

AI 框架作用



ZOMI

Building a better connected world

www.hiascend.com

www.mindspore.cn

关于本内容

1. 内容背景

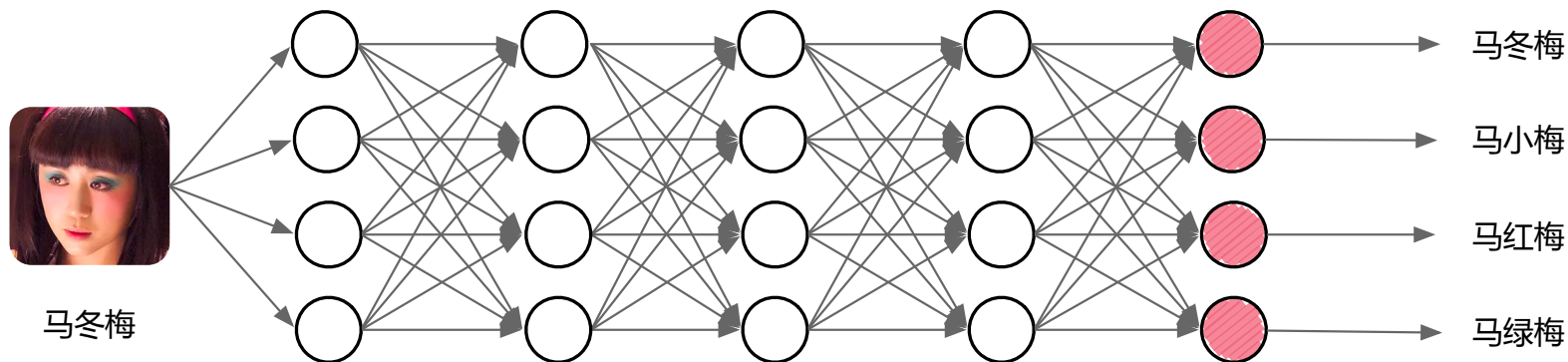
- AI框架的基础介绍

2. 具体内容

- AI框架作用：深度学习基础 – AI框架的作用 - AI框架的目的
- AI框架之争：第一代框架 – 第二代框架 – 第三代框架
- 编程范式：声明式编程 - 命令式编程

Review: Deep Learning Fundamentals

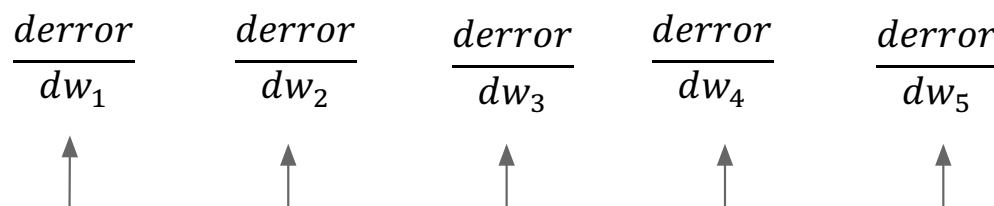
1. 定义一个神经网络:



2. 定义优化目标:



3. 计算梯度并更新权重参数:



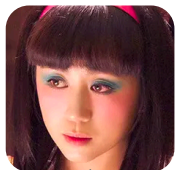
Implement a simple neural network

如何实现多线程算子加速？



```
1 for (int n = 0; n < o_n; ++n) {
2     for (int c = 0; c < o_c; ++c) {
3         for (int j = 0; j < o_h; ++j) {
4             for (int i = 0; i < o_w; ++i) {
5                 int d_start = n * i_c * i_h * i_w + j * i_w + i;
6                 int temp = 0;
7                 for (int kk = 0; kk < k_c; ++kk) {
8                     for (int kj = 0; kj < k_h; ++kj) {
9                         for (int ki = 0; ki < k_w; ++ki) {
10                            int k_idx = kk * k_h * k_w + kj * k_w + ki;
11                            int d_idx = d_start + kk * i_h * i_w + kj * i_w + ki;
12                            temp += inputs->data[d_idx] * kernel->data[k_idx];
13                        }
14                    }
15                }
16                res[n * o_c * o_h * o_w + j * o_w + i] = temp;
17            }
18        }
19    }
--
```

Implement a simple neural network



马冬梅

```
1 for (int i=0;i<row;i++)
2   for (int j = 0; j < k_h - mod_row; j++)
3     {
4       pad_map[i].push_back(map[i][col - 1]);
5     }
6 for (int j = 0; j < k_w - mod_col; j++)
7   {
8     pad_map.push_back(pad_map[row - 1]);
9   }
10
11 for(int i=0;i<out_row;i++)
12   for (int j = 0; j < out_col; j++)
13     {
14       int start_x = j*s_w;
15       int start_y = i*s_h;
16       vector<int> temp;
17       for(int ii=0;ii<k_w;ii++)
18         for (int jj = 0; jj < k_h; jj++)
19           {
20             temp.push_back(pad_map[start_y + jj][start_x + ii]);
21           }
22       sort(temp.begin(), temp.end());
23       res[i][j] = temp[temp.size() - 1];
24     }
```

