基于GPU的AI计算优化方法与案例：从训练到推理

张清，浪潮AI首席架构师

YOUR SUCCESS, WE SUCCEED

提纲

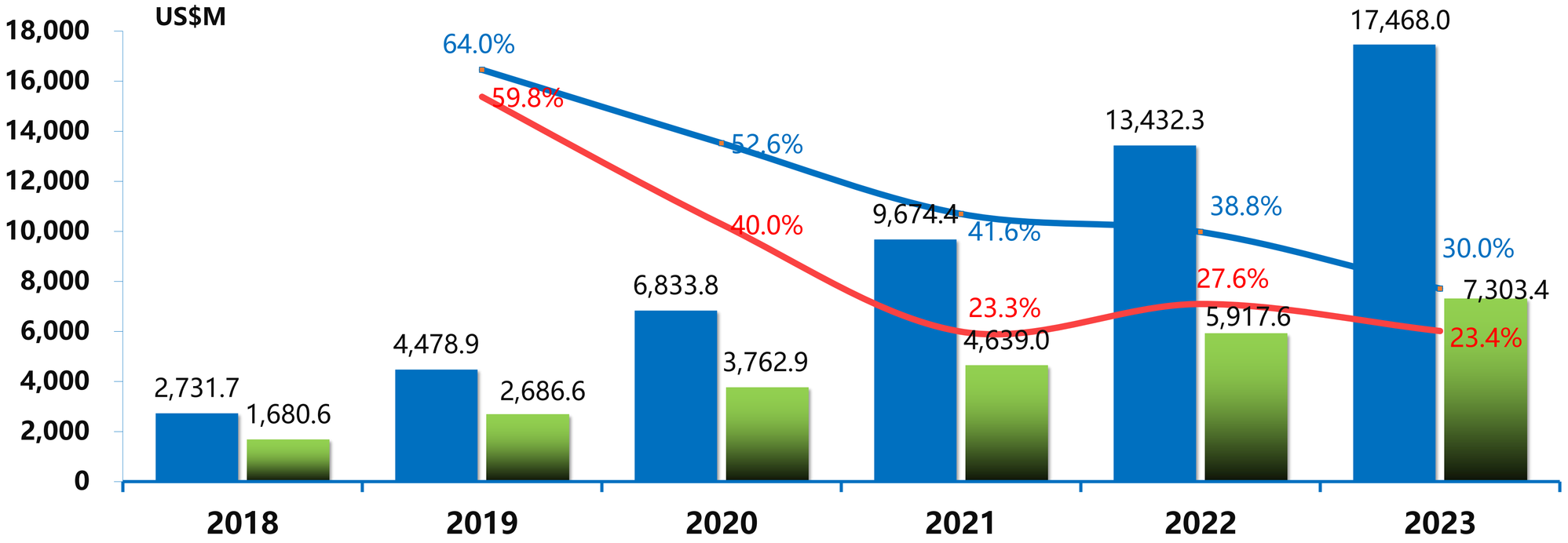
AI计算的发展趋势及其挑战

基于GPU的AI计算优化方法：从训练到推理

Case Study:基于GPU实现AutoML Suite计算优化

AI计算的发展趋势

趋势1：越来越多的场景将采用AI技术创新，未来计算投入会越来越大



整体投资 **中国人工智能总体市场规模及预测, 2018-2023**

算力投资 **中国人工智能服务器市场规模及预测, 2018-2023**

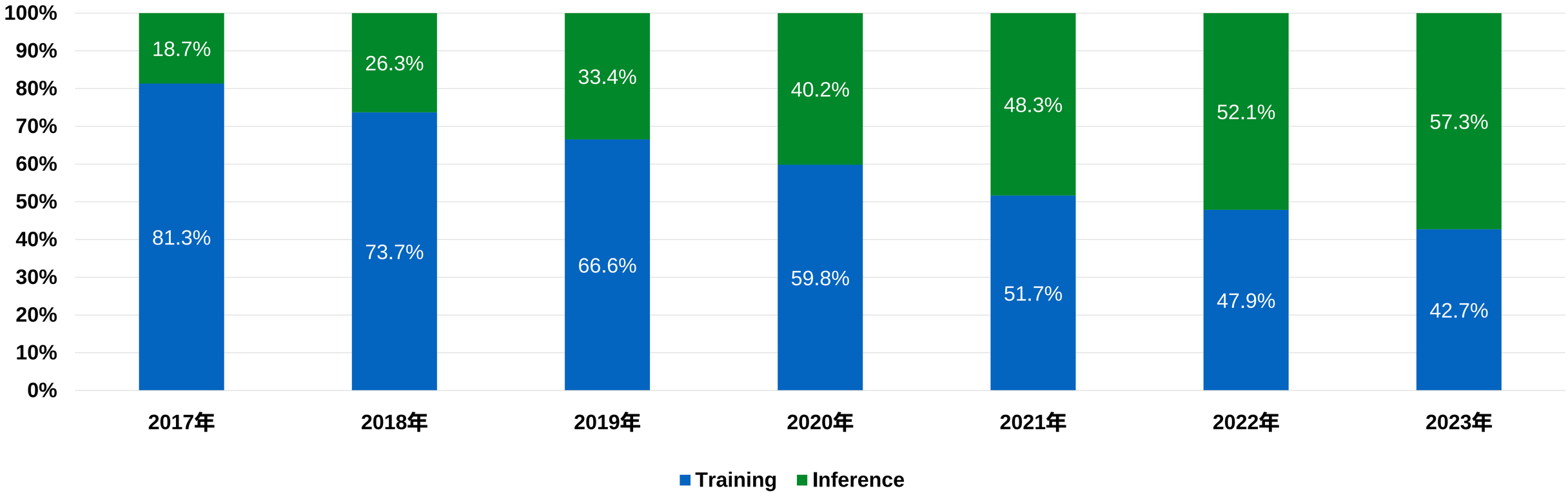
AI Spending　Growth Rate

AI Server　Growth Rate

Source :IDC 2019

AI计算的发展趋势

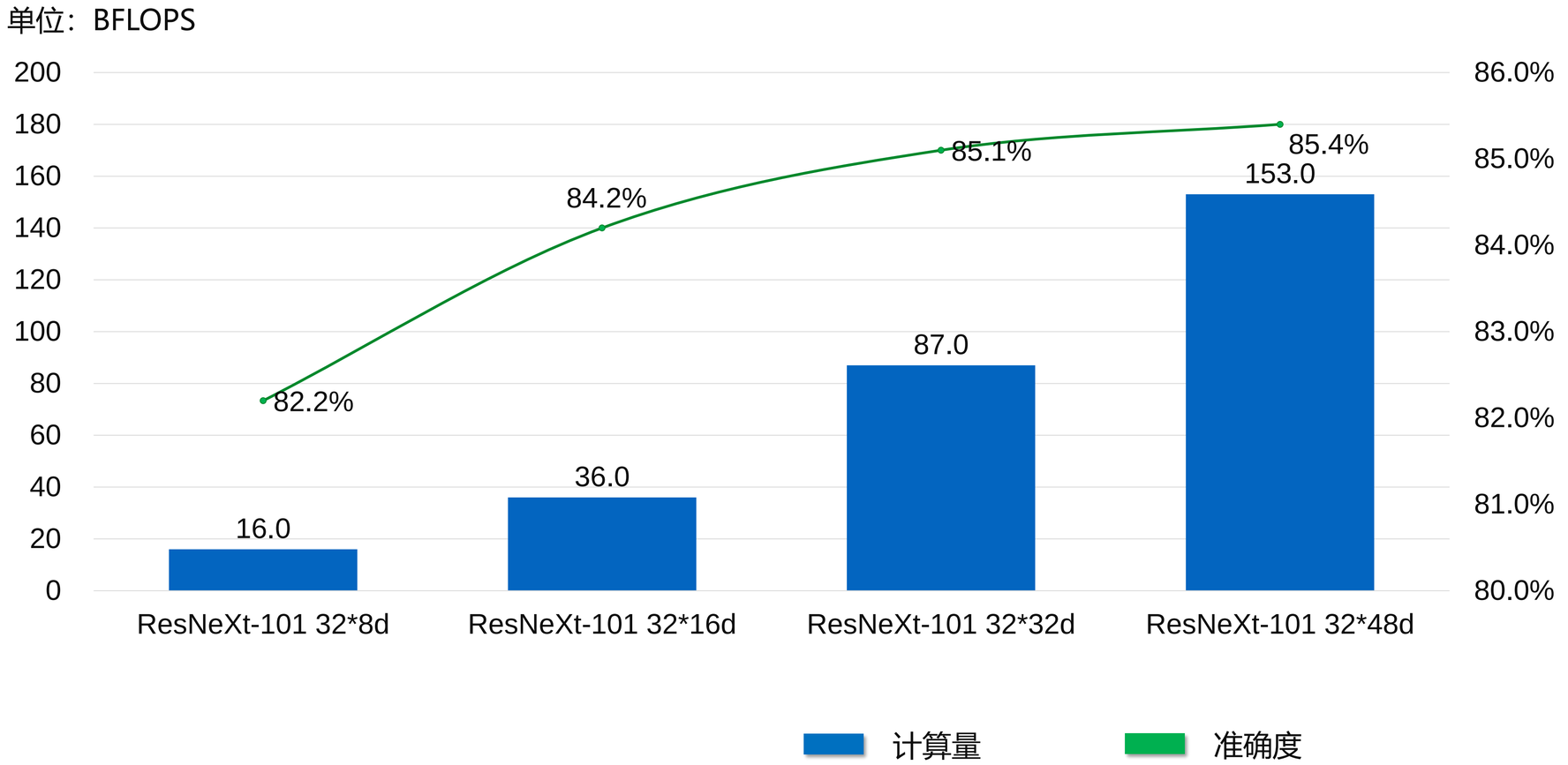
趋势2：越来越多的AI应用将进入生产阶段，未来5年推理所需计算会迅速增加



Source：IDC 2019

AI计算的发展趋势

趋势3：大数据+大模型，需要更大的计算



Source：Facebook

AI计算面临的挑战

- AI计算架构：芯片间异构与芯片内异构
  - 异构并行与协同计算
    - CPU/GPU，CUDA Core/Tensor Core
- AI计算规模：K级节点、10K级GPU卡
  - 性能与性能的可扩展性
    - 单模型K级以上GPU并行计算
- AI计算环境：不同用户、不同算法、不同数据、不同框架、不同GPU卡
  - 任务管理与资源调度
    - 生产系统K级以上模型并发调度

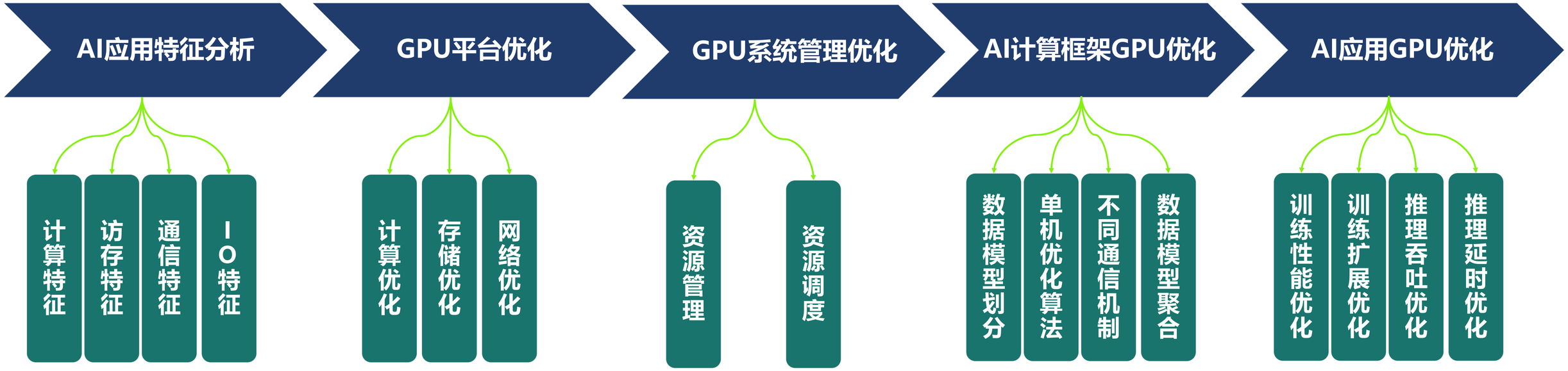
提纲

AI计算的发展趋势及其挑战

基于GPU的AI计算优化方法：从训练到推理

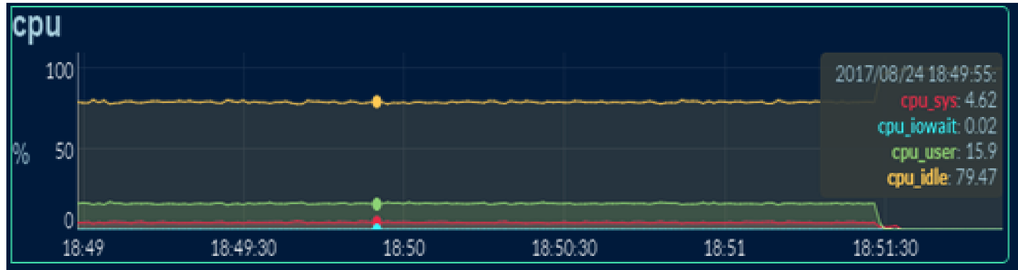
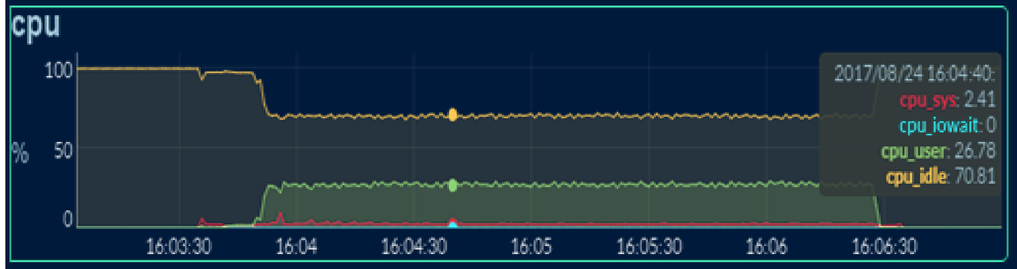
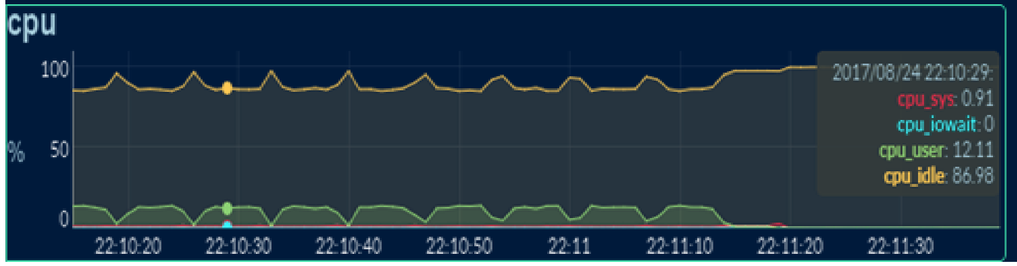
Case Study:基于GPU实现AutoML Suite计算优化

