

(12) 按照专利合作条约所公布的国际申请

(19) 世界知识产权组织
国际局

(43) 国际公布日
2020 年 8 月 20 日 (20.08.2020)



(10) 国际公布号
WO 2020/164644 A2

- (51) 国际专利分类号:
G06N 3/08 (2006.01)
- (21) 国际申请号: PCT/CN2020/084416
- (22) 国际申请日: 2020 年 4 月 13 日 (13.04.2020)
- (25) 申请语言: 中文
- (26) 公布语言: 中文
- (30) 优先权:
- | | | |
|----------------|-------------------------|----|
| 201910114927.0 | 2019年2月14日 (14.02.2019) | CN |
| 201910115130.2 | 2019年2月14日 (14.02.2019) | CN |
| 201910115162.2 | 2019年2月14日 (14.02.2019) | CN |
| 201910114967.5 | 2019年2月14日 (14.02.2019) | CN |

- (71) 申请人: 上海寒武纪信息科技有限公司 (SHANGHAI CAMBRICON INFORMATION TECHNOLOGY CO., LTD) [CN/CN]; 中国上海市浦东新区同汇路 168 号 B 座 6 层, Shanghai 201306 (CN)。
- (72) 发明人: 周玉松 (ZHOU, Yusong); 中国上海市浦东新区同汇路 168 号 B 座 6 层, Shanghai 201306 (CN)。张潇 (ZHANG, Xiao); 中国上海市浦东新区同汇路 168 号 B 座 6 层, Shanghai 201306 (CN)。吴林阳 (WU, Linyang); 中国上海市浦东新区同汇路 168 号 B 座 6 层, Shanghai 201306 (CN)。俞烨昊 (YU, Yehao); 中国上海市浦东新区同汇路

168 号 B 座 6 层, Shanghai 201306 (CN)。许云龙 (XU, Yunlong); 中国上海市浦东新区同汇路 168 号 B 座 6 层, Shanghai 201306 (CN)。

- (74) 代理人: 北京华进京联知识产权代理有限公司 (ACIP LAW OFFICES); 中国北京市海淀区知春路 7 号致真大厦 A1403, Beijing 100191 (CN)。
- (81) 指定国 (除另有指明, 要求每一种可提供的国家保护): AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JO, JP, KE, KG, KH, KN, KP, KR, KW, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, WS, ZA, ZM, ZW。
- (84) 指定国 (除另有指明, 要求每一种可提供的地区保护): ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), 欧亚 (AM, AZ, BY, KG, KZ, RU, TJ, TM), 欧洲 (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT,

(54) Title: NEURAL NETWORK MODEL SPLITTING METHOD, APPARATUS, COMPUTER DEVICE AND STORAGE MEDIUM

(54) 发明名称: 神经网络模型拆分方法、装置、计算机设备和存储介质

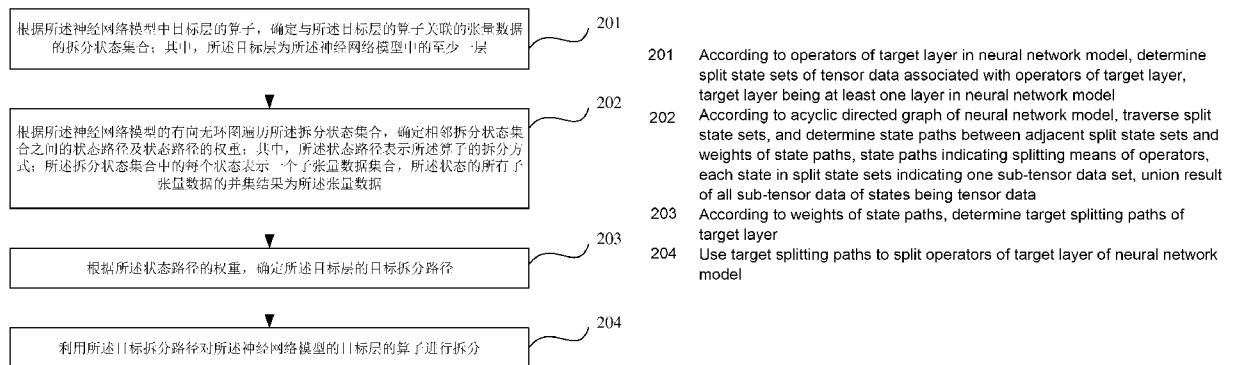


图 2

(57) Abstract: Disclosed are a neural network model splitting method and related products, wherein one operator is split into multiple smaller-scale sub-operators, for directly calling a computing library under a single-core architecture, in order to avoid extra workload of re-implementation.

(57) 摘要: 本公开披露一种神经网络模型的拆分方法及相关产品, 本方案把一个算子拆分成多个规模更小的子算子, 这样可以直接调用单核架构下的计算库, 避免了重新实现的额外工作量。

[见续页]



WO 2020/164644 A2

RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG)。

本国际公布：

- 不包括国际检索报告，在收到该报告后将重新公布(细则48.2(g))。
- 包括关于请求恢复一项或多项优先权要求的信息(细则26之二.3和48.2(b)(vii))。

神经网络模型拆分方法、装置、计算机设备和存储介质

相关申请

本申请要求 2019 年 02 月 14 日申请的申请号为 201910114927.0, 名称为“一种神经网络模型的拆分方法及相关产品”的中国专利申请的优先权, 2019 年 02 月 14 日申请的申请号为 201910114967.5, 名称为“一种神经网络模型的拆分方法及相关产品”的中国专利申请的优先权, 2019 年 02 月 14 日申请的申请号为 201910115130.2, 名称为“一种神经网络模型的拆分方法及相关产品”的中国专利申请的优先权, 以及 2019 年 02 月 14 日申请的申请号为 201910115162.2, 名称为“一种神经网络模型的拆分方法及相关产品”的中国专利申请的优先权, 在此将其全文引入作为参考。

技术领域

本申请涉及人工智能技术领域, 特别是涉及一种神经网络模型的拆分方法及相关产品。

背景技术

近年来, 深度学习加速器被不断提出, 并如同通用处理器一样, 正在由单核向多核扩展。这种扩展后的多核结构可以在训练阶段支持数据并行的方式来提高数据吞吐量, 加快训练速度。然而, 在推理阶段, 相比吞吐量深度神经网络对端到端的时延有着更高的要求, 这往往决定了加速器在某个场景下的可用性。传统的数据并行方案不能满足推理场景下对加速器小数据、低延迟的要求。

发明内容

基于此，有必要针对上述技术问题，提出一种神经网络模型的拆分方法及相关产品。

为实现上述目的，本公开提供一种神经网络模型拆分方法，其中，所述方法包括：

根据所述神经网络模型中目标层的算子，确定与所述目标层的算子关联的张量数据的拆分状态集合；其中，所述目标层为所述神经网络模型中的至少一层；

根据所述神经网络模型的有向无环图遍历所述拆分状态集合，确定相邻拆分状态集合之间的状态路径及状态路径的权重；其中，所述状态路径表示所述算子的拆分方式；所述拆分状态集合中的每个状态表示一个子张量数据集合，所述状态的所有子张量数据的并集结果为所述张量数据；

根据所述状态路径的权重，确定所述目标层的目标拆分路径；

利用所述目标拆分路径对所述神经网络模型的目标层的算子进行拆分。

为实现上述目的，本公开提供一种神经网络模型拆分装置，其中，包括：

拆分状态集合模块，用于根据所述神经网络模型中目标层的算子，确定与所述目标层的算子关联的张量数据的拆分状态集合；其中，所述目标层为所述神经网络模型中的至少一层；

状态路径模块，用于根据所述神经网络模型的有向无环图遍历所述拆分状态集合，确定相邻拆分状态集合之间的状态路径及状态路径的权重；其中，所述状态路径表示所述算子的拆分方式；所述拆分状态集合中的每个状态表示一个子张量数据集合，所述状态的所有子张量数据的并集结果为所述张量数据；

目标拆分路径模块，用于根据所述状态路径的权重，确定所述目标层的目标拆分路径；

拆分模块，用于利用所述目标拆分路径对所述神经网络模型的目标层的算子进行拆分。

为实现上述目的，本公开还提供为实现上述目的，本公开提供一种神经网络模型拆分方法，所述方法包括：

根据所述神经网络模型中目标层的算子，确定与所述目标层的算子关联的张量数据的拆分状态集合；其中，所述目标层为所述神经网络模型中的至少一层；

在所述目标层的算子与关联的拆分状态集合之间插入胶水算子，调整所述算子的张量数据的拆分状态集合中的状态；其中，所述胶水算子用于将所述张量数据按照一拆分方式获得的状态调整成按照任一种拆分方式获得的状态；

根据所述神经网络模型的有向无环图遍历所述拆分状态集合，确定相邻拆分状态集合之间的状态路径及状态路径的权重；其中，所述状态路径表示所述算子的拆分方式；所述拆分状态集合中的每个状态表示一个子张量数据集合，所述状态的所有子张量数据的并集结果为所述张量数据；

根据所述状态路径的权重，确定所述目标层的目标拆分路径；

利用所述目标拆分路径对所述神经网络模型的目标层的算子进行拆分。

为实现上述目的，本公开提供一种神经网络模型拆分装置，所述装置包括：

拆分状态集合确定模块，用于根据所述神经网络模型中目标层的算子，确定与所述目标层的算子关联的张量数据的拆分状态集合；其

中，所述目标层为所述神经网络模型中的至少一层；

插入胶水算子模块，用于在所述目标层的算子与关联的拆分状态集合之间插入胶水算子，调整所述算子的张量数据的拆分状态集合中的状态；其中，所述胶水算子用于将所述张量数据按照一拆分方式获得的状态调整成按照任一种拆分方式获得的状态；

状态路径确定模块，用于根据所述神经网络模型的有向无环图遍历所述拆分状态集合，确定相邻拆分状态集合之间的状态路径及状态路径的权重；其中，所述状态路径表示所述算子的拆分方式；所述拆分状态集合中的每个状态表示一个子张量数据集合，所述状态的所有子张量数据的并集结果为所述张量数据；

目标拆分路径确定模块，用于根据所述状态路径的权重，确定所述目标层的目标拆分路径；

拆分模块，用于利用所述目标拆分路径对所述神经网络模型的目标层的算子进行拆分。

为实现上述目的，本公开还提供一种神经网络模型拆分方法，其中，所述方法包括：

根据所述神经网络模型中目标层的算子，确定与所述目标层的算子关联的张量数据的拆分状态集合；其中，所述目标层为所述神经网络模型中的至少一层；

在所述目标层的算子与关联的拆分状态集合之间插入补偿算子，调整所述算子的输入张量数据的拆分状态集合中的状态；其中，所述补偿算子用于从所述状态的任一子张量数据的相邻子张量数据中获取目标数据，将所述目标数据与所述子张量数据合并；

根据所述神经网络模型的有向无环图遍历所述拆分状态集合，确定相邻拆分状态集合之间的状态路径及状态路径的权重；其中，所述

状态路径表示所述算子的拆分方式；所述拆分状态集合中的每个状态表示一个子张量数据集合，所述状态的所有子张量数据的并集结果为所述张量数据；

根据所述状态路径的权重，确定所述目标层的目标拆分路径；

利用所述目标拆分路径对所述神经网络模型的目标层的算子进行拆分。

为实现上述目的，本公开提供一种神经网络模型拆分装置，所述装置包括：

拆分状态集合模块，用于根据所述神经网络模型中目标层的算子，确定与所述目标层的算子关联的张量数据的拆分状态集合；其中，所述目标层为所述神经网络模型中的至少一层；

插入补偿算子模块，用于在所述目标层的算子与关联的拆分状态集合之间插入补偿算子，调整所述算子的输入张量数据的拆分状态集合中的状态；其中，所述补偿算子用于从所述状态的任一子张量数据的相邻子张量数据中获取目标数据，将所述目标数据与所述子张量数据合并；

状态路径模块，用于根据所述神经网络模型的有向无环图遍历所述拆分状态集合，确定相邻拆分状态集合之间的状态路径及状态路径的权重；其中，所述状态路径表示所述算子的拆分方式；所述拆分状态集合中的每个状态表示一个子张量数据集合，所述状态的所有子张量数据的并集结果为所述张量数据；

目标拆分路径模块，用于根据所述状态路径的权重，确定所述目标层的目标拆分路径；

拆分模块，用于利用所述目标拆分路径对所述神经网络模型的目标层的算子进行拆分。

为实现上述目的，本公开还提供一种神经网络模型拆分方法，所述方法包括：

根据所述神经网络模型中目标层的算子，确定与所述目标层的算子关联的张量数据的拆分状态集合；其中，所述目标层为所述神经网络模型中的至少一层；

在所述目标层的算子与关联的拆分状态集合之间插入胶水算子，调整所述算子的张量数据的拆分状态集合中的状态；其中，所述胶水算子用于将所述张量数据按照一拆分方式获得的状态调整成按照任一种拆分方式获得的状态；

在所述目标层的算子与关联的拆分状态集合之间插入补偿算子，调整所述算子的输入张量数据的拆分状态集合中的状态；其中，所述补偿算子用于从所述状态的任一子张量数据的相邻子张量数据中获取目标数据，将所述目标数据与所述子张量数据合并；

根据所述神经网络模型的有向无环图遍历所述拆分状态集合，确定相邻拆分状态集合之间的状态路径及状态路径的权重；其中，所述状态路径表示所述算子的拆分方式；所述拆分状态集合中的每个状态表示一个子张量数据集合，所述状态的所有子张量数据的并集结果为所述张量数据；

根据所述状态路径的权重，确定所述目标层的目标拆分路径；

利用所述目标拆分路径对所述神经网络模型的目标层的算子进行拆分。

为实现上述目的，本公开提供一种神经网络模型拆分装置，包括：

拆分状态集合模块，用于根据所述神经网络模型中目标层的算子，确定与所述目标层的算子关联的张量数据的拆分状态集合；其中，所述目标层为所述神经网络模型中的至少一层；

插入胶水算子模块，用于在所述目标层的算子与关联的拆分状态

集合之间插入胶水算子，调整所述算子的张量数据的拆分状态集合中的状态；其中，所述胶水算子用于将所述张量数据按照一拆分方式获得的状态调整成按照任一种拆分方式获得的状态；

插入补偿算子模块，用于在所述目标层的算子与关联的拆分状态集合之间插入补偿算子，调整所述算子的输入张量数据的拆分状态集合中的状态；其中，所述补偿算子用于从所述状态的任一子张量数据的相邻子张量数据中获取目标数据，将所述目标数据与所述子张量数据合并；

状态路径模块，用于根据所述神经网络模型的有向无环图遍历所述拆分状态集合，确定相邻拆分状态集合之间的状态路径及状态路径的权重；其中，所述状态路径表示所述算子的拆分方式；所述拆分状态集合中的每个状态表示一个子张量数据集合，所述状态的所有子张量数据的并集结果为所述张量数据；

