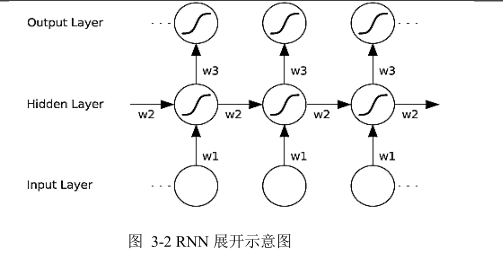
## 目标

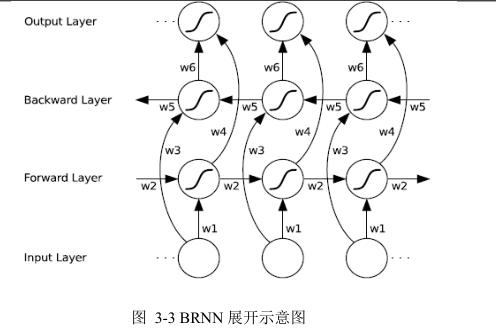
《基于LSTM的语义关系分类研究》

《零基础入门深度学习(4) - 卷积神经网络》

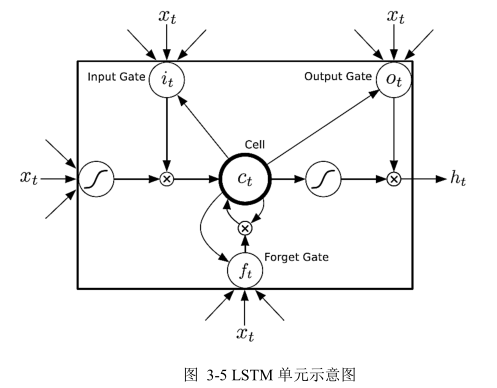
### 收获

相对于传统的RNN（只能单向输入序列数据，无法利用未来的信息），Schuster 等 [73] 在 1997 年提出了一个解决方案-双向 RNN（Bidirectionalrecurrent neural networks, BRNN）。BRNN 的核心思想是将序列数据分两个方向分别输入模型，模型使用两个隐含层分别保存两个方向输入数据的信息，并连接到相同输出层。



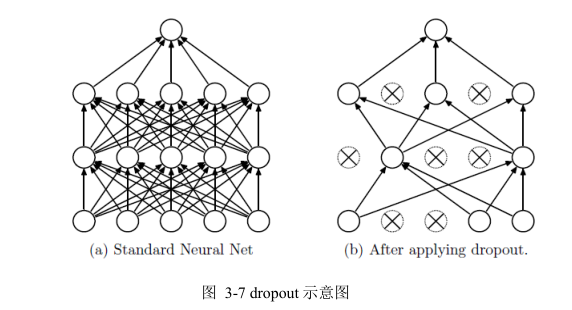


而LSTM则解决了RNN的梯度消失问题



Dropout介绍

Dropout作为一种防止神经网络模型过拟合的技术是Hintion在 2012 年的时候提出的



图左为传统多层全连接神经网络应用 dropout 之前的网络结构，图右为应dropout之后的网络结构。

dropout在模型训练时随即让一定比例的隐含节点不工作，而这些不工作的节点所对应的权值则会在这次训练中不更新（mask过程，即输出置0）。但是在模型使用的时候，所有节点都会被用到，恢复成全连接的状态。

大量的实验表明，dropout技术能极大的降低错误率。提升系统性能，尤其在原模型过拟严重的情况下。但至今却未能有一个精确的数学解释来解释dropout有效的原因。

Dropout 提出以来，被广泛的实验证明有效，但是也有例外的情况，如 H.Brendan McMahan 等人在他们的论文中提出“dropout 作用于稠密的特征表示时能够通过防止特征间的协同作用来对抗过拟合。但是如果特征本身稀疏，则dropout 不仅仅没有起到上述作用，而且还会损失本身就不多的可用信息 ”。

