人工智能计算设备性能测评报告

二〇二四年八月

目录

1. 引言

1.1 项目背景

随着人工智能应用规模的扩大，全球 AI 芯片产业也迎来了迅速的发展。深度学习模型在各种计算设备上的应用已经显著扩展，涵盖了从高性能服务器到移动设备、嵌入式系统及边缘计算设备等诸多领域。许多企业都在研究和开发 AI 芯片，例如 NVIDIA、AMD、苹果等国外企业，其中 NVIDIA 凭借其出色的产品性能和广泛的应用成为了诸多企业对比竞争的标杆。近年来，国内也有一大批优秀的企业加入 AI 芯片领域，例如寒武纪、天数智芯、华为等。对于这些采用不同硬件架构的 AI 计算设备，如何衡量和评价其性能，成为了亟待解决的问题。

目前，全球范围内尚无完善的 AI 计算设备测评体系，这在一定程度上影响了 AI 计算设备的应用。较为主流的基准测试方法是运行一些常见的神经网络来进行评价，本次评测也采用了这样的方法。

1.2 评测目标

本次评测的主要目标为使用不同的 AI 模型的训练或推理作为测试负载对单卡设备进行性能测试，因此在测试过程中不会涉及到多卡并行。由于大语言模型的训练通常使用多卡并行的方式，所以本次测试并未涉及到大语言模型的训练；同时受限于设备的条件，一些训练比较复杂的模型如 LDM，实验中也没有测试其训练性能。

值得注意的一点是，本次测试主要目的在于对比各个硬件的性能表现，而非评价模型结构的优劣，模型仅作为计算负载来测试硬件对各种 AI 训练或推理任务的胜任程度。深度学习框架也使用了比较通用的 PyTorch 框架，以减少在迁移模型上所耗费的精力，以及手动迁移模型所造成的代码劣化等。

本次评测选用 Qwen2(0.5B)，GLM-4(9B)以及 LLaMA3(8B)三个大语言模型进行单卡推理测试。选择了 LSTM、ResNet、YOLOv10-N、LDM、U-Net、BERT 作为相对小尺寸的神经网络的代表，进行单卡的训练或推理测试，其中 LDM 仅做了推理测试，其余均做了训练和推理测试。

2. 硬件设备与测试环境

2.1 硬件信息简要总览

关于本次实验中用到的硬件，当前在网上可以查阅到的可公开信息如下：

设备名称	内存容量	内存类型	最大带宽	功耗	制程工艺	峰值算力
NVIDIA A100	40GB	HBM2	1,555GB/s	400W	7nm	FP32 19.5 TFLOPS
GeForce RTX 4090	24GB	GDDR6X	1,018 GB/s	450W	4nm	4th Gen 1321 AI TOPS
NVIDIA Jetson Xavier NX	8GB	128-bit LPDDR4x	59.7GB/s	10~20 W	-	INT8 21 TOPS
MLU370-X8	48GB	LPDDR5	614.4 GB/s	250W	7nm	FP32 24 TFLOPS
天数 MR-V100	32GB	HBM2e	-	150W	-	-
天数 BI-150	32GB	HBM2e	-	350W	7nm	FP16 147 TFLOPS

表 1-1 设备标称指标对比

注：测试中使用的寒武纪产品是 MLU370-M8，目前没有它的相关标称指标可以参考，因此选用了与其相近的 MLU370-X8 来做总览和对比。

2.2 设备介绍及测试环境

2.2.1 NVIDIA 设备

NVIDIA A100-SXM4-40GB

A100 基于 NVIDIA Ampere 架构，采用 7nm 工艺制造，集成了 6912 个 CUDA 核心和 432 个第三代 Tensor 核心。它在 FP32 运算中提供 19.5 TFLOPS 的计算能力，而在 Tensor 核心运算中，其性能可达 312 TFLOPS（TF32）。A100 配备了 40GB 的 HBM2 显存，内存带宽达到 1.6TB/s, 是处理大规模深度学习模型和数据集的理想选择, 广泛应用于 AI 训练、推理、科学计算和数据分析领域。在本次实验中 NVIDIA A100 的标称性能应该是最高的。
测试环境的简要信息如下：

操作系统	NVIDIA Driver version	CUDA version	深度学习框架版本
Ubuntu 18.04.6 LTS	535.154.05	V11.2.67	torch2.3.1

表 2-1 A100 系统配置

GeForce RTX 4090

GeForce RTX 4090 基于 NVIDIA Ada Lovelace 架构，采用 5nm 工艺制造，拥有超过 16000 个 CUDA 核心。其 FP32 运算性能超过 40 TFLOPS，在高性能计算任务中表现出色，

特别是在深度学习推理任务中。GeForce RTX 4090 配备 24GB 的 GDDR6X 显存，内存带宽高达 1TB/s，为复杂计算任务提供了强有力的支持，适用于高性能计算、实时渲染和深度学习推理任务。

测试环境的简要信息如下：

操作系统	NVIDIA Driver version	CUDA version	深度学习框架版本
Ubuntu 20.04.6 LTS	535.154.05	V11.2.67	torch2.3.1

表 2-2 GeForce RTX 4090 系统配置

Jetson Xavier NX

Jetson Xavier NX 基于 NVIDIA Volta 架构，集成了 6 个 Carmel CPU 核心和 384 个 CUDA 核心，以及 48 个 Tensor 核心，能够提供 21 TOPS 的 AI 性能。它采用 8GB LPDDR4x 内存，支持 51.2GB/s 的内存带宽，专为边缘计算和嵌入式系统设计，适合在功耗受限的环境中执行 AI 推理任务，如嵌入式设备、机器人、无人机等场景。

测试环境的简要信息如下

操作系统	Jetpack	CUDA version	深度学习框架版本
Ubuntu 20.04.6 LTS	5.1.3	11.4.19	torch2.1.0a0+41361538.nv23.6

表 2-3 Jetson Xavier NX 系统配置

注：这里的 torch 版本是 NVIDIA 官方提供的、针对于 Jetson 系列设备的特供版 PyTorch，可以在 NVIDIA 官网查询到编译安装包。

该设备 Jetson Xavier NX 以下简称 Jetson。

2.2 国产设备

天数智芯 MR-V100 & BI-150

天数智芯 MR-V100 是一款基于自主研发的 DPU（Deep Learning Processing Unit）架构的高性能 AI 计算设备，设计旨在满足深度学习领域日益增长的计算需求。MR-V100 集成了大量的 AI 计算核心，具备卓越的矩阵运算和卷积操作能力，特别适用于大规模神经网络的训练和推理任务。该设备还配备了高带宽内存，支持多通道并行处理，以确保在复杂模型和大数据集的情况下，依然能够提供稳定且高效的计算性能。

天数智芯 BI-150 是 MR-V100 的增强版，采用了升级后的 DPU 架构，进一步提高了计算能力，并优化了能效表现。BI-150 支持更高精度的 AI 计算操作，配备了更大容量的高带宽内存，以适应复杂模型的训练需求和高负载的推理任务。该设备设计专注于多任务并行处理，能够在多个深度学习任务同时进行的情况下，依然保持卓越的性能表现。

MR-V100 的测试环境的简要信息如下

操作系统	Corex	CUDA version	深度学习框架版本
CentOS 7	4.1.0	10.2	torch2.1.1+corex.4.1.0

表 2-4 天数 MR-V100 系统配置

BI-150 的测试环境的简要信息如下

操作系统	Corex	CUDA version	深度学习框架版本
Ubuntu 20.04.6 LTS	4.1.0	10.2	torch2.1.1+corex.4.1.0

表 2-5 天数 BI-150 系统配置

寒武纪 MLU370-M8

寒武纪 MLU370-M8 是一款高性能的 AI 加速卡，基于寒武纪最新的 MLUarch03 计算架构和 7nm 制程工艺，这些技术提升了其运算效率和处理能力。其同系列的 MLU370-X8 加速卡支持多种计算精度，包括 FP32、FP16、BF16、INT16、INT8 和 INT4，其 INT8 的峰值性能可达到 256 TOPS，而 FP32 的性能则为 24 TFLOPS。此外，MLU370-M8 配备了 48GB 的 LPDDR5 内存和 614.4 GB/s 的内存带宽，使其在处理大型数据集时更为高效。

测试环境的简要信息如下：

操作系统	Driver version	深度学习框架版本
Ubuntu 22.04 LTS	v5.10.34	torch-2.0.1+torch_mlu-1.20.0

表 2-6 **寒武纪** MLU370-M8 **系统配置**

其余用到的寒武纪官方提供的 SDK 版本如下：

CNCL	CNCV	CNNL	CNNLextra	Mluops	CNCLbenchmark	CNtoolkit
1.16.0	2.5.0	1.25.2	1.8.1	1.1.1	1.3.0	3.10.2

表 2-7 **寒武纪** SDK **版本**

2.3 硬件设备的比较

2.3.1 应用场景

通过比较 NVIDIA A100、NVIDIA GeForce RTX 4090 和 NVIDIA Jetson NX 这几款设备可以发现，应用场景的不同，影响了硬件的设计思路。

A100 主要设计用于数据中心和高性能计算（HPC）环境，适合进行深度学习训练、科学计算、数据分析等任务。因此它提供了高带宽的内存（HBM2），以及针对 AI 和高性能计算优化的架构，如 Tensor 核心和更高的浮点运算能力。

