

一种数据处理方法、系统及相关设备

本申请要求于 2021 年 07 月 14 日提交中国专利局、申请号为 2021107952833、申请名称为“一种数据处理方法、系统及相关设备”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

本发明涉及处理器技术领域，尤其涉及一种数据处理方法、系统及相关设备。

背景技术

Pytorch 是一个开源的机器学习库，用于自然语言处理等应用程序，Pytorch 框架支持对源张量的视图类框架操作以得到视图张量，从而可以有效减少显示的张量数据拷贝所带来的性能消耗，其中，源张量和视图张量共享同一内存。

视图类框架操作所对应的算子可以称为视图类框架算子，其主要包括重塑（reshape）类和非连续类，reshape 类算子包括 view、view_as、squeeze、unsqueeze、flatten 等框架算子，这些算子对应生成的视图张量称为连续张量，其特征在于视图张量的元素按照行优先展开时，对应的索引内存与源张量一致，是连续分布的。非连续类算子包括转置（transpose）、切片（narrow）、扩张（expand）等框架算子，这些算子对应生成的视图张量称为非连续张量，其特征在于视图张量的元素按照行优先展开时，对应的索引内存与源张量不同，在共享内存上呈现非连续分布。

在当前应用中，通常都需要将非连续张量转换为连续张量，目前的转换方案主要是将非连续张量从设备侧（例如神经网络处理器（neural-network processing unit, NPU）芯片）拷贝至主机，由主机完成转换后再将其拷贝回设备侧。综上，在目前对非连续张量转换为连续张量的方案中，其转换效率较低，对设备的硬件要求较高，性能消耗较大。

因此，如何提高张量转换效率，减小转换过程中对设备的硬件依赖，提高设备转换性能是目前亟待解决的问题。

发明内容

本发明实施例公开了一种数据处理方法、系统及相关设备，通过对非连续场景推导，从而确定与该非连续场景匹配的操作并执行该操作完成转换过程，可以有效提升非连续张量转换为连续张量的转换效率，减小对设备硬件的依赖，提高转换性能。

第一方面，本申请提供了一种数据处理方法，所述方法由数据处理系统执行，所述系统包括处理器和计算核，该方法包括：所述处理器获取第一数据的元数据和第二数据的元数据，所述第二数据由所述第一数据经过第一操作得到，所述第二数据中每一行相邻位置的元素对应的内存地址是非连续的；所述处理器将所述第二数据的元数据与第一数据的元数据进行比较，确定所述第一操作；所述处理器确定与所述第一操作匹配的第二操作；所述计算核根据所述第一数据和所述第二操作得到第三数据，所述第三数据中每一行相邻位置的元素对应的内存地址是连续的。

可选的，第一数据可以是源张量，第二数据可以是源张量经过第一操作后得到的非连续张量，第三数据可以是源张量经过第二操作后得到的连续张量，第一操作可以是非连续类框架算子，例如 transpose、narrow、expand 等。

在本申请实施例中，处理器通过对源张量的元数据和非连续张量的元数据进行比较，对非连续张量的产生场景进行推导，从而确定对源张量所执行的操作，进而确定与该操作匹配的张量加速引擎（tensor boost engine, TBE）算子，由计算核对源张量执行该 TBE 算子完成转连续过程，这样，可以提升转换效率，减小对芯片的 AI CPU 的性能依赖，有效提高转换性能。

结合第一方面，在第一方面一种可能的实现方式中，所述处理器将所述第二数据的形状 shape、步幅 stride 和内存偏移量 storage_offset 和所述第一数据的 shape、stride 和 storage_offset 进行一一比对，确定所述第一操作。

在本申请实施例中，张量的元数据可以有效描述张量的特征信息，例如张量的 shape、stride、storage_offset 等特征，处理器通过对源张量的元数据和非连续张量的元数据进行一一比对，可以对非连续张量的产生场景进行合理推导，从而可以准确的确定对源张量所执行的操作。

结合第一方面，在第一方面一种可能的实现方式中，所述处理器遍历算子信息库，所述算子信息库包括多个张量加速引擎 TBE 算子；所述处理器将所述算子信息库中与所述第一操作特征相同的算子确定为与所述第一操作匹配的第二操作。

在本申请实施例中，处理器在确定对源张量所执行的操作后，可以进一步在当前算子信息库中进行寻找，当查找到某个算子的元数据所描述的特征与该操作所对应的特征相同时，可以确定该算子与该操作类型相同，即该算子为与该操作匹配的 TBE 算子。

结合第一方面，在第一方面一种可能的实现方式中，所述处理器将转换命令下发至所述计算核，所述转换命令包括所述第二操作，所述转换命令用于指示所述计算核根据所述第二操作对所述第一数据执行运算得到所述第三数据。

