直观梳理深度学习：（一）深度学习基础

张皓

zhangh0214@gmail.com

引言

深度学习目前已成为发展最快、最令人兴奋的机器学习领域之一，许多卓有建树的论文已经发表，而且已有很多高质量的开源深度学习框架可供使用。然而，论文通常非常简明扼要并假设读者已对深度学习有相当的理解，这使得初学者经常卡在一些概念的理解上，读论文似懂非懂，十分吃力。另一方面，即使有了简单易用的深度学习框架，如果对深度学习常见概念和基本思路不了解，面对现实任务时不知道如何设计、诊断、及调试网络，最终仍会束手无策。

本系列文章旨在直观系统地梳理深度学习各领域常见概念与基本思想，使读者对深度学习的重要概念与思想有一直观理解，做到"知其然，又知其所以然"，从而降低后续理解论文及实际应用的难度。本系列文章力图用简练的语言加以描述，避免数学公式和繁杂细节。本文是该系列文章中的第一篇，旨在介绍深度学习基础概念、优化算法、 调参基本思路、正则化方式等，后续文章将关注深度学习在自然语言处理、语音识别、和计算机视觉领域的应用。

基本概念

深度学习是为了解决表示学习难题而被提出的。本节，我们介绍这些深度学习相关的基本概念。

表示学习（representation learning） 机器学习旨在自动地学到从数据的表示（representation）到数据的标记（label）的映射。随着机器学习算法的日趋成熟，人们发现，在某些领域（如图像、语音、文本等），如何从数据中提取合适的表示成为整个任务的瓶颈所在，而数据表示的好坏直接影响后续学习任务（所谓 garbage in，garbage out）。与其依赖人类专家设计手工特征（难设计还不见得好用），表示学习希望能从数据中自动地学到从数据的原始形式到数据的表示之间的映射。

深度学习（deep learning，DL） 表示学习的理想很丰满，但实际中人们发现从数据的原始形式直接学得数据表示这件事很难。深度学习是目前最成功的表示学习方法，因此，目前国际表示学习大会（ICLR）的绝大部分论文都是关于深度学习的。深度学习是把表示学习的任务划分成几个小目标，先从数据的原始形式中先学习比较低级的表示，再从低级表示学得比较高级的表示。这样，每个小目标比较容易达到，综合起来我们就完成表示学习的任务。这类似于算法设计思想中的分治法（divide-and-conquer）。

深度神经网络（deep neural networks，DNN） 深度学习目前几乎唯一行之有效的实现形式。简单的说，深度神经网络就是很深的神经网络。我们利用网络中逐层对特征进行加工的特性，逐渐从低级特征提取高级特征。除了深度神经网络之外，有学者在探索其他深度学习的实现形式，比如深度森林。

深度神经网络目前的成功取决于三大推动因素。1. 大数据。当数据量小时，很难从数据中学得合适的表示，而传统算法+特征工程往往能取得很好的效果；2. 计算能力。大的数据和大的网络需要有足够的快的计算能力才能使得模型的应用成为可能。3. 算法创新。现在很多算法设计关注在如何使网络更好地训练、更快地运行、取得更好的性能。

多层感知机（multi-layer perceptrons，MLP） 多层由全连接层组成的深度神经网络。多层感知机的最后一层全连接层实质上是一个线性分类器，而其他部分则是为这个线性分类器

学习一个合适的数据表示，使倒数第二层的特征线性可分。

激活函数（activation function） 神经网络的必要组成部分。如果没有激活函数，多次线性运算的堆叠仍然是一个线性运算，即不管用再多层实质只起到了一层神经网络的作用。一个好的激活函数应满足以下性质。1. 不会饱和。sigmoid 和 tanh 激活函数在两侧尾端会有饱和现象，这会使导数在这些区域接近零，从而阻碍网络的训练。2. 零均值。ReLU 激活函数的输出均值不为零，这会影响网络的训练。3. 容易计算。

迁移学习（transfer learning） 深度学习下的迁移学习旨在利用源任务数据辅助目标任务数据下的学习。迁移学习适用于源任务数据比目标任务数据多，并且源任务中学习得到的低层特征可以帮助目标任务的学习的情形。在计算机视觉领域，最常用的源任务数据是 ImageNet。对 ImageNet 预训练模型的利用通常有两种方式。1. 固定特征提取器。用 ImageNett 预训练模型提取目标任务数据的高层特征。2. 微调（fine-tuning）。以 ImageNet 预训练模型作为目标任务模型的初始化初始化权值，之后在目标任务数据上进行微调。

