(19)国家知识产权局

(12)发明专利

(10)授权公告号 CN 111158912 B
(45)授权公告日 2023.04.21

(21)申请号 201911392475.9
(22)申请日 2019.12.30
(65)同一申请的已公布的文献号
申请公布号 CN 111158912 A
(43)申请公布日 2020.05.15
(73)专利权人 天津大学
地址 300072 天津市南开区卫津路92号
(72)发明人 张子儒 管畅 吴华明
(74)专利代理机构 天津市三利专利商标代理有限公司 12107
专利代理师 韩新城
(51)Int.Cl.
G06F 9/50(2006.01)
G06F 9/48(2006.01)
G06N 3/045(2023.01)
G06N 3/08(2023.01)

(56)对比文件
CN 109257429 A,2019.01.22
CN 110535936 A,2019.12.03
CN 110362952 A,2019.10.22
US 2018268298 A1,2018.09.20

审查员 安飞

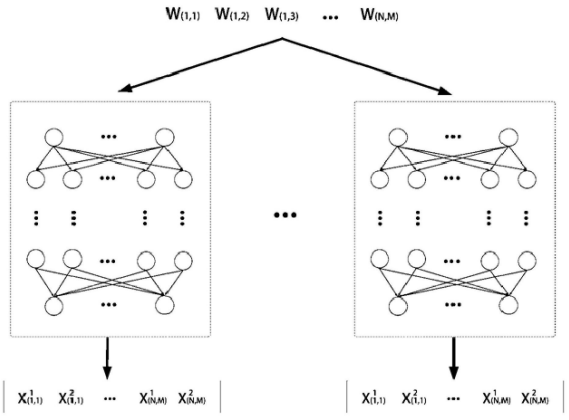
权利要求书2页 说明书6页 附图2页

(54)发明名称

云雾协同计算环境下一种基于深度学习的任务卸载决策方法

(57)摘要

本发明公开一种基于深度学习的任务卸载决策方法，包括步骤：随机生成任务矩阵W分别输入到S个并行的深度神经网络，输出决策X并计算消耗Q(W,X)，得到深度神经网络训练前给出的最优决策X_1及对应的消耗Q_1；从数据集中选取数据训练S个深度神经网络，将任务矩阵W输入训练后的S个深度神经网络，得到深度神经网络训练前给出的最优决策X_2及对应的消耗Q_2；计算$R=\frac{\min(Q_1,Q_2)}{\max(Q_1,Q_2)}$，直到R达到阈值，对新任务卸载决策，将对应任务矩阵W输入并行S个深度神经网络，从中选出消耗最小的决策，即目标决策。本发明基于深度神经网络的决策模型训练完成后，仅需简单的线性运算便可给出决策，运算量大大减小。



CN 111158912 B

1.基于深度学习的任务卸载决策方法,其特征在于,包括以下步骤:

S1.随机生成任务矩阵W分别输入到S个并行的深度神经网络,通过MSE将输出转换为决策X并计算消耗Q(W,X),得到深度神经网络训练前给出的最优决策X_1以及对应的消耗Q_1;

S2.从数据集中随机选取一系列数据训练S个深度神经网络,更新网络权重,将任务矩阵W输入S个训练后的深度神经网络,得到深度神经网络训练后给出的最优决策X_2以及对应的消耗Q_2;

S3.计算$R=\frac{\min(Q_1,Q_2)}{\max(Q_1,Q_2)}$,判断R是否达到阈值,若是结束,否则重复S1-S3;

S4.对于新的任务卸载决策,将所对应的任务矩阵W输入并行的S个深度神经网络中,从中选出消耗最小的决策,即为目标决策;

所述数据集通过以下步骤多次重复进行而形成:

随机生成任务矩阵W,将任务矩阵W分别输入到S个初始化后的深度神经网络中,得到S个决策X并计算每个决策所对应的消耗Q(W,X),将消耗最小的决策与任务矩阵W联合存入数据集中;多次重复本步骤,直到生成的数据数量达到设定好的数据集大小;