GeForce RTX 4090 更多面向消费级市场，尤其是游戏和创意内容制作，如 3D 渲染和视频编辑。它也能用于 AI 研究和开发，但主要是在个人或小规模的环境中。它具有极高的图形处理能力，提供的是 GDDR6X 这类高速显存，适合处理大量图形数据。

Jetson Xavier NX 所代表的 Jetson 系列设备主要面向边缘计算应用，如机器人、无人机和其他自动化设备中的 AI 计算。它们体积小巧，功耗低，适合嵌入式系统使用。尽管计算能力不及 A100 和 GeForce RTX 4090，但足以处理边缘设备上的图像识别和处理任务，且不会过多地浪费边缘设备的电能。

本次测试中除 GeForce RTX 4090 的设计应用场景为处理图形渲染，Jetson Xavier NX 的设计应用场景为边缘 AI 计算，其余设备的设计应用场景均为通常情况下的 AI 计算优化。

2.3.2 深度学习支持

针对于通用深度学习框架 PyTorch，不同的设备有不同的后端提供支持。

NVIDIA 的 CUDA 技术提供了强大的后端支持，使得 PyTorch 代码能够高效使用 NVIDIA 计算设备的计算能力，包括对 Tensor 核心的利用。可以大幅加速深度学习中的矩阵运算。CUDA 也拥有当前世界上最成熟的技术栈和开发社区。

天数智芯的两款设备尽管是新兴的产品，但它们正在逐步提高与当今通用的深度学习框架的兼容程度。天数智芯的产品广泛支持 PyTorch、TensorFlow 和 PaddlePaddle 等深度学习框架，能够稳定运行超过 200 种 AI 模型。例如进行模型训练时，开发者可以直接从 DeepSparkHub 模型库下载并运行 PyTorch 模型，如 ResNet50，而无需进行额外的代码修改或适配工作。并且天数智芯的 SDK 库中还提供了对于 cuda 模块的支持，使得环境搭建成功后，许多在 NVIDIA 设备上使用 torch.cuda 编写的代码或模型可以无需修改代码、直接进行移植，提高了天数设备的可扩展性。

寒武纪对 PyTorch 的支持主要通过其扩展包 CATCH（包含库 torch_mlu）实现，该扩展包允许 PyTorch 在 Cambricon Machine Learning Unit（MLU）后端上进行高性能的训练和推理。CATCH 作为一个独立于 PyTorch 的扩展，通过对原始 PyTorch 库添加补丁以提供 MLU 支持，使得 PyTorch 能够利用 MLU 加速神经网络的计算。这包括在急切模式下逐层进行推理，或者使用 TorchScript 接口在融合模式下进行推理。此外，CATCH 还支持单卡或多卡的训练。而在代码的移植过程中，MLU 设备仅需更改 torch 的引用方式（增加“import torch_mlu “），以及设备的名字（”cuda”改为”mlu”）就可以移植到 MLU 设备上进行运行。

需要注意的是，这里的移植代表了一种一般流程。一些高版本 CUDA 算子的适配由于厂家计算研发的滞后性，其支持情况还需要具体情况具体分析。

3. 深度学习模型及项目源码

3.1 模型使用情况及源码来历

实验中共测试了 9 个模型作为计算负载时硬件的性能。其中使用到的 9 个模型分别为：CV 领域的 ResNet、U-Net、YOLOv10-N、LDM，NLP 领域的 BERT、LSTM（用于文本生成），大语言模型 Qwen2、LLaMA3、ChatGLM-4，各模型参数量详情见附录 A。

特别地，大语言模型采用了统一的、依托于 transformers 库的、生成参数完全相同的 Python 脚本进行测试，避免因为代码差异对性能的评测产生影响。

下面逐一简要介绍使用到的 9 个模型以及所做出的适配：

ResNet

原作者的代码仓库为：KaimingHe/deep-residual-networks: Deep Residual Learning for Image Recognition (github.com)。但原作者的代码由 Caffe 进行实现，所以改选用 PyTorch 官网实现的模型代码以及模型权重进行测试 vision/torchvision/models/resnet.py at main · pytorch/vision (github.com)。接着使用 PyTorch 框架实现了模型的训练与推理。由于模型结构中使用的算子版本较低、较为稳定，即使跨硬件平台，在使用 PyTorch 的情况下未出现兼容性问题。

U-Net

https://github.com/milesial/Pytorch-UNet 实验没有采用原论文《U-Net: Convolutional Networks for Biomedical Image Segmentation》所附的 MATLAB 代码，而是使用了依托于 PyTorch 实现的开源项目。原项目在训练过程中会进行验证，以保证更好的模型训练效果，但这一过程会影响计算设备的测试工作，因此我们重写了训练过程的代码，仅保留了源码中的核心训练过程。另外，我们也重写了推理过程的代码，解决了源码缺少批量推理功能的问题。在测试中发现，U-Net 对设备性能要求较高，尝试将模型量化为 INT8 部署至 Jetson 后，依然仅能在 CPU 上以极低的效率进行推理，因此我们放弃了其在 Jetson 的测试。

YOLOv10-N

THU-MIG/yolov10: YOLOv10: Real-Time End-to-End Object Detection (github.com) 由于源代码封装的相当完备，不便于进行测试工作，因此删除了训练和推理过程中冗余的检查操作和联网请求，以及训练过程中的验证部分。

LDM

CompVis/latent-diffusion: High-Resolution Image Synthesis with Latent Diffusion Models (github.com) 在测试 LDM 的推理能力的过程中，对 LDM 的源码做了一些针对于高版本 PyTorch 的适配。所做的修改基本上是对 torch 等库中函数路径的重新链接，对模型性能的影响不大。实验中使用了文生图功能来做测试。

BERT

Google 发布的原版 BERT 是使用 TensorFlow 实现的 google-research/bert: TensorFlow code and pre-trained models for BERT (github.com)，与拟选择的 PyTorch 框架不符，所以选择了使用 transformers 库中封装的 BERT。模型权重与 tokenizer 来自 HuggingFace 上的 Google 原版 BERT 仓库。在跨平台运行时出现过算子不支持的情况，通过修改 transformers 库文件中部分函数的实现得到了解决。由于修改方式是将 GPU 上不支持的计算，转移到 CPU 上进行，因此会造成一定的性能损失。

LSTM

zhangzibin/char-rnn-chinese: Multi-layer Recurrent Neural Networks (LSTM, GRU, RNN) for character-level language models in Torch. Based on code of https://github.com/karpathy/char-rnn. Support Chinese and other things. 该项目中实现了一种可以用于中文文本生成的 LSTM，项目源码使用的是 Lua 语言。实验中使用的 LSTM 基于该项目构建，参照它做了一些仿写和适配。

Qwen2

模型权重来自魔搭社区：组织详情 · 魔搭社区 (modelscope.cn)。
测试程序使用 PyTorch 搭配 transformers 库实现。在跨平台运行时出现过算子不支持的情况，通过修改 transformers 库文件中部分函数的实现得到了解决。由于修改方式是将 GPU 上不支持的计算，转移到 CPU 上进行，因此会造成一定的性能损失。

LLaMA3

模型权重来自 meta-llama/llama3: The official Meta Llama 3 GitHub site。
在使用统一的脚本对 LLaMA3 模型进行测试时，对代码做出了一些调整：模型的权重参数需要使用 transformers 库中 LlamaForCausalLM 类而不是 AutoModel 类进行加载。在 GeForce RTX 4090 上，由于其显存仅 24G，而正常情况下 LLaMA3 使用全精度 FP32 运行需占用约 30G，因此在该设备上将其精度量化为 FP16 才能运行。此外，在天数智芯 MR-V100，BI-150 上，也需将模型量化为 FP16 才能运行。

GLM-4

模型权重来自 THUDM/glm-4-9b-chat · Hugging Face。
注意 GLM-4 必须在 Python3.10 及以上版本才能正常运行。在寒武纪 MLU370-M8 设备上运行 GLM-4 时做了 FP16 量化，之后遇到“@torch.jit.script“跨平台修饰器不能够正确识别设备的问题，将其删除即可解决，但是造成了一定程度的性能损失；在天数智芯 BI-150 上运行时遇到了库内部函数返回值数目不统一的问题，强制忽略了一个返回值，结果大语言模型仍然能正常对话，可能隐含地会对模型的正确性造成细微的影响。

3.2 评测方式

我们并没有使用成型的性能分析工具，比如 perf，Nsight 等。一方面在各个设备上能够使用的性能分析工具并不统一，不同的分析工具有不一样的安装需要，又会得到结构各异的性能数据；另一方面，性能分析工具得到的信息非常丰富，而在本次评测中使用的只是其中的一部分，大量冗余的信息无疑又增加了测试之后处理数据的难度。

鉴于此，我们开发了一个检测 GPU 执行情况的类 GPUMonitor，在初始化的同时启动一个线程，通过设备给予的 Api（例如 NVIDIA 的 pynvml、jtop，寒武纪的 cndev，天数的 ixsmi 等），每隔一定时间读取 GPU 的信息，并结合当前的时间戳，把结果保存成时间序列，方便后续的数据处理。

