TL:DR

- 尽可能使用迁移学习。否则，对于已经是广泛研究的问题，先从复制网络结构开始。
-
  - 网络结构应该总是由实验和验证误差来确定。
  - 更深（层多），更轻（层小）的网络更难优化，但是更容易有更好的泛化误差。Deeper (more layers), thinner (smaller layers) networks are more difficult to optimize but tend to yield better generalization error.
- 一定要使用early stopping（早停），两种方法：
-
  - 在整个数据集上使用新的参数再次重新训练模型，在到达先前模型的早停点时停止训练。
  - 保留早停点时的参数，继续在所有数据上训练，当平均训练误差降到原先早停点以下时停止。
- 使用dropout或许是个好主意。
-
  - 使用0.8作为输入层的保留概率，0.5作为隐层的保留概率。
  - dropout会需要更大的网络和更长的迭代。
- ReLU是理想的激活函数。它也有缺陷，因此使用leaky ReLU或noisy ReLU或许有性能提升，但是会有更多的参数要调。
- 要得到差不多可以接收的性能，至少需要每个类5000个数据（>=10M可以达到人类的水平或更好）
-
  - 如果低于100,000个数据，使用K折交叉验证而不是train/validation/test划分的方法。
- 使用GPU能做到的最大的batch size。
-
  - 尝试不同batch size，从32开始以2的指数增加到256，对于太大的模型可以从16开始。
- 带动量和学习率衰减的随机梯度下降是个不错的开始。
-
  - 动量超参数通常的值有0.54,0.9和0.99。可以随时间调整，从小值开始，然后增大。
  - 或者，使用Adam或RMSProp。
  - 在分布式深度学习中使用异步SGD。
- 学习率是最重要的超参数。如果时间有限，主要花时间来调它。
-
  - 可以通过绘制学习曲线（目标函数随时间的变化）来观察学习率的情况
  - 最优学习率通常高于前100次迭代后产生最佳性能的学习率，但不会高到出现不稳定的情况。The optimal learning rate is typically higher than the learning rate that yields the best performance after the first ~100 iterations, but not so high that it causes instability.
- 对于计算机视觉：
-
  - 使用数据增强，只要图像没有本质上的改变。对比度归一化是另一个安全的预处理步骤。
  - 批量归一化，池化和填充在CNN中经常使用。在批量归一化的情况下或许不需要dropout。
- 对于自然语言处理：
-
  - LSTM通常效果比其他网络好。
  - 预训练的词嵌入（word2vec,word2glove等）是很强大的方法。
- 随机搜索通常比网格搜索更快的收敛到好的超参数设置。
- 调试策略：
-
  - 可视化模型：查看模型检测的图像样本，这对确定性能指标是否合理有帮助。
  - 可视化最差的错误情况：这可以发现预处理和打标签中存在的问题。
  - 当训练误差太高时先拟合一个小的数据集：这能检查出是欠拟合问题还是软件缺陷。
  - 监视激活和梯度的直方图：完成大概一个epoch。这能告诉我们神经元是否饱和及其饱和频率。梯度（值？）应该是参数的1%左右。The gradient should be about 1% of the parameter.

完整的笔记：

1 应用数学和机器学习基础

1 Introduction

每个类别至少需要大概5000个样本才能得到可接受的性能。大概每类需要10M个样本才能达到人类水平或更好。

4 数值计算

在深度学习中，我们通常会陷入局部最优而不是全局最优，这是由于复杂性和非凸优化的问题。

5 机器学习基础

- 如果模型具有最优的容量，但是在训练和测试误差之间仍有较大差距，去收集更多的数据来。
- 通常使用20%的训练集作为验证集。
- 如果数据少于100000个样本，使用k折交叉验证而不是训练/测试集划分的方法。
- 使用均方误差时，增加容量会降低偏差但是会增加方差。
- 贝叶斯方法在训练数据有限的时候泛化性能更好，但是训练样本较多时它的计算开销很大。
- 最常使用的损失函数是负对数似然。因此最小化损失函数也就是最大似然估计。

