AiCon
全球人工智能与机器学习技术大会

深度学习框架技术剖析

袁进辉

北京一流科技有限公司
ONEFLOW TECHNOLOGY LIMITED

主办方 Geekbang 极客邦科技 InfoQ

极客时间

重拾极客精神·提升技术认知

下载极客时间App

获取有声IT新闻、技术产品专栏，每日更新

扫一扫下载极客时间App

人工智能基础课

“通俗易懂的人工智能入门课”

王天一
博士 副教授

扫一扫，免费试读

AI技术内参

你的360度人工智能信息助理

洪亮劼
Etsy 数据科学主管

扫一扫，免费试读

AI 前线

关注落地技术，探寻AI应用场景

- 14万AI领域垂直用户
- 8000+社群技术交流人员，不乏行业内顶级技术专家
- 每周一节干货技术分享课
- AI一线领军人物的访谈
- AI大会的专家干货演讲整理
- 《AI前线》月刊
- AI技能图谱
- 线下沙龙

扫码关注带你涨姿势

Geekbang. 极客邦科技 | InfoQ

ArchSummit 全球架构师峰会

2018 · 深圳站

从2012年开始算起，InfoQ已经举办了9场ArchSummit全球架构师峰会，有来自Microsoft、Google、Facebook、Twitter、LinkedIn、阿里巴巴、腾讯、百度等技术专家分享过他们的实践经验，至今累计已经为中国技术人奉上了近千场精彩演讲。

限时7折报名中，名额有限，速速报名吧！

2012.08.10-12 深圳站

2018.07.06-09 深圳站

会议：07.06-07.07

培训：07.08-07.09

TABLE OF CONTENTES

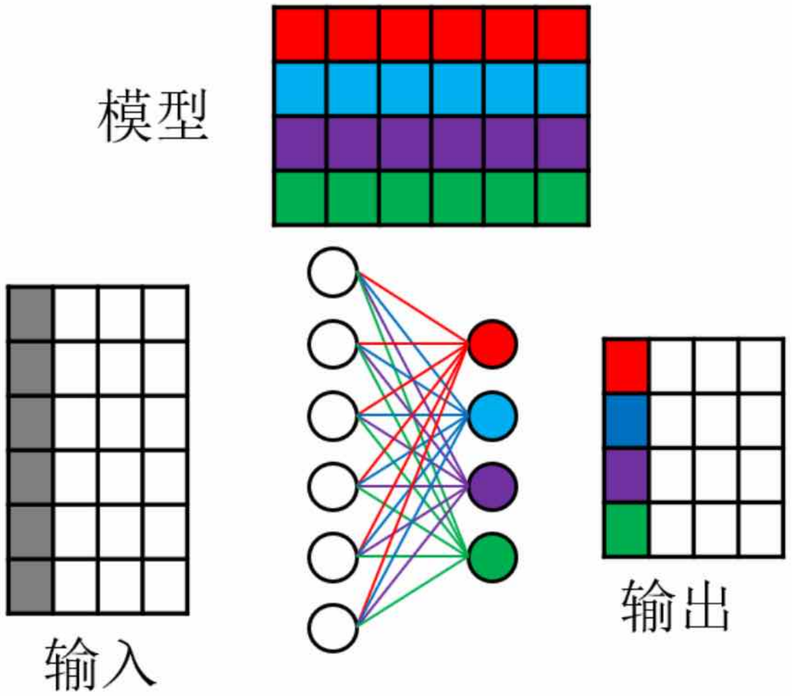
前提

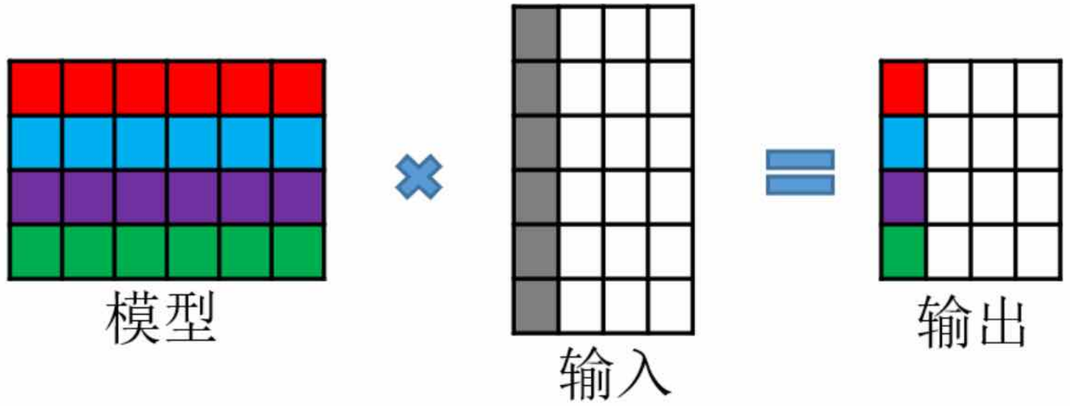
- 深度学习当前已经取得巨大成功，未来还有旺盛生命力
- 计算力在深度学习中扮演了重要角色（计算力与算法）
- 深度学习软件和硬件至少一样重要（软件与硬件）
- 计算力的瓶颈在分布式框架（宏观与微观）
- 深度学习软件解决编程不够快和程序运行不够快两个痛点

