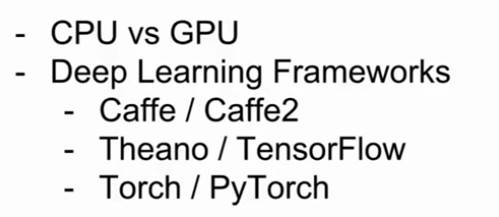
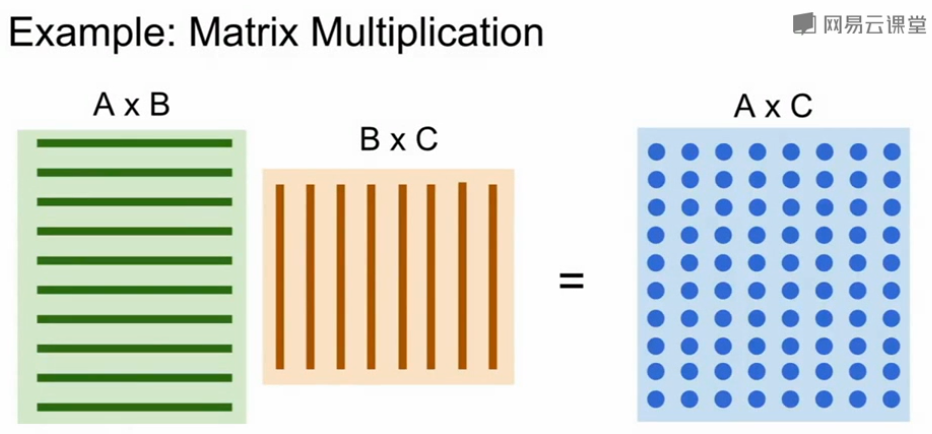
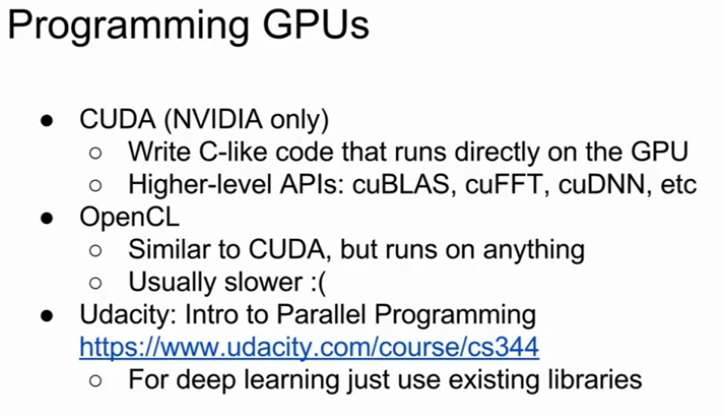
# Lecture 8:

# Hardware and Software



GPU有很多个核 ， 更适合做矩阵运算

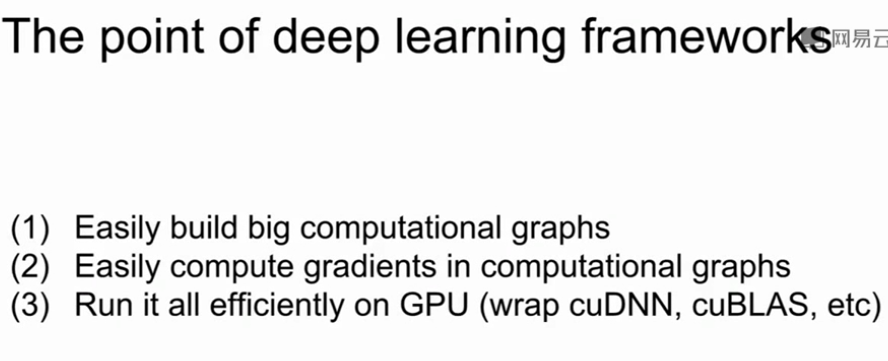




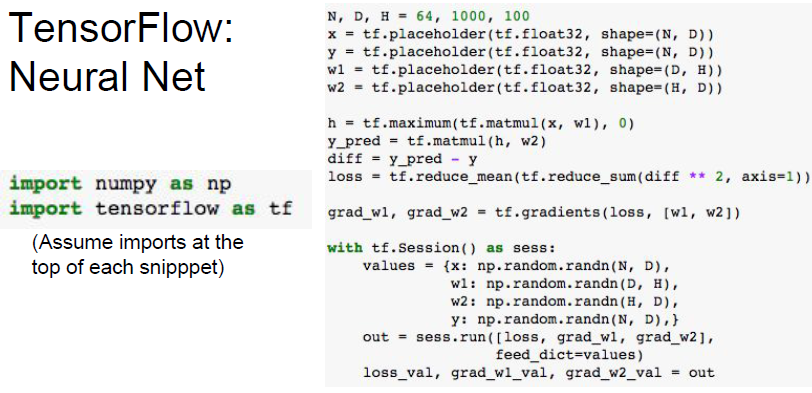
并行编程：

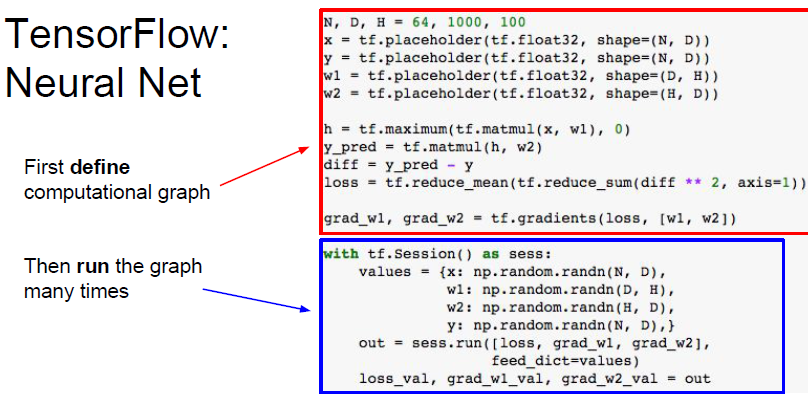
<https://cn.udacity.com/course/intro-to-parallel-programming--cs344>

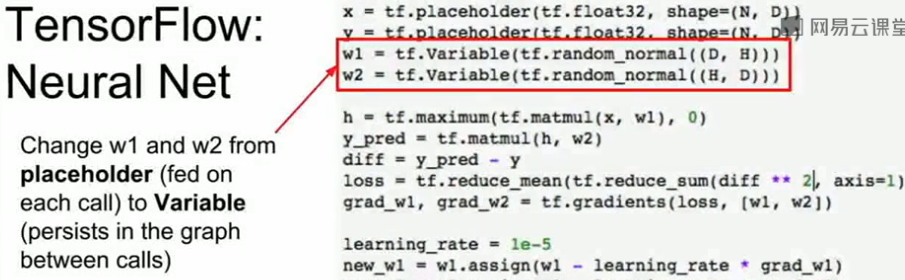
针对卷积神经网络优化过的cuDNN比没有优化过的CUDA库 要快3倍左右

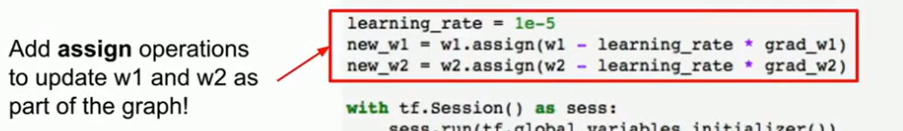


# TensorFlow(Google)



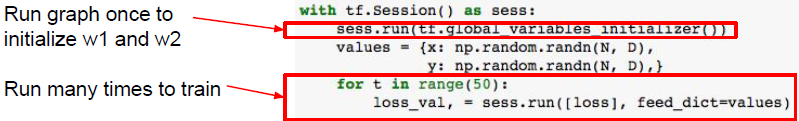


避免CPU GPU之间的数据拷贝，将w1,w2改为变量而不是占位符， 

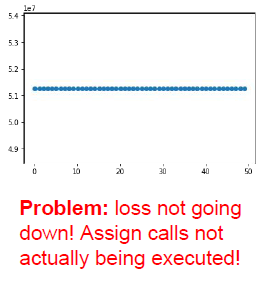


先告诉 tensor flow初始化，

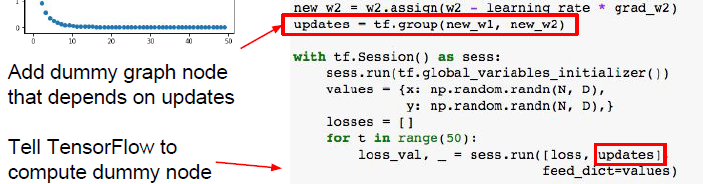
再多次执行计算图运算



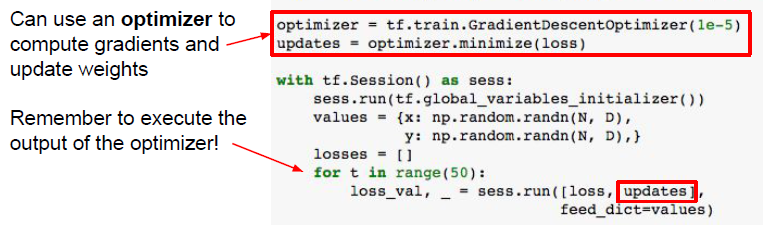
但这里有个BUG: 没有告诉tensor flow要用new\_w1, new\_w2代替 w1,w2,而这里计算Loss，不需要更新w1,w2



