# 基于共享 GPU 的深度学习训练性能实证研究

徐涣霖[1,2] 顾嘉臻[1,2] 康 昱[3] 周扬帆[1,2]

[1](复旦大学计算机科学技术学院 上海 200433)

[2](上海市智能信息处理重点实验室 上海 200433)

[3](微软亚洲研究院 北京 100080)

**摘 要** 深度学习应用的训练过程是计算密集型的,它通常依靠图形处理单元(Graphics Processing Unit,GPU)来加速训练过程。然而深度学习开发框架往往会独占 GPU,造成计算资源的浪费。针对该问题,该实证研究对两个深度学习应用共享 GPU 训练的可行性进行讨论,系统地分析了有代表性的深度学习模型的静态和运行时特性,展示了共享 GPU 训练两个模型时,不同的模型组合和特征对整体性能的影响。根据实验结果所总结的原则可以作为提高调度效率和改善 GPU 云资源利用率的指导方针。

**关键词** 性能分析 GPU 应用程序 深度学习 实证研究

**中图分类号** TP311.5 **文献标志码** A DOI:10.3969/j.issn.1000-386x.2022.12.023

# AN EMPIRICAL STUDY ON TRAINING PERFORMANCE OF DEEP LEARNING BASED ON SHARED-GPU

Xu Huanlin[1,2] Gu Jiazhen[1,2] Kang Yu[3] Zhou Yangfan[1,2]

[1](*School of Computer Science, Fudan University, Shanghai 200433, China*)

[2](*Shanghai Key Laboratory of Intelligent Information Processing, Shanghai 200433, China*)

[3](*Microsoft Research Asia, Beijing 100080, China*)

**Abstract** The training process of deep learning(DL) application is computation-intensive. It often relies on Graphics Processing Unit(GPU) to accelerate the training process. However, DL frameworks tend to monopolize the GPU, resulting in a waste of computing resource. In view of the problem, this paper conducts an empirical study on exploring the feasibility of executing two DL applications on one shared-GPU. We systematically analyzed the static and runtime characters of several typical DL models, and showed the influence of different model combinations and characteristics on the overall performance when shared-GPU trained two models. The principles summarized from experimental results can serve as guidelines to improve the scheduling efficiency and the utilization of GPU cloud resources.

**Keywords** Performance analysis GPU application Deep learning Empirical study

## 0 引 言

深度学习技术已经在自然语言处理[1]、图像分类[2]和推荐系统[3]等众多领域中被证明是卓有成效的,甚至在特定领域中超越了人类的表现[4]。这使得深度学习模型被广泛地训练和部署。通常情况下,一个深度学习模型会包含数以千万计的参数。为了能尽可能提高深度学习应用的开发效率,通用计算能力更强的图形处理单元(GPU)被大量应用于模型训练中。

因此近年来随着深度学习应用的普及,GPU 越来越被认为是一种重要的计算资源。云端提供 GPU 计算设备的服务正变得流行。诸如 Google Cloud、Amazon Web Service 和 Microsoft Azure 等云平台,已经向开发者提供了 GPU 云计算服务。出于商业考虑,云服务供应商通常会利用虚拟化技术使得多个应用程序之间共

收稿日期:2020-06-02。国家自然科学基金项目(61702107)。**徐涣霖**,硕士生,主研领域:软件可靠性。**顾嘉臻**,博士生。**康昱**,博士。**周扬帆**,副教授。

享一个 GPU 设备[5],以最大化经济效益。因此当用户使用云服务来训练其深度学习模型时,其使用的 GPU 设备可能同时为其他用户执行训练进程[6]。在运行适当的深度学习模型组合时,GPU 云平台的资源利用率有望得到提高,但也可能导致整体训练效率的大幅度下降。如何在这种情境下优化 GPU 资源分配成为一个值得研究的问题。

我们希望通过研究说明,各种类型的深度学习模型在一个 GPU 设备上同时训练时,训练进程的具体性能表现和原因。此研究结果可以为提高 GPU 云平台资源利用率和优化深度学习模型训练进程的调度提供参考,使用户和服务提供商受益。为了达到这个目标,我们精心设计了四个研究问题,并通过相关实验研究来回答这些问题。研究结果提供了训练不同的深度学习模型组合时的性能表现,并且展示了训练进程性能表现与模型属性之间的关系。