《深度学习平台技术演进》


GIAC | 全球互联网架构大会
GLOBAL INTERNET ARCHITECTURE CONFERENCE
深度学习平台技术演进
袁进辉
北京一流科技有限公司
ONEFLOW TECHNOLOGY LIMITED
2017.thegiac.com
OneFlow

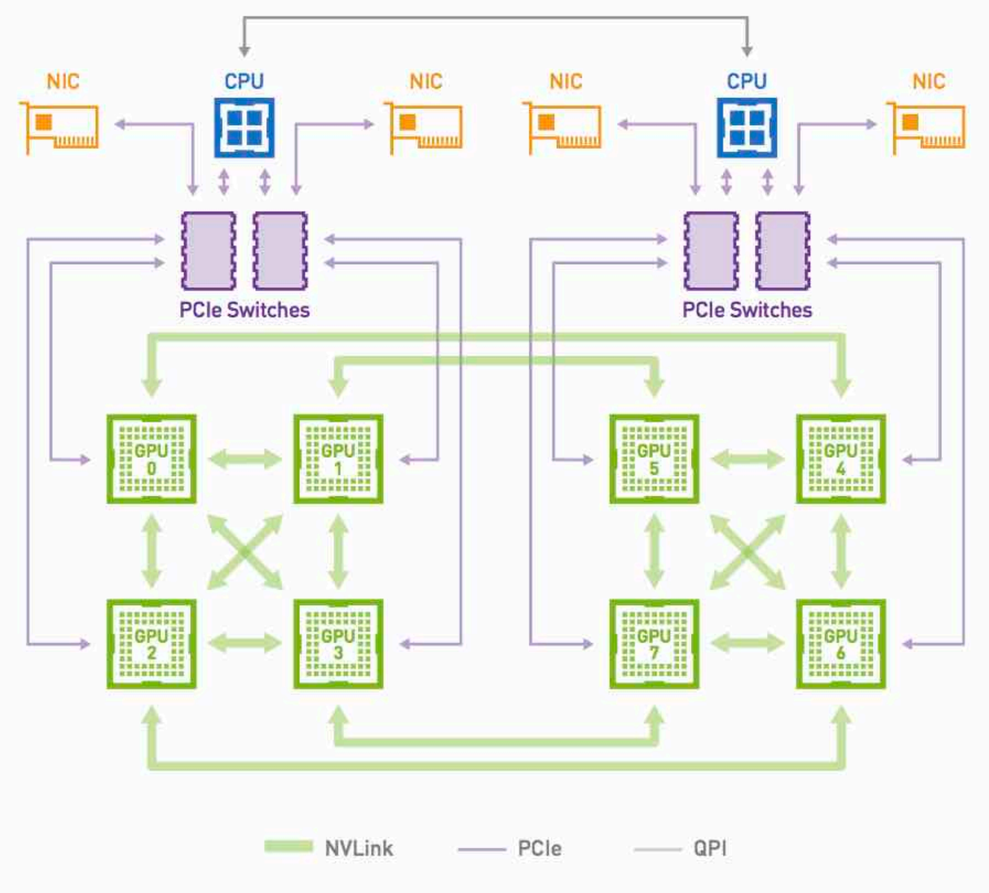
神经网络的矩阵表示





基于矩阵乘积表示的表示变换

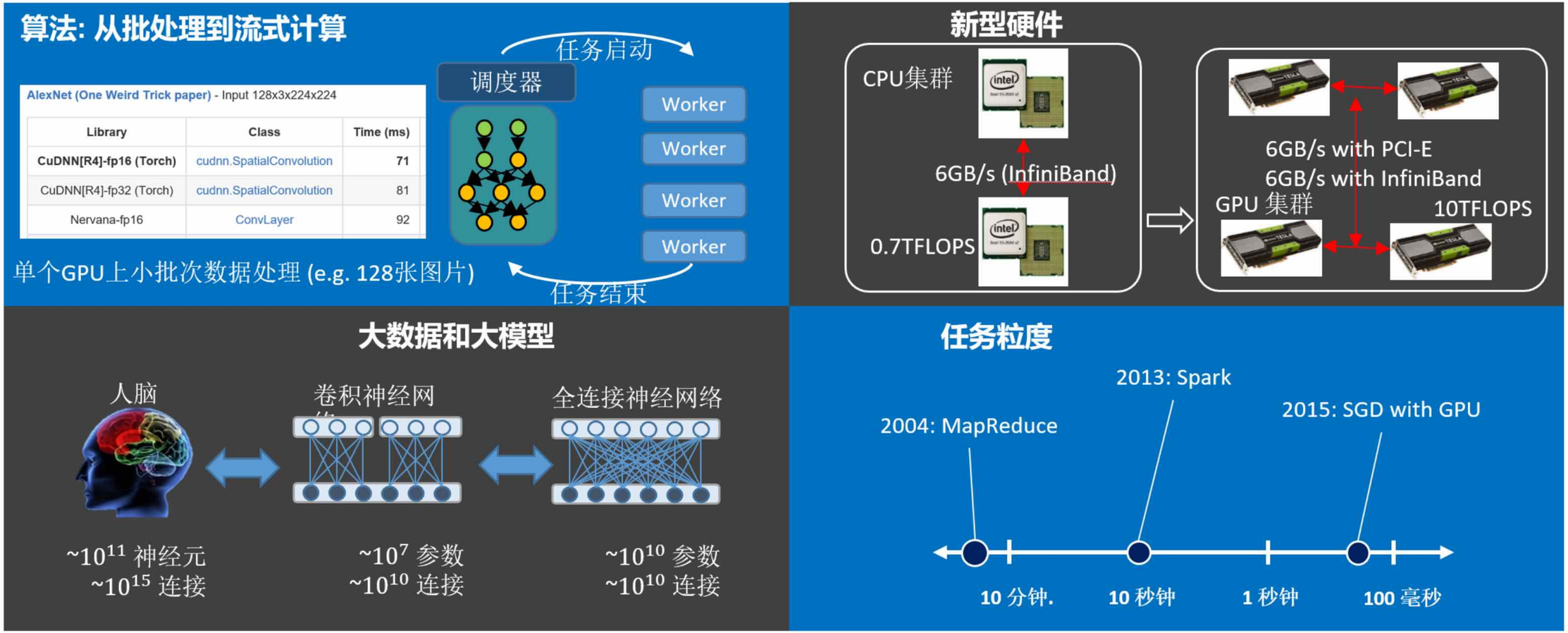
高速互联实现硬件扩展性



GPU 集群

TPU 集群

硬件越快，软件越难


算法: 从批处理到流式计算
AlexNet (One Weird Trick paper) - Input 128x3x224x224
Library | Class | Time (ms)
CuDNN[R4]-fp16 (Torch) | cudnn.SpatialConvolution | 71
CuDNN[R4]-fp32 (Torch) | cudnn.SpatialConvolution | 81
Nervana-fp16 | ConvLayer | 92
任务启动
调度器
Worker
Worker
Worker
Worker
任务结束
单个GPU上小批次数据处理 (e.g. 128张图片)
新型硬件
CPU集群
6GB/s (InfiniBand)
0.7TFLOPS
6GB/s with PCI-E
6GB/s with InfiniBand
GPU 集群
10TFLOPS
大数据和大模型
人脑
卷积神经网
全连接神经网络
~10^11 神经元
~10^15 连接
~10^7 参数
~10^10 连接
~10^10 参数
~10^10 连接
任务粒度
2004: MapReduce
2013: Spark
2015: SGD with GPU
10 分钟.
10 秒钟
1 秒钟
100 毫秒

