Cambricon MLU100 性能分析

摘要

近年来，特定领域硬件为深度学习（DL）带来了显著的性能提升。许多常用的优化技术，如数据并行化、模型并行化、数据管道、权重剪枝和量化等，已被提出来加速 DL 工作负载的推理阶段。然而，目前仍缺乏对这些优化技术的比较，以显示它们在专用加速器上的性能差异。本文在商用加速器（即 Cambricon MLU100）上评估了这些常用的优化技术。考虑到 DL 性质对准确性的要求，我们的衡量标准不仅测量推理吞吐量，还具有准确性约束。基于我们的分析方法和性能数据，我们得出了一些重要的结论和启示，这些结论和启示对未来的 DL 硬件和软件协同设计很有价值。此外，我们还探讨了标准 ResNet-50 模型和 CIFAR-10 数据集下 MLU100 推断性能的上限。

关键词：深度学习；性能分析；特定领域硬件

1 引言

深度学习（DL）给图像识别 [14], [23] 和自然语言处理 [28], [29] 等许多人工智能挑战领域带来了革命性的变化。然而，深度神经网络（DNN）引起的大量数值运算和参数给通用处理器带来了巨大的挑战。为了跟上现代 DL 工作负载不断增长的计算需求，硬件专用化已成为一种流行的方式 [3] [5] [15] [21] [25]。因此，许多专用加速器因其性能效率而越来越受欢迎。它们已被部署在边缘设备、服务器和数据中心中。例如，华为 Meta10 和 P20 手机集成了 Cambricon-1A DL 处理器内核 [27]。Cambricon 发布了 MLU100 [1]，这是一款部署在数据中心的定制 ASIC，用于加速 morden DL 工作负载的推理阶段。同样，谷歌也提出了用于加速分布式机器学习的张量处理单元（Tensor Processing Unit） [20]。

同时，有许多常用的优化技术可以加速现代 DL 工作负载的推理阶段。这些优化技术包括但不限于数据并行、模型并行、数据管道、权重剪枝和量化 [9,10,17,31]。这些优化技术之间的性能差异给未来的数据链路软硬件协同设计带来了挑战。在专用数字线路加速器上设计或选择适当的优化技术既重要又不容易。此外，目前还缺乏对这些优化技术的比较，以显示它们在专用加速器上的性能差异。在本文中，我们在商用 DL 加速器–Cambricon MLU100 上评估了这些常用的优化技术。为了系统地评估该平台，我们将这些常用的优化技术作为超参数进行扫描。我们以标准 ResNet-50 [13] 模型和 CIFAR-10 [22] 数据集为基准，这些数据集由 BenchCouncil 2019 国际人工智能系统和算法挑战赛（Cambricon Track1）提供。我们的工作负载来自 AIBench [6,7]，这是一个针对数据中心的人工智能基准。根据我们的分析方法和绩效数据，我们得出以下结论和影响：

- 数据管道和数据并行在保持合格精度的同时大大缩短了推理时间。

- 与数据并行相比，模型并行对端到端推理吞吐量的影响并不明显。
- 权重剪枝虽然可以加快推理速度，但会导致精度下降。
- 高硬件吞吐量并不意味着高端到端吞吐量。

我们的观察和启示应有助于其他研究人员和从业人员更好地设计未来的 DL 硬件和软件。此外，我们还探索了 MLU100 在标准基准下的推理性能上限，在保持目标精度的同时，推理时间达到了 384 毫秒。

2 相关工作

2.1 硬件特性

Cambricon MLU100 [1] 是一种部署在数据中心的 DL 加速器，用于加速现代 DL 工作负载的推理阶段。其 ISA 基于 Cambricon [25]。MLU100 的总体架构如 1 所示。Cambricon MLU100 基于多核架构。它包括通过片上网络（NOC）连接的四个通道。每个通道包含一个 DDR 和八个计算核心。例如，通道 0 包含一个 DDR 内存控制器（DDR0）和八个计算核心，即 C0、C1、.....、C7。DDR 负责存储 DNN 模型、DL 工作负载的输入和输出。而这些计算核心则执行 DNN 计算任务。