## 1　研究问题

我们在本节提出了四个研究问题。通过回答这些问题,旨在更好地了解在一个 GPU 设备上同时训练多个深度学习模型的性能表现。首先定义共享训练的概念。

**定义 1**　如果训练深度学习模型 A 和 B 的进程同时在同一个 GPU 设备上运行,则称之为深度学习模型 A 和 B 的共享训练。

GPU 设备的内存资源对模型训练非常重要,因为 GPU 内存大小决定训练进程是否可以正常运行至结束。在这种情况下,现有的深度学习开发框架的默认设置倾向占用尽可能多的 GPU 内存。因此,第一个研究问题如下。

**问题 1**　当一个深度学习模型单独占用 GPU 设备训练时,分配不同的 GPU 内存大小是否会影响它的训练进程运行时间?

这种独占资源的调度使得一个 GPU 设备每次只能训练一个深度学习模型。如果用于训练的应用程序实际上只需要很少的 GPU 内存,就将导致资源的浪费。这在 GPU 云平台中尤其不可取。一种提高资源利用率的朴素方案是让一个 GPU 设备同时运行多个深度学习模型的训练进程。然而在一个 GPU 设备上共享训练多个深度学习模型时,训练进程的整体效率尚不明确。我们对几个具有代表性的深度学习模型组合进行了实验,旨在讨论共享训练将使训练性能受到多大程度的影响,不同深度学习模型组合的具体性能表现如何。第二个研究问题如下。

**问题 2**　当共享训练深度学习模型 A 和 B 时,是否会影响深度学习模型 A 的训练进程运行时间?如果是,那么模型组合类型是否与训练运行时间变化有联系?是否存在共享训练效率更高的情况?

在回答以上问题后,我们进而研究为什么两个深度学习模型共享训练会对互相的训练进程性能进行产生影响。为了后续研究,分析了不同深度学习模型的底层运算属性。第三个研究问题如下。

**问题 3**　不同深度学习模型的运算属性是否存在显著差异?

深度学习模型训练进程所需系统资源主要分为 GPU 核心的计算资源和数据交换的 I/O 资源。不同类型的模型对计算资源和 I/O 资源的需求不同。一个深度学习模型可能是 I/O 密集型或(/和)计算密集型的。在共享训练中,不同进程可能会相互争夺计算和 I/O 资源。这种资源竞争可能导致整体性能下降。所以第四个研究问题如下。

**问题 4**　共享训练时是否存在系统资源竞争的情况?如果是,是否是导致每个训练进程性能下降的因素?

## 2　研究方法

### 2.1　共享训练的性能分析

首先测试共享训练时的进程性能表现。由于在模型的训练过程中每次迭代的时间是相对稳定的,可以使用每次迭代的时间来近似表示模型的总训练时间。首先独立训练每个模型并记录它们的训练时间,再将这些深度学习模型两两组合进行共享训练,以获得各模型组合的整体训练性能表现。定义性能退化系数的概念如下。

**定义 2**　假设独立训练模型 A 时,每次迭代所需要的时间是 $t_{A1}$。假设模型 A 与模型 B 同时训练时,模型 A 所需要的时间是 $t_{A2}$。将 $\frac{t_{A2}}{t_{A1}}$ 称作是模型 A 与模型 B 共享训练时的性能退化系数 $r_{AB}$。

性能退化系数 $r_{AB}$ 可以表明模型 A 和 B 在共享训练中,模型 A 的性能退化情况,系数越大性能退化越严重。如果两个模型的共享训练比分别独立训练更快,那么共享训练是有益的。共享训练下总训练时间更短的条件的推导过程如下。

首先,假设模型 A 和 B 独占 GPU 时的训练时间分别为 $t_A$ 和 $t_B$,共享训练时的性能退化系数分别为 $r_{AB}$ 和 $r_{BA}$。则模型 A 和 B 共享训练时的总训练时间 $t$ 可以

表示为：

$$t=\begin{cases}t_A\times r_{AB}+t_B\times\left(1-\dfrac{t_A\times r_{AB}}{t_B\times r_{BA}}\right) & t_A\times r_{AB}\leqslant t_B\times r_{BA}\\ t_B\times r_{BA}+t_A\times\left(1-\dfrac{t_B\times r_{BA}}{t_A\times r_{AB}}\right) & t_A\times r_{AB}>t_B\times r_{BA}\end{cases}$$