在本申请提供的方案中，处理器在算子信息库中查找到与第一操作匹配的 TBE 算子后，通知计算核对源张量执行该 TBE 算子，从而得到连续张量，该张量中每一行相邻位置的元素对应的索引内存是连续的，这样可以不必依赖 AI CPU 就能完成张量转换过程，减小了对芯片硬件的依赖。

结合第一方面，在第一方面一种可能的实现方式中所述处理器构建第四数据，所述第四数据与所述第一数据的元数据相同，所述第四数据与所述第一数据共享同一内存；所述计算核对所述第四数据执行所述第二操作得到所述第三数据。

在本申请提供的方案中，计算核在进行计算之前，处理器根据确定的 TBE 算子可以获取到该算子所需要的入参信息，该入参信息中包括输入张量，该输入张量可以采用共享内存的方式构建的临时连续张量，这个临时连续张量的元数据与源张量的元数据相同，在完成临时连续张量构建后，计算核方可以执行相应的运算，从而保证计算核能够正确的执行相应的 TBE 算子，完成张量转换过程。

结合第一方面，在第一方面一种可能的实现方式中，所述第一操作包括转置 transpose 算子、切片 narrow 算子、扩张 expand 算子中的至少一个。

结合第一方面，在第一方面一种可能的实现方式中，所述系统包括 NPU、图形处理器（graphics processing unit, GPU）、张量处理单元（tensor processing unit, TPU）、深度学习处理器（data processing unit, DPU）中的至少一个。

结合第一方面，在第一方面一种可能的实现方式中，所述处理器位于所述数据处理系统的主机中，或者所述处理器位于所述数据处理系统的芯片中。

在本申请提供的方案中，对非连续张量的产生场景进行推导的过程可以由数据处理系统中的主机完成，也可以由数据处理系统中的芯片完成，不管是通过主机完成场景推导还是通

过芯片完成场景推导，最终都将由计算核执行 TBE 算子完成非连续张量转换为连续张量，从而减少数据拷贝和对 AI CPU 的硬件依赖，提升转换效率和转换性能。

第二方面，本申请提供了一种数据处理系统，该系统包括：处理器和计算核，

所述处理器，用于获取第一数据的元数据和第二数据的元数据，所述第二数据由所述第一数据经过第一操作得到，所述第二数据中每一行相邻位置的元素对应的内存地址是非连续的；将所述第二数据的元数据与第一数据的元数据进行比较，确定所述第一操作；确定与所述第一操作匹配的第二操作；

所述计算核，用于根据所述第一数据和所述第二操作得到第三数据，所述第三数据中每一行相邻位置的元素对应的内存地址是连续的。

应理解，该芯片可以同时包括多个处理器和计算核，它们可以并行执行各自的任务，互不影响和干涉，本申请对芯片的处理器和计算核数量不作限定。

结合第二方面，在第二方面一种可能的实现方式中，所述处理器，具体用于：将所述第二数据的形状 shape、步幅 stride 和内存偏移量 storage_offset 和所述第一数据的 shape、stride 和 storage_offset 进行一一比对，确定所述第一操作。

结合第二方面，在第二方面一种可能的实现方式中，所述处理器，具体用于：遍历算子信息库，所述算子信息库包括多个 TBE 算子；将所述算子信息库中与所述第一操作特征相同的算子确定为与所述第一操作匹配的第二操作。

结合第二方面，在第二方面一种可能的实现方式中，所述处理器，还用于将转换命令下发至所述计算核，所述转换命令包括所述第二操作，所述转换命令用于指示所述计算核根据所述第二操作对所述第一数据执行运算得到所述第三数据。

结合第二方面，在第二方面一种可能的实现方式中，所述处理器，还用于构建第四数据，所述第四数据与所述第一数据的元数据相同，所述第四数据与所述第一数据共享同一内存；所述计算核，还用于对所述第四数据执行所述第二操作得到所述第三数据。

结合第二方面，在第二方面一种可能的实现方式中，所述第一操作包括转置 transpose 算子、切片 narrow 算子、扩张 expand 算子中的至少一个。

结合第二方面，在第二方面一种可能的实现方式中，所述处理器位于所述系统的主机中，或者所述处理器位于所述数据处理系统的芯片中。

结合第二方面，在第二方面一种可能的实现方式中，所述芯片包括神经网络处理器 NPU、图形处理器 GPU、张量处理单元 TPU、深度学习处理器 DPU 中的至少一个。

第三方面，本申请提供了一种芯片，包括：处理器和计算核，

所述处理器，用于获取第一数据的元数据和第二数据的元数据，所述第二数据由所述第一数据经过第一操作得到，所述第二数据中每一行相邻位置的元素对应的内存地址是非连续的；将所述第二数据的元数据与第一数据的元数据进行比较，确定所述第一操作；确定与所述第一操作匹配的第二操作；