基于GPU的AI计算优化方法

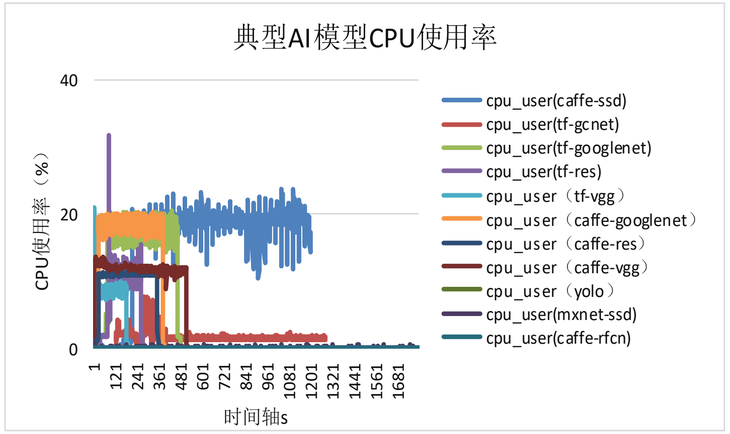


AI应用特征分析

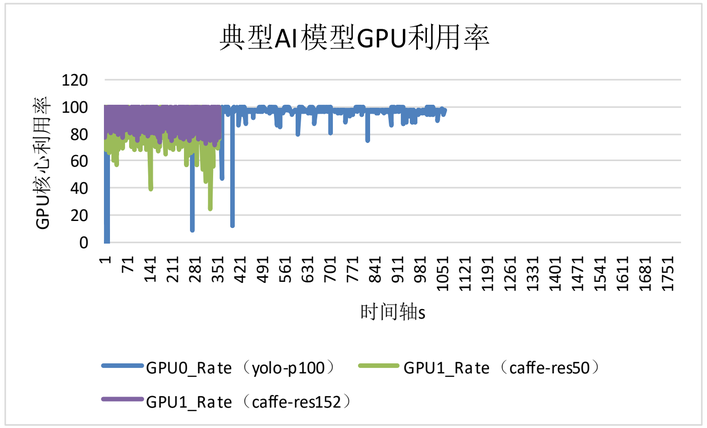
Teye工具：从微架构层次分析AI应用与框架特征，实现性能优化

	CPU	GPU
TensorFlow		
Caffe		
MXNet		

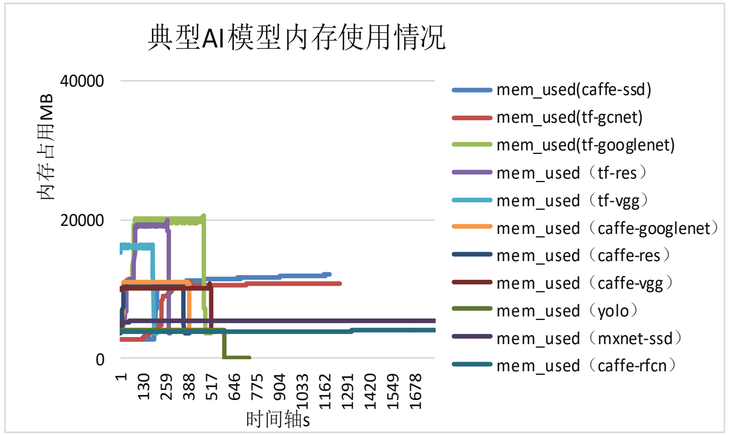
CV应用特征分析案例



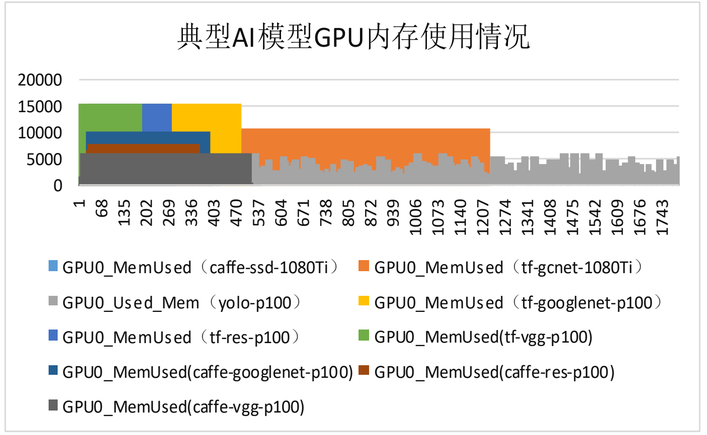
CPU利用率：5%-25%



GPU利用率：80%-100%



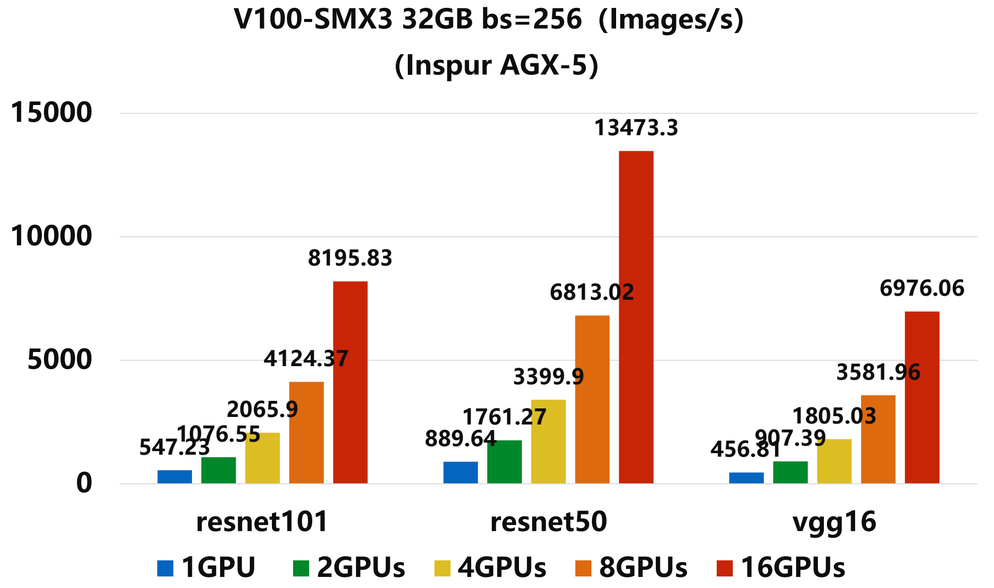
CPU内存：20GB以下

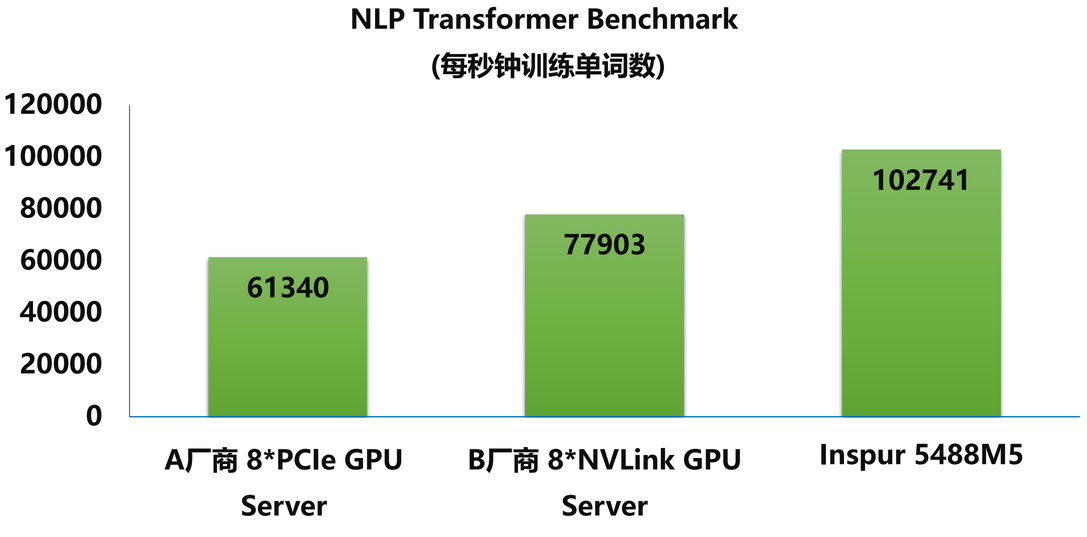


GPU内存：15GB左右

GPU平台优化

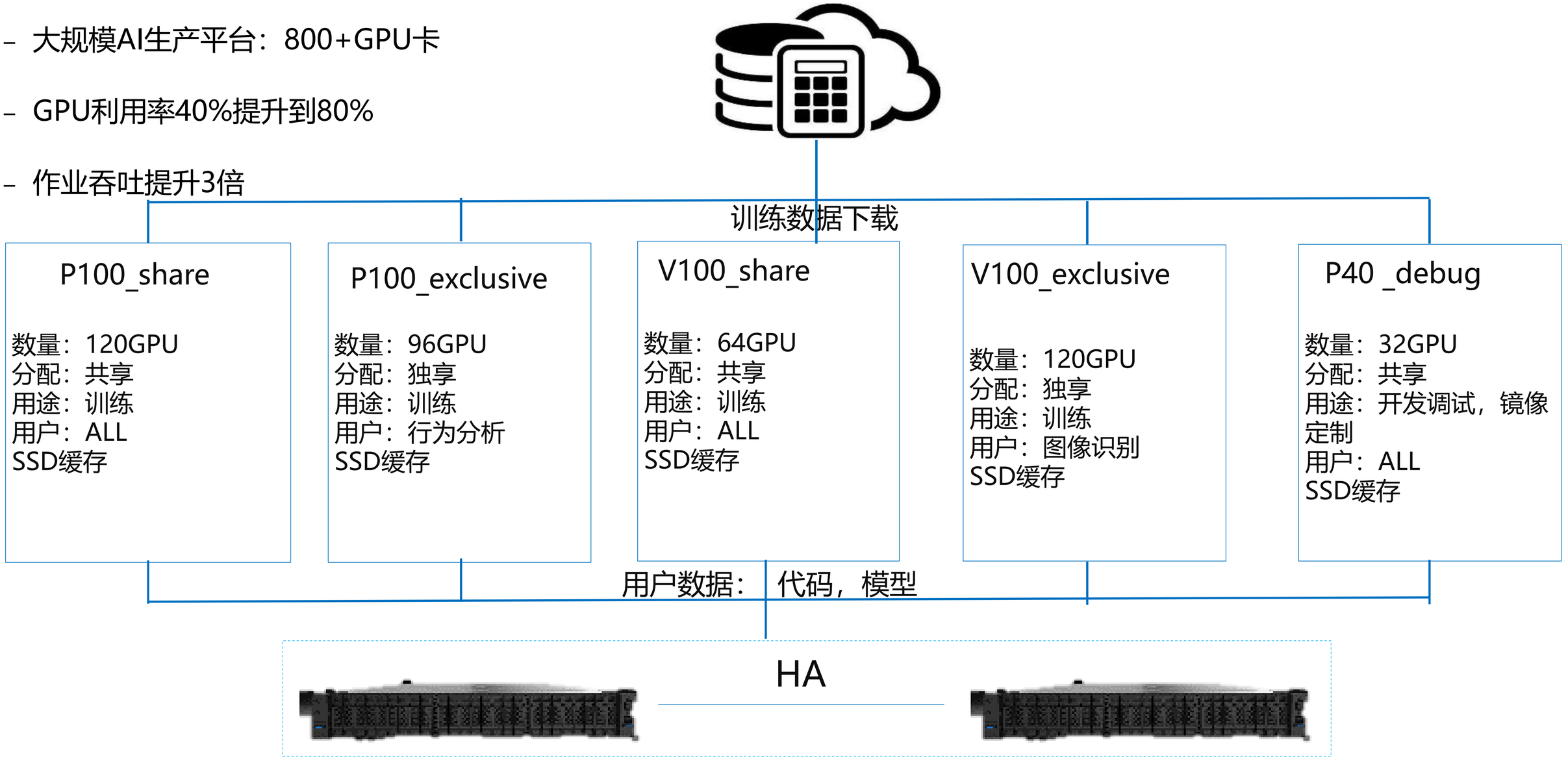
- 计算优化
  - 训练：单机8-16 V100 GPU并行
  - 推理：单机8-16 T4 GPU并行
- 网络优化
  - 训练：单机4-8个 IB卡（100GB/s-200GB/s）实现1000卡以上并行
  - 推理：单机万兆网络
- 通信优化
  - 训练：NVSwitch+RDMA
  - 推理：PCIE
- 存储优化：高性能并行存储+SSD/NVMe两级存储





