(19)国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 118819869 A
(43)申请公布日 2024.10.22

(21)申请号 202411303417.5

(22)申请日 2024.09.19

(71)申请人 山东浪潮科学研究院有限公司
地址 250101 山东省济南市高新区浪潮路1036号S02号楼

(72)发明人 郝运凯 姜凯 赵鑫鑫 薛海军

(74)专利代理机构 北京君慧知识产权代理事务所(普通合伙) 11716
专利代理师 董延丽

(51)Int.Cl.
G06F 9/50(2006.01)
G06N 3/063(2023.01)
G06N 3/084(2023.01)
G06N 3/098(2023.01)
G06N 5/04(2023.01)

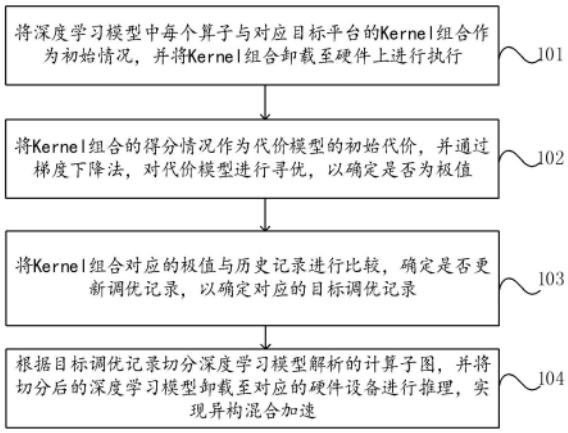
权利要求书2页 说明书11页 附图1页

(54)发明名称

一种基于多种加速卡的异构混合加速方法、设备及介质

(57)摘要

本申请公开了一种基于多种加速卡的异构混合加速方法、设备及介质,涉及并行计算技术领域。方法包括:将深度学习模型中每个算子与对应目标平台的Kernel组合作为初始情况,并将Kernel组合卸载至硬件上进行执行;将Kernel组合的得分情况作为代价模型的初始代价,并通过梯度下降法对代价模型进行寻优以确定是否为极值;将Kernel组合对应的极值与历史记录进行比较确定是否更新调优记录,以确定对应的目标调优记录;根据目标调优记录切分深度学习模型解析的计算子图,并将切分后的深度学习模型卸载至对应的硬件设备进行推理,实现异构混合加速。本申请实现了模型推理的高效协同和性能最大化,提高了整体系统的计算效率和响应速度。



CN 118819869 A

1.一种基于多种加速卡的异构混合加速方法,其特征在于,所述方法包括:

将深度学习模型中每个算子与对应目标平台的Kernel组合作为初始情况,并将Kernel组合卸载至硬件上进行执行;

将所述Kernel组合的得分情况作为代价模型的初始代价,并通过梯度下降法,对所述代价模型进行寻优,以确定是否为极值;

将所述Kernel组合对应的极值与历史记录进行比较,确定是否更新调优记录,以确定对应的目标调优记录;

根据所述目标调优记录切分所述深度学习模型解析的计算子图,并将切分后的所述深度学习模型卸载至对应的硬件设备进行推理,实现异构混合加速。

2.根据权利要求1所述的一种基于多种加速卡的异构混合加速方法,其特征在于,将深度学习模型中每个算子与对应目标平台的Kernel组合作为初始情况之前,所述方法还包括:

接收输入的深度学习模型,并对所述深度学习模型的模型结构对应的计算图进行解析;

确定对应的算子调用关系以及每个算子对应的权重系数,并将所述算子调用关系和所述权重系数存储至哈希表中;

根据所述哈希表中的算子调用关系,对所述计算图进行切分,得到切分后的计算子图;

确定所述计算子图对应的算子,以及每个算子对应的属性标记信息,并根据属性标记信息,确定每个算子与对应目标平台的Kernel组合。

3.根据权利要求1所述的一种基于多种加速卡的异构混合加速方法,其特征在于,将所述Kernel组合的得分情况作为代价模型的初始代价,具体包括:

根据计算时间、硬件算力情况和数据传输时间,建立代价模型,并分别确定所述计算时间、所述硬件算力情况和所述数据传输时间对应的权重系数;

记录所述Kernel组合中每个Kernel的执行时间和得分情况,并将所述Kernel组合的得分情况作为所述代价模型的初始代价。

4.根据权利要求1所述的一种基于多种加速卡的异构混合加速方法,其特征在于,代价模型的具体计算方式为:

$$f(x)=\alpha x_1^a+\beta x_2^b+\gamma x_3^c$$

其中,x_1^a表示计算时间,a表示所述计算时间对应的幂次,x_2^b表示硬件算力情况,b表示所述硬件算力情况对应的幂次,x_3^c表示数据传输时间,c表示所述数据传输时间对应的幂次,α表示所述计算时间对应的权重系数,β表示所述硬件算力情况对应的权重系数,γ表示所述数据传输时间对应的权重系数。

5.根据权利要求1所述的一种基于多种加速卡的异构混合加速方法,其特征在于,通过梯度下降法,对所述代价模型进行寻优,具体包括:

基于所述代价模型的初始代价,计算所述代价模型对应的模型得分,并计算所述代价模型对应的梯度函数;

在梯度大于预设最小阈值的情况下,在预先构建的算子库中选取初始Kernel组合,并

按照梯度反方向,对所述初始Kernel组合进行反向迭代;

在梯度等于所述预设最小阈值的情况下,停止迭代,并确定代价模型得分的局部最小值。

6.根据权利要求5所述的一种基于多种加速卡的异构混合加速方法,其特征在于,将所述Kernel组合对应的极值与历史记录进行比较,确定是否更新调优记录,以确定对应的目标调优记录,具体包括:

获取历史缓存调优记录,并将所述代价模型得分的局部最小值与所述历史缓存调优记录进行比较;

在所述局部最小值小于所述历史缓存调优记录的情况下,对所述历史缓存调优记录进行更新,并确定对应的目标调优记录;