所述计算核，用于根据所述第一数据和所述第二操作得到第三数据，所述第三数据中每一行相邻位置的元素对应的内存地址是连续的。

第四方面，本申请提供了一种计算设备，包括上述第一方面中的任意一种实现方式所提供的数据处理系统。

第五方面,本申请提供了一种计算机存储介质,所述计算机存储介质存储有计算机程序,当所述计算机程序被处理器执行时,可以实现上述第一方面以及结合上述第一方面中的任意一种实现方式所提供的方法。

第六方面,本申请提供了一种计算机程序产品,该计算机程序包括指令,当该计算机程序被计算机执行时,使得计算机可以执行上述第一方面以及结合上述第一方面中的任意一种实现方式所提供的方法。

附图说明

为了更清楚地说明本发明实施例技术方案,下面将对实施例描述中所需要使用的附图作简单地介绍,显而易见地,下面描述中的附图是本发明的一些实施例,对于本领域普通技术人员来讲,在不付出创造性劳动的前提下,还可以根据这些附图获得其他的附图。

图1是本申请实施例提供的一种张量转换的示意图;

图2是本申请实施例提供的一种系统结构的示意图;

图3是本申请实施例提供的一种数据处理方法的流程图;

图4是本申请实施例提供的一种转置操作的示意图;

图5是本申请实施例提供的一种转置算子的入参信息示意图;

图6是本申请实施例提供的另一种张量转换过程的示意图;

图7是本申请实施例提供的一种芯片结构的示意图。

具体实施方式

下面结合附图对本申请实施例中的技术方案进行清楚、完整的描述,显然,所描述的实施例仅仅是本申请一部分实施例,而不是全部的实施例。

首先,结合附图对本申请中所涉及的部分用语和相关技术进行解释说明,以便于本领域技术人员理解。

元数据为描述实际数据的数据,用于描述实际数据的属性信息,可以是实际数据的文件名或者该实际数据的存储地址指针等,例如,张量的元数据可以描述张量的形状、维数、格式等特征信息,同时,元数据还可以具有对应的标识,用于对该元数据进行标识,元数据以及其对应的标识可以构成键值对,每组键值对可以包括关键字(key)以及该key对应的值(value),value即为元数据本身,key用于对value进行标识。

主机又可以称为客户端,其连接了硬盘、硬盘子系统或者文件服务器,并能存储数据和IO访问的计算机系统,具体可以包括物理机、虚拟机、容器等,用于与设备进行通信并进行数据处理,例如应用服务器、多处理器机器、工作站、个人电脑等。

设备为集成了乘加、激活函数、二维数据运算、解压缩等模块的处理芯片,能够加速神经网络的运算,有效提高神经网络运算的效率,例如NPU、GPU、TPU、DPU等,处理芯片中可以包含多个处理器和计算核,它们可以并行执行各自的任務。

目前为了减少显示的张量数据拷贝带来的性能消耗,Pytorch框架支持对源张量的视图类框架操作以得到视图张量,源张量中的元素和视图张量中的元素共享同一块内存。而在实际处理数据时,通常需要将非连续张量转换为连续张量,从而进一步进行数据处理,例如统一计算设备架构(compute unified device architecture, CUDA)会计算非连续张量每个元素的内

存地址,依赖于装填(load)和储存(store)指令,处理芯片(例如GPU)可以访问任意内存位置的元素,并将其存储至一块指定的联系内存区域,保证非连续张量的元素按照行优先展开时,对应的索引内存是连续的,从而完成对该非连续张量的转换。然而当前很多处理芯片无法按照上述这种数据迁移逻辑进行高效的数据拷贝,例如NPU无法通过上述方式完成非连续张量到连续张量的转换,对于这类处理芯片,在进行转连续操作时,通常需要借助主机完成,如图1所示,主机110与设备120通过网络或高速串行计算机扩展总线标准(peripheral component interconnect express, PCIe)接口直接建立连接,主机110可以是一个服务器,设备120可以为插置于该服务器上的NPU加速卡,首先,主机110下发流同步指令至设备120,该指令将阻塞当前流所有任务执行,设备120在接收到指令后,将非连续张量拷贝至主机110,主机110根据当前非连续张量的信息计算其中每个元素的内存地址,然后根据CPU的Load/Store指令将每个元素拷贝至指定的内存区域,在完成拷贝之后可以实现对该非连续张量转换为连续张量,然后将该连续张量再拷贝至设备120,最后主机110下发结束流同步指令至设备120,释放相关资源。可以看出,在对非连续张量进行转换时,阻碍了其它任务正常执行,其次张量数据需要在主机和设备之间来回拷贝,拷贝效率较低且对主机的性能消耗较大,整个转换过程性能较差。