所述数据集是通过将任务矩阵W与对应的第一最优决策X_1联合生成新的数据替代数据集中最先生成的数据,以实现对数据集更新的。

2.根据权利要求1所述基于深度学习的任务卸载决策方法,其特征在于,所述消耗Q(W,X)的计算方法如下:

$$Q(W,X)=\sum_{n=1}^{N}\left(E_{(n)}+aT_{(n)}\right),T_{(n)}=\max\left(Tl_{(n)},Te_{(n)},Tc_{(n)}\right),E_{(n)}=\sum_{n=1}^{N}E_{(n,m)},$$

式中,a表示能量消耗与时间消耗之间的权重,$T_{(n)}$表示用户n最终时间消耗,$E_{(n)}$表示用户n的总能量消耗,$E_{(n,m)}$表示用户n的m个任务的总能量消耗,$Tl_{(n)}$表示用户n的本地运算任务消耗总时间,$Te_{(n)}$表示用户n的边缘云运算任务消耗总时间,$Tc_{(n)}$表示用户n的中央云运算任务消耗总时间,N为用户数量,每个用户n有M个任务。

3.根据权利要求2所述基于深度学习的任务卸载决策方法,其特征在于,其中,

$$Tl_{(n)}=\sum_{m=1}^{M}Tl_{(n,m)}x_{(n,m)}^{1},$$

$$Te_{(n)}=\sum_{m=1}^{M}Te_{(n,m)}(1-x_{(n,m)}^{1})x_{(n,m)}^{2},$$

$$Tc_{(n)}=\sum_{m=1}^{M}Tc_{(n,m)}\left(1-x_{(n,m)}^{1}\right)\left(1-x_{(n,m)}^{2}\right),$$

$x_{(n,m)}^{1}=1$,表示本地执行,

$x_{(n,m)}^{1}=0x_{(n,m)}^{2}=1$,表示卸载到中央云,

$x_{(n,m)}^{2}=0$,表示卸载到边缘云,

$x_{(n,m)}^{1}$、$x_{(n,m)}^{2}$表示两个取值为0或1的变量。

4.根据权利要求3所述基于深度学习的任务卸载决策方法,其特征在于,其中,

$$E_{(n,m)}=x_{(n,m)}^{1}El_{(n,m)}$$

$$+\left(1-x_{(n,m)}^{1}\right)\left(x_{(n,m)}^{2}Ee_{(n,m)}+\left(1-x_{(n,m)}^{2}\right)Ec_{(n,m)}\right),$$

$El_{(n,m)}$、$Ee_{(n,m)}$、$Ec_{(n,m)}$分别表示该任务在本地、边缘云、中央云计算所需消耗的能量。

5.根据权利要求1所述基于深度学习的任务卸载决策方法，其特征在于，其中，每个神经网络输入层含N*M个节点，输入为任务矩阵各元素的值，输出层含2*N*M个节点，输出为预决策X^*，输入层与输出层间含若干隐藏层；

将预决策X^*转化为由0,1构成的决策X，采用均方差公式MSE，其中决策X是使

$$MSE = \sum_{i=1}^{2*N*M} (x_i^* - x_i)^2,$$

式中，x_i表示神经网络各节点的输出值，x_i^*表示神经网络各节点的输出值对应的预决策的输出值。

云雾协同计算环境下一种基于深度学习的任务卸载决策方法

技术领域

[0001] 本发明涉及任务卸载决策技术领域，特别是涉及云雾协同计算环境下一种基于深度学习的任务卸载决策方法。

背景技术

[0002] 随着科技不断进步及人们生活质量提高，各种移动设备或物联网设备普及为人们生活带来了极大的便利。由于移动设备运算能力有限，对运算密集型程序如人脸识别、增强现实等，其处理速度往往很难满足用户日常需求。