2 深度网络：现代实践

6 深度前馈网络

- ReLU对于前馈神经网络而言是一个完美的激活函数。
- - 它所基于的原则是，接近线性的特性更加容易优化。
  - sigmoid应该用于ReLU不能使用的情况。例如RNN，很多的概率模型，和一些自动编码机。
- 在基于梯度的优化问题上，由于梯度消失的原因，交叉熵相比MSE和MAE更好。
- ReLU的优点：减少了梯度消失的概率，稀疏性，减少了计算量。
- - 缺点：有死区 Dying ReLU。（leaky和noisy ReLU解决了这个问题，但是引入了额外的参数）
- 大的梯度帮助学习更快，但是任意大会导致不稳定。
- 网络结构应该通过实验和监视验证集误差来确定。
- 更深的模型减少了用于表示函数的神经元数量，也降低了泛化误差。
- - 直觉上说，更深的网络更好，因为我们在学习一系列函数。Intuitively, models with deeper layers are preferred because we are learning a series of functions on the overall function.

7 深度学习的正则项

- 最好在每层中使用不同的正则系数，但是使用相同的权重衰减。
- 使用早停early stopping。这是一个很好调的超参数，而且也减少了计算量。
- - 在整个数据集上使用新的参数再次重新训练模型，在到达先前模型的早停点时停止训练。
  - 保留早停点时的参数，继续在所有数据上训练，当平均训练误差降到原先早停点以下时停止。
- 模型平均（bagging, boosting等）基本上总会提高预测性能，虽然提高了计算量。
- - 模型平均一般都工作得很好，因为不同的模型不太可能犯相同的错误。
  - dropout通过创建子网络形成了一个高效的bagging方法。
  - - dropout在宽层网络工作更好，因为它减少了从输入到输出的路径。
    - 通常输入层dropout中的保留概率是0.8，隐层的概率是0.5。
    - 使用dropout的模型通常要更大，迭代更长。
    - 如果数据集足够大，dropout没有太大帮助。另外，在很小（<5000）0的训练样本上dropout的作用很有限。
    - 批量标准化同样也引入了噪声，这提供了正则的作用，但也可能让dropout没有太大必要。

8 训练深度模型的优化

- mini-batch的大小（batch size）：大的batch size会提供更大的梯度，但是通常内存是个限制条件。
- - 让你的batch size在内存允许范围内尽可能大。
  - 在GPU中，从32到256以2的指数级别增长，对于较大的模型可以从16开始。

    - 小的batch size因为噪声的原因可能有正则的作用，但是会导致整体运行时间增加。这些情况下需要更小的学习率来提升稳定性。
- 深度学习模型有多个局部最优，但这没太大关系，因为它们都有相同的代价。最主要的问题是局部最优的损失比全局最优的损失大得多。
-
    - 为了测试局部最优的问题，可以绘制梯度的范数来看它是否随时间衰减到一个非常小的值。
    - 在高维非凸函数中鞍点比局部最优更常见，梯度下降对于鞍点相对来说更加鲁棒。
- 梯度裁剪（clipping）用于解决梯度爆炸的情况。这在RNN中是个常见的问题。
- 绘制出目标函数随时间的变化曲线来选择学习率。
- 最优学习率通常高于前100次迭代后产生最佳性能的学习率。Optimal learning rate is higher than the learning rate that yields the best performance after the first ~100 iterations.监视前几次迭代，选择一个比表现最好的学习率更高的学习率，同时注意避免不稳定的情况。Monitor the first few iterations and go higher than the best performing learning rate while avoiding instability.
- 使用高斯和均匀分布来初始看起来没有太大影响。
-
    - 但是，对尺度（scale）有影响。大的初始权重会帮助避免冗余神经元，但是太大会有不利影响。
    - 权重初始化可以看作是超参数，尤其是初始尺度稀疏或密集的情况。specifically the initial scale and if they are sparse or dense。查看一个minibatch内激活或梯度的范围或方差来选择尺度。
- 没有一个优化算法明显由于其他，这主要取决于用户对超参数调整的熟悉情况。
-
    - 随机梯度下降（SGD），带动量的SGD，RMSProp，带动量的RMSProp，AdaDelta，Adam都是流行的选择。注意：RMSProp在训练初期或许会有很高的偏差。
    - 通常动量的值有0.5,0.9，0.99。这个超参数可以随时间变化，从一个小值开始增加到大的值。
    - Adam通常是较为鲁棒的选择。但是学习率通常要根据默认值改动。
- 对转化后的值（transformed value）而不是输入做批量归一化。在引入可学习的参数β下去掉偏置项。对于CNN在每个空间位置应用范围归一化（range normalization）。
- 太浅或太深的网络更难训练，但是他们有更好的泛化误差。
- 相对于一个强大的优化算法，选择容易优化的模型更重要。

