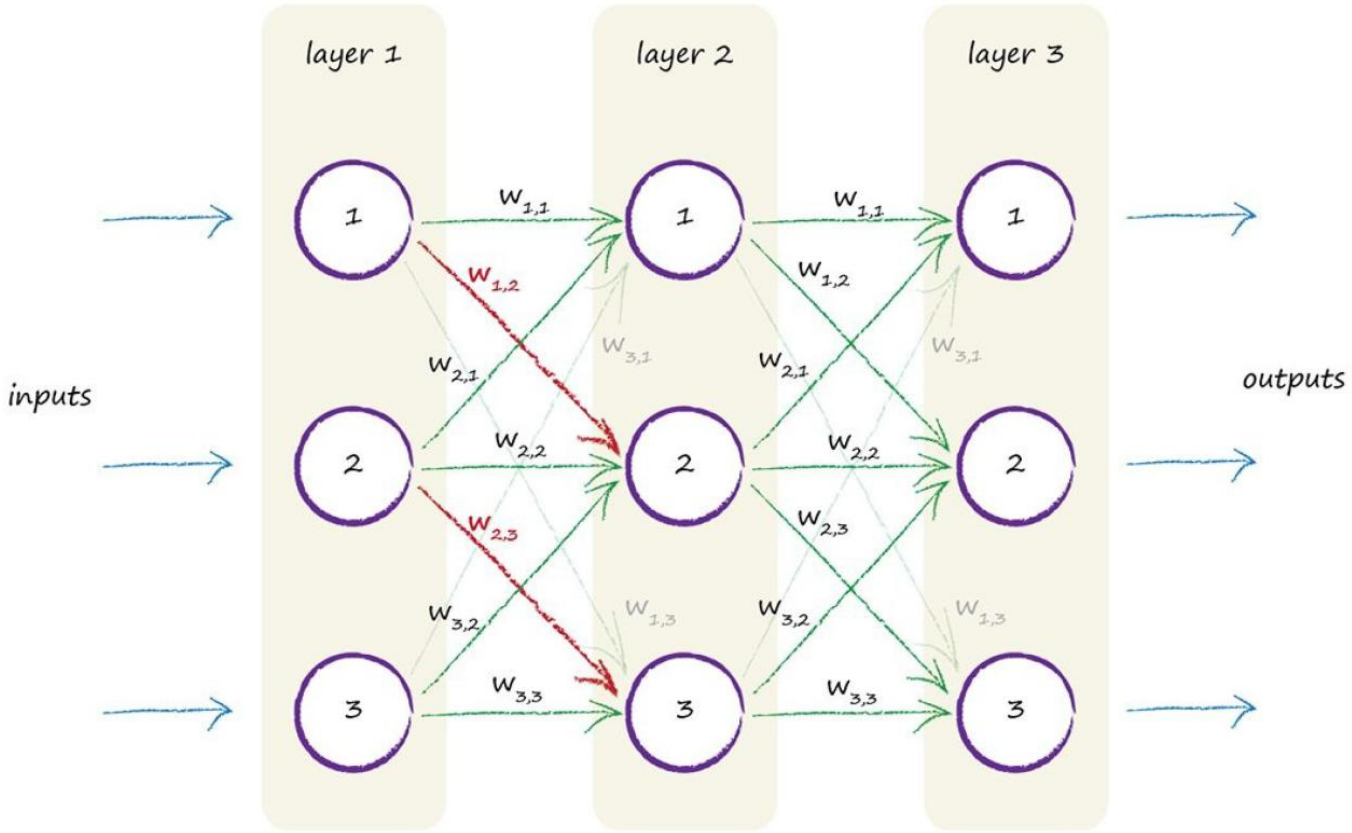
深度学习框架☞

回顾例子：手动构建



Pdf-part2_neural_network
Linear层调用

1. 常见的深度学习框架

- 随着深度学习的发展，深度学习框架如雨后春笋般诞生于高校和公司中。尤其是近两年，Google、Facebook、Microsoft等巨头都围绕深度学习重点投资了一系列新兴项目，他们也一直在支持一些开源的深度学习框架。目前研究人员正在使用的深度学习框架不尽相同，有TensorFlow、PyTorch、Caffe、Theano、Keras等。这些深度学习框架被应用于计算机视觉、语音识别、自然语言处理与生物信息学等领域，并获取了极好的效果。