ReLU

ReLU作为激活函数可以有公式表示5.png

优点：

1. 研究者认为 ReLU 将负值部分置 0，保留正值部分的做法为模型隐含层引入了稀疏性，因此提高了模型的性能。

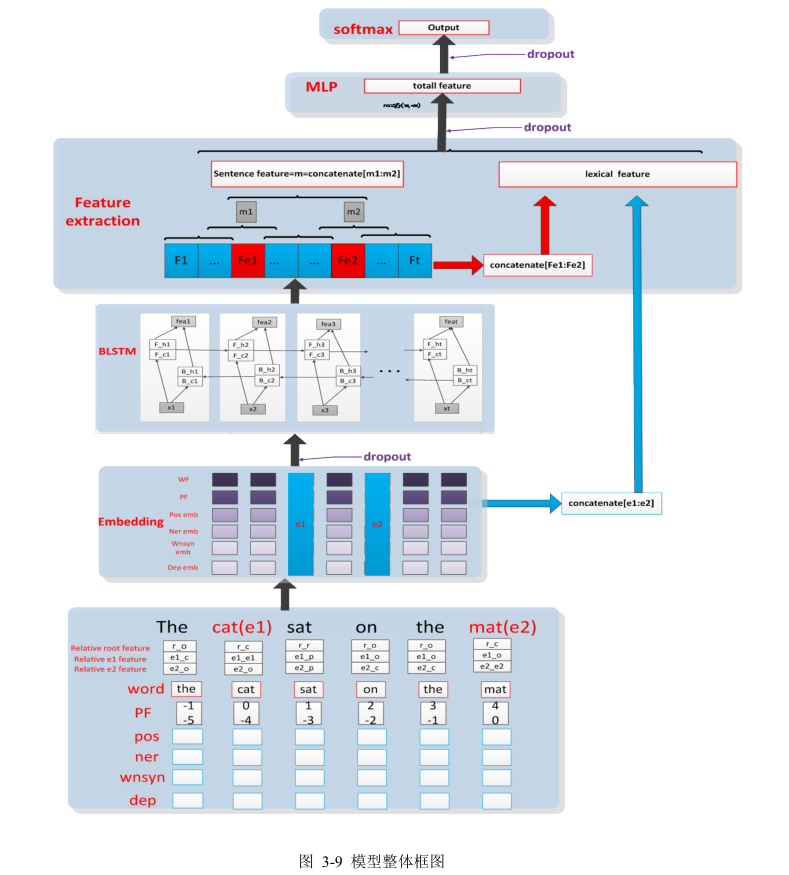
2. 传统的饱和激活函数，如 sigmoid 和 tanh 会带来梯度消失问题，而 ReLU 等非饱和激活函数不会,可以训练更加深层次的网络。

3.ReLU 等非饱和激活函数相对于饱和激活函数，能够加快模型收敛速度。

4.Bengio 等人表示，使用 ReLU 的深度模型在有监督训练之前不需要预训练就能得到相近甚至更好的结果。

基于LSTM的模型介绍

模型如图所示：



（1） 数据预处理，特征抽取。

（2） Embedding 层，将所有输入包括各种特征全部转为向量形式。

（3） BLSTM 层，经过 BLSTM 模型抽取特征。

（4）特征抽取，从之前 embedding 层和 BLSTM 的输出中抽取出两类特征（实体特征 lexical feature 和句子级别特征 sentence feature）。

（5）特征融合，分类。使用 MLP 对之前抽取的两类特征进行融合之后送入softmax 分类器分类。

卷积网络部分

全连接网络VS卷积网络

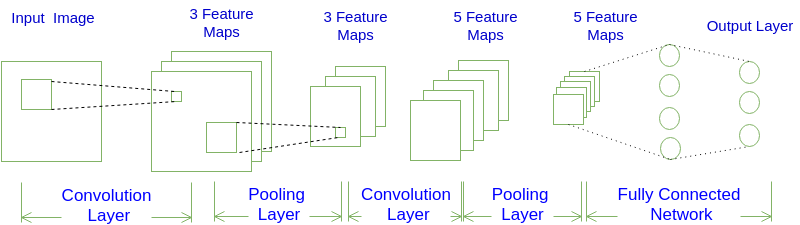
全连接网络不太适合图像识别的原因：

1. 参数数量太多：比如一个1000\*1000的像素的图片，输入层有1000\*1000的节点，假设第一个隐藏层就有100个节点，则仅这一层的图片就有1000\*1000\*100=1亿个参数，所以它的扩展性很差。
2. 没有利用像素点之前的位置关系：在图像识别任务中，每个像素与周围像素的关系是比较紧密的，但和离得很远的像素的联系就比较小了。若将一个神经元和上一层所有神经元相连，那么就相当于对于一个像素来说，把图像的所有像素都等同看待，这不符合我们的假设，当完成每个权重的学习后最终可能会发现，有大量的权重，它们的值都是很小的(也就是这些连接其实无关紧要)。努力学习大量并不重要的权重，这样的学习必将是非常低效的。
3. 网络测测你灌输的限制：通过梯度下降的方法训练深度全连接神经网络很困难，因为对于全连接神经网络，它的梯度很难传递超过三层，这将会限制它的能力。

而卷积网络则：

* 局部连接：每个神经元将不会和上一层的所有神经元相连接，这样就减少了很多参数。
* 权值共享：一组连接可以共享同一个权重，而不是每个连接有一个不同的权重
* 下采样：可以使用pooling来减少每层的样本数，进一步减少参数数量

