



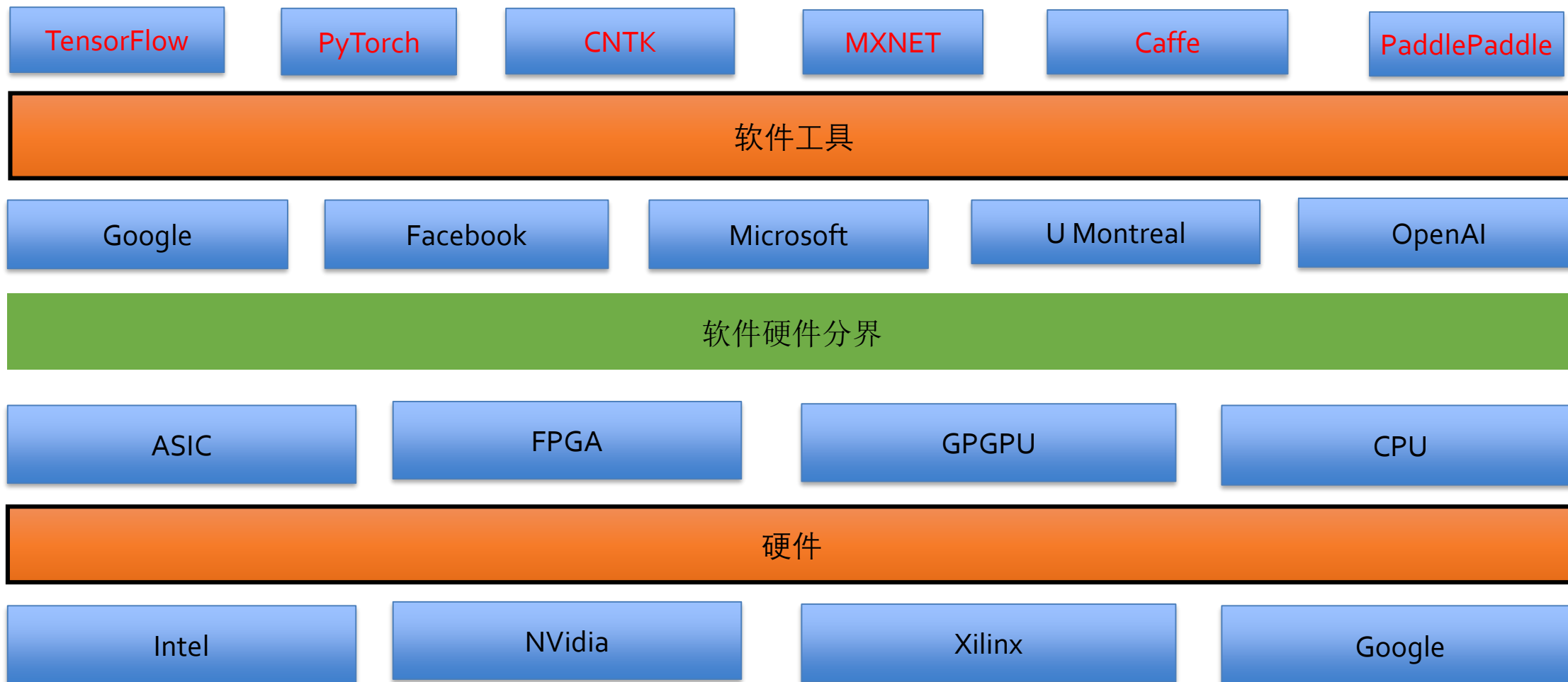
TensorFlow 机器学习库

TensorFlow Machine Intelligence Library

清华大学基础工业训练中心

深度学习软硬件布局

- 相关的硬件与软件及其产业公司



深度学习框架

- **Tensor Flow:** *Google Deep Learning Library*
 - Supports general deep learning with symbolic diff.
 - Python on top of C++ (Easy + Fast)
 - GPU, cluster, and mobile implementations
- **pyTorch:** *Facebook AI research*
 - Tensor Library
 - File I/O Interface Library
- **Berkeley Caffe:** *GPU accelerated Computer Vision*
 - Focused on computer vision and GPU acceleration
 - C++ with Python support (Very Fast + somewhat easy)
 - Rich library of pre-trained models (Caffe Model Zoo)
- **Theano:** *U of Montreal*
 - General Symbolic Diff. Modeling Framework
 - Covers many recent research models
 - Python only (easy but not fast)
- **Microsoft/CNTK**
- **DMLC/MXNET**
- **Baidu/PaddlePaddle**