1. 1 Theano

- Theano最初诞生于蒙特利尔大学LISA实验室，于2008年开始开发，是第一个有较大影响力的Python深度学习框架。

- Theano是一个Python库，可用于定义、优化和计算数学表达式，特别是多维数组（numpy.ndarray）。在解决包含大量数据的问题时，使用Theano编程可实现比手写C语言更快的速度，而通过GPU加速，Theano甚至可以比基于CPU计算的C语言快上好几个数量级。Theano结合了计算机代数系统（Computer Algebra System，CAS）和优化编译器，还可以为多种数学运算生成定制的C语言代码。对于包含重复计算的复杂数学表达式的任务而言，计算速度很重要，因此这种CAS和优化编译器的组合是很有用的。对需要将每一种不同的数学表达式都计算一遍的情况，Theano可以最小化编译、解析的计算量，但仍然会给出如自动微分那样的符号特征。

1.1 Theano

- Theano诞生于研究机构，服务于研究人员，其设计具有较浓厚的学术气息，但在工程设计上有较大的缺陷。一直以来，Theano因难调试、构建图慢等缺点为人所诟病。为了加速深度学习研究，人们在它的基础之上，开发了Lasagne、Blocks、PyLearn2和Keras等第三方框架，这些框架以Theano为基础，提供了更好的封装接口以方便用户使用。

- 2017年9月28日，在Theano 1.0正式版即将发布前夕，LISA实验室负责人，深度学习三巨头之一的Yoshua Bengio宣布Theano即将停止开发：“Theano is Dead”。尽管Theano即将退出历史舞台，但作为第一个Python深度学习框架，它很好地完成了自己的使命，为深度学习研究人员的早期拓荒提供了极大的帮助，同时也为之后的深度学习框架的开发奠定了基本设计方向：以计算图为框架的核心，采用GPU加速计算。

- 点评：由于Theano已经停止开发，不建议作为研究工具继续学习。

1.2 TensorFlow

- 2015年11月10日，Google宣布推出全新的机器学习开源工具TensorFlow。TensorFlow最初是由Google机器智能研究部门的Google Brain团队开发，基于Google 2011年开发的深度学习基础框架DistBelief构建起来的。TensorFlow主要用于机器学习和深度神经网络研究，但它是一个非常基础的系统，因此也可以应用于众多领域。由于Google在深度学习领域的巨大影响力和强大的推广能力，TensorFlow一经推出就获得了极大的关注，并迅速成为如今用户最多的深度学习框架。

1.2TensorFlow

- TensorFlow在很大程度上可以看做是Theano的后继者，不仅因为它们有很大一批共同的开发者，而且它们还拥有相近的设计理念，都是基于计算图实现自动微分系统。TensorFlow使用数据流图进行数值计算，图中的节点代表数学运算，而图中的边则代表这些节点之间传递的多维数组（张量）。

- TensorFlow编程接口支持Python和C++。随着1.0版本的公布，Java、Go、R和Haskell API的alpha版本也被支持。此外，TensorFlow还可在Google Cloud和AWS中运行。由于TensorFlow使用C++ Eigen库，所以库可在ARM架构上编译和优化。这也就意味着用户可以在各种服务器和移动设备上部署自己的训练模型，无需执行单独的模型解码器或者加载Python解释器。