使用 Api 可以直接得到需要的数据，减少冗余信息，最大化获取信息的专用性、有效性；同时使用多线程启动 Monitor 能够最大程度上不影响模型运行速度。

我们把实验过程中所构建的所有 Monitor 类的源码上传了到 github 中，可在如下链接中查看源码。tuo3288/Monitors: GPU monitors (github.com)

4. 性能评测结果

4.1 结果综述

针对于每个设备，综合得到的、有单调性的四个指标，我们设计了一个评分公式，以此来体现设备在以各个模型为负载时的综合性能。

评分的具体计算公式如下：

$$\widetilde{x^{(i)}} = -\alpha + 2\alpha * \left(\frac{x^{(i)} - x_{min}^{(i)}}{x_{max}^{(i)} - x_{min}^{(i)}} \right) \tag{4.1}$$

$$score_{model} = \sum_{i} k_i * sigmoid\left(\pm \widetilde{x^{(i)}} \right) \tag{4.2}$$

其中第一个公式是将测得的数值、单位各异的指标映射到[-α,α]区间上，避免不同设备之间差距过于悬殊，其中$x^{(i)}$表示在指定某一个模型上的第i种指标，$x_{min}^{(i)}$表示该指标在不同硬件之间的最小值，$x_{max}^{(i)}$表示该指标在不同硬件之间的最大值；第二个公式是求得评分，其中$score_{model}$是在运行指定模型时，该硬件的综合评分，k_i是根据不同指标的重要程度人为设定的一组系数。在下文综合评分对比图的计算中，α暂定为 3，指标的权重k_i暂定为平均分配。

根据每个硬件在所有模型上的综合评分，画出了代表每个设备能力的雷达图，作为测试结果的展示。结果如下：

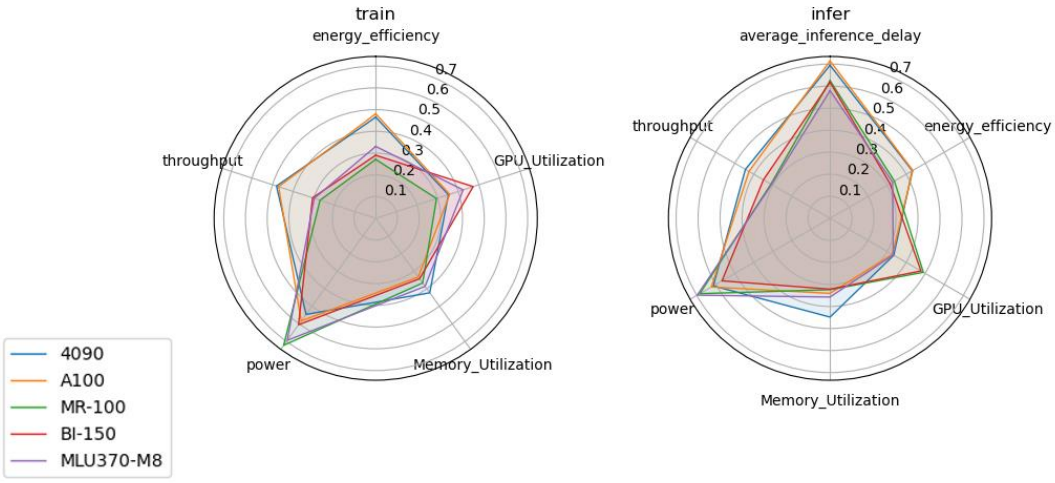

图 4-1 训练和推理部分硬件综合评分图

4.2 详细结果比较

此处使用了归一化的指标，仅作为相对关系的参考，其数值大小与相应设备性能指标呈正相关。归一化的方式为：求出各指标在某个模型的训练和推理任务中出现的最大值，其余值转化为占最大值的比例。可查看附录中的测试数据详表获知各指标的具体数值。

4.2.1 CV 任务

CV 任务训练结果如下：

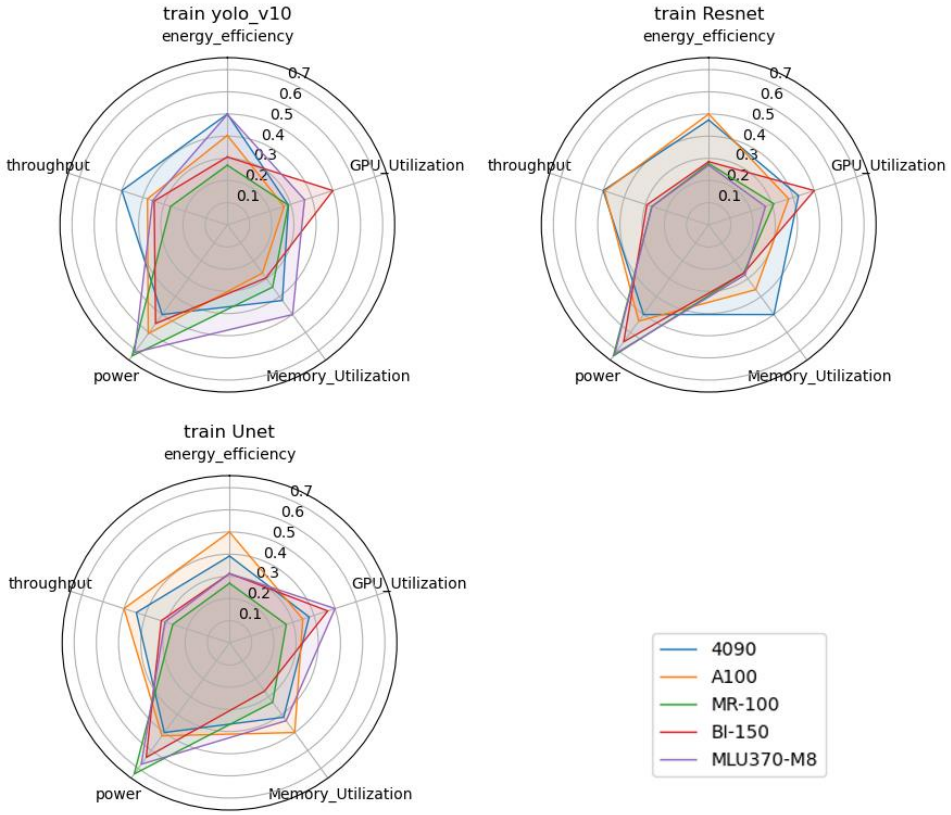

图 4-2 CV 模型训练过程各硬件指标对比

CV 任务推理测试结果如下:

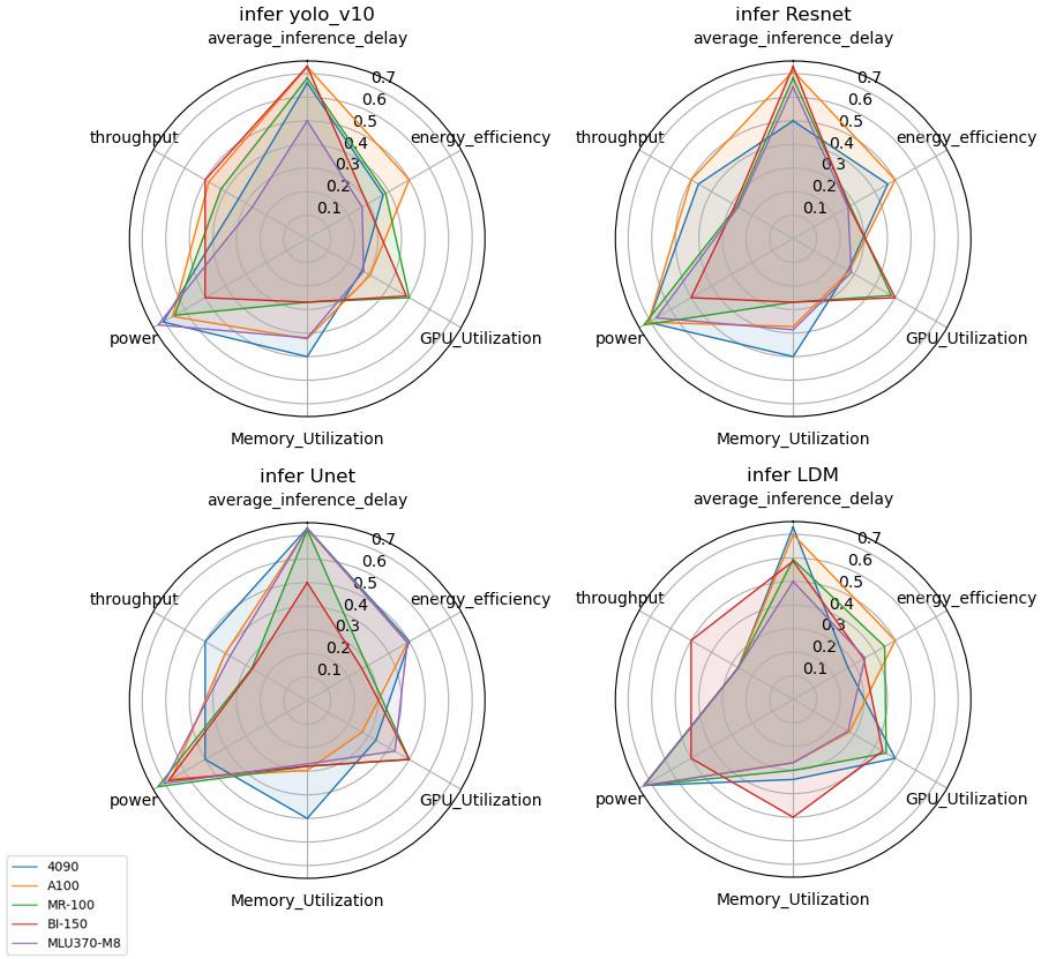

图 4-3 CV 模型推理过程各硬件指标对比

在高性能 GPU（如 NVIDIA GeForce RTX 4090）上，图像分类和目标检测模型如 ResNet 和 YOLOv10-N 表现优异，尤其是在处理大规模数据集时。而在国产设备上，不论是训练还是推理，吞吐量都有差距。