目标拆分路径模块，用于根据所述状态路径的权重，确定所述目标层的目标拆分路径；

拆分模块，用于利用所述目标拆分路径对所述神经网络模型的目标层的算子进行拆分。

本公开的技术方案可以用较小地开销实现深度学习加速器由单核向多核结构上的扩展，并且能够针对给定的网络和底层加速器特点给出一种高效的拆分方案，该方案能有效降低各种网络在多核加速器上的端到端时延。

附图说明

图 1 为共享存储多核结构的示意图；

图 2 为本公开实施例提供的一种神经网络模型拆分方法流程图之一；

图 3 为串行神经网络模型拆分示意图；

图 4 为本公开实施例提供的一种神经网络模型拆分方法流程图之二；

图 5 为算子与输入张量数据之间引入胶水算子的拆分示意图；

图 6 为补偿示意图；

图 7 为本公开实施例提供的一种神经网络模型拆分方法流程图之三；

图 8 为金字塔拆分示意图；

图 9 为本公开实施例提供的一种神经网络模型拆分方法流程图之四；

图 10 为本公开实施例提供的一种神经网络模型拆分硬件设备示意图。

具体实施方式

下面将结合附图，对本公开实施例中的技术方案进行清楚、完整地描述，更加全面地说明本公开的示例实施例和它们的多种特征及有利细节。应注意的是，图中示出的特征不是必须按照比例绘制。本公开省略了已知材料、组件和工艺技术的描述，从而不使本公开的示例实施例模糊。所给出的示例仅旨在有利于理解本公开示例实施例的实施，以及进一步使本领域技术人员能够实施示例实施例。因而，这些示例不应被理解为对本公开的实施例的范围的限制。

除非另外特别定义，本公开使用的技术术语或者科学术语应当为本公开所属领域内具有一般技能的人士所理解的通常意义。本公开中使用的“第一”、“第二”以及类似的词语并不表示任何顺序、数量或者重要性，而只是用来区分不同的组成部分。此外，在本公开各个实施例中，相同或类似的参考标号表示相同或类似的构件。

下面结合附图，对本公开实施例提供的一种神经网络模型拆分方法及相关产品的具体实施方式进行详细说明。

近年来，得益于深度学习本身在多个领域取得的巨大成功，深度学习加速器成为了快速发展的一个领域。这些新出现的加速器往往在性能功耗比上对比 GPU 有较大的优势。如同通用处理器的发展一样，深度学习加速器同样可以由单核向多核架构扩展，并且这种扩展非常适合深度学习中数据并行的训练方式。数据并行指的是通过把训练的数据集划分成若干份，使用多个处理核分别处理一部分子数据集来加快训练。在多核结构中采用这种方式，每个核并行地处理训练数据中的不同数据集，从而提高整个系统的吞吐量，加快训练速度。因此，多核的加速器架构在保持了每个核良好的性能功耗比的前提下，又可以方便地扩展整个系统的在训练阶段的计算吞吐量。

针对多核处理器结构芯片来说，如图 1 所示，这种共享存储多核结构是一种经典的多核结构。这种结构非常适合数据并行的神经网络训练方法。每个核可以作为数据并行中的一个处理器，分别读取不同的数据然后并行完成神经网络模型的正反向计算。每个核在计算阶段仍能够保持其在之前单核架构下良好的性能功耗比，与此同时整个系统的吞吐量也可以随着核数的扩展而增加。数据并行的问题在于，其扩展性依赖于处理的数据批量的大小。尽管在训练阶段这通常不会是一个问题，但是对于推理阶段这个前提则难以保证。一般来说，用于实时服务领域（包括视频监控，自动驾驶等）的神经网络模型，处理的数据通常是以流的方式串行输入，导致了每次处理的数据规模很小甚至往往是单张图片。在这种情况下，数据并行不能提供任何并行度，所有的工作任务会集中在单个核上，这使得多核带来的计算资源不能转化成处理任务的速度。

当在线下使用数据集完成了神经网络模型的训练后，就会把模型部署到云端的服务器上来处理外界发来的数据，此时的应用场景就由离线训练变成了在线推理。在在线推理阶段，一个非常重要的指标是时延，也就是从服务器收到待处理数据到返回处理后的结果的时间，进一步来说，是使用神经网络模型处理数据的时间。低时延保证云端服务器能够对客户端发来的数据在最短的时间内做出响应，在一些更

加敏感的场景下，直接决定了方案是否可用。因此，在线推理阶段对于加速器的要求就由处理大批量数据、高吞吐量转变为处理小批量数据、低时延。

在这种情况下，传统的数据并行或者模型并行难以有效降低推理任务的时延。对于数据并行来说，大批量数据是前提，这本身与在线推理小批量数据的特点矛盾。对于模型并行来说，它通常是为了解决一个规模很大的神经网络模型超过了单个设备的内存限制而采用的方法，把算子分配到不同的核上并不能降低网络的时延。为了真正能够在多核加速器上降低推理任务的时延，必须寻找一种方法，能够把对小批量数据甚至单个数据的推理计算任务合理地分配到多核架构的各个核上，保证每一时刻都有尽可能多的核参与计算，才能充分利用多核架构的资源。一种方法是把神经网络中的每个算子的计算任务都拆分到多个核上计算，这种方法即使在处理单张图片的推理任务时也能保证每一时刻都有多个核参与计算，从而达到了利用多核资源降低时延的目的。

但是，对于多核加速器来说，还有很多要解决的问题。首先，深度学习加速器通过定制化自身的硬件设计来适配深度学习算法本身的数据并行特征，提高计算吞吐量，加速器往往需要足够的数据规模才能达到较高的计算效率，而算子内的进一步拆分会减小每个核上的计算规模。当拆分达到一定粒度，每个核上计算效率的损失会超过拆分增加并行度所带来的收益。因此，必须在拆分并行和计算效率之间，在保证足够计算效率的同时提供足够的并行度。

另一方面，神经网络模型可以看作是一个由通常数以百计甚至千记的算子所构成的复杂计算图。不同种类的算子内的算法逻辑各不相同，这就导致对这些算子进行拆分的方法也不一样。每个算子的拆分，除了平衡自身的计算效率和并行度，还要考虑和前后算子的搭配，甚至于对全局的影响。深度学习的快速发展带来的是越来越多的大规模复杂网络，通过手动方式寻找一种好的并行方法是不现实的，因此需要一种自动化的方法来保证来对于不同的网络都能够给出一种较好

的拆分并行策略。

此外,还需要考虑的是对于底层加速器的可移植性。对于没有足够良好的可编程性的加速器来说,由单核扩展到多核,并且实现算子内部的拆分并行所带来的修改软件栈的工作量是非常大的。传统的数据并行和模型并行的实现仍然是基于一个处理核完成一个算子的计算任务,所以并不会带来很多额外的工作,而单个算子的跨核并行需要对算子本身实现进行修改,这种修改的难易程度依赖于加速器的可编程性和原有算子实现逻辑的复杂程度。如何减小在多核架构上实现低时延推理过程中的额外开销,缓解实现过程中工作量对于加速器本身可编程性的依赖,使得方法能够在未来对于不同的多核加速器都有一定的通用性也是一个需要考虑的问题。

基于上述分析描述,针对大规模的神经网络模型自动给出一套端到端的拆分方案,本方案把一个算子拆分成多个规模更小的子算子,这样可以直接调用单核架构下的计算库,避免了重新实现的额外工作量。比如:一个激活算子在经过拆分后可以得到许多更小的激活算子,这意味着只需要在多个核上调用原有的单核激活函数完成每个子任务,而不需要修改或者重新实现一个多核版本的激活函数。在这个过程中,既需要兼顾每个算子本身的拆分后的计算效率和并行度,也要考虑上下文算子彼此之间在拆分上的相互配合。最终目标是得到一个能够有效降低整个神经网络模型端到端的推理时延的拆分并行方案。

以自动驾驶应用为例,车辆在自动行驶过程中需要对车载传感器传来的图像、视频、语音等外部信息进行分析处理。为了保证安全性,车辆必须在最短的时间内得到处理的结果,从而做出决策。采用了多核处理器结构芯片的车辆,可以使用本方案把神经网络模型处理小批量外部信息的计算负载均衡地分配到多个处理器核上,在规定的响应时间内完成对信息的处理,返回处理结果辅助车辆自动行驶。本技术方案可以用较小地开销实现深度学习加速器由单核向多核结构上的扩展,并且该方案能有效降低各种网络在多核加速器上的端到端时延。

在上述应用场景中，多核处理器结构芯片设置在车辆上。实际中，多核处理器结构芯片可以设置在云端服务器上，车辆可以通过 3G/4G、WIFI 等网络将车载传感器传来的图像、视频、语音等外部信息发送至云端服务器。云端服务器使用本方案把神经网络模型处理小批量外部信息的计算负载均衡地分配到多个处理核上。在车辆行驶规定的响应时间内，云端服务器将处理结果通过 3G/4G、WIFI 等网络反馈至车辆。在实际中，车载传感器采集到的外部信息的规模不同。在应用之前，根据不同规模的外部信息，车载处理器利用本方案确定相应的算子拆分路径。将不同规模的外部信息对应的算子拆分方案存储对应区域，多核处理器结构芯片获取外部信息后调出对应的算子拆分路径来对神经网络模型中的算子进行拆分，把外部信息的计算负载均衡地分配到多个处理器核上。

通常，上层框架需要调用计算库来得到神经网络模型中每个算子在处理器上的指令实现。具体地，框架把每个算子的类型、参数告知计算库，计算库返回每个算子在处理器上执行所需要的机器指令。框架通过驱动把数据和所述机器指令加载到处理器上，启动处理器并完成所述算子的计算。

如果把所述算子的计算平台由单核加速器变成有着相似甚至相同核结构的多核加速器，相应地，需要对计算库进行重新设计，使其能够产生运行在多个核上的机器指令。具体地，由于多个核需要读取同一输入张量数据的不同部分，同时也需要把各自的输出写回到同一输出张量数据的不同部分，计算库需要修改每个算子的计算指令中所有关于读取和存储部分的指令。

本公开实施例所提供的神经网络拆分方法能够尽量避免对单核处理器计算库进行修改，同时也能够实现神经网络模型在多核处理器上的并行执行。具体地，上层框架通过把神经网络模型中的算子拆分成若干个可以并行执行的子算子，对每个子算子，框架调用计算库生成所述子算子在单个核上执行的机器指令，通过把所述子算子的机器指令加载到不同核上，实现算子在多核处理器上的并行计算。具体地，

因为框架使用单核处理器计算库生成子算子的计算指令，神经网络模型中所述算子的输入和输出张量数据随着所述算子被拆分成子算子同样被拆分成相应的子张量数据。

基于上述描述，如图 2 所示，为本公开实施例提供一种神经网络模型拆分方法流程图。包括：

步骤 201)：根据所述神经网络模型中目标层的算子，确定与所述目标层的算子关联的张量数据的拆分状态集合。

在本实施例中，神经网络模型通常可以看作是一个由算子和多维张量数据所构成的有向无环图，算子和张量数据彼此之间通过有向边相互连接，有向边的指向表明了数据是算子的输入或者是输出。使用 *op* 表示算子，*tensor* 表示张量数据。同时，为了统一不同算子的拆分方式的表达，框架统一选择使用与算子相关联的张量数据的拆分方式来说明不同算子的拆分方式。不妨假设网络中所有张量数据都是 4 维的，对于图像分类网络最后的全连接层和归一化指数回归层的输入数据或输出数据来说，实际维度不到 4，仍然将其表示为 4 维张量。用符号 N, C, H, W 分别表示这 4 个维度。其中，N 表示批量的大小，C 表示特征图像的个数，H 表示特征图像的高，W 表示特征图像的宽。这种假设是仅仅是出于说明的便捷性，对于框架本身来说，可以支持含有任意维度数量的张量数据的神经网络模型的处理。尽管如此，4 维对于相当大一部分的神经网络结构都足够使用。

本技术方案对神经网络模型中的算子进行拆分时，算子的种类不同，该算子支持的计算逻辑也不相同，同样有不同的拆分策略。为了能够统一地表达不同算子的拆分策略，本技术方案采用算子的输入张量数据、输出张量数据的拆分状态来表示算子本身计算逻辑的拆分。

对于本技术方案来说，可以对整个神经网络模型中的所有算子进行拆分，也可以对神经网络模型中部分算子进行拆分。且，目前深度学习领域新出现的网络结构和算法已经逐渐模糊各个数据维度的物理意义和彼此之间的界限，本技术方案可以扩展应用到更多维度下的算子拆分。

把张量数据的任意一种拆分称为该张量数据的一种状态 s ，将张量数据拆分后，获得子张量数据集合。状态 s 通过对应的子张量数据集合表征。所有可能的拆分 $\{s_0, s_1, s_2, \dots\}$ 组成了该张量数据的拆分状态集合 S ，一般来说，这是一个非常大的状态空间，这意味着由张量数据的拆分状态所表示的算子的可能的拆分方式的空间也非常巨大。

通过一些合理的假设，可以对张量数据的状态集合进行剪枝。首先，多核加速器完成一个算子的计算的时延取决于执行子任务耗时最长的那个核的时间，而多核架构中各个核在硬件结构上彼此是相互对等的，因此每个核的时间耗费的长短取决于分配给该核的任务负载的多少。所以一个合理的假设是应该保证拆分后的子算子的规模大致均衡，为此，可以从张量数据的状态集合 S 中省去那些拆分上不均衡的拆分状态。此外，多核架构中核数通常是 2 的整数次幂，如 1, 2, 4, 8, 16 等等，一个并行度不是 2 的整数次幂的任务往往会导致核的调度上产生“碎片”，因此拆分后的子算子数量应当保证是 2 的整数次幂。通过这两个假设，可以极大地缩小算子拆分策略的搜索空间。

需要说明，并非选择任意与算子关联的张量数据的拆分状态都能表示该算子的一种有效的拆分方式。张量数据的拆分的维度应该被算子所支持，例如归一化指数回归算子（Softmax）的输入数据不应该在待归一化的维度上存在拆分。此外，算子的输入张量和输出张量的拆分应该满足算子的计算逻辑，例如，卷积算子的输出数据在 H/W 维度上拆分的每一个子块的起止点都应该确实是其对应的输入数据在 H/W 维度上拆分的子块根据卷积算子的卷积核和位移步长计算出来的；卷积算子的输入数据在 C 维度上的拆分应该和权值数据在 C 维度上的拆分完全一致，输出数据在 C 维度上的拆分应该和权值数据在 N 维度上的拆分完全一致。在框架中，使用输出状态根据每个算子的具体逻辑来反向推出算子的输入状态，或者使用输入状态根据每个算子的具体逻辑来正向推出算子的输出状态。这保证了相关数据的状态始终能够表示一个有效的算子拆分方式。

步骤 202): 根据所述神经网络模型的有向无环图遍历所述拆分

状态集合，确定相邻拆分状态集合之间的状态路径及状态路径的权重。

如图 3 所示，整个神经网络模型的拆分方案 P 可以看作是由每个算子的输入张量数据的拆分状态集合中的一种拆分状态向输出张量中的一种拆分状态的跳转。前一个算子的输出张量的拆分状态即是后一个算子输入张量的拆分状态。经过算子的每一种可能的跳转对应了在该算子上的一种有效的拆分方式。因此，状态路径表示算子的拆分方式。

