# 神经网络中的超参数主要包括：

## 学习率 η；

学习率决定了权值更新的速度，在迭代更新权值的过程中，设置过大容易使训练的模型跨过最优值，导致过拟合；设置过小会使梯度下降过程过慢。这个参数是根据经验和不断实验来设置。

## 正则化参数 λ；

为了避免过拟合，必须对目标函数cost function（损失函数一般也叫价值函数）加入一些正则项。

## 神经网络的层数 L；

一般认为，增加隐层数可以降低网络误差（也有文献认为不一定能有效降低），提高精度，但也使网络复杂化，从而增加了网络的训练时间和出现“过拟合”的倾向。一般来讲应设计[神经网络](http://www.dataguru.cn/article-8976-1.html?union_site=innerlink)应优先考虑3层网络（即有1个隐层）。一般地，靠增加隐层节点数来获得较低的误差，其训练效果要比增加隐层数更容易实现。对于没有隐层的神经网络模型，实际上就是一个线性或非线性（取决于输出层采用线性或非线性转换函数型式）回归模型。

## 每一个隐层中神经元节点的个数 j；

隐层节点数的选择非常重要，它不仅对建立的神经网络模型的性能影响很大，而且是训练时出现“过拟合”的直接原因，但是目前理论上还没有一种科学的和普遍的确定方法。目前多数文献中提出的确定隐层节点数的计算公式都是针对训练样本任意多的情况，而且多数是针对最不利的情况，一般工程实践中很难满足，不宜采用。事实上，各种计算公式得到的隐层节点数有时相差几倍甚至上百倍。为尽可能避免训练时出现“过拟合”现象，保证足够高的网络性能和泛化能力，确定隐层节点数的最基本原则是：在满足精度要求的前提下取尽可能紧凑的结构，即取尽可能少的隐层节点数。研究表明，隐层节点数不仅与输入/输出层的节点数有关，更与需解决的问题的复杂程度和转换函数的型式以及样本数据的特性等因素有关。

在确定隐层节点数时必须满足下列条件：

（1）隐层节点数必须小于N-1（其中N为训练样本数），否则，网络模型的系统误差与训练样本的特性无关而趋于零，即建立的网络模型没有泛化能力，也没有任何实用价值。同理可知：输入层的节点数（变量数）必须小于N-1。

（2）训练样本数必须多于网络模型的连接权数，一般为2-10倍，否则，样本必须分成几部分并采用“轮流训练”的方法才可能得到可靠的神经网络模型。总之，若隐层节点数太少，网络可能根本不能训练或网络性能很差；若隐层节点数太多，虽然可使网络的系统误差减小，但一方面使网络训练时间延长，另一方面，训练容易陷入局部极小点而得不到最优点，也是训练时出现“过拟合”的内在原因。因此，合理隐层节点数应在综合考虑网络结构复杂程度和误差大小的情况下用节点删除法和扩张法确定。

## 学习的回合数、迭代次数：epoch，iteration；

1个epoch等于使用训练集中的全部样本训练一次；：1个iteration等于使用batchsize个样本训练一次。

## 小批量数据 minibatch的大小；

批大小。在[深度学习](https://www.baidu.com/s?wd=%E6%B7%B1%E5%BA%A6%E5%AD%A6%E4%B9%A0&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1Y3nvf4nHubrjI-njDkuHNB0ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EnW0krHRkn1c3P1R4PWbkPWfd)中，一般采用SGD训练，即每次训练在训练集中取batchsize个样本训练。

## 输出神经元的编码方式；

序号编码（Ordinal encoding）、独热编码（One-hot encoding）、二进制编码（Binary encoding）等。

## 代价函数的选择；

* 二次代价函数(quadratic cost)：

其中，表示代价函数，表示样本，表示实际值，表示输出值，表示样本的总数。

* 交叉熵代价函数(cross-entropy)：

其中，表示代价函数，表示样本，表示实际值，表示输出值，表示样本的总数。

* 对数释然代价函数(log-likelihood cost)：

对数释然函数常用来作为softmax回归的代价函数，然后输出层神经元是sigmoid函数，可以采用交叉熵代价函数。而深度学习中更普遍的做法是将softmax作为最后一层，此时常用的代价函数是对数释然代价函数。对数似然代价函数与softmax的组合和交叉熵与sigmoid函数的组合非常相似。对数释然代价函数在二分类时可以化简为交叉熵代价函数的形式。

## 权重初始化的方法；

* 把w初始化为0：

在线性回归，logistics回归的时候，基本上都是把参数初始化为0。在神经网络中，把w初始化为0是不可以的。这是因为如果把w初始化0，那么每一层的神经元学到的东西都是一样的（输出是一样的），而且在bp的时候，每一层内的神经元也是相同的，因为他们的gradient相同。

* **对w随机初始化：**

目前常用的就是随机初始化，即W随机初始化。

* **Xavier initialization：**

Xavier initialization是 Glorot 等人为了解决随机初始化的问题提出来的另一种初始化方法，他们的思想倒也简单，就是尽可能的让输入和输出服从相同的分布，这样就能够避免后面层的激活函数的输出值趋向于0。

