一种模型训练方法和电子设备

本申请要求于 2024 年 03 月 26 日提交中国国家知识产权局、申请号为 2024103536067、申请名称为“一种训练集群检查点存储、恢复方法和电子设备”的中国专利申请的优先权，其全部内容通过引用结合在本申请中。

以及，本申请要求于 2024 年 06 月 28 日提交中国国家知识产权局、申请号为 2024108539748、申请名称为“一种模型训练方法和电子设备”的中国专利申请的优先权，其全部内容通过引用结合在本申请中。

技术领域

本申请涉及计算机技术领域，特别涉及一种模型训练方法和电子设备。

背景技术

人工智能（Artificial Intelligence，AI）大模型具备的广阔潜力，使得各厂商都纷纷转向大模型的研究和使用。训练具有千亿参数量级的 AI 大模型往往需要使用成千上万张图形处理器（graphics processing unit，GPU）卡进行分布式集群训练，在这样的场景中，频繁发生的 GPU 卡故障会丢失正在训练的模型数据。模型数据丢失，就会导致模型训练的中断，需要重新进行模型训练，从而导致高昂成本的 GPU 卡算力浪费。因此，需要一种方法，在模型数据丢失时恢复模型训练状态，以避免 GPU 卡故障导致的 GPU 卡算力浪费，从而提高模型训练效率。

发明内容

针对如何避免 GPU 卡故障导致中断模型训练并浪费 GPU 算力的问题，本申请提供了一种模型训练方法、装置和计算设备集群，本申请还提供一种计算机程序产品和计算机可读存储介质。

本申请实施例采用下述技术方案：

第一方面，本申请提供一种模型训练方法，所述方法应用于训练集群，所述训练集群包括多个模型训练组，所述多个模型训练组包括第一模型训练组和第二模型训练组，所述第一模型训练组和所述第二模型训练组用于以数据并行的方式并行训练神经网络模型，所述第一模型训练组以及所述第二模型训练组包含相同的节点结构，所述第一模型训练组包含第一训练节点，所述第二模型训练组包含第二训练节点，其中，基于所述节点结构，所述第一训练节点对应所述第二训练节点，所述方法包括：

在一轮训练结束后生成检查点数据，其中，所述检查点数据为所述第一训练节点以及所述第二训练节点的检查点数据；

所述第一训练节点将所述检查点数据保存到所述第一训练节点的内存中，以及，所述第二训练节点将所述检查点数据保存到所述第二训练节点的内存中；

若所述第一训练节点在对模型进行训练过程中发生故障，则所述第一训练节点从所述第二训练节点的内存中获取所述检查点数据，以对所述第一训练节点进行数据恢复。

根据第一方面的模型训练方法，使用训练节点自身的内存备份该训练节点的检查点数据，训练节点的检查点数据备份过程中，训练节点不需要相互间进行通信，大大减少了通信资源的消耗。

根据第一方面的模型训练方法，使用训练节点自身的内存备份该训练节点的检查点数据，并且，利用多模型组冗余备份的特性对故障训练节点的检查点数据进行重传和恢复，检查点数据备份以及恢复过程不需要远端存储服务参与，远端存储服务被移出关键路径，减少了对远端存储的性能和带宽需求，显著降低了整体系统的成本。

在第一方面的一种实现方式中，所述方法还包括：

针对未发生故障的所述第二训练节点，使用所述第二训练节点自身的内存中保存的检查点数据对所述第二训练节点进行数据恢复。

在第一方面的一种实现方式中，所述在一轮训练结束后生成检查点数据，包括：

在一轮训练结束时，将所述训练集群中所有训练节点的模型参数进行平均值计算，以计算结果作为该轮训练的检查点数据。

在第一方面的一种实现方式中，所述方法还包括：

使用所述第一训练节点下载初始检查点数据；

由所述第一训练节点向所述第二训练节点发送所述初始检查点数据。

根据上述实现方式的方法，下载初始检查点数据到一个模型训练组，并通过一个模型训练组向其他模型训练组发送初始检查点数据，减少了初始检查点数据时的带宽需求，提高了初始检查点数据的下载效率。

在第一方面的一种实现方式中，所述方法还包括：

使用所述第一模型训练组上传所述第一模型训练组中各个训练节点的内存中所保存的检查点数据到远端存储服务。

根据上述实现方式，采用将远端存储移出关键路径的方案，集群内存数据异步持久化到远端，减少对远端性能的依赖；并且，根据本申请一实施例的方法，区分了内存对 CKPT 的临时存储和远端存储服务对 CKPT 的持久化存储，不增加远端存储的空间消耗。

在第一方面的一种实现方式中，在所述第一模型训练组上传检查点数据的过程中，禁止删除所述第一模型训练组中各个训练节点的内存中已保存的检查点数据，以及，禁止使用新的检查点数据覆盖所述已保存的检查点数据。

根据上述实现方式的方法，可以避免新检查点数据的生成干扰到检查点数据的上传，确保检查点数据可以顺利上传到远端存储服务。

在第一方面的一种实现方式中，在所述第一模型训练组上传检查点数据的过程中，停止将新的检查点数据保存到所述第一模型训练组的训练节点的内存中。

在第一方面的一种实现方式中，在所述第一模型训练组上传检查点数据完成后，将所述第二模型训练组的训练节点的内存中所保存的所述新的检查点数据，同步到所述第一模型训练组的训练节点的内存中。

根据上述实现方式的方法，可以避免新检查点数据的生成干扰到检查点数据的上传，确保检查点数据可以顺利上传到远端存储服务；并且，可以确保上传检查点数据的训练节点与不上传检查点数据的训练节点的检查点数据版本一致。

第二方面，本申请提供一种模型训练装置，所述装置应用于训练集群，所述训练集群包括多个模型训练组，所述多个模型训练组包括第一模型训练组和第二模型训练组，所述第一模型训练组和所述第二模型训练组用于以数据并行的方式并行训练神经网络模型，所述第一模型训练组以及所述第二模型训练组包含相同的节点结构，所述第一模型训练组包含第一训练节点，所述第二模型训练组包含第二训练节点，其中，基于所述节点结构，所述第一训练节点对应所述第二训练节点；所述训练集群在一轮训练结束后生成检查点数据，其中，所述检查点数据为所述第一训练节点以及所述第二训练节点的检查点数据；

所述装置包括：

存储控制模块，其用于指示所述第一训练节点将所述检查点数据保存到所述第一训练节点的内存中，以及，指示所述第二训练节点将所述检查点数据保存到所述第二训练节点的内存中；

恢复控制模块，其用于若所述第一训练节点在对模型进行训练过程中发生故障，则指示所述第一训练节点从所述第二训练节点的内存中获取所述检查点数据，以对所述第一训练节点进行数据恢复。

在第二方面的一种实现方式中，所述恢复控制模块还用于：

指示所述第二训练节点使用所述第二训练节点自身的内存中保存的检查点数据对所述第二训练节点进行数据恢复。

在第二方面的一种实现方式中，所述训练集群在一轮训练结束时，将所述训练集群中所有训练节点的模型参数进行平均值计算，以计算结果作为该轮训练的检查点数据。

在第二方面的一种实现方式中，所述装置还包括初始化模块，所述初始化模块用于：

指示所述第一训练节点下载初始检查点数据；

指示所述第一训练节点向所述第二训练节点广播所述初始检查点数据。

在第二方面的一种实现方式中，所述装置还包括数据上传模块，所述数据上传模块用于：

指示所述第一模型训练组上传所述第一模型训练组中各个训练节点的内存中所保存的检查点数据到远端存储服务。

在第二方面的一种实现方式中，所述存储控制模块还用于：

在所述第一模型训练组上传检查点数据的过程中，禁止删除所述第一模型训练组中各个训练节点的内存中已保存的检查点数据，以及，禁止使用新的检查点数据覆盖所述已保存的检查点数据。