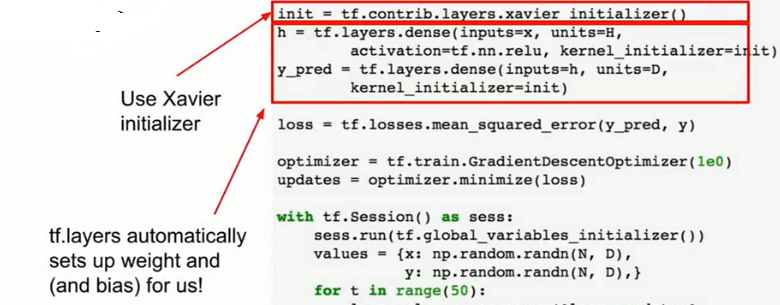
这里设了一个dummy值updatee = tf.group(new\_w1,new\_w2),并不存值，但是更新w1,w2



Tensor flow提供的简便方法：优化器 optimizer

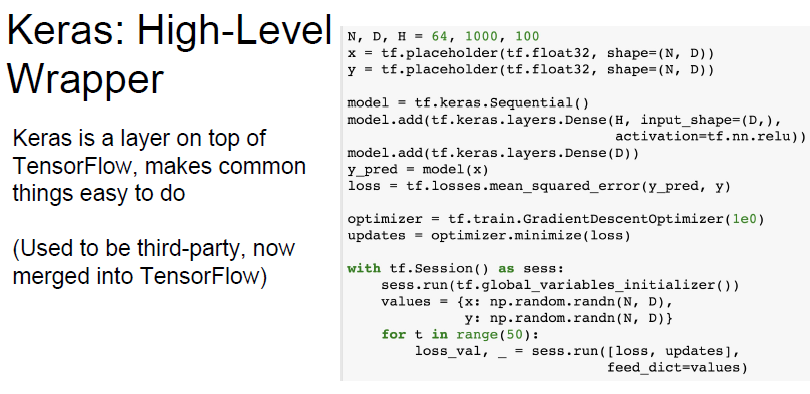


## Tf.layers:



在tensor flow上有不同的高级库，

Keras:



定义 模型，不同的层

-》 定义优化方法

–》 建立模型，指定loss函数(softmax ,svm…)

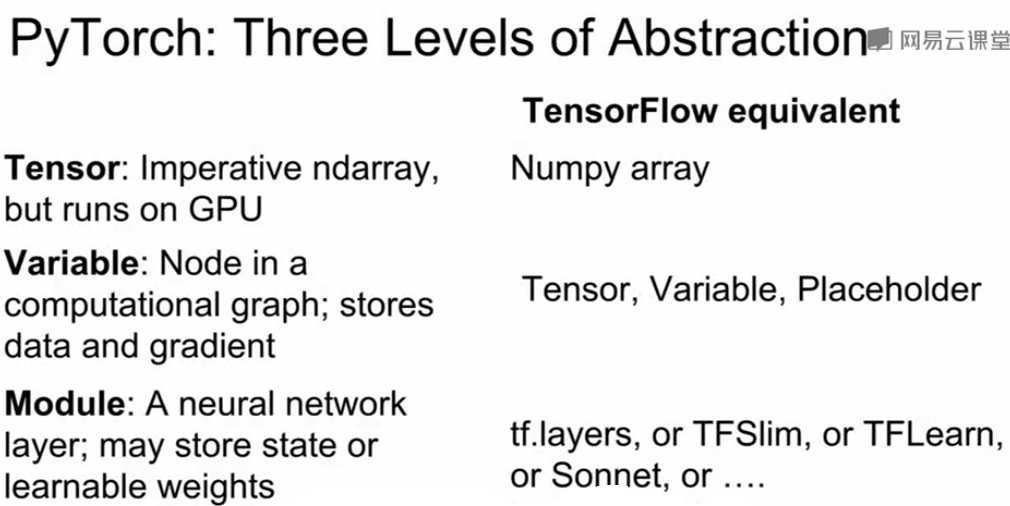
-》 训练模型 (fit )