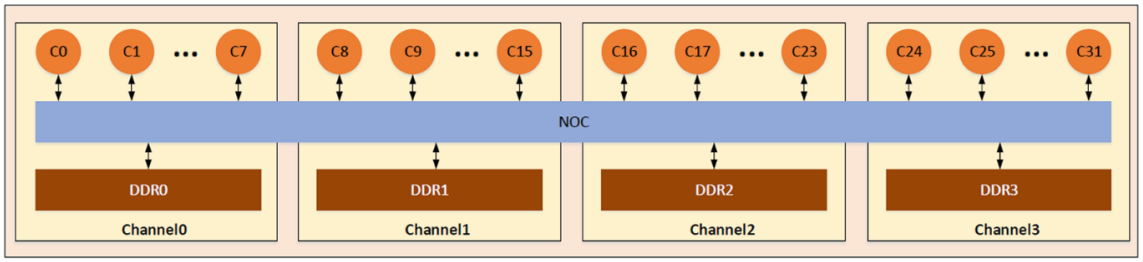


图 1. Cambricon MLU100 的架构信息

2.2 软件栈

图 2 显示了 Cambricon MLU100 的软件栈。众所周知，Caffe [18] 是一个用于 DL 训练和推理的开源软件框架。它由 C++ 编写，在研究实验和行业部署中被广泛采用。Cambricon MLU100 提供 Caffe 作为其高级编程框架。应用程序编程人员只需通过 Cambricon Caffe 就能部署自己的应用程序。CNRT 是 Cambricon MLU100 的运行工具包。它提供一些常用的底层实用程序编程接口，如设备和内存管理、内核启动、任务队列调度等。CNML 是 CNRT 的封装器。它为 DNN 模型的加载和执行提供了一些辅助函数，并为 MLU100 提供了常用的高调 DNN 运算符，如卷积和池化运算符。驱动程序和内核负责处理 MLU100 的内存管理和中断。

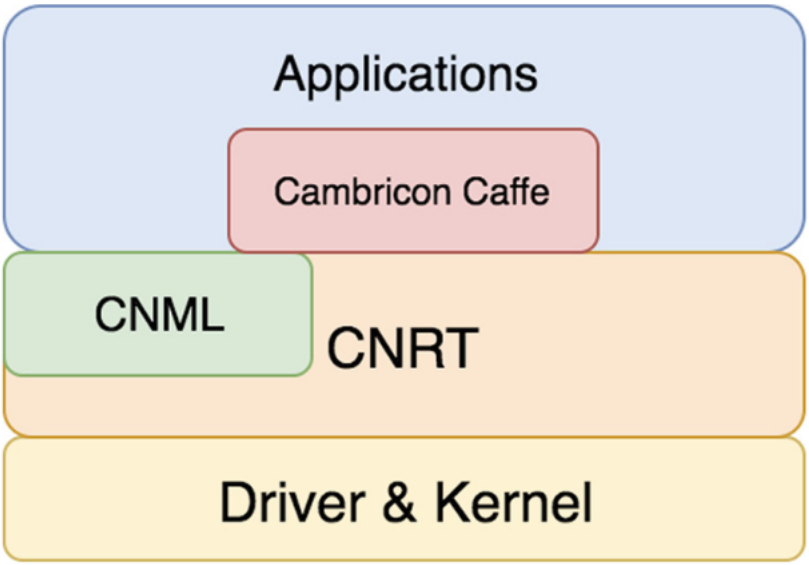


图 2. Cambricon MLU100 软件栈

2.3 优化技术

Cambricon Caffe 提供了一些常用的优化工具，如数据并行、模型并行和数据管道。在推理阶段，这些优化技术通常能提高 DNN 模型的执行性能并保持最终的 top-1 精度。此外，Cambricon Caffe 还支持权重剪枝和量化。这些优化技术通常会对最终的 top-1 精度产生副作用，但却能提高吞吐量。所有这些优化技术并不相互排斥。

数据并行：在 DL 工作负载的推理阶段，数据并行意味着给定一个 CNN 模型，将输入数据分割并分配给不同的计算核心。如图 3a 所示，不同的内核拥有 DNN 模型的完整副本。每个内核只需获取输入数据的不同部分，然后以某种方式将每个内核的结果合并，得到最终输出。数据并行可以大大提高整个过程的效率，因为输入数据的不同部分可以同时执行。

模型并行：如图 3 b 所示，模型并行是指不同的内核负责单个网络中不同部分的计算。例如，神经网络中的每一层可以分配给不同的内核。在 DL 领域，我们可以利用模型并行性，将神经网络划分为多个子网，然后将每个子网分配给 MLU100 的不同内核。模型并行还能提高整个过程的效率，因为对于单个输入，DNN 模型的不同部分可以同时执行。

数据管道：在 DL 工作负载的推理阶段，输入数据流将从磁盘获取到 CPU 的主机内存，然后上载到 MLU 的设备内存。最后，它们将被送入 MLU 的计算核心。在这种情况下，数据管道可以改善数据预取、传输和输入的工作量平衡（图 4）。