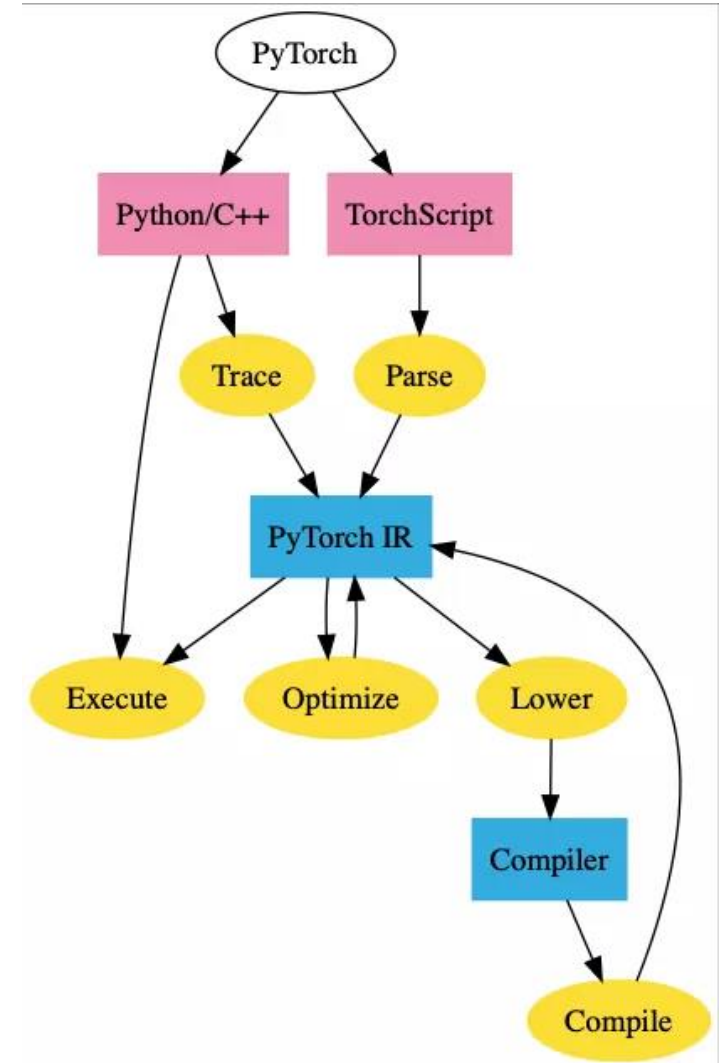


深度学习框架是什么？

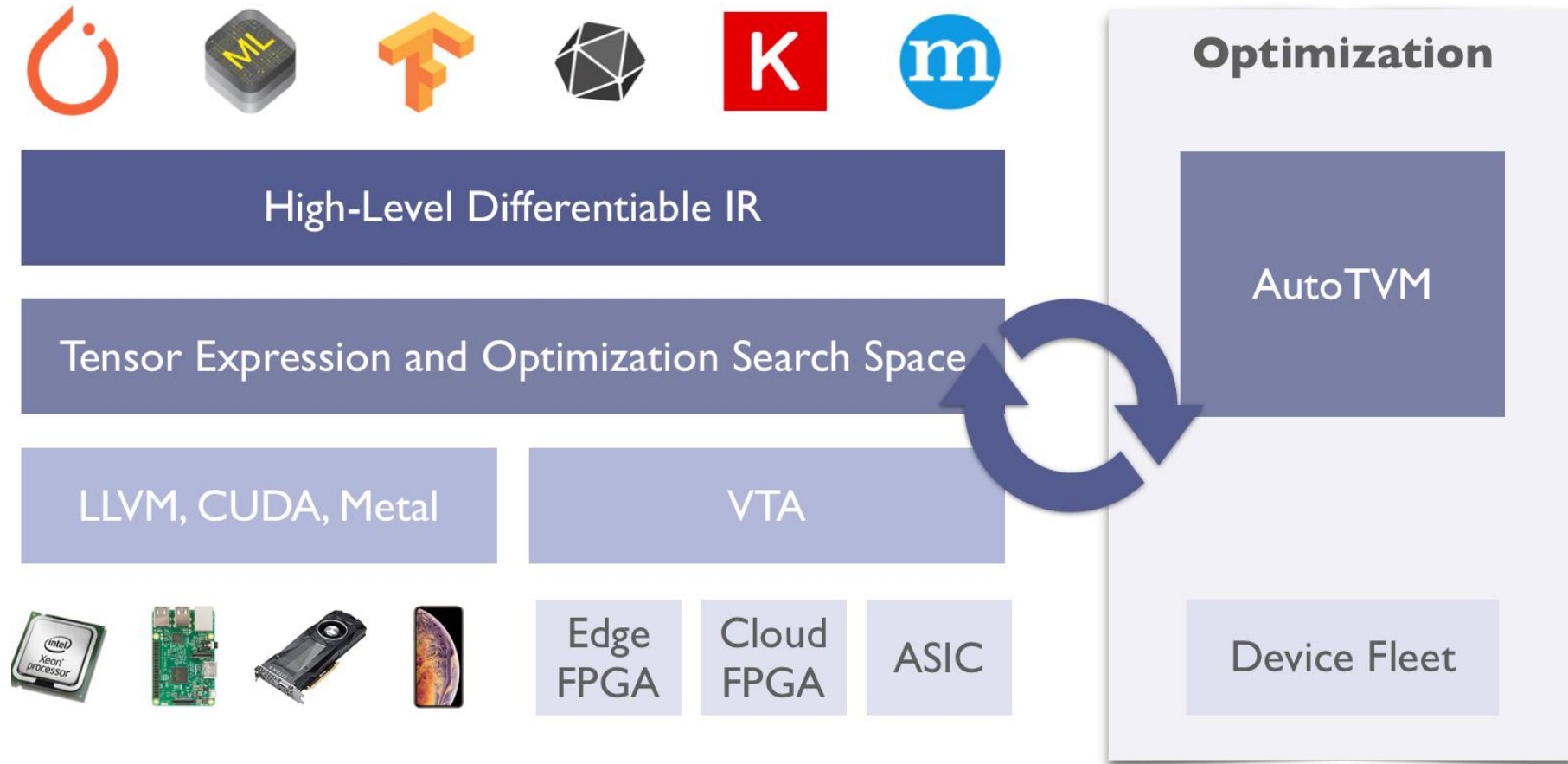
- 深度学习框架（Deep Learning Framework）是描述多层网络模型及训练推断的编程语言及工具类库。
 - 过程式语言Python / C 不同
 - 申明式编程语言Prolog 类似
- 深度学习框架包括：
 - 编程语言，解释器，编译器。
- 深度学习框架的不同，对应着编程语言内部的不同设计。
 - 动态语言和静态语言的差别，对应着 TensorFlow和PyTorch 的动态计算图和静态计算图的区别。