在所述局部最小值大于所述历史缓存调优记录的情况下,继续确定所述局部最小值是否满足当前阈值;

若否,则将所述代价模型的参数调整至大于所述局部最小值,若是,则确定所述历史缓存调优记录为目标调优记录。

7.根据权利要求1所述的一种基于多种加速卡的异构混合加速方法,其特征在于,根据所述目标调优记录切分所述深度学习模型解析的计算子图,并将切分后的所述深度学习模型卸载至对应的硬件设备进行推理,实现异构混合加速,具体包括:

根据所述目标调优记录中的目标策略,对所述深度学习模型解析的计算子图进行切分,并确定切分后每个算子对应的硬件设备;

将切分后的每个算子对应至相应的硬件设备,并将切分后每个算子的硬件信息存储至所述深度学习模型中;其中,所述硬件信息用于表示算子与硬件设备之间的算子调用关系;

接收待处理算子,并根据所述深度学习模型中硬件信息对应的算子调用关系,确定所述待处理算子对应的目标硬件设备;

将所述深度学习模型卸载至所述待处理算子对应的目标硬件设备,以对所述待处理算子进行推理,实现异构混合加速。

8.根据权利要求1所述的一种基于多种加速卡的异构混合加速方法,其特征在于,将输入模型中每个算子对应目标平台的Kernel组合作为初始情况之前,所述方法还包括:

对多种不同形态加速卡对应的接口进行统一,并对统一后的所有接口进行封装;

获取每个目标平台对应的属性标记信息,并将所述属性标记信息添加至对应的目标平台。

9.一种基于多种加速卡的异构混合加速设备,其特征在于,所述设备包括:

至少一个处理器;

以及,与所述至少一个处理器通信连接的存储器;

其中,所述存储器存储有可被所述至少一个处理器执行的指令,所述指令被所述至少一个处理器执行,以使所述至少一个处理器能够执行如权利要求1-8任一项所述的一种基于多种加速卡的异构混合加速方法。

10.一种非易失性计算机存储介质,存储有计算机可执行指令,其特征在于,所述计算机可执行指令被执行时,实现如权利要求1-8任一项所述的一种基于多种加速卡的异构混合加速方法。

一种基于多种加速卡的异构混合加速方法、设备及介质

技术领域

[0001] 本申请涉及并行计算技术领域,尤其涉及一种基于多种加速卡的异构混合加速方法、设备及介质。

背景技术

[0002] 目前,随着人工智能(AI)技术的飞速发展和广泛应用,从深度学习、机器学习到大数据处理,各个领域对高性能计算的需求急剧增长,这对算力行业提出了前所未有的挑战。AI算法的复杂性和数据量的爆炸性增长,推动了对更高效、更强大的计算能力的需求。然而,面对这一汹涌而至的需求浪潮,全球各大芯片制造商尽管在不断努力提升产能和优化设计,但仍难以满足市场对所有类型加速硬件的即时和全面供应。

[0003] 同时,在实际应用场景中,用户往往面临着多样化的计算任务需求,这些任务可能涉及图像处理、自然语言处理、科学计算等多个领域,每种任务对计算资源的要求各不相同。因此,用户环境中常常并存着多种不同形态和架构的加速计算卡,包括但不限于GPU(图形处理单元)、FPGA(现场可编程门阵列)、TPU(张量处理单元)以及各类专用AI加速芯片等。这些异构加速卡各有其独特的优势和适用场景,但同时也带来了资源整合与协同工作的复杂性。

[0004] 针对上述背景,如何在有限的硬件资源和多样化的用户需求之间找到平衡点,实现高效、灵活的算力部署,成为了亟待解决的问题。传统的单一类型加速卡方案往往无法充分利用现有硬件资源,导致算力浪费或性能瓶颈。

发明内容

[0005] 本申请实施例提供了一种基于多种加速卡的异构混合加速方法、设备及介质,用以解决上述技术问题。

[0006] 一方面,本申请实施例提供了一种基于多种加速卡的异构混合加速方法,包括:

将深度学习模型中每个算子与对应目标平台的Kernel组合作为初始情况,并将Kernel组合卸载至硬件上进行执行;

将所述Kernel组合的得分情况作为代价模型的初始代价,并通过梯度下降法,对所述代价模型进行寻优,以确定是否为极值;

将所述Kernel组合对应的极值与历史记录进行比较,确定是否更新调优记录,以确定对应的目标调优记录;

根据所述目标调优记录切分所述深度学习模型解析的计算子图,并将切分后的所述深度学习模型卸载至对应的硬件设备进行推理,实现异构混合加速。

[0007] 在本申请的一种实现方式中,将深度学习模型中每个算子与对应目标平台的Kernel组合作为初始情况之前,所述方法还包括:

接收输入的深度学习模型,并对所述深度学习模型的模型结构对应的计算图进行解析;

确定对应的算子调用关系以及每个算子对应的权重系数,并将所述算子调用关系和所述权重系数存储至哈希表中;

根据所述哈希表中的算子调用关系,对所述计算图进行切分,得到切分后的计算子图;

确定所述计算子图对应的算子,以及每个算子对应的属性标记信息,并根据属性标记信息,确定每个算子与对应目标平台的Kernel组合。

[0008] 在本申请的一种实现方式中,将所述Kernel组合的得分情况作为代价模型的初始代价,具体包括:

根据计算时间、硬件算力情况和数据传输时间,建立代价模型,并分别确定所述计算时间、所述硬件算力情况和所述数据传输时间对应的权重系数;