如果有 $t<t_A+t_B$ 则表明共享训练比顺序训练快。考虑所有情况可以计算出，共享训练比顺序训练更有效率的条件是：$r_{AB}\times r_{BA}<r_{AB}+r_{BA}$，或者表示为更容易计算的形式 $(r_{AB}-1)\times(r_{BA}-1)<1$。

综上所述，对于模型 A 和 B，如果我们得到 $r_{AB}$ 和 $r_{BA}$，可以知道：1）模型 A 和 B 的共享训练相对于每个模型独立训练的性能差别；2）模型 A 和 B 在共享训练和分别独立训练两种情况下，哪种的总训练时间更短。

### 2.2 深度学习模型属性对共享训练性能的影响

进一步研究共享训练时进程性能下降的原因。通常，GPU 计算和 I/O 操作的工作负载是影响深度学习模型训练性能的两个主要因素。与 GPU 计算相关的属性包括进程内的计算操作执行率等。与 I/O 相关的属性包括进程所需的内存大小、总体数据读取请求速率和总体数据写入请求速率等。以上这些属性描述了深度学习模型训练进程的总体执行特性。我们检验了不同深度学习模型的属性是否存在显著差异。进而比较了在共享训练的过程中，GPU 核心利用率和 I/O 吞吐量与独立训练相比有什么区别，目标是探究共享训练中是否存在系统资源竞争以及对进程性能的影响。为了验证假设结果的相关性，进行 KS 检验和卡方检验来进一步保证结果的有效性。

### 2.3 其他 GPU 设备上的结果

以上的研究旨在展示深度学习模型在共享训练过程中的表现以及造成这种现象的潜在原因。为了证明以上实验结果的普遍性，我们在不同的 GPU 设备上重复了相同的实验，并对实验结果进行了比较。

## 3 研究结果

### 3.1 实验设置说明

我们重点关注三类有代表性的深度学习模型，分别是全连接神经网络[7]（FCNs）、卷积神经网络[8]（CNNs）和递归神经网络[9]（RNNs）。这些模型的建模能力是互补的[10]，并在众多领域得到了广泛应用。由于深层 FCNs 的应用受到限制，使用 5 层全连接网络作为 FCNs 的代表[10]。为了研究不同结构的 CNN，我们实现并测试了两个具有代表性的 CNN 模型，即 VGG-19[11] 和 ResNet-50[2]。LSTM[1] 用于建模序列结构，我们将其作为 RNNs 的代表。实验中涉及到的模型配置和相应的数据集的细节如表 1 所示。

**表 1 实验模型和数据集的相关细节**

| 模型 | | 数据集 | 层数 | 参数量 |
|---|---|---|---|---|
| FCN | FCN | MNIST | 5 | 约 1 万 |
| CNN | VGG | ImageNet | 19 | 约 3 000 万 |
| | ResNet | | 50 | 约 2 500 万 |
| RNN | LSTM | IMDB | 2 | 约 20 万 |

实验中涉及的所有深度学习模型全部利用 TensorFlow 1.13 实现，Python 版本号为 3.6，实验 GPU 为 NVIDIA GTX 1080Ti，测试服务器系统为 Ubuntu 16.04，配置了 CUDA Toolkit 9.0 和 cuDNN 7.0。

### 3.2 GPU 内存大小对训练性能的影响

为了回答问题 1，首先通过配置 GPU 内存系数，来调整训练深度学习模型的应用程序可用的 GPU 内存大小，接着以独占 GPU 的模式训练四个目标深度学习模型。对每个模型进行 1 000 次迭代的训练，并记录训练进程执行的总时间作为训练性能表现的指标。为了保证统计学上的意义，对每种配置分别进行 3 次训练，计算时间的平均值作为结果。该实验结果如表 2 所示。表 2 中的小数表示在该 GPU 内存系数配置下与使用全部内存（系数为 1.0）的训练时间的比值。

**表 2 不同 GPU 内存大小下的性能变化情况**

| 模型 | GPU 内存系数 | | | |
|---|---|---|---|---|
| | 0.2 | 0.4 | 0.6 | 0.8 |
| FCN | 1.002 | 1.003 | 1.009 | 1.006 |
| VGG | – | 1.002 | 1.003 | 1.001 |
| ResNet | – | 1.002 | 1.000 | 1.001 |
| LSTM | 1.000 | 1.004 | 1.005 | 1.002 |

