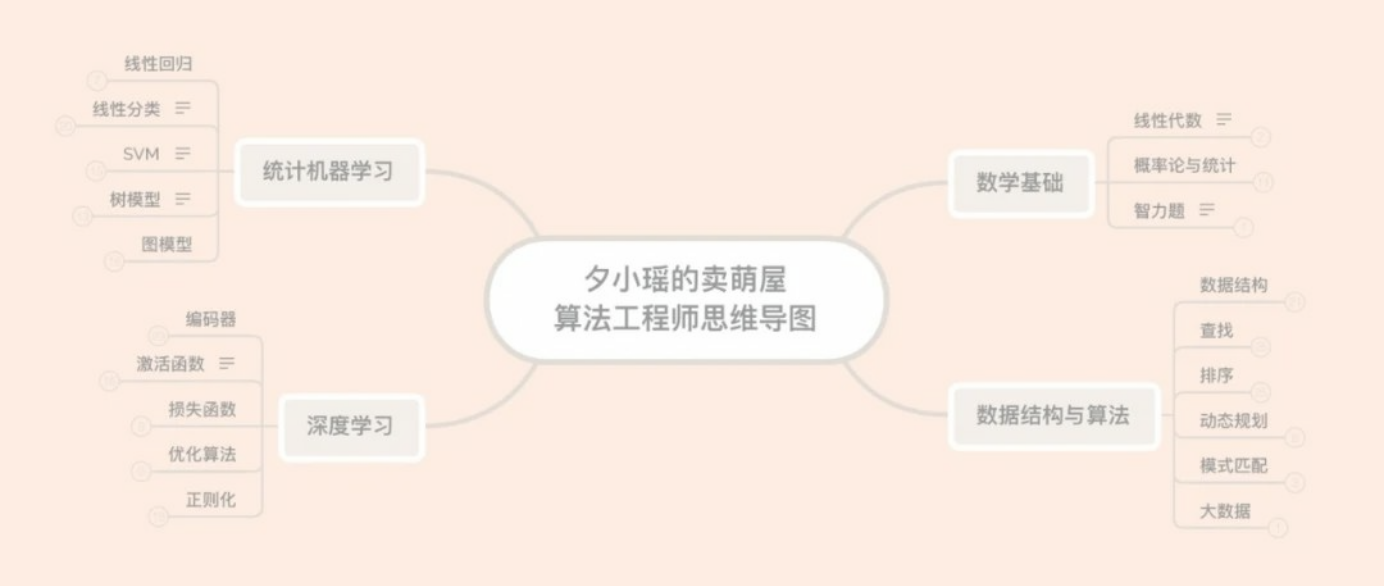
卖萌屋算法工程师思维导图part3—深度学习篇

原创 rumor酱 夕小瑶的卖萌屋 6月15日

来自专辑
卖萌屋@算法工程师思维导图



卖萌屋的妹子们（划掉）作者团整理的**算法工程师思维导图**，求职/自我提升/查漏补缺神器。该手册一共分为**数据结构与算法、数学基础、统计机器学习和深度学习**四个部分。

下面是第三部分深度学习的内容~

公众号后台回复【思维导图】获取完整手册(Xmind脑图源文件，学习起来更方便(ง •_•)ง

编码器

- DNN
  - 反向传播
  - 梯度消失与爆炸

    反向传播到梯度消失爆炸
    https://zhuanlan.zhihu.com/p/76772734

    原因:

    本质上是因为梯度反向传播中的连乘效应。其实梯度爆炸和梯度消失问题都是因为网络太深，网络权值更新不稳定造成的

    激活函数导数*权值<1，多个小于1的数连乘之后，那将会越来越小，导致靠近输入层的层的权重的偏导几乎为0，也就是说几乎不更新，这就是梯度消失的根本原因。连乘下来就会导致梯度过大，导致梯度更新幅度特别大，可能会溢出，导致模型无法收敛。

    解决方案:

梯度爆炸：正则化/截断 梯度消失：

1.改变激活函数：relu（tanh导数也小于1），但会出现dead relu

2.batchnorm：使权值w落在激活函数敏感的区域，梯度变化大，避免梯度消失，同时加快收敛

3.残差结构：求导时总有1在

- CNN
  - 归纳偏置：locality & spatial invariance
  - 1*1卷积核

    作用：1.升维降维(in_channel -> out_channel) 2.非线性 与全连接层的区别：输入尺寸是否可变，全连接层的输入尺寸是固定的，卷积层的输入尺寸是任意的
  - 反向传播

    通过平铺的方式转换成全联接层

    https://zhuanlan.zhihu.com/p/81675803

    avg pooling：相当于成了w = [1/4, 1/4, 1/4, 1/4]
  - 稀疏交互与权重共享

    每个输 出神经元仅与前一层特定局部区域内的冲经元存在连接权重 在卷积神经网络中，卷积核中的 每一个元素将作用于每一次局部输入的特定位置上 参数共享的物理意义是使得卷积层具高平移等变性。假如图像中有一 只猫，那么无论百出现在图像中的任何位置 3 我们都应该将 '8i只别为猫 在猫的 圆片上先进行卷积，再向右平移 I像素的输出，与先将圆片向右平移 J像 素再进行卷积操作的输出结果是相等的。
  - 池化本质：降采样

    平均池化：避免估计方差增大，对背景对保留效果好 最大池化：避免估计均值偏移，提取纹理信息

    油化操作除了能显著降低参数量外，还能够保持对平移、伸缩、旋 转操作的不变性。
- RNN

  https://zhuanlan.zhihu.com/p/34203833
  - 归纳偏置：sequentiality & time invariance
  - BPTT
  - 梯度消失与爆炸

    原因:

    https://zhuanlan.zhihu.com/p/76772734
    DNN中各个权重的梯度是独立的，该消失的就会消失，不会消失的就不会消失。RNN的特殊性在于，它的权重是共享的。

当距离长了，最前面的导数就会消失或爆炸，但当前时刻整体的梯度并不会消失，因为它是求和的过程。RNN 所谓梯度消失的真正含义是，梯度被近距离梯度主导，导致模型难以学到远距离的依赖关系。

解决方案: LSTM长时记忆单元

- LSTM

  消失：通过长时记忆单元，类似残差链接。但后来加了遗忘门，遗忘门介于0-1，梯度仍有可能消失 爆炸：梯度仍可能爆炸，但LSTM机制复杂，多了一层激活函数sigmoid，可以通过正则与裁剪解决https://zhuanlan.zhihu.com/p/30465140

  各模块可以使用其他激活函数吗?

  sigmoid符合门控的物理意义 tanh在-1到1之间，以0为中心，和大多数特征分布吻合，且在0处比sigmoid梯度大易收敛

  一开始没有遗忘门，也不是sigmoid，后来发现这样效果好

  relu的梯度是0/1，1的时候相当于同一个矩阵W连成，仍旧会梯度消失或爆炸的问题

  综上所述，当采用 ReLU 作为循环神经网络中隐含层的激活函数时，只有当 W的取值在单位矩阵附近时才能取得比较好的效果，因此需要将 W初始化为单位矩阵。实验证明，初始化 W为单位矩阵并使用 ReLU 激活函数在一些应用中取得了与长短期记忆模型相似的结果.

- GRU**要点**：结构、与LSTM的异同

- Transformer

  - 结构

  - QK非对称变换

    双线性点积模型，引入非对称性，更具健壮性（Attention mask对角元素值不一定是最大的，也就是说当前位置对自身的注意力得分不一定最高）。

  - Scaled Dot Product

    为什么是缩放点积，而不是点积模型？当输入信息的维度 d 比较高，点积模型的值通常有比较大方差，从而导致 softmax 函数的梯度会比较小。因此，缩放点积模型可以较好地解决这一问题。

    相较于加性模型，点积模型具备哪些优点？常用的Attention机制为加性模型和点积模型，理论上加性模型和点积模型的复杂度差不多，但是点积模型在实现上可以更好地利用矩阵乘积，从而计算效率更高（实际上，随着维度d的增大，加性模型会明显好于点积模型）。

  - Multi-head

    https://zhuanlan.zhihu.com/p/76912493

    多头机制为什么有效?

