(19)中华人民共和国国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 103282891 A
(43)申请公布日 2013.09.04

(21)申请号 201180047106.1
(22)申请日 2011.08.16
(30)优先权数据
12/857,339 2010.08.16 US
(85)PCT申请进入国家阶段日
2013.03.29
(86)PCT申请的申请数据
PCT/US2011/047975 2011.08.16
(87)PCT申请的公布数据
WO2012/024329 EN 2012.02.23
(71)申请人 甲骨文国际公司
地址 美国加利福尼亚
(72)发明人 D·D·特鲁纳格瑞
(74)专利代理机构 中国国际贸易促进委员会专利商标事务所 11038
代理人 邹姗姗
(51)Int.Cl.
G06F 12/12(2006.01)

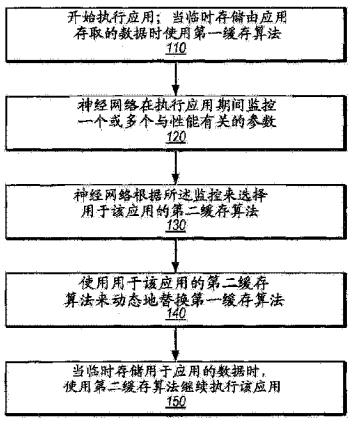
权利要求书2页 说明书14页 附图7页

(54)发明名称

用于使用神经网络来进行有效的缓存的系统和方法

(57)摘要

用于使用神经网络选择在临时存储由执行的应用存取的数据时将使用的适合的缓存算法的系统和方法可以动态地和/或迭代地替换用于该应用的初始缓存算法。神经网络的输入层可以聚集与性能有关的参数的值，例如，缓存命中率、数据吞吐率或存储器存取请求响应时间。神经网络可以检测模式或存取的模式的改变或者工作负荷、硬件组件或者操作系统参数的改变。根据这些和/或其它输入，神经网络可以选择和应用很可能改善应用的性能的缓存算法。神经网络的其它输入可以包括硬件配置参数和/或操作系统参数的值。神经网络可以例如使用增强学习来执行训练练习或者可以进行自我训练。



CN 103282891 A

1. 一种方法，包括：

由计算机实施以下操作：

开始执行所述计算机上的应用，其中，所述执行包括根据第一缓存算法在缓存中临时存储由所述应用存取的数据；以及

在所述执行期间：

神经网络监控一个或多个与性能有关的参数的值；

所述神经网络选择当临时存储由所述应用存取的数据时将使用的第二缓存算法，其中，所述选择至少部分地取决于所述监控的结果；

针对所述应用，用所述第二缓存算法动态地替换所述第一缓存算法；以及

继续执行所述应用，其中，所述继续执行包括根据所述第二缓存算法在缓存中临时存储由所述应用存取的数据。

2. 根据权利要求 1 所述的方法，其中，所述与性能有关的参数包括以下各项中的一项或多项：缓存命中率、数据吞吐率或者存储器存取请求响应时间。

3. 根据权利要求 1 或 2 所述的方法，其中，所述第一缓存算法和所述第二缓存算法中的每一个包括以下各项中的一个不同项：Memcached 算法、MemcacheDB 算法、最近最少使用算法、最近最多使用算法、最不频繁使用算法、以频率为基础的替换算法、至少倒数第 k 参考算法或者最近最不频繁使用算法。

4. 根据前述任意一项权利要求所述的方法，其中，所述第一缓存算法和所述第二缓存算法中的每一个联合以下各项中的一个不同项进行操作：组相关联缓存、全相关联缓存或者直接映射缓存。

5. 根据前述任意一项权利要求所述的方法，其中，所述监控包括确定所述一个或多个性能参数中的一个或多个的值是否满足或超过预定的阈值。

6. 根据前述任意一项权利要求所述的方法，其中，所述监控包括检测所述应用进行的存储器存取的模式。

7. 根据前述任意一项权利要求所述的方法，其中，所述监控包括检测所述应用的工作负荷的改变、所述应用进行的存储器存取的模式的改变、所述应用进行的存储器存取的频率的改变、所述计算机的硬件组件的改变或者所述计算机的操作系统的参数的改变。

8. 根据前述任意一项权利要求所述的方法，其中，所述选择包括根据所述一个或多个与性能有关的参数的所述值的加权，来选择缓存算法。

9. 根据前述任意一项权利要求所述的方法，还包括：

在所述替换之后：

所述神经网络监控所述应用的性能；以及

响应于确定所述替换未导致改进的性能，所述神经网络选择在临时存储由所述应用存取的数据时将使用的第三缓存算法。

10. 根据前述任意一项权利要求所述的方法，还包括：

在所述开始执行之前，所述神经网络选择所述第一缓存算法，其中，所述选择取决于将由所述应用进行的存储器存取的预测模式、硬件或操作系统参数的预测值或者与性能有关的参数的预测值。

11. 一种计算机程序，包括：当在一个或多个计算机上被执行时使所述一个或多个计

算机执行前述任意一项权利要求所述的方法的程序指令。

12. 一种计算机可读存储介质，其存储权利要求 11 所述的计算机程序。

13. 一种系统，包括：

一个或多个处理器，其用于执行应用；以及

神经网络；

其中，在执行所述应用期间，所述神经网络被配置为：

监控一个或多个与性能有关的参数的值；

选择当临时存储由所述应用存取的数据时将使用的缓存算法，其中，所述选择至少部分地取决于所述监控的结果；

使所选择的缓存算法在临时存储由所述应用存取的数据时被应用。

14. 根据权利要求 13 所述的系统，其中，所述与性能有关的参数包括以下各项中的一项或多项：缓存命中率、数据吞吐率或者存储器存取请求响应时间。

15. 根据权利要求 13 或 14 所述的系统，其中，所述缓存算法包括以下各项中的一个不同项：Memcached 算法、MemcacheDB 算法、最近最少使用算法、最近最多使用算法、最不频繁使用算法、以频率为基础的替换算法、至少倒数第 k 参考算法或者最近最不频繁使用算法。

16. 根据权利要求 13 至 15 中的任意一项所述的系统，其中，所选择的缓存算法联合以下各项中的一个不同项进行操作：组相关联缓存、全相关联缓存或者直接映射缓存。

17. 根据权利要求 13 至 16 中的任意一项所述的系统，其中，所述监控包括确定所述一个或多个性能参数中的一个或多个的值是否满足或超过预定的阈值。

18. 根据权利要求 13 至 17 中的任意一项所述的系统，其中，所述监控包括检测由所述应用进行的存储器存取的模式。

19. 根据权利要求 13 至 18 中的任意一项所述的系统，其中，所述监控包括检测所述应用进行的存储器存取的模式、所述应用的工作负荷的改变、所述应用进行的存储器存取的模式的改变、所述应用进行的存储器存取的频率的改变、所述计算机的硬件组件的改变或者所述计算机的操作系统的参数的改变。

20. 根据权利要求 13 至 19 中的任意一项所述的系统，其中，所述选择包括根据所述一个或多个与性能有关的参数的所述值的加权，来选择缓存算法。

21. 根据权利要求 13 至 20 中的任意一项所述的系统，其中，响应于确定所选择的缓存算法未导致改进的性能，所述神经网络被配置为选择在临时存储由所述应用存取的数据时将使用的另一缓存算法。

22. 根据权利要求 13 至 21 中的任意一项所述的系统，其中，所述神经网络被配置为选择初始缓存算法，所述选择取决于将由所述应用进行的存储器存取的预测模式、硬件或操作系统参数的预测值和 / 或与性能有关的参数的预测值。

用于使用神经网络来进行有效的缓存的系统和方法

背景技术

[0001] 缓存是临时存储区域，在该临时存储区域中，可以存储已经由应用存取或计算的数据（或者数据的副本）以在下一次存取时进行快速恢复。例如，当从计算机系统的主存储器存取数据时，可以在缓存中存储数据的副本以防再次需要该数据。如果再次需要该数据，则可以从缓存恢复存储在缓存中的副本（这通常具有更短的存取时间），而不是从主存储器重新获取该数据（这通常具有更长的存取时间）。类似地，可以在缓存中存储应用计算的数据的副本，然后如果再次需要该数据，则从缓存中恢复该数据的副本，而不是必须再次计算该数据或者从在该数据被计算出以后所存储的主存储器中获取该数据。因此，在相同的数据被频繁地或重复地存取的应用中或者在计算出的数据被接着恢复的应用中，缓存该数据可以减少用于恢复该数据的存储器存取的延迟（即，响应时间），从而改善应用的性能。

[0002] 当缓存已满时，可以根据系统中支持的策略（或多个策略）来使用各种替换策略（也称作缓存算法）来确定移除哪些数据以为要缓存的新数据留出空间。与其它类型的应用或使用模式相比，一些替换策略更适合于某些类型的应用或者使用模式。

[0003] 人工神经网络（通常简称为“神经网络”）可以用于对输入与输出之间的复杂关系进行建模（例如，建模为非线性统计模型），并且可以仿效。它们通常是使用并行架构来实现的。因此，它们呈现出快速的响应时间，并且对于一些类型的应用，可以非常适合于在实时系统中使用。

发明内容

[0004] 本文所描述的系统和方法可以使用神经网络来选择当临时存储由执行的应用存取的数据时将使用的适合的缓存算法。在一些实施方式中，响应于确定该缓存性能和 / 或总应用性能是不可接受的或者已经恶化，该神经网络可以动态地和 / 或迭代地替换用于该应用的初始缓存算法。该神经网络可以包括输入层、隐藏层（其可以处理由输入层提供的输入）和输出层。

[0005] 在一些实施方式中，神经网络的输入层可以聚集与性能有关的参数的值，例如，缓存命中率、数据吞吐率或者存储器存取请求响应时间。神经网络的其它输入可以包括各种硬件配置参数和 / 或操作系统参数的值。神经网络可以检测由应用进行的存储器存取的模式或者由应用进行的存取的模式的改变。在一些实施方式中，神经网络可以检测应用的工作负荷的改变、由应用对一个或多个地址进行的存储器存取的频率的改变、计算机的硬件组件的改变或者计算机的操作系统的参数的改变。在一些实施方式中，神经网络可以确定一个或多个性能参数中的一个或多个的值满足或超过预定的阈值，这可以触发神经网络对缓存算法的分析和 / 或选择。

[0006] 根据由输入层提供的输入以及应用和执行环境的当前状态，神经网络的隐藏层可以选择很可能改善应用的性能的缓存算法。由输出层输出的数据值或信号可以使得在应用正在执行的同时动态地应用所选择的算法（即，替换初始缓存算法）。在一些实施方式中，神经网络可以根据一个或多个与性能有关的参数和 / 或隐藏层的其它输入的值的加权来选