* **He initialization：**

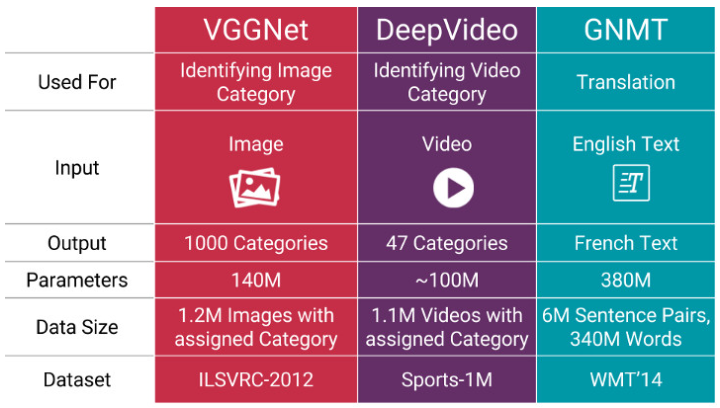
何恺明提出了一种针对ReLU的初始化方法，一般称作 He initialization。

## 神经元激活函数的种类；

* **sigmoid函数：**
* **Tanh函数：**
* **ReLU**

## 参加训练模型数据的规模；

以下是几个例子展示了最近一些模型所需要的参数数量：



* 模型大小 ∝ 数据大小 ∝ 问题复杂度

所需的数据量和模型的大小之间有一个有趣的近乎线性的关系。 基本的推理是，你的模型应该足够大，以便捕捉数据中的关系（例如图像中的纹理和形状，文本中的语法和语音中的音素）以及问题的具体细节（例如类别数量）。模型早期的层捕捉输入的不同部分之间的高级关系（如边缘和模式）。后面的层捕捉有助于做出最终决策的信息，通常能够帮助在想要的输出间进行区分。因此，如果问题的复杂性很高（如图像分类），参数数量和所需数据体量也非常大。

* 迁移学习

可以减少大概个数量级。

这些都是可以影响神经网络学习速度和最后分类结果，其中神经网络的学习速度主要根据训练集上代价函数下降的快慢有关，而最后的分类的结果主要跟在验证集上的分类正确率有关。因此可以根据该参数主要影响代价函数还是影响分类正确率进行分类，如图1所示：

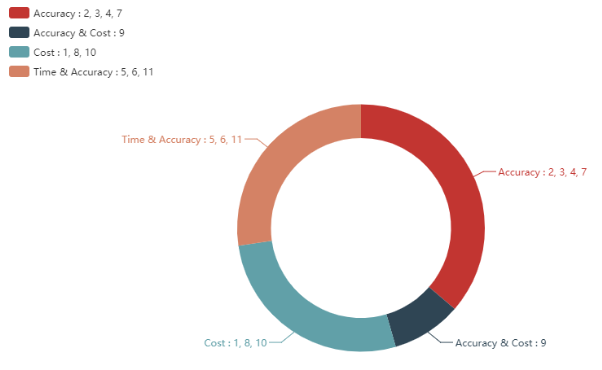


图1 超参数影响性对比图

在上图中可以看到超参数2，3，4，7主要影响的时神经网络的分类正确率；

9 主要影响代价函数曲线下降速度，同时有时也会影响正确率；

1，8，10 主要影响学习速度，这点主要体现在训练数据代价函数曲线的下降速度上；

5，6，11 主要影响模型分类正确率和训练用总体时间。

这上面所提到的时某个超参数对于神经网络想到的首要影响，并不代表着该超参数只影响学习速度或者正确率。

因为不同的超参数的类别不同，因此在调整超参数的时候也应该根据对应超参数的类别进行调整。再调整超参数的过程中有根据机理选择超参数的方法，有根据训练集上表现情况选择超参数的方法，也有根据验证集上训练数据选择超参数的方法。他们之间的关系如图2所示。

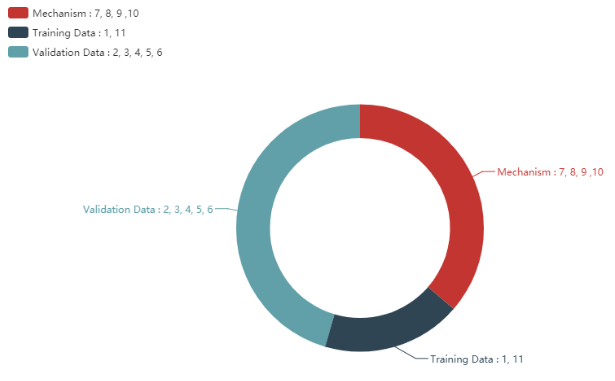


图2. 不同超参数的选择方法不同

如图2所示，超参数 7，8，9，10 由神经网络的机理进行选择。在这四个参数中，应该首先对第10个参数神经元的种类进行选择，根据目前的知识，一种较好的选择方式是对于神经网络的隐层采用sigmoid神经元，而对于输出层采用softmax的方法；根据输出层采用sotmax的方法，因此第8个代价函数采用 log-likelihood 函数（或者输出层还是正常的sigmoid神经元而代价函数为交叉熵函数），第9个初始化权重采用均值为0方差为 1nin√1nin 的高斯随机分布初始化权重；对于输出层的编码方式常常采用向量式的编码方式，基本上不会使用实际的数值或者二进制的编码方式。超参数1由训练数据的代价函数选择，在上述这两部分都确定好之后在根据检验集数据确定最后的几个超参数。