记录所述Kernel组合中每个Kernel的执行时间和得分情况,并将所述Kernel组合的得分情况作为所述代价模型的初始代价。

[0009] 在本申请的一种实现方式中,代价模型的具体计算方式为:

$$f(x) = \alpha x_1^a + \beta x_2^b + \gamma x_3^c$$

[0010] 其中,x_1^a表示计算时间,a表示所述计算时间对应的幂次,x_2^b表示硬件算力情况,b表示所述硬件算力情况对应的幂次,x_3^c表示数据传输时间,c表示所述数据传输时间对应的幂次,α表示所述计算时间对应的权重系数,β表示所述硬件算力情况对应的权重系数,γ表示所述数据传输时间对应的权重系数。

[0011] 在本申请的一种实现方式中,通过梯度下降法,对所述代价模型进行寻优,具体包括:

基于所述代价模型的初始代价,计算所述代价模型对应的模型得分,并计算所述代价模型对应的梯度函数;

在梯度大于预设最小阈值的情况下,在预先构建的算子库中选取初始Kernel组合,并按照梯度反方向,对所述初始Kernel组合进行反向迭代;

在梯度等于所述预设最小阈值的情况下,停止迭代,并确定代价模型得分的局部最小值。

[0012] 在本申请的一种实现方式中,将所述Kernel组合对应的极值与历史记录进行比较,确定是否更新调优记录,以确定对应的目标调优记录,具体包括:

获取历史缓存调优记录,并将所述代价模型得分的局部最小值与所述历史缓存调优记录进行比较;

在所述局部最小值小于所述历史缓存调优记录的情况下,对所述历史缓存调优记录进行更新,并确定对应的目标调优记录;

在所述局部最小值大于所述历史缓存调优记录的情况下,继续确定所述局部最小值是否满足当前阈值;

若否,则将所述代价模型的参数调整至大于所述局部最小值,若是,则确定所述历史缓存调优记录为目标调优记录。

[0013] 在本申请的一种实现方式中,根据所述目标调优记录切分所述深度学习模型解析

的计算子图,并将切分后的所述深度学习模型卸载至对应的硬件设备进行推理,实现异构混合加速,具体包括:

根据所述目标调优记录中的目标策略,对所述深度学习模型解析的计算子图进行切分,并确定切分后每个算子对应的硬件设备;

将切分后的每个算子对应至相应的硬件设备,并将切分后每个算子的硬件信息存储至所述深度学习模型中;其中,所述硬件信息用于表示算子与硬件设备之间的算子调用关系;

接收待处理算子,并根据所述深度学习模型中硬件信息对应的算子调用关系,确定所述待处理算子对应的目标硬件设备;

将所述深度学习模型卸载至所述待处理算子对应的目标硬件设备,以对所述待处理算子进行推理,实现异构混合加速。

[0014] 在本申请的一种实现方式中,将输入模型中每个算子对应目标平台的Kernel组合作为初始情况之前,所述方法还包括:

对多种不同形态加速卡对应的接口进行统一,并对统一后的所有接口进行封装;

获取每个目标平台对应的属性标记信息,并将所述属性标记信息添加至对应的目标平台。

[0015] 另一方面,本申请实施例还提供了一种基于多种加速卡的异构混合加速设备,所述设备包括:

至少一个处理器;

以及,与所述至少一个处理器通信连接的存储器;

其中,所述存储器存储有可被所述至少一个处理器执行的指令,所述指令被所述至少一个处理器执行,以使所述至少一个处理器能够执行如上述的一种基于多种加速卡的异构混合加速方法。

[0016] 另一方面,本申请实施例还提供了一种非易失性计算机存储介质,存储有计算机可执行指令,所述计算机可执行指令被执行时,实现如上述的一种基于多种加速卡的异构混合加速方法。

[0017] 本申请实施例提供了一种基于多种加速卡的异构混合加速方法、设备及介质,至少包括以下有益效果:

通过确定算子与硬件Kernel之间的绑定情况,能够直接获取算子在特定硬件上的实际性能数据,为后续的优化提供准确的基准;通过代价模型和梯度下降法进行优化,能够高效地搜索出最优的算子与硬件组合策略,使得深度学习模型在异构硬件环境下的性能得到最大化提升;通过持续比较和更新调优记录,确保了每次优化都能基于历史最优结果进行,从而避免了重复劳动和可能的性能倒退,保证了优化过程的持续性和有效性;通过基于调优记录的子图切分和异构部署,能够充分利用不同硬件设备的优势,实现模型推理的高效协同和性能最大化,提高了整体系统的计算效率和响应速度,实现了高效的异构混合加速,从而在实际应用中提供更快的响应速度和更好的用户体验。

附图说明

[0018] 此处所说明的附图用来提供对本申请的进一步理解,构成本申请的一部分,本申

请的示意性实施例及其说明用于解释本申请,并不构成对本申请的不当限定。在附图中:

图1为本申请实施例提供的一种基于多种加速卡的异构混合加速方法的流程示意图;

图2为本申请实施例提供的一种基于多种加速卡的异构混合加速设备的内部结构示意图。

具体实施方式

[0019] 为使本申请的目的、技术方案和优点更加清楚,下面将结合本申请具体实施例及相应的附图对本申请技术方案进行清楚、完整地描述。显然,所描述的实施例仅是本申请一部分实施例,而不是全部的实施例。基于本申请中的实施例,本领域普通技术人员在没有做出创造性劳动前提下所获得的所有其他实施例,都属于本申请保护的范围。

[0020] 以下结合附图,详细说明本申请各实施例提供的技术方案。

[0021] 图1为本申请实施例提供的一种基于多种加速卡的异构混合加速方法的流程示意图。

[0022] 本申请实施例涉及的分析方法的实现可以为终端设备,也可以为服务器,本申请对此不作特殊限制。为了方便理解和描述,以下实施例均以服务器为例进行详细描述。

[0023] 需要说明的是,该服务器可以是单独的一台设备,可以是有多台设备组成的系统,即,分布式服务器,本申请对此不做具体限定。

[0024] 如图1所示,本申请实施例提供的一种基于多种加速卡的异构混合加速方法,包括:

101、将深度学习模型中每个算子与对应目标平台的Kernel组合作为初始情况,并将Kernel组合卸载至硬件上进行执行。

[0025] 本申请公开了一种基于多种加速卡的异构混合加速方法,主要基于现有的FPGA(Xilinx U280)、GPGPU(A100、Ascend 910)、NPU(MLU220)计算加速卡实现,基于各个硬件平台实现运行任务的算子,构建整个平台系统的算子库。

[0026] 在本申请的一个实施例中,在将深度学习模型中每个算子与对应目标平台的Kernel组合作为初始情况之前,包括:

接收输入的深度学习模型,并对深度学习模型的模型结构对应的计算图进行解析;

确定对应的算子调用关系以及每个算子对应的权重系数,并将算子调用关系和权重系数存储至哈希表中;

根据哈希表中的算子调用关系,对计算图进行切分,得到切分后的计算子图;

确定计算子图对应的算子,以及每个算子对应的属性标记信息,并根据属性标记信息,确定每个算子与对应目标平台的Kernel组合。

[0027] 在一个实施例中,假设接收到了一个基于TensorFlow框架的深度学习模型,该模型用于图像分类任务。模型的结构定义为一个计算图,其中包含了多个层和节点,每个节点代表一个操作或算子,如卷积、全连接、激活函数等。

[0028] 加载模型文件,使用TensorFlow框架提供的API接口读取模型,如tf.keras.models.load_model。使用TensorFlow的图工具获取模型的计算图表示,如

tf.Graph和tf.Session,或者在TensorFlow 2.x中使用tf.function和tf.Graph的兼容模式。对计算图进行解析,从而识别出所有的节点及其相互连接关系。

[0029] 对于每个节点,记录其算子类型,如Conv2D、MatMul、ReLU等。确定每个算子的输入和输出张量,以及与其他算子的调用关系,并提取每个算子的权重系数,例如卷积核和全连接层的权重矩阵。将这些信息存储在一个哈希表中,哈希表的键是算子的唯一标识符,可以是节点的名称或ID,值是一个包含算子类型、调用关系和权重系数的结构体。

[0030] 为了优化计算或进行并行处理,将计算图切分成多个计算子图。切分的策略可以根据具体的需求,如最小化子图之间的数据传输、平衡计算负载等。例如,可以将计算图按照卷积层和全连接层进行切分,每个卷积层及其后的激活函数作为一个计算子图。使用哈希表中的调用关系,识别出这些节点的边界,并创建新的子图对象。

[0031] 对于每个计算子图,进一步分析其包含的算子,并收集每个算子的属性标记信息,如数据类型、张量形状、步长、填充方式等。假设目标平台是一个GPU,需要为每个算子选择最优的GPU Kernel。这通常涉及查找与算子属性和目标平台相匹配的Kernel实现。例如,对于Conv2D算子,可能需要根据其步长、卷积核大小、输入和输出张量的数据类型等信息,在GPU的Kernel库中选择最合适的实现。最终,将这些Kernel组合与计算子图一起,形成一个可以在目标平台上高效执行的优化后的模型表示。

[0032] 在本申请的一个实施例中,由于不同的异构加速卡采用的接口不同,给上层的集成和统一带来了巨大的挑战,因此在将输入模型中每个算子对应目标平台的Kernel组合作为初始情况之前,本申请对多种不同形态加速卡对应的接口进行统一,并对统一后的所有接口进行封装;获取每个目标平台对应的属性标记信息,并将属性标记信息添加至对应的目标平台。

[0033] 在一个实施例中,为了实现对多种不同形态加速卡对应的接口进行统一,并对统一后的所有接口进行封装,同时获取每个目标平台对应的属性标记信息,并将该属性标记信息添加至对应的目标平台,首先,系统通过预设的识别模块,自动检测并识别当前系统中连接的各种加速卡,包括但不限于GPU、FPGA、ASIC等不同类型的加速卡。

[0034] 其次,根据识别出的加速卡类型,设计一套统一的接口规范。这套规范涵盖了加速卡的基本功能,如数据传输、计算任务提交、状态查询等。之后,对于每种类型的加速卡,开发相应的接口适配器。这些适配器负责将加速卡的原生接口转换为统一接口规范下的接口,从而实现接口的统一。然后,将统一后的接口进行封装,形成一套易于使用的API库。这套API库可以屏蔽底层加速卡的差异,使得上层应用无需关心底层具体使用的是哪种加速卡。

[0035] 为每个目标平台定义一组属性标记信息,包括但不限于平台的硬件配置、操作系统版本、已安装的加速卡类型及数量等,目标平台如不同的服务器、工作站等。通过系统查询、配置文件读取等方式,自动获取每个目标平台的属性标记信息。例如,可以通过查询系统的硬件配置文件来获取硬件信息,通过读取操作系统的版本信息来获取操作系统版本等。

[0036] 将获取到的属性标记信息添加至对应的目标平台。通过在系统中创建或更新一个配置文件来实现,该配置文件记录了每个目标平台及其对应的属性标记信息。在上层应用需要调用加速卡时,可以先查询目标平台的属性标记信息,以确定该平台支持哪些类型的

加速卡及加速卡的数量等信息。然后,根据这些信息选择合适的加速卡,并通过统一接口调用其功能。

[0037] 假设有一个包含GPU和FPGA两种加速卡的服务器,系统首先通过识别模块检测出这两种加速卡,并为它们分别开发接口适配器。然后,将这些适配器的接口统一封装为一套API库。同时,系统获取该服务器的属性标记信息,如硬件配置(包含GPU和FPGA)、操作系统版本等,并将这些信息添加至服务器的配置文件中。当上层应用需要调用加速卡进行计算时,它可以先查询服务器的属性标记信息,确定服务器上有哪些加速卡可用,然后通过统一接口调用这些加速卡的功能。

