# 10分钟内为PyTorch用户提供MXNet

作者：，亚马逊首席科学家翻译自：

由于其易于理解的API和完全必要的方法，已经迅速成为最流行的深度学习框架之一。但是，您可能不知道MXNet包含了PyTrror的简单性和灵活性，同时允许您对网络进行杂交以充分利用符号图的性能优化。

在接下来的10分钟内，我们将向您展示两个框架之间的快速比较，并向您展示在从一个框架切换到另一个框架时，学习曲线可以有多小。本文以MNIST数据集上的图像分类为例。

### 安装

PyTorch默认使用conda进行安装，例如：

对于MXNet，我们使用pip。我们还可以使用--pre标志安装夜间版本：

### 多维矩阵

对于多维矩阵，PyTorch遵循Torch的命名约定并引用“张量”。MXNet遵循NumPy的约定，并引用“ndarrays”。在这里，我们创建一个二维矩阵，其中每个元素初始化为1。然后我们在每个元素中添加1并打印。

* 火把：
* MXNet公司：

除了包名之外，最大的区别是MXNet的shape输入参数需要像NumPy那样作为一个包含在括号中的元组传递。

### 模型训练

下面让我们看一个稍微复杂一点的例子。这里我们创建了一个多层感知器（MLP）来训练MINST数据集上的模型。我们把实验分成四个部分。

### 1-读取数据

我们下载MNIST数据集并将其加载到内存中，以便可以逐个读取批。

* 火把：
* MXNet公司：

这里的主要区别是MXNet首先使用transform\_来表示数据转换是在数据批的第一个元素MNIST图片上完成的，而不是在第二个元素标签上完成的。

### 2 - 创建模型

下面我们定义一个多层感知器（MLP），它有一个隐藏层，输出层有10个单位。

* 火把：
* MXNet公司：

为了构造神经网络，我们使用了顺序容器来逐层堆叠。MXNet与Pythorch的不同之处在于：

* 在MXNet中，不需要指定输入大小，它将被自动推断。
* 可以在完全连接和卷积层中直接指定激活函数。
* 您需要创建一个名称范围来为每个层附加一个唯一的名称：这是以后保存和重新加载模型所必需的。
* 您需要显式调用模型初始化函数。

对于顺序块，层被一个接一个地执行。要使用不同的执行模型，可以使用PyTorch继承nn.Module，然后自定义.forward（）函数的执行方式。类似地，在MXNet中，可以继承nn.Block来获得类似的结果。

### 3 - 损失函数及优化算法

* 火把：
* MXNet公司：

在这里我们选取一个交叉熵损失函数，并使用固定学习率为0.1的随机梯度下降算法。

### 4 - 培训

最后实现了训练算法。请注意，每次运行的结果可能会有所不同，因为权重将获得不同的初始化值，并且由于洗牌，数据将以不同的顺序读取。

* 火把
* MXNet公司

与PyTorch相比，MXNet的一些差异如下：

* 您不需要将输入放入变量中（这不再是必需的，因为Pythorch 0.4.0），但是您需要在mx.autograd.record（）范围内执行计算，以便在反向过程中自动区分它。
* 没有必要像Pytorch的trainer.zero\_grad（）那样每次都清除渐变，因为默认情况下，新渐变是写入的，而不是累积的。
* 在培训器上执行.step（）时，需要指定更新步骤大小（通常为批处理大小）。
* 您需要调用.asscalar（）将多维数组转换为标量。
* 在这个示例中，MXNet的速度是PyTorch的两倍。尽管你需要谨慎对待这样的玩具比较。

### 下一步

* 遵循更详细的MXNet教程：
* 如果您在MXNet中缺少PyTorch的某些功能，请在上创建问题或在上创建问题！
* Follow the [Gluon 60 minutes crash-course](https://gluon-crash-course.mxnet.io/)