择缓存算法。

[0007] 在一些实施方式中，使用由神经网络选择的缓存算法来替换初始缓存算法可以包括改变缓存根据其进行操作的替换策略。例如，初始缓存算法和所选择的缓存算法可以是以下各项中的不同项：Memcached 算法、MemcacheDB 算法、最近最少使用算法、最近最多使用算法、最不频繁使用算法、以频率为基础的替换算法、至少倒数第 k 参考算法或者最近最不频繁使用算法。在一些实施方式中，使用由神经网络选择的缓存算法来替换初始缓存算法可以包括改变当前的缓存算法的一个或多个参数的值、硬件配置(例如，改变缓存的大小)或者改变操作系统的参数的值。在一些实施方式中，使用由神经网络选择的缓存算法来替换初始的缓存算法可以包括改变缓存的类型，例如，如果初始缓存算法或者所选择的缓存算法联合以下各项中的不同项操作的话：组相关联缓存、全相关联缓存或者直接映射缓存。

[0008] 在一些实施方式中，如果通过用神经网络选择的缓存算法替换初始缓存算法未改善性能，则神经网络可以迭代地选择和应用一个或多个其它缓存算法以改善性能。在一些实施方式中，初始缓存算法还可以由神经网络来选择。例如，选择初始缓存算法可以取决于由应用进行的存储器存取的预测模式、硬件或操作系统参数的预测值或者与性能有关的参数的预测值(例如，基于先前执行该应用或类似的应用)。在各个实施方式中，神经网络可以执行训练锻炼以确定神经网络的一个或多个传递函数，或者可以是自我训练的(例如，使用增强学习)。

附图说明

[0009] 图 1 是示出了如本文所描述的使用神经网络来选择缓存算法的方法的一个实施方式的流程图。

[0010] 图 2 是示出了根据一个实施方式的神经网络的操作的数据流程图。

[0011] 图 3 是示出了根据一个实施方式的包括多个级处的神经元的层级的神经网络的操作的数据流程图。

[0012] 图 4 是示出了用于使用神经网络检测与性能有关的输入中的模式的方法的一个实施方式的流程图。

[0013] 图 5 是示出了用于迭代地并且动态地选择用于应用的缓存算法的方法的一个实施方式的流程图。

[0014] 图 6 是示出了用于在神经网络中应用增强学习的方法的一个实施方式的流程图。

[0015] 图 7 是示出了如本文所描述的被配置为使用神经网络来选择缓存算法的计算机系统的框图。

[0016] 虽然通过针对多个实施方式和所示出的附图以举例说明的方式在本文中描述了各个实施方式，但是本领域技术人员将认识到，实施方式不限于所描述的实施方式或附图。应当理解的是，附图和对其的详细描述并不旨在将实施方式限制于所公开的特定形式，相反，旨在覆盖落入本发明的精神和范围内的所有修改、等同形式和替换物。本文使用的任何标题仅用于组织的目的，而不意味着用于限制描述的范围。如贯穿本申请所使用的，词语“可以”是在许可的意义上使用的(即，意味着有可能)，而不是在强制的意义上使用的(即，意味着必须)。类似地，词语“包括”、“包含”和“含有”意味着包括但不限于。

具体实施方式

[0017] 如上所述，缓存是存储区域，在该存储区域中，可以临时存储由应用存取和 / 或计算的数据，并且当接下来存取所存储的数据时，可以从该存储区域快速地恢复所存储的数据。例如，一些处理器包括一个或多个缓存（例如，L1 和 L2 缓存），并且这些小型快速的存储器可以存储从较大的更慢的主存储器存取的数据和 / 或指令的副本以减小下次需要对它们进行存取时的响应时间。在另一个实施例中，单独的或者分布式的计算机系统可以包括一个或多个小型快速的存储器，这些存储器用于存储较大的更慢的存储器中（例如，系统存储器中、本地或远程磁盘存储设备或者分布式存储系统中）存储的程序代码、数据或者其它数字资源（例如，网页、文档、多媒体文件或其它对象）的副本。当在缓存中找到请求的资源（例如，数据对象、网页、文档等）时，这称作“缓存命中”。当在缓存中未找到请求的资源但是必须从主存储器或者远程存储设备获取请求的资源时，这称作“缓存未命中”。

[0018] 如上所述，当缓存已满时，可以应用缓存算法以确定移除哪些数据以为要缓存的新的数据留出空间。例如，系统可以支持一个或多个缓存算法（也称作缓存替换算法或者缓存替换策略），其包括但不限于 ：分布式缓存、MemcacheDB、最近最少使用（LRU）、最近最多使用（MRU）、最不频繁使用（LFU）、以频率为基础的替换（FBR）、至少倒数第 k 参考（LRU-k，例如，LRU-2）或者最近最不频繁使用（LFRU）替换算法。然而，如下面更详细描述的，与一些类型的应用或使用模式相比，这些替换策略中的各个替换策略可以更适合于其它类型的应用或者使用模式。当在系统使用的缓存算法非常适合于在特定的系统配置和执行环境中执行的特定的应用时，缓存可以大大地改善该应用的性能。

[0019] 如上所述，神经网络可以用于对输入与输出之间的复杂关系进行建模（例如，建模为非线性统计模型），并且可以仿效。因此，使用神经网络可以允许以灵活且强有力的方式来执行某些类型的任务。在一些实施方式中，本文所描述的系统和方法可以用于将神经网络应用于设计高效的缓存的任务，该缓存可以称作“神经缓存”。例如，在一些实施方式中，神经网络可以用于在系统中对与性能有关的输入与输出之间的关系进行建模，以在运行时迭代地并且动态地选择适合于给定的应用、资源请求和 / 或执行环境（例如，硬件和 / 或操作系统配置）的缓存算法。在一些实施方式中，本文所描述的用于将神经网络应用于缓存的设计的方法也可以是高度可扩展的。

[0020] 在一些实施方式中，神经网络可以用于设计、修改和 / 或重新配置神经缓存。在一些实施方式中，神经网络可以用于准确地确定用于针对神经缓存的给定的数据请求的最佳缓存算法，这取决于诸如以下各项的因素 ：做出请求的应用的使用模式、缓存大小、系统中的基础硬件、操作系统的参数和 / 或状态等（即，导致应用在其执行环境中的相应最佳的缓存命中率的缓存算法）。神经网络自动地（即，没有人工干预）适应应用工作负荷（或使用模式）的改变或者应用在其中执行的硬件、软件和 / 或操作系统环境中的改变的能力在实时应用中会非常有用，其中，这些改变在一些情况下可能是频繁的和 / 或不可预测的。因为神经网络非常快速并且通常可以并行地执行其计算中的大部分计算，因此这些神经网络可以快速地检测这些改变并对这些改变做出反应，并且可以动态地选择适合于新检测到的状况的缓存算法。因此，使用被设计为当状况在运行时期间改变时迭代地且动态地应用最佳的缓存算法的神经缓存可以导致更好的应用性能。

[0021] 在一些实施方式中，神经网络可以包括通过“突触”网络连接的不同类型的“神经元“。例如，输入层可以包括一个或多个输入神经元，隐藏层可以包括用作处理单元的一个或多个神经元，并且输出层可以包括一个或多个输出神经元。在一些实施方式中，突触可以携带与每一个神经元的输入的加权有关的信息，而在其它实施方式中，可以通过隐藏层中的处理单元来应用加权。在一些实施方式中，神经网络可以被配置为收集反映缓存性能和/或用于执行的应用的总应用性能的输入数据（例如，缓存命中率、数据吞吐量和/或存储器存取响应时间），以确定当前的缓存算法在应用的当前执行环境中针对该应用是否有效。例如，神经网络可以被配置为检查一个或多个CPU状态寄存器的值，该CPU状态寄存器的值反映与性能有关的信息（例如，缓存命中），以探查一个或多个总线上的流量，或者使用其它方式来聚集与性能有关的信息。如果应用的性能是不可接受的或者被确定为随着时间是下降的/退化的（例如，如果缓存命中率在期望的范围外），则这可以指示当前的缓存算法不是在考虑到当前状态和/或状况的情况下适合于应用的最佳性能的缓存算法。换言之，这可以指示当前的缓存算法正在替换还不应当替换的项目，或者正在维持缓存中的应当被替换的项目。在该实施例中，神经网络可以被配置为采取行动以通过选择和应用更适合于特定的情况的缓存算法并且当该情况再次改变时/如果该情况再次改变则迭代地对该缓存算法进行调整，来改善缓存命中率（因此，应用的总性能）。

[0022] 如上所述，神经网络可以被配置为在针对给定的应用的运行时间期间动态地选择和应用特定的缓存算法（例如，通过替换先前使用的缓存算法）。换言之，选择适合的缓存算法可以不完全基于在执行应用之前执行的静态分析，而是可以在运行时进行选择、应用和/或修改。在各个实施方式中，除了总缓存命中率、数据吞吐率和/或存储器存取响应时间以外，可以由输入层聚集并且被提供给隐藏层的输入还可以包括以下各项中的一项或多项：标识请求的资源的数据（例如，在其中存储请求的数据、网页、文档或其它资源的存储器位置的地址）、指示资源的类型的数据、指示存取请求的类型的数据（例如，存储数据的请求、恢复数据的请求、修改所存储的数据的请求）、指示针对给定的资源或资源组而做出的请求的数量和/或频率的数据、针对给定的资源或资源组的缓存命中率或者其它与性能有关的数据。此外，输入层可以向隐藏层提供指示各个硬件配置和/或操作系统参数的值的数据。例如，输入层可以提供指示向系统中的一个或多个缓存分配的内存的量（即，缓存大小）的数据。

[0023] 在一些实施方式中，神经网络可以被配置为通过分析由输入层提供的输入数据中的一些或全部来检测使用模式（和/或使用模式的改变），并且这些使用模式（或对其的改变）可以是选择适合的缓存算法的主要因素，如下面更详细描述的。例如，分许在系统中进行的存取的分布（例如，多次存取几个项目对比少量存取很多不同的项目，或者随着时间的推移或在各种不同的地址上对相同的或不同的项目进行的存取的分布）可以提供用于选择用于应用的缓存算法的重点考虑的输入。在一些实施方式中并且针对某些类型的应用，与硬件或操作系统参数相比，（例如，如应用进行的存取的总工作负荷和/或这些存取的分布中反映的）使用模式更可能在执行应用期间改变，并且可能由于该原因而被重点考虑。然而，在一些实施方式中，硬件或操作系统参数值的改变可以触发神经网络对当前输入的分析，这可能或可能不会导致缓存算法的改变。