基于上述,本申请提供了一种数据处理方法,利用处理器对非连续张量的产生场景进行推导,确定与该非连续场景匹配的操作,并由AI core执行该操作完成数据的重新拷贝,从而实现将非连续张量转换为连续张量,有效提升转换效率,减小对设备硬件,尤其是AI CPU的依赖,提高转换性能。

本申请实施例的技术方案可以应用于任何需要进行非连续张量转换的系统中,尤其适用于对AI CPU依赖较低的场景。

参见图2,图2是本申请提供的一种系统结构的示意图。如图2所示,该系统包括主机210和芯片220,该主机210中可以包括硬件层和软件层,软件层包括客户操作系统2110、任务调度器2120,硬件层包括一个或多个处理器、内存等硬件,该芯片220可以是神经网络处理器NPU、图形处理器GPU、张量处理单元TPU、深度学习处理器DPU中的至少一个。其也包括硬件层和软件层,硬件层包括一个或多个处理器(如AI CPU2220)、一个或多个计算核(AI Core2230)、内存等硬件,软件层包括各种处理单元(例如I/O处理单元2210等)以处理非连续张量转换为连续张量的相关流程,主机210和芯片220可以通过PCIe接口进行连接。在一些实施例中,芯片220可以和主机210位于不同的设备上,在另一些实施例中,芯片220可以以插卡的方式挂载于主机210上。

主机210用于协同芯片完成非连续张量的转换。处理器2130可以是CPU,还可以是其他通用处理器、数字信号处理器(digital signal processing, DSP)、专用集成电路(application specific integrated circuit, ASIC)、现场可编程门阵列(field programmable gate array, FPGA)或者其他可编程逻辑器件、分立门或者晶体管逻辑器件、分立硬件组件等。通用处理器可以是微处理器或者是任何常规的处理器等。

内存2140可以用于存储算子信息库。可以包括只读存储器和随机存取存储器,还可以是易失性存储器或非易失性存储器,或可包括易失性和非易失性存储器两者。其中,非易失性存储器可以是只读存储器(read-only memory, ROM)、可编程只读存储器(programmable ROM, PROM)、可擦除可编程只读存储器(erasable PROM, EPROM)、电可擦除可编程只读存储器(electrically EPROM, EEPROM)或闪存。易失性存储器可以是随机存取存储器(random access

memory, RAM), 其用作外部高速缓存。通过示例性但不是限制性说明, 许多形式的 RAM 可用, 例如静态随机存取存储器 (static RAM, SRAM)、动态随机存取存储器 (DRAM)、同步动态随机存取存储器 (synchronous DRAM, SDRAM)、双倍数据速率同步动态随机存取存储器 (double data rate SDRAM, DDR SDRAM)、增强型同步动态随机存取存储器 (enhanced SDRAM, ESDRAM)、同步连接动态随机存取存储器 (synchlink DRAM, SLDRAM) 和直接内存总线随机存取存储器 (direct rambus RAM, DR RAM)。

在一种可能的实现方式中, 主机 210 中的任务调度器 2120 将张量转换任务发送至处理器 2130, 处理器 2130 从内存 2140 中提取当前非连续张量的元数据, 该元数据包括非连续张量的形状、步幅、内存偏移量等信息, 然后与源张量的元数据进行比较并进行推导, 例如可以比较 shape 中某些轴的元素是否减少、stride 信息是否不变等, 确定产生该非连续张量的场景, 以及与该场景对应的第一操作, 即对源张量执行该第一操作可以得到该非连续张量, 然后处理器 2130 遍历内存 2140 中的算子信息库, 查找到与该第一操作类型相同的 TBE 算子, 并获知该 TBE 算子所需的信息, 之后处理器 2130 通过 PCIe 接口向芯片 220 下发指令, 在芯片 220 中通过共享非连续张量内存的方式构建一个临时连续张量, 并将该 TBE 算子下发至 AI Core 2230, AI Core 2230 调度该 TBE 算子对该临时连续张量进行运算, 将内存 2240 中非连续张量的每个元素重新拷贝至指定的区域, 得到一个连续张量, 该连续张量每一行中相邻两个元素对应的索引内存在内存 2240 中是连续的。

可以看出, 在进行张量转换时, 利用已知的非连续张量和源张量, 对非连续张量的产生场景进行推导, 从而可以推导得到与该场景对应的操作, 然后根据该操作可以从算子信息库中映射得到相匹配的 TBE 算子, 最后由 AI Core 执行该 TBE 算子, 生成连续张量, 不依赖于 AI CPU 等硬件性能, 可以有效提升转换效率, 提高转换性能。

结合图 2 所示的芯片的示意图, 下面将描述本申请实施例提供的数据处理方法, 参阅图 3, 图 3 示出了本申请实施例提供的一种数据处理方法的流程示意图, 该方法可以应用于如上述图 2 所示的数据处理系统中。该方法具体可以包括:

S301: 处理器获取第一数据的元数据和第二数据的元数据。

具体地, 在本申请实施例中, 第一数据可以是源张量数据, 源张量为一种 n 维的数据结构, 其具体形式包括标量、矢量、矩阵等, 如第 0 维张量即为一个标量, 源张量的元数据即为描述该源张量的数据, 包括张量的 shape、stride、storage_offset 等, 相应的, 第二数据可以是非连续张量, 该非连续张量中的元素按照行优先展开时, 每一行相邻位置的元素对应的内存地址是非连续的。

另外, 处理器可以是上述图 2 所示的主机中处理器 2130, 即下述步骤 S302 中的场景推导和步骤 S303 中的算子映射匹配由主机完成, 处理器也可以是上述图 2 所示的 AI CPU 2220, 即步骤 S302 中的场景推导和步骤 S303 中的算子映射匹配由芯片完成, 本申请对此不作限定。

需要说明的是, 第二数据是由第一数据经过第一操作得到的, 即上述非连续张量是对源张量执行第一操作得到的。第一操作为非连续操作, 例如, 在 Pytorch 架构下, 第一操作为对源张量执行非连续视图类框架算子对应的操作, 可选的, 非连续视图类框架算子包括 transpose、narrow、expand 等。

示例性的, 以第一操作为转置类操作为例, 如图 4 所示, 源张量 410 包括视图信息和基信息, 其中, 视图信息和基信息一致, 其 shape 都为 {2,3,5}, stride 都为 {15,5,1}, 该源张量是一个连续张量, 现在利用非连续操作单元 420 对该源张量 410 执行转置操作, 该转置操作

为 (1,2), 即对源张量的后两轴进行转置, 产生一个非连续视图张量 430, 可以看出, 非连续视图张量 430 中的视图信息与基信息不一致, 且非连续视图张量 430 与源张量 410 相比, 视图信息中的 shape 和 stride 在后两轴发生了对调。

应理解, 在对源张量执行非连续类视图类框架操作得到视图张量时, 仅仅是改变了源张量的元数据, 源张量和视图张量还是共享同一内存, 即源张量中的元素和视图张量的元素是一样的, 而且所占用的内存也是相同的。

S302: 处理器将第二数据的元数据与第一数据的元数据进行比较, 确定所述第一操作。

具体地, 处理器在获取到源张量的元数据和非连续张量的元数据后, 通过对源张量的各个特征 (如上述 shape、stride 和 storage_offset) 和非连续张量的各个特征进行一一比对, 确定对源张量所执行的非连续操作。

示例性的, 如上述图 4 所示, 处理器在已知非连续视图张量 430 和源张量 410 的情况下, 需要对非连续操作单元 420 进行推导, 处理器通过对非连续视图张量 430 与源张量 410 的 shape 进行比较, 可以确定张量的维度信息没有发生改变, 都是三维张量; 处理器将非连续视图张量 430 的视图信息 stride 进行排序得到 stride_sorted{15,5,1}, 同时调整 shape 的对应轴顺序得到 shape_sorted{2,3,5}, 两者与基信息的 shape 和 stride 均一致, 此外, storage_offset 为 0, 没有发生变化, 这三个特征与转置类操作特征吻合, 处理器可以确定非连续操作单元 420 是一个转置操作单元, 且转置的是后两轴。

同理, 对于其它非连续性操作也可以通过上述逻辑进行推导, 例如, 对于非连续性操作为 narrow 时, 处理器通过比对 shape, 发现 shape 中某些轴的元素减少, 通过比对 stride, 发现 stride 不变, 则处理器可以确定非连续性操作为 narrow, 然后通过比对 dim 信息确定哪些轴元素减少, 即该非连续性操作是发生在哪些轴的, 最后通过比对 storage_offset 信息, 推导出切片起始位置 start 与切分长度 length。

S303: 处理器确定与所述第一操作匹配的第二操作。

具体地, 处理器通过对非连续张量产生场景进行推导, 识别出非连续操作类别之后, 需要进一步确定与该非连续操作匹配的第二操作, 即确定是否存在某个张量加速引擎 (tensor boost engine, TBE) 算子的特征与该非连续操作对应的特征相同。

应理解, TBE 算子是由 TBE 语言编写的, 可以直接被计算核 AI Core 调用执行, 从而生成一个连续张量, 在本申请实施例中, 每一个非连续操作对应一个 TBE 算子, 例如, 转置操作对应 transpose 算子。

处理器通过遍历当前算子信息库, 寻找与该非连续操作匹配的 TBE 算子, 例如, 在上述图 4 所示的场景中, 非连续操作为转置操作, 特征为张量维度信息不变、张量的 shape 和 stride 在后两轴发生对调、张量的 storage_offset 为 0, transpose 算子的特征为: 张量维度不变、张量指定轴发生 shape 和 stride 信息的置换、张量的 storage_offset 为 0 且不变, 两者完全相同, 因此, 处理器可以确定与该转置操作匹配的 TBE 算子是 transpose, 直接将算子信息库中的 transpose 算子确定为与非连续操作匹配的算子。

