以DSSM为例说明深度学习模型训练中的若干问题

点击关注 搜索与推荐Wiki 2020-11-27

▼ 相关推荐 ▼

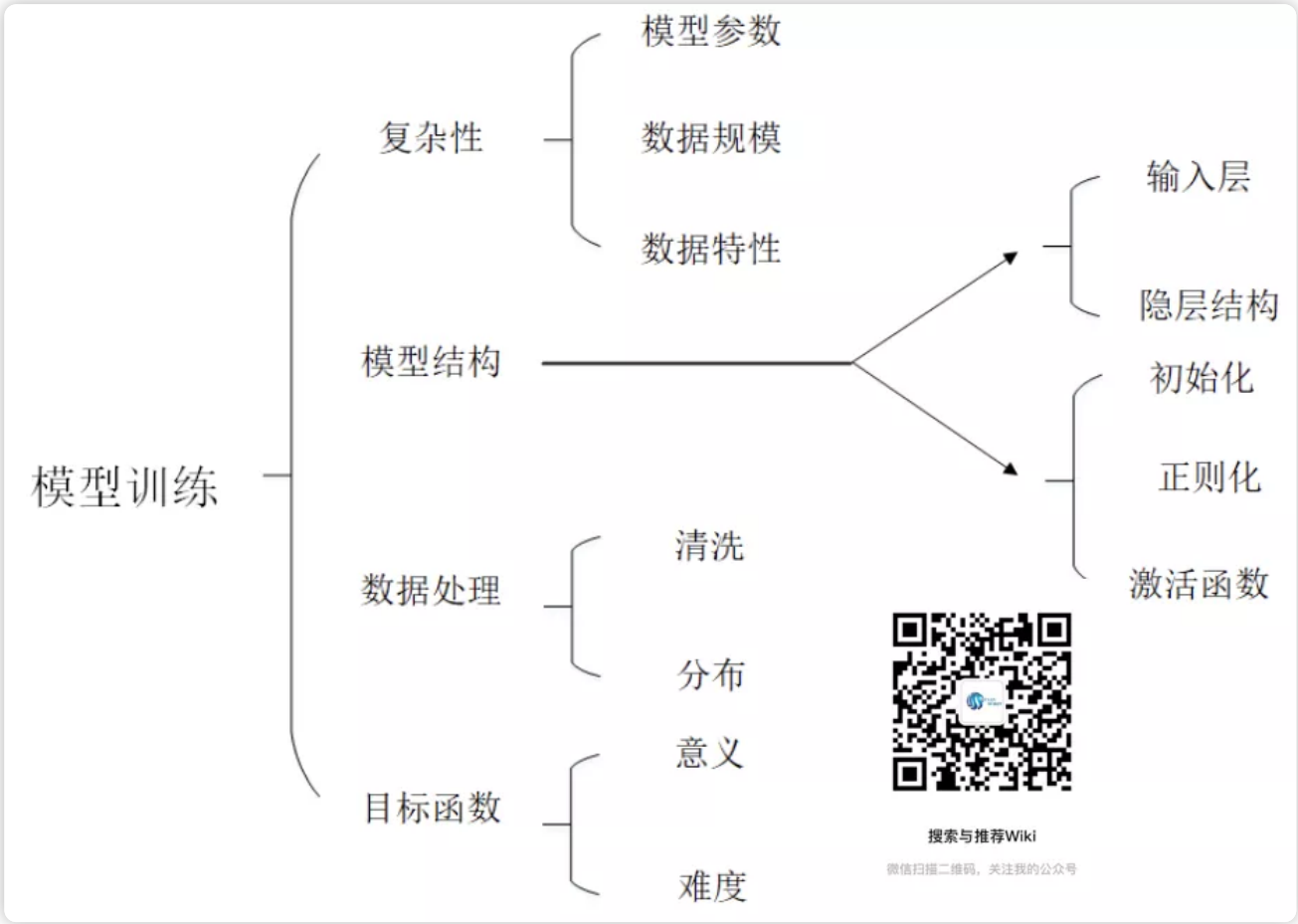
来自于：机器学习AI算法工程（原文出处未找到）

编辑：**Thinkgamer**

说明：点击文末「阅读原文」触达更多精彩

本文主要用于记录DSSM模型学习期间遇到的问题及分析、处理经验。先统领性地提出深度学习模型训练过程的一般思路和要点，再结合具体实例进行说明。全文问题和解决方案尽可能具备通用性，同样适用于一般深度学习模型的训练。