在本技术方案中，按照一种分解方式对张量数据进行分解获取子张量集合，该子张量集合对应的一种拆分状态，不同的分解方式能够获得多种拆分状态，所有分解方式获得的拆分状态组成拆分状态集合。由此可知，每种拆分状态对应一子张量集合，子张量集合包含了张量数据中的全部元素。并且，在一个子张量集合中，每个子张量的元素可以重叠，也可以不重叠。

上文已描述，状态路径表示算子的拆分方式，经所述状态路径对应的拆分方式对算子的计算逻辑进行拆分，获得对应的子算子集合。输入张量数据的状态和对应输出张量数据的状态通过状态路径连接，表示该输入张量数据的一拆分状态的子张量数据集合经过子算子集合中的子算子处理，得到该输出张量数据的对应拆分状态的子张量数据集合。

在本技术方案中，状态路径的权重为算子在某种拆分状态下在多核加速器上并行执行所用的时间进行表征，且多核加速器完成一个算子的计算的时间取决于执行子任务耗时最长的那个核的时间。计算状态路径的权重时使用参数进行估计：

1) 拆分后的 n 个子算子的计算负载 $c_1, c_2, \dots, c_n$ 。其中， c_i 根据拆分后第 i 个子算子的类型和规模计算得到；

2) n 个子算子的访存数据量 $d_1, d_2, \dots, d_n$ 。其中， d_i 根据拆分后第 i 个子算子的类型和规模计算得到；

3) 每个加速器核的计算吞吐速率 α 。 α 由加速器本身的性能参数

所决定；

4)每个核的访存带宽 β 。通常来说，多个核共享有限的访存带宽，因此 $\beta = B/n$ 。其中， B 是多核加速器的总带宽。

状态路径的权重的计算公式为：

$$t = \max_{i=1,\dots,n} (\max(c_i/\alpha, d_i/\beta)) \quad \text{式 (1)}$$

其中，内侧的取最大值操作是基于算子实现的计算部分和访存部分之间能够相互隐藏，即计算和访存可以做到尽量并发执行。对于一些加速器来说，当子算子的规模过小时会导致每个核的计算吞吐量降低，可以对 α 进行进一步修正使估值更加准确。外侧的取最大值操作就是多核加速器完成一个算子的计算的时间取决于执行子任务耗时最长的那个核的时间。

需要说明的是，上述获取状态路径的权重的方式仅仅是例举的部分情况，而不是穷举，本领域技术人员在理解本申请技术方案的精髓的情况下，可能会在本申请技术方案的基础上产生其它的变形或者变换，比如：衡量状态路径的权重不仅仅可以是执行子任务的所花费的时间，也可以是执行子任务的吞吐量。或也可以通过实际测量在多核处理器上执行所述状态路径对应的算子拆分方式下的所有子任务的时间来确定状态路径的权重。但只要其实现的功能以及达到的技术效果与本申请类似，那么均应当属于本申请的保护范围。

步骤 203)：根据所述状态路径的权重，确定所述目标层的目标拆分路径。

在步骤 203 中，利用状态路径的权重，分两种方式来确定目标层的拆分路径。第一种方式为正向遍历来确定拆分路径，步骤包括：

遍历所述目标层的所有拆分状态集合，对当前拆分状态集合，遍历每一状态，获得所有指向当前状态的状态路径以及所述状态路径的起始状态到所述目标层的输入张量数据的起始状态的拆分路径；

根据所述状态路径和所述拆分路径确定所述当前状态到所述目标层的输入张量数据的起始状态的拆分路径；

根据所述状态路径的权重和所述拆分路径的权重确定所述当前状态到所述目标层的输入张量数据的起始状态的拆分路径；其中，所述拆分路径的权重根据所述拆分路径对应的所有状态路径的权重确定；

遍历完所述目标层的所有拆分状态集合后，获得所述目标层的输入张量数据的拆分状态集合与所述目标层的输出张量数据的拆分状态集合之间的目标拆分路径。

第二种方式为反向遍历来确定拆分路径，步骤包括：

遍历所述目标层的所有拆分状态集合，对当前拆分状态集合，遍历每一状态，获得所有以当前状态为起点的状态路径以及所述状态路径的结束状态到所述目标层的输出张量数据的终止状态的拆分路径；

根据所述状态路径的权重和所述拆分路径的权重确定所述当前状态到所述目标层的输入张量数据的起始状态的拆分路径；其中，所述拆分路径的权重根据所述拆分路径对应的所有状态路径的权重确定；

遍历完所述目标层的所有拆分状态集合后，获得所述目标层的输入张量数据的拆分状态集合与所述目标层的输出张量数据的拆分状态集合之间的目标拆分路径。

下面举例详细说明如何遍历完所述目标层的所有拆分状态集合后获得所述目标层的输入张量数据的拆分状态集合与所述目标层的输出张量数据的拆分状态集合之间的目标拆分路径。

图 3 所示的神经网络模型为串行的，且整个神经网络模型的输入张量数据和输出张量数据均是不拆分状态。将图 3 整个神经网络模型的算子进行拆分，把一个包含 n 个算子的串行神经网络模型描述成一个算子序列 $(op_1, op_2, \dots, op_n)$ ，假设每个算子只有一个输入和一个输出，前一个算子的输入是后一个算子的输出，那么包括整个神经网络模型的输入张量数据和输出张量数据以及所有的算子之间的中间结果张量在内，所有的张量数据构成集合 $(tensor_0, tensor_1, \dots, tensor_n)$ ， op_i 的输入是 $tensor_{i-1}$ ，输出是 $tensor_i$ 。对每个数据张量 $tensor_i$ ，有与之对应的状态集合

S^i , 搜索策略的目标是寻找一种张量本身和其状态集合的某个状态之间的映射关系 $tensor_i \rightarrow s^i$, 通过给神经网络模型中每个张量数据确定一个具体的拆分状态, 从而可以确定所有算子的拆分方式, 因此, 把一个神经网络模型中所有张量数据到其拆分状态的一种映射关系称为该网络模型的一种拆分方案 P 。在计算阶段, 第 i 个算子 op_i 以处于拆分状态 s 的输入数据计算出处于拆分状态 r 的输出张量数据, 具体的并行计算方式由输入张量数据和输出张量数据的状态所决定, 同时, 该算子的计算时间记为 $t_{s \rightarrow r}$, 其值的大小取决于相应的拆分方式和底层加速器的硬件特性, 则整个网络的时延 T 的计算公式为:

$$T = \sum_{i=1}^n t_{s^{i-1} \rightarrow s^i}, \text{ 其中 } s^{i-1} \in S^{i-1}, s^i \in S^i, \quad (2)$$

同样与之对应的还有使用该拆分方式在多核加速器上并行执行该算子所使用的时间 t_i , 因此, t_i 可以被看作是由算子的输入张量数据的状态指向输出张量数据的状态的有向边的权重。同时, 作为整个神经网络模型的输入张量数据和输出张量数据, 它们对应的拆分状态空间中只有一种不拆分的保持整个数据块连续完整的状态, 这使得神经网络模型的拆分方案 P 由完整的输入数据开始, 到完整的输出数据结束, 外部的用户始终看到一个完整的输入和输出。此时, 对于给定的神经网络模型搜索一个好的拆分方案 P , 即是寻找一条由输入张量数据的不拆分状态到输出张量数据的不拆分状态的最短路径, 该路径在每个中间结果张量的有效状态空间中都要选择一个状态经过。式 3、式 4 给出了这种抽象的公式表示。

$$P = \{s^0, s^1, \dots, s^n\} = \arg \min_{s^i \in S^i} (T(s^0, s^1, \dots, s^n)) \quad (3)$$

$$T(s^0, s^1, \dots, s^n) = \sum_{i=1}^n t_{s^{i-1} \rightarrow s^i} \quad (4)$$

同样注意到图 3 中, 存在输入张量数据的一个拆分状态指向输出

张量数据的多个拆分状态的情况，这种情况是存在的，进一步说，正是这种情况才导致了神经网络模型拆分空间的巨大。

在本技术方案中，设整个神经网络模型的输入张量数据的不拆分状态为起始状态 s_{root} ，在初始阶段，神经网络模型的输入张量数据的不拆分状态为起始状态 s_{root} 对应的拆分路径的权重为 0，其余所有张量数据的所有状态的对应的拆分路径的权重为 ∞ 。神经网络模型中任一张量数据的任一状态 s 都有与之对应的由 s_{root} 开始到 s 的拆分路径权重为 l_s 。由前往后访问每一个拆分状态集合，在每个拆分状态集合中，依次遍历其中的每一个状态 s 。对每个状态 s ，指向后一拆分状态集合中若干拆分状态的每一个有向边 $e_1, \dots, e_{ks}$ 。以后一拆分状态集合中的拆分状态 v 为例，使用式（1）获得状态 s 到状态 v 之间的权重 t_{sv} ，利用式（5）来更新该状态路径指向的下一拆分状态集合中的状态 v 对应的由 s_{root} 开始到状态 v 的拆分路径的权重 l_v 。

$$l_v = \min(l_v, l_s + t_{sv}) \quad (5)$$

在依据神经网络模型的拓扑关系正向遍历完成所有拆分状态集合的访问后，获得整个神经网络模型的输入张量数据的不拆分状态 s_{root} 到神经网络模型的输出张量数据的不拆分状态 s_{end} 的目标拆分路径。

上述描述一条由不拆分状态 s_{root} 到不拆分状态 s_{end} 经过每个拆分状态集合中的一个状态构成的路径，即为神经网络模型的拆分路径。从神经网络模型的拆分路径中选取权重最小的作为神经网络模型的目标拆分路径。

需要说明的是，图 3 所示的神经网络模型为串行神经网络模型，且为了便于说明本技术方案，神经网络模型的输入张量数据和输出张量数据对应的拆分状态集合均为不拆分状态。在神经网络模型的输出张量数据的拆分状态集合不是不拆分状态 s_{end} ，而是多个拆分状态构成集合时，神经网络模型的输出张量数据的拆分状态集合中的每个拆分状态的拆分路径的权重中选出最小值作为整个神经网络模型的输入张量数据的拆分状态集合到神经网络模型的输出张量数据的拆分

状态集合之间的目标拆分路径。

注意到整个方案也可以转换为搜索由不拆分状态 s_{end} 到不拆分状态 s_{root} 的拆分路径，二者是等价的。同理，在神经网络模型的输入张量数据的拆分状态集合不是不拆分状态 s_{end} ，而是多个拆分状态构成集合时，从神经网络模型的输入张量数据的拆分状态集合中的每个拆分状态的拆分路径的权重中选出最小值作为整个神经网络模型的输入张量数据的拆分状态集合到神经网络模型的输出张量数据的拆分状态集合之间的目标拆分路径。

图 3 所示的神经网络模型为串行的神经网络模型，且模型的输入张量数据和输出张量数据的拆分状态集合中的状态为不拆分情况。在实际应用中，针对图 2、图 4、图 7 和图 9 所示的技术方案，均需要解决多分支神经网络模型中不同分支拆分方式一致性的问题。位于分支交汇处的算子具有 1 个以上的输入张量数据，例如对位加法算子 (Add)，对位乘法算子 (Mult)，拼接算子 (Concat)。对一个有 2 个输入的算子 A ，在访问该算子，即根据输入张量数据的拆分状态集合枚举输入张量数据的拆分状态集合结束后，两个输入张量数据 $tensor_{left}$ 、 $tensor_{right}$ 分别有对应的拆分状态集合 S_{left} 和 S_{right} 。分别沿 $tensor_{left}$ 、 $tensor_{right}$ 开始的两条之路继续向前遍历，一种情况下，两条支路会直接延伸直至遍历结束，代表整个网络有不只一个输入数据，这通常在推理任务中并不常见，另一种情况下，两条支路在某算子处合到一起。无论哪种情况，当确定拆分方案 P 时，在算子 A 的两个输入张量数据 $tensor_{left}$ 、 $tensor_{right}$ 上，可能会选中相互不匹配的拆分状态。具体来说，假设算子 A 是二元对位加法算子，回溯过程在 $tensor_{left}$ 的拆分状态集合中选中的可能是一个仅在 C 维度上有拆分的状态，而在 $tensor_{right}$ 的拆分状态集合中选中的可能是一个仅在 H 维度上有拆分的状态，这两个拆分状态所表示的加法算子本身的拆分方式是不一致的，因此会导致整个拆分方案 P 无效。为了解决这个问题，在遍历算子 A 结束前保证 $tensor_{left}$ 、 $tensor_{right}$ 对应的拆分状态集合中都只含有一个拆分状态，这确保回溯过程中在两状态集合中选择的状态

态的确定性。因此，在正向遍历阶段，当所述算子的输出张量数据被至少两个算子作为输入张量数据，或者所述算子具有至少两个输出张量数据时，所述算子的输出张量数据的拆分状态集合中保留一个拆分状态，且所述拆分状态经由所述算子的同一状态路径确定。在反向遍历阶段，当所述算子具有至少两个输入张量数据时，所述算子的输入张量数据的拆分状态集合中保留一个拆分状态，且所述拆分状态经由所述算子的同一状态路径确定。这样，在遍历分支算子结束前，将从多个输入数据的拆分状态集合中选择出对应累计权重最小的状态保留下来，移除拆分状态集合中其他的拆分状态。

需要说明的是，上述获取目标拆分路径方式类似于 viterbi 算法，此次仅仅是列举的部分情况，而不是穷举，本领域技术人员在理解本申请技术方案的精髓的情况下，可能会在本申请技术方案的基础上产生其它的变形或者变换，比如：神经网络模型的输入张量数据的拆分状态集合到神经网络模型的输出张量数据的拆分状态集合之间的每一条拆分路径的权重由对应的状态路径的权重之和确定。根据经验设置一阈值，拆分路径的权重小于设定的阈值，就可以作为目标拆分路径对神经网络模型进行拆分。但只要其实现的功能以及达到的技术效果与本申请类似，那么均应当属于本申请的保护范围。

步骤 204): 利用所述目标拆分路径对所述神经网络模型的目标层的算子进行拆分。

由上述描述可知，通过把神经网络中的算子的计算逻辑拆分成更小的子任务分配到多个核上并行执行来充分利用多核处理器结构芯片的硬件资源。

对于图 2 所示的技术方案来说，最理想的条件下，希望拆分后的子算子可以把它们的输出张量数据写到一个存放完整输出张量数据的存储空间中的对应位置上，这样一个算子的所有子算子执行完毕后得到的始终是一个完整连续的数据块。但对于部分加速器来说，这一点并不容易做到。首先，由于拆分后的算子的输出张量数据在整个输出中的存储位置可能是不连续的，使得需要重写算子输出部分的代

码,使其能够把输出结果写回到一个子张量在存储中对应的离散位置上。与此同时,加速器通常会进一步调整数据在存储中的顺序来提升计算时的访存效率,这使得修改算子输出逻辑的工作更加困难繁琐。同时,如果后续算子的计算逻辑或拆分逻辑不需要输入数据在某个维度上保证存储连续性,可以把上一层输出的在该维度上处于离散存储状态的数据直接用于下一层的计算,而不需要保证输出数据的连续性。