进一步的, 处理器在确定与非连续操作匹配的 TBE 算子后, 可以获取到该 TBE 算子所需的入参信息, 示例性的, 如图 5 所示, 是 transpose 算子所需的入参信息示意图, 其入参信息包括输入张量、结果张量和转置的轴信息, 其中输入张量是采用共享内存方式构建的临时连续张量, 结果张量接收算子执行结果, 是根据非连续张量的视图信息新建的空的连续张量, 转置的轴信息与推导得到的转置操作对应的转置轴信息相同。

需要说明的是, 处理器在确定第二操作之后, 若在算子信息库中没有找到与之匹配的 TBE

算子，可以由研发人员编写与第二操作相匹配的 TBE 算子，并将其加入算子信息库中，这样在软件层面更改算子实现，可以有效扩展适用场景，提高转换的灵活性，充分发挥 AI Core 的性能，解除对 AI CPU 的硬件依赖。

S304：计算核根据所述第一数据和所述第二操作得到第三数据。

具体地，处理器在确定与非连续操作匹配的 TBE 算子之后，将该 TBE 算子下发至计算核 AI Core，计算核 AI Core 通过执行该 TBE 算子得到连续张量。

应理解，TBE 算子在算子信息库中是以文件的形式存在的，文件记录了 TBE 算子的入参信息，处理器将该 TBE 算子对应的文件发送给计算核 AI Core，计算核 AI Core 通过执行该 TBE 算子对应的文件从而输出连续张量。

在一种可能的实现方式中，处理器在将 TBE 算子下发至计算核 AI Core 之前，根据算子所需的入参信息以共享内存的方式构建临时连续张量，该临时连续张量元数据与源张量的元数据相同，且与源张量共享同一内存，即该临时连续张量可以理解为源张量的复原。

示例性的，在上述图 4 所示的场景中，处理器根据 transpose 算子的入参信息构建临时连续张量作为输入张量 610，如图 6 所示，输入张量 610 与源张量 410 相同，利用 transpose 算子 620 对输入张量 610 进行计算，其转置的轴信息是{0,2,1}，表示对 shape 后两轴进行转置，计算后得到输出张量 630，该输入张量 630 是一个连续张量，即输出张量 630 亦可以称为连续张量 630，可以看出，连续张量 630 与非连续视图张量 430 的 shape 一致，但是 stride 发生了改变，对于连续张量 630 来说，该张量中的元素按照行优先展开时，每一行相邻位置的元素对应的内存地址是连续的。

值得说明的是，计算核 AI Core 在执行 TBE 算子时，在主存中将重新确定一块内存区域，将源张量中的元素按照连续张量所确定的内存读取方式将其有序迁移至该内存区域中，从而保证连续张量在按照行优先展开时，相邻元素的内存地址是连续的。

可以看出，执行本发明实施例，不需要改变 Pytorch 框架本身的视图类框架算子语义，通过对非连续张量产生场景进行推导确定非连续操作，进而确定与之匹配的 TBE 算子，最后利用计算核 AI Core 执行该 TBE 算子生成内存连续分布的连续张量，不依赖于 AI CPU 硬件性能，提升了转换效率和转换性能，且在软件层面更加灵活，便于扩展，能充分发挥计算核 AI Core 的性能。

应理解，上述方法不仅适用于 Pytorch 框架，对于其它具有转置、切片等非连续操作的 AI 框架都可以利用本发明所提供的方法进行场景反推，从而根据反推结果完成对非连续张量的转换。

还应理解，上述仅以转置操作为例说明了如何进行场景反推以及如何根据反推结果完成张量转连续过程，对于其它非连续性操作，例如 expand 等也可以按照同样的方法进行场景推导以及张量转换。

本申请所提供的数据处理方法可以广泛应用于 Pytorch 的模型训练与推理场景中，能够显著改善模型训练与推理的效率，减少训练耗时，加速模型训练。可以理解，若模型训练涉及将非连续张量转换为连续张量，本申请由 AI Core 执行 TBE 算子进行内存拷贝从而实现转连续，相较于主机进行内存拷贝完成转连续过程，可以减少数据来回拷贝时延，提高转换效率，从而有效改善模型训练和推理效率，产生巨大的商业价值。

上述详细阐述了本申请实施例的方法，为了便于更好的实施本申请实施例的上述方案，相应地，下面还提供用于配合实施上述方案的相关设备。

参见图 7，图 7 是本申请实施例提供的一种芯片的结构示意图。如图 7 所示，该芯片 700 包括：处理器 710 和计算核 720，该芯片 700 可以是 NPU 芯片、GPU 芯片、TPU 芯片或其它 AI 芯片，其中可以包含多个处理器和计算核，它们可以并行执行各自的任務，图 7 中以一个为例。其中，本发明实施例中所述的芯片的功能，可以参见图 2-图 6 中所述的发明实施例中的相关描述，此处不再赘述。