GPU系统管理优化

- 利用AIStation实现统一资源管理和调度
  - 大规模AI生产平台：800+GPU卡
  - GPU利用率40%提升到80%
  - 作业吞吐提升3倍



AI计算框架GPU优化

计算

- 并行机制：数据并行/模型并行/数据+模型并行/Pipline并行
- GPU计算充分发挥：FP16与FP32混合精度计算，保持训练稳定下的大batchsize训练

通信

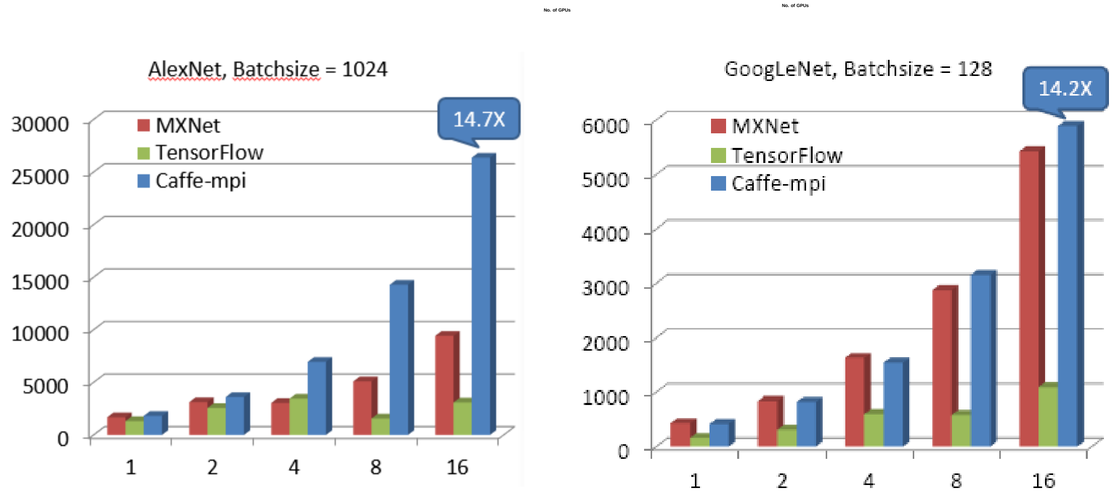
- 梯度同步通信机制：异步或半异步，ring-allreduce，2D-Torus all-reduce
- 通信优化：合并小数据，提升通信效率；计算与通信异步，实现隐藏

IO

- 并行IO，采用多线程的数据读取机制
- 数据预取、数据IO与计算异步并行

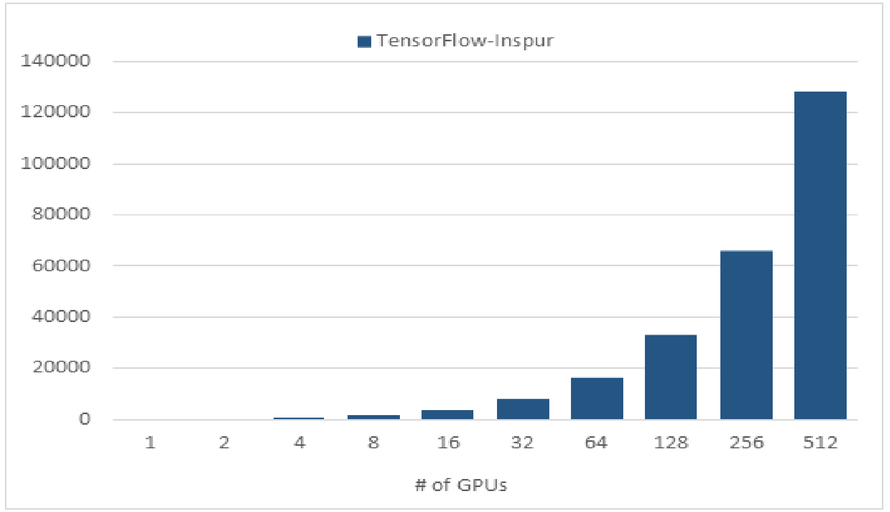
AI计算框架GPU优化案例

Inspur Caffe-MPI



开源地址：https://github.com/Caffe-MPI/Caffe-MPI.github.io

Inspur TensorFlow-Opt



实现512块GPU 24分钟完成imagenet数据集训练

- 基于HPC架构，实现数据并行，并行IO读取数据
- 基于NCCL，并采用环形通信方式
- 计算与通信异步，实现计算与通信的异步隐藏
- 实现主从模式到对等模式通信
- 合并梯度，提升通信效率
- 采用fp16通信，减少通信量

AI应用面临的挑战分析及优化思路

- 数据跟不上计算，GPU利用率低
- 模型和数据大，GPU显存溢出，如何优化
- 混合精度如何优化， Tensor Core如何高效利用
- 如何快速实现多机多GPU卡并行计算

应用瓶颈分析 → **GPU系统级优化** → **GPU代码级优化**

- Pref/NVProf
- Tensorflow-timeline
- Horovod-timeline
- Teye

- GPU-driver/CUDA/cuDNN/NCCL
- 计算框架版本匹配
- CPU/GPU端、Bios设置
- CPU与GPU、GPU与GPU、节点间通信拓扑

- 训练的性能
- 训练的扩展效率
- 推理的吞吐量
- 推理的延时

AI训练应用GPU优化方法

数据IO优化

数据格式、数据存储、数据处理、数据流水线

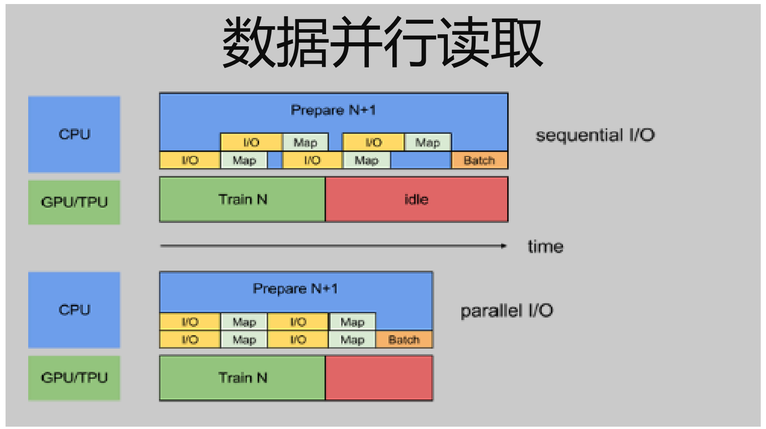
混合精度优化

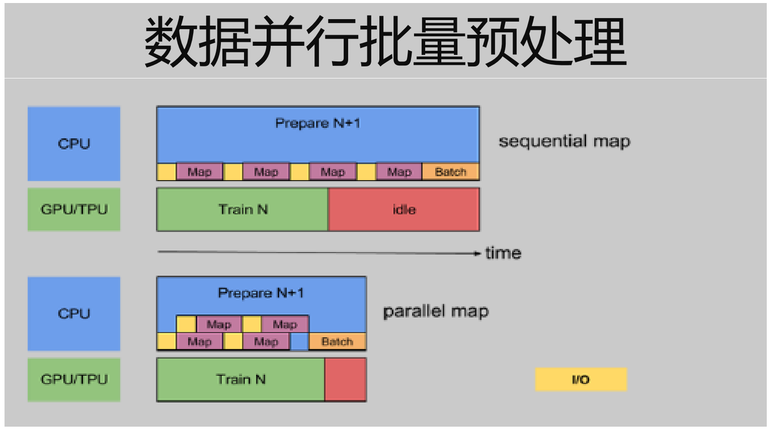
使用CUDA Core & TensorCore发挥GPU使用效率

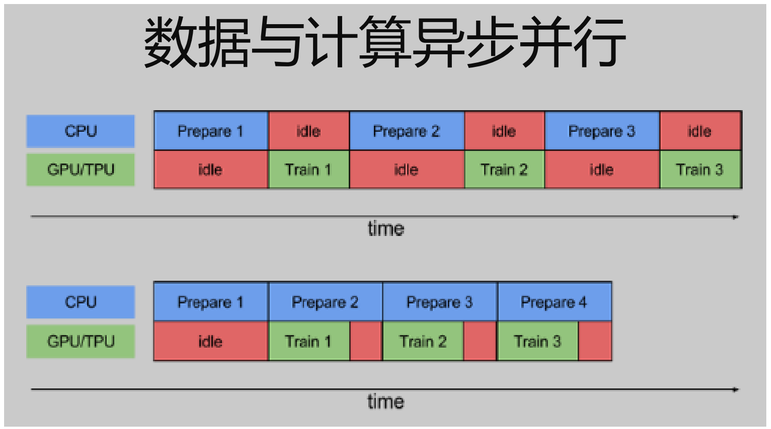
GPU并行优化

使用ring（tree）-allreduce高效并行通信方式

数据IO优化


数据并行读取
CPU
GPU/TPU
Prepare N+1
I/O
Map
Batch
Train N
idle
sequential I/O
time
parallel I/O

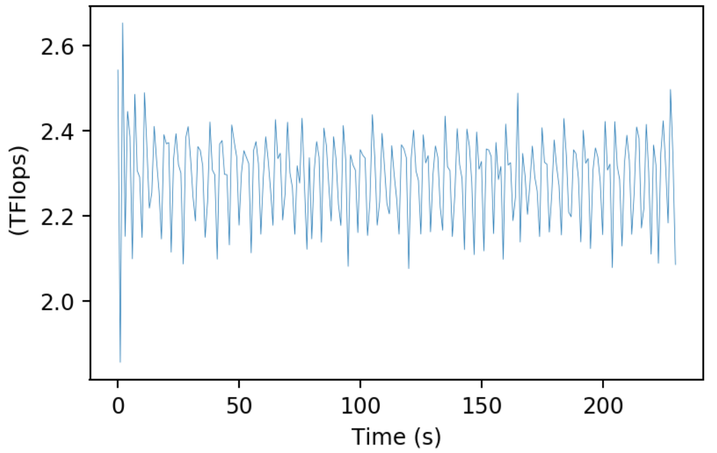

数据并行批量预处理
CPU
GPU/TPU
Prepare N+1
Map
Batch
Train N
idle
sequential map
time
parallel map
I/O

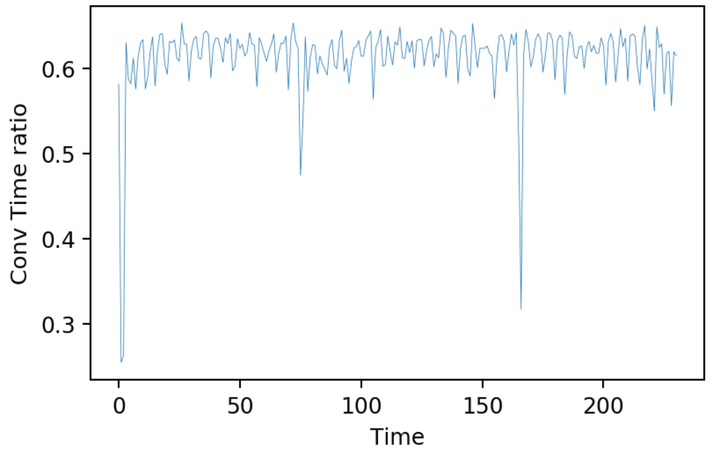

数据与计算异步并行
CPU
GPU/TPU
Prepare 1
Prepare 2
Prepare 3
Prepare 4
idle
Train 1
Train 2
Train 3
time

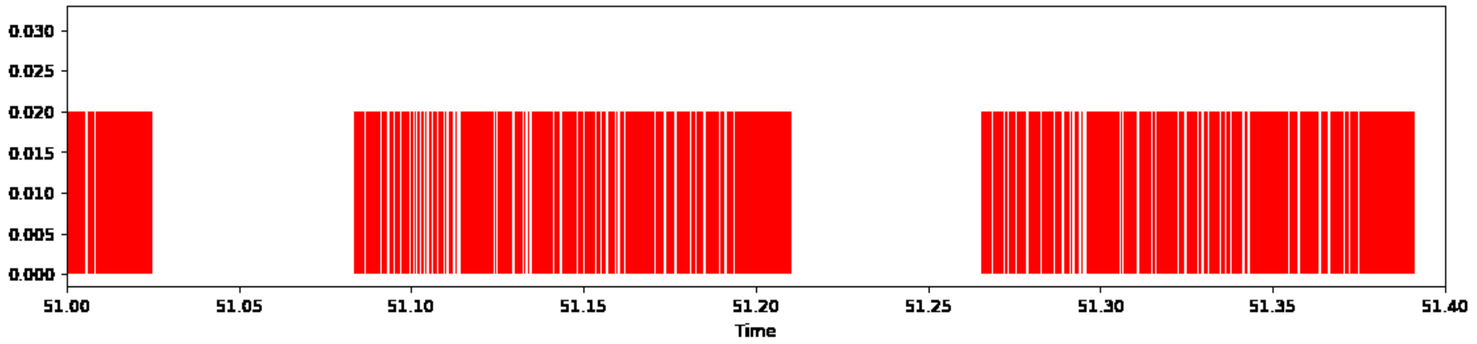
数据IO优化案例1

某图像识别CNN模型（在P100平台训练）

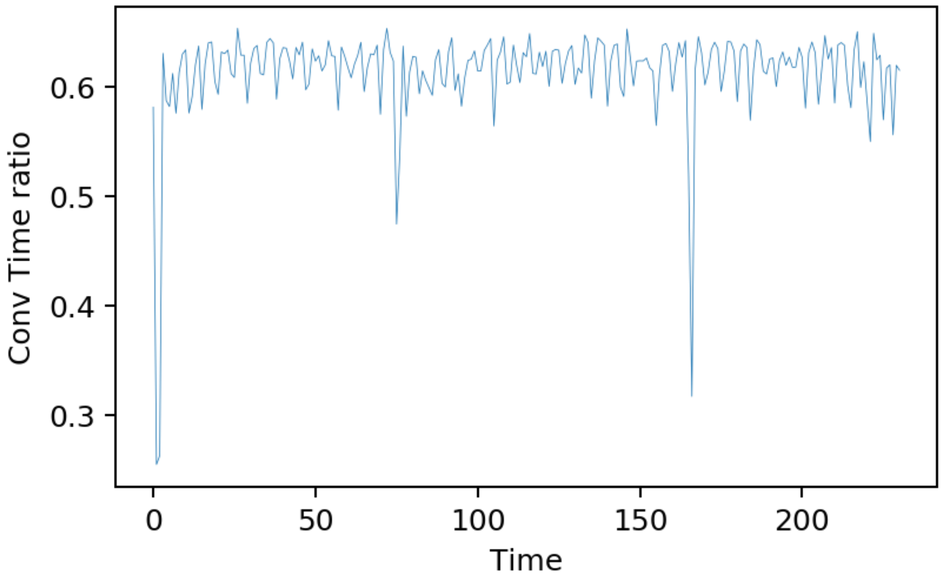
- 实测单卡计算性能只有2.3TFlops，远低于P100的理论单精度浮点性能；
- 分析GPU的利用率，发现GPU只有60%左右的时间在参与计算，剩余40%的时间处于空闲状态；
- 在毫秒尺度观察GPU的使用情况，发现有周期性的0.06s左右的GPU空闲时间