[0024] 在一些实施方式中，检测特定的输入值可以触发神经网络选择（或者确定是否选

择）替换缓存算法来临时存储由给定的执行应用存取的数据。例如，如果与性能有关的参数中的一个的值满足或超出预定的阈值（例如，缓存命中率低于给定的可接受水平），则神经网络可以被配置为分析当前的一组输入并且输出关于适合于当前的应用和执行环境的缓存算法的指示。在一些情况下，结果可能是响应于触发的缓存算法的改变。在其它情况下，即使特定的输入值触发神经网络进行分析，输出也可能指示缓存算法不应当在此时改变（例如，如果没有可用的替换缓存算法可能改善性能的话）。在一些实施方式中，神经网络可以对输入层提供的输入值执行定期采样，并且针对一些输入组合的分析的结果可以导致用于当前执行的应用的缓存算法的改变。在使用定期采样的实施方式中，采样率可以是固定的，可以适合于在系统配置或应用工作负荷中观测的改变频率，或者可以针对训练阶段和生产阶段是不同的，或者可以针对初始执行阶段和应用的性能在其中呈现为是稳定的执行阶段是不同的。

[0025] 注意，在一些实施方式中并且针对多组输入，神经网络可以输出缓存算法的选择，该缓存算法使用不同的基础来确定哪些缓存的项目应当被替换以为在缓存中存储新的项目留出空间。例如，神经网络进行的分析可以指示可以通过使用对最近使用的条目或最不经常使用的条目进行替换的缓存算法替换对最近最少使用的条目进行替换的缓存算法来改善缓存性能或者总应用性能，并且神经网络的输出可以是发起这种改变的值或信号。

[0026] 在一些实施方式中，神经网络进行的分析的结果可以是在分析以后使用相同的缓存算法，但是系统或缓存算法本身的一个或多个配置参数的值被改变。例如，虽然增加的缓存大小可以改善性能，但是缓存设计者可能必须在潜在的性能改善和针对缓存所分配的快速存取内存（这可能更昂贵并且可能占用有价值的附近的不动产）的量与其它存储功能之间进行折中。在一些实施方式中，由神经网络执行的分析可以确定具有某些使用模式的应用能够使用更小的缓存来实现可接受的性能。因此，在不同的实施例中，神经网络的输出可以值或信号，所述值或信号使系统向应用动态地分配更多的资源（例如，增加可用的缓存大小）以增加缓存性能，或者，如果使用更小的可用的缓存来针对该应用维持可接受的缓存性能的话，则使系统向应用动态地分配更少的资源（例如，通过向另一个应用重新分配资源中的一些资源）。在另一个实施例中，神经网络的输出可以是使系统动态地修改阈值的值或信号，该阈值确定有多少条目被缓存算法替换和／或何时对这些条目进行替换。在其它实施方式中，神经网络进行的分析的结果可以是缓存类型本身被改变。例如，分析可以指示可以通过使用组相关联缓存或者全相关联缓存而不是直接映射缓存来改善缓存性能或总应用性能，并且神经网络的输出可以是发起系统中的这种改变的值或信号。

[0027] 如上所述，在一些实施方式中，神经网络可以被配置为检测应用的工作负荷的改变（例如，应用完成的工作的类型的改变、应用向相同的或不同的地址或项目进行的存取的数量、频率和／或分布、应用对相同的或不同的地址或项目进行的存取之间的时间、应用存取的不同地址或项目的数量、应用进行存取的位置等），并且可以使用当前的缓存算法对缓存性能和／或应用的总性能的这种改变的效果进行统计分析。因此，根据应用于该分析的输入中的每一个输入的固定的（例如，预先确定的）或自适应的加权，神经网络可以选择可以改善性能的缓存算法，或者可以确定当前的缓存算法适合于当前的应用、状况和执行环境。

[0028] 在一些实施方式中，选择初始缓存算法可以基于针对系统或者针对给定的应用或

应用类型的默认选择。在其它实施方式中，可以由神经网络根据已知的或预测的存取模式、已知的和 / 或预测的硬件或操作系统参数值和 / 或已知的或预测的与性能有关的参数值来对用于临时存储由给定的应用存取的数据的初始缓存算法进行选择。例如，在一些实施方式中，用于先前执行给定的应用或类似的应用（即，已经向神经网络提供类似的输入的应用）的历史缓存命中率可以被输入到神经网络以选择初始缓存算法。在另一个实施例中，神经网络可以被配置为通过在针对代表给定的应用提供的输入的给定的统计范围内的其它两组或更多组输入生成的输出之间进行插值，来确定输出。在其它实施方式中，来自训练阶段（例如，如下面所述的训练阶段）的输入和 / 或输出可以由神经网络分析以选择初始缓存算法。

[0029] 注意，在一些实施方式中，在系统中支持的所有可用的缓存算法是固定的和 / 或已知的。在其它实施方式中，额外的缓存算法或者对先前已知的缓存算法的修改版本可以由神经网络生成（例如，通过改变与缓存算法或者基本硬件或软件相关联的一个或多个参数值）或者由缓存设计者随着时间来添加。类似地，在一些实施方式中，可以由神经网络或者缓存设计者（例如，响应于考虑到当前的硬件配置和 / 或操作系统参数，确定不可能导致可接受的性能）移除或禁用可用的缓存算法中的一个或多个。用于选择适合的缓存算法的神经网络可以适应可用的缓存算法选择中的这些改变。

[0030] 图 1 示出了用于使用神经网络来确定适合的缓存算法的方法的一个实施方式。如在 110 处所示，在该实施例中，该方法可以包括开始在包括神经网络的系统中执行应用。当执行开始时，该方法可以包括当临时存储（即，缓存）应用存取的数据时使用该系统中支持的多个缓存算法中的第一缓存算法。例如，该系统可以支持一个或多个缓存算法（也称作缓存替换算法或缓存替换策略），其包括但不限于 ：分布式缓存、MemcacheDB、最近最少使用（LRU）、最近最多使用（MRU）、最不频繁使用（LFU）、以频率为基础的替换（FBR）、至少倒数第 k 参考（LRU-k，例如，LRU-2）或者最近最不频繁使用（LFRU）替换算法。在多个实施方式中，在执行应用期间应用的第一缓存算法可以被选择为默认缓存算法、最近应用的缓存算法，或者基于特定的应用或类似的应用的历史或预测的工作负荷来进行选择，如下面更详细描述的。

[0031] 如图 1 中的 120 处所示，该方法可以包括神经网络在执行应用期间监控一个或多个与性能有关的参数。例如，神经网络的输入层可以聚集和 / 或接收指示以下各项的输入 ：应用进行的数据存取、应用进行的存取的速率、缓存命中率、数据吞吐率、存取响应时间（例如，响应于一个或多个存储器存取请求而经历的平均、当前或最大存储器存取时间）、各种硬件参数值、各种操作系统参数和 / 或可以反映和 / 或影响系统中的应用的性能的其它信息。如 130 处所示，该方法可以包括神经网络根据监控的结果来选择用于该应用的第二缓存算法。例如，如果所监控的参数中的任意一个参数指示不可接受的性能水平、性能的退化或者系统的可能影响应用的性能的硬件或软件参数的改变，则神经网络的隐藏层可以被配置为基于当前状态和 / 或所监控的参数中的一个或多个的值来确定更适合于该应用的缓存算法。

[0032] 如在 140 中，一旦选择了第二缓存算法，就可以使用针对该应用的第二缓存算法来动态地替换第一缓存算法（即，在其执行期间），并且如在 150 中，当临时存储用于该应用的数据时，该系统可以使用第二缓存算法继续执行该应用。

[0033] 图 2 示出了根据一个实施方式在确定用于执行应用的缓存算法时所使用的神经网络的操作。在该实施例中，神经网络的输入层监控以下输入、聚集以下输入和 / 或以其它方式向神经网络的隐藏层提供以下输入 ：硬件参数值 212、操作系统参数 213、与数据存取有关的信息 214、数据存取速率 215、缓存命中率数据 216 和数据吞吐率 217。在其它实施方式中，与图 2 中所示的与性能有关的参数值相比，神经网络的输入层可以监控、捕获和 / 或提供更多、更少或者不同的与性能有关的参数值。例如，在一些实施方式中，输入层可以捕获与以下各项有关的数据 ：当在缓存中找到所请求的项目时针对存储器存取的当前或平均存取响应时间、当在缓存中未找到所请求的项目时针对存取的当前或平均存取响应时间、针对特定的缓存项目的存取的当前或平均存取响应时间或者针对该应用请求的所有存储器存取的当前或平均存取响应时间。

[0034] 如元件 220 所示，在该实施例中，神经网络的隐藏层（有时称作“处理”层）可以检测所接收的输入的模式和 / 或这些输入的值的改变。例如，神经网络的隐藏层可以包括神经元或处理单元的集合，其中的每一个可以被配置为分析作为确定神经网络的输出的一部分而被提供给神经网络的输入数据的一部分。由神经元中的每一个应用的传递函数可以基于输入（并且在一些情况下，系统或正在执行的应用的当前状态）生成一个或多个输出，或者可以贡献于神经网络的一个或多个输出的最终值。由神经网络的隐藏层应用的传递函数可以包括一个或多个输入的值与相应的阈值的比较、一个或多个输入的值与针对其已经确定了最佳的缓存算法（例如，在训练期间或者通过系统中的先前使用）的各个输入值的组合的比较、线性或 S 型函数、插值函数或者用于确定最接近匹配当前的输入集的先前所观测的输入集的函数。在一些实施方式中，可以根据固定的（例如，预先确定的）加权或者使用已经被自动或手动调整以改善神经网络本身的性能的加权来对神经网络的输入进行加权。

[0035] 如在 230 中所示，由神经网络的输出层所输出的值可以引起基于输入的加权值、检测的模式和 / 或检测到的输入或模式的改变来选择缓存算法（和 / 或其参数）。在一些实施方式中，可能仅存在神经网络的一个输出，并且其值可以用于基于神经网络的输入来选择用于当前执行的应用的适合的缓存算法。在其它实施方式中，可能存在神经网络的多个输出，其中的一个或多个输出可以用于选择适合的缓存算法，而其中的其它输出可以用于选择和 / 或修改所选择的缓存算法和 / 或系统配置的一个或多个参数（例如，缓存大小、缓存类型或其它配置参数）的值。注意，在一些实施方式中，神经网络可以包括输入层（或其输入组件）、神经元（处理单元）和 / 或输出层的层级。在图 3 中示出并且在下面描述了包括处理单元的层级的神经网络的一个实施方式。