9 卷积网络

- 池化对于控制不同大小的输入是非常有用的。
- 0填充可以让我们独立的设置卷积核大小和输出大小，而不会让维度衰减变成一个受限因素。
- 对于测试准确率而言最优的0填充通常在：
-
    - “valid卷积”，不使用0填充，卷积核在图像范围内，但是每层输出会衰减。
    - “same卷积”，使用足够的0填充让输出的大小等于输入大小。
- 一个潜在的验证卷积结构的方法是使用随机权重，并且只训练最后一层。One potential way to evaluate convolutional architectures is to use randomized weights and only train the last layer.

10 序列模型：循环和递归网络

- 双向RNN在手写识别，语音识别和生物信息方面非常成功。
- 相比较CNN而言，RNN用于图像通常更困难，但是可以让相同特征图中的特征进行远程横向交互。RNNs applied to images are typically more expensive but allow for long-range lateral interactions between features in the same feature map.
- 无论何时RNN要表示长期依赖，长期交互的梯度要指数级别小于短期交互。Whenever a RNN has to represent long-term dependencies, the long term interactions gradients have an exponentially smaller magnitude than the gradients of the short term interactions.
- The technologies used to set the weights in echo state networks could be used to initialize weights in a fully trainable RNN – an initial spectral radius of 1.2 and sparse initialization perform well.
- 实践中最有效的序列模型是门控RNN，包括LSTM和GRU。
-
    - LSTM比简单的RNN更容易学到长期依赖。
    - 给遗忘门加入偏置1可以让LSTM像GRN变种一样强大。

    - 在LSTM中使用SGD通常考虑二阶优化方法来防止二次偏导消失。Using SGD on LSTMs typically takes care of using second-order optimization methods to prevent second derivatives vanishing with the first derivatives.
- 设计一个容易优化的模型通常比设计一个强大的算法容易些。
- 正则参数鼓励“信息流”，并预防梯度消失，但是同样需要梯度裁剪来预防梯度爆炸。Regularization parameters encourage “information flow” and prevents vanishing gradients. However, gradient clipping is needed for RNNs to also prevent gradient explosion (which would prevent learning from succeeding).但是大量的数据例如语言建模对于LSTM而言就不是那么高效了。