4.2.2 NLP 任务

NLP 任务训练结果如下:

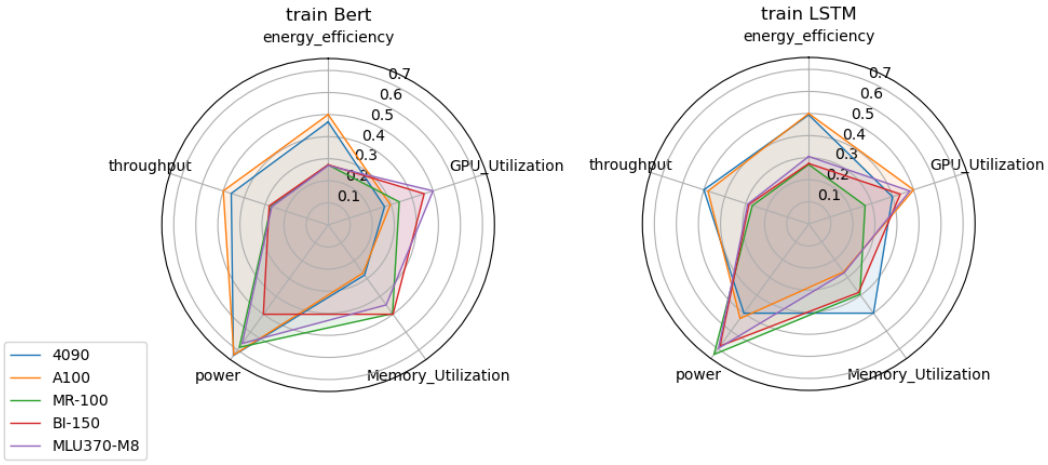

图 4-4 NLP 模型训练过程各硬件指标对比

NLP 任务推理测试结果如下:

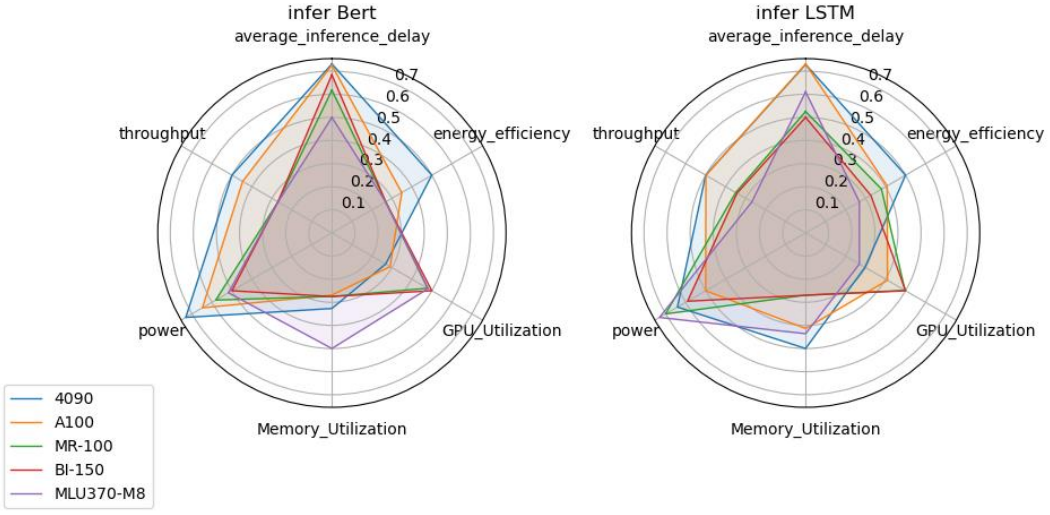

图 4-5 NLP 模型推理过程各硬件指标对比

BERT 和其他基于 transformers 的模型在国产设备上推理时展现了与 NVIDIA 相当的性能；而在训练时吞吐量和能效比不如 NVIDIA 设备。LSTM 作为一个比较早期的模型，推理的消耗不大，性能瓶颈主要在数据传输、格式转换部分。国产设备在数据传输上稍逊色，LSTM 的 GPU 利用率维持在一个较低的水平。

4.2.3 大语言模型推理

大语言模型推理测试结果如下：

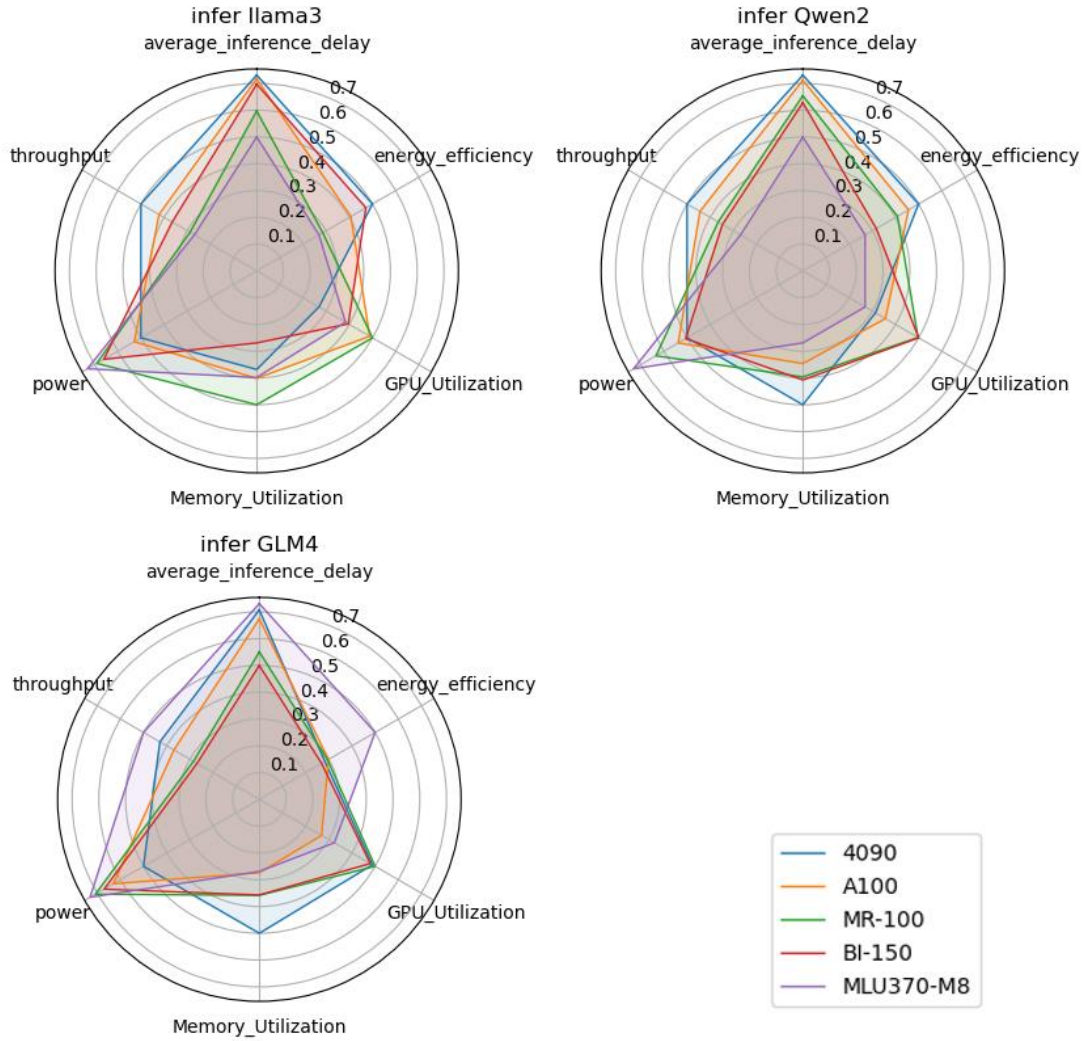

图 4-6 LLM 推理过程各硬件指标对比

大语言模型在资源丰富的环境中运行更为高效。国产设备在厂家提供的一些专用的推理优化的作用下，能够体现出与国际品牌相近的性能。

值得注意的是，4090 设备在运行三个大模型时，由于显存不足，都使用了 FP16 量化处理；寒武纪设备在运行 GLM-4 模型时也做了 FP16 量化处理。因此他们运行时测试的指标较高。