[0003] 为防止移动设备因运行大量运算所带来的高延迟与高电量消耗，设备在日常运行中往往需要依赖于云端服务器协助进行计算，将本地任务卸载到云端服务器运行，从而减少等待时间并延长电池使用寿命。云端服务器的运算能力虽然强大，但由于移动端用户的不断增多，用户对云端服务器运算能力的需求与日俱增，云端运算所带来的延迟也随之逐渐增长。对一些延迟敏感性任务，云计算已渐渐变得不符合程序的实用要求。而随着无线通信技术的不断发展，用户将本地任务卸载到所在地周边的数据基站、数据中心等边缘云设备进行运算的“雾计算”技术已成熟。与中央云相比，边缘云的计算能力与存储能力相对较低，但由于其离移动设备更近，其通信开销会变得很小，可以很大程度上减少网络操作等带来的时延，从而满足未来网络的超高带宽、超低延时以及业务和用户感知的要求，有极大的实际使用价值。

[0004] 雾计算与云计算各有优缺点，云计算的计算能力更充足，但延迟较高；雾计算的延迟极低，但其自身运算能力与存储能力有限。考虑二者的优缺点，只有将两者很好的结合起来，才能使云雾协同计算发挥出其最大的效力。综合考虑所有用户的延迟总时间及能量总消耗，如何动态的决定每个任务的卸载方式，使得所有用户的等待总时间最短及能量总消耗最小，是影响云雾协同计算效力的关键。但总决策的可能性与用户、任务数量呈指数型增长关系，实际情况中往往又涉及到大规模的卸载决策问题，使得遍历或线性规划等传统优化方法需要进行大量运算才能做出决策，很难满足实际需求。

[0005] 也就是说，现有的云雾协同卸载决策算法需要经过大量的运算才能给出，虽然理论可实现，但等待时间过长、能量消耗过大等因素使得其不符合实际使用需求。同时，现有的方法对于实际中给定的任何用户数及任务情况都需要重复同样的计算过程，已有的决策过程无法指导新任务的卸载决策，故模型无法随着使用不断跟新完善。

发明内容

[0006] 本发明的目的是针对现有技术中存在的技术缺陷，而提供一种云雾协同计算环境下基于深度学习的任务卸载决策方法，通过构建并行的多个深度神经网络，采用多神经网络的深度学习的方法，以达到在短时间内作出合理决策的目的。

[0007] 为实现本发明的目的所采用的技术方案是：

[0008] 一种基于深度学习的任务卸载决策方法，

[0009] S1.随机生成任务矩阵W分别输入到S个并行的深度神经网络,通过MSE将输出转换为决策X并计算消耗Q(W,X),得到深度神经网络训练前给出的最优决策X_1以及对应的消耗Q_1;

[0010] S2.从数据集中随机选取一系列数据训练S个深度神经网络,更新网络权重,将任务矩阵W输入S个训练后的深度神经网络,得到深度神经网络训练后给出的最优决策以及对应的消耗Q_2;

[0011] S3.计算$R=\frac{\min(Q_1,Q_2)}{\max(Q_1,Q_2)}$,判断R是否达到阈值,若是结束,否则重复S1-S3;

[0012] S4.对于新的任务卸载决策,将所对应的任务矩阵W输入并行的S个深度神经网络中,从中选出消耗最小的决策,即为目标决策。

[0013] 其中,所述数据集通过以下步骤多次重复进行而形成:

[0014] 随机生成任务矩阵W,将任务矩阵W分别输入到S个初始化后的深度神经网络中,得到S个决策X并计算每个决策所对应的消耗Q(W,X),将消耗最小的决策与任务矩阵W联合存入数据集中;多次重复本步骤,直到生成的数据数量达到设定好的数据集大小。

[0015] 其中,所述数据集是通过将任务矩阵W与对应的第一最优决策X_1联合生成新的数据替代数据集中最先生成的数据,以实现对数据集更新的。

[0016] 其中,所述消耗Q(W,X)的计算方法如下:

[0017] $Q(W,X)=\sum_{n=1}^{N}\left(E_{(n)}+aT_{(n)}\right)$, $T_{(n)}=\max\left(Tl_{(n)},Te_{(n)},Tc_{(n)}\right)$, $E_{(n)}=\sum_{n=1}^{N}E_{(n,m)}$,