    1.类似于CNN中通过多通道机制进行特征选择；

    2.Transformer中先通过切头（spilt）再分别进行Scaled Dot-Product Attention，可以使进行点积计算的维度d不大（防止梯度消失），同时缩小attention mask矩阵。

- FFN

  Transformer在抛弃了 LSTM 结构后，FFN 中的 ReLU成为了一个主要的提供非线性变换的单元。

激活函数

https://zhuanlan.zhihu.com/p/73214810

- tanh

  相比Sigmoid函数， tanh的输出范围时(-1, 1)，解决了Sigmoid函数的不是zero-centered输出问题；幂运算的问题仍然存在；tanh导数范围在(0, 1)之间，相比sigmoid的(0, 0.25)，梯度消失（gradient vanishing）问题会得到缓解，但仍然还会存在。

  要点: Xavier初始化、公式、导数

- relu

  相比Sigmoid和tanh，ReLU摒弃了复杂的计算，提高了运算速度。解决了梯度消失问题，收敛速度快于Sigmoid和tanh函数

  缺点：爆炸梯度(通过梯度裁剪来解决) 如果学习率过大，会出现dead relu的不可逆情况 — 激活为0时不进行学习(通过加参数的ReLu解决) 激活值的均值和方差不是0和1。(通过从激活中减去约0.5来部分解决这个问题。在fastai的视频力有个更好的解释)

  Leaky relu：增加了参数**要点**：He初始化、公式、导数

- gelu

  https://zhuanlan.zhihu.com/p/100175788

  https://blog.csdn.net/liruihongbob/article/details/86510622

  ReLu：缺乏随机因素，只用0和1

  https://www.cnblogs.com/shiyublog/p/11121839.html

  GeLu：在激活中引入了随机正则的思想，根据当前input大于其余inputs的概率进行随机正则化，即为在mask时依赖输入的数据分布，即x越小越有可能被mask掉，因此服从bernoulli(Φ(x))

  高斯误差线性单元:

  对于每一个输入 x，其服从于标准正态分布 N(0, 1)，它会乘上一个伯努利分布 Bernoulli(Φ(x))，其中Φ(x) = P(X ≤ x)。这么选择是因为神经元的输入趋向于正太分布，这么设定使得当输入x减小的时候，输入会有一个更高的概率被dropout掉。

  Gelu(x) = xΦ(x) = xP(X ≤ x)

- sigmoid

  激活函数计算量大（在正向传播和反向传播中都包含幂运算和除法）；

  反向传播求误差梯度时，求导涉及除法；

  Sigmoid的输出不是0均值（即zero-centered）；这会导致后一层的神经元将得到上一层输出的非0均值的信号作为输入，随着

网络的加深，会改变数据的原始分布

优点：

激活函数计算量大（在正向传播和反向传播中都包含幂运算和除法）；反向传播求误差梯度时，求导涉及除法；Sigmoid的输出不是0均值（即zero-centered）；这会导致后一层的神经元将得到上一层输出的非0均值的信号作为输入，随着网络的加深，会改变数据的原始分布

- softmax

  sigmoid是softmax的特例：

  https://blog.csdn.net/weixin_37136725/article/details/53884173

损失函数

- 分类
  - 0-1 loss
  - hinge loss
  - sigmoid loss
  - cross entropy

    求导：
    https://zhuanlan.zhihu.com/p/60042105
- 回归
  - square loss

    对异常点敏感
  - absolute loss

    对异常点鲁棒，但是y=f时不可导
  - Huber loss

优化算法

- 求解析解：凸函数
- 迭代法
  - 一阶法：梯度下降

    https://zhuanlan.zhihu.com/p/111932438

SGD：

数据要shuffle 一开始重j去采用较大的学习速率 ，当误差曲线进入平台期后，;成小学习速菜做更精细的调整。最优的学习速 率方案也通常需要调参才能得到。

随机梯度下降法无法收敛 1.batch size太小，震荡 2.峡谷和鞍点

Adam：

指数加权：

1.不用像mean一样统计个数重新计算avg

2.历史久远的权重会呈指数衰减

动量=惯性保持：累加了之前步的速度

1.增强了整体方向的速度，加快收敛

2.消减了错误方向的速度，减少震荡

AdaGrad=环境感知：根据不同参数的一些经验性判断，自适应地确定参数的学习速率，不同参数的重新步幅是不同的。

1.更新频率低的参数可以有较大幅度的更新，更新频率高的步幅可以减小。AdaGrad方法采用 “历史梯度平方和”来衡量不同参数的梯度的稀疏性 3 取值越小表明越稀疏