4.3 Jetson 设备特性分析

Jetson 设备作为一个进行边缘计算的嵌入式设备，往往除了计算速度之外，保持低能耗和较高的能效比也是非常重要的。

以 YOLOv10-N 的推理为例，Jetson 设备和其他设备在平均功率、能效比和吞吐量上的对比表格如下：

指标 \ 模型	Jetson	A100	4090	MR-100	BI-150	MLU370-M8
平均功率	2.868148	36.38059	50.93523	53.9986	95.37142	28.03716
能效比	3.689775	0.662939	0.895696	0.685889	0.505087	0.450064
吞吐量	10.58282	24.11812	45.62248	37.03704	48.17083	12.61852

表 4-1 Jetson 与其他设备在平均功率、能效比、吞吐量的对比

其中平均功率的单位为 W，能效比的单位为 Items / W，吞吐量的单位为 Items / s。

可以发现，Jetson 设备在吞吐量上的表现并不突出，但是能耗远远小于其他设备，因此具有很高的能效比。能效比也是嵌入式计算设备最关注的指标。

5. 性能分析与讨论

5.1 设备性能比较

通过对 NVIDIA 设备和国产计算设备进行详细的性能比较，可以观察到以下几点：

- **计算能力**：NVIDIA 的高端设备（如 A100 和 GeForce RTX 4090）在浮点运算能力上显著优于国产设备。然而，国产设备在针对 AI 优化的计算核心上，尤其是在特定的深度学习任务上，表现也很出色，甚至某些任务上胜过 NVIDIA 计算设备。
- **内存带宽**：NVIDIA 设备通常配备较高带宽的 HBM2 或 GDDR6X 内存，这为大规模数据集提供了更快的数据处理能力。相比之下，国产设备虽然也采用高带宽内存技术，但在某些模型中，内存带宽的瓶颈仍然明显。
- **功耗和热效率**：国产设备在功耗控制上表现更为优秀，这得益于更加精细的能源管理技术和近年来硬件优化的进步。这一点在边缘计算和嵌入式应用中尤为重要。

更详细一点，可以从不同设备运行大语言模型的性能表现与运行经典神经网络的性能表现这两个维度分别进行分析。

【运行大语言模型性能】：

本次测试，选择了 GLM-4(9B)、Qwen2(0.5B)以及 LLaMA3(8B)三个大语言模型在单个计算设备上进行推理，意在评测单卡大语言模型推理性能。分析吞吐量可以发现，同样的大语言模型在 A100、4090 上运行推理的吞吐量比在 MR-100、BI-150、MLU370-M8 上运行的吞吐量要大（寒武纪在运行 GLM-4 时做了 FP16 量化）。在这一点上 NVIDIA 的显卡较国产的显卡有一定的优势。分析平均前向推理时延可以看出，大体上 NVIDIA 的设备推理速度也具备一定的优势。

【运行经典神经网络性能】：

本次测试选择了 LSTM、ResNet、YOLOv10-N、LDM、U-Net、BERT 等相对传统的神经网络进行训练与推理测试。结果表明，国产设备在平均能耗上有着很大的优势，部分设备在特定任务上展现出了与 NVIDIA 设备相抵甚至更优的能效比；而在吞吐量方面，国产设备在某些任务上能够媲美 NVIDIA 设备，但是少有国产设备能在以用到的所有模型为负载时，都发挥出不错的性能。

5.2 影响因素分析

影响模型性能的关键因素包括：

硬件架构优化：

设备的硬件架构对于运行特定类型的深度学习模型至关重要，不同的硬件架构对于深度学习任务的支持程度不同。例如，Tensor 核心和 DPU 的优化可以显著提升在 AI 特定任务上的表现，有效加速前向传播和反向传播运算，显著提升模型训练和推理的速度。

显存和带宽：

在模型规模较大，或者使用了较为复杂的大规模数据集时，会使得模型的运行性能受到显存容量和数据传输带宽的制约。显存容量不足可能导致无法加载完整精度的模型或大规模数据集，而带宽低则限制了数据在显存与处理单元之间的传输速度，这直接影响到模型训练和推理的效率。

软件和硬件的协同优化：

深度学习框架如 PyTorch，以及一些提供硬件支持的后端，调用硬件的计算性能的能力，直接决定了模型的运行效率。软件层面的优化，包括算法的并行处理策略、数据传输模式的优化、支持新版本的算子等，都能显著提高硬件资源的利用率。

Batch Size 的选择：

在模型的训练和推理过程中，Batch Size 的大小也是影响设备性能发挥的一个重要因素。较大的 Batch Size 可以提高显存利用率和并行处理的效率，但也可能导致显存溢出或降低模型训练的准确性。适当的 Batch Size 设置需要根据具体的硬件配置和模型需求来调整，以达到训练速度和模型性能的最佳平衡。为了更明显地突出硬件性能之间的差距，被测模型使用了统一的 Batch Size，这意味着一些显存比较大的设备仍有一些继续提升利用率的潜力存在。

5.3 实验的局限性和不足

通用框架的局限性：

测试结果仅针对于通用深度学习框架 PyTorch，国产设备上测试时使用的是厂家适配后的 PyTorch 框架，而非原生的各厂家的自研框架，比如华为的 mindspore 框架，寒武纪推理加速引擎 MagicMind 等，会使得一些国产设备在运行模型时并不能把计算能力发挥到最大，产生一定的局限性。

算子支持不足：

在一些国产设备上，对于复杂模型中使用的较新版本算子支持不足，这导致必须在 CPU 上执行某些操作，从而大大影响了模型在国产设备上运行的性能。

系统差异：

不同设备的性能也受到整个计算系统配置的影响，包括 CPU 性能、系统内存和存储速度、传输带宽等。我们并没有在统一的计算系统（操作系统、CPU 型号等）配置下使用各个计算设备，这会使测试结果受到一定程度的影响。

6. 总结与展望

6.1 总结

通过较为详尽的测试和比较，我们分析了各种深度学习模型在 NVIDIA 及国产计算设备上的性能表现。

结果显示，尽管 NVIDIA 的设备在一些高负载的计算任务中仍然保持领先地位，国产计算设备也展现了强大的竞争力，在维持能耗较低的基础上，尽可能达到与 NVIDIA 设备相抵的吞吐量和能效比。

国产设备在针对于通用深度学习框架 PyTorch 的个性化适配上，展现了较为成熟的官方&社区生态，使用 PyTorch 的过程中遇到的大部分的问题都可以在文档中找到相应的资料，显示了其在本土市场的应用潜力。尽管在全球市场上，NVIDIA 设备在技术成熟度和生态系统支持方面仍具有优势，但国产计算设备正逐步缩小这一差距，国产计算设备的编程生态也在日趋完善。

6.2 展望

国产计算设备在峰值算力上已经能媲美甚至超越 NVIDIA 设备，问题在于内存带宽不够导致的数据传输瓶颈，以及软硬件协同优化的部分并不能将设备的计算性能很好的利用起来。

国产设备性能提升的下一步，除了提高专用计算单元的计算能力，增加计算核心数目，另外也要做出在数据传输带宽上的优化，以接近 NVIDIA A100 2TB/s 超大的显存带宽，突破数据传输速率带来的计算瓶颈。

另一方面，国产设备的生态也在日趋完善，对 PyTorch 等流行框架的支持越来越充分；提供了结构越来越完整的国产计算后端，如寒武纪的 CATCH，天数的 Corex。越来越多技术人员也投入到国产设备对流行 AI 模型的适配工作中。

更重要的是，希望有更多在国产计算设备上原生训练和推理的 AI 模型问世，更好地推进软硬件协同优化工作，避免因高度依赖 PyTorch 而受到限制。

附录

附录 A 深度学习模型简介

CV 领域

ResNet

ResNet（Residual Network）由何恺明（Kaiming He）、张翔宇（Xiangyu Zhang）、任少卿（Shaoqing Ren）和孙剑（Jian Sun）于 2015 年提出，并在论文《Deep Residual Learning for Image Recognition》中首次发表。

该模型是一种深度卷积神经网络，其主要特点是引入了残差模块，通过跳跃连接解决了深层神经网络中的梯度消失问题，使得网络可以更深、更复杂。ResNet 在 ImageNet 数据集上的出色表现使其成为图像分类任务的标杆模型，广泛应用于图像识别、物体检测和分割等计算机视觉任务。

YOLOv10-N

YOLO（You Only Look Once）系列模型最初由 Joseph Redmon 在 2016 年提出，并在论文《You Only Look Once: Unified, Real-Time Object Detection》中首次介绍。

YOLOv10（You Only Look Once version 10）是 YOLO 系列目标检测模型的最新版本，它延续了 YOLO 的单阶段检测框架，通过进一步优化网络结构和损失函数，提升了检测精度和速度，本次评测所选取的 YOLOv10-N 为其 2.3M 参数版本。YOLOv10 在实时目标检测任务中表现尤为出色，适用于自动驾驶、智能监控、无人机检测等对检测速度有严格要求的场景。

U-Net

U-Net 是由 Olaf Ronneberger、Philipp Fischer 和 Thomas Brox 在 2015 年提出，并在论文《U-Net: Convolutional Networks for Biomedical Image Segmentation》中首次发表。

U-Net 是一种基于全卷积网络的图像分割模型，采用对称的编码器-解码器结构，并通过跳跃连接有效结合高分辨率和低分辨率特征。

U-Net 在医学图像分割领域表现优异，广泛应用于细胞核分割、肿瘤检测和其他需要高精度分割的任务中。

LDM

Latent Diffusion Mode 是由 Jonathan Ho、Ajay Jain 和 Pieter Abbeel 于 2020 年提出，并在论文《Denoising Diffusion Probabilistic Models》中首次发表。