[0018] 式中,a表示能量消耗与时间消耗之间的权重,$T_{(n)}$表示用户n最终时间消耗,$E_{(n)}$表示用户n的总能量消耗,$E_{(n,m)}$表示用户n的m个任务的总能量消耗,$Tl_{(n)}$表示用户n的本地运算任务消耗总时间,$Te_{(n)}$表示用户n的边缘云运算任务消耗总时间,$Tc_{(n)}$表示用户n的中央云运算任务消耗总时间,N为用户数量,每个用户n有M个任务。

[0019] 其中,$Tl_{(n)}=\sum_{m=1}^{M}Tl_{(n,m)}x_{(n,m)}^{1}$,

[0020] $Te_{(n)}=\sum_{m=1}^{M}Te_{(n,m)}(1-x_{(n,m)}^{1})x_{(n,m)}^{2}$,

[0021] $Tc_{(n)}=\sum_{m=1}^{M}Tc_{(n,m)}\left(1-x_{(n,m)}^{1}\right)\left(1-x_{(n,m)}^{2}\right)$,

[0022] $x_{(n,m)}^{1}=1$,表示本地执行,

[0023] $x_{(n,m)}^{1}=0x_{(n,m)}^{2}=1$,表示卸载到中央云,

[0024] $x_{(n,m)}^{2}=0$,表示卸载到边缘云,

[0025] $x_{(n,m)}^{1}$、$x_{(n,m)}^{2}$表示两个取值为0或1的变量。

[0026] 其中,$E_{(n,m)}=x_{(n,m)}^{1}El_{(n,m)}$

[0027] $+\left(1-x_{(n,m)}^{1}\right)\left(x_{(n,m)}^{2}Ee_{(n,m)}+\left(1-x_{(n,m)}^{2}\right)Ec_{(n,m)}\right)$,

[0028] $El_{(n,m)}$、$Ee_{(n,m)}$、$Ec_{(n,m)}$分别表示该任务在本地、边缘云、中央云计算所需消耗的能

量。

[0029] 其中，每个神经网络输入层含N*M个节点，输入为任务矩阵各元素的值，输出层含2*N*M个节点，输出为预决策X^*，输入层与输出层间含若干隐藏层；

[0030] 将预决策X^*转化为由0，1构成的决策X，采用均方差公式MSE，其中决策X是使

[0031] $$MSE = \sum_{i=1}^{2*N*M} (x_i^* - x_i)^2 ,$$

[0032] 式中，x_i表示神经网络各节点的输出值，x_i^*表示神经网络各节点的输出值对应的预决策的输出值。

[0033] 本发明能够在非常短的时间内给出合适的卸载方案，从而使云雾协同计算发挥其最大效力，解决了传统卸载决策方案往往需进行大量运算，当任务数增多时，作出决策所需要的高延迟，使得云雾协同计算很难发挥其优势的问题。

附图说明

[0034] 图1是本发明的种基于深度学习的任务卸载决策方法的流程图；

[0035] 图2是本发明的深度神经网络的结构示意图。

具体实施方式

[0036] 以下结合附图和具体实施例对本发明作进一步详细说明。应当理解，此处所描述的具体实施例仅仅用以解释本发明，并不用于限定本发明。

[0037] 本发明通过构建多个并行的深度神经网络，利用多个神经网络并行生成训练所需数据集，在运用数据集训练神经网络的同时用新生成的数据不断更新数据集，从而生成满足实际需求的决策生成神经网络，从而为用户在较短的时间内给出精度较高的任务卸载方案。

[0038] 如图1所示，本发明基于深度学习的任务卸载决策方法，包括以下步骤：

[0039] 一、建立模型。

[0040] 记给定情况下的用户数量为N，每个用户有M个任务(M取各用户任务数的最大值，任务数量不足M的可用大小为0的任务补齐)，用任务矩阵为W表示各任务的大小，其中W的第n行、第m列元素$W_{(n,m)}$表示第n个用户的第m个任务的大小，矩阵W已知。记$El_{(n,m)}$、$Ee_{(n,m)}$、$Ec_{(n,m)}$分别表示该任务在本地、边缘云、中央云计算所需消耗的能量，$Tl_{(n,m)}$、$Te_{(n,m)}$、$Tc_{(n,m)}$分别表示该任务在本地、边缘云、中央云运算所需要的时间，$Tt_{(n,m)}$表示该任务数据上传所需时间，这六个变量均可看作由任务数据大小$W_{(n,m)}$所唯一确定的物理量，故可看作已知量。由于通常情况下任务运行结果与任务数据大小相比过小，故任务运算结果传回本地时间忽略不计，仅考虑任务数据上传时间。