# Tensor flow 扩展 大礼包：

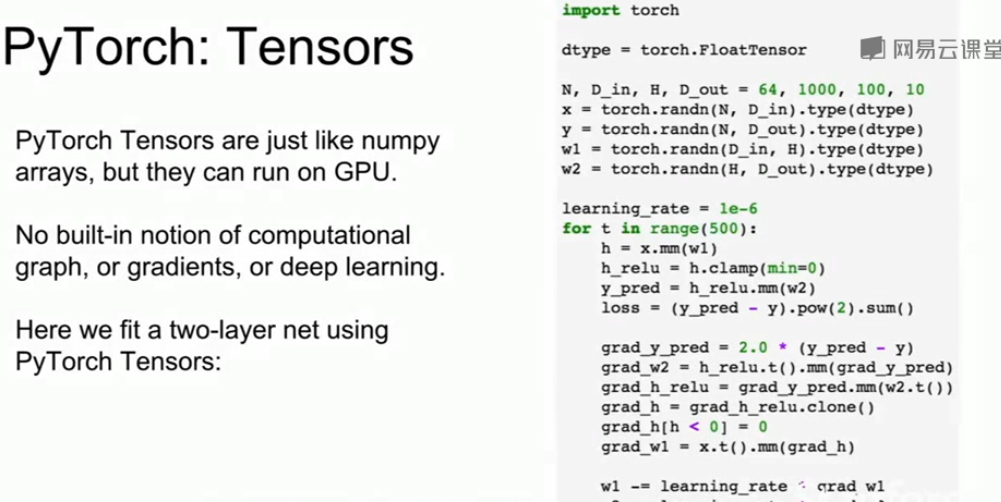


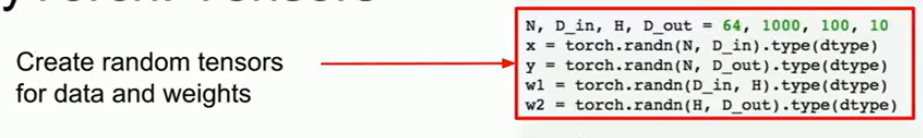
# PyTorch(Fackbook)