[0036] 在各个实施方式中，可以使用硬件元件、软件模块或使用硬件和软件组件的组合来实现神经网络。例如，在一些实施方式中，可以至少部分地使用被配置为收集与性能有关的数据并且将该数据提供给隐藏层的电路来实现神经网络的输入层。在其它实施方式中，输入层可以包括一个或多个软件模块，所述一个或多个软件模块被配置为聚集（例如，读取或者以其它方式确定）由硬件组件（例如，性能计数器或者与性能有关的状态寄存器、捕获总线流量的探测电路、操作系统参数寄存器或者硬件配置状态指示器）存储或生成的值，并且将这些值提供给神经网络的隐藏层。例如，被配置为选择用于应用的适合的缓存算法的神经网络可以被实现为计算机程序产品或者软件，该计算机程序产品或者软件可以包括其上存储有指令的非临时性计算机可读存储介质，这些指令可以用于对计算机系统（或者其

它电子设备）进行编程以执行本文所描述的技术。

[0037] 在一些实施方式中，可以至少部分地使用硬件处理单元的集合来实现隐藏层，其中所述硬件处理单元例如是一个或多个状态机或者其它逻辑 / 电路，其被配置为检测由输入层提供的输入的模式或模式的改变，并且提供使得在系统中采取各个动作的一个或多个输出，所述动作例如是选择缓存算法或其参数值，或者使用一个缓存算法替换另一个缓存算法（或者其参数），如本文所描述的。在其它实施方式中，可以通过一个或多个软件模块来至少部分地实现神经网络的隐藏层，其中所述一个或多个软件模块被配置为分析输入数据并且提供使得在系统中采取各个动作的一个或多个输出值，所述动作例如是选择缓存算法或者其参数值，或者使用一个缓存算法替换另一个缓存算法（或者其参数），如本文所描述的。在一些实施方式中，实现神经网络的硬件组件和 / 或软件模块可以被配置为同时操作，并且因此可以对输入信息执行有效的并行处理，并且提供对要在系统中采取的适当输出 / 动作的快速确定。注意，在一些实施方式中，由神经网络的隐藏层执行的分析所造成的有效动作中的一个是不改变缓存算法和系统配置（例如，如果确定性能不可能通过改变而改善的话）。

[0038] 图 3 示出了根据一个实施方式的包括神经元（处理单元）的层级并且用于确定用于执行应用的缓存算法的神经网络的操作。在该实施例中，神经网络的输入层 310 监控各种输入 315、捕获各种输入 315 和 / 或以其它方式向神经网络的隐藏层提供各种输入 315。如本文所描述的，这些输入可以包括所有以下各项中的任意一项：硬件参数值、操作系统参数、与数据存取有关的信息、数据存取速率、缓存命中率数据、数据吞吐率、存取响应时间和 / 或其它与性能有关的参数值。

[0039] 如该实施例中所示，神经网络的隐藏层可以包括多级神经元或者处理单元，例如，包括神经元 320a-320d 的第一级以及包括一个或多个更高级的神经元 330 的第二级。在该实施例中，神经元 320 中的每一个可以被配置为分析由输入层向隐藏层提供的输入 315 的不同子集。例如，一个神经元可以被配置为测试和 / 或检测给定的输入值是否满足或超过针对相应的参数的预定阈值；另一个神经元可以被配置为检测缓存命中率、吞吐率、存取响应时间或者其它与性能有关的数据速率的改变；另一个神经元可以被配置为将输入集合的值与针对其已经确定了最佳的算法（例如，在训练阶段或者通过系统中的先前使用期间）的输入集合的先前观测的值进行比较；而又一神经元可以被配置为检测一个或多个输入的值（例如，随着时间的推移的一系列或一连串的值）的模式。如图 3 中所示，由输入层提供的一些输入 315 可以用作神经元 320 中的多于一个神经元的输入，而其它输入可以被提供给单个神经元 320。如上所述，可以根据固定的（例如，预定的）加权或者使用可以被自动或手动调整以改善神经网络本身的性能的加权来对神经网络的输入进行加权。此外，应当注意的是，在一些实施方式中，可以将不同的加权应用于由输入层提供的输入中的每一个和 / 或用作隐藏层中的更高级的神经元的输入的任何中间输出（例如，用作图 3 中的神经元 330 的输入的神经元 320a-320d 的输出）。

[0040] 如在先前的实施例中一样，如在 340 中所示，由神经网络的输出层输出的值可以引起基于多级神经元的输入的加权值、检测的模式和 / 或检测到的输入或模式的改变来选择缓存算法（和 / 或其参数）。如在先前的实施例中一样，可以使用硬件组件和 / 或软件模块的任意组合来实现图 3 中所示的多级神经网络，所述硬件组件和 / 或软件模块的任意组

合被配置为执行对与性能有关的输入数据的收集、对输入数据的分析以及使得系统采取动作的一个或多个值的输出，其中所述动作涉及选择针对当前的硬件配置、操作系统配置、应用状态、工作负荷等的适合的缓存算法。还应当注意，虽然图 3 示出了神经网络包括两级神经元层级的神经网络，但是在其它实施方式中，神经网络可以在输入层、隐藏层和 / 或输出层处包括任意数量的级。

[0041] 图 4 示出了用于通过使用神经网络来检测模式以选择适合的缓存算法的方法的一个实施方式。如该实施例中所示，如在 420 中，该方法可以包括在执行应用期间神经网络的输入层监控与性能有关的参数值，例如，本文所描述的那些参数值，并且将这些参数值提供给神经网络的隐藏层。如 430 处所示，该方法可以包括隐藏层确定当前的输入的值是否匹配先前观测的输入值的组合，如在 430 中。如果匹配（如 430 中的肯定退出），则该方法可以包括神经网络基于由神经网络针对该输入值的组合所输出的先前的值来选择（或者引起选择）用于应用的缓存算法，如在 450 中。例如，神经网络可以避免选择先前被确定为对于该输入值的组合是次优的缓存算法（即，未改善性能或者导致退化的性能的缓存算法）或者选择先前被确定为适合于该输入值的组合的缓存算法（例如，改善性能或者在训练期间或者由于神经网络在当前或先前执行应用期间针对该输入值的组合的先前分析导致处于可接受的范围内的性能的缓存算法）。

[0042] 如果隐藏层确定当前的输入值组合不匹配先前观测的输入值的组合（如 430 的否定退出所示），则该方法可以包括隐藏层确定当前的输入是否指示针对应用进行的存取的存取模式或使用模式，如在 440 中。例如，隐藏层可以被配置为检测到存取是针对具有重复的或交替的地址模式的项目或者所存取的项目的地址遵循数学序列或者串，例如，递增地址，或者遵循另一个存取模式。在一些实施方式中，隐藏层可以被配置为检测存取的分布的模式（例如，几个项目被存取很多次，或者很多项目中的每一个仅被访问几次）。在其它实施方式中，隐藏层可以被配置为检测存取频率的模式或者针对相同的或不同的项目的存取之间的时间。在该实施例中，如果检测到这些或其它类型的模式中的任意一种（如 440 的肯定退出所示），则该方法可以包括神经网络基于所检测的模式来选择（或者引起选择）用于该应用的缓存算法，如在 460 中。例如，如果确定几个项目被存取很多次而其他项目很少被存取，则神经网络可以选择用于该应用的最不频繁使用（LFU）缓存算法。

[0043] 如果没有检测应用进行的存取或者存取的使用的模式（如 440 的否定退出所示），则该方法可以包括神经网络选择（或者引起选择）默认缓存算法或者基于针对最有关的输入组合生成的一个或多个输出或者通过对针对两个或更多个最有关的输入组合生成的输出进行插值为该应用选择的缓存算法，如在 470 中。

[0044] 注意，在不同的实施方式中，被配置为选择适合的缓存算法的神经网络可以通过不同的方式来“学习”。例如，一些神经网络可以通过监督学习方法来学习，在该监督学习方法中，神经网络具有输入训练集合和相应的输出。在这些实施方式中，神经网络可以被配置为通过数据集隐含的映射函数来“学习”将由隐藏层执行的传递函数。在其它实施方式中，神经网络可以通过非监督学习（即，通过自我训练）来学习。在这些实施方式中，神经网络可以具有数据和一个或多个将被最小化的成本函数（例如，缓存未命中或者平均响应时间），并且神经网络可以被配置为确定使这些成本最小化的传递函数。

[0045] 在一些实施方式中，诸如本文所描述的神经网络的神经网络可以在为各个应用选

择缓存算法时应用增强学习技术。增强学习是在其中要采取的动作不是预定的而是通过与系统的迭代交互(例如,通过尝试法)进行学习的机器学习技术。通过使用增强学习,可以确定目标,并且然后进一步精炼用于选择要采取的动作的策略,使得一个或多个长期成本测量最小化。例如,在一系列交互中的每一个交互中,该系统可以(例如,基于由神经网络的输入层聚集的与性能有关的参数值)确定当前的状态,并且可以基于当前状态来选择要采取的动作(例如,使用当前选择的缓存算法来继续执行,或者选择和应用不同的缓存算法)。然后,该系统可以(例如,通过测量系统在缓存命中率、数据吞吐量和/或存取响应时间方面的性能)测量动作的效果,并且将奖励值分配给针对该输入集采取的动作(其可以是正或负)。随着时间的推移,系统建立指定针对不同的输入组合要采取的动作的策略或策略集,并且在每当采取动作时测量奖励时,对策略进行精炼。策略可以定义针对每一个输入/输出集的预期的奖励,还可以随着时间的推移当采取更多的动作并且测量到实际的(观测的)奖励时对策略进行精炼。可以在神经网络中应用各种增强学习技术以允许神经网络学习要在不同的情形下应用的适合的缓存算法(例如,这取决于硬件或操作系统配置、工作负荷、缓存大小等)。

[0046] 可以应用的一种增强学习技术是增强学习的状态动作奖励状态动作(SARSA)方法。通过使用该方法,对于每一次迭代,系统的当前状态被(例如,神经网络的输入层)观测到,并且采取动作(例如,使用不同的缓存算法来替换当前的缓存算法)。所采取的动作可以是(例如,基于神经网络维持的当前输入/动作/奖励值)被估计为在当前的状态下要采取的最佳动作的动作、随机动作或者被认为针对当前状态是次优的动作。在一些实施方式中,选择随机的动作或者被认为是次优的动作可以允许神经网络随着时间的推移采用不同的选择并且找到最佳的解决方案。在采取动作以后,神经网络可以观测系统的任何改变以确定该动作是否改善了系统的性能(在该情况下,可以向输入/动作组合分配正奖励值)或者使性能退化(在该情况下,可以向输入/动作组合分配负奖励值)。可以在每次迭代以后更新由神经网络(例如,在数据结构中或者通过配置提供该功能的组合逻辑)维持的输入/动作/奖励值,以允许神经网络在考虑到其情形/执行环境的情况下改善其在选择用于该应用的适合的缓存算法方面的性能。基于更新的值和所观测到的任何状态改变,神经网络可以采取进一步动作以尝试识别最佳的缓存算法。

