曦云®系列通用计算 GPU

AI 训练用户手册

CSOG-23034-020-F3_V03　　2024-03-15

沐曦专有和三级保密信息　　本文档受 NDA 管控

更新记录

版本	日期	更新说明
V03	2024-03-15	新增曦云®系列 GPU 产品信息
V02	2023-12-29	更新以下章节： 1 概述 3 训练框架的支持 4.1 mcDNN 库 6 分布式训练 7.1 Megatron
V01	2023-10-16	正式版本首次发布

目录

图目录

1. 概述

本文档主要用于指导用户在 AI 训练的场景下，使用曦云®系列 GPU 进行人工智能算法的训练相关事宜。

本文档从 MXMACA®软件栈的维度来介绍曦云系列 GPU 硬件和配套软件提供的功能特性，指导用户达成对应的使用场景。

1.1 MXMACA 软件栈

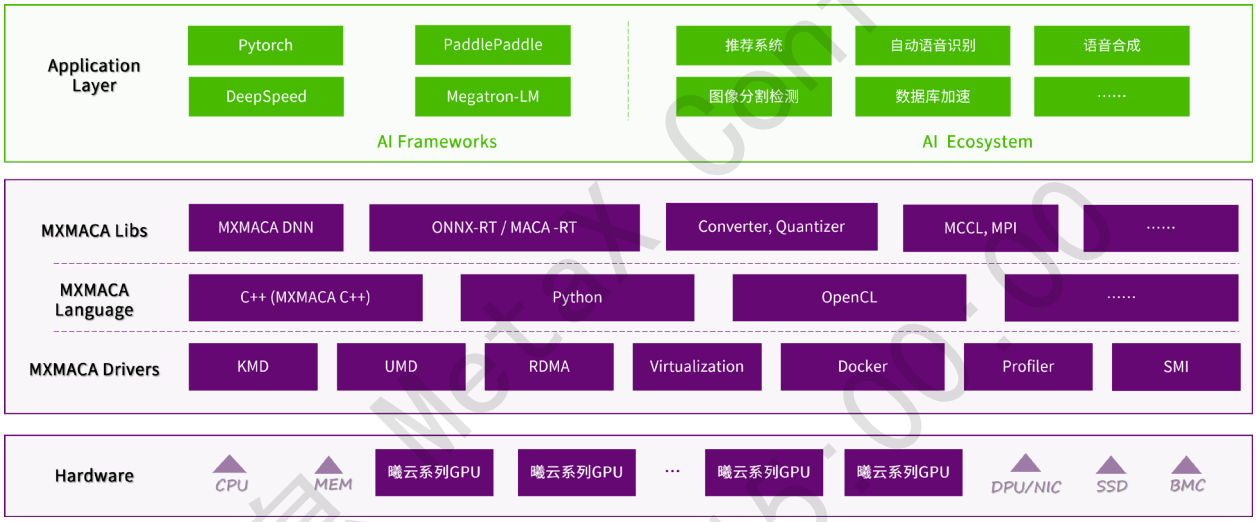

图 1-1 MXMACA 整体架构

在整个 MXMACA 软件栈中，不可或缺的基础设施是硬件底座。硬件底座提供了算力保证，亦是作为特定场景下面的加速器来使用。

执行规则依托于硬件基础设施，并服务于上层应用。首先是 MXMACA 的 Driver 驱动层，对内封装了硬件的基础能力，对外提供了友好的 API 接口；在驱动层的基础上，进一步满足了不同高级语言的使用范式；此外，提供了针对硬件高度优化的函数库。

1.2 AI 训练功能特性

曦云系列 GPU 支持以下 AI 训练功能和特性：

- 支持主流训练框架，如 PyTorch、PaddlePaddle 等
- 提供硬件高度适配的高性能算子库，如 mcDNN 库、mcBLAS 库、mcSolverIT 库等
- 提供友好的 MXMACA 生态用户编程接口
- 支持分布式训练

- 支持大模型训练框架
- 提供完备的工具链
- 支持容器化部署

2. 环境依赖及安装

2.1 环境依赖

在 AI 训练的场景下，使用曦云系列 GPU 之前，必须确保服务器已安装曦云 GPU 板卡以及驱动程序。驱动的安装参见《曦云®系列通用计算 GPU 用户指南》中“安装驱动”章节。

2.2 获取 Docker 容器镜像

MXMACA 软件栈中提供了 Docker 容器镜像，便于用户使用曦云系列 GPU。详细信息参见《曦云®系列通用计算 GPU 用户指南》中“容器相关场景支持”章节。