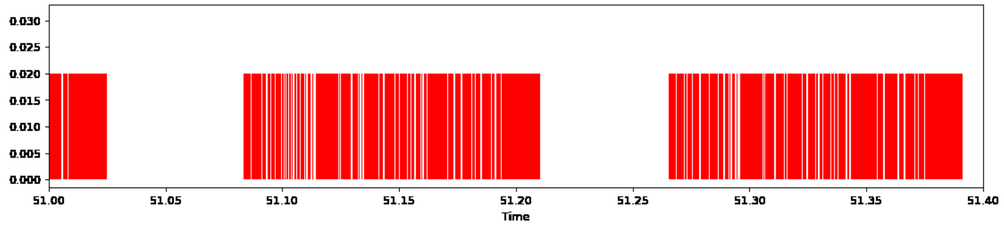


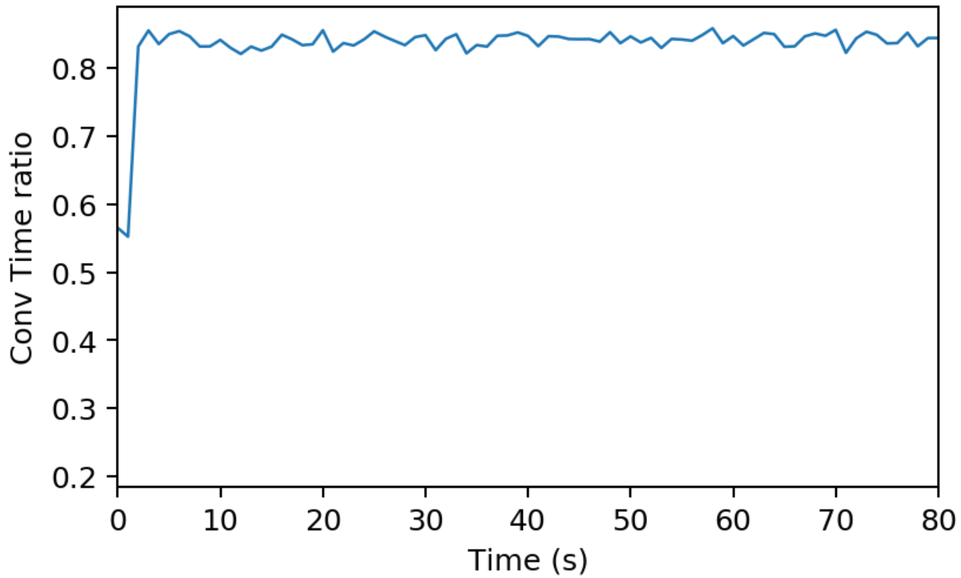


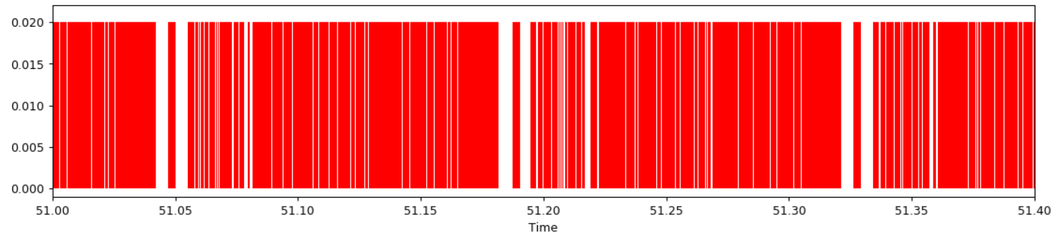


数据IO优化案例1效果





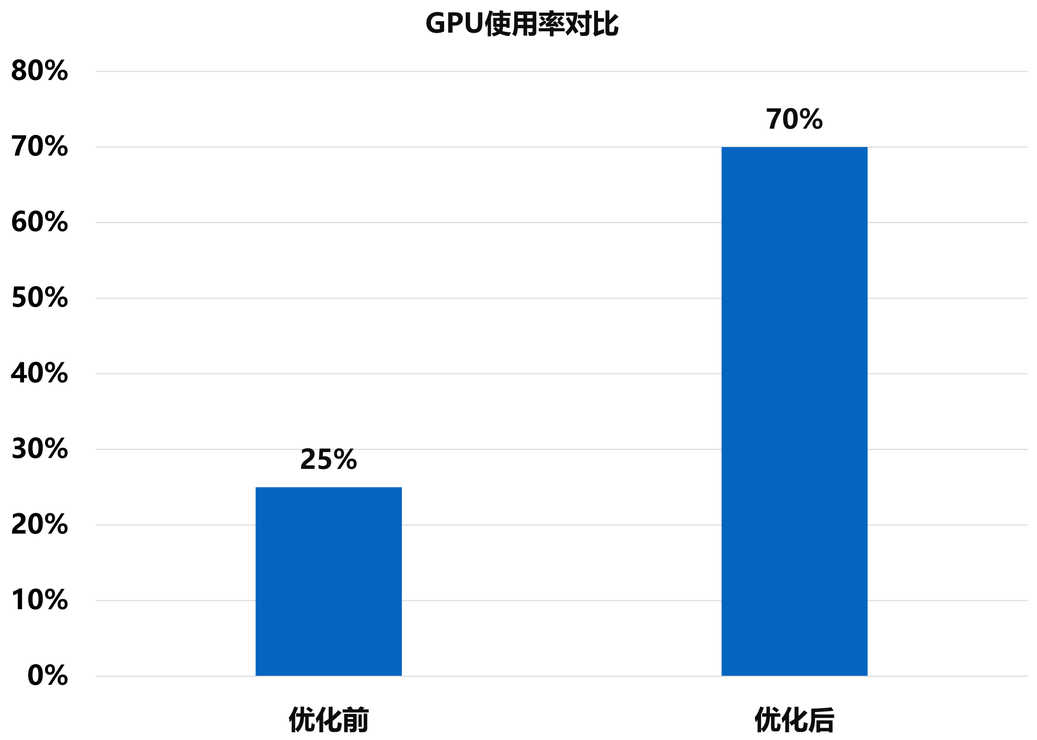




通过优化图片预处理方式，可以有效的提高GPU资源的利用率，优化后GPU的使用率提升到90%左右。

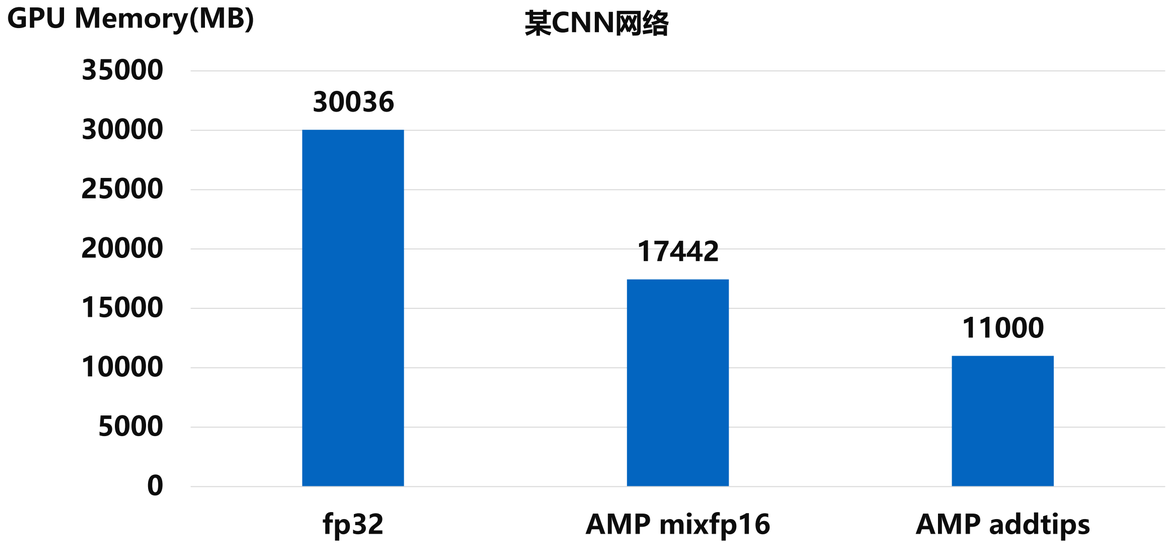
数据IO优化案例2

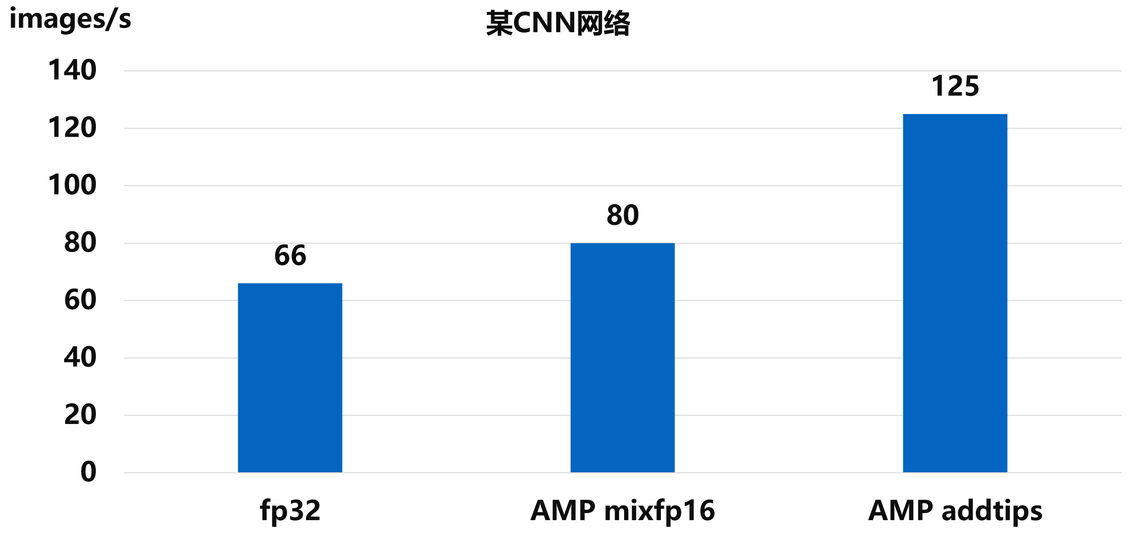
- 应用：基于双向LSTM模型在V100训练文本分类模型
- 问题分析
  - GPU核心使用率低，10%~40%
  - 高维嵌入embedding_looup显存溢出
- 优化方法
  - 数据IO优化：数据修改为TFRECORD，构建基于TFDATA的数据读入模式，C++多进程处理及读入数据
  - 显存优化：将预加载的词典从GPU显存加载到CPU内存，在内存中进行embedding



混合精度优化

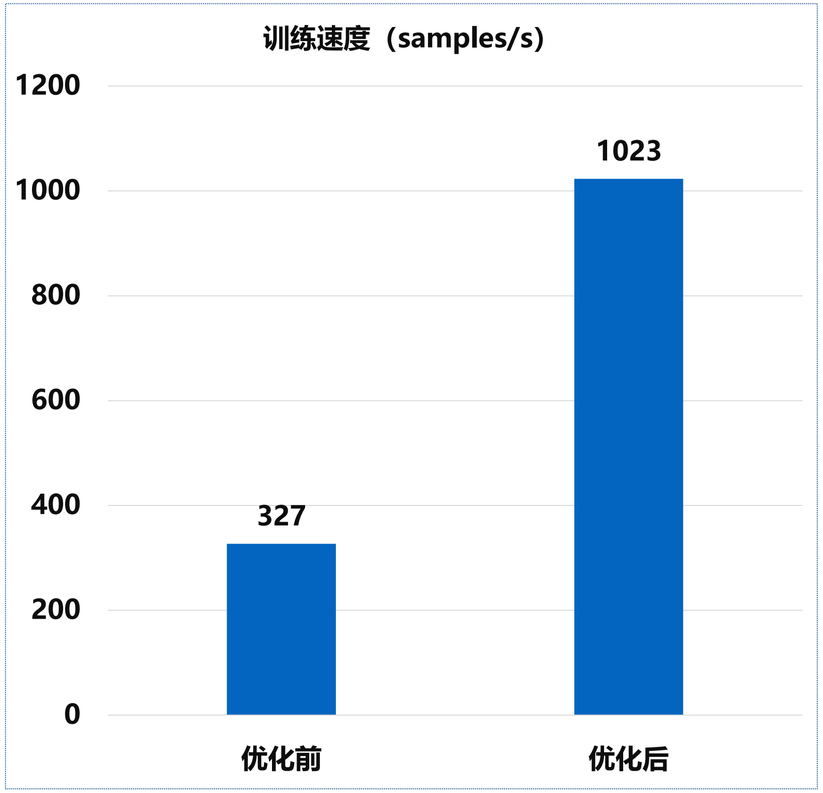
- 混合精度优化
  - 张量维度与TesorCore维度的适配，矩阵通道为8的倍数
  - 矩阵计算的OP融合
  - 增加矩阵乘加计算的比例