deep learning compiler stack

- PyTorch
- PyTorch IR
- Execution Code



deep learning compiler stack

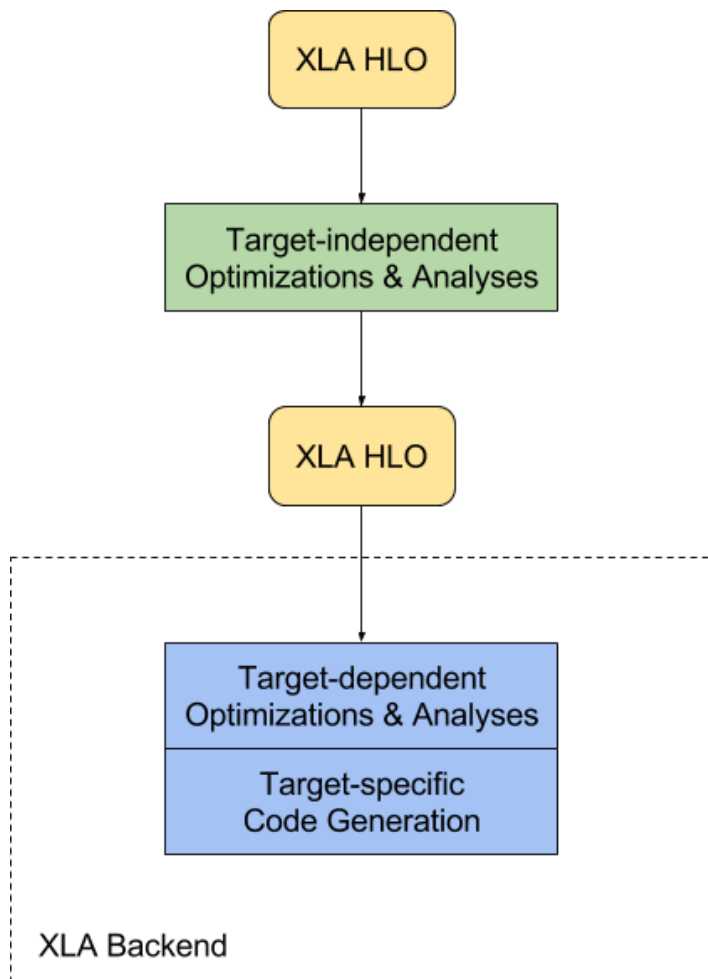


- <https://tvm.ai>

TensorFlow深度学习框架

- TensorFlow中的“计算图”，类似对应为编译器中的 data-flow graph 或者 control-flow graph
- TensorFlow会自动对代码求导，优化未知参数，使得误差最小。可以称为一种“可求导编程语言”（differentiable programming language）。
- TensorFlow的编译器XLA (Accelerated Linear Algebra) 优化TensorFlow计算图