[0038] 102、将Kernel组合的得分情况作为代价模型的初始代价,并通过梯度下降法,对代价模型进行寻优,以确定是否为极值。

[0039] 具体地,在本申请的一个实施例中,将Kernel组合的得分情况作为代价模型的初始代价,具体包括:

根据计算时间、硬件算力情况和数据传输时间,建立代价模型,并分别确定计算时间、硬件算力情况和数据传输时间对应的权重系数;

记录Kernel组合中每个Kernel的执行时间和得分情况,并将Kernel组合的得分情况作为代价模型的初始代价。

[0040] 在一个实施例中,在深度学习模型优化和部署过程中,需要综合考虑多个因素以评估不同Kernel组合在特定硬件上的性能。为此,建立一个代价模型,该模型包含三个主要组成部分:计算时间、硬件算力情况和数据传输时间。

[0041] 需要说明的是,本申请实施例中的计算时间指的是执行特定Kernel所需的时间,通常与Kernel的复杂度、输入数据的大小以及硬件的浮点运算能力有关。硬件算力情况指的是反映硬件执行特定类型计算的效率,如浮点运算、整数运算、矩阵乘法等,可以用每秒能执行的操作数(OPS)来衡量。数据传输时间指的是在硬件之间或硬件与内存之间传输数据所需的时间,这取决于数据的量、传输带宽以及传输协议。

[0042] 为了将这三个因素综合起来,为它们分别分配权重系数,这些系数反映了在特定应用场景下各因素的重要性。例如,在实时性要求较高的场景中,计算时间的权重可能较高;而在数据密集型应用中,数据传输时间的权重可能更为关键。

[0043] 假设通过专家评估、实验测量或历史数据分析,确定了以下权重系数:计算时间权重:0.5,硬件算力情况权重:0.3,数据传输时间权重:0.2。代价模型可以表示为这些因素的加权和:代价=计算时间*0.5+硬件算力情况*0.3+数据传输时间*0.2。

[0044] 在确定了代价模型后,需要对不同的Kernel组合进行评估。为此,首先记录每个Kernel在目标硬件上的执行时间和得分情况。将该组Kernel全部卸载到硬件设备上执行,并记录每个Kernel的执行时间,然后采用随机梯度下降的方式在算子库中再挑选一组对应的Kernel,将该组Kernel再卸载到硬件设备上执行,并记录对应的得分情况。

[0045] 在本申请的一个实施例中,代价模型的具体计算方式为:

$$f(x) = \alpha x_1^a + \beta x_2^b + \gamma x_3^c$$

[0046] 需要说明的是,本申请实施例中的 x_1^a 表示计算时间,a 表示计算时间对应的幂

次，x_2^b表示硬件算力情况，b表示硬件算力情况对应的幂次，x_3^c表示数据传输时间，c表示数据传输时间对应的幂次，α表示计算时间对应的权重系数，β表示硬件算力情况对应的权重系数，γ表示数据传输时间对应的权重系数。本申请中的权重系数可以根据实际情况设置，并且满足计算时间对应的权重系数、硬件算力情况对应的权重系数以及数据传输时间对应的权重系数之和为1，即$\alpha+\beta+\gamma=1$。

[0047] 在一个实施例中，代价模型对应的幂次$1<a<2$，硬件算力情况对应的幂次$1<b<2$，以及数据传输时间对应的幂次$1<c<2$，幂次可以根据实际情况进行选择，本申请对此不做具体限定。例如，$a=1.2$，$b=1.5$，$c=1.7$，则代价模型对应为$f(x)=\alpha x_1^{1.2}+\beta x_2^{1.5}+\gamma x_3^{1.7}$。

[0048] 在本申请的一个实施例中，通过梯度下降法，对代价模型进行寻优，具体包括：

基于代价模型的初始代价，计算代价模型对应的模型得分，并计算代价模型对应的梯度函数；

在梯度大于预设最小阈值的情况下，在预先构建的算子库中选取初始Kernel组合，并按照梯度反方向，对初始Kernel组合进行反向迭代；

在梯度等于预设最小阈值的情况下，停止迭代，并确定代价模型得分的局部最小值。

[0049] 在一个实施例中，采用梯度下降算法对代价模型进行寻优，具体步骤如下：选择一组初始的Kernel，并经过实际运行得到x_i参数，将该组参数作为初始值，计算代价模型得分f(x)，以及计算梯度函数$\nabla f(x)$。当梯度足够小时，停止迭代，令$x_i^*=x_i^t$，此时便找到了一组局部最小值。需要说明的是，本申请实施例中x_i^*表示一个中间值，i可以取1,2,3，x_1^*、x_2^*、x_3^*分别代表对于计算时间、硬件算力、传输时间的临时记录值，x_1^t、x_2^t、x_3^t分别表示在t时刻，计算时间、硬件算力、传输时间对应的当前的一组数据记录。

[0050] 否则，根据公式$f(x_i^t-\lambda_t\nabla f(x_i^t))=\min_{\lambda>=0}f(x_i^t-\lambda_t\nabla f(x_i^t))$求取$\lambda$。具体地，令$x_i^{t+1}=x_i^t-\lambda_t\nabla f(x_i^t)$按照梯度反方向迭代，即t+1时刻是根据t时刻的x和梯度进行求取的。

[0051] 在一个实施例中，假设正在开发一个机器学习模型优化系统，该系统旨在通过调整模型参数来最小化一个特定的代价函数，如均方误差、交叉熵损失等。首先，定义了一个代价模型，该模型基于当前模型参数计算出初始代价。例如，在一个线性回归问题中，代价模型可以是预测值与真实值之间差的平方和。接着，根据初始代价，已经预先定义的评分机制，如准确率、F1分数或代价本身的负值，计算代价模型对应的模型得分。假设评分机制是简单地取代价的负值作为得分，即得分=-初始代价。使用微积分中的偏导数方法，计算代价模型相对于每个模型参数的梯度，形成梯度向量。