深度学习软件平台的定位

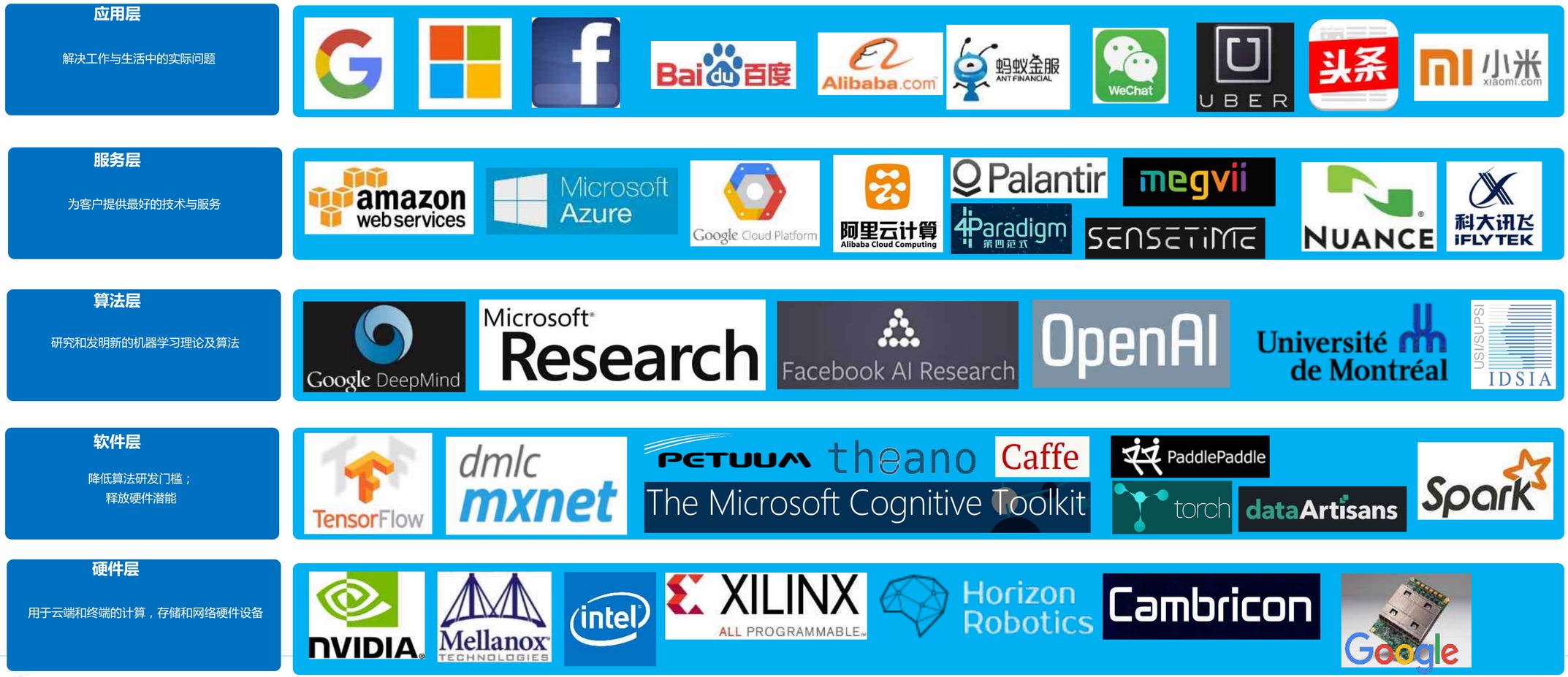


TABLE OF CONTENTES

控制流与数据流