权重剪枝和量化：由于在 DL 工作负载的推理阶段，大量的突触权重会导致密集的计算和内存访问，因此研究人员提出了许多有效的技术来探索 DNN 的稀疏性，包括权重剪枝、模型压缩和量化 [9,10,17,31]。Cambricon MLU100 尝试利用 DNN 模型的稀疏性和不规则性来提高性能和能效。它通过设置输入 DNN 模型权重的稀疏性来提供权重剪枝工具。此外，它还提供了将 DNN 模型的权重量化为低精度定点数（如 INT8）的工具。

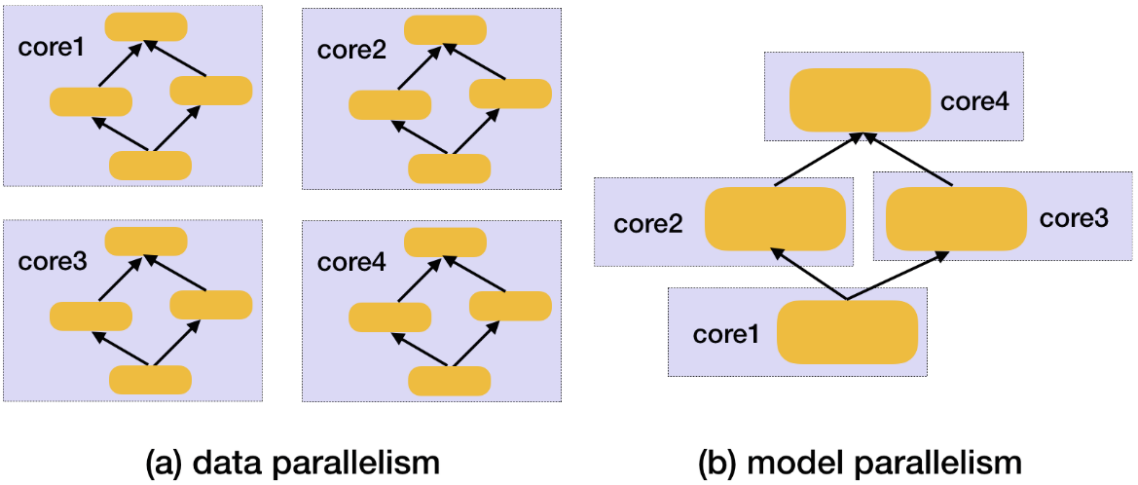


图 3. 模型并行与数据并行的阐述

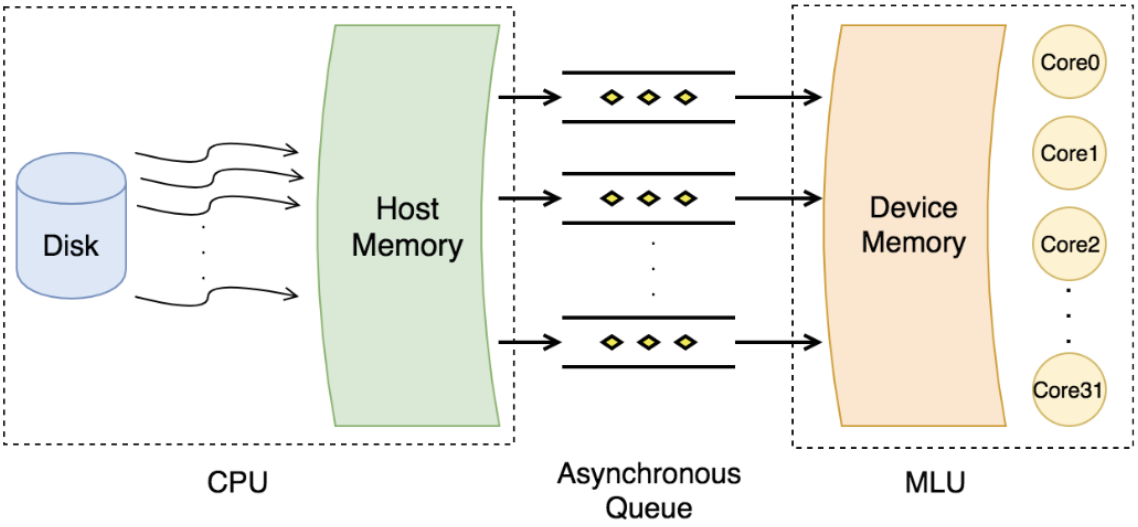


图 4. 数据管道示意图

3 本文方法

3.1 离线模式

Cambricon 加速器为用户提供两个编程库，即 CNML（Cambricon Neuware Machine Learning）库和 CNRT（Cambricon Neuware Runtime）库。这两个库允许开发人员利用用户友好的编程框架和界面，利用 Cambricon 板中的机器学习加速引擎。这两个库支持在线和离线方式在 Cambricon 加速器上部署机器学习算法。在线方法是部署机器学习模型的传统方法。模型由 CNML 框架或其他通用机器学习框架（如 Caffe 和 TensorFlow）管理。离线方法通常会将编译好的计算图和算子序列化到一个新文件中，生成一个新的模型文件（离线模型），CNRT 库可以直接加载该文件。在线或离线方法决定了系统在执行机器学习应用时是否加载机器学习框架。离线方法使机器学习模型绕过了机器学习框架，在端到端推理场景中实现了比传统方法更高的性能。

