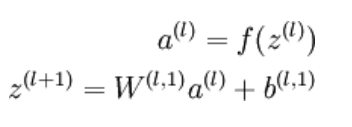
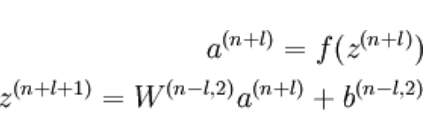
本周学习：栈式自编码算法

概念

在上次的深度网络中使用的是逐层贪婪训练算法来训练每一层，进而训练完整个深度网络。本次学习如何将自编码器“栈化”到逐层贪婪训练算法中，从而预训练深度神经网络的权重。

栈式自编码神经网络是由多层稀疏自编码器组成的神经网络，其前一层的输出作为其后一层的输入。对于一个n层的栈式自编码神经网络，假设用W（k，1），W（k，2），b（k，1），b（k，2）来表示第k个自编码器对应的W（1），W（2），b（1），b（2）参数，那么栈式自编码神经网络的编码过程是，按照从前向后的顺序执行每一层的自编码器的编码步骤如下：

那么解码过程就是从后往前顺序执行每一层自编码器的解码步骤：



其中，a（n）是最深层隐藏单元的激活值，这个向量时对输入值的更高阶的表示。通过将a（n）作为softmax分类器的输入特征，可以将栈式自编码神经网络中学到的特征用于分类问题。

训练

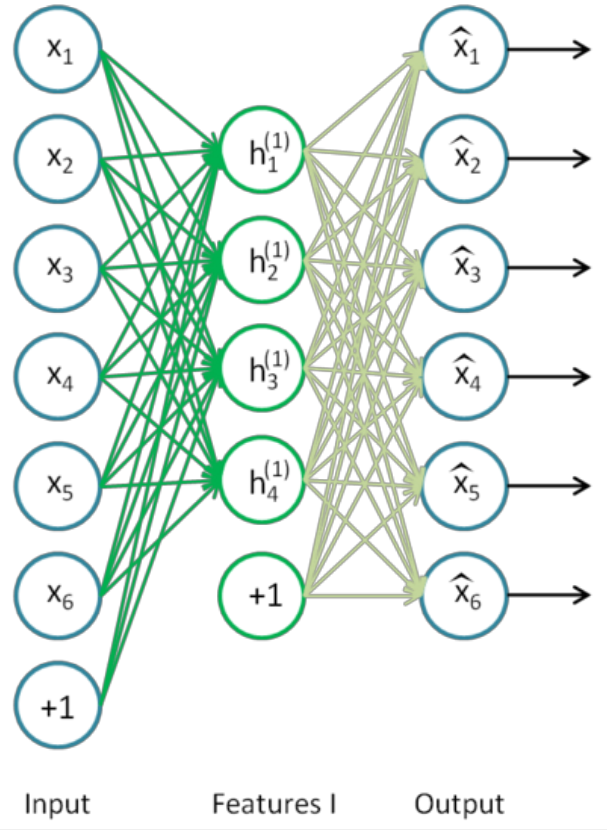
一种较好的获取栈式自编码神经网络参数的方法是采取逐层贪婪训练法进行训练。即先利用原始输入来训练网络的第一层，得到其参数W（1，1），W（1，2），b（1,1），b（1,2）；然后网络第一层将原始输入转化为优隐藏单元激活值组成的向量A，接着把向量A作为第二层的输入，继续训练得到第二层的参数W（2,1），W（2,2），b（2,1），b（2,2）；最后，对后面的各层采取相同的策略，即将前一层的输出作为下一层的输入的方式训练。

对于上述训练方式，在训练每一层参数的时候，会固定其它各层参数保持不变。所以，如果想得到更好的结果，在上述预训练过程完成之后，可以通过反向传播算法同时调整所有层的参数以改善结果，这个过程一般被称作“微调（fine-tuning）”。 实际上，使用逐层贪婪训练方法将参数训练达快要收敛时，应该使用微调。反之，如果直接在随机化的初始权重上使用微调，那么虎得到不好的结果，因为参数会收敛到局部最优。

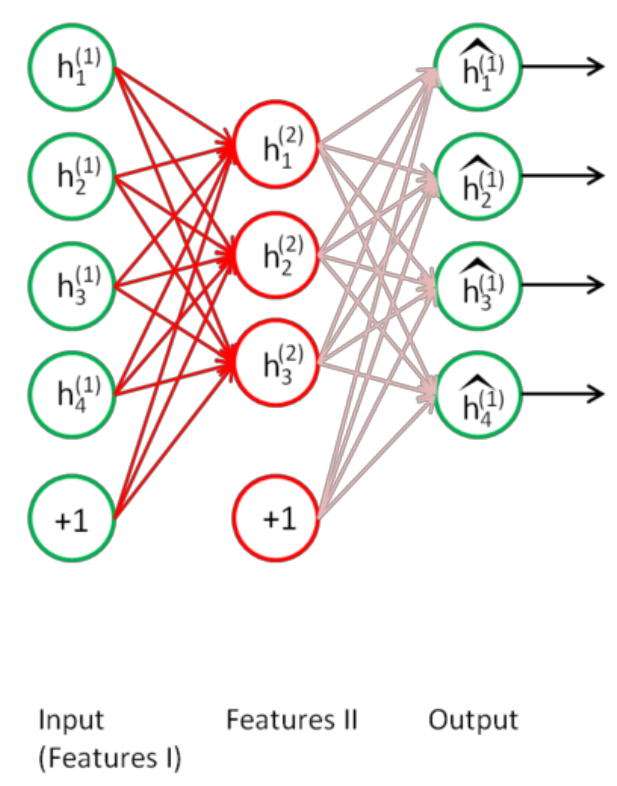
如果只是对以分类为目的的微调感兴趣，那么惯用的做法是丢掉栈式自编码网络的“解码”层，直接把最后一个隐藏层的作为特征输入到softmax分类器进行分类，这样，分类器（softmax）的分类错误的梯度值就可以直接反向传播给编码层了。