3. 训练框架的支持

MXMACA 软件栈支持主流的训练框架，包括 PyTorch 和 PaddlePaddle 等。

3.1 PyTorch

PyTorch 是一个开源的 Python 机器学习库，基于 Torch，底层由 C++实现，应用于人工智能领域，如计算机视觉和自然语言处理。

MXMACA 软件栈以 Wrapper 的方式兼容 PyTorch 下的 GPU 代码，从而实现为 PyTorch 支持 MetaX 的硬件后端。适配优化后的 PyTorch 以 PIP 安装包和源码的形式提供。

3.2 PaddlePaddle

PaddlePaddle（飞桨）深度学习框架有大量的官方模型库，支持大规模训练和端到端的部署，详细信息可参见官方网站。

基于 MXMACA 软件对 PaddlePaddle 的适配和优化，使得 MXMACA 程序开发人员可以在曦云系列 GPU 上方便高效地使用 PaddlePaddle。适配优化后的 PaddlePaddle 以 PIP 安装包和源码的形式提供。

4. 高性能库

MXMACA 软件栈提供硬件高度适配的高性能算子库，包括 mcDNN 库、mcBLAS 库、mcSOLVER 库、mcRAND 库、mcFFT 库等，初步涵盖了计算机视觉、人工智能、优化求解、分子动力学等领域。

4.1 mcDNN 库

mcDNN 深度神经网络库是一个 GPU 加速的深度神经网络原语库。mcDNN 为标准例程提供了高度调优的实现，例如前向和后向卷积、池化、规范化和激活层；加速了广泛使用的深度学习框架，包括 Caffe2、Keras、PaddlePaddle 和 PyTorch 等。

mcDNN 库的主要功能特性包括：

- 张量核心加速支持所有流行的卷积操作，包括 2D 卷积、3D 卷积、分组卷积、深度可分离卷积等，支持 channel first 和 channel last 的输入和输出数据排布
- 为计算机视觉和语音模型提供深度优化的内核算子，包括 ResNet、ResNext、EfficientNet、EfficientDet、SSD、MaskRCNN、Unet、VNet、BERT、GPT-2、Tacotron2 和 WaveGlow 等
- 支持 FP32、FP16、BF16 和 TF32 浮点格式和 INT8
- 支持将内存受限的操作（如逐点和简化）与计算受限的操作（如卷积和 matmul）融合在一起

mcDNN 库的详细信息，参见《曦云®系列通用计算 GPU mcDNN API 参考》。

4.2 mcBLAS 库

mcBLAS 库提供了基本线性代数子程序（BLAS）的 GPU 加速实现。通过针对曦云系列 GPU 高度优化的 BLAS API，加速 AI 和 HPC 应用程序。

mcBLAS 库的主要功能特性包括：

- 完全支持所有 152 个标准 BLAS 例程
- 支持半精度和整数矩阵乘法
- 硬件张量核心的 GEMM 和 GEMM 扩展优化
- GEMM 性能调整用于各种深度学习模型的规格和数据类型
- 支持流并发操作

mcBLAS 库的详细信息，参见《曦云®系列通用计算 GPU mcBLAS API 参考》。

4.3 mcSolverIT 库

mcSolverIT 库提供了密集和稀疏的直接线性求解器和特征求解器的集合。基于曦云系列 GPU，为计算机视觉、CFD、ODE、计算化学和线性优化应用提供了显著加速。

mcSolverIT 库支持以下求解器或预条件器：

- 代数多网格（Algebraic Multigrid，AMG）提供经典的 AMG 和聚合 AMG，其中包含 AMG 求解过程的不同选项
- 简单的松弛迭代求解器，例如高斯-赛戴尔（Gauss-Seidel）、雅可比（Jacobi）及其变体
- 多项式迭代求解器，例如切比雪夫多项式（Chebyshev Polynomial）方法
- 克雷洛夫求解器，例如共轭梯度（Conjugate Gradient，CG）、广义最小残差（Generalized Minimum Residual，GMRES）及其变体
- 不完全直接求解器，例如 Cholesky 分解，LU 分解，SVD 分解和 QR 分解等
- 预条件求解器，例如预条件共轭梯度（Preconditioned Conjugate Gradient，PCG）、预条件广义最小残差（Preconditioned Generalized Minimum Residual，PGMRES）等

mcSolverIT 库的详细信息，参见《曦云®系列通用计算 GPU mcSolverIT API 参考》。

4.4 mcRAND 库

mcRAND 库提供高性能 GPU 加速随机数生成（RNG）。mcRAND 库充分利用曦云系列 GPU 的高并发能力，以极致的速度提供高质量的随机数。

mcRAND 库的主要功能特性包括：

- 灵活的使用模式
  - Host API 用于在 GPU 上批量生成随机数
  - 内联实现允许在 GPU 函数/内核中使用，或者在您的 Host 代码中使用
- 高质量的 RNG 算法
  - MRG32k3a
  - MTGP Merseinne Twister
  - XORWOW 伪随机生成
  - Sobol 的准随机数生成器，包括对加密的和 64 位 RNG 的支持
- 多个 RNG 分布选项
  - 均匀分布