多任务学习（multi-task learning） 与其针对每个任务训练一个小网络，深度学习下的多任务学习旨在训练一个大网络以同时完成全部任务。这些任务中用于提取低层特征的层是共享的，之后产生分支，各任务拥有各自的若干层用于完成其任务。多任务学习适用于多个任务共享低层特征，并且各个任务的数据很相似的情况。

端到端学习（end-to-end learning） 深度学习下的端到端学习旨在通过一个深度神经网络直接学习从数据的原始形式到数据的标记的映射。端到端学习并不应该作为我们的一个追求目标，是否要采用端到端学习的一个重要考虑因素是：有没有足够的数据对应端到端的过程，以及我们有没有一些领域知识能够用于整个系统中的一些模块。

优化算法

在网络结构确定之后，我们需要对网络的权值（weights）进行优化。本节，我们介绍优化深度神经网络的基本思想。

梯度下降（gradient descent，GD） 想象你去野足但却迷了路，在漆黑的深夜你一个人被困住山谷中，你知道谷底是出口但是天太黑了根本看不清楚路。于是你确定采取一个贪心(greedy)算法：先试探在当前位置往哪个方向走下降最快（即梯度方向），再朝着这个方向走一小步，重复这个过程直到你到达谷底。这就是梯度下降的基本思想。

梯度下降算法的性能大致取决于三个因素。1. 初始位置。如果你初始位置就离谷底很近，自然很容易走到谷底。2. 山谷地形。如果山谷是“九曲十八弯”，很有可能你在里面绕半天都绕不出来。3. 步长。你每步迈多大，当你步子迈太小，很可能你走半天也没走多远，而当你步子迈太大，一不小心就容易撞到旁边的悬崖峭壁，或者错过了谷底。

误差反向传播（error back-propagation，BP） 结合微积分中链式法则和算法设计中动态规划思想用于计算梯度。 直接用纸笔推导出中间某一层的梯度的数学表达式是很困难的，但链式法则告诉我们，一旦我们知道后一层的梯度，再结合后一层对当前层的导数，我们就可以得到当前层的梯度。动态规划是一个高效计算所有梯度的实现技巧，通过由高层往低层逐层计算梯度，避免了对高层梯度的重复计算。

滑动平均（moving average） 要前进的方向不再由当前梯度方向完全决定，而是最近几次梯度方向的滑动平均。利用滑动平均思想的优化算法有带动量（momentum）的 SGD、Nesterov 动量、Adam（ADAptive Momentum estimation）等。

自适应步长 自适应地确定权值每一维的步长。当某一维持续震荡时，我们希望这一维的步长小一些；当某一维一直沿着相同的方向前进时，我们希望这一维的步长大一些。利用自适应步长思想的优化算法有 AdaGrad、RMSProp、Adam 等。

学习率衰减 当开始训练时，较大的学习率可以使你在参数空间有更大范围的探索；当优化

接近收敛时，我们需要小一些的学习率使权值更接近局部最优点。

深度神经网络优化的困难　有学者指出，在很高维的空间中，局部最优是比较少的，而大部分梯度为零的点是鞍点。平原区域的鞍点会使梯度在很长一段时间内都接近零，这会使得拖慢优化过程。

初始化

权值初始化对网络优化至关重要。早年深度神经网络无法有效训练的一个重要原因就是早期人们对初始化不太重视。本节，我们介绍几个适用于深度神经网络的初始化方法。

初始化的基本思想　方差不变，即设法对权值进行初始化，使得各层神经元的方差保持不变。

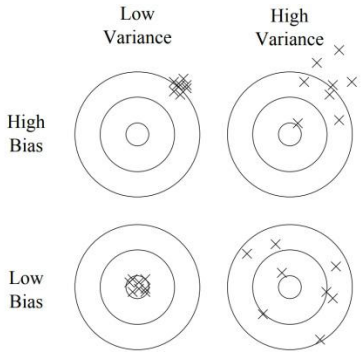
Xavier 初始化　从高斯分布或均匀分布中对权值进行采样，使得权值的方差是 1/n，其中 n 是输入神经元的个数。该推导假设激活函数是线性的。