在第二方面的一种实现方式中，所述存储控制模块还用于：

在所述第一模型训练组上传检查点数据的过程中，停止将新的检查点数据保存到所述第一模型训练组的训练节点的内存中。

在第二方面的一种实现方式中，所述存储控制模块还用于：

在所述第一模型训练组上传检查点数据完成后，将所述第二模型训练组的训练节点的内存中所保存的所述新的检查点数据，同步到所述第一模型训练组的训练节点的内存中。

第三方面，本申请提供一种计算设备集群，包括至少一个计算设备，每个计算设备包括处理器和存储器；

所述至少一个计算设备的处理器用于执行所述至少一个计算设备的存储器中存储的指令，以使得所述计算设备集群执行如第一方面所述的方法。

第四方面，本申请提供一种包含指令的计算机程序产品，当所述指令被计算设备集群运行时，使得所述计算设备集群执行如第一方面所述的方法。

第五方面，本申请提供一种计算机可读存储介质，包括计算机程序指令，当所述计算机程序指令由计算设备集群执行时，所述计算设备集群执行如第一方面所述的方法。

附图说明

图 1 所示为根据本申请一实施例的模型训练集群读写 CKPT 示意图；

图 2 所示为根据本申请一实施例的直接远端存储读写数据示意图；

图 3 所示为根据本申请一实施例的分布式缓存读写数据流程；

图 4 所示为根据本申请一实施例的使用内存读写 CKPT 数据流程；

图 5 所示为根据本申请一实施例的模型训练方法流程示意图；

图 6 所示为根据本申请一实施例的训练集群结构示意图；

图 7 所示为根据本申请一实施例的模型训练方法流程示意图；

图 8 所示为根据本申请一实施例的模型训练装置示意图；

图 9 所示为根据本申请一实施例的训练节点部署场景示意图；

图 10 所示为根据本申请一实施例的训练节点部署场景示意图；

图 11 所示为根据本申请一实施例的计算设备示意图；

图 12 所示为根据本申请一实施例的计算设备集群示意图；

图 13 所示为根据本申请一实施例的计算设备集群网络连接示意图。

具体实施方式

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。

本申请的实施方式部分使用的术语仅用于对本申请的具体实施例进行解释，而非旨在限定本申请。

针对如何避免GPU卡故障导致中断模型训练并浪费GPU算力的问题,在一种可行的技术方案中,采用定期存储检查点（Checkpoint，CKPT）的方式避免训练过程中的训练数据丢失。

例如，在云服务器上提供持久化存储的服务，每3个小时将模型数据存储到云服务器中。若在第4个小时发生了GPU卡故障，则训练集群就可以加载上一次在云服务器存储的模型数据，无需从头开始训练。

然而，大规模分布式训练集群存储和恢复CKPT性能低、成本高。具体的，定期存储检查点的方式对远端存储的网络带宽和性能要求较高，在大规模分布式训练集群中，远端存储的网络带宽和性能会成为整个系统的瓶颈，引入很长的时间开销来恢复训练状态，造成极大的GPU卡资源浪费。

分布式训练集群会对一个大模型进行切分，放到多个节点上进行并行训练，组成一个完整模型的所有节点称为一个模型组。

以512个节点训练700亿参数模型为例，为了提升模型的并行训练效率，512个节点可以切分成32个模型组来并行训练32个模型，16个节点为一个模型组共同训练单个模型。

例如，图1所示为根据本申请一实施例的模型训练集群读写CKPT示意图。

32个模型组中的任意两个模型组如图1所示的模型（model）组101以及模型（model）组102所示。

700亿参数模型，一个模型的CKPT大小为1TB，一张卡放不下如此大的模型，会将该模型分成128份并放到128张卡上进行训练。模型（model）组101包含节点（Node）0~15，模型（model）组102包含节点（Node）16~31。每个节点包含8张GPU卡（GPU 0~7）。

每一轮训练开始时，32个模型组共同加载同一份CKPT以确保到相同的初始化状态，即，在每一轮训练开始时，模型组之间的参数相同。

在训练过程中，32个模型组8张GPU卡流水线并行、优化器并行并且数据并行。

在一轮训练结束时将每个模型组的训练结果进行汇总得到平均值以作为本轮训练的CKPT结果，将CKPT结果同步到所有的模型组之后再进行下一轮训练。

在每轮训练结束后，本轮训练的CKPT结果上传到远端的对象存储服务（Object Storage Service，OBS）中。

每一轮训练开始时32个模型组都需要从OBS中拉取数据，因此OBS需要提供32TB的CKPT数据下载流量来保证32个模型组的数据服务。如果进一步要求30min内下载完，就需要OBS提供至少32TB/30min=145.64Gbps的带宽。在某些应用场景中，OBS还需要提供16TB的数据集下载流量，如果进一步要求30min内下载完，就需要OBS提供至少48TB/30min=218.453Gbps的带宽。这对远端存储的网络带宽需求非常高。随着对下载时长的缩短、模型参数量的增多，对远端存储的带宽需求将会更高，从而显著增加了远端存储的成本。

另一方面，在所有节点存储和下载CKPT期间，GPU训练卡都处于闲置状态，造成极大的算力浪费,且GPU卡都按时间计费,上万张卡每分钟会有几万元的成本开销。值得注意的是,从保存ckeckpoint到集群故障发生时的计算结果是全丢失的，这期间的GPU卡算力浪费也是成本来源之一，因此模型的保存频率也极大地影响了训练集群的性能和成本。

为提升存储和读取CKPT的性能并降低其成本，一种可行的解决方案中，采用直接远端存储。

图2所示为根据本申请一实施例的直接远端存储读写数据示意图。

提升存储和读取CKPT最直接的做法就是提升远端存储服务的性能，因此，通过提升远端存储服务的网络带宽、将远端存储服务放到距离训练集群更近的网络或开发高效的网络文件系统等方式来提升远端存储服务的性能。

如图2所示，远端存储服务被放到距离训练集群的节点（Node）更近的网络。通过这样的方式，训练节点可以使用较高的带宽和较短的网络路径来存储和读取CKPT，代表的技术方案包括亚马逊S3、华为云OBS和华为云弹性文件服务（Scalable File Service，SFS）等。然而这样的方式在性能和成本方面并没有较大的优势，因为该方案仍然需要有大量的流量需要走外部网络，网络路径相比集群内要长很多，且网络带宽越高其成本就越贵。

为提升存储和读取 CKPT 的性能并降低其成本，另一种可行的解决方案中，采用分布式缓存。

图 3 所示为根据本申请一实施例的分布式缓存读写数据流程。

分布式缓存是将存储和读取 CKPT 的路径放到模型训练的集群内。

如图 3 所示，将每个节点的本地固态硬盘（solid-state drive，SSD）磁盘连接起来组成缓存池来扩大存储容量，将 CKPT 存储在缓存池中，并将完整的 CKPT 在缓存池中做均匀打散以保证集群内的网络负载均衡。以这样的方式，集群节点在下载 CKPT 数据时后优先从集群的缓存池中读取，从而利用节点间的网络削减对远端存储服务的网络依赖。代表性的方案包括 JuiceFs, Alluxio 和 MemArts，然而这些方案的性能依赖极致的预取算法来提升缓存池的命中率，且依赖极致的本地亲和调度算法来减少集群内的网络延迟，其性能仍有较大可提升的空间。

为提升存储和读取 CKPT 的性能并降低其成本，另一种可行的解决方案中，采用内存存储。

图 4 所示为根据本申请一实施例的使用内存读写 CKPT 数据流程。

如图 4 所示，采用内存存储和读取 CKPT，是直接将 CKPT 存储到内存（Memory）中，通过快速的内存存储和访问提升 CKPT 的存取性能，减少集群节点对网络的依赖，从而减少网络延迟和带宽的影响，显著提升训练集群存储和读取 CKPT 的性能。但此种方案的缺点在于内存的数据会在系统崩溃时不可用，只有保证保存到内存中的数据在系统崩溃时仍然可以被访问，才能避免集群访问远端存储网络。

在一种内存存储方案（例如，Gemini）中，采用对集群节点进行分组备份的方案，即同组内的节点互相备份该组内其他节点的数据，提高集群节点在系统崩溃时仍然可用的概率，从而避免内存数据不可用。然而，该内存存储方案需要消耗至少一半的内存来备份其他节点的数据，内存消耗较大；且其为了将一个节点的数据备份到其他节点，在训练过程中调度参数面网络来传输备份的 CKPT 数据，可能会干扰正常的模型训练的流量，影响训练效率。

上述方案无法兼具高性能和低成本，其中高性能指的是在训练过程中能尽可能减少 GPU 卡故障的影响，快速恢复到训练状态，提升训练效率；低成本指的是所需的存储空间少、网络带宽需求小、GPU 卡算力浪费少。具体分析如下表 1 所示。