[0047] 图5示出了用于迭代地且动态地选择用于临时存储由应用存取的数据的各种缓存算法的方法的一个实施方式。如该实施例中的505处所示,在一些实施方式中,该方法可以包括执行训练锻炼以播种神经网络的知识库和/或初始化神经网络的一个或多个状态机。在其它实施方式中,可能不存在训练阶段。相反,神经网络可以使用增强学习或者其它自我训练技术(例如,本文所描述的那些自我训练技术)来自我训练,同时该神经网络被用于动态地选择一个或多个缓存算法以用于在生产(即,真实)计算环境中执行的应用。

[0048] 如510处所示,在一些实施方式中,该方法可以包括至少部分地基于预测的使用和当前的硬件和/或操作系统参数来选择将在临时存储用于应用的数据时使用的初始缓存算法。例如,在一些实施方式中,当在考虑到当前的硬件和/或软件参数值的情况下采用各种缓存算法时的应用的使用和/或性能可以基于训练锻炼的结果来预测,或者在类似的情况下基于系统的先前的性能来预测。在其它实施方式中,可以基于用于系统、用于当前的应用或者用于相同类型或类似类型的应用的默认缓存算法来选择初始缓存算法。

[0049] 如 520 处所示，该方法可以包括输入层监控和收集与性能有关的数据，其包括但不限于：与性能有关的参数和 / 或本文所描述的系统参数中的任意一个或全部的值。如果在执行应用的任何点处，输入的组合指示应用的性能是不可接受的或者已经恶化（如 530 的肯定退出所示），则该方法可以包括选择并且动态地应用不同的缓存算法以在临时存储用于该应用的数据时使用（如在 550 中），并且该选择可以基于应用的当前状态和当前的输入。例如，在一些实施方式中，可以通过检测一个或多个性能测量的阈值（例如，通过缓存命中率或数据吞吐率下降到可接受阈值以下）或者通过检测吞吐量或缓存命中率（例如，大于或等于预定的百分比改变的下降）的下降或者当前、平均或最大存取响应时间的增加来触发对新缓存算法的动态选择。

[0050] 类似地，如果在执行应用期间的任何点处，输入的组合指示硬件或软件配置参数的一个或多个值已经改变（例如，如果操作系统参数值已经改变或者系统中的内存分配已经改变），则该方法可以包括选择并且动态地应用不同的缓存算法以在临时存储用于该应用的数据时使用，并且该选择可以基于应用的当前状态和当前的输入。这被示出为从 540 和 550 的肯定退出的路径。应当再次注意的是，在一些情况下，神经网络可以将当前的缓存算法选择作为最佳的缓存算法（即，不导致缓存算法或者其任何参数发生改变）。

[0051] 如图 5 所示，直到与性能有关的问题（或者另一个与性能有关的问题）被检测到为止或者除非与性能有关的问题（或者另一个与性能有关的问题）被检测到，或者硬件或软件配置参数的改变（或者另一个硬件或软件配置参数改变）被检测到，该方法才可以包括输入层继续监控和收集与性能有关的参数值，同时该应用继续执行（例如，直到执行完成为止）。当满足任意条件时，该方法可以包括选择并且动态地应用用于该应用的另一种缓存算法。该迭代过程是通过从 540 和 550 到 520 的反馈回路在图 5 中示出的。

[0052] 如上所述，在一些实施方式中，神经网络的处理单元（神经元）中的一个或多个可以在确定用于给定的应用和 / 或执行环境的适合的缓存算法时应用增强学习技术。图 6 示出了一种用于在神经网络中应用增强学习以选择用于应用的缓存算法的方法的一个实施方式。例如，在一些实施方式中，该方法可以用于选择用于图 5 中所示的方法的迭代中的一个迭代的缓存算法。如该实施例中所示，该方法可以包括确定先前被估计为针对当前的状态和在执行应用期间收集的输入的最佳缓存算法的缓存算法，如在 600 中。在一些实施方式中，应用的当前状态（相对于其缓存性能或总性能）可以被反映在一个或多个观测的与性能有关的值（例如，缓存命中率或者数据吞吐率，其为“高”、“中等”或“低”或者其处于“范围之内”或“范围之外”）中。在一些实施方式中，缓存算法可能已经在先前基于针对相同的状态和输入组合所采取的一个或多个动作和响应于该动作所观测的相应成本或奖励而被估计为针对当前的状态和在训练阶段期间或者在自我训练技术的先前迭代期间的输入的最佳缓存算法。

[0053] 在确定先前被估计为针对当前的状态和输入（或者针对类似的状态和输入组合）的最佳缓存算法的缓存算法以后，该方法可以包括选择先前识别（估计）的最佳缓存算法或者先前被估计为针对当前的状态和输入（或者针对类似的状态和输入组合）为次优的缓存算法，如在 610 中。例如，在一些实施方式中，使用增强学习技术的神经网络可以被配置为偶尔选择先前还未针对当前的输入组合选择的缓存算法或者先前已经被确定为针对当前的输入组合为次优的缓存算法，以发现或精炼这种选择的效果的估计。

[0054] 如 620 处所示，该方法可以包括基于缓存算法的选择来采取行动，在该情况下，使用所选择的缓存算法来替换当前的缓存算法。如 630 处所示，在该实施例中，在一些实施方式中，该方法可以包括响应于缓存算法的改变，确定该应用的新状态。如图 6 所示，该方法还可以包括确定由于该改变而引起的改变的成本和 / 或奖励（是正还是负）。该信息可以被存储以用于神经网络中的增强学习技术的随后应用，例如，用于如图 5 中所示的技术的后续迭代。例如，可以将该信息添加到由神经网络维持的输入 / 动作 / 奖励值的集合（例如，在数据结构中或者通过配置提供该功能的组合的逻辑）中或者在由神经网络维持的输入 / 动作 / 奖励值的集合中更新该信息。然后，该方法可以包括使用替换缓存算法继续执行，如在 640 中。在各个实施方式中，可以由本文所描述的神经网络中的神经元中的任意一个或全部来应用增强学习技术。

[0055] 在各个实施方式中，本文所描述的用于使用神经网络来选择适合的缓存算法的方法可以应用于各种各样的应用，并且可以在包括缓存的各种系统配置上应用。例如，它们可以被应用以针对存储在处理器中的第一级（L1）或第二级（L2）缓存、片上或片下第三级（L3）缓存或者被分配用作特定的计算机系统中的缓存的主存储器或磁盘存储器的一部分中的数据选择替换策略。在其它实施方式中，这些技术可以用于在分布式计算系统中（例如，在向多个应用或用户提供数据、网页、文档或其它资源的云计算环境中）选择替换策略。在一些实施方式中，神经网络可以用于响应于应用的工作负荷的改变或者由负荷平衡机制对资源的分配或重新分配，来选择或修改缓存算法。例如，频繁请求的网页（例如，“频繁询问的问题”页面）可以被缓存在应用服务器上，每当这些网页被请求时，该应用服务器将这些网页提供给用户而不必从磁盘存储设备重新获取。在该情况下，该缓存可以应用最少使用策略来替换缓存中的项目。另一方面，在以更随机的方式存取网页的系统中，用于替换项目的最少使用策略可能不合适。相反，更适合于随机存取的替换策略可以被选择，例如，当确定在缓存中替换哪些页面时考虑页面排序的替换策略。

[0056] 当与仅应用单个缓存算法的系统（例如，当缓存已满时总是删除最旧的项目的缓存算法或者当缓存已满时总是删除最新的项目的缓存算法）相比时，使用神经网络来选择用于给定的应用及其执行环境的适合的缓存算法的系统可以导致改进的应用性能。在一些实施方式中，提供给神经网络的数据越多，其选择可能更准确。在这些实施方式中，在高流量情形下，神经网络可以更快速地学习。在一些实施方式中，除了从已知的缓存算法的集合中选择缓存算法以外，神经网络还可以被配置为应用新的（例如，自组）缓存算法来改善性能。

[0057] 如前所述，在一些实施方式中，本文所描述的神经网络可以被提供作为计算机程序产品或软件，其可以包括其上存储有指令的非临时性计算机可读存储介质，所述指令可以用于对计算机系统（或者其它电子设备）编程以执行本文所描述的技术。例如，根据不同的实施方式，本文所描述的各个神经元的功能可以体现在各个单独的软件模块中或者运行库中的模块中。在各个实施方式中，计算机可读介质可以包括用于以机器（例如，计算机）可读的形式（例如，软件、处理应用）存储或发送信息的任何机制。计算机可读存储介质可以包括但不限于磁存储介质（例如，软盘）；光学存储介质（例如，CD-ROM）；磁光存储介质；只读存储器（ROM）；随机存取存储器（RAM）；可擦除可编程存储器（例如，EPROM 和 EEPROM）；闪存；适合于存储程序指令的电子或其它类型的介质。此外，可以使用光学、声学或其它形式的传

播信号(例如,载波、红外线信号、数字信号等)来传送程序指令。

[0058] 在一些实施方式中,可以以各种各样的计算系统中的任意一种来执行本文所描述的用于使用神经网络来选择适合的缓存算法的技术。图 7 示出了被配置为使用本文所描述的和根据各个实施方式的神经网络来选择适合的缓存算法的计算系统。计算机系统 700 可以是各种类型的设备中的任意一种,其包括但不限于 :个人计算机系统、台式计算机、膝上型或笔记本计算机、大型计算机系统、手持式计算机、工作站、网络计算机、消费者设备、应用服务器、存储设备、外围设备(例如,开关、调制解调器、路由器等),或者通常任何类型的计算设备。

[0059] [0059] 如图 7 中所示,计算机系统 700 可以包括系统存储器 710(例如,缓存、SRAM DRAM、RDRAM、EDO RAM、DDR RAM、SDRAM、Rambus RAM、EEPROM 等中的一个或多个)、一个或多个处理器 770(其中的每一个可以包括一个或多个缓存,例如,L1 缓存 772 和 L2 缓存 775)、存储器管理单元(MMU)780(其可以包括变换索引缓冲,例如,TLB785)、一个或多个输入 / 输出接口 760、第三级(L3)缓存 720、神经网络 730 和互连 790。在一些实施方式中,计算机系统 700 可以通信地耦合到一个或多个远程存储设备,例如,磁存储设备 750,并且还可以耦合到一个或多个输入 / 输出设备 755。