3.2 多线程

Cambricon MLU100 提供 n 个机器学习加速核心和 m 个内存控制器（通道）（n mod m = 0），每个通道管理一个外部 DRAM，并在操作系统内核中创建一个硬件队列（HQ）。对于单线程程序，n 个加速内核从 1 个外部 DRAM 访问数据，这将导致带宽争用，降低加速器性能。为了利用内存控制器的并行性，我们创建了一个 m 线程程序，并将每个线程绑定到不同的通道（操作系统中的硬件队列），以最大限度地提高加速内核与外部 DRAM 之间的数据访问带宽。在 MLU100 上利用多个加速内核有两种方法，即模型并行和数据并行。在模型并行方法中，模型被划分为 n 个内核，n 个内核将协同处理 1 幅图像。在数据并行方法中，允许每个内核访问整个模型，n 个内核将并发、独立地处理 n 幅图像。由于中间结果的复制和融合，模型并行会带来额外的开销。为了利用多核并行性，我们将模型并行性设为 1，数据并行性设为 n（核数）。

3.3 数据结构转换

为了充分了解程序的瓶颈，我们收集了程序中每个步骤的运行时间，运行时间明细如图 8 (a) 所示。利用上述两种优化方法，数据加载、模型处理和结果写入文件分别只花费了总处理时间的 6.06%、25.74% 和 0.81%，而图像数据类型转换则花费了总时间的 67.39%。数据结构转换开销是由不匹配问题造成的。加载的图像是按照 CIFAR-10 数据集的数据结构格式化的，每幅图像有 1 个字节的标签数据和 3,072 个字节的像素数据。然而，MLU100 将图像数据的连续地址空间作为输入，每个像素用 float32 值表示。

4 复现细节

4.1 与已有开源代码对比

原文没有提供源代码，且没有相似开源代码。

4.2 实验环境搭建

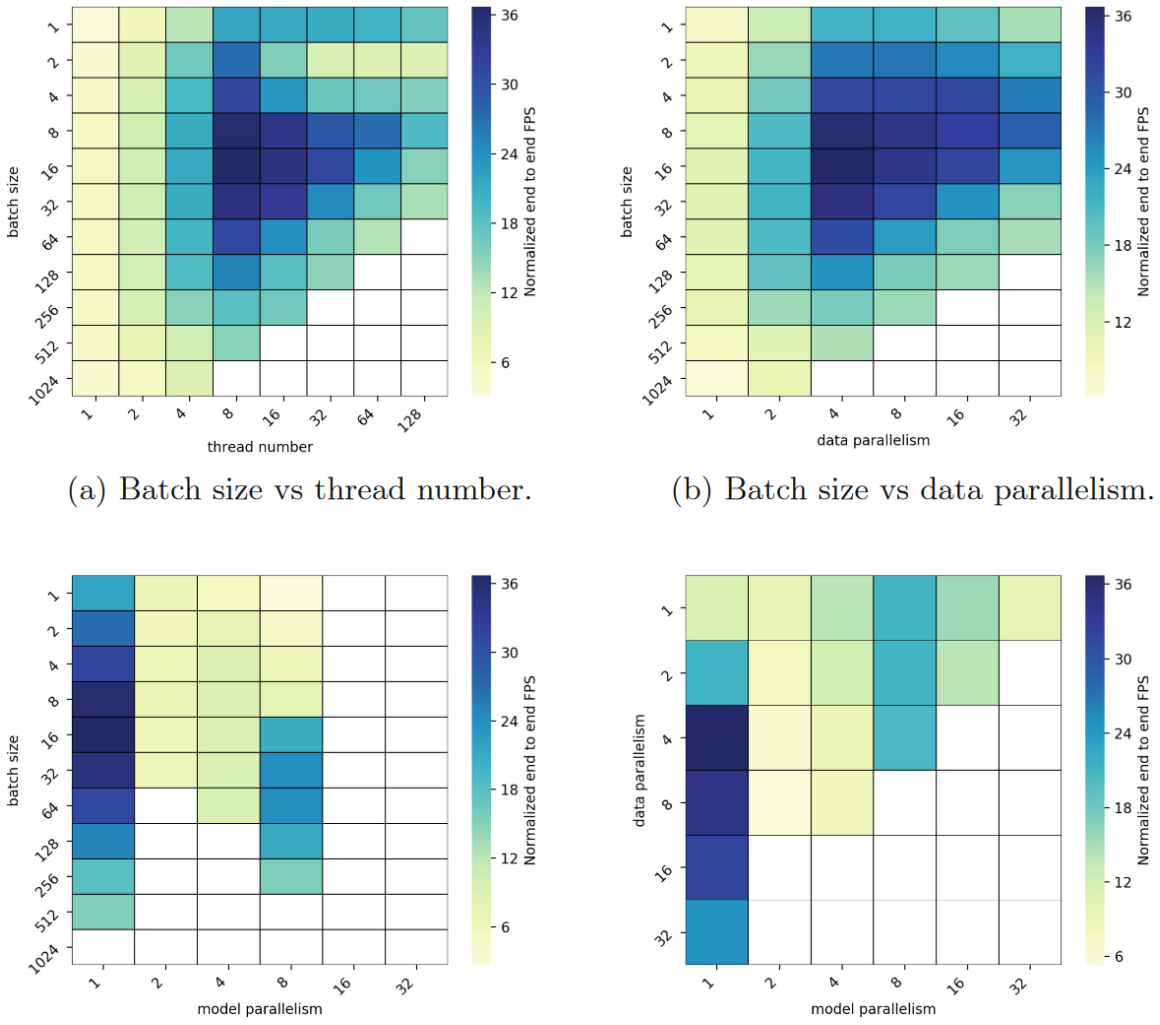
实验在异构环境中运行。主机 CPU 为 2.10 GHz Intel(R) Xeon(R) CPU E5-2620 v4，每个插槽有 16 个内核/32 个线程和 20 MB 三级缓存以及 128 GB 内存，运行 Ubuntu 16.04.10 LTS 和 GCC 5.4.0。设备加速器为 Cambricon MLU100 [1] 。Cambricon MLU100 的设备内存为 8GB，带宽为 102.4GB/s。MLU100 的峰值性能为 16 TFLOPS。