因此,框架把调整张量数据拆分形式的任务从算子的计算任务中剥离出来,抽象为一个新的算子,称为胶水算子,这种剥离避免了对每个算子输出逻辑的修改,增强了框架对于底层不同加速器的可移植性。胶水算子用来把张量按照某种方式拆分的子数据块调整成按照另一种拆分方式形成的子数据块。如表 1 所示,不同种类的算子所允许的拆分方式表现到它们的输入张量数据、输出张量数据上是不同的。当上一层算子的输出张量数据的拆分方式并不被下一层算子所允许,就需要使用胶水算子对该张量数据的拆分方式进行调整,从而把前后算子“粘接”起来。此外,即使上一层输出的拆分方式被下一层所支持,也可以通过胶水算子把张量数据的拆分进行调整成更利于与下一层计算的形式。

表 1

<i>Operation</i>	允许拆分的输入维度
Convolution	N, C, H, W (H 和 W 不应小于 kernel)
FC	N, C
Relu	N, C, H, W
Scale	N, C, H, W
BatchNorm	N, C, H, W
Softmax	无法拆分要规范化的维度
Pooling	N,C,H,W(H 和 W 不应小于 kernel)

基于上述描述,在图 2 的基础上,本实施例还提出另一种神经网络

络模型的拆分方法。如图 4 所示，在图 2 的基础上，还包括：

步骤 201')：在所述目标层的算子与关联的拆分状态集合之间插入胶水算子，调整所述算子的张量数据的拆分状态集合中的状态；其中，所述胶水算子用于将所述张量数据按照一拆分方式获得的状态调整成按照任一种拆分方式获得的状态。

在本步骤中，通过胶水算子来表示调整张量数据的拆分状态的行为方式，神经网络模型的每一层的计算规模随着网络的延伸不断变化，随着神经网络模型拆分趋势的变化，需要对算子的拆分方式做出相应的调整，也就是对中间结果的状态进行调整。如图 5 所示，图 3 中的 Op_2 和其输入 Tensor1 之间加入了胶水算子，可以把张量数据的任意一种拆分状态转换成另一种拆分状态。对胶水算子而言，其输入张量数据和输出张量数据有着相同的形状和相同的状态空间，由输入张量数据的任一拆分状态，存在指向输出张量数据所有拆分状态的有向边，因此在输入张量数据的拆分状态集合和输出张量数据的拆分状态集合之间形成了全连接的网状结构。这使得任意一种输入张量数据的拆分状态可以在算子 Op_2 前转换成另一种拆分状态，给拆分方案 P 的搜索空间中引入了在每个算子计算开始前调整其输入张量数据的拆分状态，也即是调整算子本身拆分方式的可能性。

需要说明的是，图 5 示出了算子与对应的输入张量数据之间插入胶水算子，也可以在算子与对应的输出张量数据之间插入胶水算子，更可以在算子与对应的输入张量数据、输出张量数据之间均插入胶水算子，此次仅仅是例举的部分情况，而不是穷举，本领域技术人员在理解本申请技术方案的精髓的情况下，可能会在本申请技术方案的基础上产生其它的变形或者变换，但只要其实现的功能以及达到的技术效果与本申请类似，那么均应当属于本申请的保护范围。

在神经网络模型的目标层的算子与关联的拆分状态集合之间插入胶水算子，虽然对算子的拆分方式作出相应的调整，但是这种调整又会带来额外的开销，如何在整个神经网络模型中恰当地插入胶水算子来提高神经网络模型的性能。为了解决这个问题，在神经网络模型

的目标层的算子与关联的拆分状态集合之间插入胶水算子, 获取包含所述胶水算子在内的神经网络模型的有向无环图; 根据所述有向无环图遍历所述目标层的所有张量数据对应的拆分状态集合, 确定相邻拆分状态集合之间的状态路径及状态路径的权重; 根据所述状态路径的权重, 确定包含所述胶水算子在内的神经网络模型的目标层的拆分路径; 利用包含所述胶水算子在内的神经网络模型的目标层的拆分路径对插入的每个胶水算子进行选择, 对不需要插入的胶水算子删除, 对需要插入的胶水算子保留。

对于胶水算子来说, 胶水算子内部采用拆分-拼接、拼接-拆分、拼接、拆分这四种方式中的其中一种实现方式, 在拼接阶段可以把在任意维度上相邻的子数据块凭借成一个新的数据块, 在拆分阶段, 可以把任意一个子数据块拆分成两个更小的子数据块。任意一种拆分可以通过这样一个两阶段的过程转换成另外一种拆分形式。为了说明这一点, 不妨假设数据本身是一维的, 调整前的拆分形式表示为 $\{(0, p_1), (p_1, p_2), \dots, (p_{n-1}, end)\}$, 每一段代表一维数据拆分后的一个子段, 调整后的拆分形式是 $\{(0, q_1), (q_1, q_2), \dots, (q_{m-1}, end)\}$, 如果调整前的某相邻两段 $(p_{i-1}, p_i), (p_i, p_{i+1})$ 是调整后的某一段 (q_j, q_{j+1}) , 即 $p_{i-1} = q_j, p_{i+1} = q_{j+1}$, 在调整这一部分时只需要在拼接阶段把 $(p_{i-1}, p_i), (p_i, p_{i+1})$ 拼接在一起, 跳过拆分阶段。同样另一种情况下, 如果调整前的某一子段是调整后的若干子段的集合, 则跳过拼接阶段, 在拆分阶段执行相应的拆分。最坏情况下, 可以把拼接阶段把所有数据组合成一个完整一维数据, 在拼接阶段在进行对应的拆分。

以胶水算子采用拆分-拼接或拼接-拆分为例, 假设待调整张量数据的总大小为 M , 且两个阶段均不能跳过, 且每个阶段都要针对 4 个维度进行拼接或者拆分。为了便于移植, 拼接和拆分通常会使用神经网络算法中自带的拼接算子 (Concat) 和拆分算子 (Slice) 实现, 由于这两个算子每次只能处理一个维度, 整个胶水在最差情况下会带来 $8M$ 的存储读写开销。所以, 必须在调整拆分状态和引入的额外开销之间寻找一个最佳的平衡点, 再在引入尽量少的胶水算子的情况下

又能符合网络结构的规律在合理的地方对算子的拆分方式进行调整。

进一步详说，胶水算子和普通的神经网络算子做相同处理，每个胶水算子对张量数据拆分状态的调整都有与之对应的时间 t ，该时间作为相应的状态路径的权重。仍然利用公式（5）获得包括胶水算子在内的神经网络模型的目标层的输入张量数据的不拆分状态 s_{root} 到神经网络模型的输出张量数据的不拆状态 s_{end} 的目标拆分路径。选择胶水算子时，在拆分路径中，检查每个胶水算子的输入张量数据对应的一拆分状态及输出张量数据对应的一拆分状态，如果该两个拆分状态相同，即图 5 中的拆分状态集合 Tensor_1 中的拆分状态 status_1 通过状态路径与拆分状态集合 Tensor_1' 中的拆分状态 status_1 相连，这两个拆分状态相同。说明神经网络模型的目标层的拆分路径 P 并不需要调整算子 Op_2 的输入张量数据的拆分状态，并且这是一个基于前后算子和整体性能统筹考虑的结果。算子 Op_2 与对应输入张量数据之间插入的胶水算子会从网络中移除。否则，需要对插入的胶水算子保留。

需要说明的是，胶水算子的实现使用神经网络模型中原有的算子。拼接阶段对应的是神经网络模型中的 Concat 算子，而拆分阶段对应的是神经网络模型中的 Slice 算子，任何已经支持这两种算子的加速器都可以很快地实现胶水算子。并且，在本实施例中，上述获取目标拆分路径方式类似于 viterbi 算法，此次仅仅是例举的部分情况，而不是穷举，本领域技术人员在理解本申请技术方案的精髓的情况下，可能会在本申请技术方案的基础上产生其它的变形或者变换，比如：神经网络模型的输入张量数据的拆分状态集合到神经网络模型的输出张量数据的拆分状态集合之间的每一条拆分路径的权重由对应的状态路径的权重之和确定。根据经验设置一阈值，拆分路径的权重小于设定的阈值，就可以作为目标拆分路径对神经网络模型进行拆分。但只要其实现的功能以及达到的技术效果与本申请类似，那么均应当属于本申请的保护范围。

需要强调的是，图 2 所示的关于算子拆分的技术方案内容适用于

图 4 所示的技术方案，这里不再赘述。

对于神经网络模型来说，卷积算子是比较特殊的算子，在一些情况下需要额外的辅助算子完成拆分任务。当把计算按照输入张量数据的 H/W 维度进行划分时，如果卷积核窗口的尺寸超其每次移动的步长，即 $\text{kernel} > \text{stride}$ ，拆分后的卷积算子在计算时会出现框移动到边界处超过子张量数据的边界的情况，而缺少的这一部分数据位于相邻子张量数据上。为了处理这种子任务之间的输入张量数据相互重叠的情况，同时保证可移植性，同样把这种需要访问相邻子张量数据的边界数据的行为剥离出来，形成一个新的辅助算子，称为补偿算子。

如图 6 所示，补偿算子用来把一个子张量数据之外的相邻子张量数据中获取目标数据，将目标数据和子张量数据合并在一起形成一块更大的数据，此时计算阶段窗口的移动范围不会超过这个经过补偿的数据块的边界。除了卷积算子之外，池化算子以及目前不是很常见的局部响应归一化算子（Local Response Normalization，简称 LRN），同样有这种拆分后的子任务依赖相邻数据块上数据的问题，池化算子与卷积算子类似，主要是由于池化窗口大于窗口本身的移动步长所引起的，而局部响应归一化算子不同，其计算逻辑是，即为了计算输出张量数据在 C 维度上的一个点的结果，需要输入张量数据在 C 维度上与之对应的一个点以及左右相邻的 $k/2$ 个点的数值。因此，如果把局部响应归一化算子的计算按照 C 维度拆分成多个 LRN 算子，每个新算子同样需要相邻子张量数据中的元素数据来计算位于 C 维度边界上的值。

根据算子计算输出张量数据在某个维度上的一个数据所需要的在输入张量数据在该维度上的数据的范围，可以把算子归纳为三类。一类是点对点的算子，即计算输出张量数据的一个数据点只需要输入张量上与之对应的一个数据点的值，这一类算子包括激活算子（Relu, pRelu），批标准化算子（BatchNorm），以及对位加减乘除的基本算子（Add, Sub, Mult, Div）。这类算子在任何维度上可以进行任务拆分，得到的子算子在计算阶段只需要对应的子张量数据作为输入。另

一类算子是全依赖类型的算子，即计算输出张量数据的一个数据点只需要输入张量数据上在该维度上的所有数据，例如卷积算子和全连接算子在计算输出张量数据 C 维度上的一个点需要输入张量数据 C 维度上的所有点，尽管可以通过事后累加部分和的方式实现卷积算子在输入 C 维度上的拆分，当算子在 C 维度上的计算逻辑更加复杂时，例如归一化指数回归算子（Softmax），式（6）给出在其归一化维度上的计算公式。

$$O_i = \frac{e^{I_i}}{\sum_j e^{I_j}} \quad (6)$$

其中， I 是输入张量数据在归一化维度上的向量， O 是输出张量数据在归一化维度上的向量。不同于卷积的部分和累加，这里的计算逻辑较为复杂，很难实现拆分。从这个角度看来，补偿算子实际是用来处理位于点对点类型的算子和全依赖类型的算子之间的第三种情况，即计算输出张量数据上的一个点需要输入张量数据在对应位置附近区域内的数据。对应位置附近区域根据补偿参数确定。这种情况下，算子在计算逻辑上实际上仍然是可以拆分的，尽管它们会依赖与子张量数据之外的数据，使用补偿算子可以统一地解决这个问题。

基于此，如图 7 所示，为本公开实施例提供的一种神经网络模型拆分方法流程图之三。在图 2 的基础上，还包括：

步骤 201'')：在所述目标层的算子与关联的拆分状态集合之间插入补偿算子，调整所述算子的输入张量数据的拆分状态集合中的状态；其中，所述补偿算子用于从所述状态的任一子张量数据的相邻子张量数据中获取目标数据，将所述目标数据与所述子张量数据合并。

在本技术方案中，为了解决卷积算子、池化算子由于窗口小于位移步长导致在沿 H/W 维度进行任务拆分的情况下，窗口超过输入子张量数据的边界的问题，框架引入了补偿算子在计算开始前，对一子张量数据集合来说，在每个子张量数据周围补上位于相邻的子张量数

据的元素。这种方式避免了修改拆分后的卷积算子或者池化算子本身的计算逻辑,使得对相邻子张量数据的依赖行为对卷积算子、池化算子本身不可见,有利于这个系统的快速实现,便于在不同结构的加速器上一致。但是,补偿算子本身会带来额外的开销,假设原始数据块的大小为 M ,在不考虑补偿后子张量数据彼此之间重叠部分的情况下,补偿算子引入了 $2M$ 的访存开销。卷积算子和池化算子是组成神经网络尤其是图像分类神经网络的主要算子,为了减轻补偿行为带来的开销,将插入的补偿算子采用一种金字塔形式的结构来合并。如图 8 所示,该神经网络模型是由两个卷积算子构成的串行序列,两个卷积算子均按照 H/W 维度进行任务拆分,分别拆成了 4 个更小的卷积算子,图中省略了数据的 N 维度和 C 维度。假设两个卷积算子的卷积核尺寸分别是 k_1 、 k_2 ,为了简化计算,位移步长均为 1。正常情况下,卷积算子 Conv1 在计算前在子张量数据外围补偿的数据宽度为 $k_1/2$,这保证拆分后的卷积任务在计算时卷积核不会超过输入子张量数据的边界。但在这里,卷积算子 Conv1 在计算前在子张量数据外围补偿的数据宽度为 $k_1/2 + k_2/2$,这使得其输出数据 Tensor1 的子张量数据彼此之间有 k_2 宽度的相互重叠,所以卷积算子 Conv2 在计算开始前不需要对其输入子张量数据进行数据补偿。

通过这种方法,可以把串行算子序列中使用的多个补偿算子合并成顶端的一个,尽管这使得第一次补偿的访存开销变大,但在补偿宽度远小于子数据块自身的尺寸的情况下,可以有效减少了模型拆分后补偿算子的访存开销。但另一方面,这种方法会带来重复计算,图 8 中卷积算子 Conv1 的输出张量数据 Tensor1 的子张量数据的重叠部分的结果在多个拆分后的卷积算子中都被计算了。此外,对输入的特征图像尺寸较小的卷积网络,由于不再满足补偿宽度远小于子张量数据自身尺寸的条件,需要更加谨慎地评估合并多个补偿算子前后总的访存开销的变化情况。为了解决这个问题,同样把补偿算子的合并加入的拆分方案的搜索空间中。整个遍历过程由正向遍历转变为反向遍历,这两种输入在原理上是相通的,但后者更加适合引入了补偿算子