表 1

代表方案	S3,OBS,SFS	JuiceFs, Alluxio, Memarts	AWS Gemini
主要思想	直接远端存储	分布式缓存	内存存储
缺点	网络延迟高； 带宽需求大	依赖极致的预取算法和本地亲和调度算法	内存消耗量大；存在流量干扰
故障恢复性能	低	中等	高，但不稳定
存储空间消耗	大	大	较小
网络带宽需求	高	低	低
卡算力浪费	大	中等	低，但不稳定

其中，直接远端存储由于过于依赖远端存储服务，因此在大模型训练集群中，其网络延迟高，带宽需求大；当系统故障时，需要较长的时间读取远端数据，其故障恢复性能弱；若提升 CKPT 的存储频率，其存储空间消耗大；由于故障时，大量卡处于等待故障恢复的闲置状态，其算力浪费大；

分布式缓存将数据打散在集群的不同节点上，因此其在故障恢复过程中的性能依赖极致的预取算法来保证缓存命中，且依赖本地亲和调度算法来保证数据在本节点命中；如果提升 CKPT 的存储频率，其存储空间消耗大；其削减了对远程网络的带宽需求；但由于较难提升 CKPT 的存储频率，其无法恢复到较近的训练状态，其卡算力浪费较大；

Gemini 采用内存存储的方案，但其依赖每个节点的内存为其他节点做备份，因而内存消耗量大，且存在流量干扰；在故障恢复时，需保证内存节点的数据仍然可用，而 Gemini 依赖备份节点的数据也可用，其性能较高，但不稳定；其将临时数据存储到内存中，可以显著提升 CKPT 的存储频率；该方案从内存中存取数据，削减了远端网络带宽的需求；故障发生时，该方案可以恢复到较近的训练状

态，其卡算力浪费小，但依赖所恢复的数据在节点内存中可用，因此其性能不稳定。

针对大模型训练集群的检查点存储的低性能和高成本问题，本申请一实施例提出了一种模型训练方法，模型训练方法包含训练集群检查点存储流程。

在大模型训练集群中，每个节点的内存往往是节点所有显卡内存的 3 倍以上，有足够的空间可以存储多份的 CKPT 数据。因此，在本申请一实施例的模型训练方法中，直接使用节点的内存存储每一轮训练的 CKPT 数据。

另外，分布式训练集群往往采用多个模型组并行的方式进行训练，而多个模型组在每轮训练结束时会进行数据汇总和同步，在每轮数据同步时，多个模型组的数据是相同的。因此，在本申请一实施例的模型训练方法中，利用多模型组的数据冗余特性为内存数据做备份冗余。

为实现多模型组间 CKPT 数据的相互备份，在本申请一实施例的模型训练方法中，通过追踪每个节点的 CKPT 版本确保多模型组之间的备份数据处于相同的状态以确保数据一致性。

具体的，节点间的 GPU 卡直接通信并用于训练参数交互的网络称为参数面网络。在本申请一实施例的模型训练方法中，采用分布式集群内存协同管理，节点间 GPU 卡连通参数面网络。通过集群内存协同管理控件感知和控制模型 CKPT 的存储和恢复，通过集群内存协同管理控件协同集群节点将每个节点的 CKPT 直接存储到本机内存，实现对 CKPT 的存储；通过追踪和协同所有节点的 CKPT 版本，利用本机内存和参数面网络传输故障恢复所需的 CKPT，实现集群训练状态的恢复。通过集群内存协同管理控件感知训练过程中的 CKPT 存储状态，包括生成时间、存储位置、规模大小、版本号等信息，将 CKPT 直接存储到本机内存或存储器件中，实现 CKPT 的快速存储。

图 5 所示为根据本申请一实施例的模型训练方法流程示意图。

S500，启动模型训练流程。

具体的，在 S500 中，使用训练集群进行模型训练。

本申请一实施例的模型训练方法应用于用于进行模型训练的训练集群。

训练集群包含多个模型训练组，多个模型训练组用于以数据并行的方式并行训练神经网络模型。每个模型训练组针对一个完整的模型结构。每个模型训练组包含多个训练节点，不同的模型训练组包含相同的节点结构，即，不同模型训练组包含的训练节点数相同，不同模型训练组所包含的训练节点一一对应。

例如，训练集群至少包含两个模型训练组（第一模型训练组以及第二模型训练组），第一模型训练组以及第二模型训练组均可对应一个完整的模型结构。第一模型训练组至少包含第一训练节点，第二模型训练组至少包含第二训练节点。基于节点结构，第一训练节点对应第二训练节点。

可以理解的，第一模型训练组还可以包含第三训练节点，第二模型训练组还可以包含第四训练节点。基于节点结构，第三训练节点对应第四训练节点。

S501，在一轮训练结束后生成当前轮次训练的模型参数训练结果，即，生成当前轮次训练的 CKPT。

当前轮次训练的 CKPT 对应一个完整的模型。即，当前轮次训练的 CKPT 包含一个模型训练组中所有训练节点所对应的 CKPT。

具体的，当前轮次训练的 CKPT 至少包含 CKPT A，CKPT A 为第一训练节点以及第二训练节点的 CKPT。可以理解的，当前轮次训练的 CKPT 还可以包含 CKPT B，CKPT B 为第三训练节点以及第四训练节点的 CKPT。

具体的，在 S501 中，通过与不同训练节点进行数据交互，得到训练好的参数，并在一轮训练结束时所有节点的模型参数汇聚到中心节点上进行平均值计算，计算结果作为该轮训练的模型参数训练结果，即当前训练轮次的 CKPT。中心节点可以是训练集群中任意一个训练节点或任意多个训练节点的组合。中心节点也可以是训练集训中所有训练节点以外的任意一个计算节点。

S510，将当前训练轮次的 CKPT 同步给所有模型训练组（所有训练节点）。

具体的，在 S510 中，所有的模型训练组加载同一份 CKPT（当前训练轮次的 CKPT），同一模型训练组中的训练节点分别加载其对应的 CKPT。即，基于节点结构，当前轮次训练的 CKPT 中对应一个训

练节点的部分 CKPT，被加载到模型训练组中的对应训练节点上。

具体的，将 CKPT A 同步到第一训练节点以及第二训练节点，将 CKPT B 同步到第三训练节点以及第四训练节点。

S520，将每个训练节点的 CKPT 保存到该训练节点的内存。

具体的，在 S520 中，第一训练节点将 CKPT A 保存到第一训练节点的内存中，以及，第二训练节点将 CKPT A 保存到第二训练节点的内存中。可以理解的，第三训练节点将 CKPT B 保存到第三训练节点的内存中，以及，第四训练节点将 CKPT B 保存到第二训练节点的内存中。

图 6 所示为根据本申请一实施例的训练集群结构示意图。

如图 6 所示，在一实施例中，训练集群至少包含模型训练组 61 以及模型训练组 62。

模型训练组 61 包含训练节点 600~615，模型训练组 62 包含训练节点 616~631。

在 S500 中，模型训练组 61 中的训练节点进行数据交互进行模型训练，模型训练组 62 中的训练节点进行数据交互进行模型训练。在一轮训练完成后，模型训练组 61（训练节点 600~615）以及模型训练组 62（训练节点 616~631）训练好的模型参数汇总进行平均值计算后得到当前训练轮次的模型参数训练结果，即当前训练轮次的 CKPT。当前训练轮次的 CKPT 包括 CKPT00~15。

当前训练轮次的 CKPT 同步到模型训练组 61 以及模型训练组 62。模型训练组 61 的训练节点 600~615 分别加载当前训练轮次的 CKPT 中其对应的 CKPT，模型训练组 62 的训练节点 616~631 分别加载当前训练轮次的 CKPT 中其对应的 CKPT。即，将 CKPT00~15 分别同步到训练节点 600~615，以及，将 CKPT00~15 分别同步到训练节点 616~631。

在 CKPT 同步后，训练节点 600~615 所加载的 CKPT 分别与训练节点 616~631 所加载的 CKPT 一致。例如，训练节点 600 所加载的 CKPT 与训练节点 616 所加载的 CKPT 一致。

进一步的，将 CKPT00~15 分别保存到训练节点 600~615 的内存中，并且，将 CKPT00~15 分别保存到训练节点 616~631 的内存中。

根据本申请实施例的模型训练方法，使用训练节点自身的内存备份该训练节点的 CKPT，备份过程中训练节点不需要相互间进行通信，大大减少了通信资源的消耗。

可选的，在一实施例中，在训练节点的内存中保存最新一轮训练的 CKPT。

可选的，在另一实施例中，在训练节点的内存中保存包含最新一轮训练的多轮训练的 CKPT。具体的，在训练节点上维护一个键值（Key Value，KV），其中 Key 为 CKPT 基于时间戳的版本号，Value 为 CKPT 相关的元数据，包括该 CKPT 的大小、存储位置等。