[0052] 接下来，检查梯度向量中的每个元素，判断其绝对值是否大于预设的最小阈值。这

个阈值是用来决定何时停止迭代,避免在数值上接近零但理论上不为零的梯度上浪费计算资源。如果梯度大于预设最小阈值,从预先构建的算子库中选择一组初始Kernel组合。这些Kernel是预先优化过的基本计算单元,如卷积核、激活函数等,用于加速梯度下降过程中的参数更新。

[0053] 之后,使用选定的初始Kernel组合,按照梯度反方向(即负梯度方向)进行反向迭代,更新模型参数,使得下一次迭代的代价函数值预期会降低。迭代过程中,每一步都会重新计算梯度,并动态调整Kernel组合以适应当前梯度状态,确保优化过程的高效性和稳定性。

[0054] 当所有梯度的绝对值均等于或小于预设的最小阈值时,迭代停止。此时,认为代价模型得分达到了一个局部最小值,进一步的参数调整无法继续带来显著的代价减少。记录下此时的模型参数配置和对应的代价模型得分,作为优化过程的结果。

[0055] 103、将Kernel组合对应的极值与历史记录进行比较,确定是否更新调优记录,以确定对应的目标调优记录。

[0056] 具体地,在本申请的一个实施例中,将Kernel组合对应的极值与历史记录进行比较,确定是否更新调优记录,以确定对应的目标调优记录,具体包括:

获取历史缓存调优记录,并将代价模型得分的局部最小值与历史缓存调优记录进行比较;

在局部最小值小于历史缓存调优记录的情况下,对历史缓存调优记录进行更新,并确定对应的目标调优记录;

在局部最小值大于历史缓存调优记录的情况下,继续确定局部最小值是否满足当前阈值;

若否,则将代价模型的参数调整至大于局部最小值,若是,则确定历史缓存调优记录为目标调优记录。

[0057] 在一个实施例中,在一个旨在优化深度学习模型性能的场景中,实施了一套自动化的调优流程,该流程包括代价模型得分的计算、与历史缓存调优记录的比较,以及基于比较结果的决策制定。

[0058] 首先从一个专门的数据库或缓存系统中检索历史调优记录。这些记录包含了之前调优过程中代价模型得分的最小值、对应的模型参数配置、以及调优的时间戳等信息。在当前调优迭代中,已经通过代价模型计算得到了一个新的局部最小值。接下来,将这个局部最小值与历史缓存调优记录中的最小值进行比较。

[0059] 如果当前局部最小值小于历史缓存调优记录中的最小值,说明找到了一个更优的解。此时,更新历史缓存调优记录,将当前局部最小值、对应的模型参数配置以及当前时间戳等信息存入数据库。同时,确定当前记录为目标调优记录。

[0060] 如果当前局部最小值大于历史缓存调优记录中的最小值,并不立即放弃当前解,而是进一步评估其是否满足当前设定的阈值。这个阈值可能是基于业务需求、计算资源限制或调优策略动态确定的。

[0061] 若当前局部最小值不满足当前阈值(即比阈值大),则认为当前解还不够优,需要继续调优。此时,可以采取策略,如调整代价模型的参数,使其跳出当前局部最小值,探索其他可能的解空间。若当前局部最小值满足当前阈值(即比阈值小或等于阈值),虽然它不是

历史最优,但在当前条件下已经足够好。此时,确定历史缓存调优记录(即之前的最优解)为目标调优记录,并可能结束当前调优迭代,或根据业务需求进入下一轮调优。

[0062] 104、根据目标调优记录切分深度学习模型解析的计算子图,并将切分后的深度学习模型卸载至对应的硬件设备进行推理,实现异构混合加速。

[0063] 具体地,在本申请的一个实施例中,根据目标调优记录切分深度学习模型解析的计算子图,并将切分后的深度学习模型卸载至对应的硬件设备进行推理,实现异构混合加速,具体包括:

根据目标调优记录中的目标策略,对深度学习模型解析的计算子图进行切分,并确定切分后每个算子对应的硬件设备;

将切分后的每个算子对应至相应的硬件设备,并将切分后每个算子的硬件信息存储至深度学习模型中;其中,硬件信息用于表示算子与硬件设备之间的算子调用关系;

接收待处理算子,并根据深度学习模型中硬件信息对应的算子调用关系,确定待处理算子对应的目标硬件设备;

将深度学习模型卸载至待处理算子对应的目标硬件设备,以对待处理算子进行推理,实现异构混合加速。

[0064] 在一个实施例中,服务器根据目标调优记录中通过梯度下降法找到的最优解,并按照该组最优解调整算子设备的对应关系,即对计算子图进行切分,以确保切分后的算子与对应的硬件设备之间的对应关系为最优解。

[0065] 在一个实施例中,在一个深度学习模型优化与部署的场景中,旨在通过合理利用异构硬件资源(如CPU、GPU、FPGA等)来实现模型的高效推理。首先根据目标调优记录中的目标策略,对深度学习模型解析后的计算子图进行切分。这个切分过程是基于模型的计算结构和依赖关系,将模型拆分成多个可以独立执行的算子。在切分过程中,考虑算子的计算量、数据依赖、以及硬件设备的特性,以确保切分后的算子能够高效地映射到硬件设备上。

[0066] 对于切分后的每个算子,根据其计算特性和硬件设备的性能特点,确定其对应的目标硬件设备。例如,计算密集型的算子可能被分配到GPU上,而数据预处理或简单的逻辑运算可能被分配到CPU上。还考虑硬件设备之间的数据传输开销,尽量减少跨设备的数据移动,以提高整体推理效率。

[0067] 将切分后每个算子与其对应的硬件设备信息存储到深度学习模型中,硬件信息不仅包括了算子与硬件设备的映射关系,还可能包括设备上的具体执行单元(如GPU的某个核心)、内存地址等详细信息。硬件信息用于表示算子与硬件设备之间的算子调用关系,是后续模型推理时调度算子的重要依据。