从结果可以发现，当分配的 GPU 内存足够使训练进程正常运行后，训练进程的运行时间变化与 GPU 内存系数增加没有显著关系。

**结论 1** 当深度学习模型独占 GPU 训练时，如果分配的 GPU 内存大小已经足够训练进程正常运行，那么额外分配的 GPU 内存对训练时间没有显著影响。

### 3.3 共享训练对整体性能的影响

将之前实验的四种模型进行两两组合，对这些组合进行共享训练，分别记录总训练时间。为了模拟实际应用中的情况，首先在系统后台运行训练模型 A 的进程，然后启动训练模型 B 的进程，来研究共享训练模式对模型 A 的训练进程的性能影响。给每个进程分配所需的最少 GPU 内存，在共享训练的模式下进行

1 000 次迭代,并将得到的执行时间与相同配置在独占 GPU 模式下训练的进程执行时间相除,得到比值作为性能退化系数记录下来。结果如表 3 所示。

**表 3　不同模型组合的共享训练性能退化比率**

| 已有模型 | 新模型 | | | |
|---|---|---|---|---|
| | FCN | VGG | ResNet | LSTM |
| FCN | 1.142 | 4.576 | 2.518 | 1.622 |
| VGG | 1.091 | 2.073 | 1.736 | 1.131 |
| ResNet | 1.023 | 1.895 | 1.573 | 1.020 |
| LSTM | 1.334 | 4.146 | 2.532 | 1.166 |

可以看出,在共享训练时每个模型相比于独立训练都会出现不同程度的性能退化。不同的模型组合的性能变化差别非常显著。同时根据之前的讨论,当 $r_{AB} \times r_{BA} < r_{AB} + r_{BA}$ 时,共享训练该模型组合比分别独立训练它们所需要的时间更短。而一些模型组合确实符合该要求。

**结论 2**　共享训练会使单个模型的训练进程运行时间延长。模型组合类型与训练运行时间变化情况存在联系,且一些模型组合的共享训练效率比分别独立训练更高。

## 3.4　模型底层运算属性的情况

为了研究模型属性与共享训练的关系,需要分析不同的深度学习模型的底层运算属性是否存在区别。我们利用 TensorFlow 框架内建的分析功能,收集训练进程运行时所有涉及到的运算符种类和其执行时间百分比形成一个运行时运算符集合。

图 1 以 FCN 模型为例,展示了运行时运算符集合的分布情况。该模型的运行时运算符集合内包含 Mul 运算符,并且其数值标记为 38%。即表示在该模型中 Mul 运算符消耗 38% 的训练时间。

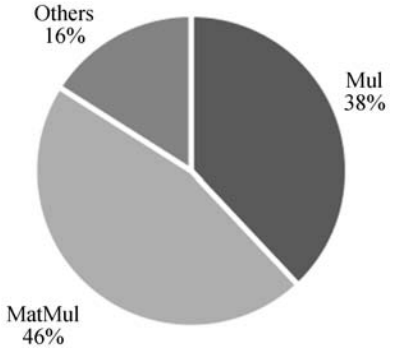


图 1　FCN 模型的运算符运行时间百分比

在收集到所有模型的运算符集合分布情况后,计算它们的标准化 Jaccard 相似度。度量结果见表 4。可以看出,不同深度学习模型的训练进程执行的底层操作差距确实非常显著。

**表 4　不同模型运算符集合间的 Jaccard 相似度**

| 模型 | FCN | VGG | ResNet | LSTM |
|---|---|---|---|---|
| FCN | 1 | 0.028 5 | 0.192 8 | 0.421 1 |
| VGG | 0.028 5 | 1 | 0.208 3 | 0.028 5 |
| ResNet | 0.192 8 | 0.208 3 | 1 | 0.274 4 |
| LSTM | 0.421 1 | 0.028 5 | 0.274 4 | 1 |

**结论 3**　不同深度学习模型的底层运算属性存在显著差异。

## 3.5　I/O 操作和 GPU 计算特性与共享训练性能的关系

在明确了不同模型的运算属性存在差异后,我们进一步研究了不同情况下训练时模型各运算符的 CPU 和 GPU 执行时间。表 5 展示了 FCN 模型的相关结果。可以看出,共享训练与独占训练比较,模型各运算符 GPU 执行时间的分布情况几乎相同,而 CPU 执行时间分布情况表现出显著差异。