1.2 TensorFlow

作为当前最流行的深度学习框架，TensorFlow获得了极大的成功，对它的评价也不绝于耳：

- 过于复杂的系统设计，TensorFlow在GitHub代码仓库的总代码量超过100万行。这么大的代码仓库，对于项目维护者来说维护成为了一个难以完成的任务，而对于读者来说，学习TensorFlow底层运行机制更是一个极其痛苦的过程，并且大多数时候这种尝试以放弃告终。
- 频繁变动的接口。TensorFlow的接口一直处于快速迭代之中，并且没有很好地考虑向后兼容性，这导致现在许多开源代码已经无法在新版的TensorFlow上运行，同时也间接导致了许多基于TensorFlow的第三方框架出现Bug。
- 接口设计过于晦涩难懂。在设计TensorFlow时，创造了图、会话、命名空间、Place-Holder等诸多抽象概念，对普通用户来说难以理解。同一个功能，TensorFlow提供了多种实现，这些实现良莠不齐，使用中还有细微的区别，很容易将用户带入坑中。
- 文档混乱脱节。TensorFlow作为一个复杂的系统，文档和教程众多，但缺乏明显的条理和层次，虽然查找很方便，但用户却很难找到一个真正循序渐进的入门教程。

1.2 TensorFlow

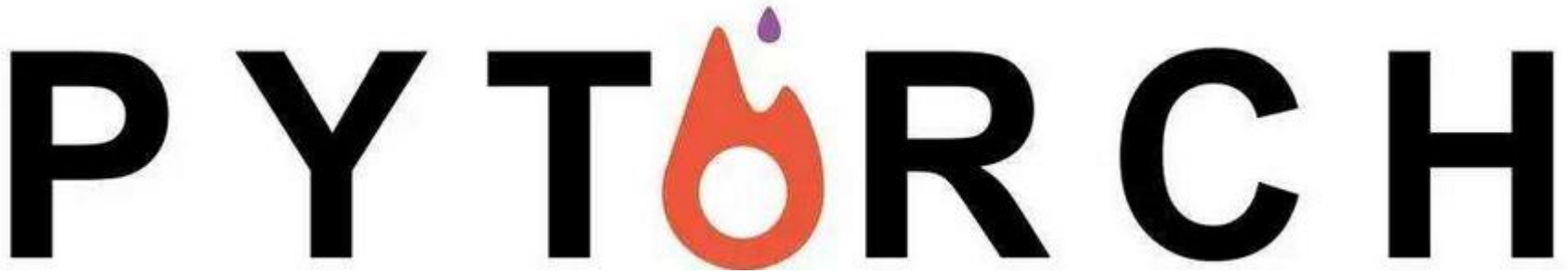
- 由于直接使用TensorFlow的生产力过于低下，包括Google官方等众多开发者尝试基于TensorFlow构建一个更易用的接口，包括Keras、Sonnet、TFLearn、TensorLayer、Slim、Fold、PrettyLayer等数不胜数的第三方框架每隔几个月就会在新闻中出现一次，但是又大多归于沉寂，至今TensorFlow仍没有一个统一易用的接口。

- 凭借Google强大的推广能力，TensorFlow已经成为当今最炙手可热的深度学习框架，但是由于自身的缺陷，TensorFlow离最初的设计目标还很遥远。另外，由于Google对TensorFlow略显严格的把控，目前各大公司都在开发自己的深度学习框架。

- 点评：不完美但最流行的深度学习框架，社区强大，适合生产环境。

1.3 PyTorch

2017年1月，Facebook人工智能研究院（FAIR）团队在GitHub上开源了PyTorch，并迅速占领GitHub热度榜榜首。下面是PyTorch的Logo（英文Torch是火炬的意思，所以Logo中有火焰）：