[0068] 在模型推理阶段,接收待处理的算子。根据深度学习模型中存储的硬件信息,可以快速确定待处理算子对应的目标硬件设备。这个过程是通过查找硬件信息表或调用相应的API来实现的,确保了算子能够准确地调度到正确的硬件设备上。

[0069] 将深度学习模型(或模型的部分)卸载到待处理算子对应的目标硬件设备上。这通常涉及到将模型的权重、计算图、以及必要的运行时环境传输到目标设备上。在目标设备上,利用相应的推理引擎或库来执行待处理算子,实现深度学习模型的推理。由于算子已经根据硬件设备进行了优化和调度,因此可以实现异构混合加速,提高推理效率。不同类型的算子被合理地分配到不同的硬件设备上执行,充分利用了各硬件设备的优势,提高了模型

的整体推理性能,实现了深度学习模型在异构硬件环境下的混合加速。

[0070] 以上为本申请提出的方法实施例。基于同样的发明构思,本申请实施例还提供了一种基于多种加速卡的异构混合加速设备,其结构如图2所示。

[0071] 图2为本申请实施例提供的一种基于多种加速卡的异构混合加速设备的内部结构示意图。如图2所示,设备包括:

至少一个处理器;

以及,与至少一个处理器通信连接的存储器;

其中,存储器存储有可被至少一个处理器执行的指令,指令被至少一个处理器执行,以使至少一个处理器能够:

将深度学习模型中每个算子与对应目标平台的Kernel组合作为初始情况,并将Kernel组合卸载至硬件上进行执行;

将Kernel组合的得分情况作为代价模型的初始代价,并通过梯度下降法,对代价模型进行寻优,以确定是否为极值;

将Kernel组合对应的极值与历史记录进行比较,确定是否更新调优记录,以确定对应的目标调优记录;

根据目标调优记录切分深度学习模型解析的计算子图,并将切分后的深度学习模型卸载至对应的硬件设备进行推理,实现异构混合加速。

[0072] 本申请实施例还提供了一种非易失性计算机存储介质,存储有计算机可执行指令,计算机可执行指令被执行时能够:

将深度学习模型中每个算子与对应目标平台的Kernel组合作为初始情况,并将Kernel组合卸载至硬件上进行执行;

将Kernel组合的得分情况作为代价模型的初始代价,并通过梯度下降法,对代价模型进行寻优,以确定是否为极值;

将Kernel组合对应的极值与历史记录进行比较,确定是否更新调优记录,以确定对应的目标调优记录;

根据目标调优记录切分深度学习模型解析的计算子图,并将切分后的深度学习模型卸载至对应的硬件设备进行推理,实现异构混合加速。

[0073] 本申请中的各个实施例均采用递进的方式描述,各个实施例之间相同相似的部分互相参见即可,每个实施例重点说明的都是与其他实施例的不同之处。尤其,对于设备和介质实施例而言,由于其基本相似于方法实施例,所以描述的比较简单,相关之处参见方法实施例的部分说明即可。

[0074] 本申请实施例提供的设备和介质与方法是一一对应的,因此,设备和介质也具有与其对应的方法类似的有益技术效果,由于上面已经对方法的有益技术效果进行了详细说明,因此,这里不再赘述设备和介质的有益技术效果。

[0075] 本领域内的技术人员应明白,本申请的实施例可提供为方法、系统、或计算机程序产品。因此,本申请可采用完全硬件实施例、完全软件实施例、或结合软件和硬件方面的实施例的形式。而且,本申请可采用在一个或多个其中包含有计算机可用程序代码的计算机可用存储介质(包括但不限于磁盘存储器、CD-ROM、光学存储器等)上实施的计算机程序产品的形式。

[0076] 本申请是参照根据本申请实施例的方法、设备(系统)、和计算机程序产品的流程图和/或方框图来描述的。应理解可由计算机程序指令实现流程图和/或方框图中的每一流程和/或方框、以及流程图和/或方框图中的流程和/或方框的结合。可提供这些计算机程序指令到通用计算机、专用计算机、嵌入式处理机或其他可编程数据处理设备的处理器以产生一个机器,使得通过计算机或其他可编程数据处理设备的处理器执行的指令产生用于实现在流程图一个流程或多个流程和/或方框图一个方框或多个方框中指定的功能的装置。

[0077] 这些计算机程序指令也可存储在能引导计算机或其他可编程数据处理设备以特定方式工作的计算机可读存储器中,使得存储在该计算机可读存储器中的指令产生包括指令装置的制造品,该指令装置实现在流程图一个流程或多个流程和/或方框图一个方框或多个方框中指定的功能。

[0078] 这些计算机程序指令也可装载到计算机或其他可编程数据处理设备上,使得在计算机或其他可编程设备上执行一系列操作步骤以产生计算机实现的处理,从而在计算机或其他可编程设备上执行的指令提供用于实现在流程图一个流程或多个流程和/或方框图一个方框或多个方框中指定的功能的步骤。

[0079] 在一个典型的配置中,计算设备包括一个或多个处理器 (CPU)、输入/输出接口、网络接口和内存。

[0080] 内存可能包括计算机可读介质中的非永久性存储器,随机存取存储器 (RAM) 和/或非易失性内存等形式,如只读存储器 (ROM) 或闪存(flash RAM)。内存是计算机可读介质的示例。

[0081] 计算机可读介质包括永久性和非永久性、可移动和非可移动媒体可以由任何方法或技术来实现信息存储。信息可以是计算机可读指令、数据结构、程序的模块或其他数据。计算机的存储介质的例子包括,但不限于相变内存 (PRAM)、静态随机存取存储器 (SRAM)、动态随机存取存储器 (DRAM)、其他类型的随机存取存储器 (RAM)、只读存储器 (ROM)、电可擦除可编程只读存储器 (EEPROM)、快闪记忆体或其他内存技术、只读光盘只读存储器 (CD-ROM)、数字多功能光盘 (DVD) 或其他光学存储、磁盒式磁带,磁带磁盘存储或其他磁性存储设备或任何其他非传输介质,可用于存储可以被计算设备访问的信息。按照本文中的界定,计算机可读介质不包括暂存电脑可读媒体 (transitory media),如调制的数据信号和载波。