## 三层抽象：

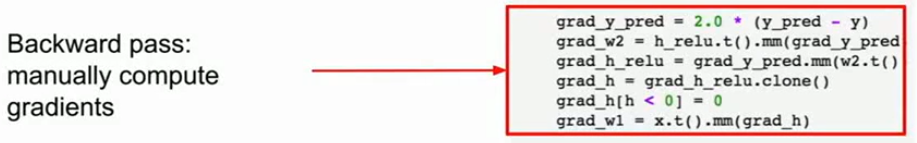


代码实例：



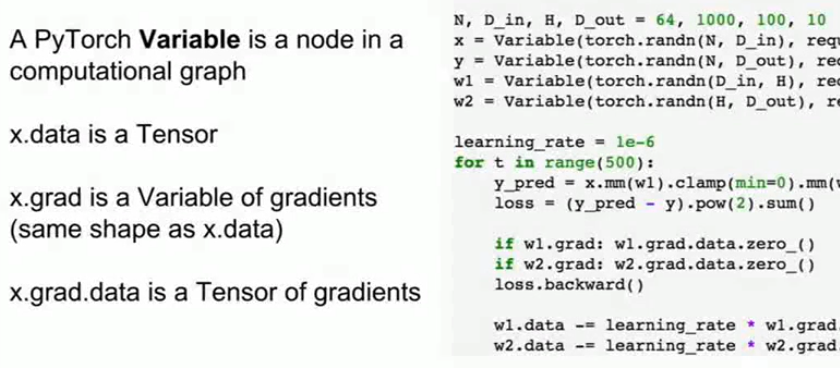






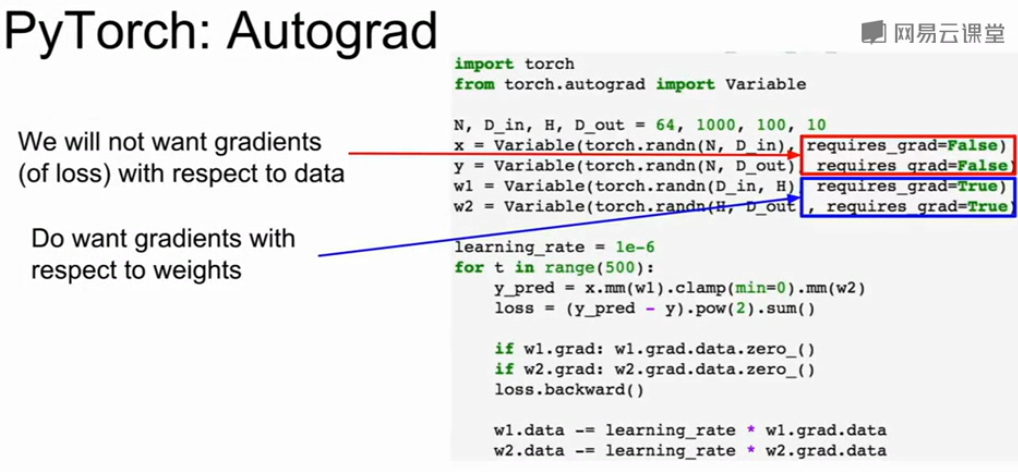


### 抽象层： Variable

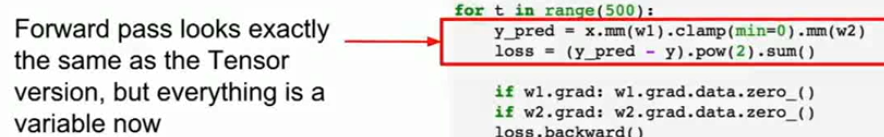


PyTorch 的 Tensor 和 Variable有相同的API, 他们会记得他们前向传播时的信息(类似于我们卷积算法中的cache)，用来做反向传播时用。

这里告诉PyTorch 那些variable需要计算梯度，要保留cache