**表 5　不同 FCN 模型组合共享训练与独占训练的总计、CPU 和 GPU 执行时间的 Jaccard 相似性**

| 模型组合 | 总计 | GPU | CPU |
|---|---|---|---|
| FCN 与 FCN | 0.095 | 0.955 | 0.088 |
| FCN 与 VGG | 0.079 | 0.986 | 0.079 |
| FCN 与 ResNet | 0.079 | 0.993 | 0.078 |
| FCN 与 LSTM | 0.308 | 0.979 | 0.263 |

以上的实验结果,尤其是共享训练与独占训练差异很大的 CPU 执行时间,表明应该深入研究深度学习模型训练时的 I/O 操作和 GPU 运算操作情况。我们利用 NVIDIA 的 nvprof 工具记录训练进程的两个度量指标,进程的资源加载和存储事务共计为 I/O 事务,展示了系统 I/O 请求情况;GPU 指令数表明 GPU 设备上的计算请求情况。结果如图 2 所示。

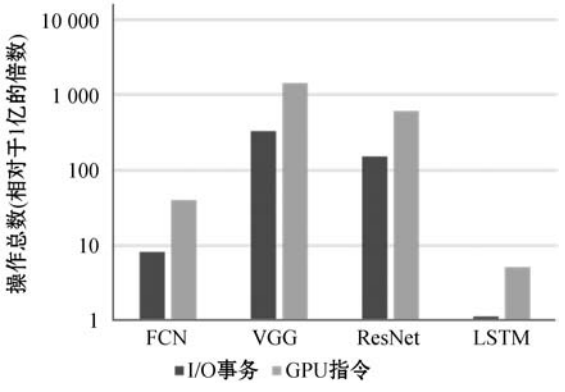


图 2　模型每秒的操作情况(对数刻度)

可以看出,每个模型的操作量与表 1 中的模型参数规模呈正相关。同时联系表 3 中的数据,训练进程的操作量越大,使得共享训练中其他模型性能退化得越多。所以可以得到如下结论。

**结论 4** 如果参与共享训练的深度学习模型的参数规模有显著差异,则较小模型的共享训练性能退化比较大的更加严重。

最后,将探究共享训练过程中是否存在资源竞争情况及其影响,在该实验中关注 GPU 计算和 I/O 操作的指标。我们将一种模型独占训练和将该模型与其他模型组合共享训练,收集 GPU 实际占用率作为训练进程的 GPU 计算情况指标;收集系统内存到 GPU 内存的拷贝吞吐量作为训练进程的 I/O 操作特性。应用 KS 检验和卡方假设检验,验证独占训练和共享训练下的两种指标是否属于同一分布。

图 3 和图 4 展示了独占 GPU 训练与共享训练下的拷贝吞吐量直方图。假设检验得到的 $p$ 值接近于 0,这说明吞吐量分布有显著区别,表明共享训练会影响系统 I/O 操作。人工比较了不同模式下的吞吐量,发现共享训练时的吞吐量出现了显著下降。


图 3 FCN 模型独占训练时的系统内存吞吐率


图 4 FCN 与 FCN 模型共享训练时的系统内存吞吐率

图 5 和图 6 展示了独占 GPU 训练与共享训练下的 GPU 占用率直方图。假设检验的结果表明它们属于同一分布。人工比较了不同模式下的 GPU 实际占用率,发现它们几乎相同。这说明在共享训练过程中,每个模型的 GPU 计算过程没有受到其他模型的显著影响。