[0041] 对任务$W_{(n,m)}$，运用两个取值为0或1的变量$x_{(n,m)}^1$、$x_{(n,m)}^2$表示任务的卸载方式。其中，当$x_{(n,m)}^1 = 1$时表示任务在本地进行运算。当$x_{(n,m)}^1 = 0$时表示任务选择卸载到云端进行运算，此时，$x_{(n,m)}^2 = 1$时表示卸载到边缘云，$x_{(n,m)}^2 = 0$表示卸载到中央云。

[0042] 则其卸载方式可表示如下：

[0043] $x^1_{(n,m)}=1$——本地执行，

[0044] $x^1_{(n,m)}=0 x^2_{(n,m)}=1$——卸载到中央云，

[0045] $x^2_{(n,m)}=0$——卸载到边缘云，

[0046] 则最终能量消耗由下式表示：

[0047] $$E_{(n,m)}=x^1_{(n,m)}El_{(n,m)}$$

[0048] $$+\left(1-x^1_{(n,m)}\right)\left(x^2_{(n,m)}Ee_{(n,m)}+\left(1-x^2_{(n,m)}\right)Ec_{(n,m)}\right) \quad (1)$$

[0049] 由此，用户n的总能量消耗$E_{(n)}$可表示为：

[0050] $$E_{(n)}=\sum_{n=1}^{N}E_{(n,m)} \quad (2)$$

[0051] 对用户n，其本地运算任务消耗总时间可表示为：

[0052] $$Tl_{(n)}=\sum_{m=1}^{M}Tl_{(n,m)}x^1_{(n,m)} \quad (3)$$

[0053] 边缘云运算任务消耗总时间可表示为：

[0054] $$Te_{(n)}=\sum_{m=1}^{M}Te_{(n,m)}(1-x^1_{(n,m)})x^2_{(n,m)} \quad (4)$$

[0055] 中央云运算任务消耗总时间可表示为：

[0056] $$Tc_{(n)}=\sum_{m=1}^{M}Tc_{(n,m)}\left(1-x^1_{(n,m)}\right)\left(1-x^2_{(n,m)}\right) \quad (5)$$

[0057] 由于在等待云端数据处理的同时，用户可在本地运行未卸载的任务，且中央云与边缘云可分别进行数据处理，所以用户等待总时间为这三部分时间的最长值，即用户n最终时间消耗$T_{(n)}$可由下式表示：

[0058] $T_{(n)}=\max(Tl_{(n)},Te_{(n)},Tc_{(n)})$ (6)

[0059] 用a表示能量消耗与时间消耗之间的权重，则对给定的卸载决策X，总消耗Q可表示为：

[0060] $$Q(W,X)=\sum_{n=1}^{N}\left(E_{(n)}+aT_{(n)}\right) \quad (7)$$

[0061] 二、初始化训练所需要的数据集。

[0062] 2.1随机初始化S个并行的深度神经网络，其中每个神经网络输入层含N*M个节点，输入为任务矩阵各元素的值，输出层含2*N*M个节点，输出为预决策X^*。输入层与输出层间含若干隐藏层。

[0063] 设定数据集大小，规定需要保存的数据的数量。

[0064] 2.2由神经网络计算方式可知，神经网络各节点输出为小数，不符合模型建立中用0和1代表决策的方法。为将预决策X^*转化为与之最接近的由0，1构成的决策X，可采用均方差公式MSE将预决策X^*进行转化。其中X是使