He 初始化/MSRA 初始化　从高斯分布或均匀分布中对权值进行采样，使得权值的方差是 2/n。该推导假设激活函数是 ReLU。因为 ReLU 会将小于 0 的神经元置零，大致上会使一半的神经元置零，所以为了弥补丢失的这部分信息，方差要乘以 2。

批量规范化（batch-normalization，BN）　每层显式地对神经元的激活值做规范化，使其具有零均值和单位方差。批量规范化使激活值的分布固定下来，这样可以使各层更加独立地进行学习。批量规范化可以使得网络对初始化和学习率不太敏感。此外，批量规范化有些许正则化的作用，但不要用其作为正则化手段。

偏差/方差（bias/variance）

优化完成后，你发现网络的表现不尽如人意，这时诊断网络处于高偏差/高方差状态是对你下一步调参方向的重要指导。与经典机器学习算法有所不同，因为深度神经网络通常要处理非常高维的特征，所以网络可能同时处于高偏差/高方差的状态，即在特征空间的一些区域网络处于高偏差，而在另一些区域处于高方差。本节，我们对偏差/方差作一简要介绍。



偏差　偏差度量了网络的训练集误差和贝叶斯误差（即能达到的最优误差）的差距。高偏差的网络有很高的训练集误差，说明网络对数据中隐含的一般规律还没有学好。当网络处于高偏差时，通常有以下几种解决方案。1. 训练更大的网络。网络越大，对数据潜在规律的拟合能力越强。2. 更多的训练轮数。通常训练时间越久，对训练集的拟合能力越强。3. 改变网络结构。不同的网络结构对训练集的拟合能力有所不同。

方差　方差度量了网络的验证集误差和训练集误差的差距。高方差的网络学习能力太强，把训练集中自身独有的一些特点也当作一般规律学得，使网络不能很好的泛化（generalize）到验证集。当网络处于高方差时，通常有以下几种解决方案。1. 更多的数据。这是对高方差问题最行之有效的解决方案。2. 正则化。3. 改变网络结构。不同的网络结构对方差也会有影响。

正则化（regularization）

正则化是解决高方差问题的重要方案之一。本节，我们将对常用正则化方法做一介绍。

正则化的基本思想 正则化的基本思想是使网络的有效大小变小。网络变小之后，网络的拟合能力随之降低，这会使网络不容易过拟合到训练集。

L2 正则化 L2 正则化倾向于使网络的权值接近 0。这会使前一层神经元对后一层神经元的影响降低，使网络变得简单，降低网络的有效大小，降低网络的拟合能力。L2 正则化实质上是对权值做线性衰减，所以 L2 正则化也被称为权值衰减（weight decay）。

随机失活（dropout） 在训练时，随机失活随机选择一部分神经元，使其置零，不参与本次优化迭代。随机失活减少了每次参与优化迭代的神经元数目，使网络的有效大小变小。随机失活的作用有两点。1. 降低神经元之间耦合。因为神经元会被随机置零，所以每个神经元不能依赖于其他神经元，这会迫使每个神经元自身要能提取到合适的特征。2. 网络集成。随机失活可以看作在训练时每次迭代定义出一个新的网络，这些网络共享权值。在测试时的网络是这些网络的集成。

数据扩充（data augmentation） 这实质是获得更多数据的方法。当收集数据很昂贵，或者我们拿到的是第二手数据，数据就这么多时，我们从现有数据中扩充生成更多数据，用生成的“伪造”数据当作更多的真实数据进行训练。以图像数据做分类任务为例，把图像水平翻转、移动一定位置、旋转一定角度、或做一点色彩变化等，这些操作通常都不会影响这幅图像对应的标记。并且你可以尝试这些操作的组合，理论上讲，你可以通过这些组合得到无穷多的训练样本。