实例

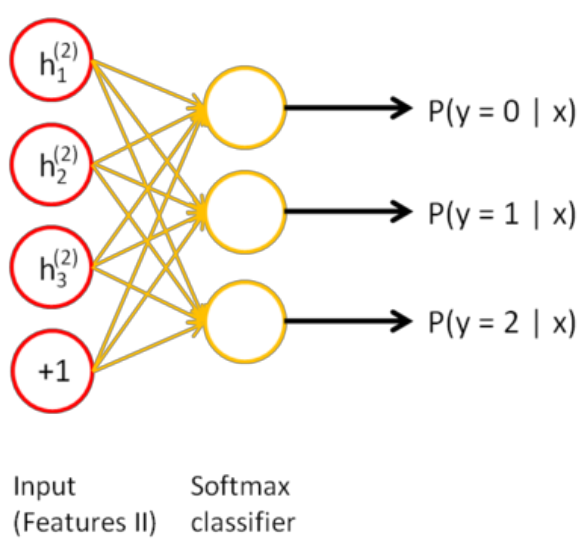
假设想要训练一个包含两个隐含层的栈式自编码网络，用来进行MNIST手写数字分类（。 首先，需要用原始输入x(k)训练第一个自编码器，它能够学习得到原始输入的一阶特征表示h（1）（k）（如下图所示）。



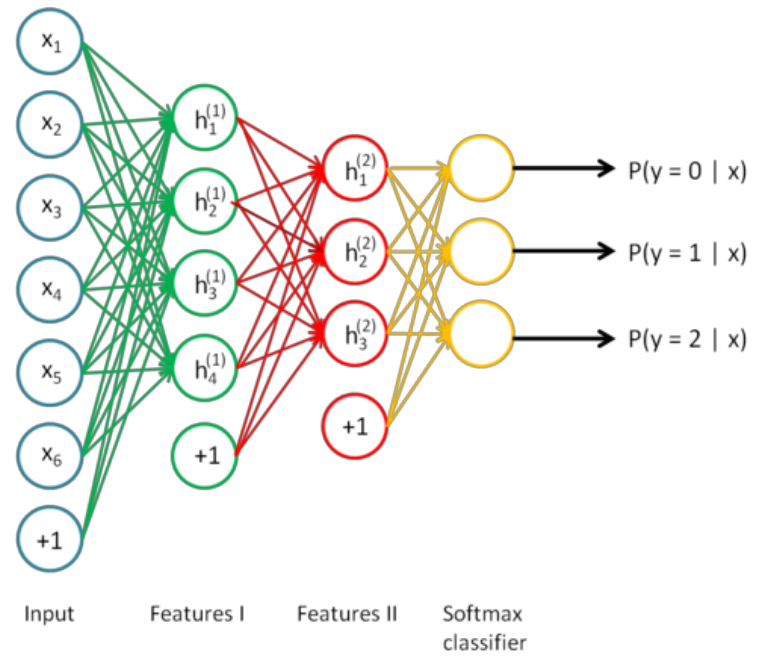
接着，你需要把原始数据输入到上述训练好的稀疏自编码器中，对于每一个输入x（k），都可以得到它对应的一阶特征表示h（1）（k）。然后你再用这些一阶特征作为另一个稀疏自编码器的输入，使用它们来学习二阶特征h（2）（k）。（如下图所示）



同样，再把一阶特征输入到刚训练好的第二层稀疏自编码器中，得到每个h（1）（k）对应的二阶特征激活值h（2）（k）。接下来，可以把这些二阶特征作为softmax分类器的输入，训练得到一个能将二阶特征映射到数字标签的模型。



如下图所示，最终，可以将这三层结合起来构建一个包含两个隐藏层和一个最终softmax分类器层的栈式自编码网络，这个网络能够如你所愿地对MNIST数字进行分类。



讨论

栈式自编码神经网络具有强大的表达能力及深度神经网络的所有优点。  
更进一步，它通常能够获取到输入的“层次型分组”或者“部分-整体分解”结构。为了弄清这一点，回顾一下，自编码器倾向于学习得到能更好地表示输入数据的特征。因此，栈式自编码神经网络的第一层会学习得到原始输入的一阶特征（比如图片里的边缘），第二层会学习得到二阶特征，该特征对应一阶特征里包含的一些模式（比如在构成轮廓或者角点时，什么样的边缘会共现）。栈式自编码神经网络的更高层还会学到更高阶的特征。  
举个例子，如果网络的输入数据是图像，网络的第一层会学习如何去识别边，第二层一般会学习如何去组合边，从而构成轮廓、角等。更高层会学习如何去组合更形象且有意义的特征。例如，如果输入数据集包含人脸图像，更高层会学习如何识别或组合眼睛、鼻子、嘴等人脸器官。

参考资料：

1.UFLDL深度学习教程