可选的，在一实施例中，为了避免内存占用溢出，监控训练节点的内存中 CKPT 的大小，当训练节点的内存中 CKPT 的大小超出预设值时，删除内存中最老版本的 CKPT。

可选的，在一实施例中，为了避免内存占用溢出，限制训练节点的内存中所保存的 CKPT 的版本数。当训练节点的内存中 CKPT 的版本数达到预设值时，删除内存中最老版本的 CKPT。

考虑到训练节点的内存的容量总是有限的，为了扩展内存的容量，可选的，在一实施例中，将训练节点的内存中保存的 CKPT 部分或全部的保存到内存以外的其他存储介质（例如，SSD）中。

当模型训练组数较多时，可以实现较大的模型并行，但由于不同的模型训练组的训练节点内存所保存的 CKPT 数据一致，因此会导致 CKPT 数据的备份冗余也会很大。为了降低备份冗余，可选的，在一实施例中，通过对模型训练组进行分组，一个分组内的多个模型训练组共同对一份 CKPT 数据进行备份。例如，让 4 个模型组备份同一份 CKPT 数据，从而让集群内存保存更多的模型版本。

根据本申请实施例的模型训练方法，使用训练节点的内存存储 CKPT 数据，内存里的 CKPT 数据很快会被新的版本所替代，属于临时存储的数据。但在实际使用过程中，需要每隔一段时间就将某个版本的 CKPT 进行持久化以用作将来的进一步分析和使用。

针对上述情况，在一实施例中，根据 CKPT 持久化的需求将多个模型训练组中的一个模型训练组组的所有训练节点内存中的 CKPT 数据上传到远端存储服务中。

根据本申请一实施例的方法，采用将远端存储移出关键路径的方案，集群内存数据异步持久化到远端，减少对远端性能的依赖；并且，根据本申请一实施例的方法，区分了内存对 CKPT 的临时存储和远端存储服务对 CKPT 的持久化存储，不增加远端存储的空间消耗。

在一实施例中，为确保内存不溢出，在新的检查点数据保存到内存时，新的检查点数据会覆盖旧的检查点数据。或者，监控内存空间占用，在空间不足时删除旧的检查点数据。但是，在数据上传阶段，会出现某版本的检查点数据上传到一半，该检查点数据被删除或被新版本数据覆盖的情况。为避免该情况的发生，在一实施例中，在 CKPT 数据上传到远端存储服务期间，避免该模型训练组的内存数据被删除以及被新版本替换。

根据本申请一实施例的方法，可以避免新检查点数据的生成干扰到检查点数据的上传，确保检查点数据可以顺利上传到远端存储服务。

具体的，在一实施例中，针对正在上传 CKPT 数据到远端存储服务的模型训练组，禁止删除内存中已保存的 CKPT 数据，以及，禁止使用新的 CKPT 数据覆盖已保存的 CKPT 数据。即，在整个数据上传过程中，冻结内存中的已保存的 CKPT 数据。但是，在 CKPT 数据上传过程中，可以将新的 CKPT 数据保存到内存中的未使用空间中。

进一步的，为避免内存存储空间溢出，在另一实施例中，针对正在上传 CKPT 数据到远端存储服务的模型训练组，不执行 S520。即，在整个数据上传过程中，不将新的 CKPT 数据保存到内存中，这样，内存中已保存的 CKPT 数据就不会被新的 CKPT 数据覆盖，也不会出现因存储空间不足导致已保存的 CKPT 数据被删除。在整个数据上传过程中，保证内存中已保存的 CKPT 数据不变。在 CKPT 数据上传到远端存储服务之后，从其他模型训练组（例如，相邻模型训练组）中同步，CKPT 数据上传期间，未保存到内存中的最新版本的 CKPT 数据。

根据本申请一实施例的方法，可以避免新检查点数据的生成干扰到检查点数据的上传，确保检查点数据可以顺利上传到远端存储服务；并且，可以确保上传检查点数据的训练节点与不上传检查点数据的训练节点的检查点数据版本一致。

基于本申请一实施例提出的训练集群检查点存储流程，本申请一实施例的模型训练方法还包括训练集群检查点恢复流程。

在本申请一实施例的模型训练方法中，利用训练节点的内存中所保存的 CKPT 和参数面网络做快速的数据恢复。

具体的，当训练节点发生故障时，修复故障的训练节点并将所有训练节点的状态恢复到最近的 CKPT 以保证所有节点处于一致状态。

对于未发生故障的训练节点，优先从本机内存中读取最近的 CKPT 数据。由于读内存的速度快，因此该方式恢复 CKPT 的效率高。

对于已修复故障的训练节点，其内存中没有之前存储的 CKPT，因此，通过参数面网络读取其他模型训练组中相对应的训练节点的内存中的 CKPT 数据进行 CKPT 状态恢复。

图 7 所示为根据本申请一实施例的模型训练方法流程示意图。

S700，启动模型训练流程。参照 S500。

S710，判断是否存在训练节点故障。

如不存在训练节点故障，返回 S700。

如存在训练节点故障，执行 S711。

S711，中断所有训练节点的训练操作。

S720，修复发生故障的训练节点。

S730，针对发生故障的训练节点以外的训练节点，使用训练节点自身内存中保存的最新版本的 CKPT 数据进行训练节点数据恢复。

S731，针对发生故障的训练节点，使用其他模型组中，基于节点结构对应该故障训练节点的训练节点内存中保存的最新版本的 CKPT 数据进行训练节点数据恢复。

例如，第一模型训练组中的第一训练节点发生故障，针对第二模型训练组中的第二训练节点，使

用第二训练节点自身内存中保存的最新版本的 CKPT 数据进行训练节点数据恢复。针对第一模型训练组中的第一训练节点，第一训练节点从第二训练节点的内存中获取检查点数据，以对第一训练节点进行数据恢复。

如图 6 所示，在一实施例中，如果训练节点 615 发生故障，中断训练节点 600~615 以及训练节点 616~631 的训练。修复训练节点 615 的故障，分别使用训练节点 600~614 的内存中最新版本的 CKPT 数据恢复训练节点 600~614，分别使用训练节点 616~631 的内存中最新版本的 CKPT 数据恢复训练节点 616~631，使用训练节点 631 的内存中最新版本的 CKPT 数据恢复训练节点 615。

根据本申请实施例的方法，在大模型训练集群中，可以解决存储和读取 CKPT 的低性能和高成本问题，降低存储和读取 CKPT 的时间，提升存储 CKPT 的存储频率且不增加存储空间的消耗，减少对远端存储服务的带宽需求，减少 GPU 卡的算力浪费。

根据本申请一实施例的方法，对于发生故障的场景，利用多模型组冗余备份的特性对数据进行重传和恢复。通过这样的方式，远端存储服务被移出关键路径，减少了对远端存储的性能和带宽需求，显著降低了整体系统的成本。

进一步的，在模型训练前，训练集群的训练节点在初始状态加载 CKPT 时，所有训练节点的本机内存都没有保存 CKPT 数据，如果让所有训练节点都从远程存储服务中拉取 CKPT 数据会导致远端存储的瞬时带宽需求非常大，性能严重下降。

针对上述情况，在一实施例中，在训练集群的训练节点在初始状态加载 CKPT 时，令多个模型训练组中的一个模型训练组从远程存储服务中拉取 CKPT。从远程存储服务中下载 CKPT 的模型训练组的训练节点，通过参数面网络向其他模型训练组的训练节点发送（例如，通过广播方式）CKPT。

例如，使用第一模型训练组中的第一训练节点下载对应该训练节点的初始检查点数据；由第一模型训练组中的第一训练节点向第二模型训练组中的第二训练节点广播第一训练节点下载到的初始检查点数据。

可选的，在广播的过程中，也可以让已经得到广播数据的模型组广播给更多的模型组。

例如，使用第一模型训练组中训练节点下载初始检查点数据；由第一模型训练组中的训练节点向第二模型训练组中的训练节点广播下载到的初始检查点数据；由第二模型训练组中的训练节点向第三模型训练组中的训练节点广播下载到的初始检查点数据。

根据本申请一实施例的方法，下载初始检查点数据到一个模型训练组，并通过一个模型训练组向其他模型训练组发送初始检查点数据，减少了初始检查点数据时的带宽需求，提高了初始检查点数据的下载效率。