参数中每个维度的更新速率都不一样！！！

2.随着时间的推移，学习率越来越小，保证了结果的最终收敛

缺点：即使Adam有自适应学习率，也需要调整整体学习率（warmup）

AdamW是Adam在权重上使用了L2正则化，这样小的权重泛化性能更好。

- 二阶法：牛顿法

  在高维情况下， Hessian ~E 阵求逆的计算复杂度很大 3 而且当目标函数非口时，二阶法有可能会收 敛到鞍点(Saddle Point) 。

  鞍点：一个不是局部最小值的驻点（一阶导数为0的点）称为鞍点。数学含义是：目标函数在此点上的梯度（一阶导数）值为 0， 但从改点出发的一个方向是函数的极大值点，而在另一个方向是函数的极小值点。

正则化

- 修改数据
  - **增加数据**
  - label smoothing
- 修改结构：Normalisation
  - Batchnorm：

为什么对NN层中归一化?

随着网络训练的进行 ， 每个隐层的参数变化使得后一层的输入 发生变化 3 从而每-批训练数据的分布也随之改变 3 致使网络在每次迭 代中都需要拟合不罔的数据分布，增大训练的复杂度以及过拟合的风险。

为什么增加新的分布?

以Sigmoid函数为例，批量归一化 之后数据整体处于函数的非饱和区域，只包含线性变躁，破坏了之前学 习到的特征分布。

在CNN的应用：

在全连接网络中是对每个神经元进行归一化，也就是每个神经元都会学习一个γ和β。批量归一化在卷积神经网络中应用时，需要注意卷积神经网络的参数共享机制 。每一个卷积核的参数在不同位置的楠经元当中是共享 的， 因此也应该被一起归一化。在卷积中，每层由多少个卷积核，就学习几个γ和β

预测：

在预测时无法计算均值和方差，通常需要在训练时根据mini-batch和指数加权平均计算，直接用于预测

- Layernorm

  对比BatchNorm：

  1.对于RNN来说，sequence的长度是不一致的，换句话说RNN的深度不是固定的，不同的time-step需要保存不同的statics特征，可能存在一个特殊sequence比其他sequence长很多，这样training时，计算很麻烦。

  2.不依赖batch size：在hidden size的维度进行layernorm，跟batch和seq_len无关。beta和gamma的维度都是(hidden_size,)，每个神经元有自己的均值和方差，因为不同单元是不同的feature，量纲不一样。normalisaion通常在非线性函数之前，LN在BERT中主要起到白化的作用，增强模型稳定性（如果删除则无法收敛）。

- 修改结构：Dropout

  本质上是模型集成。

  实现：1.训练时不动，预测时乘p 2.反向传播传播时除p，预测不动。

- 修改结构：weight decay

  在更新w时减去一个常数，跟L2求导之后的公式一致
  https://bbabenko.github.io/weight-decay/

- Weight decay和L2正则在SGD情况下等价，Adam下不等：https://zhuanlan.zhihu.com/p/40814046

  权重越大惩罚应该越大，但adam的adagrad调整使得惩罚变小

- 修改结构：正则项

  - L1

    稀疏解的好处：1.特征选择，减少计算 2.避免过拟合，增强鲁棒性

解空间的解释：加上了菱形约束，容易在尖角处碰撞出解

贝叶斯角度解释：加了laplace分布，在0点的概率要更高

- L2

解空间角度：加了球形约束，等高线切在圆上**贝叶斯角度**：加了高斯分布，在0点附近的概率更大且相近

- 训练技巧
  - early stopping
  - warmup

    刚开始小一些，防止对前几个batch的过拟合，之后见过了不少数据，可以慢慢升高。之后参数基本上稳定了，就小学习率精细调整。

公众号后台回复【思维导图】获取完整手册(Xmind脑图源文件，学习起来更方便(ง •_•)ง