超参数总结

损失函数选择

1. 回归任务通常选择`MSE`、`MEAN_ABSOLUTE_ERROR`等。
2. 分类任务通常选择`MCXENT`、`NEGATIVELOGLIKELIHOOD`。

权重初始化

Xavier权重初始化方法通常是比较好的选择。对于使用修正线性（relu）或带泄露的修正线性（leaky relu）激活函数的网络而言，RELU权重初始化方法比较合适。

学习率（learning rate）

学习速率是最重要的超参数之一。如果学习速率过高或过低，网络可能学习效果非常差、学习速度非常慢，甚至完全没有进展。
学习率的选取主要观察损失函数得分，我们希望损失函数的得分虽然有跌宕但是**总体看是下降的**（因为可能对不同batch数据的拟合效果不同，使得损失函数的分数有跌宕）。
dl4j所给的建议是开始可以尝试三种不同的学习速率：1e-1、1e-3、1e-6。当损失函数得分一直无明显改善的时候，尝试增加学习率，因为说明学习率过低导致模型参数更新变化太小，此时应该增大学习率，加速模型参数的更新速率，使得模型快速学习，但是也有可能因为学习率过大会跳过局部极小值。而在理想状态下，可以同时以不同的学习速率运行模型，以便节省时间。

学习速率计划（learning rate schedule）

为神经网络设定学习速率策略，让学习速率随着时间推移逐渐“放缓”，帮助网络收敛至更接近局部极小值的位置，进而取得更好的学习效果。
dl4j中提供`learningRateDecayPolicy`超参数进行学习速率策略的选择，例如：

- Score 当损失函数的得分不再改善的时候，应用衰减率（可以使用`lrPolicyDecayRate`进行配置，衰减方式为`lr = lr * lrPolicyDecayRate`）
- Schedule 可以在网络配置的时候给定参数，当迭代到指定次数的时候使用新的学习速率（可以使用`learningRateSchedule`参数进行配置，这个接受的参数类型为`Map<Integer, Double>`）。

```
//例如这里是在神经网络迭代第100次的时候，学习率更改为0.01
map.put(100, 0.01);
```

还提供了其他的学习速率策略。

注： 最近实验使用学习速率策略，让模型以不同的学习速率运行模型的时候，拟合效果并不好，尤其是使用Score策略时候。

激活函数

隐藏（非输出）层的激活函数。“relu”或“leakyrelu”激活函数一般是比较好的选择。其他一些激活函数（tanh、sigmoid等）更容易出现梯度消失问题，进而大幅增加深度神经网络学习的难度。
但是，LSTM层仍然普遍使用tanh激活函数。对于LSTM，可使用softsign（而非softmax）激活函数替代tanh（更快且更不容易出现饱和（约0梯度））。

Epoch数量和迭代次数

一个epoch周期的定义是完整地遍历数据集一次。DL4J将迭代次数定义为每个微批次中的参数更新次数。
在训练中，一般应让训练持续多个epoch，而将迭代次数设为一次（.iterations(1)选项）；一般仅在对非常小的数据集进行完整批次的训练时才会采用大于1的迭代次数。
如果epoch数量太少，网络就没有足够的时间学会合适的参数；epoch数量太多则有可能导致网络对训练数据过拟合。

更新器和优化算法

在DL4J中，“更新器”一词指动量、RMSProp、adagrad等训练机制。相比于“普通”的随机梯度下降，使用这些方法可以大大提高网络训练速度。可以用.updater(Updater)配置选项设置更新器。
多数情况下比较好的默认选择是使用随机梯度下降优化算法，与动量/rmsprop/adagrad更新器中的一种相结合，实践中常常使用动量更新器。注意动量更新器称为NESTEROVS（指Nesterov提出的动量），可以用.momentum(double)选项来设置动量。

梯度标准化（Gradient Normalization）

梯度标准化有时可以帮助避免梯度在神经网络训练过程中变得过大（即所谓的梯度膨胀问题，在循环神经网络中较常见）或过小。
RNN网络的常见问题就是梯度消失，在增加门控单元衍生到LSTM之后，梯度消失问题有所改善，而梯度爆炸的问题并没有得到改善。因此我们面对梯度爆炸的情况主要还是采用梯度标准化。尤其在我们使用多层LSTM对数据进行训练的时候，梯度标准化对模型的拟合效果提升有较多的影响。

微批次大小（miniBatch）

微批次大小指计算梯度和参数更新值时一次使用的样例数量。通常可以从32开始尝试，常见的微批次大小范围在16～128之间（有时也可能更大或更小，取决于具体的应用和网络类型）。
而我们的数据集比较少，为此每次选择的微批次数值都比较小，相对合适的微批次大小主要还是靠模型的拟合效果试出来的。

网络层数和隐藏单元个数

网络层数并不是越多越好，如果设计的浅层（3到5层）网络没有学习任何特征，那么设计的超深（如100层）网络也会没有效果，甚至更加糟糕。
隐藏单元太多或者太少，都会导致网络难以训练。隐藏单元太少，可能会没有能力表达所需的任务；太多单元又会导致网络缓慢、难以训练，残留噪声难以消除。
但是网络层数和隐藏单元个数共同决定了网络模型的参数个数，从理论上讲，模型的的参数越多，模型的拟合能力也就越强。但是也容易出现过拟合。`参数 > 样例数 = 问题`，例如不要尝试用10,000个样例来学习一百万个参数。

正则化

正则化方法有助于避免在训练时发生过拟合。过拟合指网络对训练数据集的预测效果非常好，但对于从未见过的数据无法给出很好的预测。可以把过拟合理解成网络在“死记”训练数据（而非学习其中存在的一般关联）。
常见的正则化方法包括：

- L1和L2正则化会惩罚较大的网络权重，避免权重变得过大。实践中普遍采用一定程度的L2正则化。但要注意的是，如果L1和L2正则化系数过高，就有可能过度惩罚网络，导致网络停止学习。L2正则化的常用值为1e-3到1e-6。
- 丢弃法（dropout）是一种常用且很有效的正则化方法。丢弃法通常采用0.5的丢弃率。

额外链接

1. 含有LSTM超参数调试。 https://deeplearning4j.org/cn/lstm#tune
2. 我搭的神经网络不work该怎么办！看看这11条新手最容易犯的错误 https://mp.weixin.qq.com/s?__biz=MzIxMjAzNDY5Mg==&mid=2650791830&idx=1&sn=da81a253d4753e78d0ad5040ecf3ca29&chksm=8f474a7db830c3

更多文档可以查看 https://github.com/sjsdfg/deeplearning4j-issues。
欢迎star