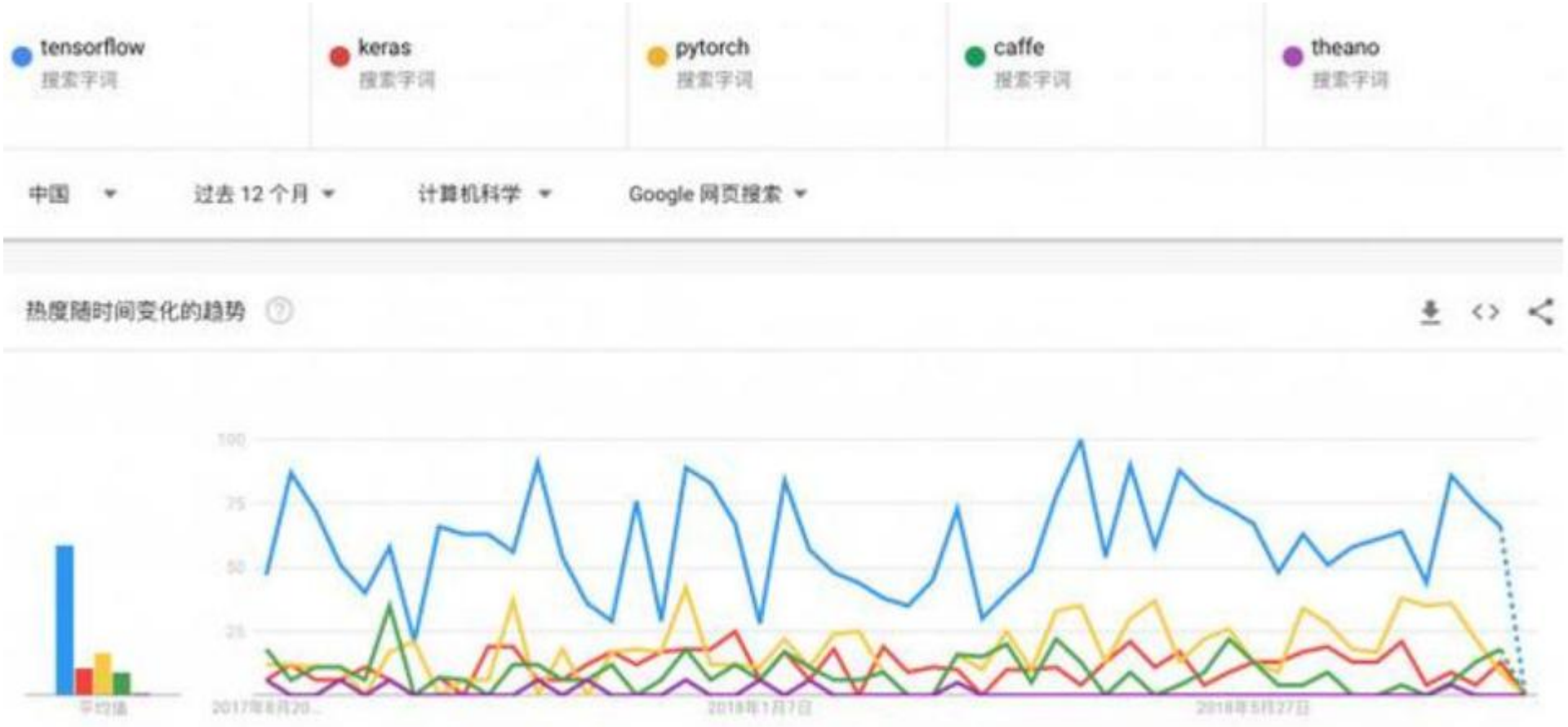


- 考虑到Python在计算科学领域的领先地位，以及其生态完整性和接口易用性，几乎任何框架都不可避免地要提供Python接口。

- 终于，在2017年，Torch的幕后团队推出了PyTorch。

- PyTorch不是简单地封装Lua Torch提供Python接口，而是对Tensor之上的所有模块进行了重构，并新增了最先进的自动求导系统，成为当下最流行的动态图框架。

- PyTorch一经推出就立刻引起了广泛关注，并迅速在研究领域流行起来。下图所示为Google指数，PyTorch自发布起关注度就在不断上升，PyTorch的热度已然超过其他三个框架（Caffe、MXNet和Theano），并且其热度还在持续上升中。