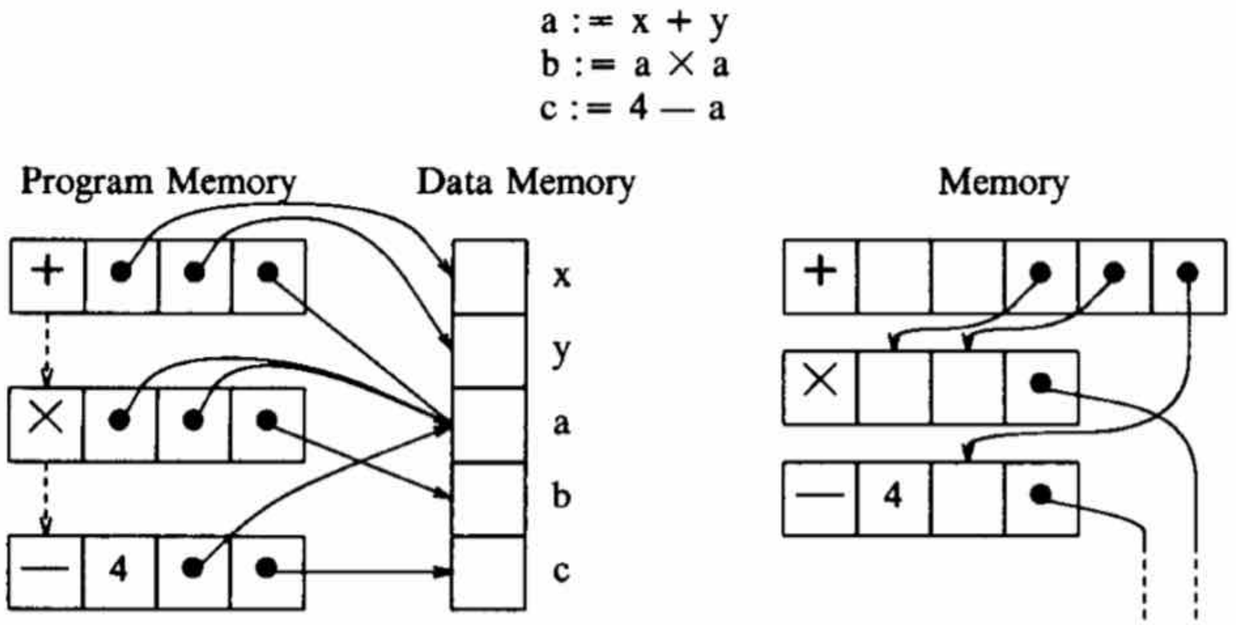
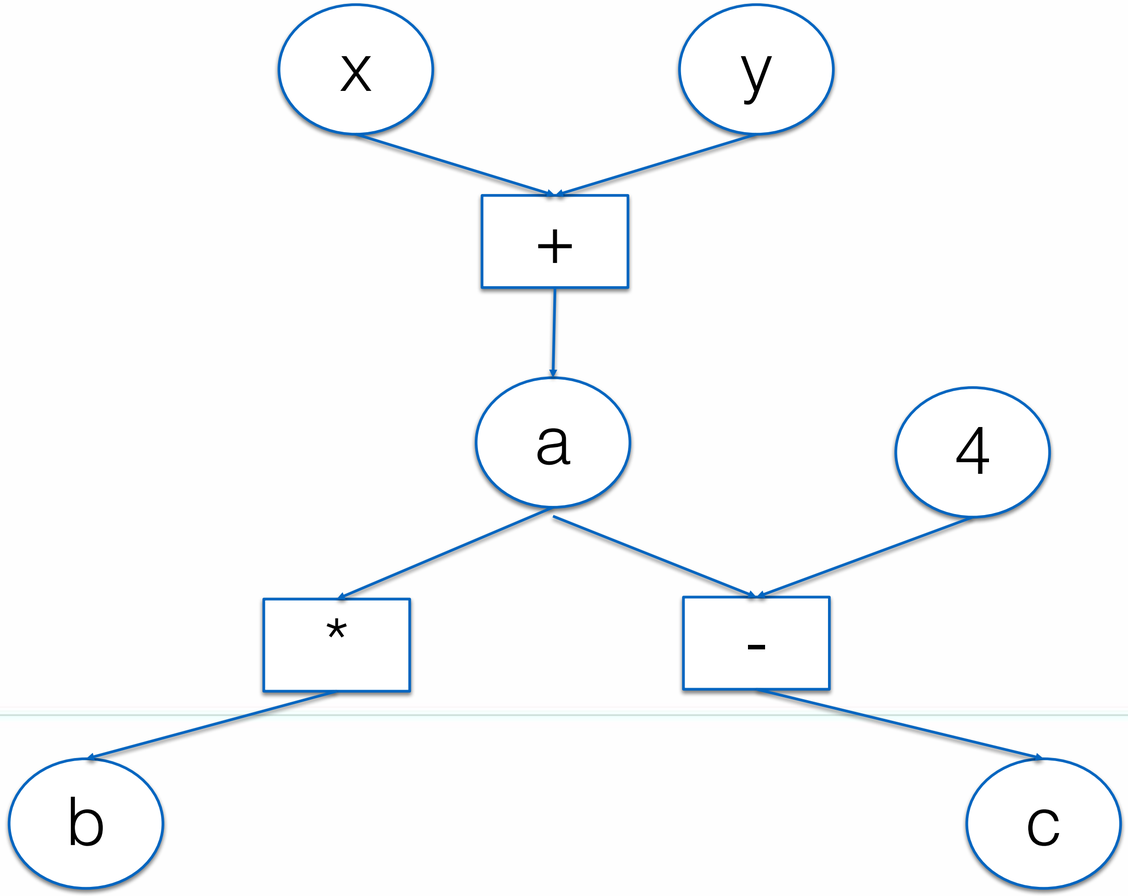