表 1 总结了本文所选的超参数及其使用方法。Cambricon Caffe 提供了为推理任务设置数据并行性和模型平行性的工具。线程数是指从磁盘读取数据时要启动的线程数。批量大小是一个超参数，用于定义推理任务当前迭代中要加载的图像样本数量。

我们评估了 Cambricon MLU100 在标准 ResNet50 [13] 模型和 CIFAR-10 [22] 数据集下的性能。本文中的编程框架是 Cambricon Caffe。对于所有工作负载，我们运行 20 次并计算平均值。

表 1. 本文所选超参数的范围

Variable	Batch size	Data parallelism	Model parallelism	Thread number
Min	1	1	1	1
Max	1024	32	32	128
Inc	*2	*2	*2	*2



(a) Batch size vs thread number.

(b) Batch size vs data parallelism.

(c) Batch size vs model parallelism.

(d) Data parallelism vs model parallelism.

图 5. 批量大小、线程数量、数据并行性和模型并行性对执行性能的影响。注：空白处表示 NaN。FPS 值归一化为端到端情况下的加速比，其中批量大小、数据并行性、模型并行性和线程数分别为 1、1、1、1。

4.3 创新点

在复现原文实验的基础上，发现尽管一些硬件加速器在微基准测试中具有高能效，但在加速器上执行端到端应用时，仍有大量潜力处于性能上限之下。为了探索 mlu 在特定端到端推理场景下的性能上限，本文采用了三种优化方法，同时减少内存拷贝开销，并通过更好地利用并行性来提高速度。