如何跑在GPU上？

Implement a simple neural network



```
1 float* conv(float *image, float *weight, **argv){...}
2
3 float* relu(float *image, **argv){...}
4
5 float* maxpool(float *image, **argv){...}
6
7 float* fc(float *image, **argv){...}
8
9 float* softmax(float *image, **argv){...}
```

如何暴露给其它用户？

```
12 int main(){
13     std::memset(image, 220 * 220 * sizeof(double));
14     conv0 = conv(image, ...)
15     relu0 = relu(conv0)
16     maxpool0 = maxpool(relu0)
17     conv1 = conv(maxpool0)
18     relu1 = relu(conv1)
19     maxpool1 = maxpool(relu1)
20     fc0 = fc(maxpool1)
21     fc1 = fc(fc0)
22     softmax = softmax(fc1)
23     return softmax
24 }
```

如何实现后向算子？

如何优化这份代码？

Few questions AI framework wants to answer

- 前端（面向用户）：如何灵活的表达一个深度学习模型？
- 算子（执行计算）：如何保证每个算子的执行性能和泛化性？
- 求导（更新参数）：如何自动、高效地提供求导运算？
- 后端（系统相关）：如何将同一个算子跑在不同的加速设备上？
- 运行时：如何自动地优化和调度网络模型进行计算？

Example : An extreme calculation method

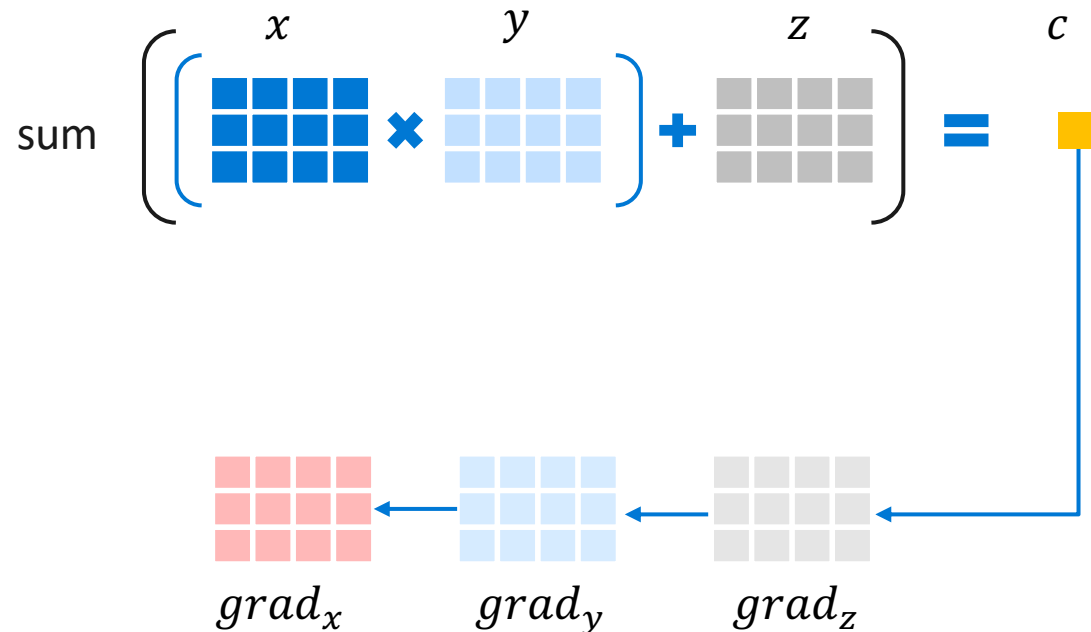
- 用高级语言从头实现一个模型的计算过程

```
import numpy as np

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
```



Example : An extreme calculation method

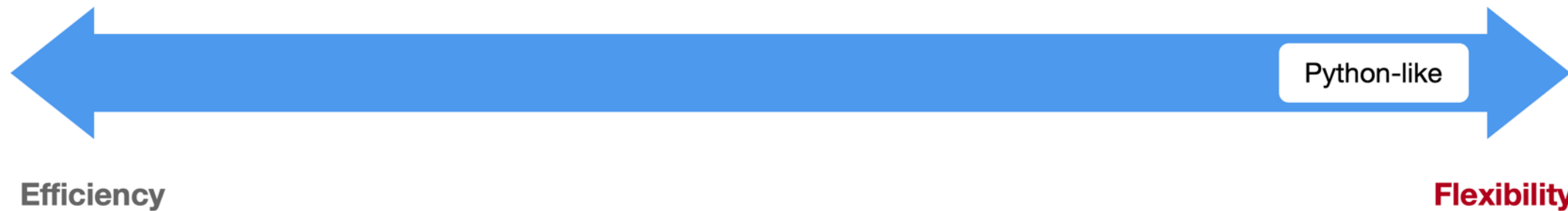
- 用高级语言从头实现一个模型的计算过程

```
import numpy as np

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)
```