11 实践方法

- 在不确定的时候让模型拒绝决策也是很有帮助的，但是这之间也存在折中。It can be useful to have a model refuse to make a decision if it is not confident, but there is a tradeoff.收敛是机器学习算法可以响应的样本部分，和准确率之间也存在折中。Coverage is the fraction of samples for which the machine learning sample is able to produce a response for, and it is a tradeoff with accuracy.
- 对于基线模型而言，ReLU及其变种是理想的激活函数。
- 带动量和学习率衰减的SGD是基线优化算法中一个不错的选择。衰减方法包括：
-
    - 线性衰减直到一个固定小的学习率。
    - 指数衰减。
    - 每次验证误差不变时减少2到10分之一。
- 另一个不错的基线优化算法是Adam。
- 如果优化出现了问题，立马使用批量归一化。
- 如果训练集不足10M，一开始就采用中等强度的正则项。
-
    - 基本上肯定要用early stopping。
    - 在大多数结构中dropout是个不错的选择。批量归一化也是个可选的替代品。
- 如果当前的问题已经被研究烂了，直接拷贝模型就是个不错主意，或许可以拷贝训练过的模型。
- 如果已知无监督学习对于你的应用很重要（例如NLP中的词嵌入），那么在基线中就把它包含进来。
- 决定什么时候收集更多的数据：
-
    - 如果训练性能很差，增加模型的大小，调整学习算法。如果还是很差，那是数据质量问题，重新开始收集更干净的数据或更多的特征。
    - 如果训练性能不错但是测试性能很差，在可行的情况下收集更多的数据。或者，尝试降低模型的大小或增加正则强度。如果这些没有帮助，那你的确需要更多的数据。
    -
        - 如果无法获得更多的数据，最后的办法是尝试提升学习算法。
        - 使用对数比例的学习曲线来决定还需要多少数据。
- 学习率是最重要的超参数，因为它以一种复杂的方式控制着模型的有效容量。如果时间有限，就调它。调其他的超参数需要监视训练和测试误差来判断模型是欠拟合还是过拟合。
-
    - 如果训练误差高于目标误差，增加容量。如果没有使用正则项，并确定优化算法工作正确，使用更多的隐层。
    - 如果测试误差高于目标误差。那么较大的模型加合适的正则会带来最好的性能。
- 只要训练误差很低，你总是可以通过收集更多数据来减低泛化误差。
- 网格搜索：通常在少于四个超参数时使用。
-
    - 通常最好使用对数尺度来挑选值，并且不断重复来减少搜索范围。
    - 计算量随着超参数的数量指数增加，即使是并行也不能有很好的帮助。
- 随即搜索：使用起来很简单，相比于网格搜索更快收敛到好的超参数。
-
    - 在几个超参数不是很强烈的影响性能指标时，随机搜索相对于网格搜索可以达到指数级别的效率。
    - 我们也许要重复运行它来得到更好的结果。
    - 随机搜索比网格搜索快，因为它不需要指数级别的运行。
- 通常不建议基于超参数来调整模型，因为它很少超过人类并且经常失败。
- 调试策略：

- 可视化模型的行为。例如，查看图像样本，和模型检测情况。这可以看到量化的性能指标是否合理。
  - 可视化最差的错误情况：这可以发现预处理和打标签中存在的问题。
  - 当训练误差太高时先拟合一个小的数据集：这能检查出是欠拟合问题还是软件缺陷。
  - 监视激活和梯度的直方图：完成大概一个epoch。这能告诉我们神经元是否饱和及其饱和频率。梯度（值？）应该是参数的1%左右。The gradient should be about 1% of the parameter.稀疏的数据（如NLP）有很多参数很少更新，一定要记得这一点。

12 应用

- 在分布式系统中，使用异步SGD。每一步的平均提升是很小的，但是步骤速率加快也导致了整体的加快。
- 级联（cascade）分类器是目标检测中一个高效的方法。一个分类器有高召回率，另一个有高精确率，好的，结果有了。
- 集成方法中一个减少推理时间的方法是训练一个控制器来挑选哪个网络应该来做推理。
- 标准化像素尺度是计算机视觉中唯一一个强制的预处理步骤。
- 对比度归一化通常是一个安全的计算机视觉预处理步骤。
- 全局对比度归一化（GCN）是其中一个方法，但是它在低对比度的情况下会降低边缘的检测。
  - 尺度参数或者可以设置为1，或者让每个像素在整体样本上有接近1的标准梯度。
    - 近似剪切的图像数据集可以安全的设置λ=0,ϵ=1e-8。
    - 小的随机剪切的图像数据集需要更高的正则强度，例如λ=10,ϵ=0。
  - 局部对比度归一化通常可以用分离卷积计算特征图的局部均值/方差来实现，然后在不同的特征图上做元素级别减/除。相比于全局对比度归一化更能凸显边缘。
- 在NLP的实践中，分层softmax相比于基于采样的方法测试结果更差。