混合精度优化案例：影评分类

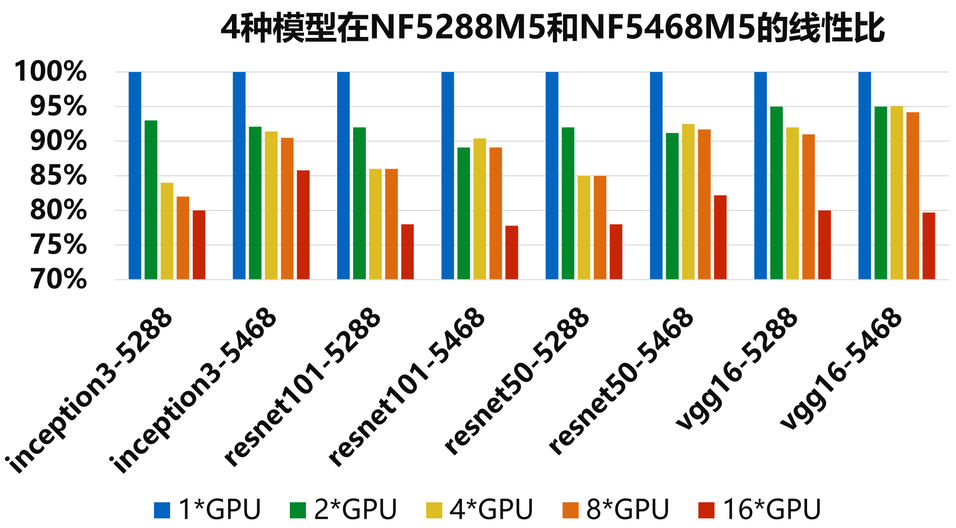
- 应用：影评分类
- 问题分析
  - Attention模块采用两次循环生成单样本的attention向量
  - 循环一次产生众多中间量及分散的OP处理
- 优化方法
  - Attention模块采用两次矩阵点乘生成attention向量
  - 矩阵乘采用cublas或其它矩阵库，操作高度融合，中间变量少

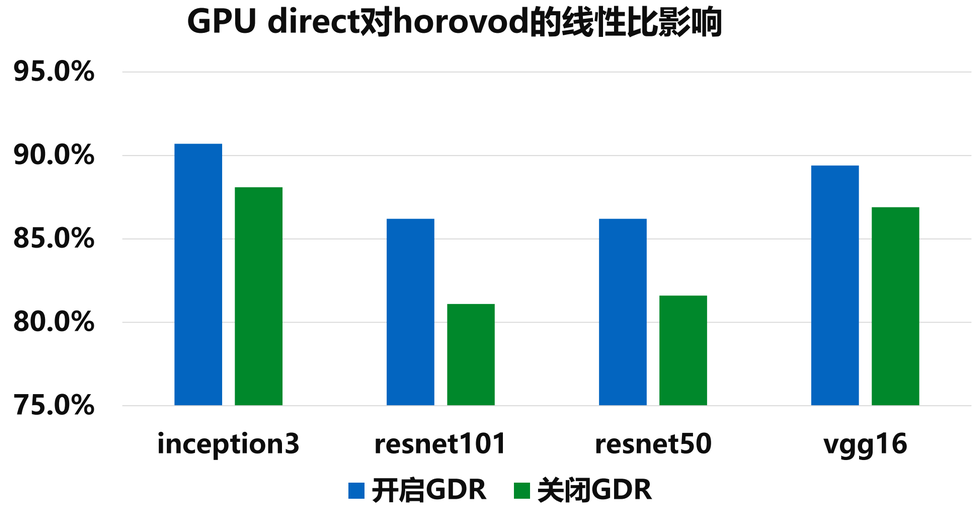


中文文本分词 → 文本嵌入 → CNN-ATTENTION模型 → 文本分类 → 影评分类

GPU并行优化

- 数据并行
  - PS/Ring-allreduce/Tree-allreduce
  - MPI/Horovod(Tensorfusion/AutoTuning/GDR Support)
  - NCCL/Blink（ “Blink:Fast and Generic Collectives for Distributed ML” ）
- 模型并行/数据+模型并行(Mesh-Tensorflow)
- PipLine并行（Gpip@NIPS2019）

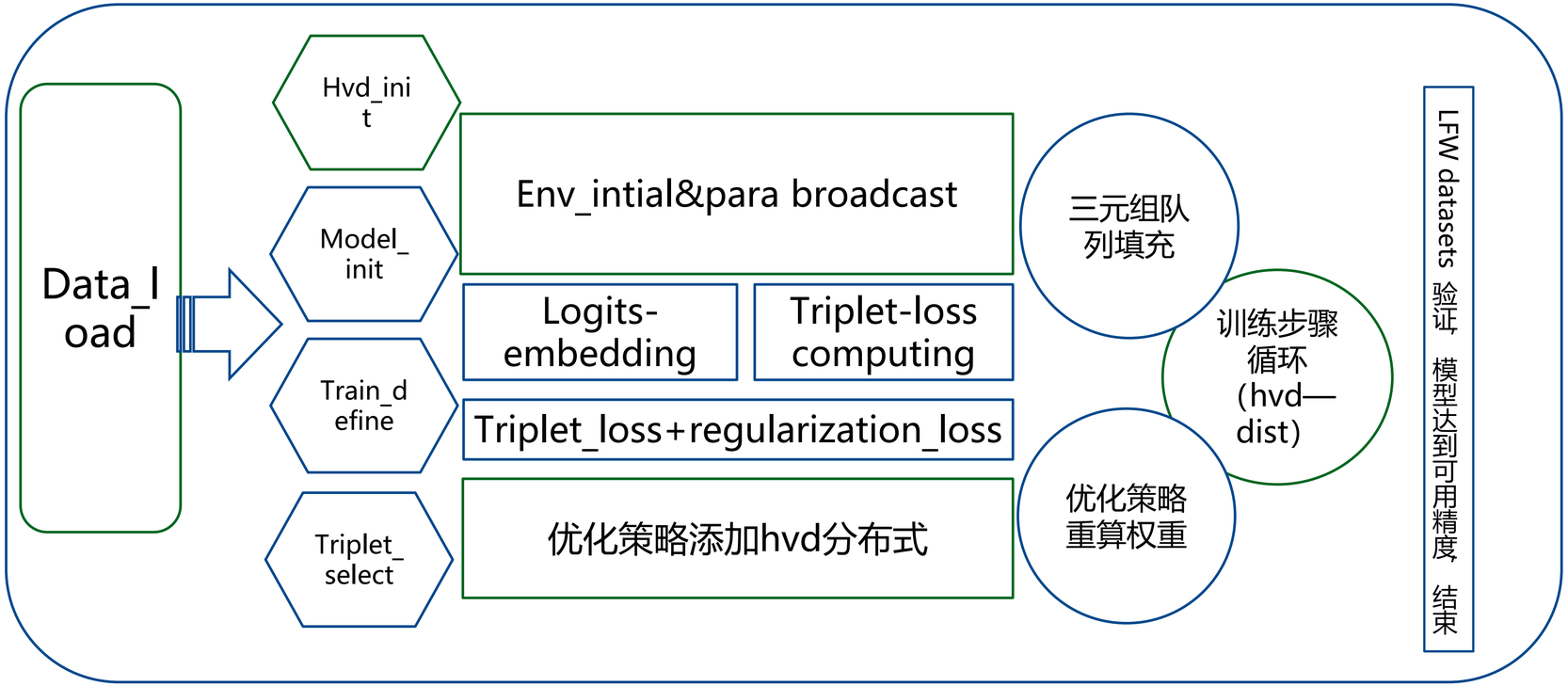




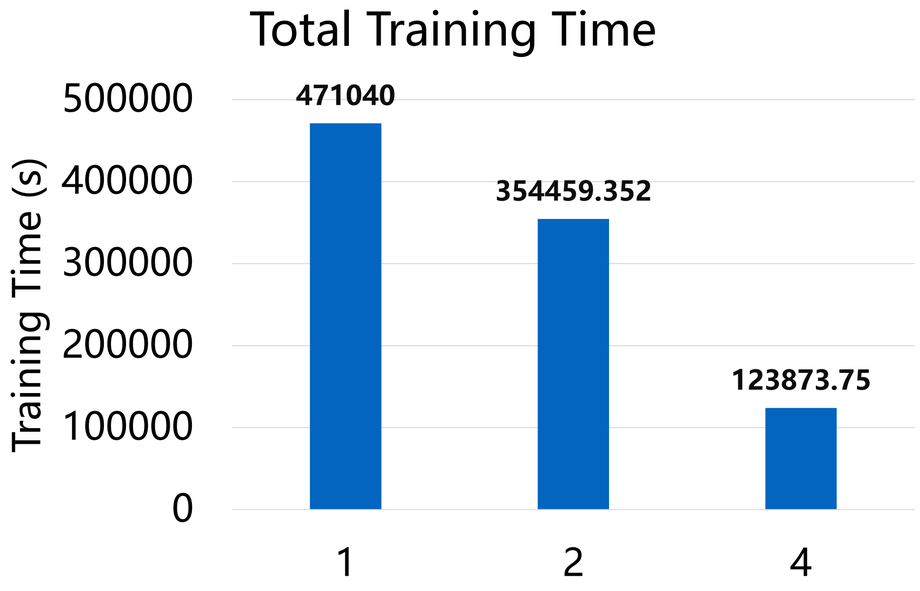
GPU并行优化案例1：大规模图像识别训练

- 应用：大规模图像识别
- 问题分析
  - 单GPU卡训练，40万图像需训练10h
  - 未来训练数据将达到10亿级，存在巨大的计算挑战
- 优化方法：
  - 使用Horovod的TF并行模式实现
  - 调整学习率以适应大规模训练的梯度调整　lr×$\sqrt{hvd.size()}$
  - 梯度融合通信
  - 混合精度训练

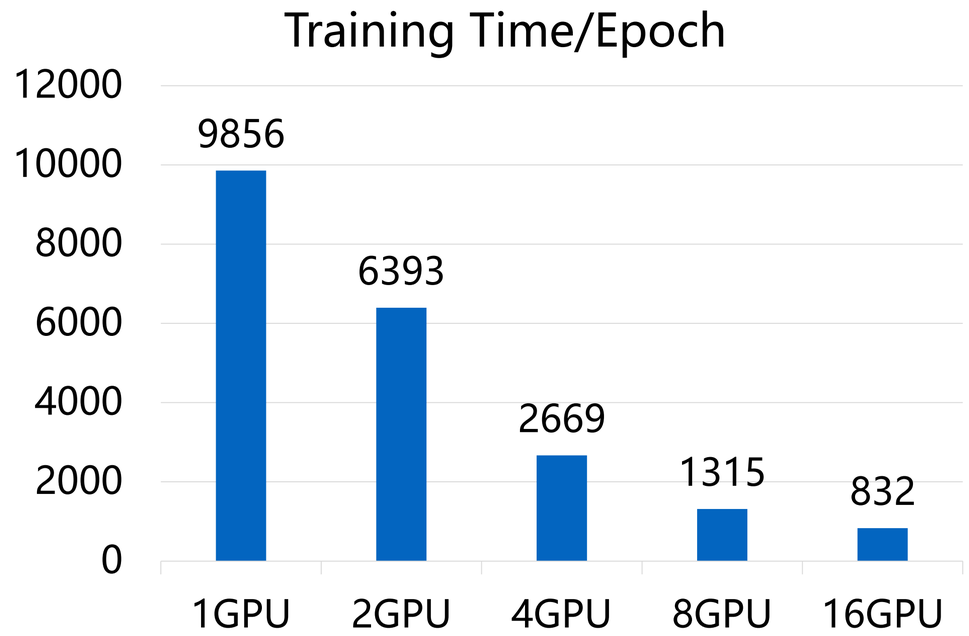
基于Horovod的CNN训练实现流程
✓分布式GPU环境初始化
✓将GPU0的全局变量broadcast到所有GPU
✓进行GPU分布式训练