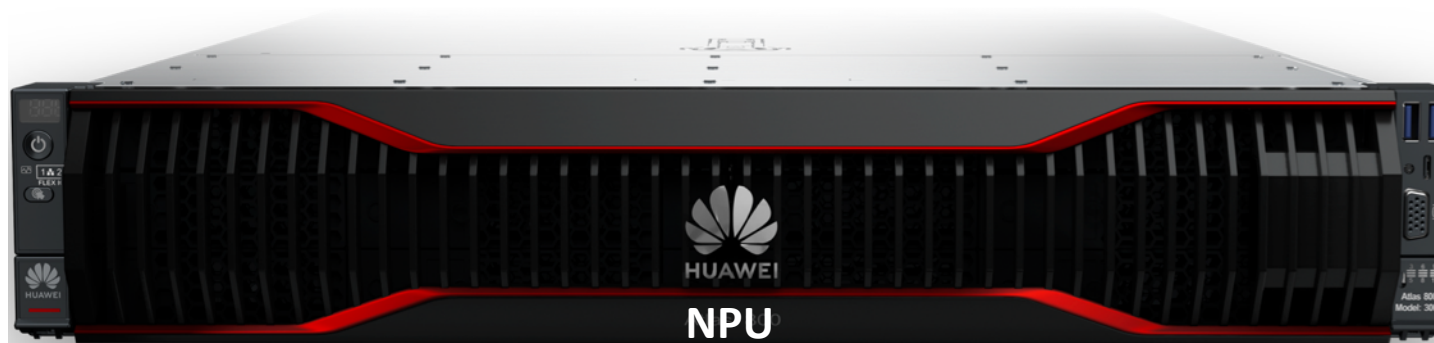
```
grad_c = 1.0
grad_b = grad_c * np.ones((N, D))
grad_a = grad_b.copy()
grad_z = grad_b.copy()
grad_x = grad_a * y
grad_y = grad_a * x
```



Example : Another extreme calculation method

- 为常用模型在NPU加速设备上实现一个高度优化的计算库

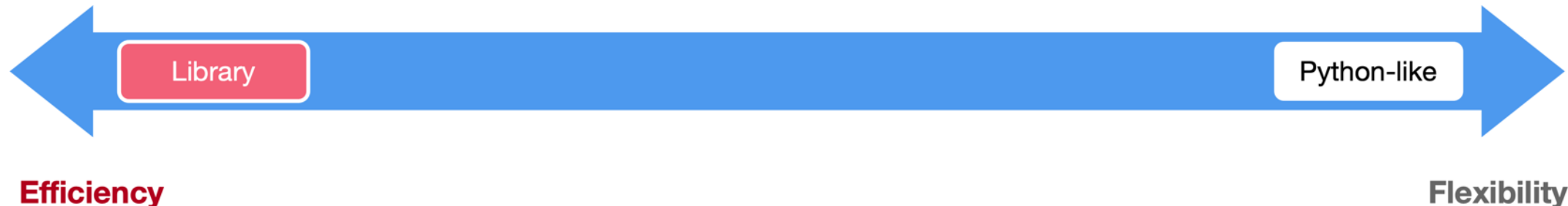
```
import xxlib  
  
x, y = load_data()  
  
y = xxlib.resnet152(x)
```



Example : Another extreme calculation method

- 为常用模型在NPU加速设备上实现一个高度优化的计算库

```
import xxlib  
  
x, y = load_data()  
  
y = xxlib.resnet152(x)
```



The role of AI frameworks



AI Framework Propose

提供灵活的编程模型和编程接口

- 自动推导计算图
- 较好的支持与现有生态环境融合
- 提供直观的模式构建方式，简洁的神经网络计算编程语言

提供高效和可扩展的计算能力

- 自动编译优化算法（子表达式消除、内核融合、内存优化等）
- 根据不同体系结构和硬件设备自动并行化（自动分布式化、扩展多计算节点等）

Summary

1. 回顾了深度学习的基本流程
2. 了解了AI框架上承应用算法、下接芯片使能的作用
3. 知道了AI框架提供灵活的编程接口和可扩展的计算能力2个目的



BUILDING A BETTER CONNECTED WORLD

THANK YOU

Copyright©2014 Huawei Technologies Co., Ltd. All Rights Reserved.

The information in this document may contain predictive statements including, without limitation, statements regarding the future financial and operating results, future product portfolio, new technology, etc. There are a number of factors that could cause actual results and developments to differ materially from those expressed or implied in the predictive statements. Therefore, such information is provided for reference purpose only and constitutes neither an offer nor an acceptance. Huawei may change the information at any time without notice.