TensorFlow XLA



- XLA(Accelerated Linear Algebra 加速线性代数)
- XLA是一种能够优化 TensorFlow 计算的编译器

TensorFlow 简介

- 谷歌大脑团队出品,
- 在谷歌内部应用广泛, 2015年11月开源
- <http://www.tensorflow.org/>
- <http://tensorflow.google.cn>



谷歌大脑, Google Brain , <https://research.googleblog.com/>

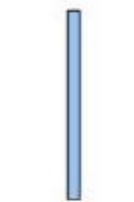
TensorFlow简介

- 开发者：Google Brain Team（Google Research）
- 历史：
 - DistBelief: 第一代深度学习系统：DistBelief: First Generation Deep Learning System
 - TensorFlow: 第二代深度学习系统：TensorFlow: Second Generation Deep Learning System
- 网址：<http://www.tensorflow.org/>
- 源代码：<https://github.com/tensorflow/tensorflow>
- 论文与白皮书：<http://tensorflow.org/whitepaper2015.pdf>
- OSDI论文
 - TensorFlow: A system for large-scale machine learning



Tensor简介

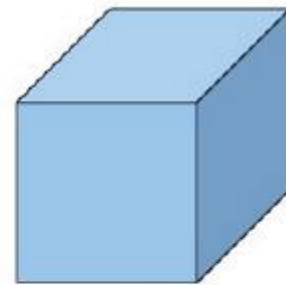
- Tensor（张量）意味着N维数组。
- 1维Tensor的形式是向量；
- 2维Tensor的形式是矩阵；
- 3维Tensor的形式是彩色图像，
可以用（行，列，颜色）来表示。



1d-tensor



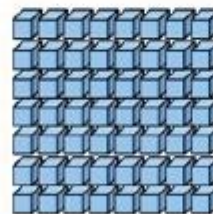
2d-tensor



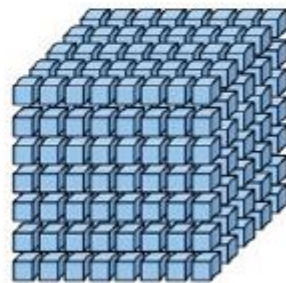
3d-tensor



4d-tensor



5d-tensor



6d-tensor

TensorFlow名称来历

- TensorFlow是一个用数据流图进行数值计算的软件库。图中的节点表示的数学运算，而图的边代表它们之间传送的多维数据阵列（张量）。
- 张量Tensor从图的一端流动到另一端，这就是“TensorFlow”（张量流）名称来源。
- Tensor是TensorFlow的核心
- 在TensorFlow框架中，
 - Tensor的形式有三种：constant, placeholder, variables
 - Tensor的属性有：rank, shape, datatype

TensorFlow 1.0版本

- Release 1.13
 - Support for Python3.7 on all operating systems.
 - TensorFlow Lite has moved from contrib to core.
 - TensorFlow GPU binaries are now built against CUDA 10 and TensorRT 5.0.
 - Moved NCCL to core.
- Release 1.14.0 (2019.8.19)
 - Turn on MKL-DNN contraction kernels by default. MKL-DNN dynamically dispatches the best kernel implementation based on CPU vector architecture.
- Release 1.15.0 (2019.10.23)

TensorFlow 2.0版本

- TensorFlow 2.0 focuses on simplicity and ease of use, featuring updates like:
 - Easy model building with Keras and **eager execution**
 - Robust model deployment in production on any platform
 - Powerful experimentation for research
 - API simplification by reducing duplication and removing deprecated endpoints
- <http://tf.wiki>

Eager execution in TensorFlow 2.0

- A call to a function `f(g(x+y), 2*x)`.
- In a language with **eager evaluation**, you'd first compute the values of the parameters and only invoke the function after the parameters had been computed.
 - You'd compute `g(x+y)` and `2*x` before invoking `f`; and when you were computing `g(x+y)`, you'd first compute `x+y` before invoking the function `g`.
 - This is the way that many familiar languages actually work: for example, this is how **C, Java, JavaScript, and Python** work.
- In **lazy evaluation**, you don't compute the value of any expression until you need to.
 - You'd invoke `f` first. You wouldn't invoke `g(x+y)` until `f` tried to use the value of that expression. If `f` never specifically used the value of the parameter expression `g(x+y)`, then it would never get computed and `g` would never get invoked.
 - This kind of evaluation turns out to be really useful, and it's the basis of languages like **Haskell and Miranda**.