合并以后的搜索策略。设整个神经网络模型的输出张量数据的不拆分状态为终点状态 s_{end} ，神经网络模型中任一张量数据的任一状态 s 都有与之对应的由 s 开始到 s_{end} 的拆分路径的权重为 l_s 。在遍历开始前， s_{end} 对应的权重为 0，其余所有张量数据的所有状态的对应权重为 ∞ 。依据神经网络模型的拓扑关系反向遍历每个算子，在由输出张量数据的拆分状态枚举可能的输入张量数据的拆分状态的过程中，除了枚举正常情况下的张量数据彼此没有重叠需要引入补偿过程的拆分状态，同样枚举彼此有重叠不需要补偿的输入拆分状态。后者对应的状态路径的权重在计算上考虑了冗余计算的部分所增加的时间。仍然利用公式（5）获得包括补偿算子在内的神经网络模型的目标层的输入张量数据的不拆分状态 s_{root} 到神经网络模型的输出张量数据的不拆分状态 s_{end} 的目标拆分路径。

对于补偿算子来说，将神经网络模型中插入的多个补偿算子采用金字塔结构进行合并，可以获得一个合并后的补偿算子，也可以获得多个合并后的补偿算子。这种情况下，合并后的补偿算子的个数少于合并之前的补偿算子的个数。

需要说明的是，上述获取目标拆分路径方式类似于 viterbi 算法，此次仅仅是例举的部分情况，而不是穷举，本领域技术人员在理解本申请技术方案的精髓的情况下，可能会在本申请技术方案的基础上产生其它的变形或者变换，比如：神经网络模型的输入张量数据的拆分状态集合到神经网络模型的输出张量数据的拆分状态集合之间的每一条拆分路径的权重由对应的状态路径的权重之和确定。根据经验设置一阈值，拆分路径的权重小于设定的阈值，就可以作为目标拆分路径对神经网络模型进行拆分。但只要其实现的功能以及达到的技术效果与本申请类似，那么均应当属于本申请的保护范围。

需要强调的是，图 2 所示的关于算子拆分的技术方案内容适用于图 7 所示的技术方案，这里不再赘述。

如图 9 所示，为本公开实施例提供的一种神经网络模型拆分方法流程图之四。将胶水算子和补偿算子均引入算子拆分方案中，此时拆

分方法包括：

步骤 a)：根据所述神经网络模型中目标层的算子，确定与所述目标层的算子关联的张量数据的拆分状态集合；其中，所述目标层为所述神经网络模型中的至少一层；

步骤 b)：在所述目标层的算子与关联的拆分状态集合之间插入胶水算子，调整所述算子的张量数据的拆分状态集合中的状态；其中，所述胶水算子用于将所述张量数据的拆分状态集合中的状态调整成所述张量数据的任一拆分状态；

步骤 c)：在所述目标层的算子与关联的拆分状态集合之间插入补偿算子，调整所述算子的输入张量数据的拆分状态集合中的状态；其中，所述补偿算子用于从所述状态的任一子张量数据的相邻子张量数据中获取目标数据，将所述目标数据与所述子张量数据合并；

步骤 d)：根据所述神经网络模型的有向无环图遍历所述拆分状态集合，确定相邻拆分状态集合之间的状态路径及状态路径的权重；其中，所述状态路径表示所述算子的拆分方式；所述拆分状态集合中的每个状态表示一个子张量数据集合，所述状态的所有子张量数据的并集结果为所述张量数据；

步骤 e)：根据所述状态路径的权重，确定所述目标层的目标拆分路径；

步骤 f)：利用所述目标拆分路径对所述神经网络模型的目标层的算子进行拆分。

在神经网络模型的每个算子和其输入张量数据之间插入胶水算子，在神经网络模型的输出张量数据和产生该输出张量数据的算子之间拆入胶水算子。为神经网络模型中的每个张量数据 $tensor_i$ 初始化其状态集合 S_i ，状态集合中存储状态本身以及由该数据的该拆分状态开始执行到网络最后的输出数据的输出状态 s_{root} 的最短时间，该数值对用 (s, t) 表示。整个神经网络模型的输出张量数据对应的状态集合 S_{root} 中有该数据的不拆分状态以及对应的最短时间 $(s_{root}, 0)$ ，其余所有集合为空。对于给定神经网络模型，对神经网络模型中的所有算子按照彼

此之间的依赖关系给出拓扑序 λ ，该拓扑序需满足：对任意算子 A ，所有依赖于 A 的算子必须在拓扑序上排在 A 之后，而所有 A 依赖的算子必须在拓扑序上排在 A 之前。

考虑到补偿算子的插入，反向遍历神经网络模型的每个算子的拆分状态集合。在反向遍历阶段，按照逆 λ 的顺序逐个遍历神经网络模型中的算子，对 m 输入 n 输出的算子 A ，有输入张量数据 $u_1, \dots, u_m$ ，输出张量数据 $v_1, \dots, v_n$ ，图2所示的在神经网络模型中关于算子拆分的技术方案内容适用于图9所示的技术方案，图4所示的在插入胶水算子的神经网络模型中关于胶水算子的技术方案内容适用于图9所示的技术方案，图7所示的在插入补偿算子的神经网络模型中关于补偿算子的技术方案内容适用于图9所示的技术方案，这里均不再赘述。反向遍历的时间复杂度为 $O(NM^2)$ ，其中， N 是神经网络模型中算子的个数， M 表示所有张量数据的拆分状态集合中最大拆分状态集合中的状态个数。

需要强调的是，图2所示的关于算子拆分技术方案内容适用于图9所示的技术方案，图4所示的基于胶水算子的算子拆分技术方案中关于胶水算子的内容适用于图9所示的技术方案，图7所示的基于补偿算子的算子拆分技术方案中关于补偿算子的内容适用于图9所示的技术方案，这里均不再赘述。

图2、图4、图7和图9所示的技术方案通过把神经网络模型中的目标层的每个算子拆分成更小的子任务，并分配到多个核上并行执行来充分利用多核系统的硬件资源。在图4、图7和图9所示的技术方案中，通过引入胶水算子或补偿算子，保证了神经网络模型在拆分后的计算图仍然可以使用单核上的算子核函数来实现，避免了框架在移植过程底层加速器需要修改或者重新实现大量算子的软件栈工作，使其对不具有良好可编程性的加速器更加友好。框架能够自动生成一套对于给定神经网络和多核加速器的高效拆分方案，在方案的生成过程中，能够根据算子的类型和规模结合底层硬件的计算吞吐速率和访存带宽合理调整算子的拆分方式，在硬件核的计算效率和算子本身的

拆分度之间取得较好的平衡，同时也会兼顾上下文算子之间在拆分上的相互配合，统筹规划多个算子的拆分选择。

本公开提供一种神经网络模型拆分装置，其中，包括：

拆分状态集合模块，用于根据所述神经网络模型中目标层的算子，确定与所述目标层的算子关联的张量数据的拆分状态集合；其中，所述目标层为所述神经网络模型中的至少一层；

状态路径模块，用于根据所述神经网络模型的有向无环图遍历所述拆分状态集合，确定相邻拆分状态集合之间的状态路径及状态路径的权重；其中，所述状态路径表示所述算子的拆分方式；所述拆分状态集合中的每个状态表示一个子张量数据集合，所述状态的所有子张量数据的并集结果为所述张量数据；

目标拆分路径模块，用于根据所述状态路径的权重，确定所述目标层的目标拆分路径；

拆分模块，用于利用所述目标拆分路径对所述神经网络模型的目标层的算子进行拆分。

优选地，所述目标拆分路径模块包括：

第一遍历单元，用于遍历所述目标层的所有拆分状态集合，对当前拆分状态集合，遍历每一状态，获得所有指向当前状态的状态路径以及所述状态路径的起始状态到所述目标层的输入张量数据的起始状态的拆分路径；

第一拆分路径确定单元，用于根据所述状态路径的权重和所述拆分路径的权重确定所述当前状态到所述目标层的输入张量数据的起始状态的拆分路径；其中，所述拆分路径的权重根据所述拆分路径对应的所有状态路径的权重确定；

第一选择目标拆分路径单元，用于遍历完所述目标层的所有拆分状态集合后，获得所述目标层的输入张量数据的拆分状态集合与所述目标层的输出张量数据的拆分状态集合之间的目标拆分路径。

优选地，所述目标拆分路径模块包括：

第二遍历单元，用于遍历所述目标层的所有拆分状态集合，对当

前拆分状态集合，遍历每一状态，获得所有以当前状态为起点的状态路径以及所述状态路径的结束状态到所述目标层的输出张量数据的终止状态的拆分路径；

第二拆分路径确定单元，用于根据所述状态路径的权重和所述拆分路径的权重确定所述当前状态到所述目标层的输出张量数据的终止状态的拆分路径；其中，所述拆分路径的权重根据所述拆分路径对应的所有状态路径的权重确定；

第二选择目标拆分路径单元，用于遍历完所述目标层的所有拆分状态集合后，获得所述目标层的输入张量数据的拆分状态集合与所述目标层的输出张量数据的拆分状态集合之间的目标拆分路径。

优选地，还包括：

第一拆分状态集合优化模块，用于在正向遍历阶段，当所述算子的输出张量数据被至少两个算子作为输入张量数据，或者所述算子具有至少两个输出张量数据时，所述算子的输出张量数据的拆分状态集合中保留一个拆分状态，且所述拆分状态经由所述算子的同一状态路径确定。

优选地，还包括：

第二拆分状态集合优化模块，用于在反向遍历阶段，当所述算子具有至少两个输入张量数据时，所述算子的输入张量数据的拆分状态集合中保留一个拆分状态，且所述拆分状态经由所述算子的同一状态路径确定。

如图 10 所示，为本申请实施例提出的一种神经网络模型拆分硬件设备示意图。包括存储器、处理器以及存储在所述存储器上并可在所述处理器上运行的计算机程序，所述处理器执行所述计算机程序时实现上述所述的神经网络模型拆分方法。

本说明书实施方式提供的一种神经网络模型拆分硬件设备，其存储器和处理器实现的具体功能，可以与本说明书中的前述实施方式相对照解释，并能够达到前述实施方式的技术效果，这里便不再赘述。

针对上述技术问题，还提出了一种神经网络模型的拆分方法及相

关产品，除了神经网络模型拆分装置的相关描述，相关的拆分方法及相关产品的描述和上述各实施例描述的一致，此处不再加以赘述。关于神经网络模型拆分装置的相关描述如下。

本公开提供一种神经网络模型拆分装置，所述装置包括：

拆分状态集合确定模块，用于根据所述神经网络模型中目标层的算子，确定与所述目标层的算子关联的张量数据的拆分状态集合；其中，所述目标层为所述神经网络模型中的至少一层；

插入胶水算子模块，用于在所述目标层的算子与关联的拆分状态集合之间插入胶水算子，调整所述算子的张量数据的拆分状态集合中的状态；其中，所述胶水算子用于将所述张量数据按照一拆分方式获得的状态调整成按照任一种拆分方式获得的状态；

状态路径确定模块，用于根据所述神经网络模型的有向无环图遍历所述拆分状态集合，确定相邻拆分状态集合之间的状态路径及状态路径的权重；其中，所述状态路径表示所述算子的拆分方式；所述拆分状态集合中的每个状态表示一个子张量数据集合，所述状态的所有子张量数据的并集结果为所述张量数据；

目标拆分路径确定模块，用于根据所述状态路径的权重，确定所述目标层的目标拆分路径；

拆分模块，用于利用所述目标拆分路径对所述神经网络模型的目标层的算子进行拆分。

优选地，所述目标拆分路径确定模块包括：

第一遍历单元，用于遍历所述目标层的所有拆分状态集合，对当前拆分状态集合，遍历每一状态，获得所有指向当前状态的状态路径以及所述状态路径的起始状态到所述目标层的输入张量数据的起始状态的拆分路径；

第一拆分路径确定单元，用于根据所述状态路径的权重和所述拆分路径的权重确定所述当前状态到所述目标层的输入张量数据的起始状态的拆分路径；其中，所述拆分路径的权重根据所述拆分路径对应的所有状态路径的权重确定；

第一选择目标拆分路径单元,用于遍历完所述目标层的所有拆分状态集合后,获得所述目标层的输入张量数据的拆分状态集合与所述目标层的输出张量数据的拆分状态集合之间的目标拆分路径。

优选地,所述目标拆分路径确定模块包括:

第二遍历单元,用于遍历所述目标层的所有拆分状态集合,对当前拆分状态集合,遍历每一状态,获得所有以当前状态为起点的状态路径以及所述状态路径的结束状态到所述目标层的输出张量数据的终止状态的拆分路径;

第二拆分路径确定单元,用于根据所述状态路径的权重和所述拆分路径的权重确定所述当前状态到所述目标层的输出张量数据的终止状态的拆分路径;其中,所述拆分路径的权重根据所述拆分路径对应的所有状态路径的权重确定;

第二选择目标拆分路径单元,用于遍历完所述目标层的所有拆分状态集合后,获得所述目标层的输入张量数据的拆分状态集合与所述目标层的输出张量数据的拆分状态集合之间的目标拆分路径。

优选地,所述插入胶水算子模块包括:

插入单元,用于在所述目标层的算子与关联的拆分状态集合之间插入胶水算子,获取包含所述胶水算子在内的神经网络模型的有向无环图;

状态路径单元,用于根据所述有向无环图遍历所述目标层的所有张量数据对应的拆分状态集合,确定相邻拆分状态集合之间的状态路径及状态路径的权重;

确定目标拆分路径单元,用于根据所述状态路径的权重,确定包含所述胶水算子在内的神经网络模型的目标层的目标拆分路径;

选择单元,用于利用包含所述胶水算子在内的神经网络模型的目标层的目标拆分路径对插入的每个胶水算子进行选择,对不需要插入的胶水算子删除,对需要插入的胶水算子保留。

优选地,所述插入胶水算子模块插入的胶水算子用于将胶水算子的输入张量数据的拆分状态集合中的状态进行拼接。

优选地，所述插入胶水算子模块插入的胶水算子用于将胶水算子的输入张量数据的拆分状态集合中的状态进行拆分。

优选地，所述插入胶水算子模块插入的胶水算子用于将胶水算子的输入张量数据的拆分状态集合中的状态进行拼接，再对经拼接处理后的拆分状态集合中的状态进行拆分。

优选地，所述插入胶水算子模块插入的胶水算子用于将胶水算子的输入张量数据的拆分状态集合中的状态进行拆分，再对经拆分处理后的拆分状态集合中的状态进行拼接。

优选地，还包括：

第一拆分状态集合优化模块，用于在正向遍历阶段，当所述算子的输出张量数据被至少两个算子作为输入张量数据，或者所述算子具有至少两个输出张量数据时，所述算子的输出张量数据的拆分状态集合中保留一个拆分状态，且所述拆分状态经由所述算子的同一状态路径确定。