[0060] 如本文所描述的,神经网络 730 可以包括输入层 732、隐藏层 734 (其可以实现为单级处理单元或者处理单元的层级,如本文所描述的)和输出层 736 (其可以提供输出以控制各个动作,例如,选择和动态地替换缓存算法和 / 或其参数)。在各个实施方式中,神经网络 730 可以实现在硬件(例如,专用电路或协同处理器)、软件(例如,当在一个或多个处理器 770 上执行时实现神经网络的输入层、隐藏层或输出层的功能的程序指令)或用于执行神经网络的功能的程序指令和支持电路的组合中。

[0061] 在一些实施方式中,系统存储器 710 可以包括被配置为执行神经网络的各个模块(例如,一个或多个输入神经元、一个或多个隐藏层神经元(即,处理单元)和 / 或一个或多个输出神经元)的程序指令和数据。在各个实施方式中,系统存储器 710 可以存储应用代码 715 和 / 或可以包括数据存储部分 717。在一些实施方式中,系统存储器 710 还可以包括被配置为执行操作系统代码 712 的程序指令和数据。在一些实施方式中,输入层 732 可以包括一个或多个输入神经元,其被配置为接收和 / 或检测各个与性能有关的参数值、与性能有关的触发、与应用进行的存取有关的信息、硬件或操作系统参数值或者在运行时的其它与性能有关的信息。

[0062] 注意,可以用各种编程语言或方法中的任意一种来实现应用代码 715、执行神经网络 730 的任何代码和 / 或操作系统代码 712 中的每一个。例如,在一个实施方式中,应用代码 715、神经网络 730 和操作系统代码 712 可以是基于 JAVA 的,而在其它实施方式中,可以使用 C 或 C++ 编程语言来写应用代码 715、神经网络 730 和操作系统代码 712。此外,在一些实施方式中,可能不能使用相同的编程语言来实现应用代码 715、神经网络 730 和操作系统代码 712。例如,应用源代码 715 可以是基于 C++ 的,而可以使用 C 来开发神经网络 730。

[0063] 在一些实施方式中,系统存储器 710 可以包括数据存储区域 717。在一些实施方式中,数据存储设备 717 可以存储硬件和 / 或操作系统的参数值、由应用代码 715 存取或生成的数据和 / 或由神经网络 730 维持的动作 / 奖励值(例如,当神经网络 730 使用增强学习或另一种自我学习技术时)。在其它实施方式中,数据存储区域 717 可以被划分为多个数据存

储区域，和／或可以分布在多个机器或计算机系统上。在一个实施方式中，这些数据存储区域中的一个或多个可以位于远程存储设备（例如，磁存储设备 750）上。在一些实施方式中，可以从系统存储器 710 或数据存储设备 717 分配 L3 缓存，而在其它实施方式中，可以将 L3 缓存实现为不同的内存块（如 720 所示）。

[0064] 处理器 770 可以被配置为执行各个指令集架构（例如，x86、SPARC、PowerPC 等）中的任意一个。在一些实施方式中，处理器 2370 可以包括单个 CPU 内核、多个 CPU 内核或者一个或多个通用 CPU 内核与专用内核的任意组合（例如，数字信号处理器、硬盘加速计、协同处理器等）。在各个实施方式中，处理器 770 可以被配置为执行超标量架构或者可以被配置为执行多线程。在一些实施方式中，处理器 770 可以是芯片多线程（CMT）处理器。

[0065] 如图 7 所示并且如上所述，处理器 770 可以包括 L1 缓存和 L2 缓存。在该实施例中，可以通过从一个或多个本地缓存（例如，L1 缓存 772 或 L2 缓存 775）、从 L3 缓存 720、从系统存储器 710 或者从磁存储设备 750 中获取缓存的数据，来满足存储器存取请求。当满足数据请求时，该响应可以包括关于数据的源是否是本地的而不是远程的和／或关于数据的源是否是系统中的缓存的指示（例如，关于 L1、L2 或 L3 缓存命中的指示或者关于 L1、L2 或 L3 缓存未命中的指示）。

[0066] 如所示的，互连 790 可以将处理器 770 耦合到存储器 710。在一些实施方式中，互连 790 和输入／输出接口 760 可以被配置为实现各个接口或网络标准中的一个或多个，例如，外围组件互连（PCI）、以太网、超传输（HT）、无限宽带或者这些或其它适合的输入／输出协议的任何变形或继任者。在一些实施方式中，可以在计算机系统 700 中包括一个或多个硬件事件计数器（未示出）以在执行应用代码 715 期间收集与性能有关的数据。

[0067] 虽然本文已经参照具体实施方式并且在具体实施方式的上下文中描述了各个系统和方法，但是将理解的是，这些实施方式是说明性的并且本发明的范围不限于这些具体的实施方式。很多改变、修改、添加和改善是可能的。例如，在说明书中标识的框和逻辑单元是为了理解所描述的实施方式而不意味着限制本发明。在本文所描述的或者使用不同的术语所描述的系统和方法的各种实现中，可以用框区别地分离或组合功能。

[0068] 这些实施方式意味着是说明性的而不是限制性的。因此，可以针对本文作为单个实例描述的组件提供多个实例。各个组件、操作和数据存储之间的边界有时是任意的，并且在具体的说明性配置的上下文中示出了特定的操作。功能的其它分配被设想并且可以落入下面的权利要求的范围内。最后，呈现为示例性的配置中的分立组件的结构和功能可以实现为组合的结构或组件。这些和其它变化、修改、添加和改善可以落入下面的权利要求所定义的本发明的范围内。

[0069] 虽然已经详细地描述了上面的实施方式，但是一旦完全理解上面的公开内容，大量改变和修改就将变得显而易见。期望将下面的权利要求解释为涵盖所有这些改变和修改。

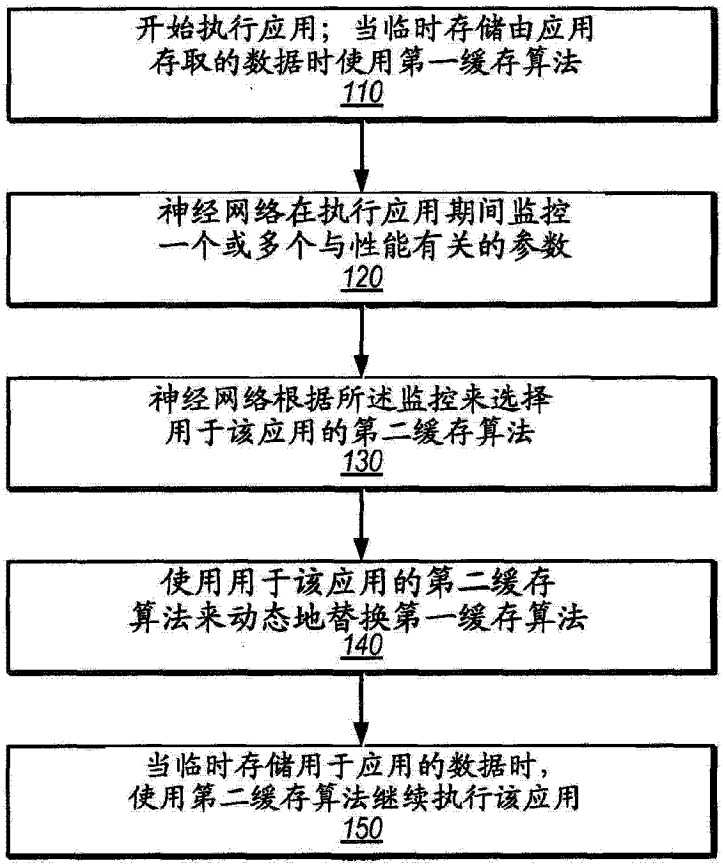

开始执行应用；当临时存储由应用存取的数据时使用第一缓存算法
110
神经网络在执行应用期间监控一个或多个与性能有关的参数
120
神经网络根据所述监控来选择用于该应用的第二缓存算法
130
使用用于该应用的第二缓存算法来动态地替换第一缓存算法
140
当临时存储用于应用的数据时，使用第二缓存算法继续执行该应用
150

图 1

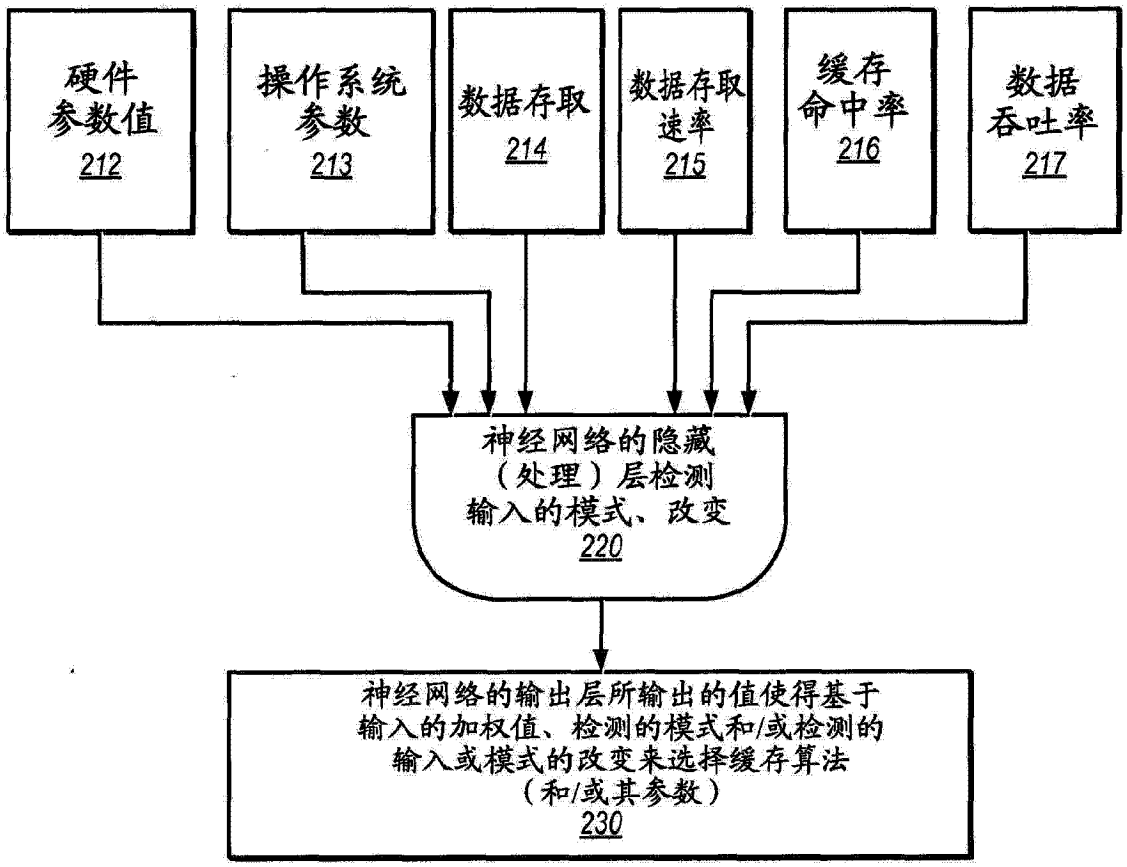

硬件
参数值
212
操作系统
参数
213
数据存取
214
数据存取
速率
215
缓存
命中率
216
数据
吞吐率
217
神经网络的隐藏
（处理）层检测
输入的模式、改变
220
神经网络的输出层所输出的值使得基于
输入的加权值、检测的模式和/或检测的
输入或模式的改变来选择缓存算法
（和/或其参数）
230