[0065] $$MSE = \sum_{i=1}^{2*N*M} (x_i^* - x_i)^2 \qquad (8)$$

[0066] 最小的一系列有0，1构成的整数。

[0067] 不难证明当$x_i^* > \frac{1}{2}$时，$(x_i^* - 1)^2 < (x_i^* - 0)^2$，故$x_i$取值为1，当$x_i^* \leqslant \frac{1}{2}$时，$(x_i^* - 1)^2 \geq (x_i^* - 0)^2$，故$x_i$取值为0。则通过将神经网络各节点的输出值与$\frac{1}{2}$进行比较，可转化为符合要求的决策X。

[0068] 2.3随机生成任务矩阵W，将W分别输入到S个深度神经网络中，由过程2.2可得到S个决策方案$X_1, X_2, X_3, \cdots, X_S$.根据公式Q(W,X)，分别计算出这S个决策方案所对应的消耗，将消耗最小的决策方案与任务矩阵联合，存入数据集中。

[0069] 2.4重复过程2.3，直到生成的数据数量已达到设定好的数据集大小。

[0070] 三、根据数据集对神经网络进行训练，同时不断用新的数据更新数据集。

[0071] 3.1继续随机生成任务矩阵W，重复过程2.3，得到深度神经网络给出的最优决策X_1，及对应的消耗Q_1，并将该任务矩阵与决策联合生成新的数据，用该数据替代数据集中最先生成的数据，从而实现对数据集更新。

[0072] 3.2从数据集中随机挑选出一系列数据，对这S个深度神经网络训练，更新神经网络权重。

[0073] 3.3对任务矩阵W再次执行过程2.3，得到训练后的神经网络所给出的最优决策X_2及对应的消耗Q_2。

[0074] 3.4定义$R = \frac{\min(Q_1,Q_2)}{\max(Q_1,Q_2)}$，理想状态下，当深度神经网络趋于收敛时，对相同任务矩阵，再次对神经网络进行训练后所给出的方案不再变化，即$Q_1 = Q_2$，R＝1，故把R作为衡量神经网络是否收敛的指标，重复过程3.1-3.3，直到R趋于1，此时可认为神经网络已训练完毕。

[0075] 对于新任务卸载决策情况，将该情况对应的任务矩阵输入并行的S个深度神经网络中，从中挑选出消耗最小的决策，即可得到理想的决策方案。

[0076] 需要说明的是，深度学习是机器学习的一种，通过深度神经网络学习标准样本数据的内在规律和表示层次，使得计算机能够像人一样具有分析能力，对新产生的情况，计算机能够根据已有的训练结果给出合理决策的方法和理论。

[0077] 传统深度学习案例往往仅需一个深度神经网络，并且需要大量已知数据作为训练基础，但由于用户数与任务数的组合多种多样，往往很难得到满足各种实际需求的数据，故理论上可行但很难应用于生产实践上，而本发明引入了多个并行的深度神经网络，通过不断循环更新数据与训练过程，很好解决了缺乏数据的问题，使得该模型能更加便捷的应用

于不同的实际情况。

[0078] 现有的决策模型大多需要进行大量运算，虽然理论上可行，但当用户和任务数增多时运算量往往超过实际可接受范围。而本发明的运算量增长速度较现有方案更平缓，在运算能力等条件相同的情况下可以处理更复杂的决策问题。

[0079] 现有技术每一次做出新决策时，都需要重复已有运算过程，已有决策数据无法指导新决策的生成。本发明基于深度神经网络的决策模型在训练完成后，仅需进行简单的线性运算便可给出决策，运算量大大减小，故可将训练好的神经网络直接复制到各云端服务器甚至移动端直接投入使用，可移植性高。

[0080] 总之，本发明提出的多神经网络的深度学习是一种通过并行构建多个深度神经网络，在没有数据的情况下先随机生成深度学习所需数据集，在训练深度神经网络的同时更新数据集，使得数据集不断趋向于标准的数据，同时使深度神经网络不断趋于收敛，从而将非监督学习过程转化为监督学习的过程。

[0081] 本发明中的深度神经网络采用现有的深度学习工具箱所提供的深度神经网络构建功能、交叉熵计算功能、Adam优化算法等功能，综合考虑用户能量消耗及时间消耗，能快速给出能满足实际需求的卸载决策方案，模型训练完毕后可极大程度上减小给出决策所需要的运算量，且精确度较高。