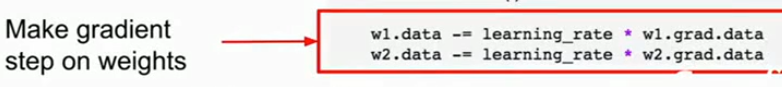
向前传播：



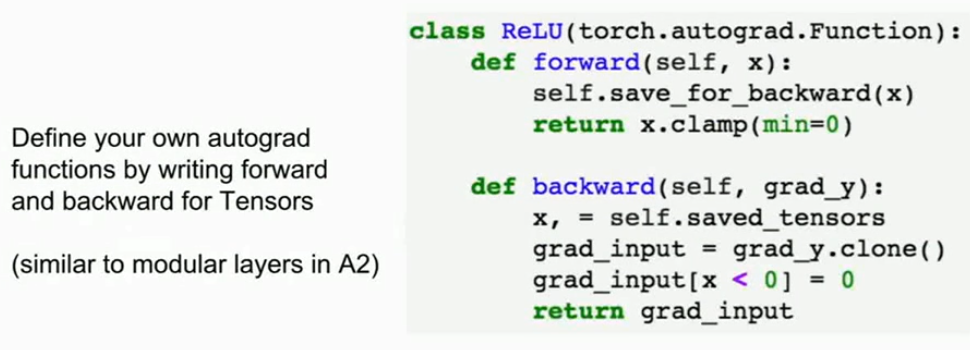
计算梯度：



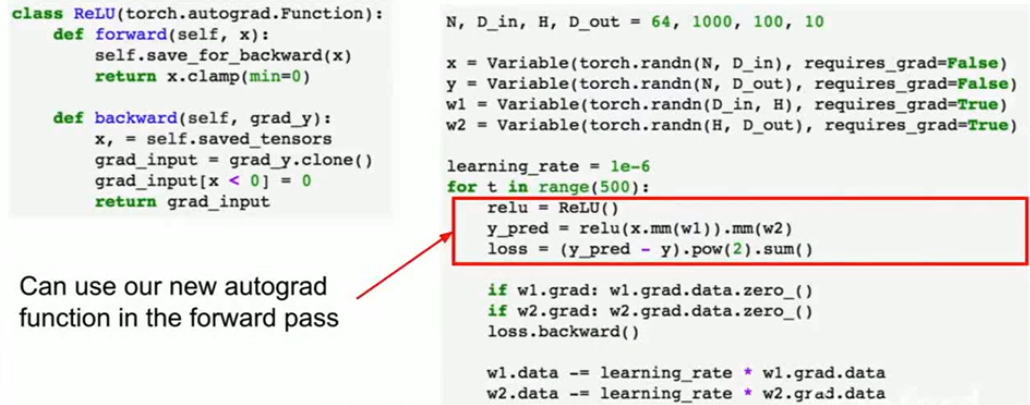
更新梯度：



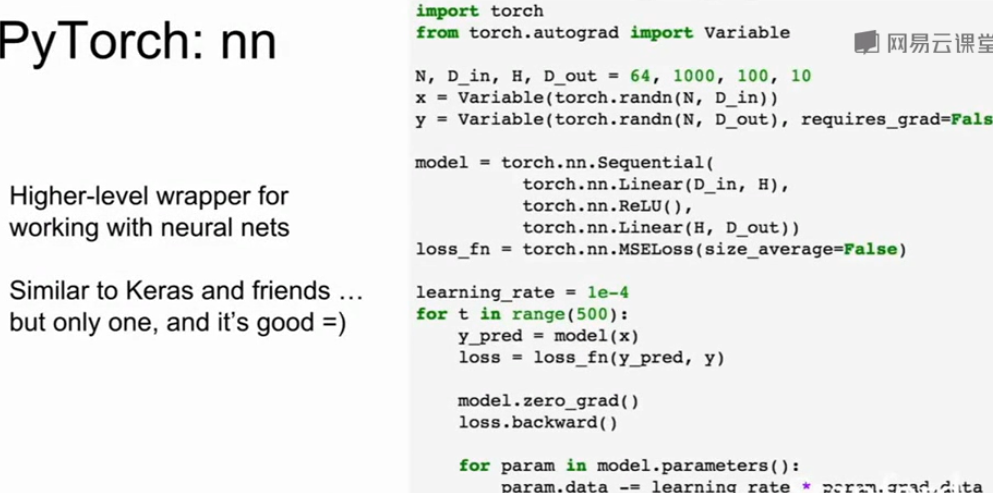
对不同的层定义不同的前向后向



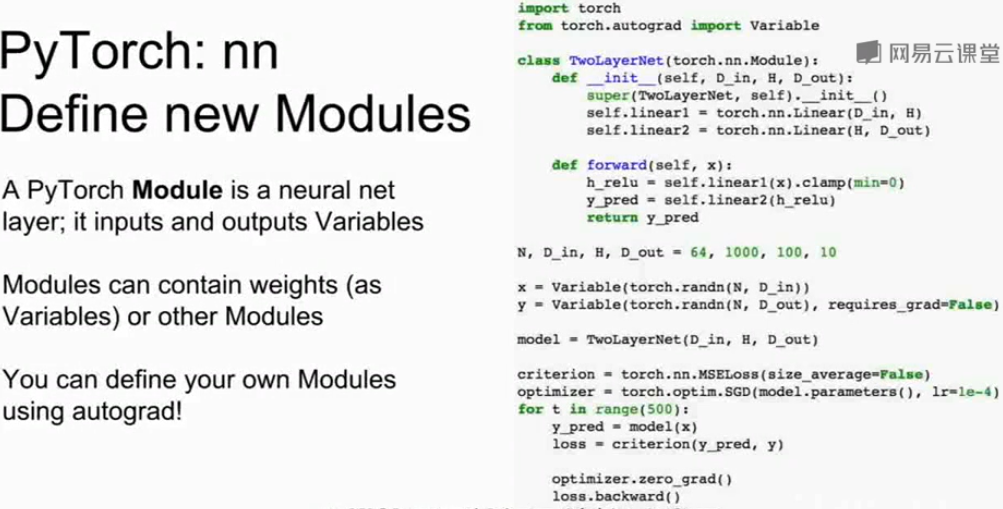
然后用在代码块中：



神经元相关高级包：nn

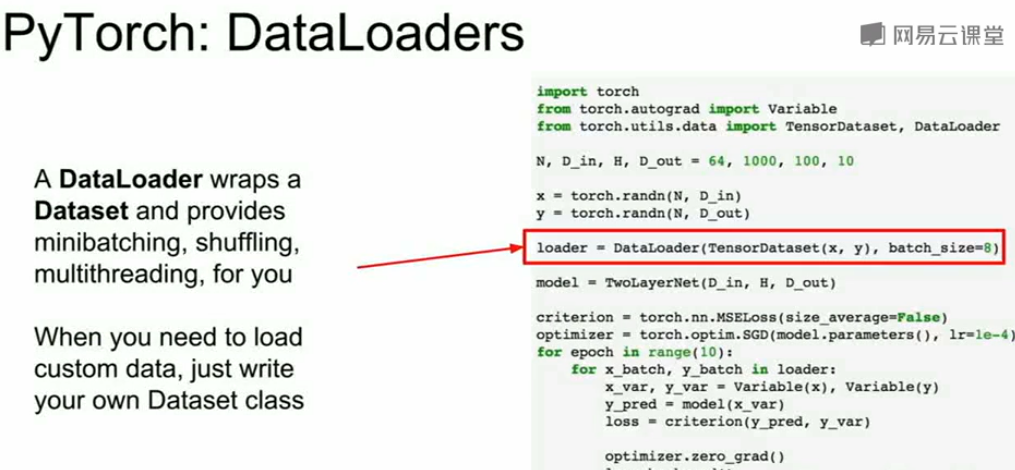


定义自己的nn类实例 **Module**，类似于自定义层