大规模图像识别训练GPU并行优化效果

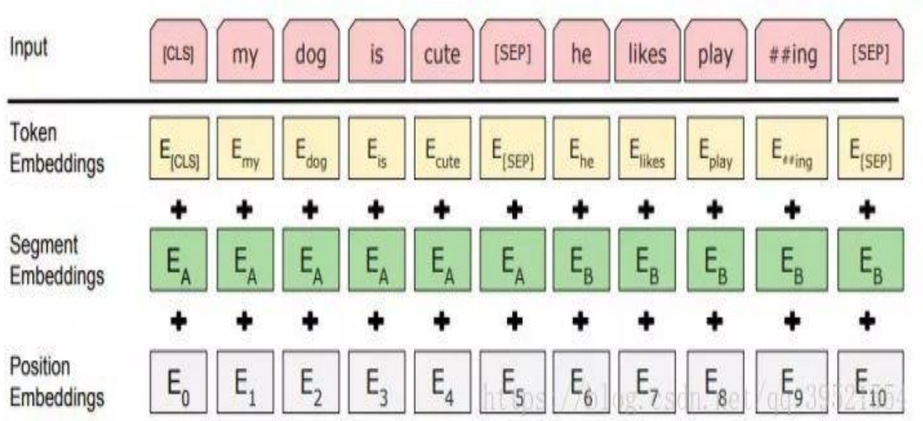


✓ 4卡扩展效率近线性，16卡扩展效率近14倍



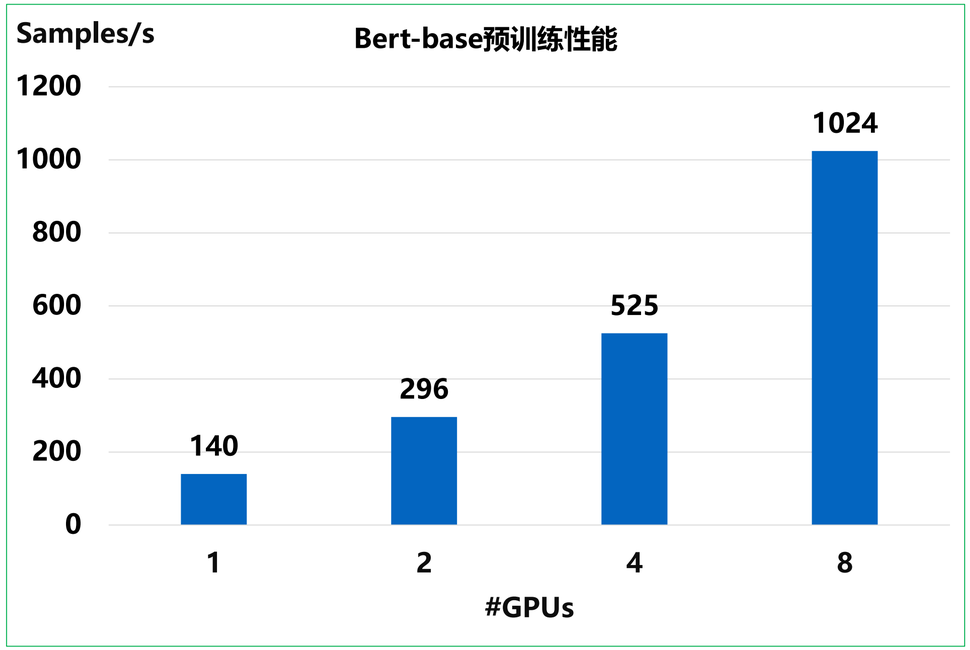
GPU并行优化案例2：Bert预训练

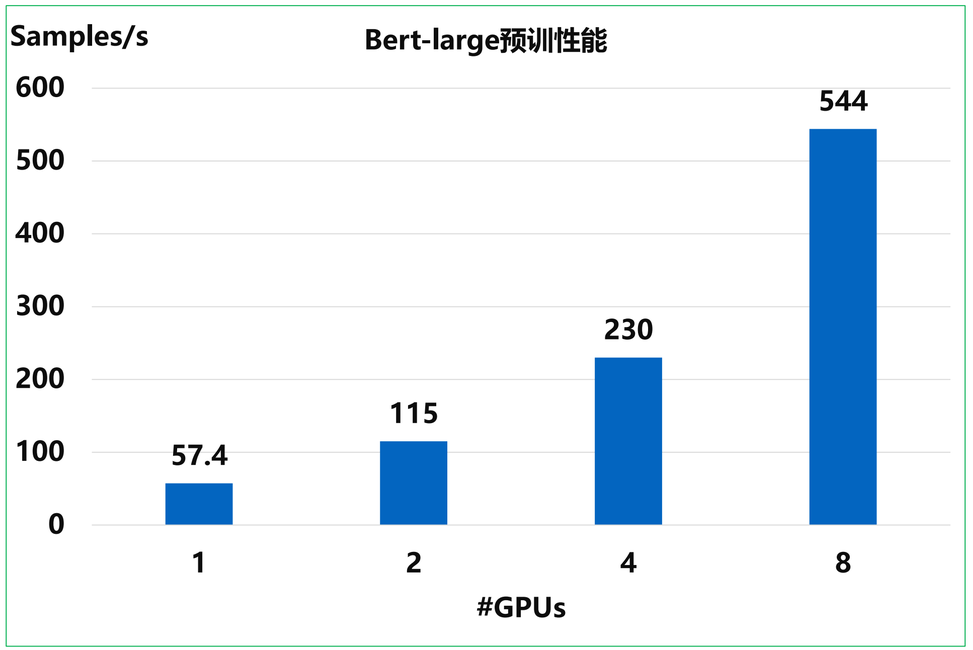
- 应用：Bert预训练
- 问题分析
  - 参数众多，收敛难度大，需要巨大的数据量
  - 单个V100-GPU在核心及带宽利用率都接近饱和的前提上述训练时间仍以年计算
- 优化方法：数据并行策略-基于Horovod
  - 变量在多进程中广播
  - BERT前向传播构建
  - 选择优化器，并将优化器定义为horovod并行模式
  - 启动多进程训练



BERT-base：L=12, H=768, A=12, Total Parameters=110M
BERT-large：L=24, H=1024, A=16, Total Parameters=340M

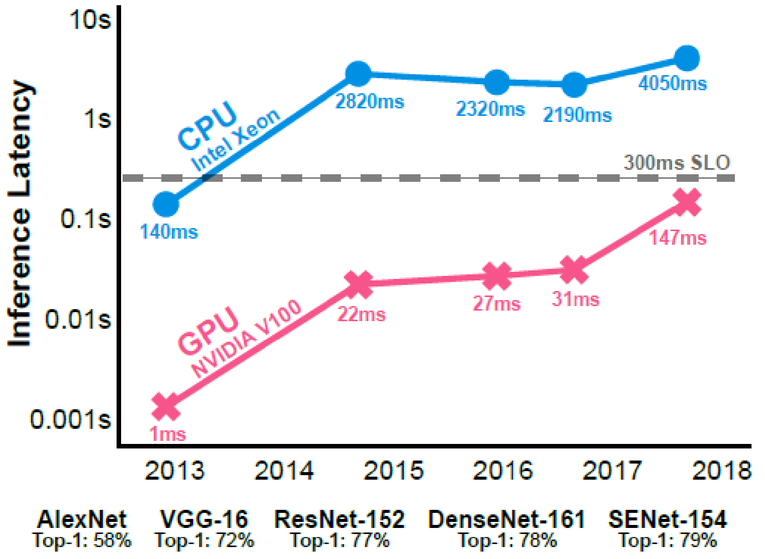
Bert预训练GPU并行优化效果

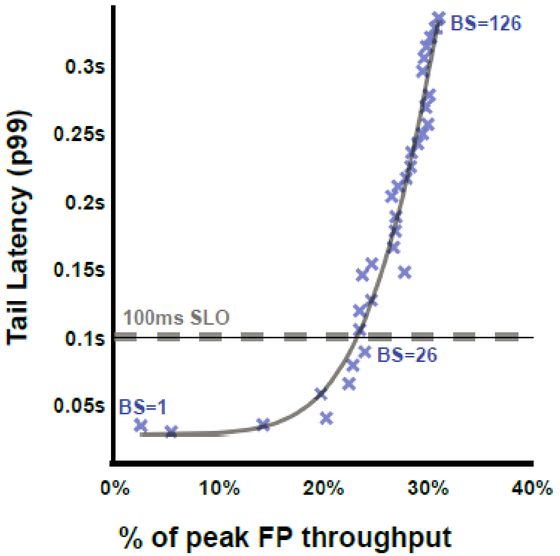




GPU应用推理优化方法

- 单模型单数据优化
  - 算法级：模型裁剪、量化、压缩、蒸馏
  - 工具级：推理引擎(算子融合、访存优化、量化)
- 单模型多数据并行
  - batch批量处理
- 多模型多数据并行
  - NVIDIA MPS，CUDA Streams

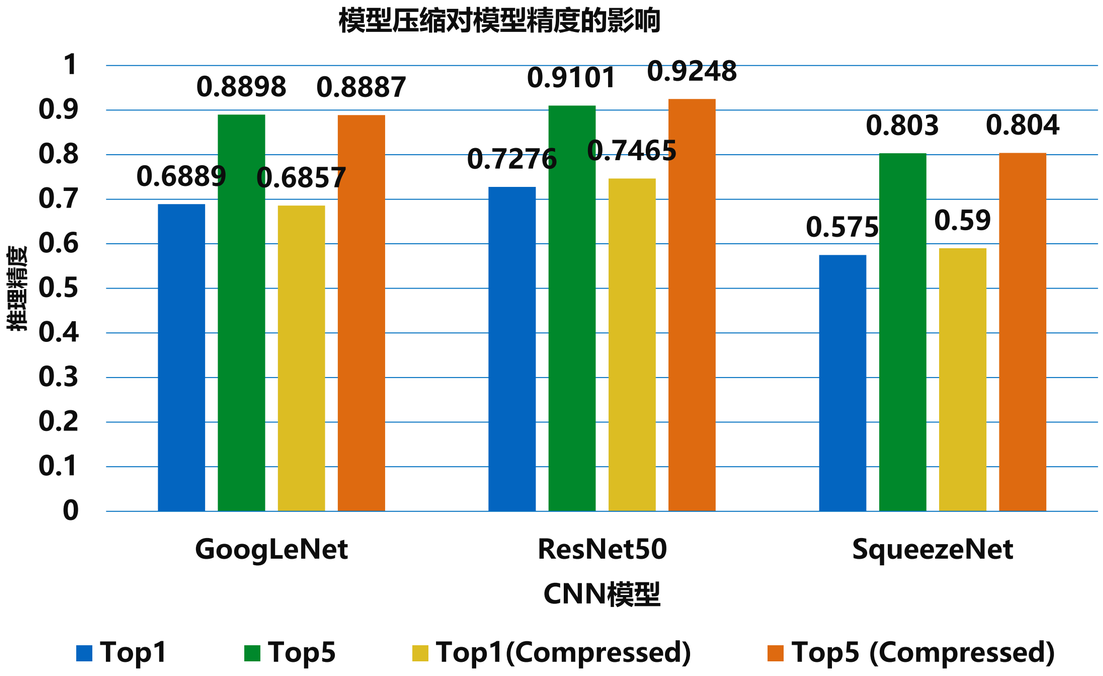




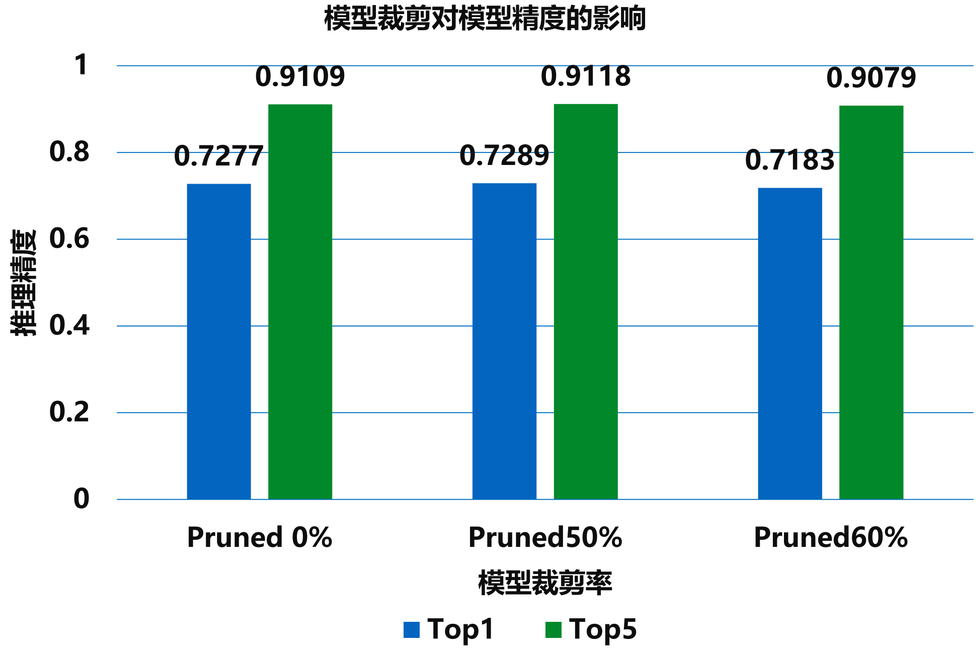
Ref: “Dynamic Space-Time Scheduling for GPU Inference”

