5.

[15] SALDANHA L B,BOBDA C. Sparsely connected neural networks in FPGA for handwritten digit recognition[C]//17th International Symposium on Quality Electronic Design,ISQED 2016. Santa Clara,CA,USA,2016:113-117.

[16] NG A. Sparse autoencoder[J]. CS294A lecture notes,2011,72(2011):1-19.

[17] ZHANG Y D,HOU X X,LÜ Y D,et al. Sparse autoencoder based deep neural network for voxelwise detection of cerebral microbleed[C]//22nd IEEE International Conference on Parallel and Distributed Systems,ICPADS 2016. Wuhan,China,2016:1229-1232.

[18] KORDMAHALLEH M M,SEFIDMAZGI M G,HOMAIFAR A. A sparse recurrent neural network for trajectory prediction of atlantic hurricanes[C]//Proceedings of the 2016 on Genetic and Evolutionary Computation Conference. Denver,CO,USA,2016:957-964.

[19] ZHANG S T,DU Z D,ZHANG L,et al. Cambricon-x:An accelerator for sparse neural networks[C]//49th Annual IEEE/ACM International Symposium on Microarchitecture,MICRO 2016. Taipei,China,2016:20:1-20.

[20] Open neural network exchange(onnx)model zoo[EB/OL]. [2019-03-01]. https://github.com/onnx/models.

[21] MOCANU D C,MOCANU E,STONE P,et al. Scalable training of artificial neural networks with adaptive sparse connectivity inspired by network science[J]. Nature communications,2018,9(1):2383-2395.

[22] KEPNER J,GADEPALLY V,JANANTHAN H,et al. Sparse deep neural network exact solutions[C]//2018 IEEE High Performance Extreme Computing Conference,HPEC 2018. Waltham,MA,USA,2018:1-8.

[23] LIU L,DENG L,HU X,et al. Dynamic sparse graph for efficient deep learning[J]. CoRR,2018,abs/1810.00859.

[24] CHICKERING D M,HECKERMAN D. Fast learning from sparse data[J]. CoRR,2013,abs/1301.6685.

[责任编辑:顾晓天]