# DataLoaders:

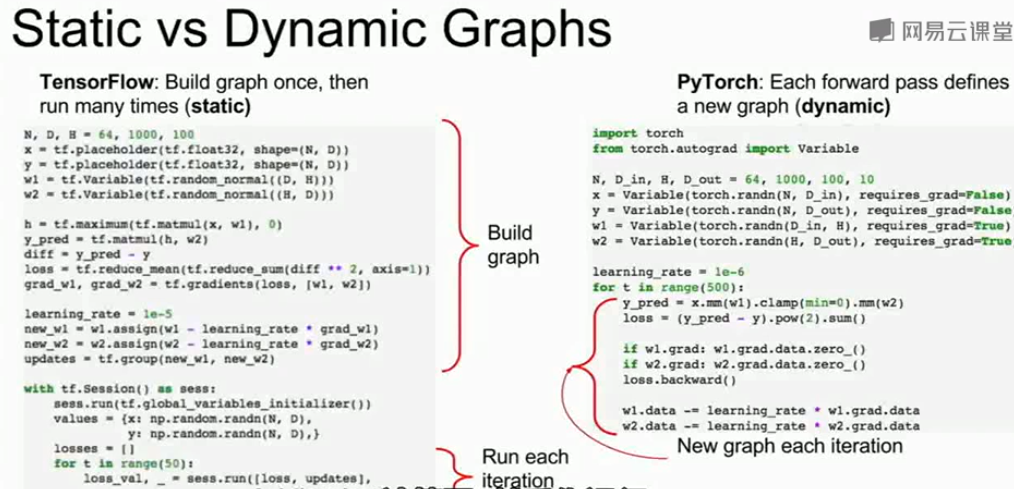
# 多线程，平行加载内存与硬盘等



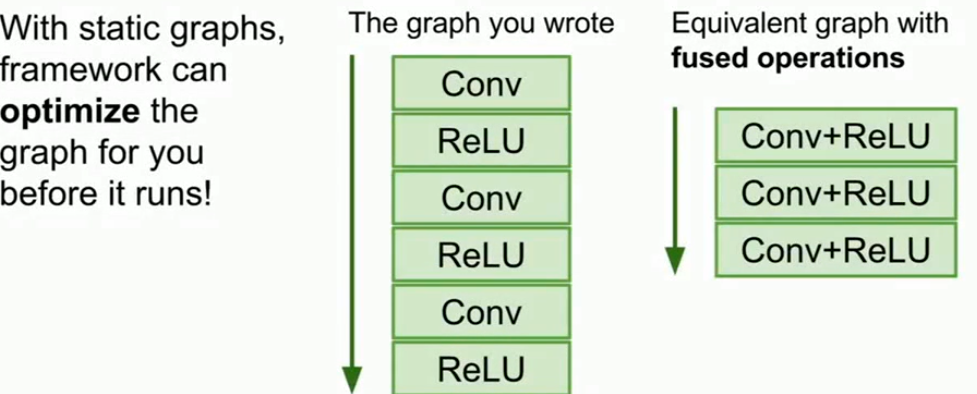
预训练模型：



## PyTorch ~ TensorFlow 动态~静态

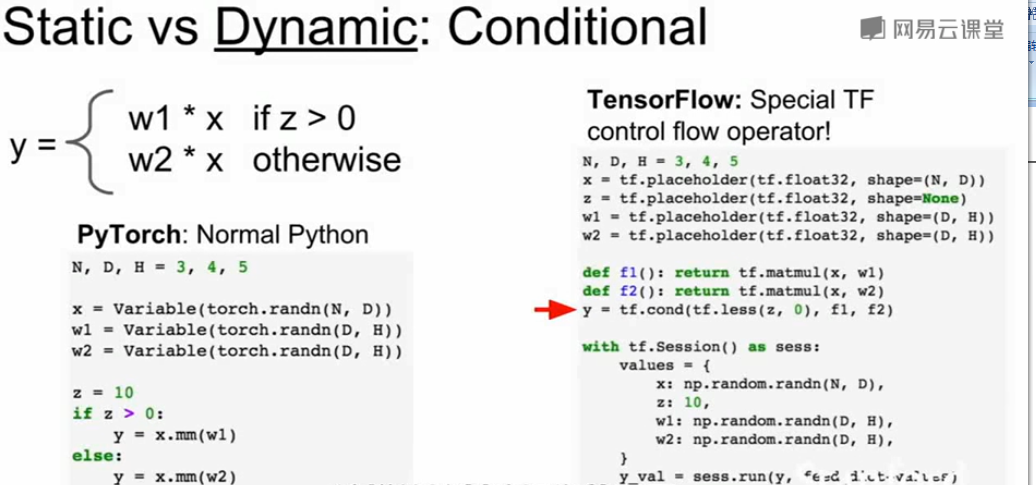


静态图更好优化



序列化：应用于C JAVA 等其他环境

条件判断时：动态图更方便



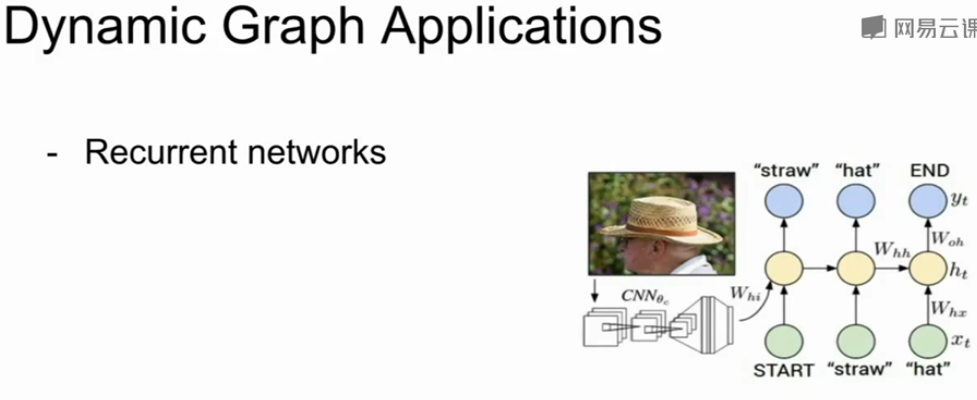
循环：PyTorch很容易，但TF要在一开始就建立所有的图，所以比较麻烦了，需要设置一个单独的标记（foldl）。