- 正态分布
- 对数正态分布
- 单精度或双精度
- 泊松分布

mcRAND 库的详细信息，参见《曦云®系列通用计算 GPU mcRAND API 参考》。

4.5 mcFFT 库

mcFFT 库是基于曦云系列 GPU 的快速傅里叶变换（FFT）的实现，用于构建跨学科的应用程序，如深度学习、计算机视觉、计算物理、分子动力学、量子化学以及地震和医学成像。使用 mcFFT，应用程序自动受益于定期的性能改进和新的 GPU 架构。

mcFFT 库的主要功能特性包括：

- 复数和实数类型的 1D、2D、3D 傅里叶变换算法
- 提供与 Fastest Fourier Transform in the West（FFTW）类似的 API 接口
- 灵活的数据布局允许在单个元素和数组维度之间任意跨越
- 流式异步执行
- 半精度、单精度和双精度变换
- 批处理执行
- In-place 和 out-of-place 转换
- 线程安全，可从多个主机线程调用

mcFFT 库的详细信息，参见《曦云®系列通用计算 GPU mcFFT API 参考》。

4.6 mcThrust 库

Thrust 是一个强大的并行算法和数据结构库，为 GPU 编程提供了一个灵活的高级接口，极大地提高了开发人员的工作效率。使用 Thrust，C++开发人员只需编写几行代码就可以执行 GPU 加速的排序、扫描、转换和约简操作。Thrust 为一些算法和数据结构提供类似 STL 的模板接口，这些算法和数据结构是为高性能异构并行计算而设计的。

为了减少用户迁移的成本，MXMACA 软件栈提供了 mcThrust，针对曦云系列 GPU 架构做了深层次的优化，但是面向用户的 API 接口不变。当用户的应用使用了 Thrust 的 API 接口，MXMACA 软件栈提供了 wrapper 工具来帮助代码的移植。只需要切换到 MXMACA 软件栈的环境变量，相应的编译器工具 mxcc 将会自动参与后续的工作，对用户不感知。

mcThrust 库的详细信息，参见《曦云®系列通用计算 GPU mcThrust API 参考》。

4.7 mcCUB 库

CUB 是一个 C++头文件 only 的函数库，为编程模型的每一层提供了最先进的可重用软件组件，包括并行的原语和一些工具。这些并行原语包括 warp 维度的原语、block 维度的原语和 device 维度的原语。工具集包括智能迭代器、线程和线程块 IO、device，kernel 和存储管理。一般 CUB 会与 Thrust 库配合起来使用。

MXMACA 软件栈提供了 mcCUB，针对曦云系列 GPU 架构做了深层次的优化，但是面向用户的 API 接口不变。当安装了 MXMACA 软件栈的 Toolkit 以后，将会自动拷贝 CUB 和 Thrust 的头文件到系统中，默认是 MXMACA 软件栈路径。不需要单独构建 CUB，要在代码中使用 CUB 原语，只需要获取最新的 CUB 发行版，并包含 CUB 原语的 C++头文件**<cub/cub.cuh>**，或者特定的头文件。然后使用 MXMACA 软件栈的 mxcc 编译器编译您的程序即可。

mcCUB 库的详细信息，参见《曦云®系列通用计算 GPU mcCUB API 参考》。

5. 用户编程接口

在整个 MXMACA 软件栈中，为了满足不同用户和不同场景的诉求，提供了更细粒度、更友好的 MXMACA 软件栈用户编程接口。对于 AI 训练领域，主要会涉及到编译器的接口、运行时接口、集合通信接口以及图像处理和视频编解码等方面的接口。

5.1 编译器接口

mxcc（MXMACA C/C++ Compiler）是 MXMACA 软件栈中针对 MetaX GPU 的硬件架构和功能特性设计和发布的编译器，其底层实现也是在 LLVM 的基础之上，可以支持 C、C++、原生代码以及 MXMACA 软件栈中编程语言、接口和范式，最终生成可以在 MetaX GPU 上运行的高性能的可执行文件。

mxcc 工具的详细介绍，参见《曦云®系列通用计算 GPU mxcc 编译器用户指南》。

5.2 运行时接口

运行时接口规范了基于 MetaX GPU 的编程模型和使用的范式，可以帮助开发人员利用曦云系列 GPU 提供的计算资源，快速构建自己的应用，并在特定的领域中发挥 GPU 加速的最佳性能。