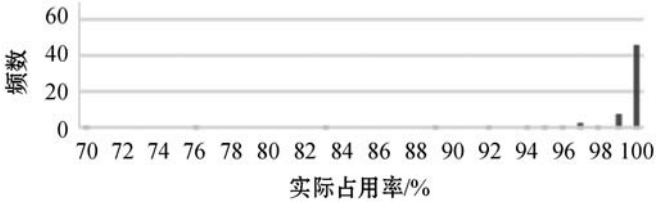


图 5 FCN 模型独占训练时的 GPU 实际占用率

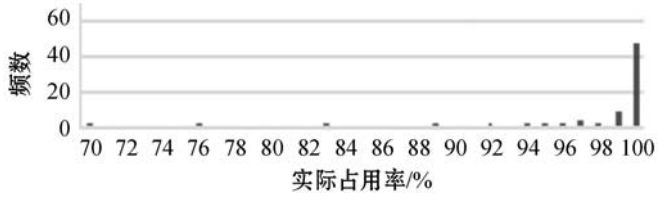


图 6 FCN 与 FCN 模型共享训练时的 GPU 实际占用率

我们联系 GPU 的执行运算的机制分析。由于 GPU 设备上的不同宿主进程会创建独立的进程上下文,且不同上下文中的操作在执行时都会被底层驱动程序序列化而顺序执行,导致无论模型是独占 GPU 训练还是共享训练,都会得到几乎相同的实际占有率。即使调度两个训练进程共享训练,GPU 在某一时刻也只能进行一个模型的底层运算。这使得共享训练时不同的训练进程轮流占用 GPU,导致额外的系统 I/O 开销,进而使得进程不能很好地利用内存的空间局部性。

**结论 5** 共享训练时存在 I/O 操作竞争的情况,并导致了训练进程性能下降。

### 3.6 其他 GPU 设备上共享训练的性能表现

为了验证以上实验中的共享训练性能变化情况是否具有普遍性,在性能较弱的 GPU 设备 GTX 1080 上重复了所有前面的实验。我们得到的绝大多数结果与之前在 GTX 1080Ti 上的结果相似。为了避免冗余,表 6 展示了最主要的共享训练性能退化系数情况。可以看到,GTX 1080 上的共享训练性能变化情况与 GTX 1080Ti 几乎相同。

**表 6 GTX 1080 上模型组合的共享训练性能退化比率**

| 已有模型 | 新模型 | | | |
|---|---|---|---|---|
| | FCN | VGG | ResNet | LSTM |
| FCN | 1.332 | 5.124 | 2.964 | 1.861 |
| VGG | 1.171 | — | — | 1.156 |
| ResNet | 1.034 | — | — | 1.083 |
| LSTM | 1.433 | 4.683 | 2.862 | 1.293 |

**结论 6** 共享训练时的性能变化是普遍存在的,且在相同架构的 GPU 中性能下降的趋势是相似的。

## 4 实践建议

根据以上实验结论,我们提出了优化 GPU 云平台进程调度的若干参考原则。

在分配 GPU 资源时,开发者一般需要考虑 GPU 内存的分配。根据结论 1,可以知道当分配给训练进程的 GPU 内存足以完成训练任务时,继续分配额外的 GPU 内存对训练效率几乎没有影响,这将会产生资源

的浪费。由此可以得到:

**原则 1**　GPU 资源调度程序只需确保每个深度学习模型训练进程已经被分配正常运行所需的最小 GPU 内存。

当一个深度学习模型训练进程所需的资源已经满足时,就可以将剩余的 GPU 内存分配给其他进程。结论 2 表明,是否应该同时执行两个或多个模型训练进程,取决于具体情况。总结如下:

**原则 2**　当模型组合满足 $r_{AB} \times r_{BA} < r_{AB} + r_{BA}$ 的条件时,共享训练可以提高整体的模型训练效率。

结论 4 显示在共享训练时,较小模型的训练进程性能更容易受到影响。根据结论 5,I/O 竞争使得每个训练进程的性能都会不同程度的下降。而结论 3 表明不同模型的运算属性差异明显,这将增加系统的 I/O 操作量,使得 I/O 竞争更加严重。为了尽量避免这种影响,可以得到:

**原则 3**　虽然某些情况下共享训练可以提高整体训练效率,但其中某些模型的训练进程效率可能会严重退化。在实践中为了保证每个用户的使用体验,应该尽量让参数规模和运算属性相近的模型共享训练。

根据结论 6,我们所提出的以上参考原则适用于所有利用 CUDA Toolkit 执行通用计算任务的 NVIDIA GPU。

## 5　相关工作

### 5.1　GPU 云平台资源调度优化

为了满足现代云计算系统中对通用计算,尤其是深度学习相关的应用程序的高性能需求,开发者引入了 GPU 虚拟化技术[12-13]。Giunta 等[14]实现了提高云平台 GPU 虚拟化性能的 gVirtuS 组件。Shi 等[15]提供 vCUDA 以重定向虚拟机中的 CUDA API 从而启用 GPU 加速。还有一系列的研究集中在 GPU 云平台的资源管理上[16-17]。Phull 等[18]提出了一个驱动的管理框架,以提高 GPU 集群的资源利用率。Qi 等[19]提出了 VGRIS 框架,用来虚拟化和高效调度云游戏中的 GPU 资源。Li 等[20]分析和评估 GPU 云计算中多媒体处理服务的定价策略。