5 实验结果分析

离线部署方法的性能如图 6 所示。与在线部署方法相比，离线方法平均快 1.58 倍，在最佳浴槽大小（64）上快 1.52 倍。

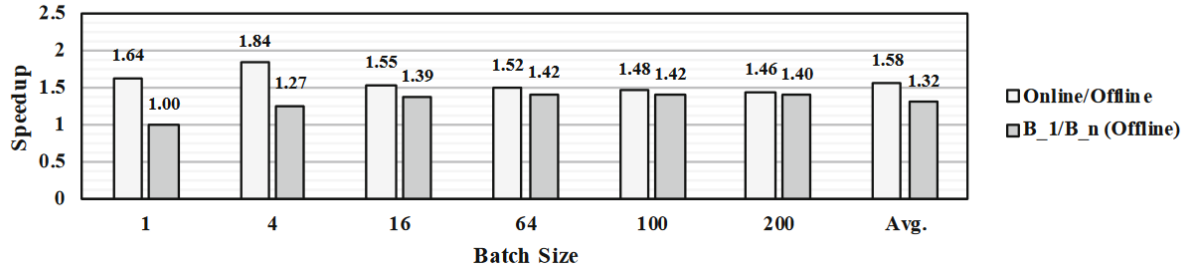


图 6. 离线模型优化法比在线优化法的速度，以及离线模型法中批量大小 = n 比批量大小 = 1 的速度

如图 7 所示，利用 MLU100 的并行性后，与单线程方法相比，并行编程方法平均快 11.83 倍，在最佳浴槽大小（64）上快 12.17 倍。

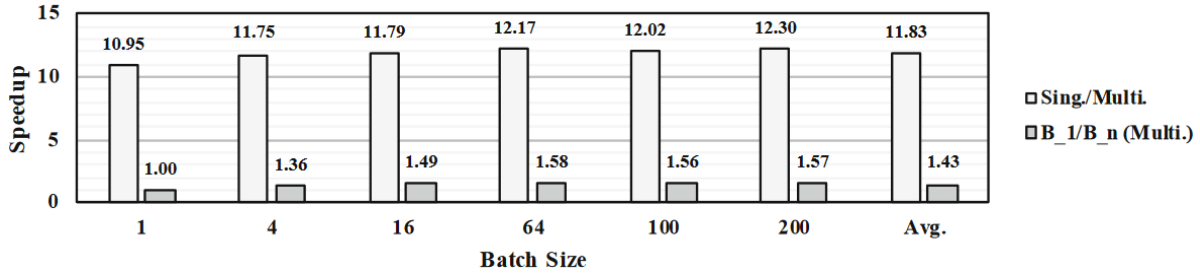
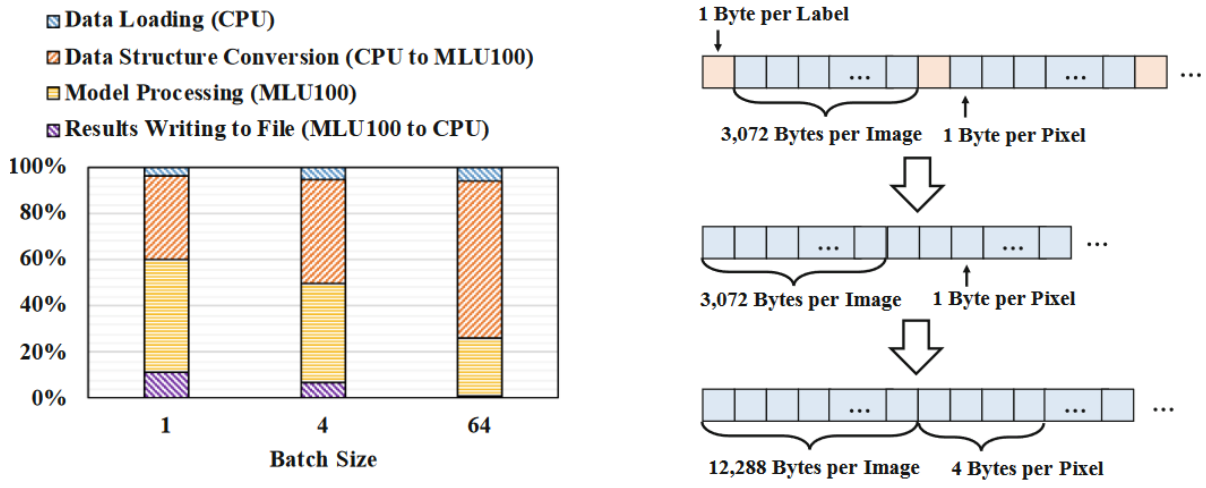


图 7. 多线程优化法比单线程优化法的速度提升，以及多线程优化法中批量大小 = n 比批量大小 = 1 的速度提升



(a) Running time breakdown　　(b) Data structure conversion

图 8. 运行时间明细和数据结构转换

如图 8 所示，为了充分了解程序的瓶颈，我们收集了程序中每个步骤的运行时间，运行时间明细如图 4（a）所示。利用上述两种优化方法，数据加载、模型处理和结果写入文件分别只花费了总处理时间的 6.06%、25.74% 和 0.81%，而图像数据类型转换则花费了总时间的 67.39%。数据结构转换开销是由不匹配问题造成的。加载的图像是按照 CIFAR-10 数据集的数据结构格式化的，每幅图像有 1 个字节的标签数据和 3,072 个字节的像素数据。然而，MLU100 将图像数据的连续地址空间作为输入，每个像素用 float32 值表示。数据类型转换过程如图 8（b）所示。为了减少数据类型不匹配造成的额外处理时间，我们根据 MLU100 的输入数据结构重新格式化图像数据，其性能如图 9 所示。消除数据类型不匹配问题后，性能平均提高了 1.60 倍，在最佳浴槽大小（64）上提高了 1.73 倍。