优选地，还包括：

第二拆分状态集合优化模块，用于在反向遍历阶段，当所述算子具有至少两个输入张量数据时，所述算子的输入张量数据的拆分状态集合中保留一个拆分状态，且所述拆分状态经由所述算子的同一状态路径确定。

针对上述技术问题，还提出了一种神经网络模型的拆分方法及相关产品，除了神经网络模型拆分装置的相关描述，相关的拆分方法及相关产品的描述和上述各实施例描述的一致，此处不再加以赘述。关于神经网络模型拆分装置的相关描述如下。

本公开提供一种神经网络模型拆分装置，所述装置包括：

拆分状态集合模块，用于根据所述神经网络模型中目标层的算子，确定与所述目标层的算子关联的张量数据的拆分状态集合；其中，所述目标层为所述神经网络模型中的至少一层；

插入补偿算子模块，用于在所述目标层的算子与关联的拆分状态集合之间插入补偿算子，调整所述算子的输入张量数据的拆分状态集

合中的状态；其中，所述补偿算子用于从所述状态的任一子张量数据的相邻子张量数据中获取目标数据，将所述目标数据与所述子张量数据合并；

状态路径模块，用于根据所述神经网络模型的有向无环图遍历所述拆分状态集合，确定相邻拆分状态集合之间的状态路径及状态路径的权重；其中，所述状态路径表示所述算子的拆分方式；所述拆分状态集合中的每个状态表示一个子张量数据集合，所述状态的所有子张量数据的并集结果为所述张量数据；

目标拆分路径模块，用于根据所述状态路径的权重，确定所述目标层的目标拆分路径；

拆分模块，用于利用所述目标拆分路径对所述神经网络模型的目标层的算子进行拆分。

优选地，所述插入补偿算子模块包括：

插入单元，用于在所述目标层中的特定类型算子与关联的输入张量数据的拆分状态集合之间插入补偿算子；其中，所述特定类型算子的特征为：被用于计算该类算子的输出张量数据的元素对应的输入张量数据的元素，同样被用于计算所述输出张量数据的元素的相邻元素。

优选地，所述插入单元插入的补偿算子适用的特定类型算子为卷积算子、池化算子、局部响应归一化算子。

优选地，所述插入补偿算子模块还包括：

合并单元，用于采用金字塔形式的结构来合并所述目标层中的多个补偿算子。

优选地，所述目标拆分路径确定模块包括：

遍历单元，用于遍历所述目标层的所有拆分状态集合，对当前拆分状态集合，遍历每一状态，获得所有以当前状态为起点的状态路径以及所述状态路径的结束状态到所述目标层的输出张量数据的终止状态的拆分路径；

拆分路径确定单元，用于根据所述状态路径的权重和所述拆分路

径的权重确定所述当前状态到所述目标层的输出张量数据的终止状态的拆分路径；其中，所述拆分路径的权重根据所述拆分路径对应的所有状态路径的权重确定；

选择目标拆分路径单元，用于遍历完所述目标层的所有拆分状态集合后，获得所述目标层的输入张量数据的拆分状态集合与所述目标层的输出张量数据的拆分状态集合之间的目标拆分路径。

优选地，所述神经网络模型拆分装置还包括：

第一拆分状态集合优化模块，用于在正向遍历阶段，当所述算子的输出张量数据被至少两个算子作为输入张量数据，或者所述算子具有至少两个输出张量数据时，所述算子的输出张量数据的拆分状态集合中保留一个拆分状态，且所述拆分状态经由所述算子的同一状态路径确定。

优选地，所述神经网络模型拆分装置还包括：

第二拆分状态集合优化模块，用于在反向遍历阶段，当所述算子具有至少两个输入张量数据时，所述算子的输入张量数据的拆分状态集合中保留一个拆分状态，且所述拆分状态经由所述算子的同一状态路径确定。

针对上述技术问题，还提出了一种神经网络模型的拆分方法及相关产品，除了神经网络模型拆分装置的相关描述，相关的拆分方法及相关产品的描述和上述各实施例描述的一致，此处不再加以赘述。关于神经网络模型拆分装置的相关描述如下。

本公开提出一种神经网络模型拆分装置，其中，包括：

拆分状态集合模块，用于根据所述神经网络模型中目标层的算子，确定与所述目标层的算子关联的张量数据的拆分状态集合；其中，所述目标层为所述神经网络模型中的至少一层；

插入胶水算子模块，用于在所述目标层的算子与关联的拆分状态集合之间插入胶水算子，调整所述算子的张量数据的拆分状态集合中的状态；其中，所述胶水算子用于将所述张量数据按照一拆分方式获得的状态调整成按照任一种拆分方式获得的状态；

插入补偿算子模块,用于在所述目标层的算子与关联的拆分状态集合之间插入补偿算子,调整所述算子的输入张量数据的拆分状态集合中的状态;其中,所述补偿算子用于从所述状态的任一子张量数据的相邻子张量数据中获取目标数据,将所述目标数据与所述子张量数据合并;

状态路径模块,用于根据所述神经网络模型的有向无环图遍历所述拆分状态集合,确定相邻拆分状态集合之间的状态路径及状态路径的权重;其中,所述状态路径表示所述算子的拆分方式;所述拆分状态集合中的每个状态表示一个子张量数据集合,所述状态的所有子张量数据的并集结果为所述张量数据;

目标拆分路径模块,用于根据所述状态路径的权重,确定所述目标层的目标拆分路径;

拆分模块,用于利用所述目标拆分路径对所述神经网络模型的目标层的算子进行拆分。

优选地,所述插入胶水算子模块包括:

第一插入单元,用于在所述目标层的算子与关联的拆分状态集合之间插入胶水算子,获取包含所述胶水算子在内的神经网络模型的有向无环图;

状态路径单元,用于根据所述有向无环图遍历所述目标层的所有张量数据对应的拆分状态集合,确定相邻拆分状态集合之间的状态路径及状态路径的权重;

第一确定目标拆分路径单元,用于根据所述状态路径的权重,确定包含所述胶水算子在内的神经网络模型的目标层的目标拆分路径;

选择单元,用于利用包含所述胶水算子在内的神经网络模型的目标层的目标拆分路径对插入的每个胶水算子进行选择,对不需要插入的胶水算子删除,对需要插入的胶水算子保留。

优选地,所述插入胶水算子模块插入的胶水算子用于将胶水算子的输入张量数据的拆分状态集合中的状态进行拼接。

优选地,所述插入胶水算子模块插入的胶水算子用于将胶水算子

的输入张量数据的拆分状态集合中的状态进行拆分。

优选地，所述插入胶水算子模块插入的胶水算子用于将胶水算子的输入张量数据的拆分状态集合中的状态进行拼接，再对经拼接处理后的拆分状态集合中的状态进行拆分。

优选地，所述插入胶水算子模块插入的胶水算子用于将胶水算子的输入张量数据的拆分状态集合中的状态进行拆分，再对经拆分处理后的拆分状态集合中的状态进行拼接。

优选地，所述插入补偿算子模块包括：

第二插入单元，用于在所述目标层中的特定类型算子与关联的输入张量数据的拆分状态集合之间插入补偿算子；其中，所述特定类型算子的特征为：被用于计算该类算子的输出张量数据的元素对应的输入张量数据的元素，同样被用于计算所述输出张量数据的元素的相邻元素。

优选地，所述第二插入单元插入的补偿算子适用的特定类型算子为卷积算子、池化算子、局部响应归一化算子。

优选地，所述插入补偿算子模块还包括：

合并单元，用于采用金字塔形式的结构来合并所述目标层中的多个补偿算子。

优选地，所述目标拆分路径确定模块包括：

遍历单元，用于遍历所述目标层的所有拆分状态集合，对当前拆分状态集合，遍历每一状态，获得所有以当前状态为起点的状态路径以及所述状态路径的结束状态到所述目标层的输出张量数据的终止状态的拆分路径；

拆分路径确定单元，用于根据所述状态路径的权重和所述拆分路径的权重确定所述当前状态到所述目标层的输出张量数据的终止状态的拆分路径；其中，所述拆分路径的权重根据所述拆分路径对应的所有状态路径的权重确定；

第二确定目标拆分路径单元，用于遍历完所述目标层的所有拆分状态集合后，获得所述目标层的输入张量数据的拆分状态集合与所述

目标层的输出张量数据的拆分状态集合之间的目标拆分路径。

优选地，还包括：

第一拆分状态集合优化模块，用于在正向遍历阶段，当所述算子的输出张量数据被至少两个算子作为输入张量数据，或者所述算子具有至少两个输出张量数据时，所述算子的输出张量数据的拆分状态集合中保留一个拆分状态，且所述拆分状态经由所述算子的同一状态路径确定。

优选地，还包括：

第二拆分状态集合优化模块，用于在反向遍历阶段，当所述算子具有至少两个输入张量数据时，所述算子的输入张量数据的拆分状态集合中保留一个拆分状态，且所述拆分状态经由所述算子的同一状态路径确定。

在本实施方式中，所述存储器可以包括用于存储信息的物理装置，通常是将信息数字化后再以利用电、磁或者光学等方法的媒体加以存储。本实施方式所述的存储器又可以包括：利用电能方式存储信息的装置，如 RAM、ROM 等；利用磁能方式存储信息的装置，如硬盘、软盘、磁带、磁芯存储器、磁泡存储器、U 盘；利用光学方式存储信息的装置，如 CD 或 DVD。当然，还有其他方式的存储器，例如量子存储器、石墨烯存储器等等。

在本实施方式中，所述处理器可以按任何适当的方式实现。例如，所述处理器可以采取例如微处理器或处理器以及存储可由该（微）处理器执行的计算机可读程序代码（例如软件或固件）的计算机可读介质、逻辑门、开关、专用集成电路（Application Specific Integrated Circuit, ASIC）、可编程逻辑控制器和嵌入微控制器的形式等等。

在本实施例中，本申请实施例还提供一种可读存储介质，其上存储有计算机程序，所述计算机程序被执行时实现上述所述的神经网络模型拆分方法。

由上可见，本公开的技术方案可以用较小地开销实现深度学习加速器由单核向多核结构上的扩展，并且能够针对给定的网络和底层加

速器特点给出一种高效的拆分方案。实验结果表明，该方案能有效降低各种网络在多核加速器上的端到端时延。

本领域技术人员也知道，除了以纯计算机可读程序代码方式实现客户端和服务端以外，完全可以通过将方法步骤进行逻辑编程来使得客户端和服务端以逻辑门、开关、专用集成电路、可编程逻辑控制器和嵌入微控制器等的形式来实现相同功能。因此这种客户端和服务端可以被认为是一种硬件部件，而对其内包括的用于实现各种功能的装置也可以视为硬件部件内的结构。或者甚至，可以将用于实现各种功能的装置视为既可以是实现方法的软件模块又可以是硬件部件内的结构。

通过以上的实施方式的描述可知，本领域的技术人员可以清楚地了解到本申请可借助软件加必需的通用硬件平台的方式来实现。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可以以软件产品的形式体现出来，该计算机软件产品可以存储在存储介质中，如 ROM/RAM、磁碟、光盘等，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方式或者实施方式的某些部分所述的方法。

本说明书中的各个实施方式均采用递进的方式描述，各个实施方式之间相同相似的部分互相参见即可，每个实施方式重点说明的都是与其他实施方式的不同之处。尤其，针对客户端和服务端的实施方式来说，均可以参照前述方法的实施方式的介绍对照解释。

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。

虽然通过实施方式描绘了本申请，本领域普通技术人员知道，本

申请有许多变形和变化而不脱离本申请的精神，希望所附的权利要求包括这些变形和变化而不脱离本申请的精神。

权 利 要 求

1.一种神经网络模型拆分方法，其特征在于，所述方法包括：

根据所述神经网络模型中目标层的算子，确定与所述目标层的算子关联的张量数据的拆分状态集合；其中，所述目标层为所述神经网络模型中的至少一层；

根据所述神经网络模型的有向无环图遍历所述拆分状态集合，确定相邻拆分状态集合之间的状态路径及状态路径的权重；其中，所述状态路径表示所述算子的拆分方式；所述拆分状态集合中的每个状态表示一个子张量数据集合，所述状态的所有子张量数据的并集结果为所述张量数据；

根据所述状态路径的权重，确定所述目标层的目标拆分路径；

利用所述目标拆分路径对所述神经网络模型的目标层的算子进行拆分。

2.如权利要求1所述的方法，其特征在于，确定所述目标层的目标拆分路径的步骤包括：

遍历所述目标层的所有拆分状态集合，对当前拆分状态集合，遍历每一状态，获得所有指向当前状态的状态路径以及所述状态路径的起始状态到所述目标层的输入张量数据的起始状态的拆分路径；

根据所述状态路径的权重和所述拆分路径的权重确定所述当前状态到所述目标层的输入张量数据的起始状态的拆分路径；其中，所述拆分路径的权重根据所述拆分路径对应的所有状态路径的权重确定；

遍历完所述目标层的所有拆分状态集合后，获得所述目标层的输入张量数据的拆分状态集合与所述目标层的输出张量数据的拆分状态集合之间的目标拆分路径。

3.如权利要求1所述的方法，其特征在于，确定所述目标层的目标拆分路径的步骤包括：

遍历所述目标层的所有拆分状态集合，对当前拆分状态集合，遍历每一状态，获得所有以当前状态为起点的状态路径以及所述状态路径的结束状态到所述目标层的输出张量数据的终止状态的拆分路径；

根据所述状态路径的权重和所述拆分路径的权重确定所述当前状态到所述

目标层的输出张量数据的终止状态的拆分路径；其中，所述拆分路径的权重根据所述拆分路径对应的所有状态路径的权重确定；

遍历完所述目标层的所有拆分状态集合后，获得所述目标层的输入张量数据的拆分状态集合与所述目标层的输出张量数据的拆分状态集合之间的目标拆分路径。

4.如权利要求1所述的方法，其特征在于，所述神经网络模型的目标层的算子拆分后获得的子算子数量为2的整数次幂。

5.如权利要求1所述的方法，其特征在于，所述神经网络模型的目标层的算子的输入张量数据的拆分状态集合中的状态根据所述算子的计算逻辑和对应输出张量数据的拆分状态集合中的状态确定。

6.如权利要求1所述的方法，其特征在于，所述神经网络模型的目标层的算子的输出张量数据的拆分状态集合中的状态根据所述算子的计算逻辑和对应输入张量数据的拆分状态集合中的状态确定。

7.如权利要求1所述的方法，其特征在于，还包括：

在正向遍历阶段，当所述算子的输出张量数据被至少两个算子作为输入张量数据，或者所述算子具有至少两个输出张量数据时，所述算子的输出张量数据的拆分状态集合中保留一个拆分状态，且所述拆分状态经由所述算子的同一状态路径确定。