深度学习模型训练要素概图



补充：目标函数一般包含经验风险(损失函数或代价函数)和结构风险(正则化项)，此处仅指损失函数。

训练深度学习模型，主要需要考虑四个方面(受限于当前认知水平，仅总结了四个方面)，分别是：

- 数据处理，包含数据清洗和分布；
- 模型结构，包括网络层结构设计和一些细节处理，前者主要有输入层设计和隐层设计(输出层设计划分至目标函数)，后者主要有初始化、正则化和激活函数；
- 目标函数设计，包含目标的意义和难度，前者决定了模型的学习方向，后者影响对模型能否收敛影响很大；
- 模型的复杂性，主要包括模型结构复杂性(量化表现是参数数量)和数据复杂性(数据规模与数据本身的特性)。

问题与处理

负样本采集方式过简

最初为了迅速跑通模型，对DSSM-LSTM做了简单的复现，此时的负样本并未采用随机负采样，而是统一选取了负样本空间的前n个(此部分工作已有人完成，我随后接手)。

实际使用模型时，负样本数量远多于正样本，而模型训练时只使用了固定的几类负样本，间接造成正样本多于负样本，显然是不合理的。为了使模型尽可能多地学到负样本特征，采用随机负采样为正样本配平负样，初期正负样本1:4。

由此引发了学习过程中最大的问题——模型无法收敛。

模型不能收敛

使用随机负采样将负样本变得丰富，本是正常操作，却由此导致模型不能收敛(loss大多只在前三个epoch有明显下降，最终loss与最初相比下降幅度不足1/4)，实在是不应该，这只能说明模型设计本身存在问题。

模型无法收敛，排除梯度问题以外，通常是问题或目标的复杂性超过了模型的学习能力，数据杂乱、数据复杂、模型结构复杂、损失函数“太难”等。

最初并没有这些经验，先是调整了batch_size和学习率，这仅仅改变了loss的绝对大小，并未改变loss居高不下的问题。随后更改了网络层神经元数量、梯度优化器，也尝试加入激活函数tanh，几乎没有效果。

在此过程中注意到另一个问题——batch_loss变化幅度大，即便在最初三个能下降的epoch中，batch_loss震荡也很厉害。

loss震荡幅度大

正常情况下，每个epoch中batch_loss是逐渐减小的，若loss较大且反复震荡，则会导致模型无法收敛，若loss很小，震荡则是趋于收敛的表现。

batch_loss较大，并且震荡，说明数据分布不均匀，经过检查发现数据是和标准问题对应的，比如前50个问题对应问题A，51-110问题对应问题B，其分布具有特定性而非随机性。

因此，每个batch包含的数据差别较大，以batch论，这些batch已经“不算一个数据集”了。解决方法就是随机打乱数据，使其分布没有“特点”，batch之间越接近，数据分布越好。

调整数据分布后，batch_loss相对稳定，loss有了进一步下降，与最初loss相比，最终loss约下降1/3(这是远远不够的，loss下降90%才可初步体现模型效果，至少下降95%才能有较好表现)。

续模型不能收敛

当数据和模型结构无法影响模型收敛性之后，只好试图修改目标函数。修改前，计算loss之前使用softmax函数对输出做了归一化，模型的学习目标由query与正样本的相似度接近1变成了对应的softmax输出接近1。

为了对softmax的输出有直观的认识，模拟了几组数据：

```
a=[[0.1,0.05,0.15,0.1,0.6] , softmax(a)=[0.17695288 0.1683228  0.18602546 0.17695288 0.291746 ]
b=[0.02,0.01,0.02,0.05,0.9] , softmax(b)=[0.15548703,0.15393992,0.15548703,0.16022234,0.37486365] ,
c=[0.01,0.015,0.015,0.03,0.03,0.9],
softmax(c)=[0.13359058 0.1342602  0.1342602  0.13628928 0.13628928 0.32531038]
d=[0.05,0.05,0.9] , softmax(d)=[0.23043351 0.23043351 0.539133 ]
```