进一步的，根据本申请实施例提出的模型训练方法，本申请一实施例还提出了一种模型训练装置，该模型训练装置应用于训练集群。模型训练装置所执行的方法步骤可以参照之前方法实施例中的方法步骤。

图 8 所示为根据本申请一实施例的模型训练装置示意图。

如图 8 所示，在一实施例中，模型训练装置 800 应用于训练集群 810。

训练集群 810 至少包含第一模型训练组 811 以及第二模型训练组 812，第一模型训练组 811 以及第二模型训练组 812 分别针对完整模型结构，第一模型训练组 811 以及第二模型训练组 812 包含相同的节点结构。

如图 8 所示，第一模型训练组 811 至少包含第一训练节点 813，第二模型训练组 812 至少包含第二训练节点 814，其中，基于节点结构，第一训练节点 813 对应第二训练节点 814。

第一模型训练组 811 还包含第三训练节点 815，第二模型训练组 812 还包含第四训练节点 816，其中，基于节点结构，第三训练节点 815 对应第四训练节点 816。

训练集群 810 在一轮训练结束后生成当前轮次训练的模型参数训练结果，即，生成当前轮次训练的 CKPT。具体的，训练集群 810 在一轮训练结束时，将所有训练节点的模型参数进行平均值计算，以计算结果作为该轮训练的检查点数据。

具体的，训练集群 810 在一轮训练结束后至少生成 CKPT A，CKPT A 为第一训练节点 813 以及第二训练节点 814 的 CKPT。可以理解的，训练集群 810 在一轮训练结束后还生成 CKPT B，CKPT B 为第三训练节点 815 以及第四训练节点 816 的 CKPT。

训练集群 810 将当前轮次训练的 CKPT 同步到训练集群 810 中的每一个模型训练组。具体的，训练集群 810 基于节点结构，将当前轮次训练的 CKPT 中单个训练节点的 CKPT 同步到每一个模型训练组的对应训练节点。即，CKPT A 同步到第一训练节点 813 以及第二训练节点 814，CKPT B 同步到第三训练节点 815 以及第四训练节点 816。

在一实施例中，模型训练装置 800 包括存储控制模块 801，存储控制模块 801 用于指示训练集群 810 中的每一个训练节点，将自身获取到的 CKPT 保存到自身的内存中。

即，存储控制模块 801 指示第一训练节点 813 将第一训练节点 813 获取到的 CKPT A 保存到第一训练节点 813 的内存中，以及，存储控制模块 801 指示第二训练节点 814 将第二训练节点 814 获取到的 CKPT A 保存到第二训练节点 814 的内存中；

存储控制模块 801 还指示第三训练节点 815 将第三训练节点 815 获取到的 CKPT B 保存到第三训练节点 815 的内存中，以及，存储控制模块 801 还指示第四训练节点 816 将第四训练节点 816 获取到的 CKPT B 保存到第四训练节点 816 的内存中。

在模型训练过程中，如果训练集群 810 中的训练节点发生故障，则训练集群 810 中断所有训练节点的训练操作，修复发生故障的训练节点。

在一实施例中，模型训练装置 800 还包括恢复控制模块 802，恢复控制模块 802 用于在训练集群 810 中的某一个训练节点发生故障时，指示发生故障的训练节点，使用其他模型组中，基于节点结构对应该故障训练节点的训练节点内存中保存的最新版本的 CKPT 数据进行训练节点数据恢复；以及，指示发生故障的训练节点以外的训练节点，使用训练节点自身内存中保存的最新版本的 CKPT 数据进行训练节点数据恢复。

例如，若第一训练节点 813 在对模型进行训练过程中发生故障，则恢复控制模块 802 指示第一训练节点 813 从第二训练节点 814 的内存中获取检查点数据，以对第一训练节点 813 进行数据恢复。以及，恢复控制模块 802 指示第二训练节点 814 从第二训练节点 814 的内存中获取检查点数据，以对第二训练节点 814 进行数据恢复；指示第三训练节点 815 从第三训练节点 815 的内存中获取检查点数据，以对第三训练节点 815 进行数据恢复；指示第四训练节点 816 从第四训练节点 816 的内存中获取检查点数据，以对第四训练节点 816 进行数据恢复。

可选的，在一实施例中，模型训练装置 800 还包括初始化模块 803。初始化模块 803 用于，在训练集群 810 开始训练模型之前，指示训练集群 810 中的一个模型训练组下载对应完整模型结构的初始检查点数据，即，该模型训练组中的每一个训练节点下载对应该训练节点的初始检查点数据；以及，指示完成下载初始检查点数据的模型训练组，将下载的初始检查点数据发送到其他模型训练组，即，完成下载初始检查点数据的训练节点，将下载的初始检查点数据发送到其他模型训练组中的对应训练节点。

具体的，初始化模块 803 指示第一训练节点 813 下载对应第一训练节点 813 的初始检查点数据；以及，指示第一训练节点 813 向第二训练节点 814 广播下载到的初始检查点数据。可以理解的，初始化模块 803 还指示第三训练节点 815 下载对应第三训练节点 815 的初始检查点数据；以及，指示第三训练节点 815 向第四训练节点 816 广播下载到的初始检查点数据。

在一实施例中，模型训练装置 800 还包括数据上传模块 804，数据上传模块 804 用于指示训练集群 810 中的一个模型训练组上传该模型训练组中各个训练节点的内存中所保存的检查点数据到远端存储服务；在一个模型训练组上传该模型训练组中各个训练节点的内存中所保存的检查点数据到远端存储服务的过程中，其他模型训练组不上传检查点数据到远端存储服务。

例如，数据上传模块 804 指示第一模型训练组 811 上传第一模型训练组 811 中各个训练节点的内存中所保存的检查点数据到远端存储服务，与此同时，第二模型训练组 812 不上传检查点数据到远端存储服务。

可选的，在一实施例中，存储控制模块 801 还用于，在某一个模型训练组上传检查点数据到远端

存储服务的过程中，禁止删除该模型训练组中各个训练节点的内存中已保存的检查点数据，以及，禁止使用新的检查点数据覆盖该模型训练组中各个训练节点的内存中已保存的检查点数据。

例如，在第一模型训练组 811 上传检查点数据的过程中，存储控制模块 801 禁止删除第一模型训练组 811 中各个训练节点的内存中已保存的检查点数据，以及，禁止使用新的检查点数据覆盖第一模型训练组 811 中各个训练节点的内存中已保存的检查点数据。

可选的，在另一实施例中，存储控制模块 801 还用于，在某一个模型训练组上传检查点数据到远端存储服务的过程中，停止将新的检查点数据保存到该模型训练组的训练节点的内存中。

例如，在第一模型训练组 811 上传检查点数据的过程中，存储控制模块 801 停止将新的检查点数据保存到第一模型训练组 811 的训练节点的内存中。

进一步的，在一实施例中，存储控制模块 801 还用于，在某一模型训练组上传检查点数据完成后，将另一个模型训练组的训练节点的内存中所保存的新的检查点数据（该新的检查点数据为，模型训练组上传检查点数据的过程中，未能保存到模型训练组的训练节点的内存中的检查点数据），同步到上传检查点数据的模型训练组的训练节点的内存中。

例如，在第一模型训练组 811 上传检查点数据完成后，将第二模型训练组 812 的训练节点的内存中所保存的新的检查点数据，同步到第一模型训练组 811 的训练节点的内存中。

在本申请实施例的描述中，为了描述的方便，以功能分为各种模块分别描述，各个模块的划分仅仅是一种逻辑功能的划分，在实施本申请实施例时可以把各模块的功能在同一个或多个软件和/或硬件中实现。

具体的，本申请实施例所提出的装置在实际实现时可以全部或部分集成到一个物理实体（例如，GPU 或其他类型处理器）上，也可以物理上分开。且这些模块可以全部以软件通过处理元件调用的形式实现；也可以全部以硬件的形式实现；还可以部分模块以软件通过处理元件调用的形式实现，部分模块通过硬件的形式实现。例如，检测模块可以为单独设立的处理元件，也可以集成在电子设备的某一个芯片中实现。其它模块的实现与之类似。此外这些模块全部或部分可以集成在一起，也可以独立实现。在实现过程中，上述方法的各步骤或以上各个模块可以通过处理器元件中的硬件的集成逻辑电路或者软件形式的指令完成。