3 深度学习研究

13 线性因子模型

- 线性因子模型可以扩展到自动编码器和深度概率模型，它们做相同的任务但是更灵活更强大。

14 自动编码器

- 稀疏自动编码器在学习特征如分类等任务上表现不错。
- 自动编码器在学习隐含变量解释输入方面很有用。它们可以学到有用的特征。
- 虽然很多自动编码器只有一个编码/解码层，但它们有深度前馈网络一样的优点。
- 在强化对比度例如稀疏性方面尤其有用。
  - 深度减少了指数级别的计算量，以及表示某些函数所需训练数据的数量。
  - 通常训练深度编码器的方法是通过一系列浅的自动编码器来贪心的预训练深度结构。

15 表示学习

- 在深度学习中，一个好的表示可以让后续的学习任务更加容易。例如监督前馈网络：每一层都为最后的分类层学习一个更好的表示。
- 贪心的层间无监督训练对于分类测试误差有帮助，但是其他任务上不行。
- 对于图像分类没啥作用，但是对于NLP很有帮助（例如词嵌入），这是因为初始表示非常差。
  - 在标签样本很少，或无标签样本很大的时候，正则器非常有用。
  - 在要学习的函数极其复杂的时候它最有用。
  - 根据监督阶段的验证集误差来选择预训练阶段的超参数。
  - 无监督预训练基本已经被抛弃了，除了NLP领域（例如词嵌入）。
- 在有些特征对于不同的任务设置有帮助的时候，迁移学习，多任务学习和域适应都可以通过表示学习来完成。Transfer learning, multitask learning, and domain adaptation can be achieved with representation learning when there are features useful for different tasks/settings corresponding to underlining factors appearing in multiple settings.

- 当一个很复杂的结构可以用更少的参数紧凑表示时，分布式表示相比于非分布式表示更具有统计优势。Distributed representations can have a statistical advantage to non-distributed representations when an already complicated structure can be compactly represented using a smaller number of parameters.一些传统的非分布式算法能泛化是由于其平滑的假设，但是会受限于维度诅咒。

16 深度学习的结构化概率模型

- 结构化概率模型提供了一个框架，用于对随机间隔的直接交互进行建模，这使得模型可以使用更少的参数。正因为此，他们可以在更少的数据下可靠地估计，并且减少了存储模型，执行推理和采样的计算量。
- 很多深度学习的生成模型或没有隐含变量，或这用一层隐含变量。他们在模型中使用深度计算图来定义条件分布。Many generative models in deep learning have either no latent variables or only use one layer of latent variables. These use deep computational graphs to define the conditional distributions within a model. 这和大部分深度学习应用中有比可观察变量更多的隐含变量形成强烈对比。他们是从非线性交互中学得的。This is in contrast to most deep learning applications where there tend to be more latent variables than observed variables. These are learned for nonlinear interactions.
- 深度学习中的隐含变量是不受限的，但是很难通过可视化来解释。Latent variables in deep learning are unconstrained but are difficult to interpret outside of rough characterization via visualization.
- 循环信念传播（loopy belief propagation）基本上在深度学习中从未使用，因为大部分深度学习模型是使用吉布斯采样或变分推断算法设计的。Loopy belief propagation is almost never used in deep learning because most deep learning models are designed to make Gibbs sampling or variational inference algorithms efficient.

17 Monte Carlo方法

- Monte Carlo Markov Chains（MCMC）计算量很大，这是由于在平衡分布"燃烧"需要的时间，以及为了保证样本间不相关而让每n个样本有序。Monte Carlo Markov Chains (MCMC) can be computationally expensive to use because of the time required to "burn in" the equilibrium distribution and to keep every n sample in order to assure your samples aren't correlated.
- 当在深度学习中从MCMC采样时，通常需要运行一定数量的并行马尔科夫链，数量与一个minibatch的样本数一样，让后从中采样。通常使用的数字是100、。
- 马尔科夫链会到达平衡状态，但我们不知道要多久，除非它已经到达了。We can test if it has mixed with heuristics like manually inspecting samples or measuring correlation between successive samples.
- While the Metropolis-Hastings algorithm is often used with Markov chains in other disciplines, Gibbs sampling is the de-facto method for deep learning.