Latent-Diffusion Model 是一种生成模型，利用扩散过程和潜在空间表示进行高质量图像生成。该模型在图像生成、风格迁移和文本生成图像等任务中展现出强大的能力，是近年来生成对抗网络（GAN）和变分自编码器（VAE）的重要补充。

经典 NLP 领域

LSTM

LSTM（Long Short-Term Memory）由 Sepp Hochreiter 和 Jürgen Schmidhuber 在 1997 年提出，并在论文《Long Short-Term Memory》首次发表。

LSTM（Long Short-Term Memory）是一种特殊的递归神经网络（RNN），通过引入遗忘门、输入门和输出门解决了传统 RNN 在长序列数据处理中的梯度消失和爆炸问题。LSTM 在处理时间序列数据、语言建模和机器翻译等任务中表现出色，是语音识别、文本生成和序列预测等应用中的常用模型。

BERT

BERT（Bidirectional Encoder Representations from transformers）由 Jacob Devlin、Ming-Wei Chang、Kenton Lee 和 Kristina Toutanova 于 2018 年提出，并在论文《BERT: Pre-training of Deep Bidirectional transformers for Language Understanding》中首次发表。

BERT（Bidirectional Encoder Representations from transformers）是一种基于 Transformer 的预训练模型，通过双向编码器对上下文信息进行充分建模，显著提升了自然语言理解任务的表现。BERT 在问答、情感分析、文本分类等多种 NLP 任务中取得了卓越的效果，并催生了众多变体模型，如 RoBERTa、ALBERT 等，进一步拓展了其应用范围。

大语言模型

LLaMA3

LLaMA（Large Language Model Meta AI）是 Meta（Facebook AI Research）团队于 2023 年提出的大规模语言模型系列。LLaMA 模型在论文《LLaMA: Open and Efficient Foundation Language Models》中首次被详细描述。

该模型通过数十亿参数的训练，能够生成高质量的文本，擅长文本生成、翻译和对话等任务。LLaMA 在生成内容的连贯性和上下文理解上具有显著优势，适用于需要高度智能化语言处理的应用场景。

Qwen2

Qwen（通义千问）是由阿里巴巴达摩院发布的一种大规模多模态模型，专门设计用于处理文本、图像、语音等多种输入形式。Qwen 凭借其在多模态信息处理和融合方面的创新设计，成为了阿里巴巴在 AI 领域的重要技术成果。Qwen 模型结合了强大的自然语言处理能力和跨模态的理解与生成能力，在智能客服、多媒体内容生成、增强现实和其他复杂场景中表现出色。在测试中，主要体现 Qwen 的语言推理能力。

GLM-4

ChatGLM-4 是由清华大学的张敏（Ming Zeng）团队开发的一种针对对话优化的大规模语言模型。ChatGLM-4 的开发基于生成预训练模型技术，通过专门的对话优化策略，能够理解复杂的对话上下文并生成自然、流畅的交互内容。ChatGLM-4 被广泛应用于智能助理、聊天机器人和对话生成等领域，是中文自然语言处理领域的重要进展之一。

模型参数量：

评测所涉及的模型参数量如下：

模型	参数量
LLaMA3 8B	8,030,261,248
U-Net	31,037,698
YOLOv10-N	2,775,520
GLM-4 9B	9,483,857,920
LSTM	3,313,261
LDM	1,537,948,389
Qwen2 0.5B	494,032,768
ResNet	21,335,972
Bert-base	107,721,218
Bert-tiny	4,369,666

表 A-1 推理能效比对比

附录 B 模型使用数据集简介

ImageNet

ImageNet 是一个大型视觉数据库，用于支持各种计算机视觉任务的研究。该数据集包含超过 1400 万张标注的图像，并分为 20,000 多个类别。ImageNet 最广为人知的部分是用于 ImageNet 大规模视觉识别挑战赛（ILSVRC）的子集，其中包含约 1000 个类别和 120 万张训练图像。该数据集可以通过以下链接获取：http://www.image-net.org/

ImageNet 数据集的文件结构如下：

```
ImageNet/
│
├── train/
│   ├── n01440764/
│   │   ├── image1.JPEG
│   │   └── ...
│   └── ...
├── val/
│   ├── n01440764/
│   │   ├── image1.JPEG
│   │   └── ...
│   └── ...
└── labels.txt
```

本次测试中使用了裁剪版的 ImageNet 数据集对 ResNet 模型进行训练与推理测试。

COCO

COCO（Common Objects in Context）是一个大规模图像数据集，主要用于对象检测、分割和图像标注等计算机视觉任务。COCO 数据集包含了超过 330,000 张图像，标注了 80 种对象类别，涵盖了丰富的日常生活场景。每张图像不仅有物体的标注，还包括场景上下文的描述，使其在目标检测和分割任务中非常流行。COCO 数据集是评价模型在复杂视觉任务中表现的标准基准。数据集可以通过以下链接获取：https://cocodataset.org/#home

COCO 数据集的数据结构如下：

```
{
  "images": [
    {
      "id": 1,
      "file_name": "000000000001.jpg",
      "height": 480,
      "width": 640
    }
  ],
  "annotations": [
    {
      "id": 1,
      "image_id": 1,
      "category_id": 18,
      "bbox": [473.07, 395.93, 38.65, 28.67],
      "area": 1105.6275,
      "iscrowd": 0
    }
  ],
  "categories": [
    {
      "id": 18,
      "name": "dog",
      "supercategory": "animal"
    }
  ]
}
```

测试仅使用了其中目标检测的部分，而且训练集和验证集分别被裁剪至 5000 和 3000 items（由于设备性能限制，Jetson 所用训练集为 500 items），供 YOLOv10-N 训练和推理使用。

SQuAD v2.0

SQuAD v2.0（Stanford Question Answering Dataset v2.0）是斯坦福大学发布的一个阅读理解数据集，扩展自 SQuAD v1.1 版本。SQuAD v2.0 包含了超过 10 万个问答对，以及一些无对应答案的问题，这些问题的设计旨在测试模型的推理能力，尤其是识别无法从给定段落中回答的问题的能力。该数据集广泛用于自然语言处理领域的阅读理解和问答系统的评估。数据集可以通过以下链接获取：https://rajpurkar.github.io/SQuAD-explorer/
数据集的数据结构如下：

```
{
  "data": [
    {
      "title": "Super_Bowl_50",
      "paragraphs": [
        {
          "context": "Super Bowl 50 was an American football game...",
          "qas": [
            {
              "id": "56be4db0acb8001400a502ec",
              "question": "What was the Super Bowl 50?",
              "answers": [
                {
                  "text": "an American football game",
                  "answer_start": 22
                }
              ],
              "is_impossible": false
            },
            {
              "id": "56be4db0acb8001400a502ed",
              "question": "Which Super Bowl was played in 2016?",
              "answers": ....
              "is_impossible": true
            }
          ]
        }
      ]
    }
  ]
}
```

其中包含有文段，答案和答案在文中的位置。可以利用答案的位置，很好地训练 LSTM 的关联能力。实验在 SQuAD v2.0 数据集上训练了 BERT、LSTM，并测试了其推理效果。

CMRC2018

CMRC2018（Chinese Machine Reading Comprehension 2018）是一个中文机器阅读理解的数据集，由哈工大社会计算与信息检索研究中心（HIT-SCIR）发布。该数据集用于测试模型在阅读理解任务中的表现，特别是在中文上下文中对问题进行回答的能力。数据集中包含了从中文维基百科中提取的近 20,000 个问题，涵盖多个领域和主题。CMRC2018 常用于中文自然语言处理模型的训练和评估，尤其是中文问答系统和阅读理解任务。数据集可以通过以下链接获取：https://github.com/ymcui/cmrc2018

数据集的数据结构如下：


```
{
  "data": [
    {
      "paragraphs": [
        {
          "context": "在自然语言处理中，理解文本内容的能力至关重要。",
          "qas": [
            {
              "id": "1",
              "question": "在自然语言处理中，什么能力至关重要？",
              "answers": [
                {
                  "text": "理解文本内容的能力",
                  "answer_start": 7
                }
              ]
            },
            {
              "id": "2",
              "question": "文本中提到的处理是什么？",
              "answers": [
                {
                  "text": "自然语言处理",
                  "answer_start": 1
                }
              ]
            }
          ]
        }
      ]
    }
  ]
}
```

CMRC2018 数据集的数据结构和 SQuAD v2.0 基本相同；而其身为中文问答数据集的特殊性，也更有利于测试国产大语言模型在中文输入下的推理能力。实验使用该数据集对 LLaMA3，Qwen2 以及 GLM-4 进行了推理测试。

在使用 CMRC 数据集验证大语言模型推理能力的过程中，数据的结构并不完全适合作为大语言模型的输入。因此我们对数据集进行了结构重整，把多个问题合并为一个问题，并放在文段后，最终形成整段文字作为大语言模型的输入。将所得答案与数据集中的标准答案进行对比，也能够从侧面获知大语言模型是否按照预期正确推理。