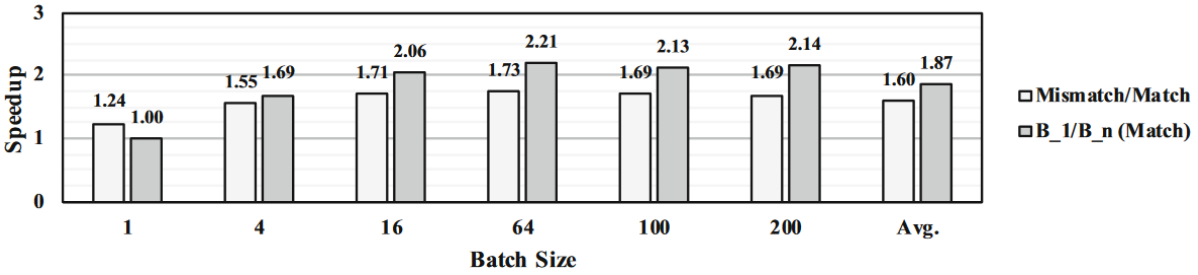


图 9. 运行时间明细和数据结构转换

本文研究了许多常用的优化技术，包括但不限于数据并行、模型并行、数据流水线、权重剪枝和量化。并对这些优化技术进行比较，以显示它们在 Cambricon MLU100 上的吞吐量和最高精度方面的性能差异。基于我们的分析方法和性能数据，我们总结了我们的观察结果和影响，这将有助于改进未来的 DL 硬件和软件协同设计。作为未来的工作，我们计划在更多 DL 加速器上评估更多 DNN 模型。

6 总结与展望

本文的优化方案可以作为未来研究的指导，比如 DNN 任务调度，模型服务的最佳效益配置等。

参考文献

[1] Cambricon: Cambricon MLU100. http://www.cambricon.com/index.php?c=page& id=20

[2] Chen, M., Chen, T., Chen, Q.: HYGON-ALSWR: an efficient implementation of the ALS-WR algorithm on hygon x86 CPUS. In: Gao, W., et al. (eds.) International Symposium on Benchmarking, Measuring and Optimization (Bench 2019). LNCS, vol. 12093, pp. 116–122. Springer, Heidelberg (2020)

[3] Chen, T., et al.: Diannao: a small-footprint high-throughput accelerator for ubiquitous machine-learning. In: ACM Sigplan Notices. vol. 49, pp. 269–284. ACM (2014)

[4] Deng, W., Wang, P., Wang, J., Li, C., Guo, M.: PSL: exploiting parallelism, sparsity and locality to accelerate matrix factorization on x86 platforms. In: Gao, W., et al. (eds.) International Symposium on Benchmarking, Measuring and Optimization (Bench 2019). LNCS, vol. 12093, pp. 101–109. Springer, Heidelberg (2020)

[5] Du, Z., et al.: Shidiannao: shifting vision processing closer to the sensor. In: ACM SIGARCH Computer Architecture News, vol. 43, pp. 92–104. ACM (2015)

[6] Gao, W., et al.: AIBench: towards scalable and comprehensive datacenter AI benchmarking. In: Zheng, C., Zhan, J. (eds.) Bench 2018. LNCS, vol. 11459, pp. 3–9. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-32813-9 1

[7] Gao, W., et al.: AIBench: an industry standard internet service AI benchmark suite. arXiv preprint arXiv:1908.08998 (2019)

[8] Gong, T., Huiqian, N.: An implementation of resnet on the classification of RGB-D images. In: Gao, W., et al. (eds.) International Symposium on Benchmarking, Measuring and Optimization (Bench 2019). LNCS, vol. 12093, pp. 149–155. Springer, Heidelberg (2020)

[9] Han, S., et al.: EIE: efficient inference engine on compressed deep neural network (2016)

[10] Han, S., Mao, H., Dally, W.J.: Deep compression: compressing deep neural networks with pruning, trained quantization and huffman coding (2016)

[11] Hao, T., et al.: Edge AIBench: towards comprehensive end-to-end edge computing benchmarking. In: Zheng, C., Zhan, J. (eds.) Bench 2018. LNCS, vol. 11459, pp. 23–30. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-32813-9 3

[12] Hao, T., Zheng, Z.: The implementation and optimization of matrix decomposition based collaborative filtering task on the hygon x86 platform. In: Gao, W., et al. (eds.) International Symposium on Benchmarking, Measuring and Optimization (Bench 2019). LNCS, vol. 12093, pp. 110–115. Springer, Heidelberg (2020)