8.如权利要求1所述的方法，其特征在于，还包括：

在反向遍历阶段，当所述算子具有至少两个输入张量数据时，所述算子的输入张量数据的拆分状态集合中保留一个拆分状态，且所述拆分状态经由所述算子的同一状态路径确定。

9.如权利要求1所述的方法，其特征在于，所述状态路径的权重根据算子的类型和规模、多核处理器硬件参数确定。

10.一种神经网络模型拆分装置，其特征在于，包括：

拆分状态集合模块，用于根据所述神经网络模型中目标层的算子，确定与所述目标层的算子关联的张量数据的拆分状态集合；其中，所述目标层为所述神经网络模型中的至少一层；

状态路径模块，用于根据所述神经网络模型的有向无环图遍历所述拆分状态

集合，确定相邻拆分状态集合之间的状态路径及状态路径的权重；其中，所述状态路径表示所述算子的拆分方式；所述拆分状态集合中的每个状态表示一个子张量数据集合，所述状态的所有子张量数据的并集结果为所述张量数据；

目标拆分路径模块，用于根据所述状态路径的权重，确定所述目标层的目标拆分路径；

拆分模块，用于利用所述目标拆分路径对所述神经网络模型的目标层的算子进行拆分。

11.如权利要求 10 所述的装置，其特征在于，所述目标拆分路径模块包括：

第一遍历单元，用于遍历所述目标层的所有拆分状态集合，对当前拆分状态集合，遍历每一状态，获得所有指向当前状态的状态路径以及所述状态路径的起始状态到所述目标层的输入张量数据的起始状态的拆分路径；

第一拆分路径确定单元，用于根据所述状态路径的权重和所述拆分路径的权重确定所述当前状态到所述目标层的输入张量数据的起始状态的拆分路径；其中，所述拆分路径的权重根据所述拆分路径对应的所有状态路径的权重确定；

第一选择目标拆分路径单元，用于遍历完所述目标层的所有拆分状态集合后，获得所述目标层的输入张量数据的拆分状态集合与所述目标层的输出张量数据的拆分状态集合之间的目标拆分路径。

12.如权利要求 10 所述的装置，其特征在于，所述目标拆分路径模块包括：

第二遍历单元，用于遍历所述目标层的所有拆分状态集合，对当前拆分状态集合，遍历每一状态，获得所有以当前状态为起点的状态路径以及所述状态路径的结束状态到所述目标层的输出张量数据的终止状态的拆分路径；

第二拆分路径确定单元，用于根据所述状态路径的权重和所述拆分路径的权重确定所述当前状态到所述目标层的输出张量数据的终止状态的拆分路径；其中，所述拆分路径的权重根据所述拆分路径对应的所有状态路径的权重确定；

第二选择目标拆分路径单元，用于遍历完所述目标层的所有拆分状态集合后，获得所述目标层的输入张量数据的拆分状态集合与所述目标层的输出张量数据的拆分状态集合之间的目标拆分路径。

13.如权利要求 10 所述的装置，其特征在于，还包括：

第一拆分状态集合优化模块，用于在正向遍历阶段，当所述算子的输出张量

数据被至少两个算子作为输入张量数据,或者所述算子具有至少两个输出张量数据时,所述算子的输出张量数据的拆分状态集合中保留一个拆分状态,且所述拆分状态经由所述算子的同一状态路径确定。

14.一种神经网络模型拆分硬件设备,包括存储器及处理器,所述存储器上存储有可在处理器上运行的计算机程序,其特征在于,所述处理器执行所述计算机程序时实现权利要求 1~9 中任一项所述方法的步骤。