[0082] 以上所述仅是本发明的优选实施方式，应当指出的是，对于本技术领域的普通技术人员来说，在不脱离本发明原理的前提下，还可以做出若干改进和润饰，这些改进和润饰也应视为本发明的保护范围。

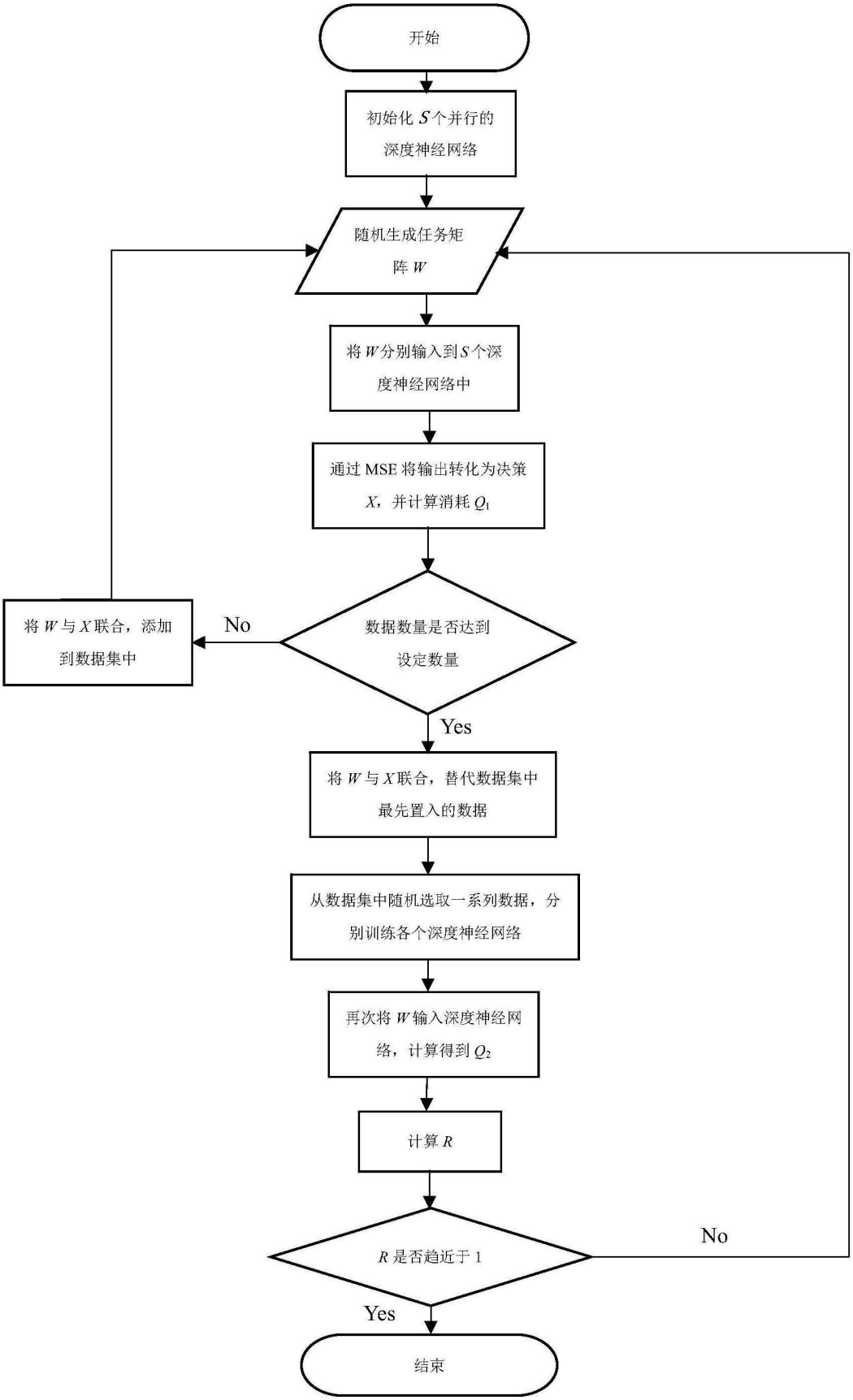

开始
初始化 S 个并行的深度神经网络
随机生成任务矩阵 W
将 W 分别输入到 S 个深度神经网络中