循环网络：

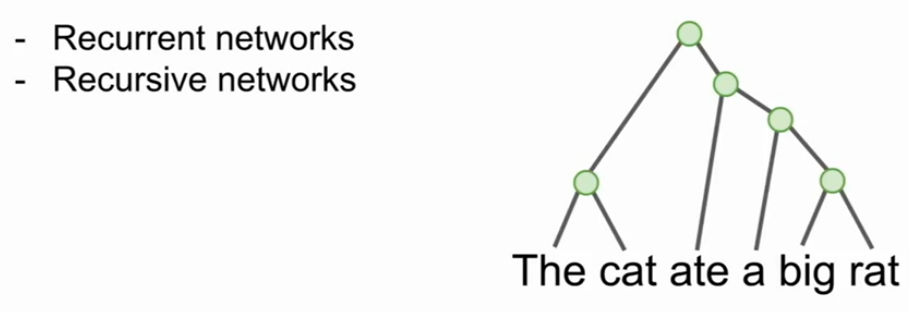
Karpathy and Fei-Fei, “Deep Visual-Semantic Alignments for

Generating Image Descriptions”, CVPR 2015

Figure copyright IEEE, 2015. Reproduced for educational purposes.



递归神经网络：



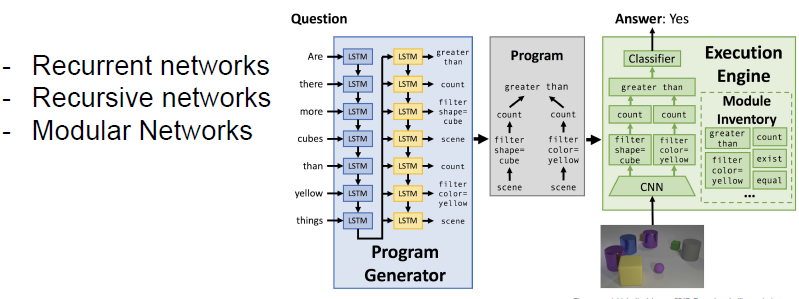
这样在TF中很难实现，但是PT中就比较好实现

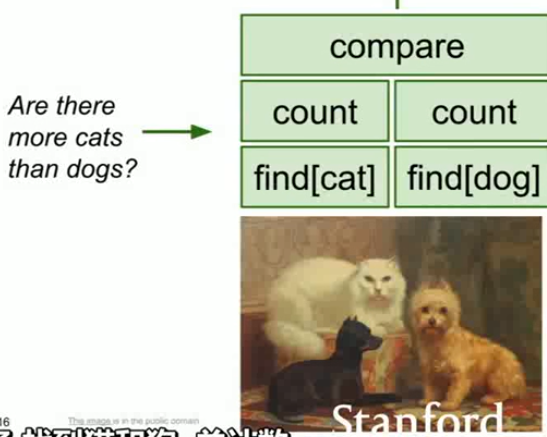
模块神经网络 neuron module network – 针对可视化问题

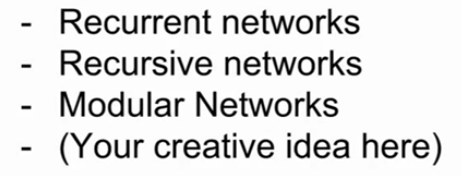
Andreas et al, “Neural Module Networks”, CVPR 2016

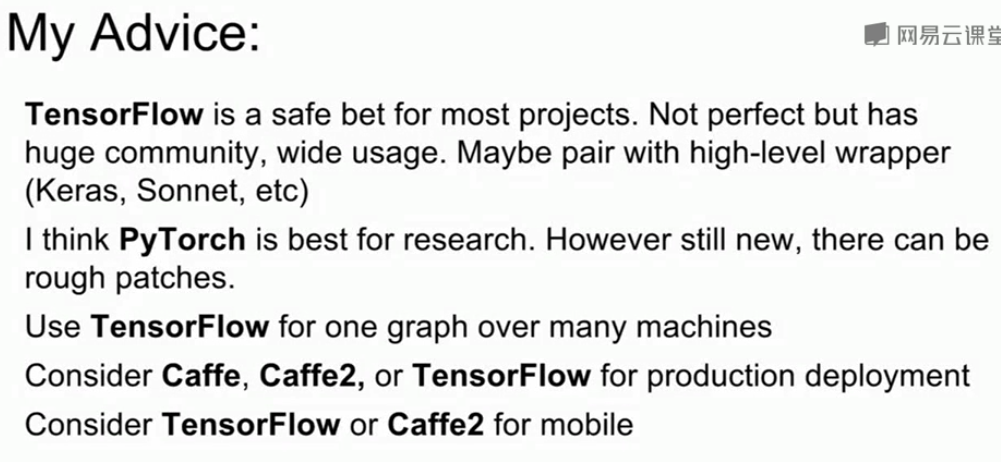
Andreas et al, “Learning to Compose Neural Networks for Question Answering”, NAACL 2016

Johnson et al, “Inferring and Executing Programs for Visual Reasoning”, ICCV 2017









下一节：主流CNN结构与实例！