15.一种计算机可读存储介质,其上存储有计算机程序,其特征在于,所述计算机程序被处理器执行时实现权利要求 1~9 中任一项所述的方法的步骤。

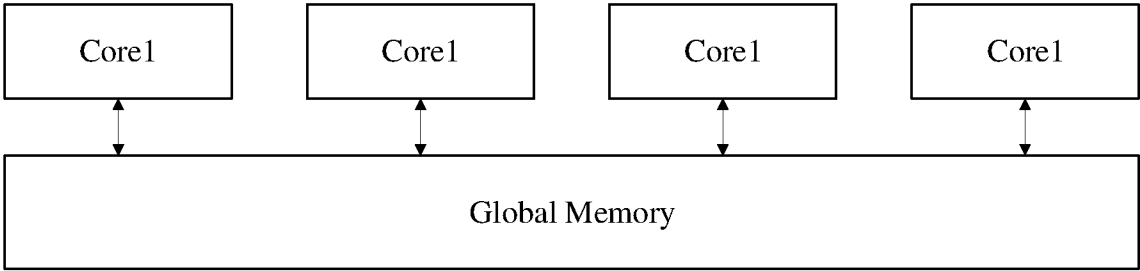


图 1

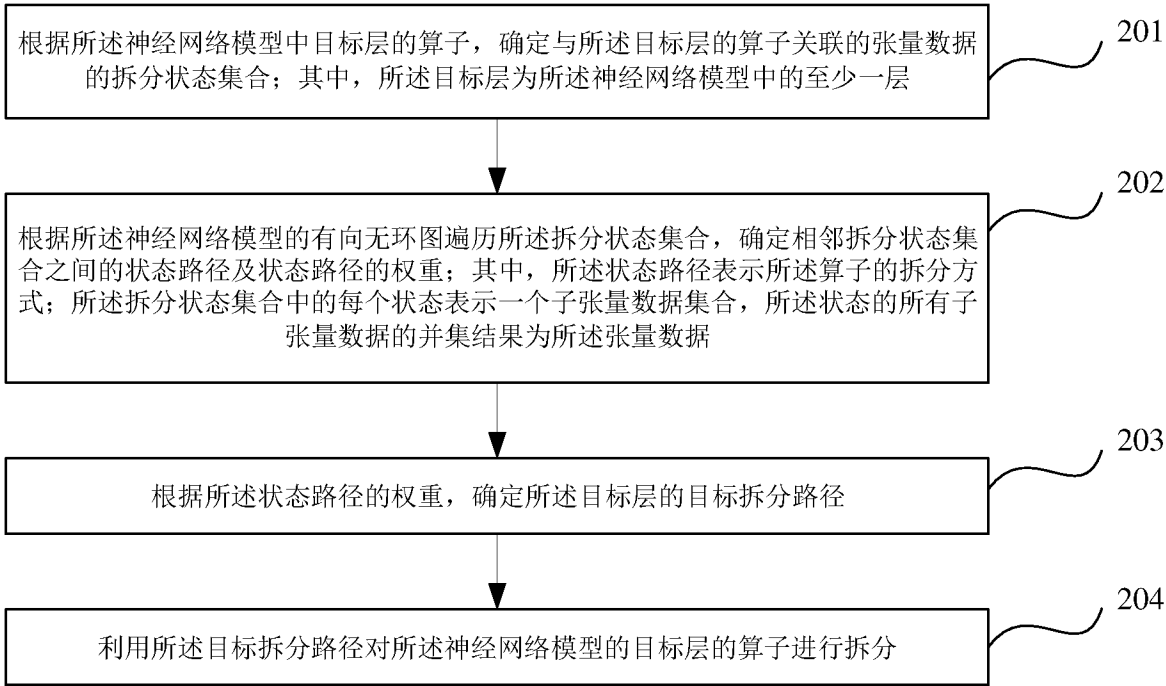


图 2

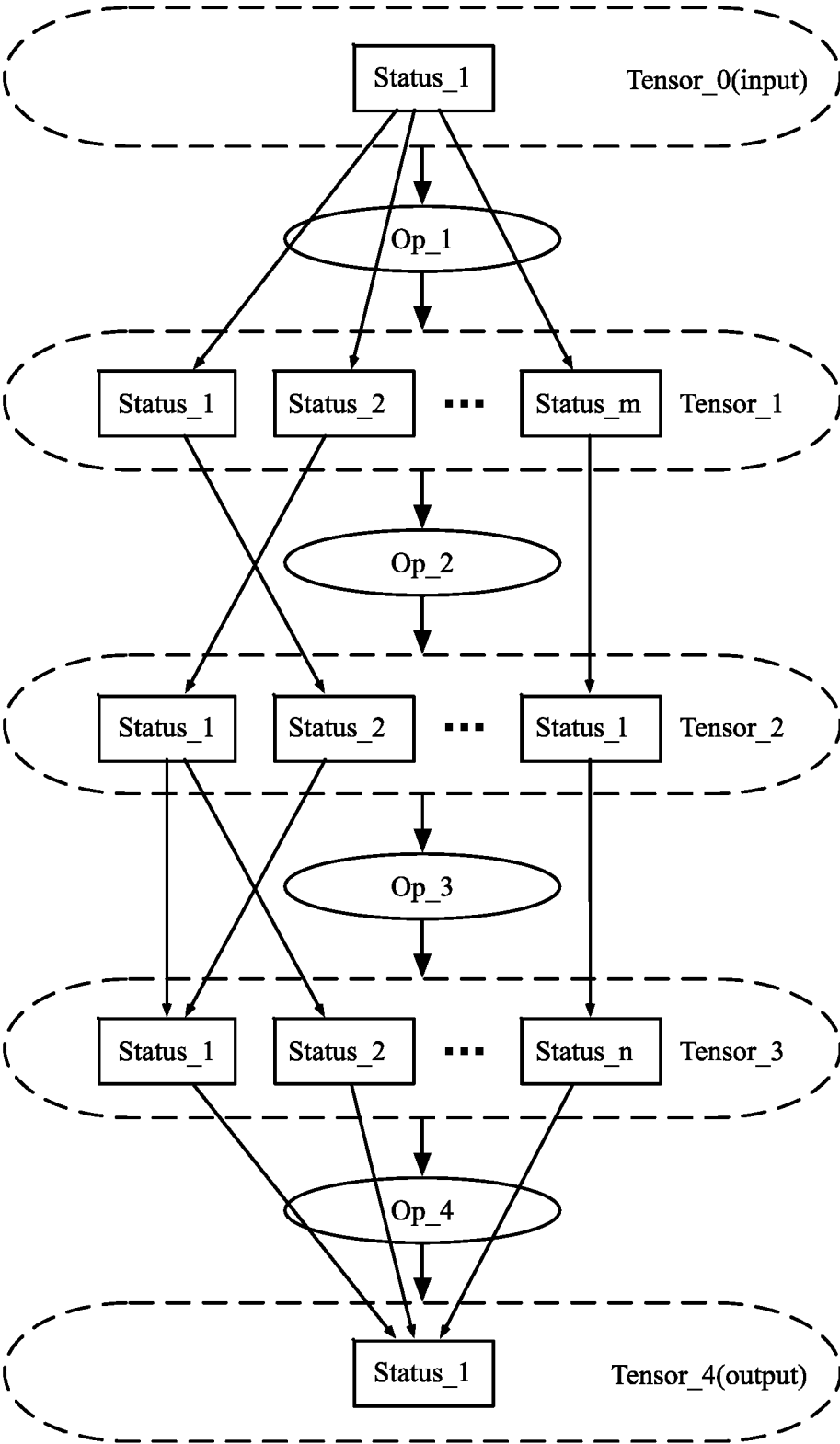


图 3

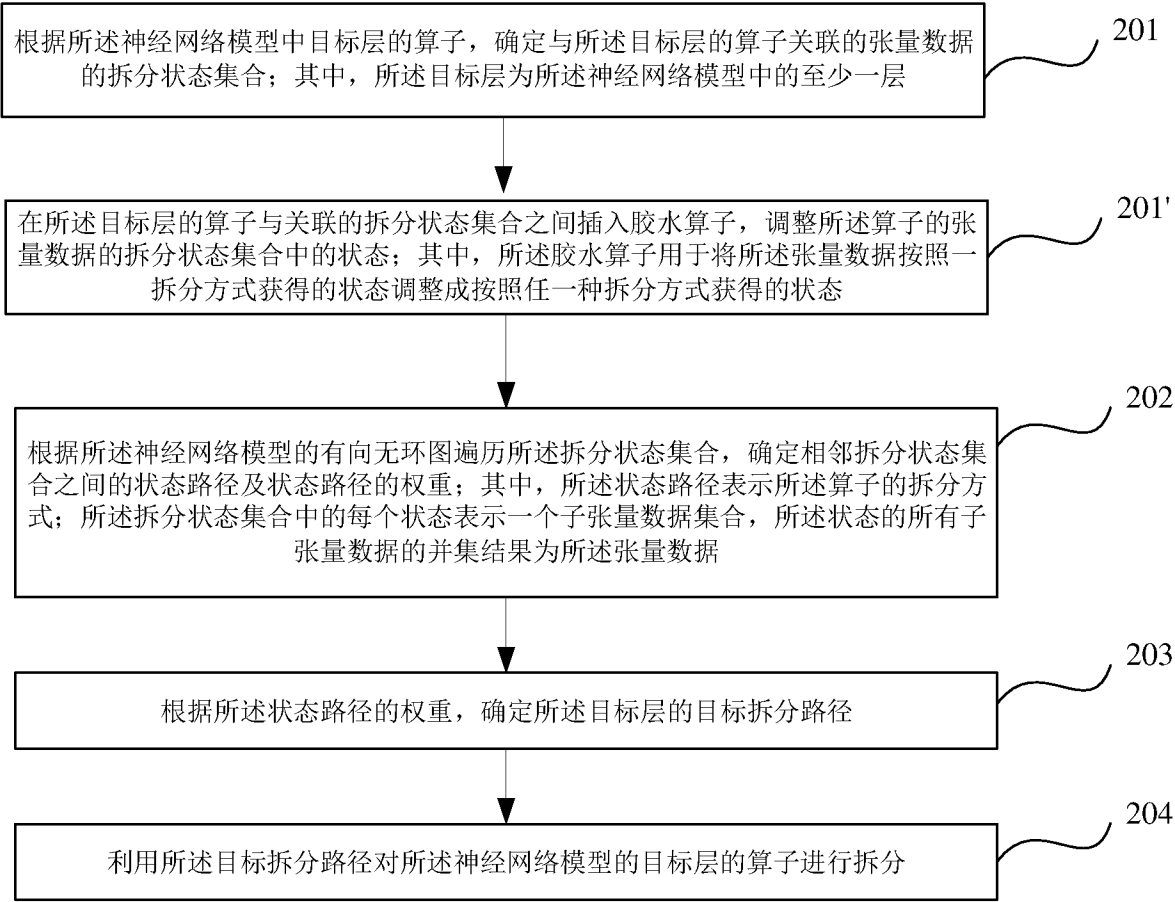


图 4

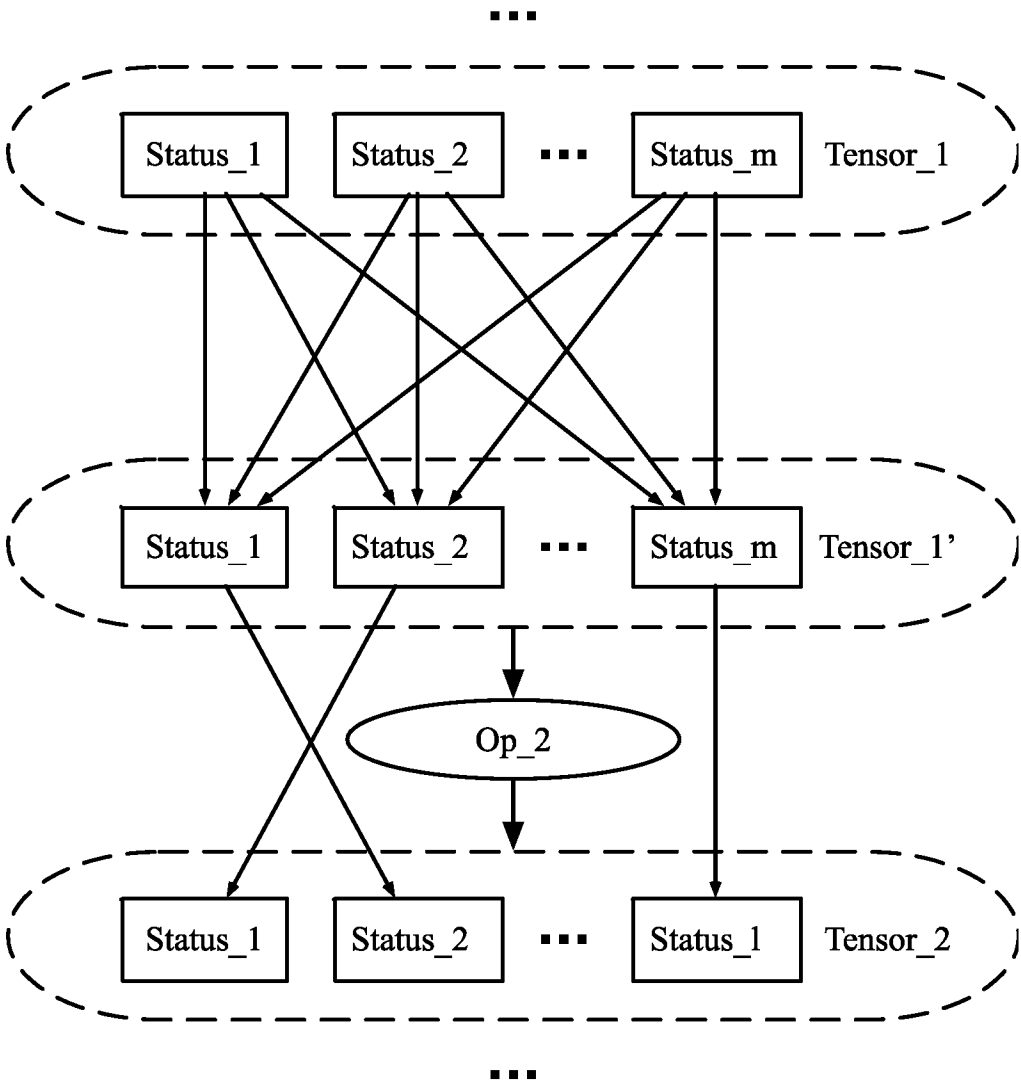


图 5

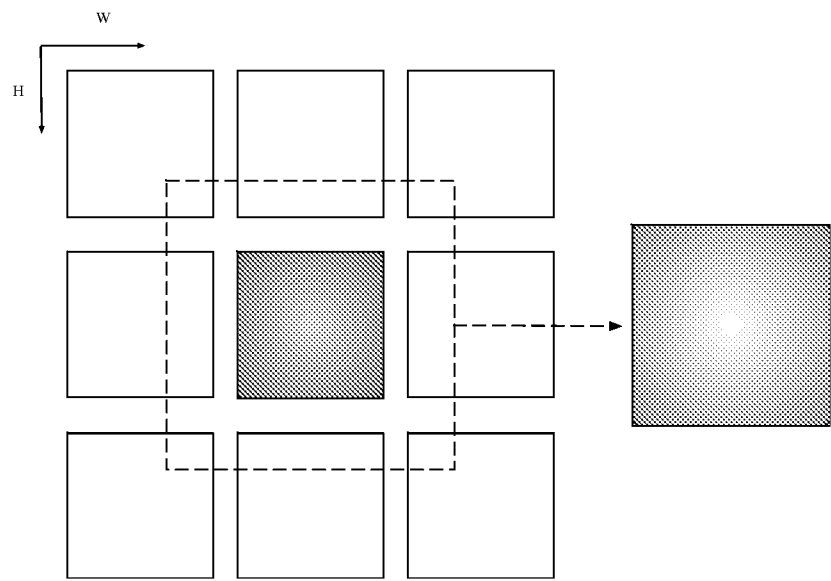


图 6

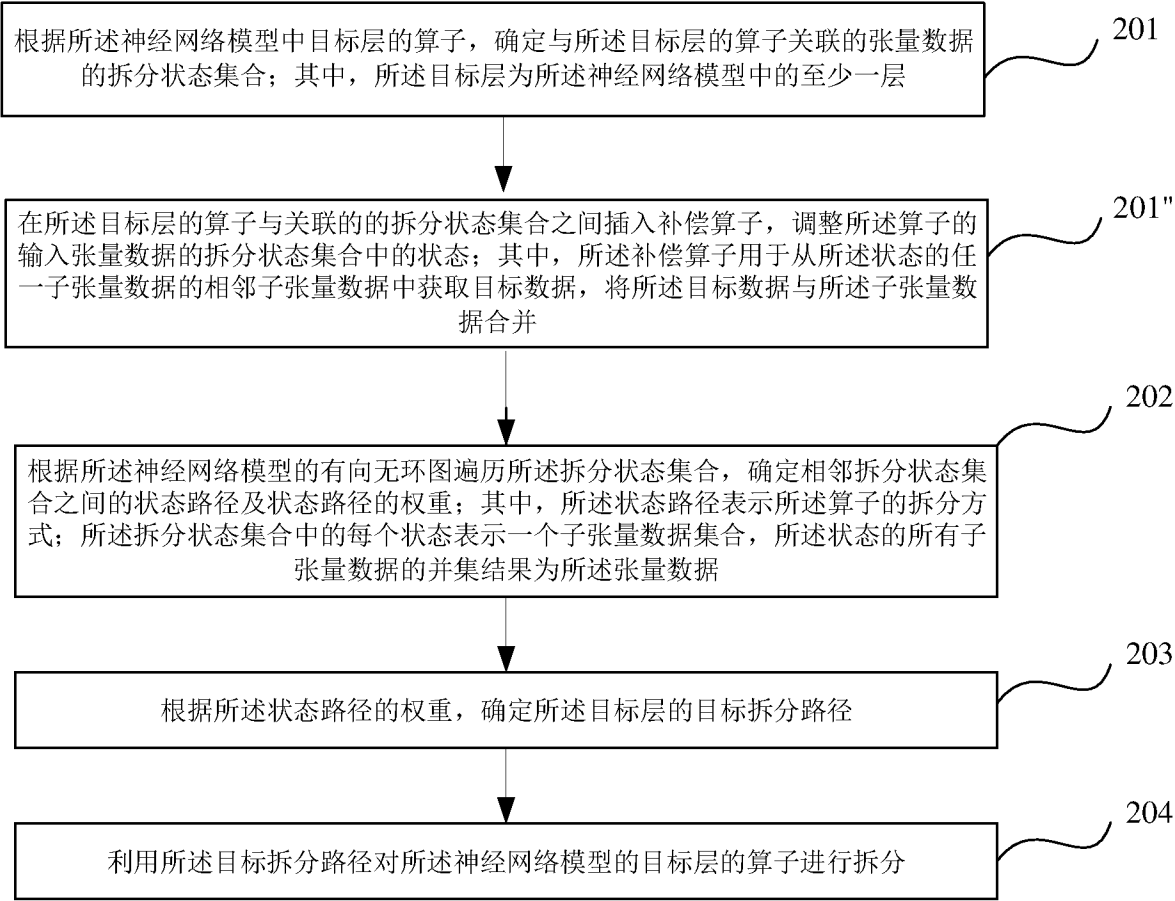


图 7

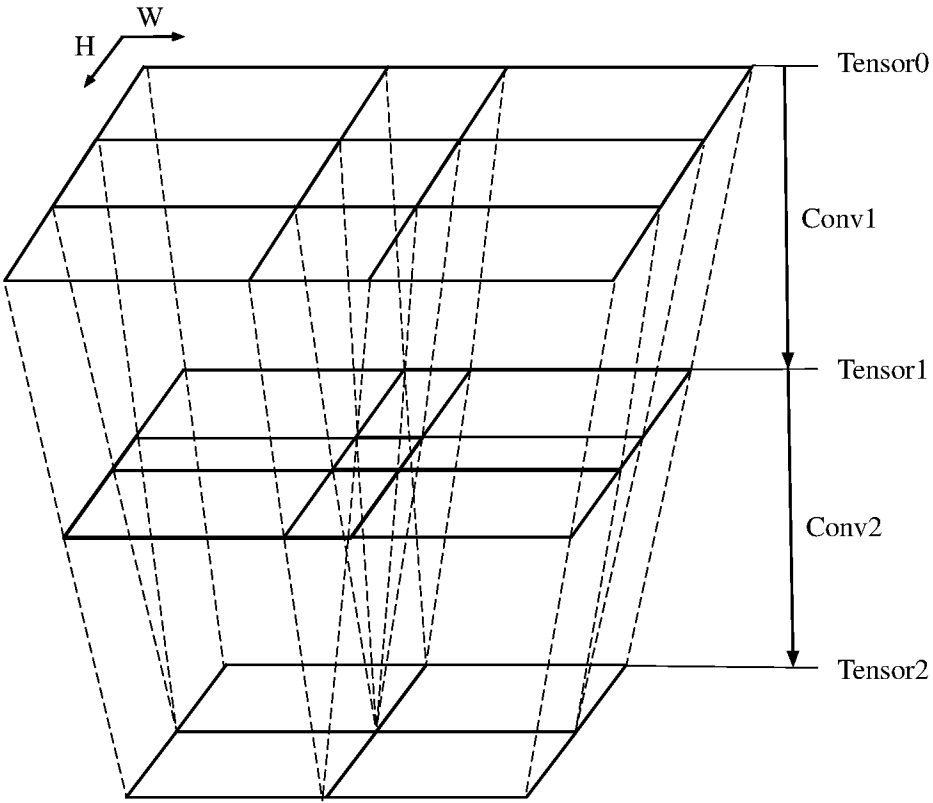


图 8

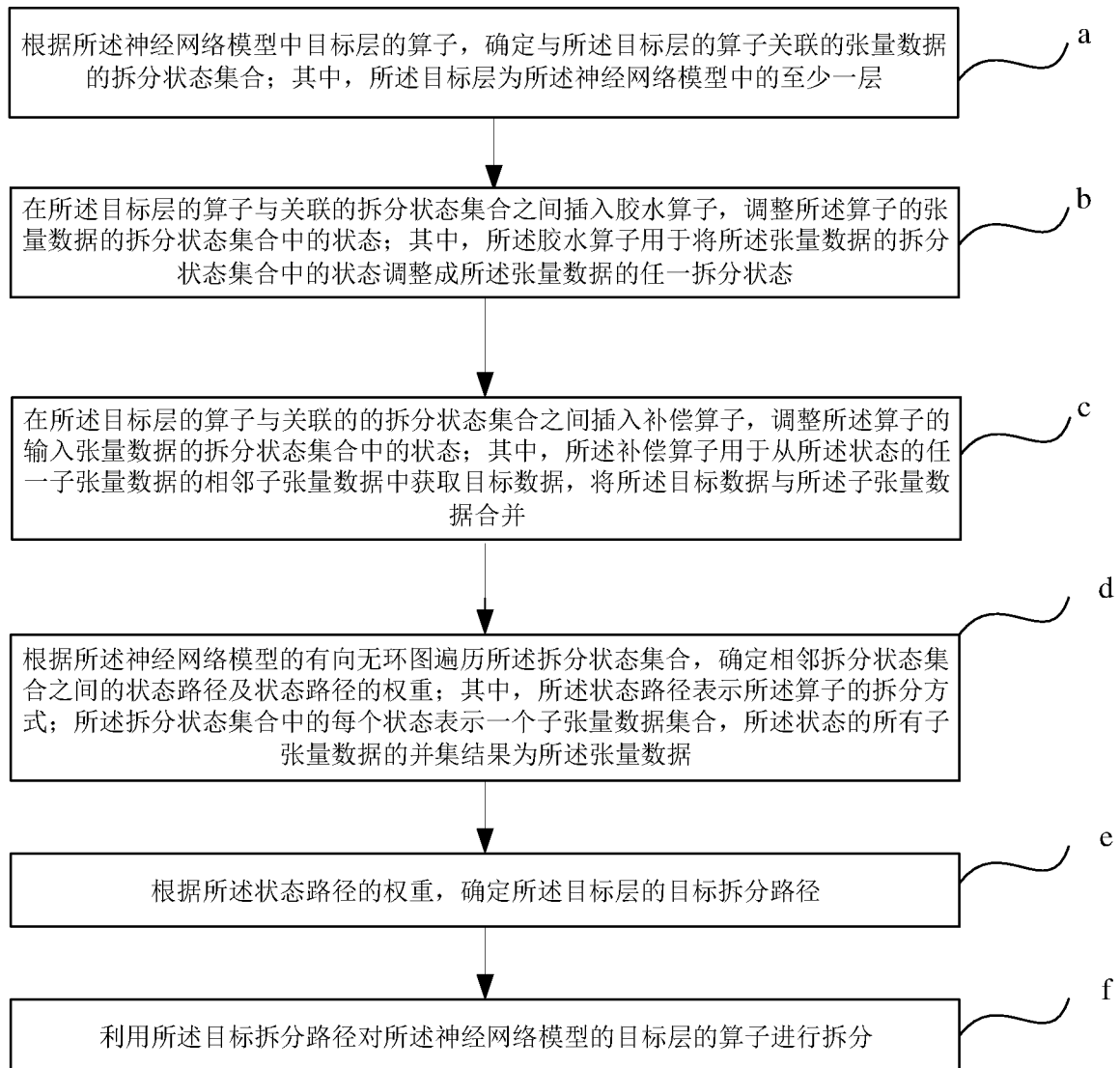


图 9

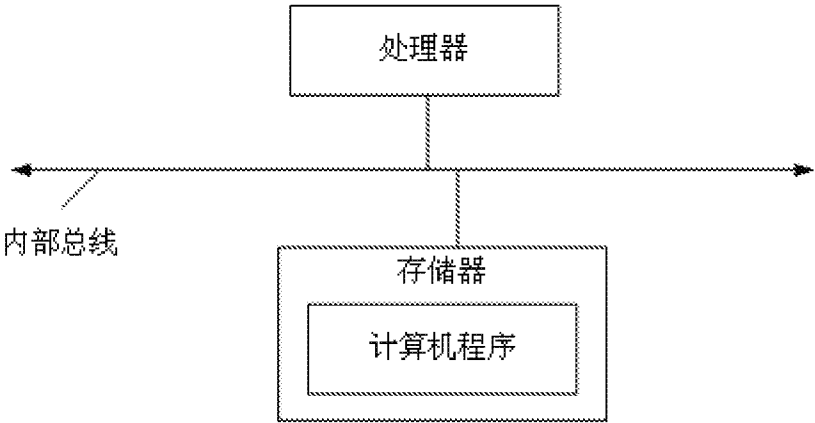


图 10