图 2

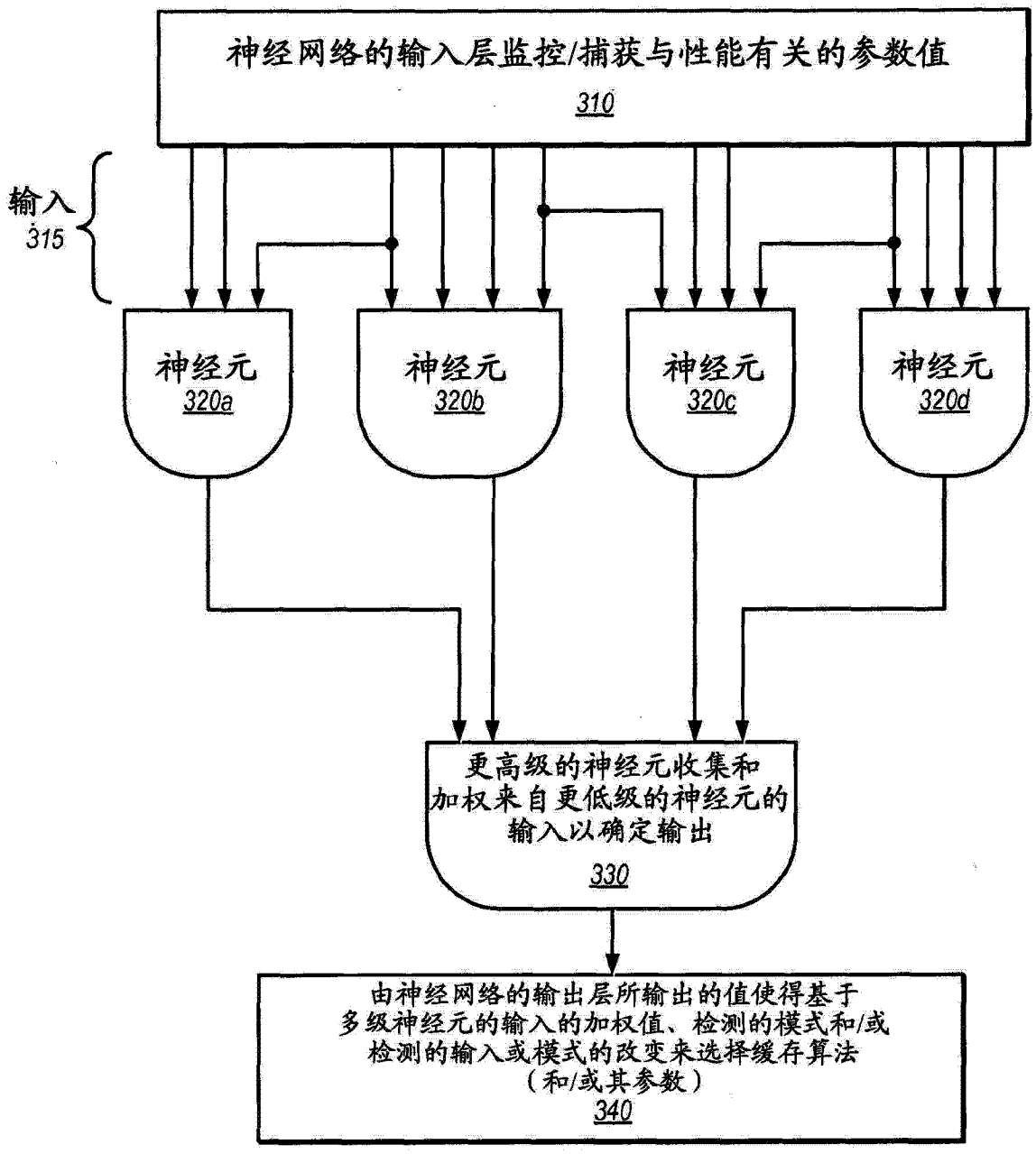

神经网络的输入层监控/捕获与性能有关的参数值
310
输入
315
神经元
320a
神经元
320b
神经元
320c
神经元
320d
更高级的神经元收集和
加权来自更低级的神经元的
输入以确定输出
330
由神经网络的输出层所输出的值使得基于
多级神经元的输入的加权值、检测的模式和/或
检测的输入或模式的改变来选择缓存算法
（和/或其参数）
340

图 3

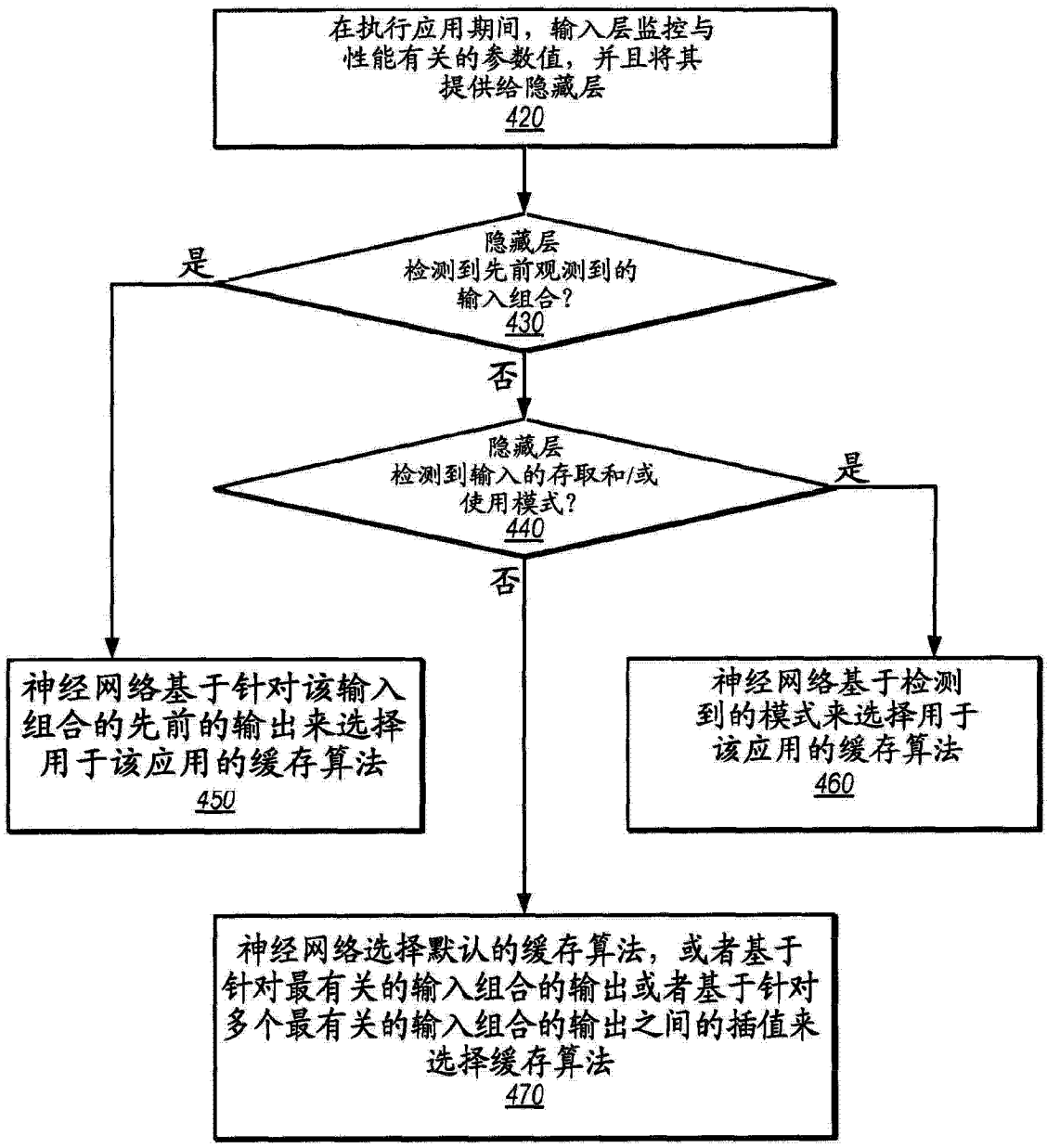

在执行应用期间，输入层监控与性能有关的参数值，并且将其提供给隐藏层
420
隐藏层检测到先前观测到的输入组合？
430
是
否
隐藏层检测到输入的存取和/或使用模式？
440
是
否
神经网络基于针对该输入组合的先前的输出来选择用于该应用的缓存算法
450
神经网络基于检测到的模式来选择用于该应用的缓存算法
460
神经网络选择默认的缓存算法，或者基于针对最有关的输入组合的输出或者基于针对多个最有关的输入组合的输出之间的插值来选择缓存算法
470