1.4 Keras

- Keras是一个高层神经网络API，由纯Python编写而成并使用TensorFlow、Theano以及CNTK作为后端。Keras为支持快速实验而生，能够把想法迅速转换为结果。Keras应该是深度学习框架中最容易上手的一个，它提供了一致而简洁的的API，能够极大地减少一般应用下用户的工作量，避免用户重复造轮子。

- 严格意义上讲，Keras并不能称为一个深度学习框架，它更像一个深度学习接口，它构建于第三方框架之上。Keras的缺点很明显：过度封装导致丧失灵活性。Keras最初作为Theano的高级API而诞生，后来增加了TensorFlow和CNTK作为后端。为了屏蔽后端的差异性，提供一致的用户接口，Keras做了层层封装，导致用户在新增操作或是获取底层的数据信息时过于困难。同时，过度封装也使得Keras的程序过于缓慢，许多Bug都隐藏在封装之中，在绝大多数场景下，Keras是所介绍的所有框架中最慢的一个。

1.4 Keras

- 学习Keras十分容易，但是很快就会遇到瓶颈，因为它缺少灵活性。另外，在使用Keras的大多数时间里，用户主要是在调用接口，很难真正学到深度学习的内容。

- 点评：入门很简单，但是不够灵活，使用受限。

1.5 Caffe/Caffe2

- Caffe的全称是Convolutional Architecture for Fast Feature Embedding，它是一个清晰、高效的深度学习框架，核心语言是C++，它支持命令行、Python和MATLAB接口，既可以在CPU上运行，也可以在GPU上运行。

- Caffe的优点是简洁快速，缺点是缺少灵活性。不同于Keras因为太多的封装导致灵活性丧失，Caffe灵活性的缺失主要是因为它的设计。在Caffe中最主要的抽象对象是层，每实现一个新的层，必须要利用C++实现它的前向传播和反向传播代码，而如果想要新层运行在GPU上，还需要同时利用CUDA实现这一层的前向传播和反向传播。这种限制使得不熟悉C++和CUDA的用户扩展Caffe十分困难。

注：CUDA（Compute Unified Device Architecture），CUDA™是一种由显卡厂商NVIDIA推出的通用并行计算架构，该架构使GPU能够解决复杂的计算问题。它包含了CUDA指令集架构（ISA）以及GPU内部的并行计算引擎。

1.5 Caffe/Caffe2

- Caffe凭借其易用性、简洁明了的源码、出众的性能和快速的原型设计获取了众多用户，曾经占据深度学习领域的半壁江山。但是在深度学习新时代到来之时，Caffe已经表现出明显的力不从心，诸多问题逐渐显现（包括灵活性缺少、扩展难、依赖众多环境难以配置、应用局限等）。尽管现在在GitHub上还能找到许多基于Caffe的项目，但是新的项目已经越来越少。

- Caffe的作者从加州大学伯克利分校毕业后加入了Google，参与过TensorFlow的开发，后来离开Google加入FAIR，担任工程主管，并开发了Caffe2。Caffe2是一个兼具表现力、速度和模块性的开源深度学习框架。它沿袭了大量的Caffe设计，可解决多年了在Caffe的使用和部署中发现的瓶颈问题。Caffe2的设计追求轻量级，在保有扩展性和高性能的同时，Caffe2也强调了便携性。Caffe2从一开始就以性能、扩展、移动端部署作为主要设计目标。Caffe2的核心C++库能提供速度和便携性，而其Python和C++ API使用户可以轻松地在Linux、Windows、iOS、Android，甚至Raspberry Pi和NVIDIA Tegra上进行原型设计、训练和部署。