[0082] 还需要说明的是,术语“包括”、“包含”或者其任何其他变体意在涵盖非排他性的包含,从而使得包括一系列要素的过程、方法、商品或者设备不仅包括那些要素,而且还包括没有明确列出的其他要素,或者是还包括为这种过程、方法、商品或者设备所固有的要素。在没有更多限制的情况下,由语句“包括一个……”限定的要素,并不排除在包括所述要素的过程、方法、商品或者设备中还存在另外的相同要素。

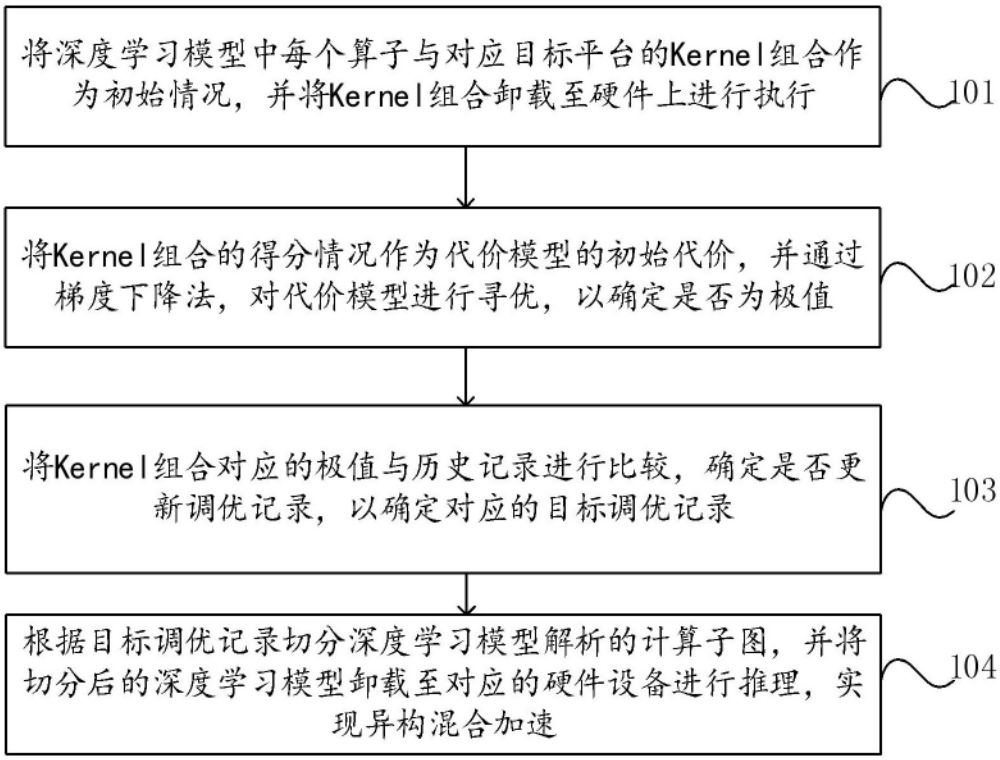

将深度学习模型中每个算子与对应目标平台的Kernel组合作为初始情况，并将Kernel组合卸载至硬件上进行执行
101
将Kernel组合的得分情况作为代价模型的初始代价，并通过梯度下降法，对代价模型进行寻优，以确定是否为极值
102
将Kernel组合对应的极值与历史记录进行比较，确定是否更新调优记录，以确定对应的目标调优记录
103
根据目标调优记录切分深度学习模型解析的计算子图，并将切分后的深度学习模型卸载至对应的硬件设备进行推理，实现异构混合加速
104

图1

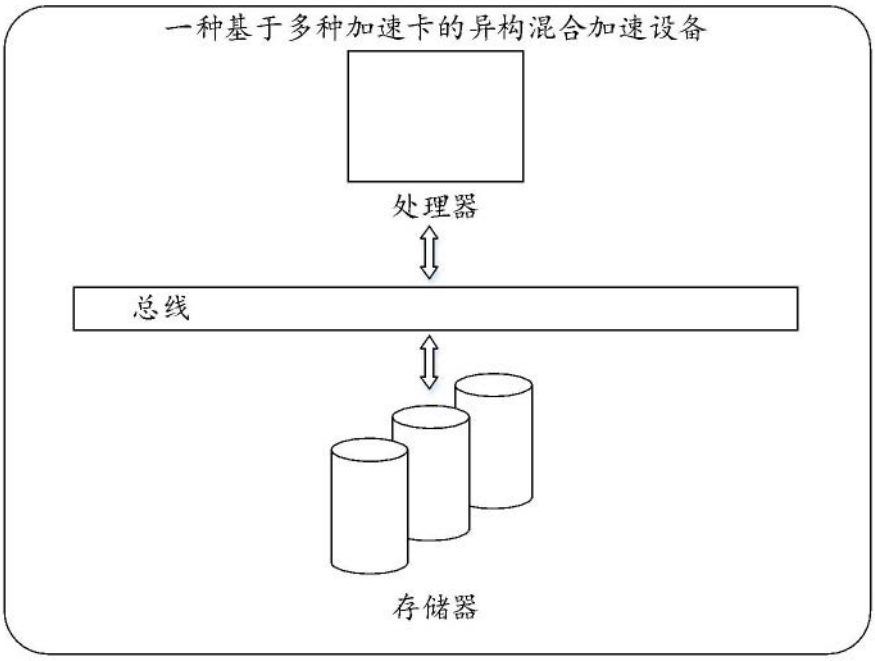

一种基于多种加速卡的异构混合加速设备
处理器
总线
存储器

图2