Poem-Tang

Poem-Tang 是一个专门用于中文古诗生成和研究的数据集，收集了唐代的所有诗歌，并进行了系统的标注。该数据集包含数万首唐诗，为研究和训练生成模型提供了丰富的素材。Poem-Tang 数据集在中文自然语言生成任务中，特别是在古诗生成和文化文本处理领域，具有重要的应用价值。虽然 Poem-Tang 数据集的具体发布链接可能不太常见，但类似数据集可以参考以下链接获取：

https://github.com/chinese-poetry/chinese-poetry

Poem-Tang 的数据结构如下

```
{
  "title": "静夜思",
  "author": "李白",
  "dynasty": "唐",
  "content": [
    "床前明月光，",
    "疑是地上霜。",
    "举头望明月，",
    "低头思故乡。"
  ],
  "id": "1"
}
```

实验使用 Poem-Tang 数据集训练了 BERT 模型和 LSTM 模型，用以检验其文本生成能力在中文上的泛化能力。

Carvana Image Masking 2017

Carvana Image Masking 2017 数据集作为 Kaggle 平台上一项计算机视觉挑战的一部分，向参与者提供了一个解决复杂图像分割任务的独特机会。该数据集由在线汽车零售公司 Carvana 提供，包含高分辨率的汽车图像。该数据集包含大量汽车图像（格式为 .jpg）。每辆车都有正好 16 张图片，每张图片都是从不同的角度拍摄的。每辆车都有一个唯一的 ID，图像根据 ID 命名，如 id_01.jpg、id_02.jpg……id_16.jpg。除了这些图像之外，数据集还提供了一些关于汽车品牌、型号、年份和配置的基本元数据。

数据集可以通过以下链接获取：https://kaggle.com/competitions/carvana-image-masking-challenge

数据集的数据结构如下：

```
Carvana/
 |
 ├──── train/
 |      ├──── image1.jpg
 |      ├──── image2.jpg
 |      └──── ...
 |
 ├──── test/
 |      ├──── image1.jpg
 |      ├──── image2.jpg
 |      └──── ...
 |
 ├──── train_masks/
 |      ├──── image1_mask.gif
 |      ├──── image2_mask.gif
 |      └──── ...
 |
 ├──── train_masks.csv
 |
 ├──── sample_submission.csv
 |
 └──── metadata.csv
```

实验使用 Carvana Image Masking 2017 数据集训练了 U-Net 模型，用以检测其在图像分割领域的能力。

附录 C 测试数据详表

下方表格中将 Jetson Xavier NX 简写为 Jetson

GPU 利用率

模型训练：

单位：%

模型 \ 设备	A100	4090	Jetson	MR-100	BI-150	MLU370-M8
BERT	8.827338	20.09613	20.15075	36.19388	77.5	91.43353
LSTM	58.9596	76.18571	98.66567	34.07442	65.43111	73.22259
ResNet	71.8087	60.41509	96.49836	41.91813	89.44327	30.58469
U-Net	51.20734	46.12729	--	31.78909	65.32341	70.65008
YOLOv10-N	35.7284	30.29134	38.8362	35.40068	82.92208	54.02041

表 C-1 训练 GPU 利用率对比

注：除 Jetson Xavier NX 使用 BERT-Tiny，其余均使用 BERT-Base

模型推理：

单位：%

模型 / 设备	A100	4090	Jetson	MR-100	BI-150	MLU370-M8
LLaMA3	76.65263	94.81712	--	95.73871	87.64961	86.65621
Qwen2	38.57368	50.0515	--	84.79233	85.43137	25.54132
GLM-4	88.33333	62.348	--	89.605	87.33019	69.56037
BERT	20.75	29.5	90.76789	92.10373	98.78142	92.94464
LDM	91.37931	79.82938	--	89.2575	88.41954	79.37178
LSTM	19.7931	50	54.77709	71.68293	72.40741	11.33333
ResNet	3.965116	6.418919	43.12069	80.77941	87	10.75221
U-Net	45.1875	19.69185	--	99.03966	98.89562	76.57595
YOLOv10-N	9.273492	23.1	60.64506	80.38143	76.35821	12.01782

表 C-2 推理 GPU 利用率对比

注：除 Jetson Xavier NX 使用 BERT-Tiny，其余均使用 BERT-Base

显存占用

模型训练：

单位：%

模型 \ 设备	A100	4090	Jetson	MR-100	BI-150	MLU370-M8
BERT	4.040569	3.168238	49.83453	17.74404	17.95349	14.84686
LSTM	51.52992	33.81676	47.80271	43.97116	43.18425	34.23908
ResNet	16.35334	9.878523	66.46226	5.020666	5.023676	5.593409
U-Net	83.75464	95.97029	--	71.00549	60.50191	86.57086
YOLOv10-N	61.09966	34.91242	61.91851	49.03458	40.04699	73.19934

表 C-3 训练显存占用对比

模型推理：

单位：%

模型 / 设备	A100	4090	Jetson	MR-100	BI-150	MLU370-M8
BERT	3.539987	2.881073	51.21195	2.950049	2.954102	5.25993
GLM-4	81.54828	49.57252	--	62.40355	62.2174	48.71737
LDM	29.76031	18.61287	--	23.84729	51.79922	18.59989
LLaMA3	73.78615	80.14072	--	98.22885	52.97179	79.47027
LSTM	4.594165	3.581413	41.90871	1.70556	1.715088	3.84942
Qwen2	68.66205	41.99951	--	50.95127	52.91527	26.71875
ResNet	5.804667	3.358033	78.69763	1.120534	1.1245	3.633003
U-Net	65.98308	40.04746	--	37.55744	37.43284	36.03099
YOLOv10-N	4.991318	3.74079	67.51118	0.866447	0.863875	3.66692

表 C-4 推理显存占用对比

平均功率

模型训练：

单位：W

模型 \ 设备	A100	4090	Jetson	MR-100	BI-150	MLU370-M8
BERT	67.48631	68.91281	3.281344	85.25982	147.5889	93.29402
LSTM	266.0757	244.2239	4.37797	80.05158	122.6451	111.73
ResNet	238.5439	215.0004	3.99968	79.47351	138.7055	88.16512
U-Net	212.6901	203.2849	--	85.65243	140.1686	118.8201
YOLOv10-N	140.4068	109.7566	1.283521	69.74185	126.1137	76.58044

表 C-5 训练平均功率对比

模型推理：

单位：W

模型 / 设备	A100	4090	Jetson	MR-100	BI-150	MLU370-M8
BERT	48.94356	100.9884	4.98036	139.0811	182.565	173.1493
GLM-4	269.4794	171.3673	--	106.2477	138.5689	85.21441
LDM	231.2866	225.902	--	116.2091	1684975	121.2475
LLaMA3	244.4481	226.2731	--	117.7487	139.3786	86.50179
LSTM	99.47307	153.3965	3.069349	74.42748	119.3378	60.38255
Qwen2	137.5509	126.5102	--	97.81042	135.8134	65.36719
ResNet	65.54061	65.43352	0.733691	60.70497	104.2489	73.16399
U-Net	196.3906	137.5071	--	110.8536	131.9767	122.858
YOLOv10-N	36.38059	50.93523	2.868148	53.9986	95.37142	28.03716

表 C-6 推理平均功率对比

能效比

模型训练：

单位：Items / W

模型 / 设备	A100	4090	Jetson	MR-100	BI-150	MLU370-M8
BERT	4.600963	5.282783	19.11805	0.20818	0.225129	0.123653
LSTM	40.60339	41.19064	48.50426	16.60379	17.18766	21.00744
ResNet	3.991829	4.320772	5.277158	1.381378	1.510417	1.264795
U-Net	0.030627	0.039281	--	0.019607	0.023789	0.023773
YOLOv10-N	0.874444	0.697805	1.95354	0.422042	0.505027	0.871572

表 C-7 训练能效比对比

模型推理：

单位: Items / W

模型 / 设备	A100	4090	Jetson	MR-100	BI-150	MLU370-M8
BERT	67.14802	25.70266	44.20912	0.62303	0.532234	0.480704
GLM-4	0.000924	0.001146	--	0.00109	0.000721	0.003618
LDM	0.00124	0.002204	--	0.002006	0.001589	0.001617
LLaMA3	0.001076	0.000854	--	0.000537	0.001009	0.000489
LSTM	79.33039	51.19469	73.52428	42.23645	24.5275	3.870783
Qwen2	0.006105	0.005366	--	0.004533	0.002842	0.001854
ResNet	17.31674	19.95288	52.67148	2.589408	2.031798	1.619718
U-Net	0.089487	0.087514	--	0.035784	0.028558	0.087184
YOLOv10-N	0.662939	0.895696	3.689775	0.685889	0.505087	0.450064

表 C-8 推理能效比对比

平均温度：

模型训练：

单位：摄氏度

模型 / 设备	A100	4090	Jetson	MR-100	BI-150	MLU370-M8
BERT	50.53237	36.47059	43.44856	59.96939	43.97727	41.54913
LSTM	57.0101	49.48571	41.48638	65.09302	40.92444	40.09302
ResNet	57.84348	43.50943	46.96675	65.46316	43.22955	41.26224
U-Net	61.42176	47.79077	--	72.92121	41.53882	47.43969
YOLOv10-N	48.20988	36.7874	41.0481	55.71233	39.7013	37.77551