本申请实施例提供了一种计算设备，该计算设备可以是上述图 2 所示的数据处理系统中的主机，芯片以插卡的方式集成在主机上。主机和芯片可以协同执行如图 2-图 6 中所述的发明实施例中的相关描述，此处不再赘述。

本申请实施例还提供一种计算机可读存储介质，其上存储有计算机程序，该程序被处理器执行时，可以实现上述方法实施例中记载的任意一种的部分或全部步骤。

本申请实施例还提供了一种计算机程序产品，当其在计算机或处理器上运行时，使得计算机或处理器执行上述任一个方法中的一个或多个步骤。上述所涉及的设备的各组成模块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吋，可以存储在所述计算机可读存储介质中。

在上述实施例中，对各个实施例的描述各有侧重，某个实施例中沒有详述的部分，可以参见其它实施例的相关描述。

应理解，本文中涉及的第一、第二、第三、第四以及各种数字编号仅为描述方便进行的区分，并不用来限制本申请的范围。

应理解，本文中术语“和/或”，仅仅是一种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 这三种情况。另外，本文中字符“/”，一般表示前后关联对象是一种“或”的关系。

还应理解，在本申请的各种实施例中，上述各过程的序号的大小并不意味着执行顺序的先后，各过程的执行顺序应以其功能和内在逻辑确定，而不对本申请实施例的实施过程构成任何限定。

本领域普通技术人员可以意识到，结合本文中所公开的实施例描述的各示例的单元及算法步骤，能够以电子硬件、或者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，上述描述的系统、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。

在本申请所提供的几个实施例中，应该理解到，所揭露的系统、装置和方法，可以通过其它的方式实现。例如，以上所描述的装置实施例仅仅是示意性的，例如，所述单元的划分，仅仅为一种逻辑功能划分，实际实现时可以有另外的划分方式，例如多个单元或组件可以结合或者可以集成到另一个系统，或一些特征可以忽略，或不执行。另一点，所显示或讨论的相互之间的耦合或直接耦合或通信连接可以是通过一些接口，装置或单元的间接耦合或通信连接，可以是电性，机械或其它的形式。

所述作为分离部件说明的单元可以是或者也可以不是物理上分开的，作为单元显示的部件可以是或者也可以不是物理单元，即可以位于一个地方，或者也可以分布到多个网络单元上。可以根据实际的需要选择其中的部分或者全部单元来实现本实施例方案的目的。

另外，在本申请各个实施例中的各功能单元可以集成在一个处理单元中，也可以是各个

单元单独物理存在，也可以两个或两个以上单元集成在一个单元中。

所述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，可以存储在一个计算机可读存储介质中。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。而前述的存储介质包括：U 盘、移动硬盘、只读存储器（Read-Only Memory, ROM）、随机存取存储器（Random Access Memory, RAM）、磁碟或者光盘等各种可以存储程序代码的介质。

本申请实施例方法中的步骤可以根据实际需要进行顺序调整、合并和删减。

本申请实施例装置中的模块可以根据实际需要进行合并、划分和删减。

以上所述，以上实施例仅用以说明本申请的技术方案，而非对其限制；尽管参照前述实施例对本申请进行了详细的说明，本领域的普通技术人员应当理解：其依然可以对前述各实施例所记载的技术方案进行修改，或者对其中部分技术特征进行等同替换；而这些修改或者替换，并不使相应技术方案的本质脱离本申请各实施例技术方案的范围。

权利要求书

1.一种数据处理方法，其特征在于，所述方法由数据处理系统执行，所述数据处理系统包括处理器和计算核，所述方法包括：

所述处理器获取第一数据的元数据和第二数据的元数据，所述第二数据由所述第一数据经过第一操作得到，所述第二数据中每一行相邻位置的元素对应的内存地址是非连续的；

所述处理器将所述第二数据的元数据与第一数据的元数据进行比较，确定所述第一操作；

所述处理器确定与所述第一操作匹配的第二操作；

所述计算核根据所述第一数据和所述第二操作得到第三数据，所述第三数据中每一行相邻位置的元素对应的内存地址是连续的。

2.如权利要求 1 所述的方法，其特征在于，所述处理器将所述第二数据的元数据与第一数据的元数据进行比较，确定所述第一操作，包括：

所述处理器将所述第二数据的形状 shape、步幅 stride 和内存偏移量 storage_offset 和所述第一数据的 shape、stride 和 storage_offset 进行一一比对，确定所述第一操作。