Figure 2. A comparison of control flow and dataflow programs. On the left a control flow program for a computer with memory-to-memory instructions. The arcs point to the locations of data that are to be used or created. Control flow arcs are indicated with dashed arrows; usually most of them are implicit. In the equivalent dataflow program on the right only one memory is involved. Each instruction contains pointers to all instructions that consume its results.

http://web.cecs.pdx.edu/~alaa/courses/ece588/winter2009/papers/veen_acmcs_1986.pdf

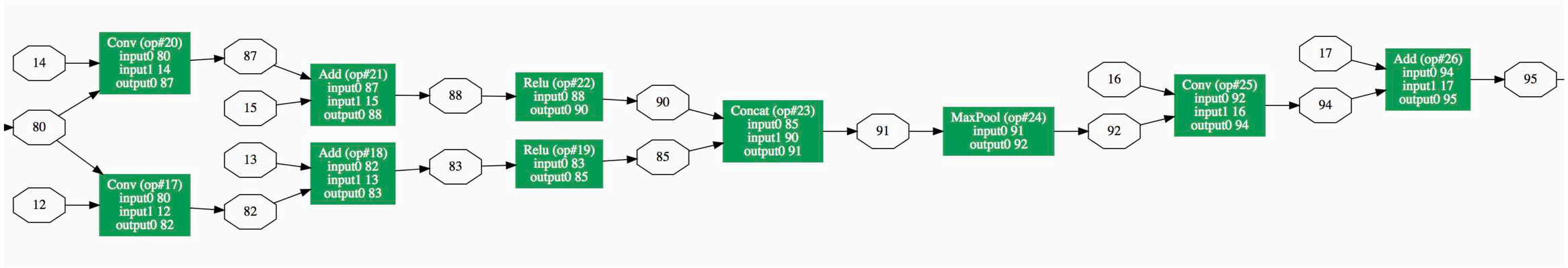
数据流：有向无环图

- 显式描述所有并行机会；简洁优雅的执行引擎



操作符与张量

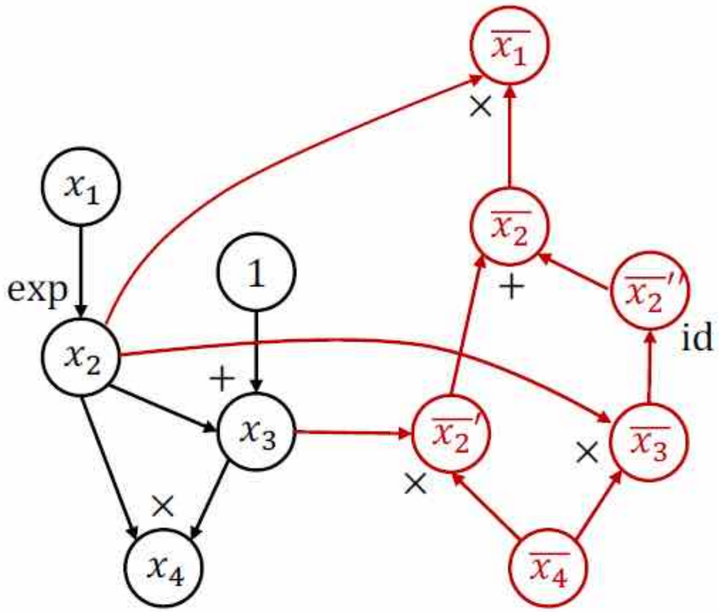
- 操作符：常见神经网络基本运算（Op/Kernel, 粒度问题）
- 数据：张量（Eigen, Mshadow）