[13] He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. CoRR abs/1512.03385 (2015). http://arxiv.org/abs/1512.03385

[14] He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)

[15] Hennessy, J.L., Patterson, D.A.: A new golden age for computer architecture. Commun. ACM 62(2), 48–60 (2019)

[16] Hou, P., Yu, J., Miao, Y., Tai, Y., Wu, Y., Zhao, C.: RVTensor: a light-weight neural network inference framework based on the risc-v architecture. In: Gao, W., et al. (eds.) International Symposium on Benchmarking, Measuring and Optimization (Bench 2019). LNCS, vol. 12093, pp. 85–90. Springer, Heidelberg (2020)

[17] Hubara, I., Courbariaux, M., Soudry, D., El-Yaniv, R., Bengio, Y.: Quantized neural networks: training neural networks with low precision weights and activations. J. Mach. Learn. Res.

[18] http://dl.acm.org/citation.cfm?id= 3122009.3242044 18. Jia, Y., et al.: Caffe: convolutional architecture for fast feature embedding. arXiv preprint arXiv:1408.5093 (2014)

[19] Jiang, Z., et al.: HPC AI500: a benchmark suite for HPC AI systems. In: Zheng, C., Zhan, J. (eds.) Bench 2018. LNCS, vol. 11459, pp. 10–22. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-32813-9 2

[20] Jouppi, N.P., et al.: In-datacenter performance analysis of a tensor processing unit. In: Proceedings of the 44th Annual International Symposium on Computer Architecture, pp. 1–12. ISCA 2017, ACM, New York (2017). https://doi.org/10. 1145/3079856.3080246, https://doi.org/10.1145/3079856.3080246

[21] Jouppi, N.P., et al.: In-datacenter performance analysis of a tensor processing unit. In: 2017 ACM/IEEE 44th Annual International Symposium on Computer Architecture (ISCA), pp. 1–12. IEEE (2017)

[22] Krizhevsky, A.: The CIFAR-10 dataset. http://www.cs.toronto.edu/ kriz/cifar. html

[23] Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems, pp. 1097–1105 (2012)

[24] Li, G., Wang, X., Ma, X., Liu, L., Feng, X.: XDN: towards efficient inference of residual neural networks on cambricon chips. In: Gao, W., et al. (eds.) International Symposium on Benchmarking, Measuring and Optimization (Bench 2019). LNCS, vol. 12093, pp. 51–56. Springer, Heidelberg (2020)

[25] Liu, S., et al.: Cambricon: an instruction set architecture for neural networks. In: ACM SIGARCH Computer Architecture News, vol. 44, pp. 393–405. IEEE Press (2016)

[26] Luo, C., et al.: AIoT bench: towards comprehensive benchmarking mobile and embedded device intelligence. In: Zheng, C., Zhan, J. (eds.) Bench 2018. LNCS, vol. 11459, pp. 31–35. Springer, Cham (2019). https://doi.org/10.1007/978-3-03032813-9 4

[27] Medium: Huawei 7nm Kirin 810 Beats Snapdragon 855 and Kirin 980 on AI Benchmark Test. https://medium.com/syncedreview/huawei-7nm-kirin-810beats-snapdragon-855-and-kirin-980-on-ai-benchmark-test-af31996fb10e

[28] Sutskever, I., Vinyals, O., Le, Q.V.: Sequence to sequence learning with neural networks. In: Advances in Neural Information Processing Systems, pp. 3104–3112 (2014)

[29] Vaswani, A., et al.: Attention is all you need. In: Advances in Neural Information Processing Systems, pp. 5998–6008 (2017)

[30] Wang, Y., Zeng, C., Li, C.: Exploring the performance bound of cambricon accelerator in end-to-end inference scenario. In: Gao, W., et al. (eds.) International Symposium on Benchmarking, Measuring and Optimization (Bench 2019). LNCS, vol. 12093, pp. 67–74. Springer, Heidelberg (2020)

[31] Xiong, X., Wen, X., Huang, C.: Improving RGB-D face recognition via transfer learning from a pretrained 2D network. In: Gao, W., et al. (eds.) International Symposium on Benchmarking, Measuring and Optimization (Bench 2019). LNCS, vol. 12093, pp. 141–148. Springer, Heidelberg (2020) 32. Zhang, S., et al.: Cambricon-x: an accelerator for sparse neural networks. In: The 49th Annual IEEE/ACM International Symposium on Microarchitecture, pp. 20:1–20:12. MICRO-49, IEEE Press, Piscataway (2016). http://dl.acm.org/citation.cfm?id=3195638.3195662