TensorFlow安装与使用 (1.0版本)

- 安装
 - '\$pip install tensorflow'
 - '\$pip install tensorflow-gpu'
- 使用, 运行python
 - >>>import tensorflow as tf
 - >>>a=tf.constant(1.0)
 - >>>b=tf.constant(3.0)
 - >>> c=a+b
 - >>> sess = tf.Session()
 - >>> sess.run(c)

TensorFlow 简单示例 (1.0版本)

- `# tensorflow`
- `>>> a = tf.placeholder(tf.int8)`
- `>>> b = tf.placeholder(tf.int8)`
- `>>> sess = tf.Session()`
- `>>> sess.run(a+b, feed_dict={a: 10, b: 32})`

TensorFlow 2.0版本

- 安装
 - '\$pip install tensorflow'
 - '\$pip install tensorflow-gpu'
- 使用, 运行python
 - >>>import tensorflow as tf
 - >>>a=tf.constant(1.0)
 - >>>b=tf.constant(3.0)
 - >>> c=a+b
 - >>> print(c.numpy())

TensorFlow 2.0版本Keras示例

```
import tensorflow as tf
```

```
mnist = tf.keras.datasets.mnist
```

```
(x_train, y_train), (x_test, y_test) = mnist.load_data()  
x_train, x_test = x_train / 255.0, x_test / 255.0
```

```
model = tf.keras.models.Sequential()
```

```
model.add(tf.keras.layers.Flatten(input_shape=(28, 28)))  
model.add(tf.keras.layers.Dense(128, activation='relu'))  
model.add(tf.keras.layers.Dropout(0.2))  
model.add(tf.keras.layers.Dense(10, activation='softmax'))
```

```
model.compile(optimizer='adam', loss='sparse_categorical_crossentropy', metrics=['accuracy'])
```

```
model.fit(x_train, y_train, epochs=5)
```

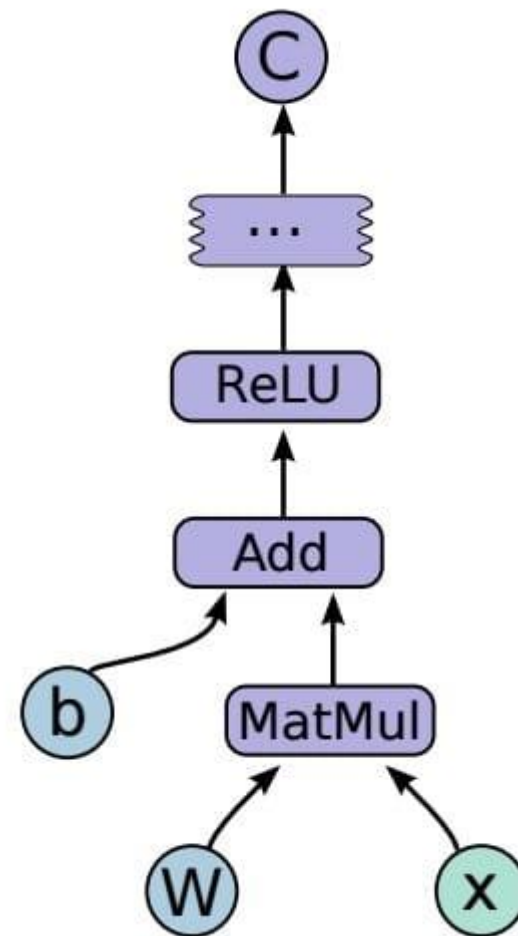
```
model.evaluate(x_test, y_test, verbose=2)
```

TensorFlow计算图CG (1.0版本)

- TensorFlow是一种元编程 (meta programming) , 构建计算图的语言
- 基本人工神经元的代码分析:

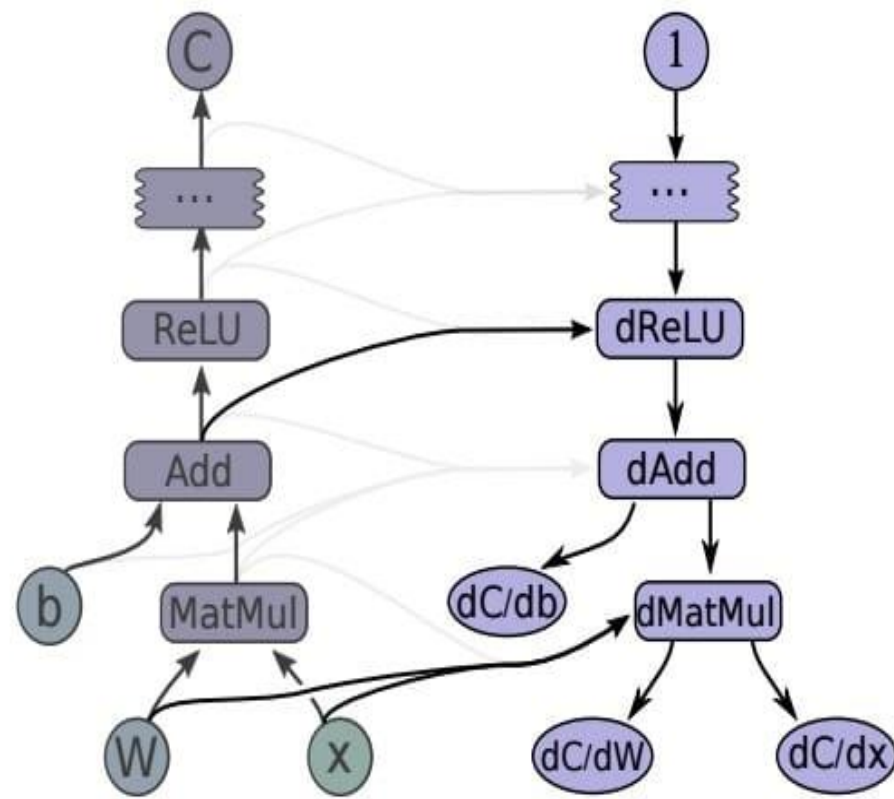
```
import tensorflow as tf
b = tf.Variable(tf.zeros([100])) # 100-d vector, init to zeroes
W = tf.Variable(tf.random_uniform([784,100],-1,1)) # 784x100 matrix w/rnd vals
x = tf.placeholder(name="x") # Placeholder for input
relu = tf.nn.relu(tf.matmul(W, x) + b) # Relu(Wx+b)
C = [...] # Cost computed as a function of Relu
```

```
sess = tf.Session()
for step in xrange(0, 10):
    input = ...construct 100-D input array ... # Create 100-d vector for input
    result = sess.run(C, feed_dict={x: input}) # Fetch cost, feeding x=input
    print step, result
```



TensorFlow求导

- TensorFlow求导采用符号微分方法
- 在图g上，由后向前：
 - 从结果C开始，查找C的所有依赖节点I，并计算C的梯度，插入新的计算图中
 - 递归地从I开始，查找I的所有依赖节点I'，并计算I的梯度，插入新的计算图中
- 最终，自顶向下地形成梯度的计算图g'。

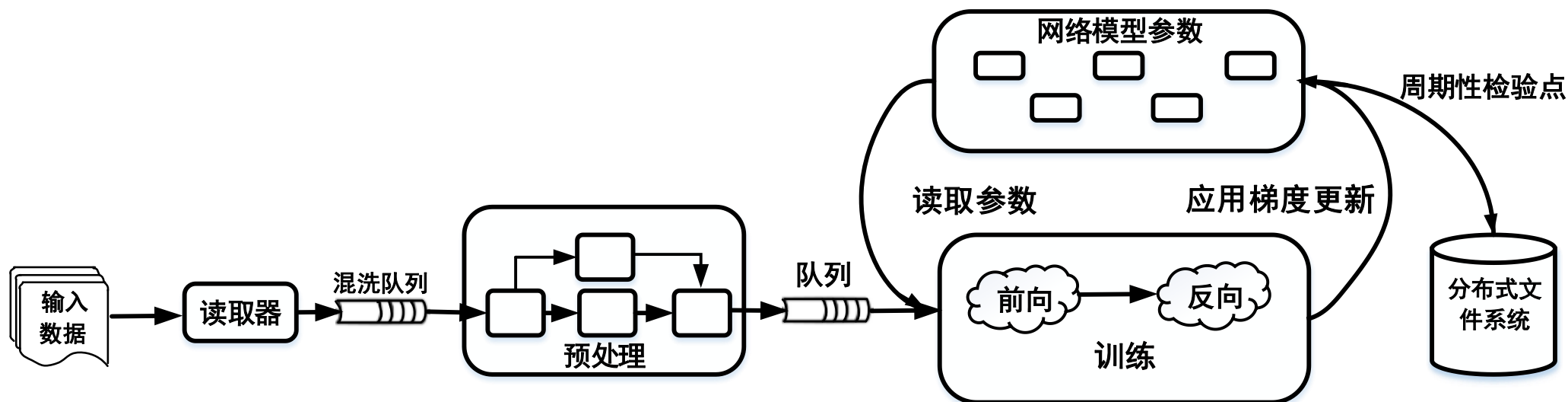


TensorFlow底层

- Eigen is a C++ template library for linear algebra: matrices, vectors, numerical solvers, and related algorithms.
- <http://eigen.tuxfamily.org/>
- Eigen::Tensor实现的Tensor定义和运算主要是调用
- gemmlowp (Low-precision matrix multiplication)低精度矩阵库加快量化计算
- <https://github.com/google/gemmlowp>