https://github.com/onnx/tutorials/blob/master/tutorials/VisualizingAModel.md

自动梯度计算（autograd）

- 用户只须声明前向计算图，系统自动推导后向计算图
- 链式法则
- 依赖前向计算结果
- 多个消费者依赖同一数据时较tricky



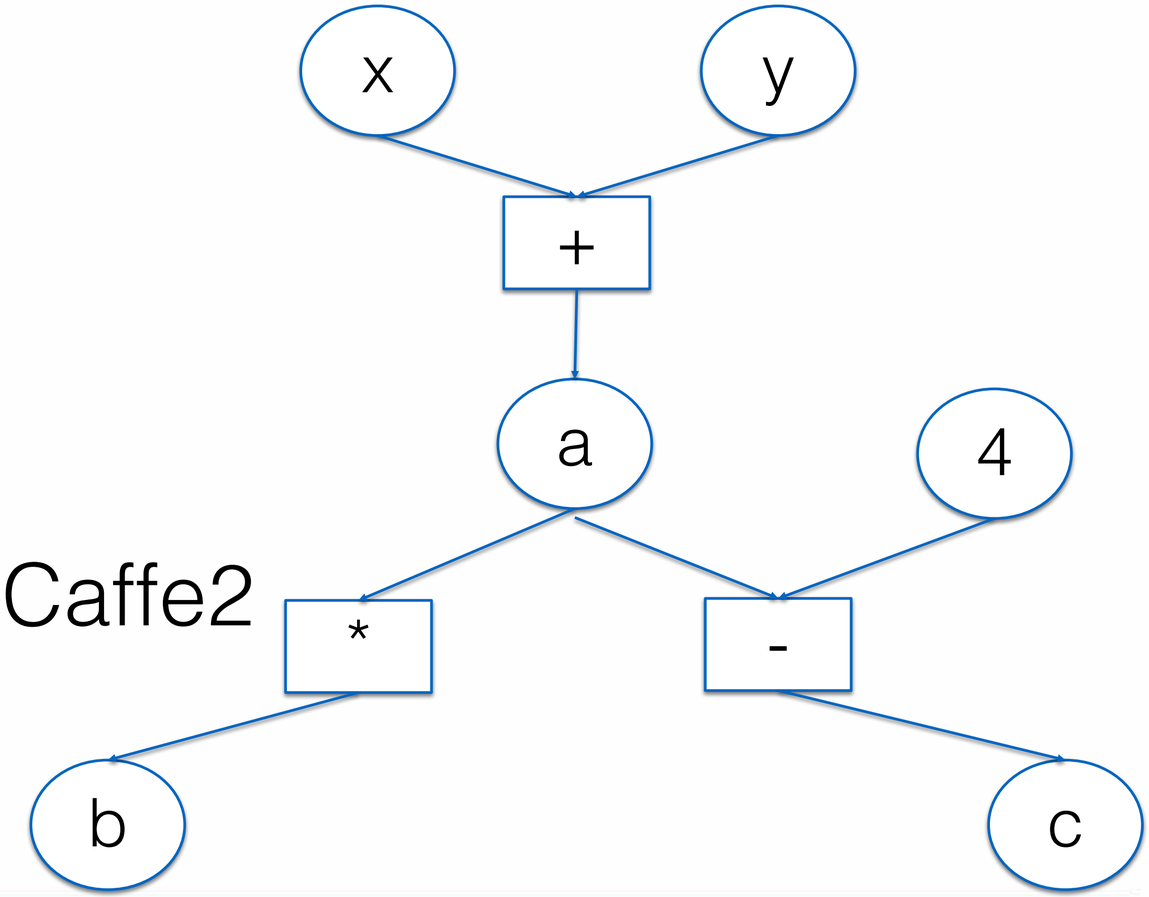
http://dlsys.cs.washington.edu/pdf/lecture4.pdf

图重写（优化）

- 编译器技术：Pattern match 或 Pass
- 无用代码消除（Dead code elimination）
- 公共子表达式（Common sub expression）
- 操作符融合（Operator fusion）
- 类型/形状推导（Shape inference）
- 内存优化（Memory planning）

执行引擎（Execution Engine）

- 拓扑序
- CPU调度，GPU执行
- 所有大数据引擎的内核
- 建议观摩：Tensorflow, MXNet, Caffe2



编程接口

Numpy

```
import numpy as np
np.random.seed(0)

N, D = 3, 4

x = np.random.randn(N, D)
y = np.random.randn(N, D)
z = np.random.randn(N, D)

a = x * y
b = a + z
c = np.sum(b)

grad_c = 1.0
grad_b = grad_c * np.ones((N, D))
grad_a = grad_b.copy()
grad_z = grad_b.copy()
grad_x = grad_a * y
grad_y = grad_a * x
```

TensorFlow

```
import numpy as np
np.random.seed(0)
import tensorflow as tf

N, D = 3, 4

with tf.device('/gpu:0'):
    x = tf.placeholder(tf.float32)
    y = tf.placeholder(tf.float32)
    z = tf.placeholder(tf.float32)

    a = x * y
    b = a + z
    c = tf.reduce_sum(b)

grad_x, grad_y, grad_z = tf.gradients(c, [x, y, z])

with tf.Session() as sess:
    values = {
        x: np.random.randn(N, D),
        y: np.random.randn(N, D),
        z: np.random.randn(N, D),
    }
    out = sess.run([c, grad_x, grad_y, grad_z],
                   feed_dict=values)
    c_val, grad_x_val, grad_y_val, grad_z_val = out
```

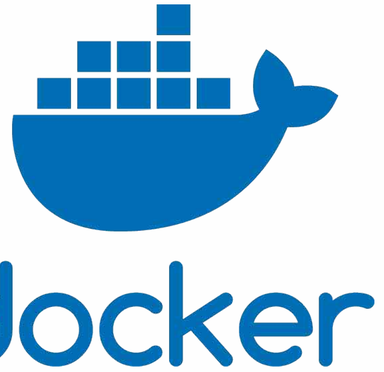
PyTorch

```
import torch
from torch.autograd import Variable

N, D = 3, 4

x = Variable(torch.randn(N, D).cuda(),
             requires_grad=True)
y = Variable(torch.randn(N, D).cuda(),
             requires_grad=True)
z = Variable(torch.randn(N, D).cuda(),
             requires_grad=True)

a = x * y
b = a + z
c = torch.sum(b)

c.backward()

print(x.grad.data)
print(y.grad.data)
print(z.grad.data)
```