例如，在上述实施例中，存储控制模块 801、恢复控制模块 802、初始化模块 803 和数据上传模块 804 均可以通过软件实现，或者可以通过硬件实现。示例性的，接下来以存储控制模块 801 为例，介绍存储控制模块 801 的实现方式。类似的，恢复控制模块 802、初始化模块 803 和数据上传模块 804 的实现方式可以参考存储控制模块 801 的实现方式。

模块作为软件功能单元的一种举例，存储控制模块 801 可以包括运行在计算实例上的代码。其中，计算实例可以包括物理主机（计算设备）、虚拟机、容器中的至少一种。进一步地，上述计算实例可以是一台或者多台。例如，存储控制模块 801 可以包括运行在多个主机/虚拟机/容器上的代码。需要说明的是，用于运行该代码的多个主机/虚拟机/容器可以分布在相同的区域（region）中，也可以分布在不同的 region 中。进一步地，用于运行该代码的多个主机/虚拟机/容器可以分布在相同的可用区（availability zone，AZ）中，也可以分布在不同的 AZ 中，每个 AZ 包括一个数据中心或多个地理位置相近的数据中心。其中，通常一个 region 可以包括多个 AZ。

同样，用于运行该代码的多个主机/虚拟机/容器可以分布在同一个虚拟私有云（virtual private cloud，VPC）中，也可以分布在多个 VPC 中。其中，通常一个 VPC 设置在一个 region 内，同一 region 内两个 VPC 之间，以及不同 region 的 VPC 之间跨区通信需在每个 VPC 内设置通信网关，经通信网关实现 VPC 之间的互连。

模块作为硬件功能单元的一种举例，存储控制模块 801 可以包括至少一个计算设备，如服务器等。或者，存储控制模块 801 也可以是利用专用集成电路（application-specific integrated circuit，ASIC）实现、或可编程逻辑器件（programmable logic device，PLD）实现的设备等。其中，上述 PLD 可以是复杂程序逻辑器件（complex programmable logical device，CPLD）、现场可编程门阵列（field-programmable gate array，FPGA）、通用阵列逻辑（generic array logic，GAL）或其任意组合实现。

存储控制模块 801 包括的多个计算设备可以分布在相同的 region 中，也可以分布在不同的 region 中。存储控制模块 801 包括的多个计算设备可以分布在相同的 AZ 中，也可以分布在不同的 AZ 中。同样，存储控制模块 801 包括的多个计算设备可以分布在同一个 VPC 中，也可以分布在多个 VPC 中。其中，所述多个计算设备可以是服务器、ASIC、PLD、CPLD、FPGA 和 GAL 等计算设备的任意组合。

需要说明的是，在其他实施例中，存储控制模块 801 可以用于执行 XXX 方法 中的任意步骤，恢复控制模块 802 可以用于执行 XXX 方法中的任意步骤，初始化模块 803 可以用于执行 XXX 方法中的任意步骤，存储控制模块 801、恢复控制模块 802、以及初始化模块 803 负责实现的步骤可根据需要指定，通过存储控制模块 801、恢复控制模块 802、以及初始化模块 803 分别实现 XXX 方法中不同的步骤来实现 YY 装置的全部功能。

具体的，在一实施例中，存储控制模块 801、恢复控制模块 802、初始化模块 803 和数据上传模块 804 中的一个或多个可以基于模型训练框架实现。

在本申请实施例中，开发者用来开发深度学习模型的代码库称为模型训练框架，可完成模型定义、训练、推理等完整流程。模型训练框架提供对训练节点的内存管理，以实现将 CKPT 备份到内存以及从内存中的 CKPT 恢复训练节点。

图 9 所示为根据本申请一实施例的训练节点部署场景示意图。

如图 9 所示，每个训练节点配备多个用于模型训练的显卡和大容量的内存，训练节点与训练节点之间除了传统的传输控制协议（Transmission Control Protocol，TCP）网络外，训练节点的显卡间还包括高带宽的参数面网络用于模型的参数传输。

在每个训练节点的操作系统中安装用于模型训练的训练框架和根据本申请一实施例的方法的软件包，其中，根据本申请一实施例的方法的软件包中提供一个可以被训练框架调用的软件开发包（Software Development Kit，SDK），该 SDK 作为训练框架与根据本申请一实施例的方法的软件包交互的接口使用，用于调用根据本申请一实施例的方法的存储和恢复 CKPT 的功能。

在训练集群外包括提供数据持久化服务的远端存储服务。远端存储服务提供可移植操作系统接口 POSIX（Portable Operating System Interface of UNIX，POSIX）。

根据本申请一实施例的方法，采用集群内存协同管理，节点内存存储整份 CKPT，提升存储 CKPT 的频率；并且，集群内存控件管理多节点 CKPT，支持分组冗余备份；并且，CKPT 直接存储到内存，让存储 CKPT 的时延大幅降低。

根据本申请一实施例的方法，采用本机内存和参数面网络协同恢复，直接读取本机内存的数据，大幅缩短了 CKPT 的恢复时延；并且，故障节点通过参数面网络读取数据，大幅缩短了恢复时延。

本申请实施例所提供的方法以及装置，其涉及的产品包括为大模型分布式训练集群提供 CKPT 存储和下载的服务，可能以加快 CKPT 存储和访问效率的服务形式，绑定到大模型训练的集群中进行售卖。

具体的，在本申请实施例所提供的方法以及装置所涉及的产品中：

训练节点内存存储该节点完整的 CKPT 数据：在计算机系统结构中，有效地使用内存来存储或缓存数据的设计和产品有很多。根据本申请一实施例的方法，使用内存的一大特征就是使用节点的内存存储该节点完整的 CKPT 数据，而非缓存部分数据。这样做的好处是，在系统需要恢复 CKPT 数据时，只要节点内保存了对应版本的数据，就可以完全从本节点的内存中读取相应数据，而无需去其他地方读取；

参数面网络和本机内存协同快恢：与从远端存储中读取 CKPT 方案不同的是，根据本申请一实施例的方法，在故障恢复时优先从本机内存中读取数据，由于每个节点存储的恰好为该节点完整的 CKPT 数据，因此该方案可以做到秒级的故障恢复；如果本机恰好为故障后被替换的节点，则通过参数面网络从集群中的冗余备份节点读取数据，对于有 32 个模型组的 AI 训练集群，每个节点最多可以有 32 个冗余备份，这么多的冗余备份大大提升了节点内存备份数据的可用性；

对 CKPT 数据区分临时存储和持久存储：根据本申请一实施例的方法，在训练过程中，将每一轮

的 CKPT 数据保存到内存中作为临时存储，并按照持久化需求异步地将内存数据持久化到远端存储上，将远端存储移出关键路径，兼顾了存储和网络带宽的成本及存储和恢复的性能。

图 10 所示为根据本申请一实施例的训练节点部署场景示意图。

训练集群至少包含模型训练组 901 以及模型训练组 902。模型训练组 901 包含训练节点 911~914，模型训练组 902 包含训练节点 915~918。

训练集群中安装有训练框架及相关组件，让其运行在训练集群的操作系统运行环境中，以提供训练框架服务 920。

训练节点 911~914、训练节点 915~918 安装有相关功能的 SDK 与训练框架对接。

在模型训练过程中，训练框架服务 920 会调用 SDK 的存储功能接口以进行 CKPT 存储。

训练集群以外部署有远端存储服务 930。在模型训练过程中，训练框架服务 920 定期将训练节点 911~914 或训练节点 915~918 的内存中保存的 CKPT 上传到远端存储服务 930。

具体的，在开始模型训练前，训练框架服务 920 调用训练节点的 SDK 提供的加载接口，当检测到训练节点没有保存 CKPT 数据时，在所有模型训练组选择一个（模型训练组 901 或 902），让模型训练组 901 或 902 从远端存储服务 920 下载 CKPT。

从远端存储服务 920 下载 CKPT 数据的模型训练组 901 或 902 通过参数面网络向其他训练节点（模型训练组 902 或 901 的训练节点）广播数据。

训练框架服务 920 启动训练流程，通过与不同训练节点进行数据交互，得到训练好的参数，并在一轮训练结束时所有节点的模型参数会汇聚到中心节点上进行平均值计算，以作为该轮训练的模型参数训练结果，即 CKPT。

中心节点可以是训练集群中的任意训练节点，或者，中心节点也可以是在训练集群中配置的，训练节点以外的独立节点。

在一轮训练结束并得到 CKPT 后，中心节点会将本轮的最终 CKPT 同步给所有节点以准备进行下一轮训练，同时，训练框架服务 920 调用每个训练节点的 SDK 的存储 CKPT 功能以将 CKPT 保存到每个训练节点的内存。

