# TensorFlow

## 语言：C++或Python

TensorFlow是Google一个使用数据流图进行数值计算的开源框架。这个框架被称为允许在任何CPU或GPU上进行计算的框架，无论是台式机，服务器还是移动设备，另外这个框架在Python编程语言中是可用的（这也是python在AI领域火爆的原因之一）。

## 优点

使用易于学习的语言（Python）；

使用计算图抽象；

可以使用可视化的TensorBoard；

## 缺点

很慢，因为Python不是编程语言中最快的；

缺乏许多预先训练的模型；

不完全开源；

# CNTK

## 语言：C++

我们可以称之为它是微软对Google的TensorFlow的回应。

微软的CNTK是一个增强分离计算网络模块化和维护的库，它提供了学习算法和模型描述。在需要大量服务器进行操作的情况下，CNTK可以同时利用多台服务器。

据说它的功能与Google的TensorFlow相近，但是，它更快，在这里了解更多。

## 优点

非常灵活；

允许分布式培训；

支持C++，C#，Java和Python；

## 缺点

它以一种新的语言——Network Description Language（NDL）来实现；

# Theano

## 语言：Python

Theano是TensorFlow的强有力竞争者，它是一个功能强大的Python库，允许以高效率的方式进行多维数组的数值操作。

该库使用GPU来执行数据密集型计算而不是CPU，因此操作效率很高。出于这个原因，Theano已经被用于为大规模的计算密集型操作长达10年的时间。然而，于2017年9月，Theano的1.0版本停止。

但这并不意味着它不再是一个强大的库，仍然可以随时进行深入的学习研究。

## 优点

优化CPU和GPU；

有效的计算任务；

## 缺点

与其他库相比，原生Theano有点低级；

需要与其他库一起使用，以获得高度的抽象；

AWS使用它有点bug；

# Caffe

“快速，开放的深度学习框架”

## 语言：C++

Caffe是一个强大的深度学习框架，像这个清单上的其他框架一样，深度学习的研究速度非常快。

借助Caffe，可以非常轻松地构建用于图像分类的卷积神经网络。Caffe在GPU上运行的也很不错，这有助于在运行期间提高速度。

## 优点

Python和MATLAB都可用；

表现的很好；

无需编写代码即可进行模型的训练；

## 缺点

对于RNN网络不太友好；

对于新体系结构不太友好；

# Keras

“为人类普及深度学习”

## 语言：Python

Keras是一个用Python编写的开源的神经网络。与TensorFlow，CNTK和Theano不同，Keras并不意味着是一个端到端的机器学习框架。

相反，它作为一个接口，提供一个高层次的抽象，这使得神经网络的配置变得简单，无论它在哪个框架上。

Google的TensorFlow目前支持Keras作为后端，而微软的CNTK也会在很短的时间内做到这一点。

## 优点

对用户非常友好；

很容易扩展；

在CPU和GPU上无缝运行；

与Theano和TensorFlow无缝工作；

## 缺点

不能有效的用作独立的框架；

# Torch

“一个开源的机器学习库”

## 语言：C

Torch是一个用于科学和数字操作的开源机器学习库。

这是一个基于Lua编程语言的库而不是Python。

它通过提供大量的算法，使得深度学习研究更容易，并且提高了效率和速度。它有一个强大的N维数组，这有助于切片和索引等操作。它还提供了线性代数程序和神经网络模型。

## 优点

非常灵活；

高水平的速度和效率；

大量的预训练模型可用；

## 缺点

缺乏即时使用的代码；

它基于一种不那么流行的语言：Lua；

# Accord.NET

“机器学习，计算机视觉，统计和.NET通用科学计算”

## 语言：C#

这里是一个为C#程序员存在的机器学习框架。Accord.NET框架是一个.NET机器学习框架，使音频和图像处理变的简单。

这个框架可以有效的处理数值优化，人工神经网络，甚至可视化。除此之外，Accord.NET对计算机视觉和信号处理功能非常强大，同时也使得算法的实现变的简单。

## 优点

有一个庞大而积极的开发团队；

有据可查的框架；

有质量可视化；

## 缺点

不是一个非常流行的框架；

比TensorFlow慢；

# Spark MLlib

“可扩展的机器学习库”

## 语言：Scala

Apache的Spark MLlib是一个非常可扩展的机器学习库。它非常适用于java，Scala，Python甚至R语言。它非常搞笑，因为它可以与Python库中的numpy进行互操作。

MLlib可以轻松插入到Hadoop工作流程中。它还提供了机器学习算法，如分类，回归和聚类。这个强大的库在处理大型数据时非常快速。

## 优点

对于大规模数据非常快速；

提供支持多语言；

## 缺点

不是很完美的学习曲线；

即插即用仅适用于Hadoop；

# Sci-kit learn

“用Python进行机器学习”

## 语言：Python

Sci-kit learn是一个非常强大的机器学习Python库，主要用于构建模型。

使用numpy，SciPy和natplotlib等其他库构建，对统计建模技术（如分类，回归和聚类）非常有效。

Sci-kit learn带有监督学习算法，无监督学习算法和交叉验证等功能。

## 优点

许多主要算法的可用性很高；

能够进行有效的数据挖掘；

## 缺点

不是创建模型的最佳选择；

GPU效率不高；

# MLPack

“可扩展的C++机器学习库”

## 语言：C++

MLPack是一个用C++实现的可扩展的机器学习库，因为它是用C++编写的，所以可想而知它对于内存管理非常的好。

MLPack以极高的速度运行，可以支持高质量的机器学习算法与库一起运行。这个库对新手很友好，它还提供了一个简单的API帮助新手使用。

## 优点

非常容易扩展；

Python和C++都可用；

## 缺点

没有具体的参考资料；