http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture8.pdf

部署运维

- 容错：检查点机制
- 数据预处理：Hadoop & Spark
- Docker & Kubernetes
- 框架转换：ONNX



ONNX

kubernetes

TABLE OF CONTENTES

Define-and-run 与 Define-by-run

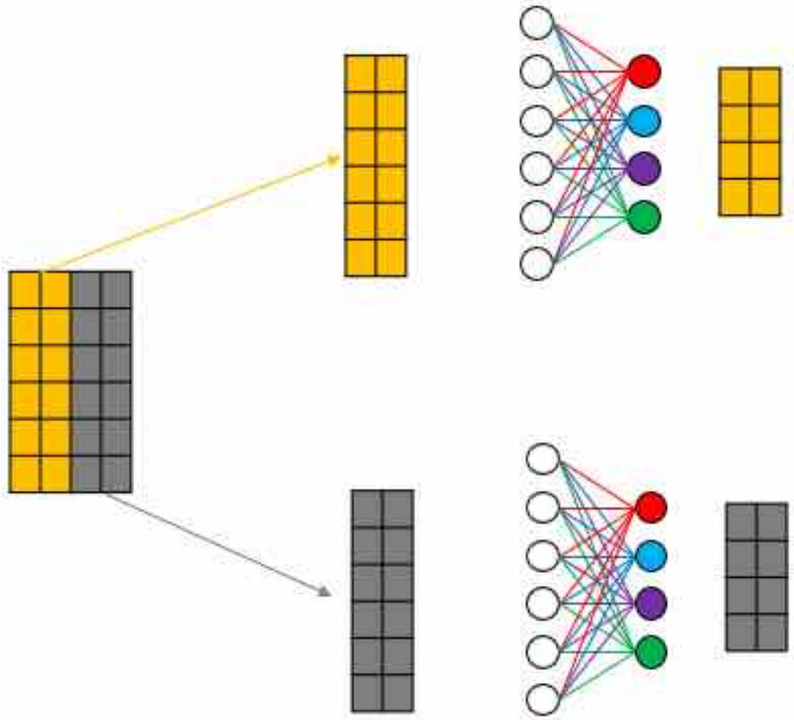
- Lazy evaluation 与 eager evaluation
- Declarative programming 与 imperative programming
- Dataflow 与 control flow
- 高效性与灵活性

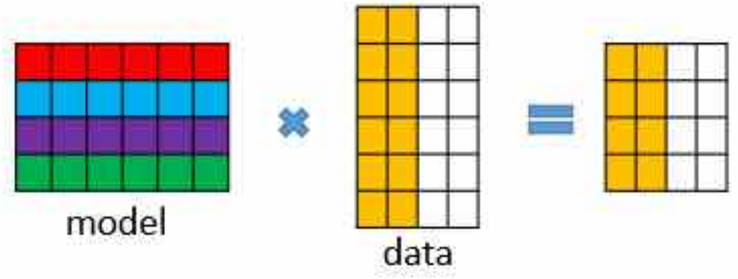
分布式训练

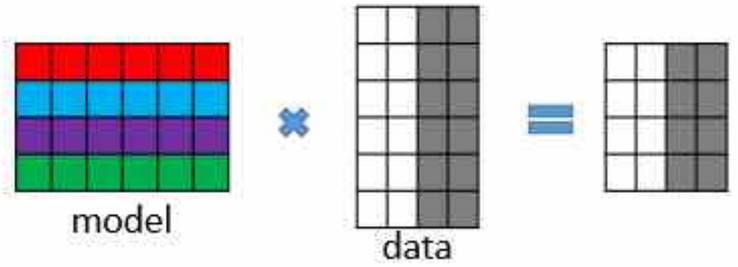
- 数据并行已经解决
- 模型并行支持的不好
- 流水线并行支持的不好
- 参数服务器，或者MPI, 或者Client-Master-Worker架构