训练节点的 SDK 在得到存储临时 CKPT 的调用时，通知训练节点检查本机内存是否有足够空间存储当前的 CKPT，如果本机内存不足则检查用户配置了溢出到磁盘的选项。如果用户配置了溢出到磁盘的选项，则后续 CKPT 的存储流程会直接存储到本地磁盘上，否则会先删除本机内存中最老版本的 CKPT。

训练节点将其 CKPT 存储到预先分配好的本机内存或磁盘中，并在本机维护一个 KV，其中 Key 为 CKPT 基于时间戳的版本号，Value 为 CKPT 相关的元数据，包括该 CKPT 的大小、存储位置等。

在进行多轮模型训练的过程中，每个预设的时间间隔（例如，3 小时）或训练轮次（例如，30 轮），训练框架服务 920 将当前的 CKPT 放到远端存储服务 930 中，以便将来做进一步的分析和使用。

具体的，训练框架服务 920 在启动时，会配置持久化到远端存储服务 930 的时间（例如，3 小时），并在满足持久化存储的条件时由训练框架服务 920 通过训练节点的 SDK 调用持久化存储的接口以进行持久化存储。

训练框架服务 920 在所有模型训练组中选择其中一个作为持久化存储的模型训练组，并让该模型训练组将其本机内存存储的 CKPT 上传到远端存储服务中，该模型组中最新版本的 CKPT 在上传期间会被锁定，不会被删除。被选中的模型训练组将 CKPT 持久化到远端存储后，完成 CKPT 的持久化操作并解除对 CKPT 的锁定状态。

在故障恢复时，训练框架服务 920 调用训练节点的 SDK 来加载模型 CKPT 以准备恢复训练。具体的，训练框架服务 920 确定所保存的最新的 CKPT 版本，训练节点从自身的内存中查找是否保存有该版本对应的 CKPT。如果训练节点自身的内存中保存有该版本对应的 CKPT，训练节点直接从内存中读

取最新的 CKPT 数据，加载到显卡上以进行下一轮模型训练，并结束恢复操作。如果训练节点自身的内存中没有保存有该版本对应的 CKPT，则训练节点通过参数面网络从其他模型训练组的对应训练节点中读取该版本的 CKPT 数据，加载到显卡上以进行下一轮模型训练，并结束恢复操作。

本申请还提供一种计算设备。

图 11 所示为根据本申请一实施例的计算设备示意图。

如图 11 所示，计算设备 1100 包括：总线 1102、处理器 1104、存储器 1106 和通信接口 1108。处理器 1104、存储器 1106 和通信接口 1108 之间通过总线 1102 通信。计算设备 1100 可以是服务器或终端设备。应理解，本申请不限定计算设备 1100 中的处理器、存储器的个数。

总线 1102 可以是外设部件互连标准（peripheral component interconnect，PCI）总线或扩展工业标准结构（extended industry standard architecture，EISA）总线等。总线可以分为地址总线、数据总线、控制总线等。为便于表示，图 11 中仅用一条线表示，但并不表示仅有一根总线或一种类型的总线。总线 1102 可包括在计算设备 1100 各个部件（例如，存储器 1106、处理器 1104、通信接口 1108）之间传送信息的通路。

处理器 1104 可以包括中央处理器(central processing unit，CPU)、图形处理器（graphics processing unit，GPU）、微处理器（micro processor，MP）或者数字信号处理器（digital signal processor，DSP）等处理器中的任意一种或多种。

存储器 1106 可以包括易失性存储器(volatile memory)，例如随机存取存储器(random access memory，RAM)。处理器 1104 还可以包括非易失性存储器(non-volatile memory)，例如只读存储器(read-only memory，ROM)，快闪存储器，机械硬盘(hard disk drive，HDD)或固态硬盘(solid state drive，SSD)。

存储器 1106 中存储有可执行的程序代码，处理器 1104 执行该可执行的程序代码以分别实现前述存储控制模块 801、恢复控制模块 802、初始化模块 803 和数据上传模块 804 的功能，从而实现本申请实施例提出的模型训练方法。也即，存储器 1106 上存有用于执行本申请实施例提出的模型训练方法的指令。

通信接口 1108 使用例如但不限于网络接口卡、收发器一类的收发模块，来实现计算设备 1100 与其他设备或通信网络之间的通信。

本申请实施例还提供了一种计算设备集群。该计算设备集群包括至少一台计算设备。该计算设备可以是服务器，例如是中心服务器、边缘服务器，或者是本地数据中心中的本地服务器。在一些实施例中，计算设备也可以是台式机、笔记本电脑或者智能手机等终端设备。

图 12 所示为根据本申请一实施例的计算设备集群示意图。

如图 12 所示，所述计算设备集群包括至少一个计算设备 1100。计算设备集群中的一个或多个计算设备 1100 中的存储器 1106 中可以存有相同的用于执行本申请实施例所提出的模型训练方法的指令。

在一些可能的实现方式中，该计算设备集群中的一个或多个计算设备 1100 的存储器 1106 中也可以分别存有用于执行本申请实施例所提出的模型训练方法的部分指令。换言之，一个或多个计算设备 1100 的组合可以共同执行用于执行本申请实施例所提出的模型训练方法的指令。

需要说明的是，计算设备集群中的不同的计算设备 1100 中的存储器 1106 可以存储不同的指令，分别用于执行本申请实施例所提出的模型训练装置的部分功能。也即，不同的计算设备 1100 中的存储器 1106 存储的指令可以实现存储控制模块 801、恢复控制模块 802、初始化模块 803 和数据上传模块 804 中的一个或多个模块的功能。

在一些可能的实现方式中，计算设备集群中的一个或多个计算设备可以通过网络连接。其中，所述网络可以是广域网或局域网等等。

图 13 所示为根据本申请一实施例的计算设备集群网络连接示意图。

如图 13 所示，两个计算设备 1100A 和 1100B 之间通过网络进行连接。具体地，通过各个计算设备中的通信接口与所述网络进行连接。

图 13 所示的计算设备集群之间的连接方式可以是考虑到本申请提供的模型方法需要多训练节点

间交互、多训练节点数据汇总以及多训练节点数据同步。

应理解，图 13 中示出的计算设备 1100A 的功能也可以由多个计算设备 1100 完成。同样，计算设备 1100B 的功能也可以由多个计算设备 1100 完成。

本申请实施例还提供了另一种计算设备集群。该计算设备集群中各计算设备之间的连接关系可以类似的参考图 12 和图 13 所述计算设备集群的连接方式。不同的是，该计算设备集群中的一个或多个计算设备 1100 中的存储器 1106 中可以存有相同的用于执行模型训练方法的指令。

在一些可能的实现方式中，该计算设备集群中的一个或多个计算设备 1100 的存储器 1106 中也可以分别存有用于执行模型训练方法方法的部分指令。换言之，一个或多个计算设备 1100 的组合可以共同执行用于执行模型训练方法方法的指令。

本领域内的技术人员应明白，本申请实施例可提供为方法、装置、或计算机程序产品。因此，本发明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，本发明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上实施的计算机程序产品的形式。

在本申请所提供的几个实施例中，任一功能如果以软件功能单元的形式实现并作为独立的产品销售或使用时，可以存储在一个计算机可读取存储介质中。基于这样的理解，本申请的技术方案本质上或者说对现有技术做出贡献的部分或者该技术方案的部分可以以软件产品的形式体现出来，该计算机软件产品存储在一个存储介质中，包括若干指令用以使得一台计算机设备（可以是个人计算机，服务器，或者网络设备等）执行本申请各个实施例所述方法的全部或部分步骤。

具体的，本申请实施例还提供了一种包含指令的计算机程序产品。所述计算机程序产品可以是包含指令的，能够运行在计算设备上或被储存在任何可用介质中的软件或程序产品。当所述计算机程序产品在至少一个计算设备上运行时，使得至少一个计算设备执行本申请实施例所提出的模型训练方法。

本申请实施例还提供了一种计算机可读存储介质。所述计算机可读存储介质可以是计算设备能够存储的任何可用介质或者是包含一个或多个可用介质的数据中心等数据存储设备。所述可用介质可以是磁性介质，（例如，软盘、硬盘、磁带）、光介质（例如，DVD）、或者半导体介质（例如固态硬盘）等。该计算机可读存储介质包括指令，所述指令指示计算设备执行本申请实施例所提出的模型训练方法。