从 $softmax(a)$ 和 $softmax(b)$ 可以看出原本巨大的输入差异，在输出层被缩小了，在b中0.9远大于0.01，对应的输出分别为0.37和0.15，差异没有那么大，在a中，0.6也远大于0.05，对应的输出分别为0.29和0.19，差异也没有那么大。

d与b、c相比可以看出最后一个维在整体数据中占比都是90%，但是随着维度的增加，其输出在逐渐下降。

这反映了softmax的两个特性：

- 其一，缩小原本数据之间的大小差异；
- 其二，随着维度的增加优势输入(在整体数据中占比较大)的输出会削弱，即输出逐渐下降。

由数据b、c和d可以看出，最后一维这种占比90%的绝对优势维度，其输出也不会达到0.9，且随着维度的增加其值越来越小。因此以某一维度的softmax输出逼近1为学习目标，几乎不可能实现，即损失函数的学习目标太难。

由此，以0.4作为softmax输出的学习目标，间接达到softmax的输入值大于0.9，即query与正样本的相似度大于0.9。更改损失函数后，模型loss迅速下降，终于可以正常训练。

模型差异较大

模型调试阶段，一直以A语料为训练数据，以Top10的语义召回率R为评价指标，随着参数调优，R从0.6逐渐上升，一度达到0.91，由此确定了模型的最佳参数。使用最佳参数配置训练了B语料的模型，R只达到了0.76，同样的配置使用C语料训练模型，R只有0.61。处理同样的任务，

A、B、C语料来自于同样的场景，在模型结果上差距较大，这基本不是模型的问题，更多的可能是数据的问题。在这种假设下，对三种语料的特点做了对比分析。

Data	Data_size	Ques_types	Quiz<=3	R
C	10005	1035	0.76	0.61
B	56014	983	0.13	0.76
A	54844	396	0.08	0.91

注)：data_size数据集大小，ques_types多分类总类别，quiz<=3，数据量不超过3的类别比例。

从上表中可以看出一条基本规律：数据规模越小，数据类别越多的语料训练出来的模型效果越差。数据规模小说明数据不充分，这对于深度学习模型训练来说确实不利，数据类别多说明数据特性复杂，会增大模型训练的难度。

此外，在C语料中76%类别的问题对应的样本不超过3条， 在B语料中13%类别的问题对应的样本不超过3条，在A语料中仅有8%类别的问题对应的样本不超过3条 ，这表明C语料不仅在整体数据上不充分，在单个类别上更加缺少数据。B语料类别虽然与C接近，但其数据规模相对充分，因此模型训练效果比C的好；同时，B语料规模与A语料接近，但其类别远多于A，因此其模型训练效果不如A。

总之，对于多文本分类问题，语料规模越大，单个类别样本越充足，其模型训练效果越好。

语料模型的微调

上文已分析了机票模型表现差的原因，即数据不充分、特性复杂，但是这并不意味着完全丧失了进一步优化的可能性。

数据就是这个情况，难以改变，目标函数也已被证实有效，无需大的变动，剩下的唯有调整模型结构了和一些超参数了。考虑到数据规模小，相应的应该减少模型参数(模型结构调整)，于是从输入层和隐层两个角度对其神经元数量做了削减。

结果表明，输入层神经元减少不仅无益于模型性能提升，反而下降了。这主要是因为，输入层负责将文本转为语义向量对其进行语义表征，而维度降低也意味着表征能力下降，所以不利于模型学习。

而对隐层神经元数量的减小则进一步加快了模型的收敛，并且使模型性能有了一定提升，最终将C语料训练的模型的语义召回率从0.61提升至0.7。此后，再怎么调整模型语义召回率也难以超越0.7。

所以，数据不好是深度学习模型训练的硬伤，虽然可以在算法设计层面进行一定优化，但这种优化是有限的，治标不治本，要想从根本上解决问题，仍需提升数据质量。