数据并行

- 所有开源深度学习框架都支持

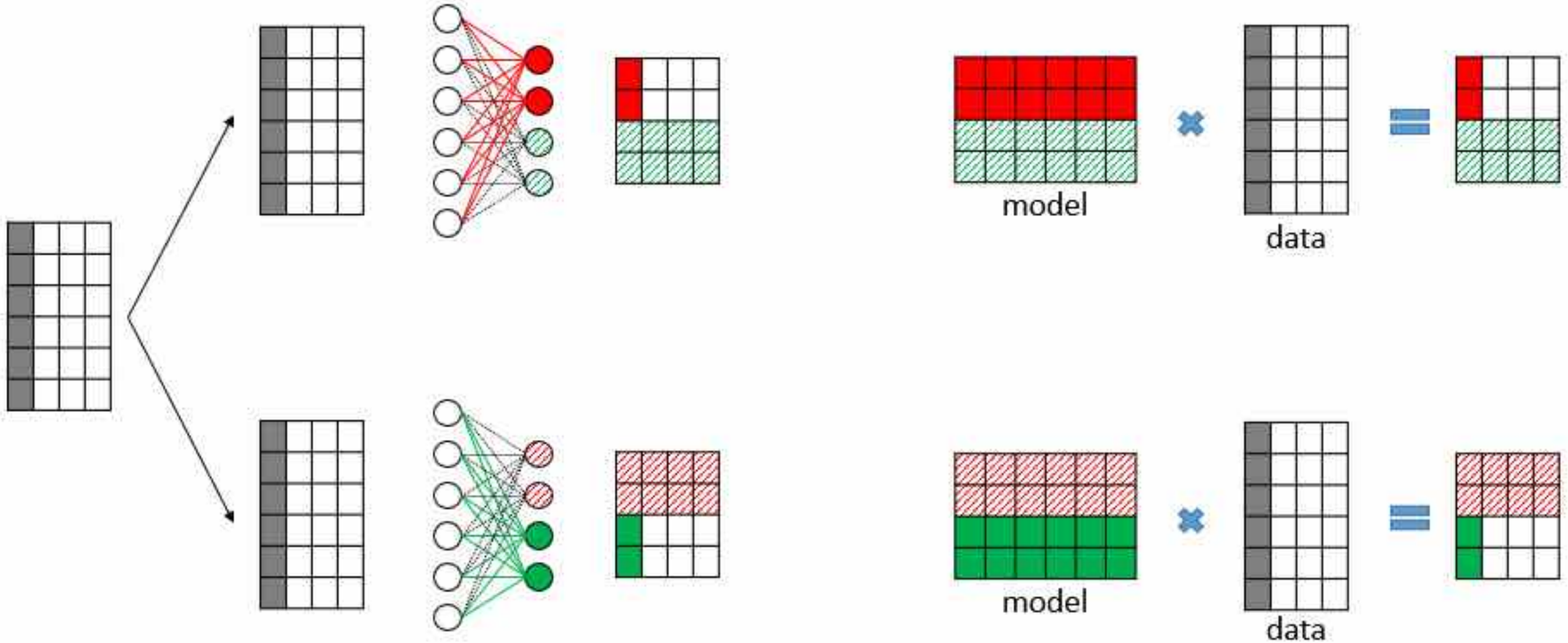






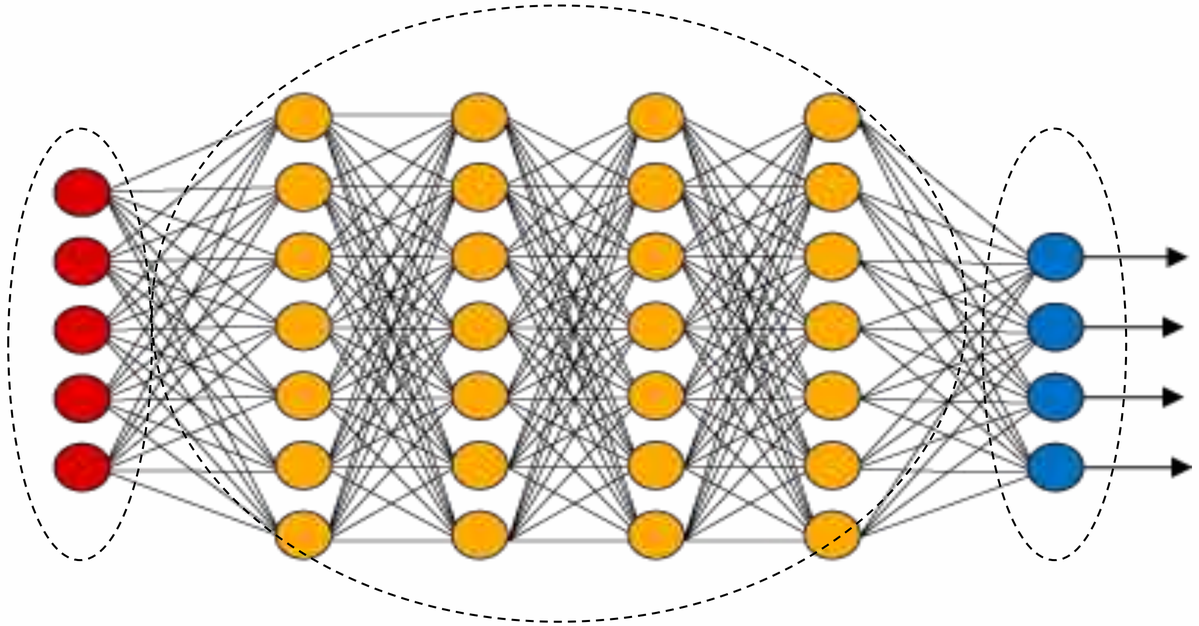
模型并行

- 极少开源框架支持



流水线并行

- 磁盘IO, 网络，PCIe，计算



GPU1 GPU2 GPU3

TABLE OF CONTENTES

我比较关注的框架

PYTORCH

Caffe2

系统完整度最高

需要解决性能问题（须壮士断腕）

训练大规模RNN/LSTM的唯一选择

NLP 应用首选
单机场景下的王者
难以支持大规模应用

大胆尝试各种新技术（杂）

CV场景有优势

TVM 是一个很有特色微观优化模块

CV 场景有优势

代码干净利落

网络库gloo独具特色

百度实战认证
上一版架构比较老
重构版优势还不明朗

开发一个深度学习框架不难
开发一个显著超越前人的系统很难

TABLE OF CONTENTES

展望

- 计算机视觉领域也需要模型并行（不仅数据大，模型也大）
- 模型并行得到解决
- 深度学习向更多场景渗透
- 深度学习框架技术逐渐收敛（同质化严重，需要新的空气）
- 深度学习框架变得像Hadoop一样不可或缺

Thanks!