本申请中的实施例描述是参照根据本申请实施例的方法、设备（装置）、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。

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。

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。

还需要说明的是，本申请实施例中，“至少一个”是指一个或者多个，“多个”是指两个或两个以上。“和/或”，描述关联对象的关联关系，表示可以存在三种关系，例如，A 和/或 B，可以表示单独存在 A、同时存在 A 和 B、单独存在 B 的情况。其中 A，B 可以是单数或者复数。字符“/”一般表示前后关联对象是一种“或”的关系。“以下至少一项”及其类似表达，是指的这些项中的任意组合，包括单项或复数项的任意组合。例如，a，b 和 c 中的至少一项可以表示：a，b，c，a 和 b，a 和 c，b 和 c 或 a 和 b 和 c，其中 a，b，c 可以是单个，也可以是多个。

本申请实施例中，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。

本申请可以在由计算机执行的计算机可执行指令的一般上下文中描述，例如程序模块。一般地，程序模块包括执行特定任务或实现特定抽象数据类型的例程、程序、对象、组件、数据结构等等。也可以在分布式计算环境中实践本申请，在这些分布式计算环境中，由通过通信网络而被连接的远程处理设备来执行任务。在分布式计算环境中，程序模块可以位于包括存储设备在内的本地和远程计算机存储介质中。

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，对于装置实施例而言，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。

本领域普通技术人员可以意识到，本申请实施例中描述的各单元及算法步骤，能够以电子硬件、计算机软件和电子硬件的结合来实现。这些功能究竟以硬件还是软件方式来执行，取决于技术方案的特定应用和设计约束条件。专业技术人员可以对每个特定的应用来使用不同方法来实现所描述的功能，但是这种实现不应认为超出本申请的范围。

所属领域的技术人员可以清楚地了解到，为描述的方便和简洁，上述描述的装置、装置和单元的具体工作过程，可以参考前述方法实施例中的对应过程，在此不再赘述。

以上所述，仅为本申请的具体实施方式，任何熟悉本技术领域的技术人员在本申请公开的技术范围内，可轻易想到变化或替换，都应涵盖在本申请的保护范围之内。本申请的保护范围应以所述权利要求的保护范围为准。

权 利 要 求 书

1. 一种模型训练方法，其特征在于，所述方法应用于训练集群，所述训练集群包括多个模型训练组，所述多个模型训练组包括第一模型训练组和第二模型训练组，所述第一模型训练组和所述第二模型训练组用于以数据并行的方式并行训练神经网络模型，所述第一模型训练组以及所述第二模型训练组包含相同的节点结构，所述第一模型训练组包含第一训练节点，所述第二模型训练组包含第二训练节点，其中，基于所述节点结构，所述第一训练节点对应所述第二训练节点，所述方法包括：

在一轮训练结束后生成检查点数据，其中，所述检查点数据为所述第一训练节点以及所述第二训练节点的检查点数据；

所述第一训练节点将所述检查点数据保存到所述第一训练节点的内存中，以及，所述第二训练节点将所述检查点数据保存到所述第二训练节点的内存中；

若所述第一训练节点在对模型进行训练过程中发生故障，则所述第一训练节点从所述第二训练节点的内存中获取所述检查点数据，以对所述第一训练节点进行数据恢复。

2.根据权利要求 1 所述的方法，其特征在于，所述方法还包括：

针对未发生故障的所述第二训练节点，使用所述第二训练节点自身的内存中保存的检查点数据对所述第二训练节点进行数据恢复。

3.根据权利要求 1 所述的方法，其特征在于，所述在一轮训练结束后生成检查点数据，包括：

在一轮训练结束时，将所述训练集群中所有训练节点的模型参数进行平均值计算，以计算结果作为该轮训练的检查点数据。

4.根据权利要求 1 所述的方法，其特征在于，所述方法还包括：

使用所述第一训练节点下载初始检查点数据；

由所述第一训练节点向所述第二训练节点广播所述初始检查点数据。

5.根据权利要求 1-4 中任一项所述的方法，其特征在于，所述方法还包括：

使用所述第一模型训练组上传所述第一模型训练组中各个训练节点的内存中所保存的检查点数据到远端存储服务。

6.根据权利要求 5 所述的方法，其特征在于，在所述第一模型训练组上传检查点数据的过程中，禁止删除所述第一模型训练组中各个训练节点的内存中已保存的检查点数据，以及，禁止使用新的检查点数据覆盖所述已保存的检查点数据。

7.根据权利要求 5 所述的方法，其特征在于，在所述第一模型训练组上传检查点数据的过程中，停止将新的检查点数据保存到所述第一模型训练组的训练节点的内存中。

8.根据权利要求 7 所述的方法，其特征在于，在所述第一模型训练组上传检查点数据完成后，将所述第二模型训练组的训练节点的内存中所保存的所述新的检查点数据，同步到所述第一模型训练组的训练节点的内存中。

9. 一种模型训练装置，其特征在于，所述装置应用于训练集群，所述训练集群包括多个模型训练组，所述多个模型训练组包括第一模型训练组和第二模型训练组，所述第一模型训练组和所述第二模型训练组用于以数据并行的方式并行训练神经网络模型，所述第一模型训练组以及所述第二模型训练组包含相同的节点结构，所述第一模型训练组包含第一训练节点，所述第二模型训练组包含第二训练节点，其中，基于所述节点结构，所述第一训练节点对应所述第二训练节点；所述训练集群在一轮训练结束后生成检查点数据，其中，所述检查点数据为所述第一训练节点以及所述第二训练节点的检查点数据；

所述装置包括：

存储控制模块，其用于指示所述第一训练节点将所述检查点数据保存到所述第一训练节点的内存中，以及，指示所述第二训练节点将所述检查点数据保存到所述第二训练节点的内存中；

恢复控制模块，其用于若所述第一训练节点在对模型进行训练过程中发生故障，则指示所述第一训练节点从所述第二训练节点的内存中获取所述检查点数据，以对所述第一训练节点进行数据恢复。

10.根据权利要求 9 所述的装置，其特征在于，所述恢复控制模块还用于：

指示所述第二训练节点使用所述第二训练节点自身的内存中保存的检查点数据对所述第二训练节点进行数据恢复。

11.根据权利要求 9 所述的装置，其特征在于，所述训练集群在一轮训练结束时，将所述训练集群中所有训练节点的模型参数进行平均值计算，以计算结果作为该轮训练的检查点数据。

12.根据权利要求 9 所述的装置，其特征在于，所述装置还包括初始化模块，所述初始化模块用于：

指示所述第一训练节点下载初始检查点数据；

指示所述第一训练节点向所述第二训练节点广播所述初始检查点数据。

13.根据权利要求 9-12 中任一项所述的装置，其特征在于，所述装置还包括数据上传模块，所述数据上传模块用于：

指示所述第一模型训练组上传所述第一模型训练组中各个训练节点的内存中所保存的检查点数据到远端存储服务。

14.根据权利要求 13 所述的装置，其特征在于，所述存储控制模块还用于：

在所述第一模型训练组上传检查点数据的过程中，禁止删除所述第一模型训练组中各个训练节点的内存中已保存的检查点数据，以及，禁止使用新的检查点数据覆盖所述已保存的检查点数据。

15.根据权利要求 13 所述的装置，其特征在于，所述存储控制模块还用于：

在所述第一模型训练组上传检查点数据的过程中，停止将新的检查点数据保存到所述第一模型训练组的训练节点的内存中。

16.根据权利要求 15 所述的装置，其特征在于，所述存储控制模块还用于：

在所述第一模型训练组上传检查点数据完成后，将所述第二模型训练组的训练节点的内存中所保存的所述新的检查点数据，同步到所述第一模型训练组的训练节点的内存中。

17.一种计算设备集群，其特征在于，包括至少一个计算设备，每个计算设备包括处理器和存储器；

所述至少一个计算设备的处理器用于执行所述至少一个计算设备的存储器中存储的指令，以使得所述计算设备集群执行如权利要求 1-8 中任一项所述的方法。

18.一种包含指令的计算机程序产品，其特征在于，当所述指令被计算设备集群运行时，使得所述计算设备集群执行如权利要求的 1-8 中任一项所述的方法。

19. 一种计算机可读存储介质，其特征在于，包括计算机程序指令，当所述计算机程序指令由计算设备集群执行时，所述计算设备集群执行如权利要求 1-8 中任一项所述的方法。