3.如权利要求 1 或 2 所述的方法，其特征在于，所述处理器确定与所述第一操作匹配的第二操作，包括：

所述处理器遍历算子信息库，所述算子信息库包括多个张量加速引擎 TBE 算子；

所述处理器将所述算子信息库中与所述第一操作特征相同的算子确定为与所述第一操作匹配的第二操作。

4.如权利要求 1-3 任一项所述的方法，其特征在于，在所述计算核得到所述第三数据之前，所述方法还包括：

所述处理器将转换命令下发至所述计算核，所述转换命令包括所述第二操作，所述转换命令用于指示所述计算核根据所述第二操作对所述第一数据执行运算得到所述第三数据。

5.如权利要求 1-4 任一项所述的方法，其特征在于，所述计算核得到所述第三数据包括：

所述处理器构建第四数据，所述第四数据与所述第一数据的元数据相同，所述第四数据与所述第一数据共享同一内存；

所述计算核对所述第四数据执行所述第二操作得到所述第三数据。

6.如权利要求 1-5 任一项所述的方法，其特征在于，所述第一操作包括转置 transpose 算子、切片 narrow 算子、扩张 expand 算子中的至少一个。

7.如权利要求 1-6 任一项所述的方法，其特征在于，所述数据处理系统还包括主机和芯片，所述处理器位于所述主机中，或者所述处理器位于所述芯片中。

8.如权利要求 7 所述的方法，其特征在于，所述芯片包括神经网络处理器 NPU、图形处理器 GPU、张量处理单元 TPU、深度学习处理器 DPU 中的至少一个。

9.一种数据处理系统，其特征在于，包括：处理器和计算核；

所述处理器，用于获取第一数据的元数据和第二数据的元数据，所述第二数据由所述第一数据经过第一操作得到，所述第二数据中每一行相邻位置的元素对应的内存地址是非连续的；将所述第二数据的元数据与第一数据的元数据进行比较，确定所述第一操作；确定与所述第一操作匹配的第二操作；

所述计算核，用于根据所述第一数据和所述第二操作得到第三数据，所述第三数据中每一行相邻位置的元素对应的内存地址是连续的。

10.如权利要求9所述的数据处理系统，其特征在于，所述处理器，具体用于：

将所述第二数据的形状 shape、步幅 stride 和内存偏移量 storage_offset 和所述第一数据的 shape、stride 和 storage_offset 进行一一比对，确定所述第一操作。

11.如权利要求9或10所述的数据处理系统，其特征在于，所述处理器，具体用于：

遍历算子信息库，所述算子信息库包括多个 TBE 算子；

将所述算子信息库中与所述第一操作特征相同的算子确定为与所述第一操作匹配的第二操作。

12.如权利要求9-11任一项所述的数据处理系统，其特征在于，

所述处理器，还用于将转换命令下发至所述计算核，所述转换命令包括所述第二操作，所述转换命令用于指示所述计算核根据所述第二操作对所述第一数据执行运算得到所述第三数据。

13.如权利要求9-12任一项所述的数据处理系统，其特征在于，

所述处理器，还用于构建第四数据，所述第四数据与所述第一数据的元数据相同，所述第四数据与所述第一数据共享同一内存；

所述计算核，还用于对所述第四数据执行所述第二操作得到所述第三数据。

14.如权利要求9-13任一项所述的数据处理系统，其特征在于，所述第一操作包括转置 transpose 算子、切片 narrow 算子、扩张 expand 算子中的至少一个。

15.如权利要求9-14任一项所述的数据处理系统，其特征在于，所述数据处理系统还包括主机和芯片，所述处理器位于所述主机中，或者所述处理器位于所述芯片中。

16.如权利要求15所述的系统，其特征在于，所述芯片包括神经网络处理器 NPU、图形处理器 GPU、张量处理单元 TPU、深度学习处理器 DPU 中的至少一个。

17.一种芯片，其特征在于，包括：处理器和计算核，

所述处理器，用于获取第一数据的元数据和第二数据的元数据，所述第二数据由所述第一数据经过第一操作得到，所述第二数据中每一行相邻位置的元素对应的内存地址是非连续的；将所述第二数据的元数据与第一数据的元数据进行比较，确定所述第一操作；确定与所述第一操作匹配的第二操作；

所述计算核，用于根据所述第一数据和所述第二操作得到第三数据，所述第三数据中每一行相邻位置的元素对应的内存地址是连续的。

18.一种计算设备，其特征在于，包括：

如权利要求 1-8 任一项所述的数据处理系统。

19.一种计算机可读存储介质，其特征在于，所述计算机可读存储介质存储有计算机程序，该计算机程序被处理器执行时实现上述权利要求 1-8 任一项所述的方法。

20.一种计算机程序，其特征在于，所述计算机程序包括指令，当所述计算机程序被处理器执行时，使得所述处理器执行如上述权利要求 1-8 任一项所述的方法。