图 4

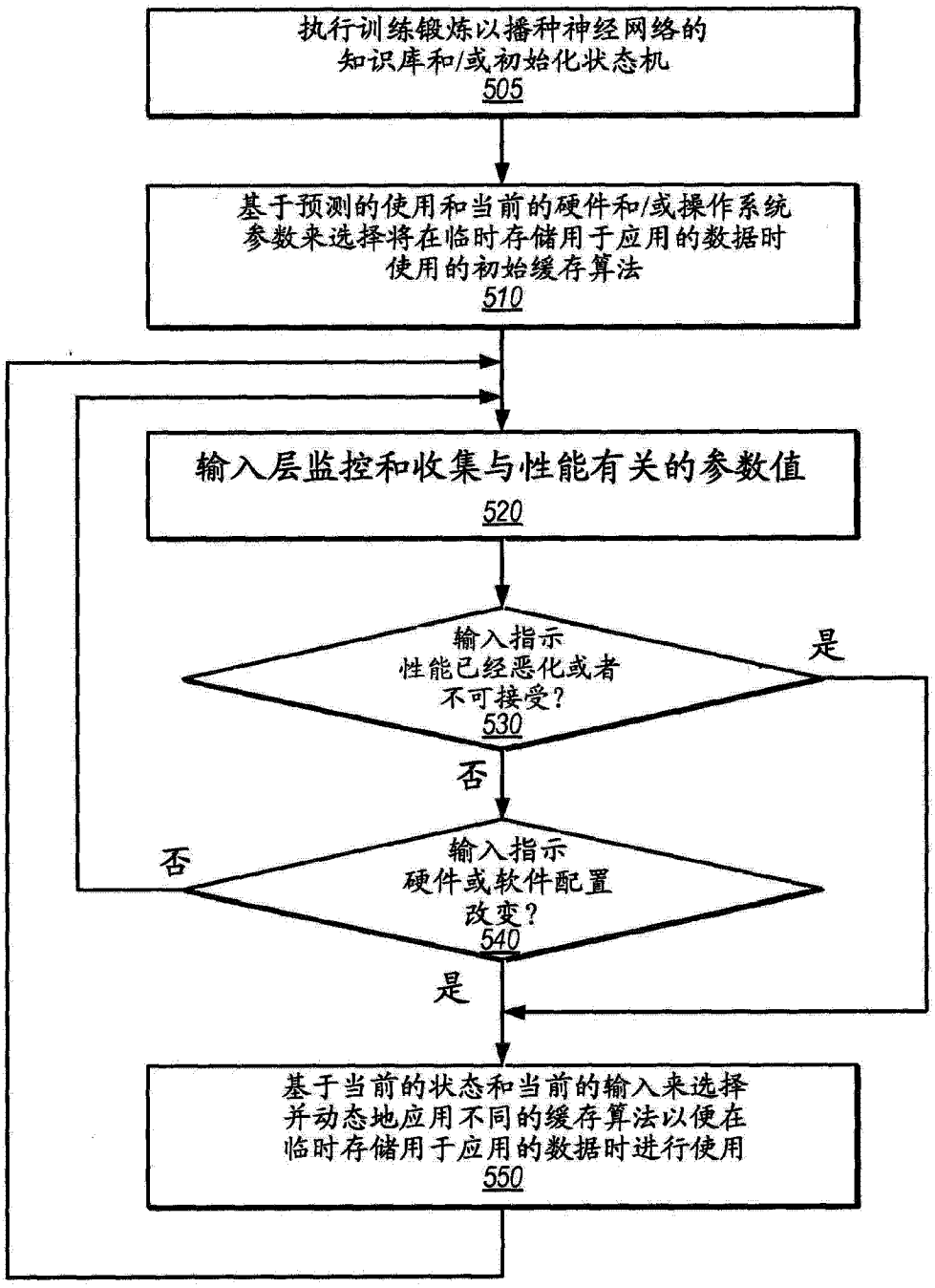

执行训练锻炼以播种神经网络的
知识库和/或初始化状态机
505
基于预测的使用和当前的硬件和/或操作系统
参数来选择将在临时存储用于应用的数据时
使用的初始缓存算法
510
输入层监控和收集与性能有关的参数值
520
输入指示
性能已经恶化或者
不可接受?
530
是
否
输入指示
硬件或软件配置
改变?
540
否
是
基于当前的状态和当前的输入来选择
并动态地应用不同的缓存算法以便在
临时存储用于应用的数据时进行使用
550

图 5

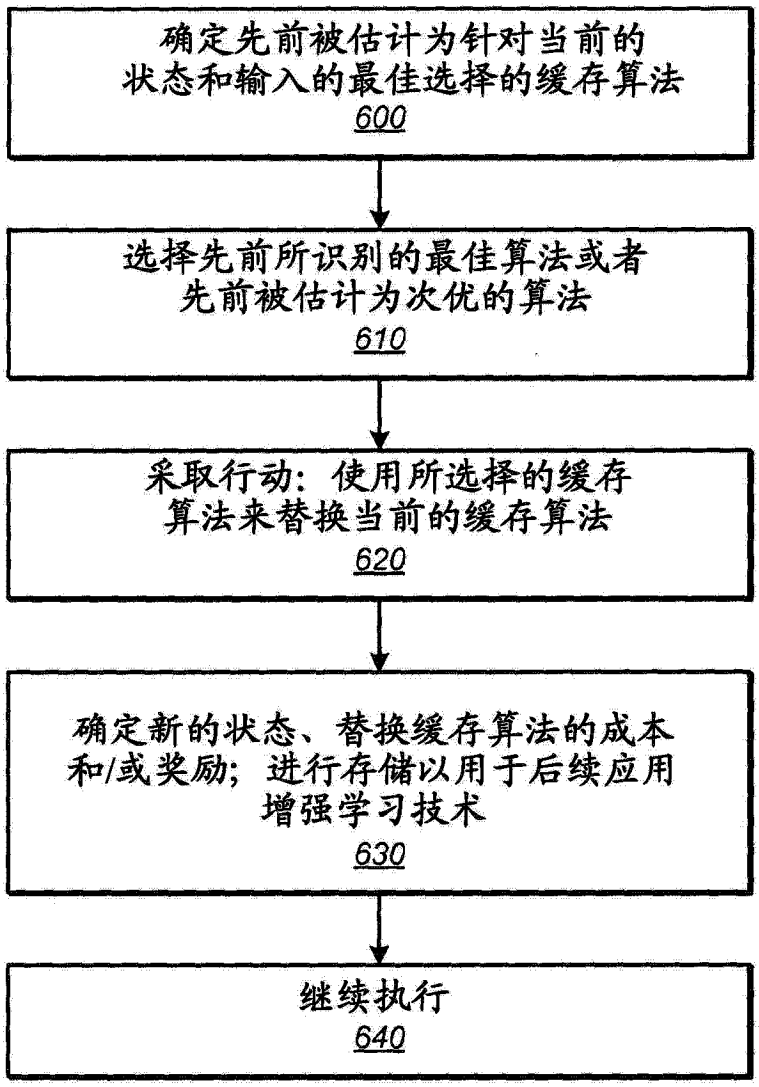

确定先前被估计为针对当前的
状态和输入的最佳选择的缓存算法
600
选择先前所识别的最佳算法或者
先前被估计为次优的算法
610
采取行动：使用所选择的缓存
算法来替换当前的缓存算法
620
确定新的状态、替换缓存算法的成本
和/或奖励；进行存储以用于后续应用
增强学习技术
630
继续执行
640

图 6

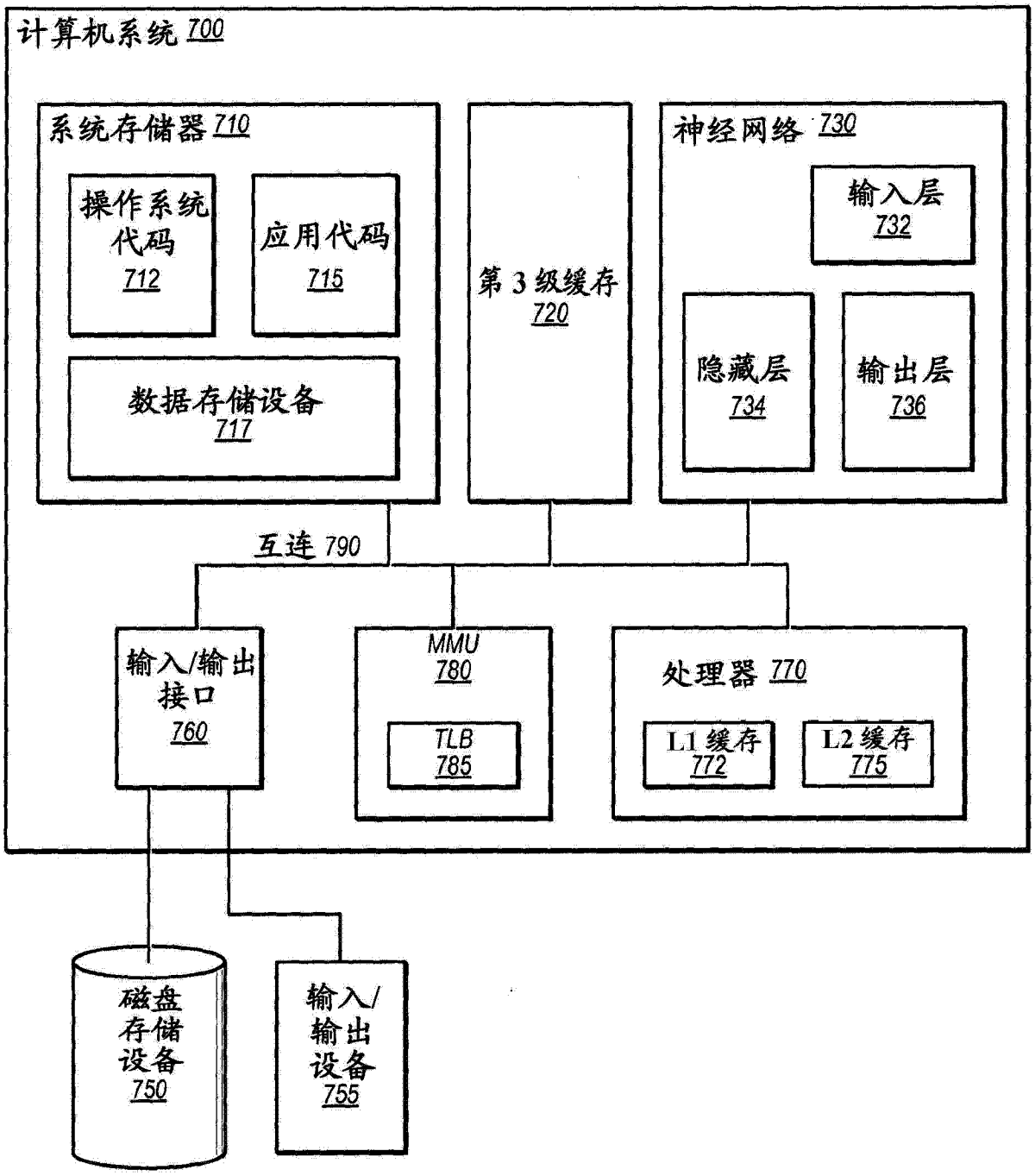

计算机系统 700
系统存储器 710
操作系统代码 712
应用代码 715
数据存储设备 717
第 3 级缓存 720
神经网络 730
输入层 732
隐藏层 734
输出层 736
互连 790
输入/输出接口 760
MMU 780
TLB 785
处理器 770
L1 缓存 772
L2 缓存 775
磁盘存储设备 750
输入/输出设备 755

图 7