算法级优化：模型裁剪与压缩

•网络模型从32位浮点数到4位整数，压缩为1/8

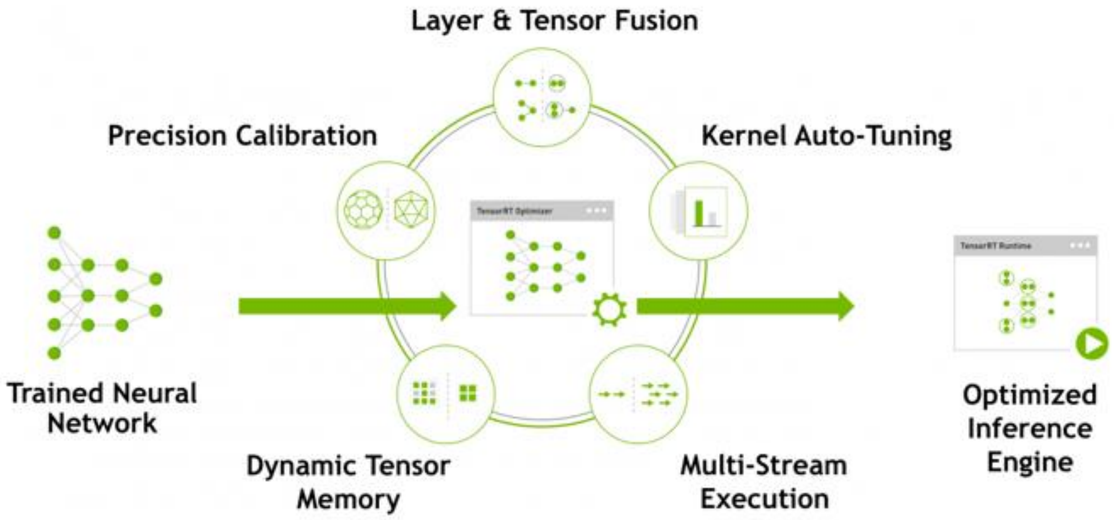


•模型数据计算量减少1/2，不改变计算结构

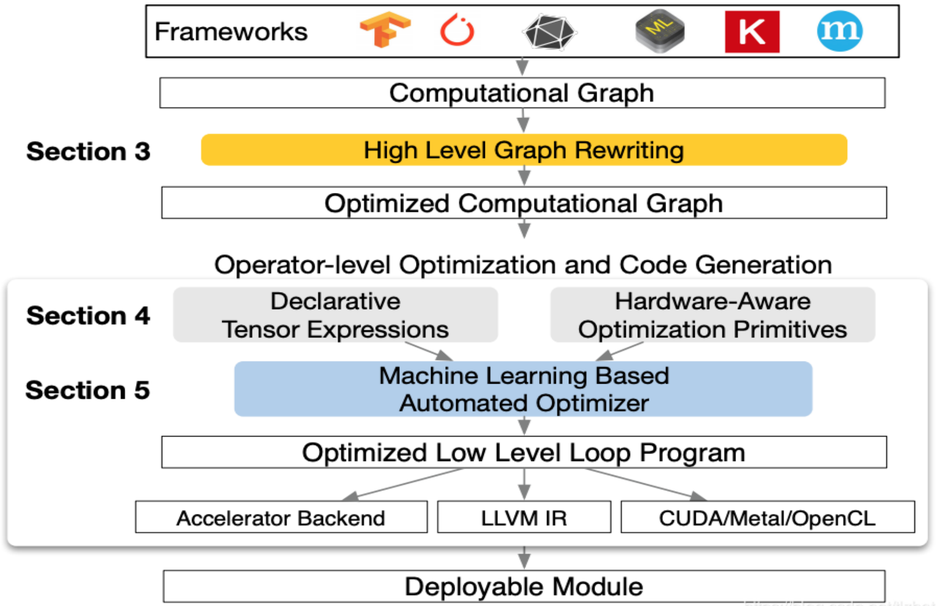


工具级优化：推理引擎

TensorRT



TVM



提纲

AI计算的发展趋势及其挑战

基于GPU的AI计算优化方法：从训练到推理

Case Study:基于GPU实现AutoML Suite计算优化

AutoML技术

- 人工AI建模过程

原始数据

数据处理

- 数据清洗
- 数据增强
- 文本编码
- 数据分割

特征工程

- 特征选择
- 特征迁移
- 序列嵌入
- 图像嵌入

模型设计

- SVM
- MLP
- CNN
- RNN

模型验证

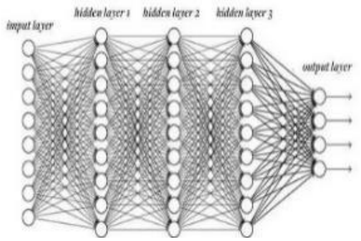
- 混淆矩阵
- P/R测试

- AutoML技术：是采用自动化地方式和程式化的手段，根据开发任务自主地实现模型构建、筛选的一种技术

视觉图片数据

AutoML

高效模型搜索

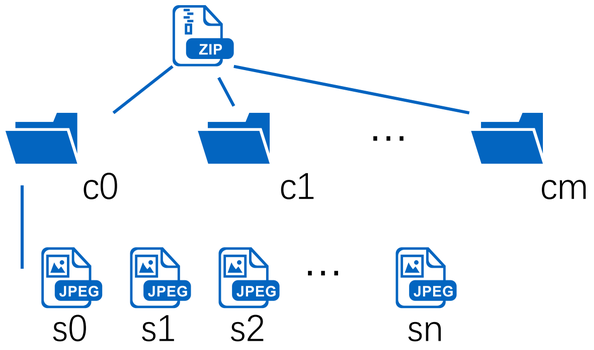


模型部署

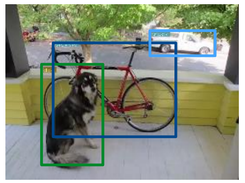
浪潮AutoML Suite

- 企业级一站式模型自动构建系统
  - 本地化On-Premise部署，一站可视化处理
  - 自动搜索图像分类、目标检测模型
  - 多机多GPU卡高效并行模型搜索
  - 按部署端所需的计算量、内存输出模型

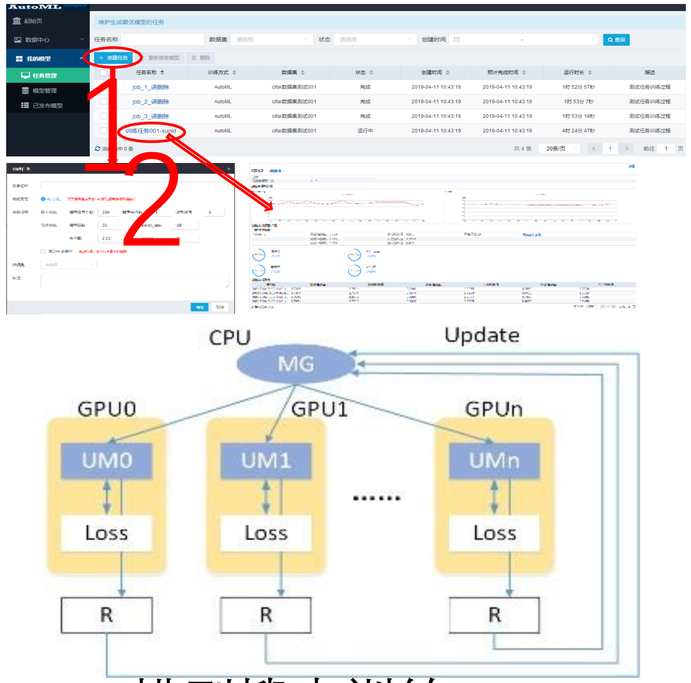
图像分类



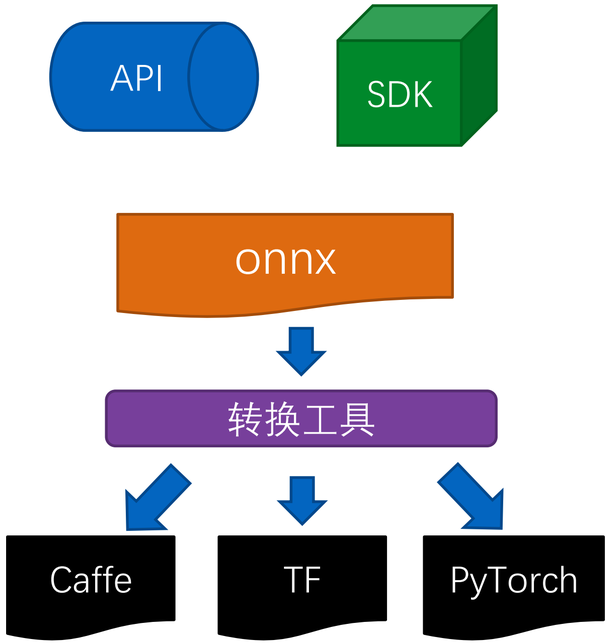
目标检测



数据准备



模型搜索训练



模型部署

AutoML应用案例

- 智慧城市路口车辆颜色分类应用
  - 挑战：各个路口特征不一，人工设计的模型相对通用，无法泛化到各个路口，而针对设计，需大量的模型开发工作
  - 数据：某路口采集16类车辆颜色数据，共40万张图片，进行AutoML建模与人工建模对比

不同模型在当前路口测试集分类精度对比

模型	平均	白天	晚上
Base模型（专家）	94.7	95.7	93.7
VCCNet17（AutoML）	94.6	95.99	93.15

新整理的测试集，测试两个模型的泛化能力

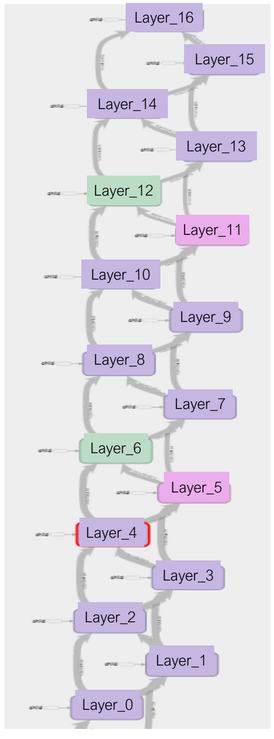
模型	平均	白天	夜晚
Base模型（专家）	84.9	91.3	77.5
VCCNet17（AutoML）	87.8	91.5	83.6

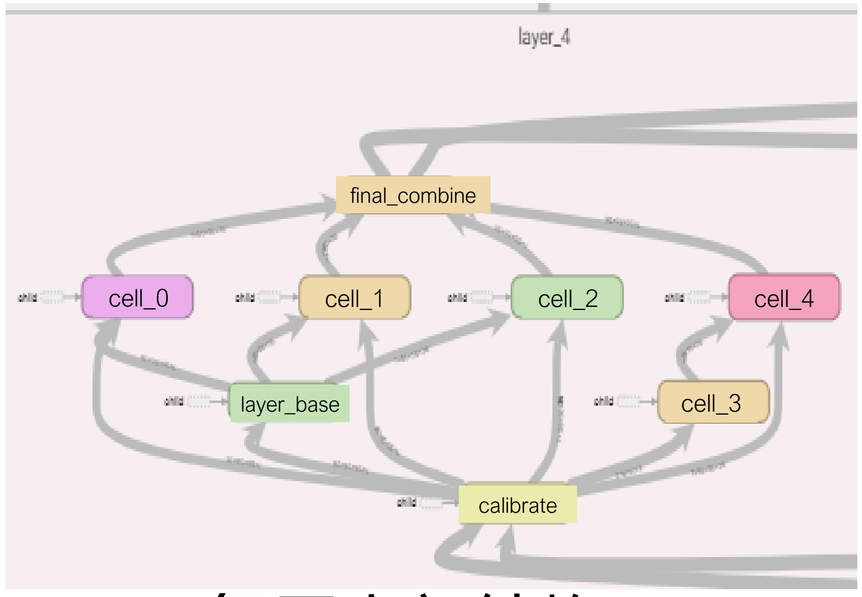
AutoML实际应用面临的计算性能挑战

- 单机单卡运行，搜索训练慢
- 搜索的最佳模型的层数和分支过多，推理慢

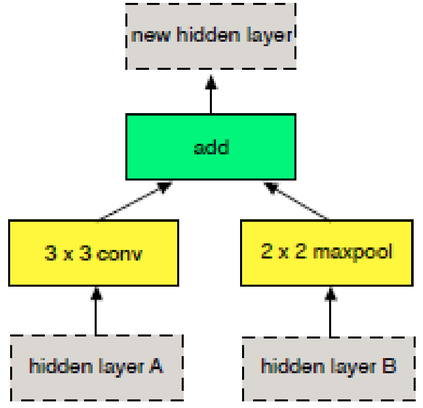
数据量	GPU数目	搜索模型数目	搜索耗时（h）	最佳平均模型精度	训练耗时（h）	模型前向耗时(ms/张)
Train:35.7万	1	2000	54	94.6	196	59ms