MXMACA 软件栈中运行时接口主要包括设备管理 API，事件管理 API，流管理 API，内存管理 API，模型推理 API，图像处理 API 以及视频处理 API 等。

MXMACA 软件栈中运行时接口的详细介绍，参见《曦云®系列通用计算 GPU 运行时 API 编程指南》。

5.3 集合通信接口

MetaX Collective Communications Library（MCCL，发音为“Mickel”）是一个提供 GPU 间通信原语的通讯库，这些原语具有拓扑感知能力，并且使用方便。MCCL 实现集合通信和点对点发送/接收原语。

MCCL 通信库的详细介绍，参见《曦云®系列通用计算 GPU MCCL 编程指南》。

5.4 图像处理和视频编解码接口

MXMACA 软件栈中，对于数据预处理和视频编解码提供对应的接口和方法，支持主流的 H264、H265、AV1、AVS2 和 JPEG 等格式。此外，基于 FFmpeg 在多媒体领域的广泛应用和用户的开发需求，曦云系列 GPU 支持将 MCRUNTIME 中 VPU 和 G2D 集成到 FFmpeg 中，并选取 FFmpeg4.3 作为集成的基线。

视频编解码库的详细介绍，参见《曦云®系列通用计算 GPU 视频编解码 VPU 编程指南》。

6. 分布式训练

分布式训练主要为了解决单节点设备的算力和内存不足的问题。通常主流的训练框架会提供对应的分布式训练的接口和使用方式。MXMACA 软件栈通过对主流训练框架的支持，提供分布式训练的能力。

此外对于大型和超大型模型来说，其本身的模型特点和超出常规的参数量，使得只能通过分布式的算力基础设施来做训练。对此在 PyTorch 训练框架的基础上，涌现出了类似于 Megatron 和 DeepSpeed 这些框架，降低大语言模型训练的门槛。所有的技术底层都离不开分布式训练的各种并行方式和组合策略。本章简要介绍各种并行的方式。

6.1 数据并行

数据并行的基本使用场景是需要更大的 batch 来训练某个模型，但是过大的 batch 会导致显存的 OOM，因此单机多卡甚至是多机多卡的数据并行训练将是一个很好的方式。大的 batch 会被切分到 mini-batch 供给单卡训练，每个单卡都是一个完整的训练实例，通过对每个 step 的所有训练实例的权重进行聚合操作并求取均值，然后用这个均值来更新单个训练实例的梯度。

6.2 模型并行

随着模型规模和参数量的激增，特别是大语言模型的流行，模型并行将是更好的选择。通过将模型进行拆分，放置在不同的训练卡上以满足训练要求。通常有两种类型的并行：张量并行和流水线并行。

6.2.1 张量并行

张量并行指的是将计算图中层内的参数切分到不同设备，即层内并行，如矩阵-矩阵乘法的拆分。张量并行的关键点在于如何切分模型到不同的设备上和如何保证切分的正确性，因此需要额外的通信开销来保证计算的正确性。例如，对于矩阵乘操作的切分，可以按照左右矩阵的行或列进行切分。不同的切分方式将影响在执行后续操作前是否需要 reduce 操作以及各种通信和计算的开销。

6.2.2 流水线并行

流水线并行指的是按模型 layer 层切分到不同设备，即层间并行。在前向传递过程中，每个设备将中间的激活传递给下一个阶段。在后向传递过程中，每个设备将输入张量的梯度传回给前一个流水线阶段。这允许设备同时进行计算，并增加了训练的吞吐量。流水线并行训练会有一些设备参与计算的冒泡时间，可能导致计算资源的浪费。

6.3 混合并行

通常单一的并行方式并不能得到最优的资源利用率，特别是在超大模型的训练过程中。为了进一步消除计算设备的冒泡时间，会采取混合的并行方式，即数据并行和模型并行协作的方式来提高训练的效率。

以训练 GPT-3 为例，它首先被分为 64 个阶段，进行流水并行。每个阶段都运行在 6 台主机上。在 6 台主机之间，进行的是数据并行训练；每台主机有 8 张 GPU 显卡，同一台机器上的 8 张 GPU 显卡之间是进行模型并行训练。