1.5 Caffe/Caffe2

- Caffe2继承了Caffe的优点，在速度上令人印象深刻。Facebook人工智能实验室与应用机器学习团队合作，利用Caffe2大幅加速机器视觉任务的模型训练过程，仅需1小时就训练完ImageNet这样超大规模的数据集。然而尽管如此，Caffe2仍然是一个不太成熟的框架，官网至今没提供完整的文档，安装也比较麻烦，编译过程时常出现异常，在GitHub上也很少找到相应的代码。

- 极盛的时候，Caffe占据了计算机视觉研究领域的半壁江山，虽然如今Caffe已经很少用于学术界，但是仍有不少计算机视觉相关的论文使用Caffe。由于其稳定、出众的性能，不少公司还在使用Caffe部署模型。Caffe2尽管做了许多改进，但是还远没有达到替代Caffe的地步。

- 点评：文档不够完善，但性能优异，几乎全平台支持（Caffe2），适合生产环境。

1.6 MXNet

- MXNet是一个深度学习库，支持C++、Python、R、Scala、Julia、MATLAB及JavaScript等语言；支持命令和符号编程；可以运行在CPU、GPU、集群、服务器、台式机或者移动设备上。MXNet是CXXNet的下一代，CXXNet借鉴了Caffe的思想，但是在实现上更干净。在2014年的NIPS上，同为上海交大校友的陈天奇与李沐碰头，讨论到各自在做深度学习Toolkits的项目组，发现大家普遍在做很多重复性的工作，例如文件loading等。于是他们决定组建DMLC（Distributed/Deep Machine Learning Community），号召大家一起合作开发MXNet，发挥各自的特长，避免重复造轮子。

- MXNet以其超强的分布式支持，明显的内存、显存优化为人所称道。同样的模型，MXNet往往占用更小的内存和显存，并且在分布式环境下，MXNet展现出了明显优于其他框架的扩展性能。

1.6 MXNet

- 由于MXNet最初由一群学生开发，缺乏商业应用，极大地限制了MXNet的使用。2016年11月，MXNet被AWS正式选择为其云计算的官方深度学习平台。2017年1月，MXNet项目进入Apache基金会，成为Apache的孵化器项目。

- 尽管MXNet拥有最多的接口，也获得了不少人的支持，但其始终处于一种不温不火的状态。这在很大程度上归结于推广不给力及接口文档不够完善。MXNet长期处于快速迭代的过程，其文档却长时间未更新，导致新手用户难以掌握MXNet，老用户常常需要查阅源码才能真正理解MXNet接口的用法。

- 为了完善MXNet的生态圈，推广MXNet，MXNet先后推出了包括MinPy、Keras和Gluon等诸多接口，但前两个接口目前基本停止了开发，Gluon模仿PyTorch的接口设计，MXNet的作者李沐更是亲自上阵，在线讲授如何从零开始利用Gluon学习深度学习，诚意满满，吸引了许多新用户。

- 点评：文档略混乱，但分布式性能强大，语言支持最多，适合AWS云平台使用。

1.7 CNTK

- 2015年8月，微软公司在CodePlex上宣布由微软研究院开发的计算网络工具集CNTK将开源。5个月后，2016年1月25日，微软公司在他们的GitHub仓库上正式开源了CNTK。早在2014年，在微软公司内部，黄学东博士和他的团队正在对计算机能够理解语音的能力进行改进，但当时使用的工具显然拖慢了他们的进度。于是，一组志愿者组成的开发团队构想设计了他们自己的解决方案，最终诞生了CNTK。

- 根据微软开发者描述，CNTK的性能比Caffe、Theano、TensorFlow等主流工具都要强。CNTK支持CPU和GPU模式，和TensorFlow/Theano一样，它把神经网络描述成一个计算图的结构，叶子节点代表输入或者网络参数，其他节点代表计算步骤。在Microsoft内部使用的目的而开发的，一开始甚至没有Python接口，而是使用了一种几乎没什么人用的语言开发的，而且文档有些晦涩难懂，推广不是很给力，导致现在用户比较少。但就框架本身的质量而言，CNTK表现得比较均衡，没有明显的短板，并且在语音领域效果比较突出。