19 近似推断

- Maximum a posteriori (MAP)推断通常用于特征提取和学习机制，主要是稀疏编码模型。

20 深度生成模型

- 玻尔兹曼机的变种早已超过了最初的流行度。Variants of the Boltzmann machine surpassed the popularity of the original long ago. 玻尔兹曼机对于观察变量就像一个线性预测器，但是对于未观察到的变量更加强大。Boltzmann machines act as a linear estimator for observed variables, but are typically more powerful for unobserved variables.
- 从一系列受限玻尔兹曼机中初始化一个深度玻尔兹曼机时，稍微修改下参数是很有必要的。
- 今天很少使用深度信念网络（DBN），因为其他算法已经超过它了，但是在历史上有重要地位。由于DBN是生成模型，一个训练过的DBN可以用于初始化一个MLP的权重来做分类。While deep belief networks are generative models, the weights from a trained DBN can be used to initialize the weights for a MLP for classification as an example of discriminative fine tuning.
- 【略去玻尔兹曼机的部分内容】
- 虽然变分自动编码器（VAE）很简单，但是它们通常能得到不错的结果，也是最后的生成模型之一。来自VAE的图像通常模糊，原因未知。

- 不收敛是GAN欠拟合（一个网络抵达局部最优，另一个抵达局部最大）的一个问题，但是这个问题的长度还不清楚。
- 虽然GAN有稳定性的问题，但是通常在精心选择的模型和超参数情况下效果很不错。
- GAN的一个变种LAPGAN，从低分辨率开始不断加入细节，它的结果经常能骗过人类。
- 为了保证GAN的生成器不会在任意点变成0概率，需要在最后一层给所有图像加入高斯噪声。
- 在GAN的判别器中一定要使用dropout，不这样做结果很差。
- Visual samples from generative moment matching networks are disappointing, but can be improved by combining them with autoencoders.
- 在生成图像时，使用转置卷积操作通常会产生更真实的图像，相比于没有参数共享的全连接层使用更少的参数。When generating images, using the “transpose” of the convolution operator often yields more realistic images while using fewer parameters than using fully connected layers without parameter sharing.
- 即使在卷积生成网络中上采样的假设不真实，但是生成的样本总体来说还是不错的。
- 虽然有很多使用生成模型的方法来生成样本，MCMC采样，ancestral sampling或把二者结合是比较流行的做法。
- 当比较生成模型时，预处理的改变（即使很小，很微妙）是完全不能接收的，因为它会改变分布从而根本上改变了任务。When comparing generative models, changes in preprocessing, even small and subtle ones, are completely unacceptable because it can change the distribution and fundamentally alters the task.
- 如果通过观察样本图片来衡量生成模型，最好在不知道样本源的情况下去做实验。另外，由于一个差的模型也可能产生好的样本，必须确保模型不是仅仅复制了训练图片。使用欧氏距离来进行检查。
- 如果计算上可行的化，最好的衡量生成模型样本的方法是评估模型分配给测试数据的对数似然。A better way to evaluate samples from a generative model is to evaluate the log-likelihood that the model assigns to the test data if it is computationally feasible to do so.这个方法也有缺陷，例如一个固定的简单图片（如空白背景）有很高的似然。
- 生成模型有很多用处，因此根据用途来挑选评估指标。
-
    - 例如，一些生成模型更擅长为最真实的点分配高概率，而其他一些模型给不真实的点不分配高概率做的更好。some generative models are better at assigning a high probability to the most realistic points, and others are better at rarely assigning high probabilities to unrealistic points.
    - 即使当指标缩小到最合适的任务上，所有的指标也都有很严重的问题。