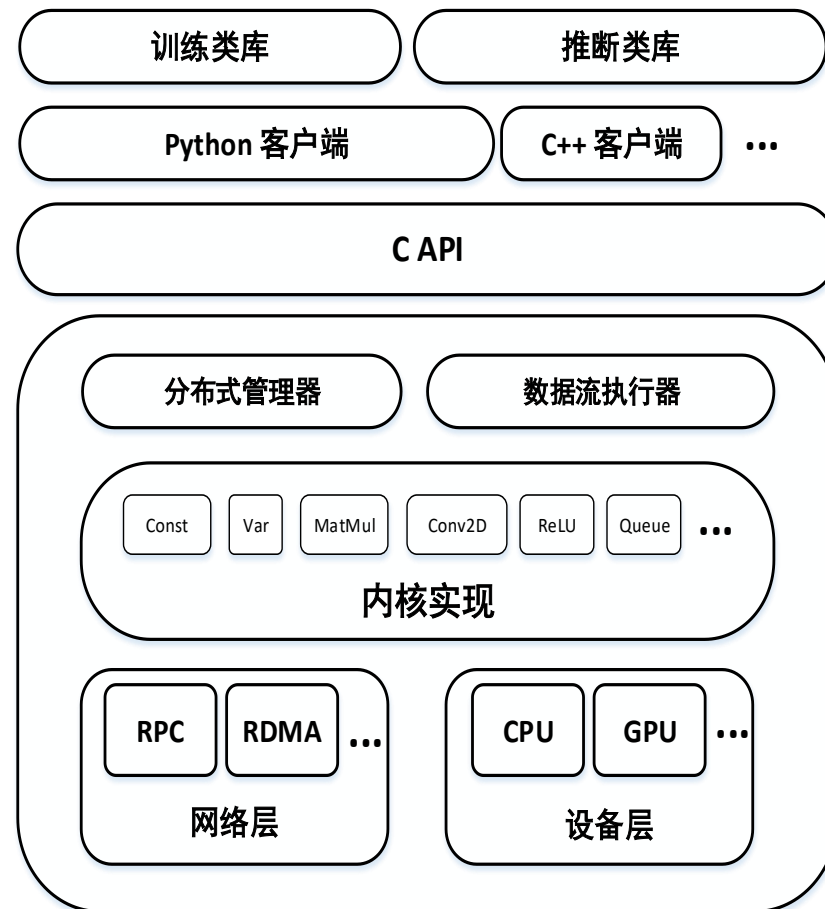
TensorFlow数据流图

- TensorFlow用于模型训练过程的数据流图，包括训练数据的读取和转换，队列，参数的更新以及周期性监测点生成。
- 图中的操作都是并发执行的，图中的节点的可变状态（Mutable states）在图的执行中是可以共享的。