表 C-9 设备训练温度对比

模型推理：

单位：摄氏度

模型 / 设备	A100	4090	Jetson	MR-100	BI-150	MLU370-M8
BERT	48.5	36.1	41.48165	84.51867	51.45355	44.42561
GLM-4	65.78607	40.652	--	73.75	42.75708	38.06502
LDM	68.64428	48.27962	--	85.26698	41.72605	42.23067
LLaMA3	60.34211	46.3035	--	86.25308	39.08661	39.44595
LSTM	46.51724	35.59259	40.12186	57.73171	38.92593	36
Qwen2	49.96316	38.60944	--	72.19808	42.48693	38.71418
ResNet	43.98837	33.95946	40.80251	49.76471	35.80952	38.68142
U-Net	58.38194	39.34892	--	85.5872	40.57411	46.74684
YOLOv10-N	44.13675	33.44	44.11835	46.6813	35.02239	37.44664

表 C-10 设备推理温度对比

吞吐量

模型训练：

单位： Items / s

模型 / 设备	A100	4090	Jetson	MR-100	BI-150	MLU370-M8
BERT	50.53237	36.47059	43.44856	59.96939	43.97727	41.54913
LSTM	57.0101	49.48571	41.48638	65.09302	40.92444	40.09302
ResNet	57.84348	43.50943	46.96675	65.46316	43.22955	41.26224
U-Net	61.42176	47.79077	--	72.92121	41.53882	47.43969
YOLOv10-N	48.20988	36.7874	41.0481	55.71233	39.7013	37.77551

表 C-11 训练吞吐量对比

模型推理：

单位： Items / s

模型 / 设备	A100	4090	Jetson	MR-100	BI-150	MLU370-M8
BERT	3286.463	2595.671	220.1773	86.65162	97.16729	83.23352
GLM-4	0.248952	0.196448	--	0.115847	0.099865	0.308297
LDM	0.286895	0.497998	--	0.233093	2677.334	0.19604
LLaMA3	0.263063	0.19331	--	0.063234	0.140657	0.042324
LSTM	7891.238	7853.086	225.6717	3143.552	2927.057	233.7277
Qwen2	0.839686	0.678877	--	0.443341	0.386042	0.12116
ResNet	1134.95	1305.587	38.6446	157.19	211.8128	118.5051
U-Net	17.57442	12.03386	--	3.966812	3.769009	10.7112
YOLOv10-N	24.11812	45.62248	10.58282	37.03704	48.17083	12.61852

表 C-12 推理吞吐量对比

平均前向传播时延(仅推理)

模型推理：

单位：s

模型 / 设备	A100	4090	Jetson	MR-100	BI-150	MLU370-M8
BERT	0.001591	0.002047	0.01196	0.01011	0.00529	0.017963
GLM-4	4.013713	5.073373	--	8.623728	10.01019	3.23527
LDM	0.031474	0.038232	--	0.058569	0.059761	0.07491
LLaMA3	3.799365	5.16859	--	15.80738	7.104918	23.62746
LSTM	0.101609	0.099584	0.273896	0.265005	0.283669	0.200777
Qwen2	9.526832	11.78363	--	18.04355	20.72267	33.01368
ResNet	0.022193	0.005266	3.19083	0.007882	0.003433	0.011078
U-Net	0.000382	0.001098	--	0.002327	0.053315	0.000803
YOLOv10-N	0.013499	0.009613	0.046385	0.012378	0.00953	0.021495

表 C-13 平均前向传播时延对比

附录 D 术语解释

CV：

CV 指的是计算机视觉（Computer Vision），这是人工智能领域的一个重要分支，涉及让计算机从图像或多维数据中解析、理解和识别内容。计算机视觉的应用包括图像识别、视频分析、图像重建等。

DeepSpark：

天数智芯开发了自己的训练和推理框架，主要体现在其 DeepSpark 开源社区及相关技术支持上。DeepSpark 不仅是一个模型库，它还提供了完整的开源应用算法模型，支持 PyTorch 等框架，并与天垓和智铠系列 GPU 产品紧密集成。这些框架和模型专为天数智芯的硬件优化，能够提供更有效的训练和推理性能。

Hugging Face：

Hugging Face 是一个面向机器学习社区的平台，汇集了超过 12 万种模型、2 万多种数据集和 5 万多个演示，用户可以在此共享、探索并试验开源的机器学习项目。平台提供版本控制、模型卡、推理小部件等丰富功能，支持 NLP、计算机视觉、音频任务等领域的最新模型展示和交互应用。Hugging Face 致力于通过开放和社区协作推动机器学习技术的发展与进步。

NLP：

NLP 指的是自然语言处理（Natural Language Processing），这是一门专门研究机器如何处理及分析人类语言的计算机科学领域。自然语言处理的目的是让计算机能够理解和产生人类语言的内容，常见的应用包括语言翻译、情感分析、自动摘要和问答系统等。

transformers：

transformers 库由 Hugging Face 开发，是一款面向自然语言处理的强大工具库，提供了大量预训练模型来处理文本分类、信息提取、问答等多种任务。这个库支持众多流行的模型，如 BERT、GPT 和 RoBERTa，并且能够以高效率执行模型推理，使得它在处理需要大规模数据的复杂问题时表现出色。它不仅支持多语言处理，而且兼容主流的深度学习框架如 PyTorch 和 TensorFlow。transformers 库提供了简便的 API 来促进先进的 NLP 技术的普及和应用。此外它是开源的，拥有活跃的社区支持，并且持续更新，反映了最新的 AI 研究成果。

参考文献

[1] 赵玥, 肖梦燕, 罗军, 王小强, 罗道军. 人工智能芯片及测评体系分析 [J]. 电子与封装, 2023, 23 (05): 31-37.

[2] 孟博. 人工智能软硬件国产化应用情况分析 [J]. 中国安防, 2023, (12): 82-87.

[3] 王晨,邓昌义,李嘉伟,等. 人工智能芯片测评研究现状及未来研究趋势[J]. 新型工业化, 2021, 11 (10): 82-87.

[4] 中国电子技术标准化研究院,人工智能芯片面向云侧的深度学习芯片测试指标与测试方法: T/CESA1119-2020[5].北京:中国电子工业标准化技术协会, 2020.

[5] 中国电子技术标准化研究院,人工智能芯片面向边缘侧的深度学习芯片测试指标与测试方法: T/CESA1120-2020[5].北京:中国电子工业标准化技术协会. 2020.

[6] 中国电子技术标准化研究院,人工智能芯片面向端侧的深度学习芯片测试指标与测试方法: T/CESA1121-2020[51.北京:中国电子工业标准化技术协会, 2020.

[7] NVIDIA. NVIDIA A100 Tensor Core GPU[EB/OL]. https://www.NVIDIA.com/content/dam/en-zz/Solutions/Data-Center/a100/pdf/NVIDIA-a100-datasheet-us-NVIDIA-1758950-r4-web.pdf.

[8] NVIDIA. NVIDIA Jetson Xavier[EB/OL]. https://www.NVIDIA.cn/autonomous-machines/embedded-systems/jetson-xavier-series/

[9] 天数智芯. 天垓 150 加速卡产品说明书 [EB/OL]. https://support.iluvatar.com/#/DocumentCentre

[10] 天数智芯. 智铠 100 加速卡产品说明书 [EB/OL]. https://support.iluvatar.com/#/DocumentCentre

[11] Jeremy Laird. New Chinese 7nm GPU rivals NVIDIA and AMD for performance[EB/OL]. (2021-1-20)[2024-8-13]. https://www.pcgamer.com/new-chinese-7nm-gpu-rivals-NVIDIA-and-amd-for-performance/

[12] 寒武纪. MLU-M8 Intelligent Accelerating Card Product Manual Issue 0.9.1[EB/OL]. https://fccid.io/2ARVF-MLU370-M8/User-Manual/TempConfidential-MLU370-M8-user-manual-V-5528126

[13] He K, Zhang X, Ren S, et al. Deep residual learning for image recognition[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2016: 770-778.

[14] Ronneberger O, Fischer P, Brox T. U-net: Convolutional networks for biomedical image segmentation[C]//Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18. Springer International Publishing, 2015: 234-241.

[15] Redmon J, Divvala S, Girshick R, et al. You only look once: Unified, real-time object detection[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2016: 779-788.

[16] Wang A, Chen H, Liu L, et al. Yolov10: Real-time end-to-end object detection[J]. arXiv preprint arXiv:2405.14458, 2024.

[17] Rombach R, Blattmann A, Lorenz D, et al. High-resolution image synthesis with latent diffusion models[C]//Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2022: 10684-10695.

[18] Devlin J. BERT: Pre-training of deep bidirectional transformers for language

understanding[J]. arXiv preprint arXiv:1810.04805, 2018.
[19] Schmidhuber J, Hochreiter S. Long short-term memory[J]. Neural Comput, 1997, 9(8): 1735-1780.
[20] Yang A, Yang B, Hui B, et al. Qwen2 technical report[J]. arXiv preprint arXiv:2407.10671, 2024.
[21] Touvron H, Lavril T, Izacard G, et al. Llama: Open and efficient foundation language models[J]. arXiv preprint arXiv:2302.13971, 2023.
[22] Team G L M, Zeng A, Xu B, et al. Chatglm: a family of large language models from glm-130b to glm-4 all tools[J]. arXiv e-prints, 2024: arXiv: 2406.12793.