早停（early stopping） 随着训练的进行，当你发现验证集误差不再变化或者开始上升时，提前停止训练。

调参技巧

深度神经网络涉及很多的超参数，如学习率大小、L2 正则化系数、动量大小、批量大小、隐层神经元数目、层数、学习率衰减率等。本节，我们介绍调参的基本技巧。

随机搜索 由于你事先并不知道哪些超参数对你的问题更重要，因此随机搜索通常是比网格搜索（grid search）更有效的调参策略。

对数空间搜索 对于隐层神经元数目和层数，可以直接从均匀分布采样进行搜索。而对于学习率、L2 正则化系数、和动量，在对数空间搜索更加有效。例如：

```
import random
learning_rate = 10 ** random.uniform(-5, -1)     # From 1e-5 to 1e-1
weight_decay = 10 ** random.uniform(-7, -1)     # From 1e-7 to 1e-1
momentum = 1 - 10 ** random.uniform(-3, -1)     # From 0.9 to 0.999
```

实现技巧

图形处理单元（graphics processing units, GPU） 深度神经网络的高效实现工具。简单来说，CPU 擅长串行、复杂的运算，而 GPU 擅长并行、简单的运算。深度神经网络中的矩阵运算都十分简单，但计算量巨大。因此，GPU 无疑具有非常强大的优势。

向量化（vectorization） 代码提速的基本技巧。能少写一个 for 循环就少写一个，能少做一次矩阵运算就少做一次。实质是尽量将多次标量运算转化为一次向量运算；将多次向量运算转化为一次矩阵运算。因为矩阵运算可以并行，这将会比多次单独运算快很多。

参考文献

[1] Y. Bengio, N. Boulanger-Lewandowski, and R. Pascanu. Advances in optimizing recurrent networks. In proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, pages 8624--8628, 2013.

[2] J. Bergstra and Y. Bengio. Random search for hyper-parameter optimization. Journal of Machine Learning Research, 13(12):281--305, 2012.

[3] A. Choromanska, M. Henaff, M. Mathieu, G. B. Arous, and Y. LeCun. The loss surfaces of multilayer networks. In proceedings of the International Conference on Artificial Intelligence and Statistics, pages 192--204, 2015.

[4] P. Domingos. A few useful things to know about machine learning. Communications of the ACM, 55(10):78--87, 2012.

[5] J. Duchi, E. Hazan, and Y. Singer. Adaptive subgradient methods for online learning and stochastic optimization. Journal of Machine Learning Research, 12(12):2121--2159, 2011.

[6] I. Goodfellow, Y. Bengio, and A. Courville. Deep learning. MIT press, 2016.

[7] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In proceedings of the IEEE International Conference on Computer Vision, pages 1026--1034, 2015.

[8] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In proceedings of the International Conference on Machine Learning, pages 448-456, 2015.

[9] D. Kingma and J. Ba. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980, 2014.

[10] F.-F. Li, A. Karpathy, and J. Johnson. CS231n: Convolutional Neural Networks for Visual Recognition. Stanford, 2016.

[11] F.-F. Li, J. Johnson, and S. Yeung. CS231n: Convolutional Neural Networks for Visual Recognition. Stanford, 2017.

[12] A. Ng. Machine learning yearning. Draft, 2016.

[13] A. Ng. CS229: Machine learning. Stanford.

[14] A. Ng. Neural networks and deep learning. deeplearning.ai.

[15] A. Ng. Improving deep neural networks: Hyperparameter tuning, regularization and optimization. deeplearning.ai.

[16] A. Ng. Structuring machine learning projects. deeplearning.ai.

[17] S Ruder. An overview of multi-task learning in deep neural networks. arXiv preprint arXiv:1706.05098, 2017.

[18] N. Srivastava, G. E Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. Journal of machine learning research, 15(1):1929--1958, 2014.

[19] T. Tieleman and G. Hinton. Lecture 6.5-RMSProp: Divide the gradient by a running average of its recent magnitude. COURSERA: Neural networks for machine learning, 4(2):26--31, 2012.

[20] G. Xavier and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In proceedings of the International Conference on Artificial Intelligence and Statistics, pages 249--256, 2010.

[21] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson. How transferable are features in deep neural networks? In proceedings of Advances in neural information processing systems, pages 3320--3328, 2014.
[22] Z.-H. Zhou and J. Feng. Deep forest: Towards an alternative to deep neural networks. In Proceedings of the International Joint Conference on Artificial Intelligence, pages 3553--3559, 2017.
[23] 周志华. 机器学习. 清华大学出版社, 2016.