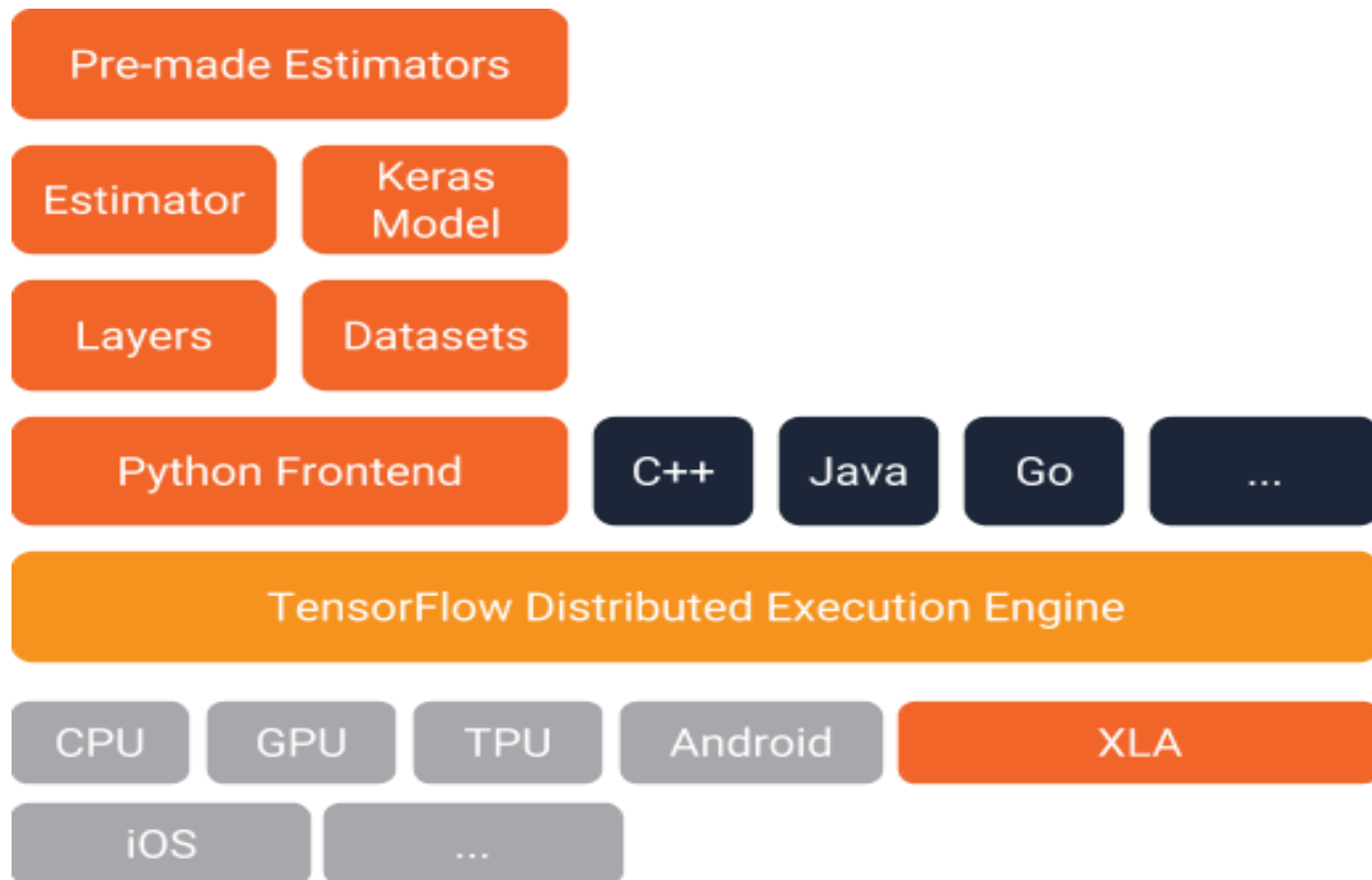


TensorFlow架构

- 上层是训练库（Training library）和推断库（Inference libs），部署最终的生成模型在不同的设备上。
- 中间层是Python和C++接口，方便程序员进行调用。
- 底层是网络层和设备层，TensorFlow可以灵活的运行在通过网络连接的不同计算设备上。
- 统一API（Python、C++等）调用，部署在一个或多个CPU或GPU的桌面电脑、服务器或移动设备。

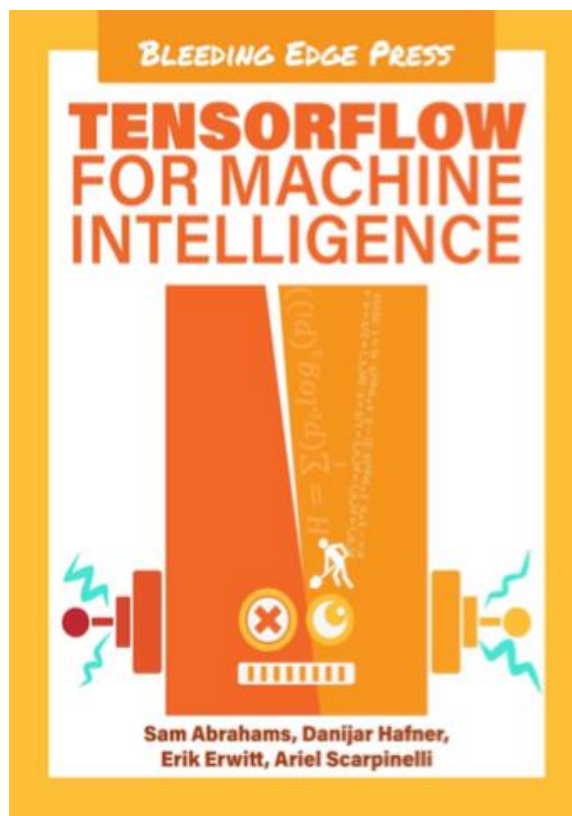


TensorFlow 编程模型



参考书

- Abrahams, Sam, Danijar Hafner, Erik Erwit, and Ariel Scarpinelli. TensorFlow for Machine Intelligence: A Hands-on Introduction to Learning Algorithms. Bleeding Edge Press, 2016.



谢谢指正！

zhenchen@tsinghua.edu.cn

万物皆数

- “万物皆数”是毕达哥拉斯学派 (*the Pythagoreans*) 的观点，毕达哥拉斯学派融合了数学和神秘主义的观点
- “... in all nature numbers are the first, they supposed the elements of numbers to be the elements of all things.”
- 毕达哥拉斯定理和无理数的发现
- <https://en.wikipedia.org/wiki/Pythagoreanism>