### 5.2　深度学习模型训练效率优化

深度学习模型的精度在不断提升的同时,模型的结构和参数规模也变得越来越大,这使得训练和部署变得困难。因此,一些工作的重点是在保持深度学习模型精度的同时减小模型的规模,使模型更适合在云平台上训练和部署[21]。Iandola 等[22]提出了一个与 AlexNet 精度相同的较小的图像分类模型。Luo 等[23]通过从神经元中提取信息来压缩人脸识别模型。一部分开发者利用分布式训练的方法提升效率。Storm[24]通过扩展分布式随机梯度下降来加速训练神经网络。Chen 等[25]利用数据并行性,通过增量训练加速深度学习机的训练。Iandola 等[26]设计了 FireCaffe,它专注于跨集群扩展深度学习模型训练,并减少集群间的通信开销。

## 6　结　语

本文对深度学习模型在 GPU 设备上的共享训练性能进行了研究。我们重点关注在共享 GPU 设备资源的情况下,深度学习训练进程的性能表现将如何变化以及产生这种变化的原因。通过实验研究,我们发现在共享训练的情况下,每个深度学习模型的训练进程的执行时间都会延长,部分模型组合的总训练时间会缩短。造成训练进程性能退化的主要原因是额外的系统 I/O 操作使得进程的等待时间变得更长。这也导致规模相差悬殊的模型共享训练时,小模型的训练进程性能退化的情况更加严重。最后我们证明了以上实验结论具有普遍性,所总结的原则有助于提高 GPU 云平台上的资源利用率。

## 参　考　文　献


[1] Sundermeyer M, Schluter R, Ney H, et al. LSTM neural networks for language modeling[C]//Conference of the International Speech Communication Association, 2012:194-197.

[2] He K, Zhang X, Ren S, et al. Deep residual learning for image recognition [C]//IEEE Conference on Computer Vision & Pattern Recognition,2016: 770-778.

[3] Oord A V D, Dieleman S, Schrauwen B, et al. Deep content-based music recommendation[C]//Neural Information Processing Systems,2013:2643-2651.

[4] Lecun Y, Bengio Y, Hinton G E, et al. Deep learning[J]. Nature,2015,521(7553):436-444.

[5] Hong C H, Spence I, Nikolopoulos D S. GPU virtualization and scheduling methods: A comprehensive survey[J]. ACM Computing Surveys, 2017,50(3):1-37.

[6] Goswami A, Young J, Schwan K, et al. GPUShare: Fair-Sharing middleware for GPU clouds[C]//2016 IEEE International Parallel and Distributed Processing Symposium Workshops(IPDPSW),2016: 1769-1776.

[7] Lecun Y, Boser B, Denker J, et al. Backpropagation applied to handwritten zip code recognition[J]. Neural Com-