模型结构



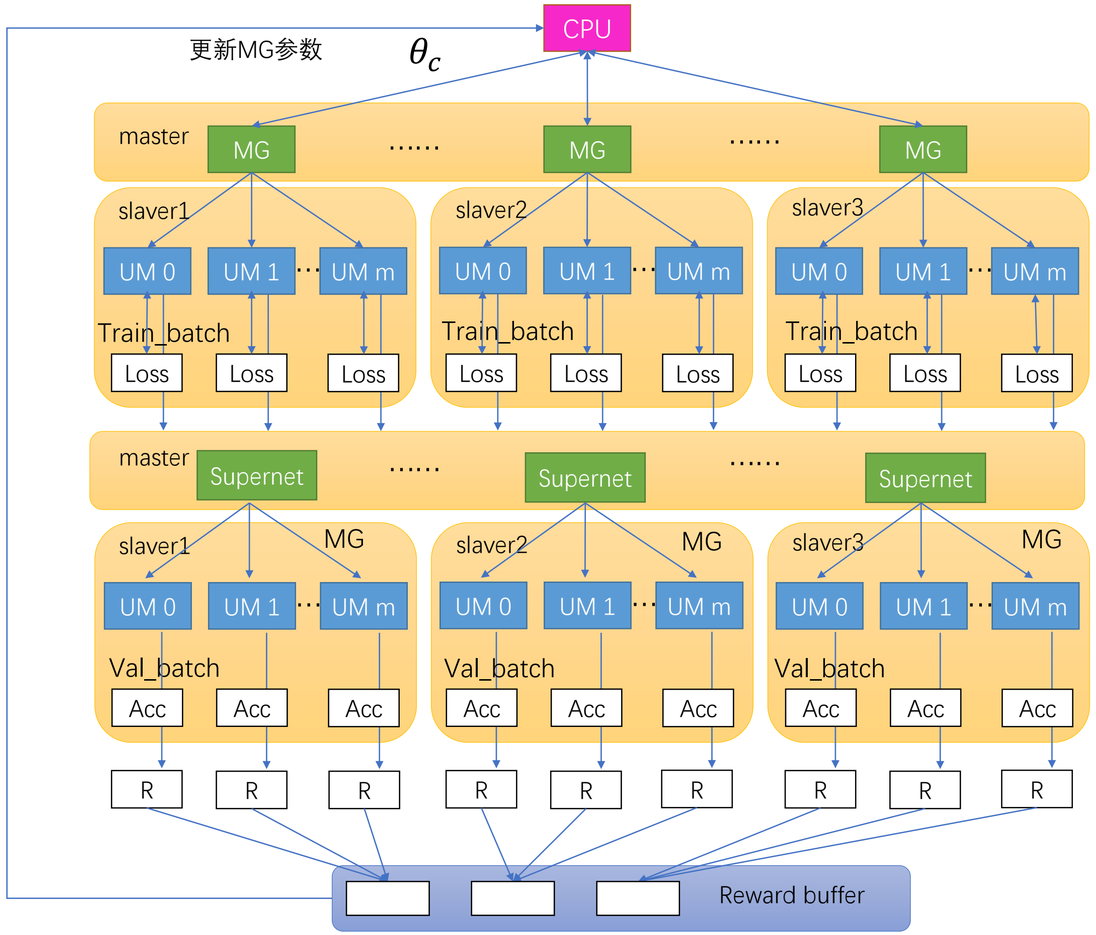


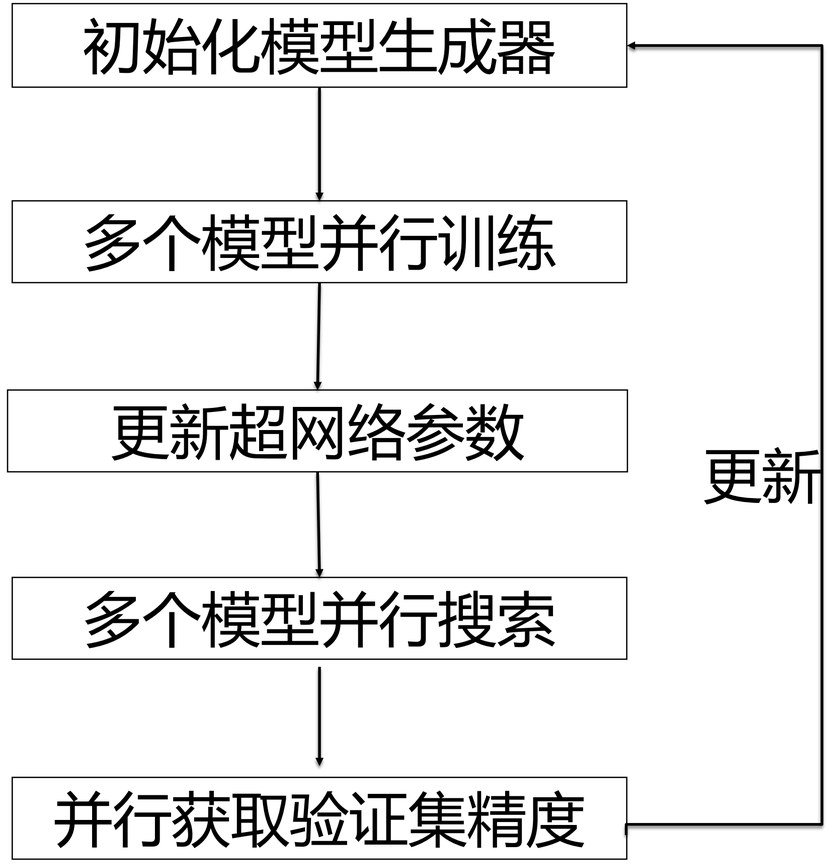
每层内部结构



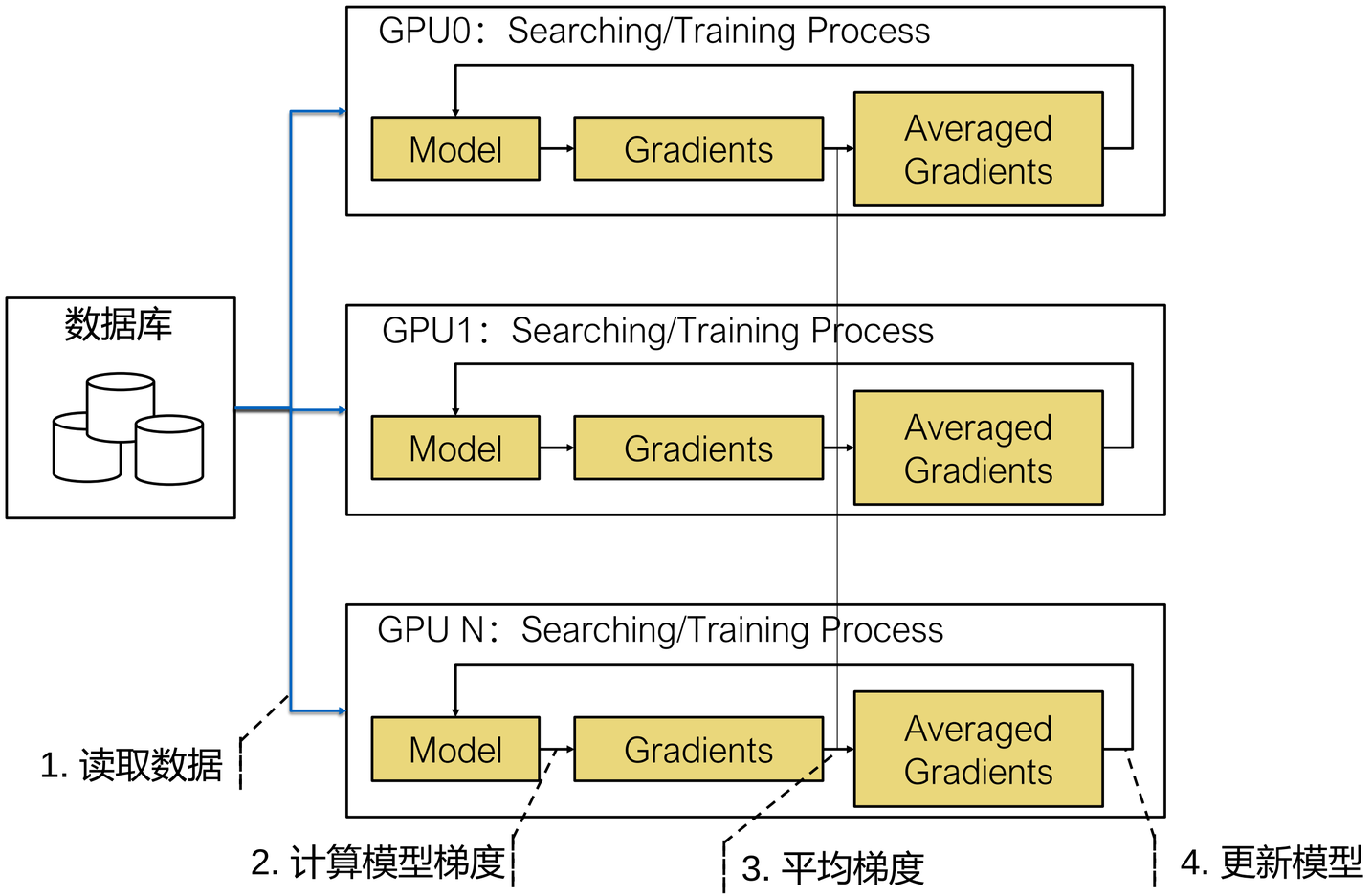
每个cell内部结构

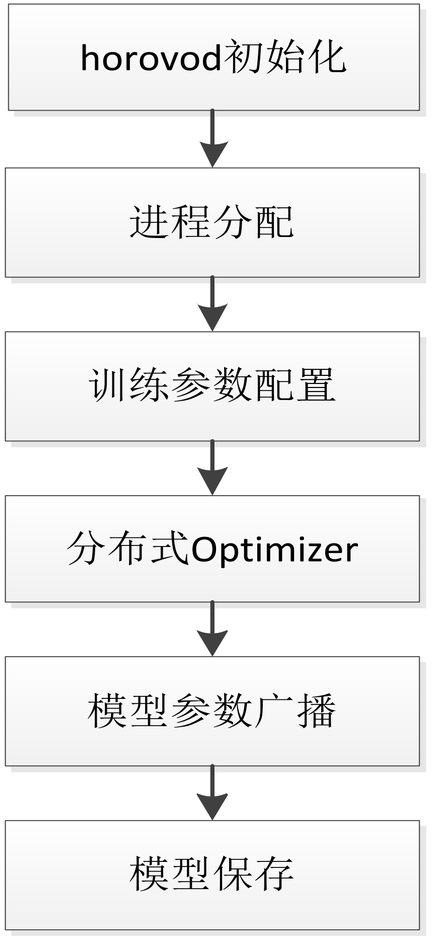
AutoML Suite：模型搜索并行





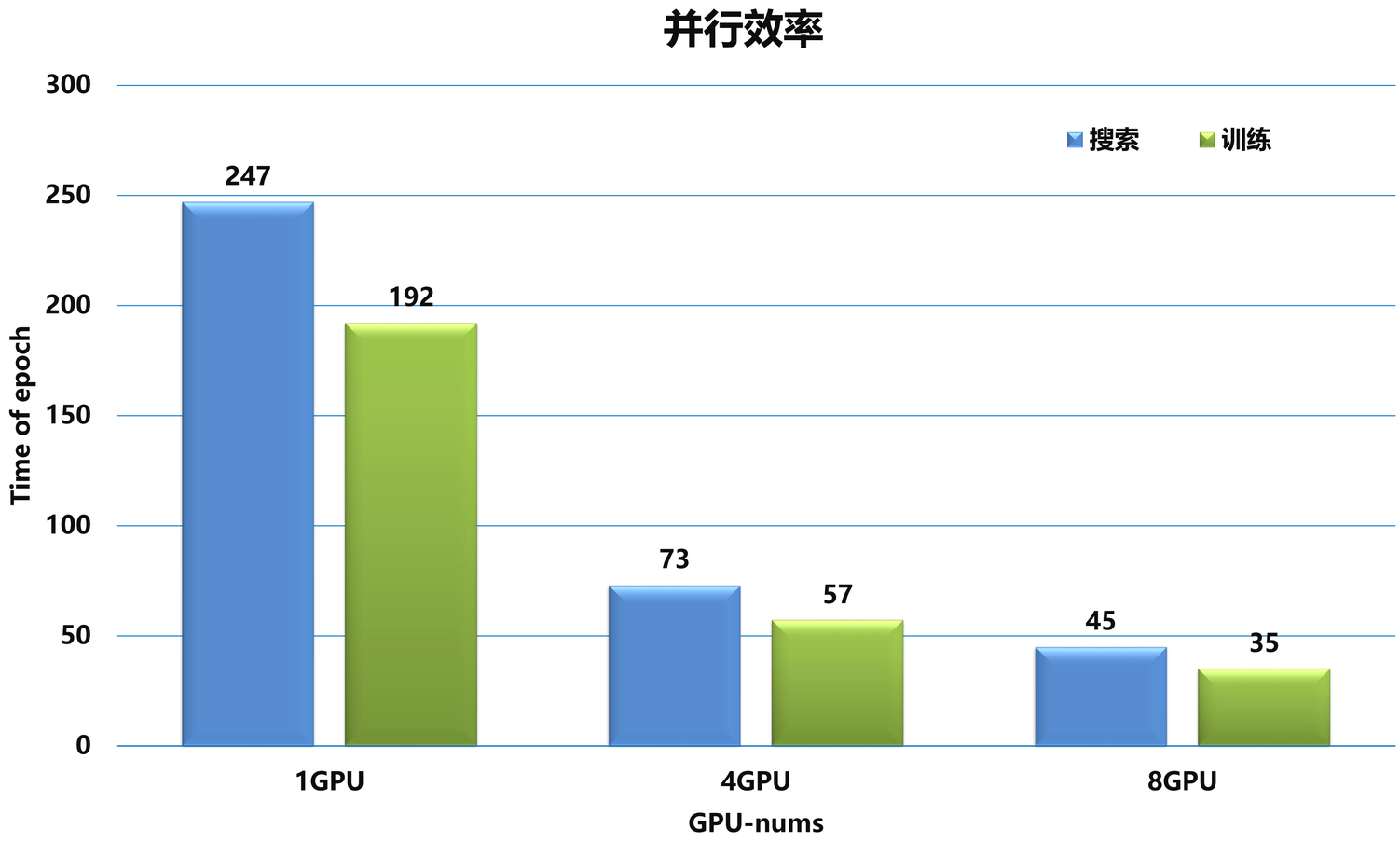
AutoML Suite：模型训练并行





模型搜索和训练并行效果

- 基于cifar10的AutoML在搜索和训练中平均每个epoch耗时



模型推理优化

- 分支裁剪，由5个分支减少到3个
- 层数减少，由15层减少到12层

模型类型	精度 /100%	FLOPs /G	参数量 /M	前向耗时 /ms
Base (专家)	94.7	1.54	5.18	--
Automl (15层，5分支)	94.6	1.81	5.26	50.67
Automl (12层，3分支)	94.23	1.04	3.58	30.05

总结

- AI应用算法的效果仍需要大量的计算支撑，而且未来推理计算需求将超过训练
- AI计算面临巨大的挑战，特别是针对单一模型的千级GPU卡并行训练及生产系统千级模型场景的并发使用，其面临异构计算架构的效率、计算规模扩展、复杂计算环境的管理与优化的挑战
- 基于GPU的AI计算优化，需从系统的角度考虑，从应用的特征出发，设计合理的GPU平台、优化GPU资源、AI框架工具、并从训练与推理应用的程序级实现更细粒度的优化，才能取得更好的性能、扩展性、吞吐与延时

Thanks!

Thanks!

Thanks!