- 点评：社区不够活跃，但是性能突出，擅长语音方面的相关研究。

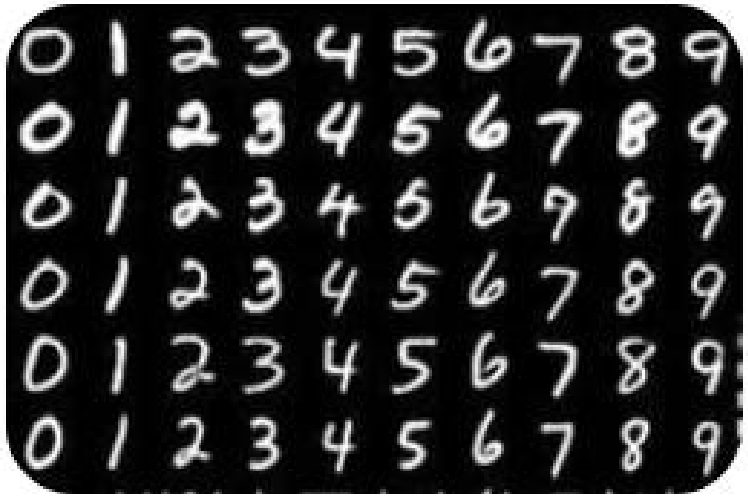
1.8 其他框架

- 除了上述的几个框架，还有不少框架，都有一定的影响力和用户。比如百度开源的PaddlePaddle，CMU开发的DyNet，简洁无依赖符合C++11标准的tiny-dnn，使用Java开发并且文档及其优秀的DeepLearning4J，还有英特尔开源的Nervana，Amazon开源的DSSTNE。这些框架各有优缺点，但是大多流行度和关注度不够，或者局限于一定的领域，因此不做过多的介绍。此外，还有许多专门针对移动设备开发的框架，如CoreML、MDL，这些框架纯粹为部署而诞生，不具有通用性，也不适合作为研究工具，同样不做介绍。

2、AI实践：MNIST手写数字识别

- Mnist数据集包含70000张手写数字图片，分别是60000张训练图片和10000张测试图片，训练集由来自250个不同人手写的数字构成，一般来自高中生，一半来自工作人员，测试集（test set）也是同样比例的手写数字数据，并且保证了测试集和训练集的作者不同。

- 每个图片都是2828个像素点，数据集/会把一张图片的数据转成一个2828=784的一维向量存储起来。每张图是0-9的手写数字黑底白字的图片，存储时，黑色用0表示，白色用0-1的浮点数表示。



2、AI实践：MNIST手写数字识别

- 为什么讲Keras呢，因为它简洁好用啊。Keras提供了一个简单和模块化的API来构建和训练我们需要的神经网络，比如卷积神经网络，循环神经网络等等。还有一个优点就是使用Keras可以不用关心大部分函数实现的复杂细节，可真的太棒了。
- 安装依赖的库和深度学习库，Numpy、scipy、scikit-learn、matplotlib、pandas、graphviz、pydot、h5py、Theano、Tensofflow、Keras。打开cmd，输入pip install packagename即可。
- 使用Keras搭建卷积神经网络LeNet用于在MNIST数据集上实现手写数字识别。

LeNet-5 architecture

Layer	Type	Maps	Size	Kernel size	Stride	Activation
Out	Fully Connected	–	10	–	–	RBF
F6	Fully Connected	–	84	–	–	tanh
C5	Convolution	120	1×1	5×5	1	tanh
S4	Avg Pooling	16	5×5	2×2	2	tanh
C3	Convolution	16	10×10	5×5	1	tanh
S2	Avg Pooling	6	14×14	2×2	2	tanh
C1	Convolution	6	28×28	5×5	1	tanh
In	Input	1	32×32	–	–	–

具体见Keras-MNIST-LeNet.html