putation,1989,1(4):541 - 551.
[8] Krizhevsky A, Sutskever I, Hinton G. ImageNet classification with deep convolutional neural networks[J]. Advances in Neural Information Processing Systems,2012,25(2):1097 - 1105.
[9] Mikolov T, Karafiát M, Burget L, et al. Recurrent neural network based language model[C]//11th Annual Conference of the International Speech Communication Association,2010: 1045 - 1048.
[10] Sainath T N, Vinyals O, Senior A, et al. Convolutional, long short-term memory, fully connected Deep Neural Networks[C]//2015 IEEE International Conference on Acoustics, Speech and Signal Processing(ICASSP), 2015:4580 - 4584.
[11] Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition[EB]. arXiv:1409.1556,2014.
[12] Xue M, Ma J, Li W, et al. Scalable GPU virtualization with dynamic sharing of graphics memory space[J]. IEEE Transactions on Parallel and Distributed Systems,2018,29(8): 1823 - 1836.
[13] Ma J, Zheng X, Dong Y, et al. gMig: Efficient GPU live migration optimized by software dirty page for full virtualization[C]//14th ACM SIGPLAN/SIGOPS International Conference,2018,53(3):31 - 44.
[14] Giunta G, Montella R, Agrillo G, et al. A GPGPU transparent virtualization component for high performance computing clouds[C]//European Conference on Parallel Processing, 2010:379 - 391.
[15] Shi L, Chen H, Sun J, et al. vCUDA: GPU-accelerated high-performance computing in virtual machines[J]. IEEE Transactions on Computers, 2012, 61(6): 804 - 816.
[16] Iserte S, Peña-Ortiz R, Gutierrez-Aguado J, et al. GSaaS: A service to cloudify and schedule GPUs[J]. IEEE Access, 2018,6:39762 - 39774.
[17] Shao J, Ma J, Li Y, et al. GPU Scheduling for short tasks in private cloud[C]//2019 IEEE International Conference on Service-Oriented System Engineering(SOSE),2019.
[18] Phull R, Li C, Rao K, et al. Interference-driven resource management for GPU-based heterogeneous clusters[C]// High Performance Distributed Computing,2012:109 - 120.
[19] Qi Z, Yao J, Zhang C, et al. VGRIS: Virtualized GPU resource isolation and scheduling in cloud gaming[J]. ACM Transactions on Architecture and Code Optimization,2014, 11(2):1 - 25.
[20] Li H, Ota K, Dong M, et al. Multimedia processing pricing strategy in GPU-accelerated cloud computing[J]. IEEE Transactions on Cloud Computing,2020,8(4):1264 - 1273.
[21] Ba J, Caruana R. Do deep nets really need to be deep[C]//Neural Information Processing Systems, 2014:2654 - 2662.
[22] Iandola F N, Han S, Moskewicz M W, et al. SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size[EB]. arXiv:1602.07360,2016.
[23] Luo P, Zhu Z, Liu Z, et al. Face model compression by distilling knowledge from neurons[C]//30th AAAI Conference on Artificial Intelligence,2016: 3560 - 3566.
[24] Strom N. Scalable distributed DNN training using commodity GPU cloud computing[C]//Conference of the International Speech Communication Association, 2015:1488 - 1492.
[25] Chen K, Huo Q. Scalable training of deep learning machines by incremental block training with intra-block parallel optimization and blockwise model-update filtering[C]//2016 IEEE International Conference on Acoustics, Speech and Signal Processing(ICASSP),2016:5880 - 5884.
[26] Iandola F, Moskewicz M W, Ashraf K, et al. FireCaffe: near-linear acceleration of deep neural network training on compute clusters[C]//Computer Vision and Pattern Recognition,2016:2592 - 2600.

(上接第 72 页)

[12] 黄奇峰,杨世海,邓欣宇,等.基于欠完备自编码器的用户用电行为分类分析方法[J].电力工程技术,2019,38(6): 24 - 30.
[13] 杨学良,陶晓峰,熊霞,等.基于深度森林算法的窃电行为检测方法研究[J].智慧电力,2019,47(10):85 - 92.
[14] Zhou Z H, Feng J. Deep forest: Towards an alternative to deep neural networks[C]//Proceedings of the 26 International Joint Conference on Artificial Intelligence,2017:3553 - 3559.
[15] Cieslak D, Hoens T, Chawla N, et al. Hellinger distance decision trees are robust and skew-insensitive[J]. Data Mining and Knowledge Discovery,2011,24(1): 136 - 158.
[16] Hinton G E, Osindero S, Teh Y W. A fast learning algorithm for deep belief nets[J]. Neural Computation,2006,18 (7): 1527 - 1554.
[17] Cieslak D, Chawla N. Learning decision trees for unbalanced data[C]//Proceedings of 2008 European Conference on Machine Learning and Knowledge Discovery in Databases. Springer,2008: 241 - 256.
[18] 沈智勇,苏翀,周扬,等.一种面向非均衡分类的随机森林算法[J].计算机与现代化,2018(12):56 - 60.
[19] 苏翀,任瞳,王国品,等.利用决策树建立慢性阻塞性肺病中医诊断模型[J].计算机工程与应用,2019,55(3),225 - 230.
[20] Vincent P, Larochelle H, Bengio Y, et al. Extracting and composing robust features with denoising autoencoders[C]// Proceedings of the 25th International Conference on Machine Learning,2008:1096 - 1103.