通过 MSE 将输出转化为决策 X，并计算消耗 Q_1
数据数量是否达到设定数量
No
将 W 与 X 联合，添加到数据集中
Yes
将 W 与 X 联合，替代数据集中最先置入的数据
从数据集中随机选取一系列数据，分别训练各个深度神经网络
再次将 W 输入深度神经网络，计算得到 Q_2
计算 R
R 是否趋近于 1
No
Yes
结束

图1

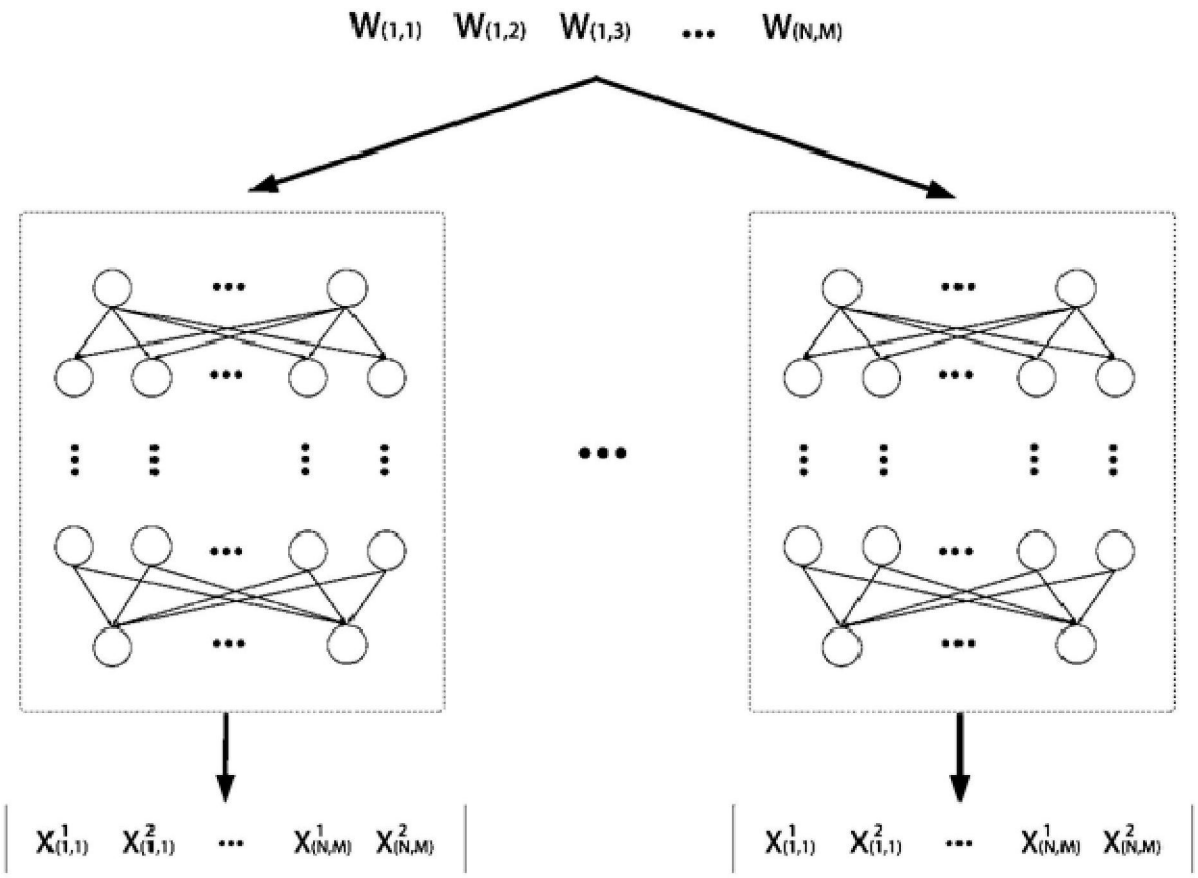


图2