7. 大模型训练框架的支持

7.1 Megatron

Megatron-LM 是一套基于 PyTorch 训练框架的开源实现库，支持大规模大型 Transformer 语言模型训练。它为基于 Transformer 的预训练语言模型，如 GPT（Decoder Only）、BERT（Encoder Only）和 T5（Encoder-Decoder），提供了高效的张量、流水线和基于序列的模型并行解决方案。即使用混合精度开发了高效、模型并行（张量、序列和管道）和基于 Transformer 的模型多节点预训练。

Megatron-LM 对于大语言模型的训练，充分体现了数据并行和模型并行的混合使用，尽可能减少额外的通信，尽量做到节点间数据并行，节点内张量并行，提高利用率和训练效率。

MXMACA 软件栈以 Wrapper 的方式兼容 Megatron 下的 GPU 代码，从而实现为 Megatron 支持 MetaX 的硬件后端。适配优化后的 Megatron 以源码的形式提供。容器中**/workspace** 文件夹下有适配完成的 Megatron-LM 源码，该环境可以直接上手使用，使用方式和原始的 Megatron-LM 相同。

7.2 DeepSpeed

DeepSpeed 是一款易于使用的深度学习优化软件套件，可为深度学习训练和推理提供前所未有的规模和速度。DeepSpeed-Chat 具有以下三大核心功能：

- 简化 ChatGPT 类型模型的训练和强化推理体验：只需一个脚本即可实现多个训练步骤，包括使用 Huggingface 预训练的模型、使用 DeepSpeed-RLHF 系统运行 InstructGPT 训练的所有三个步骤、甚至生成自己的类 ChatGPT 模型。此外，还提供了一个易于使用的推理 API，用于用户在模型训练后测试对话式交互。
- DeepSpeed-RLHF 模块：DeepSpeed-RLHF 复刻了 InstructGPT 论文中的训练模式，并确保包括监督微调（SFT）、奖励模型微调和基于人类反馈的强化学习（RLHF）在内的三个步骤与其一一对应。此外，还提供了数据抽象和混合功能，以支持用户使用多个不同来源的数据源进行训练。
- DeepSpeed-RLHF 系统：将 DeepSpeed 的训练（training engine）和推理能力（inference engine）整合到一个统一的混合引擎（DeepSpeed Hybrid Engine，DeepSpeed-HE）中用于 RLHF 训练。DeepSpeed-HE 能够在 RLHF 中在推理和训练模式之间无缝切换，使其能够利用来自 DeepSpeed-Inference 的各种优化，如张量并行计算和高性能 MXMACA 算子进行语言生成，同时训练部分还能从 ZeRO-based 和 LoRA-based 内存优化策略中受益。DeepSpeed-HE 还能够自动在 RLHF 的不同阶段进行智能的内存管理和数据缓存。

MXMACA 软件栈以 Wrapper 的方式兼容 DeepSpeed 下的 GPU 代码，从而实现为 DeepSpeed 支持 MetaX 的硬件后端。适配优化后的 DeepSpeed 以 PIP 安装包和源码的形式提供。

8. 容器化安装部署

MXMACA 软件栈提供对 Docker 以及 Kubernetes 的支持。开发人员可以借助于容器引擎，运行预先构建好的 MXMACA 容器镜像，快速建立软件开发或运行所需的环境。开发人员仅需执行一条 `docker run` 命令就可获得一个干净而完整的加速卡开发环境。

在运行 MXMACA 容器镜像前，请确认环境满足以下条件：

- 已正确安装曦云系列 GPU 内核驱动。
- 已安装 Docker，Docker 版本≥18.09。

容器的获取、安装以及具体的使用方式，参见《曦云®系列通用计算 GPU 用户指南》中“容器相关场景支持”章节。

9. 附录

9.1 术语/缩略语

术语/缩略语	全称	说明
CFD	Computational Fluid Dynamics	计算流体力学
Cholesky 分解		Cholesky 分解是把一个对称正定的矩阵表示成一个下三角矩阵 L 和其转置的乘积的分解
LU 分解	Lower–Upper Decomposition	LU 分解是矩阵分解的一种，可以将一个矩阵分解为一个单位下三角矩阵和一个上三角矩阵的乘积
MXMACA	MetaX Advanced Compute Architecture	沐曦推出的 GPU 软件栈，包含了沐曦 GPU 的底层驱动、编译器、数学库及整套软件工具套件
OOM	Out Of Memory	内存耗尽
ODE	Ordinary Differential Equation	常微分方程
QR 分解		正交三角分解，用于将矩阵转换为 A = QR 形式
RNG	Random Number Generation	随机数生成
SVD	Singular Value Decomposition	奇异值分解，是线性代数中一种重要的矩阵分解

声明