卷积网络示意图：



由上可知，一个卷积网络由若干卷积层、Pooling层、全连接层组成

其构架模式常为：INPUT -> [[CONV]\*N -> POOL?]\*M –> [FC]\*k

N个卷积层叠加，然后可选一个Pooling层，重复这个结构M次，最后叠加K个全连接层，对于图一展示的卷积神经网络：INPUT ->CONV ->POOL -> CONV -> FC -> FC

对于图1展示的神经网络，我们看到输入层的宽度和高度对应于输入图像的宽度和高度，而它的深度为1。接着，第一个卷积层对这幅图像进行了卷积操作(后面我们会讲如何计算卷积)，得到了三个Feature Map。这里的"3"可能是让很多初学者迷惑的地方，实际上，就是这个卷积层包含三个Filter，也就是三套参数，每个Filter都可以把原始输入图像卷积得到一个Feature Map，三个Filter就可以得到三个Feature Map。至于一个卷积层可以有多少个Filter，那是可以自由设定的。也就是说，卷积层的Filter个数也是一个超参数。我们可以把Feature Map可以看做是通过卷积变换提取到的图像特征，三个Filter就对原始图像提取出三组不同的特征，也就是得到了三个Feature Map，也称做三个通道(channel)。

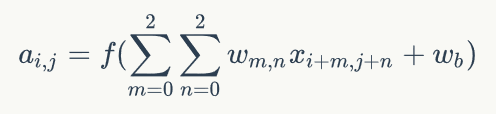
在第一个卷积层之后，Pooling层对三个Feature Map做了下采样

，得到了三个更小的Feature Map。接着，是第二个卷积层，它有5个Filter。每个Fitler都把前面下采样之后的3个\*\*Feature Map卷积在一起，得到一个新的Feature Map。这样，5个Filter就得到了5个Feature Map。接着，是第二个Pooling，继续对5个Feature Map**进行**下采样\*\*，得到了5个更小的Feature Map。

网络的最后两层是全连接层。第一个全连接层的每个神经元，和上一层5个Feature Map中的每个神经元相连，第二个全连接层(也就是输出层)的每个神经元，则和第一个全连接层的每个神经元相连，这样得到了整个网络的输出。

卷积神经网络输出值的计算

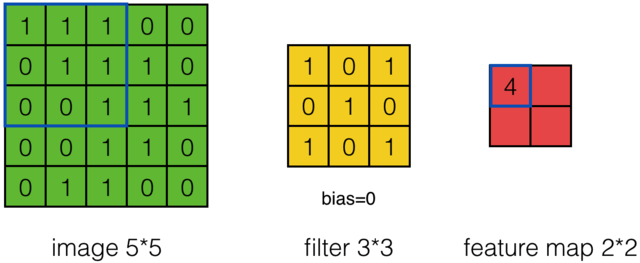
假设一个5\*5的图像，使用一个3\*3的filter进行卷积，想得到一个3\*3的Feature Map。

用表示图像的第 I 行第 j 列元素；对filter的每个权重进行编号，用表示第 I 行第 j 列权重，用表示filter的偏置项；对Feature Map的每个元素进行编号，用表示Feature Map的第 I 行第 j 列元素；用表示激活函数(这个例子选择 relu 函数作为激活函数)。然后，使用下列公式计算卷积：

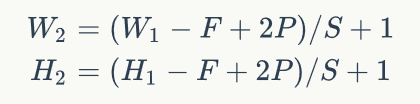
比如计算结果如下图所示：



上述计算过程，步幅为1，下列为a0，0计算过程且步幅为2

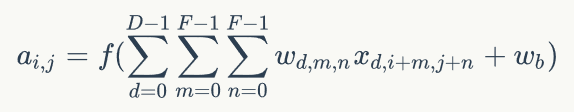


图像大小、步幅和卷积后的Feature Map大小是有关系的且关系为：



W2是卷积后Feature Map的宽度，W1是卷积前图像的宽度，F的filter的宽度，P是Zero Padding数量，Zero Padding是指在原始图像周围补几圈0.若P的值是1，那么就补1圈0，H2是卷积后Feature Map的高度，H1是卷积前图像的宽度。

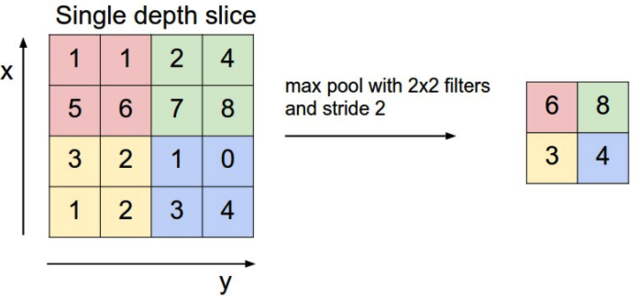
若深度大于一的计算方法：



D是深度，Wd，m，n表示filter的第d层第m行第n列权重，ad，i，j表示图像的第d层第i行第j列像素。

Pooling层输出值的计算

Pooling层的主要作用是下采样，去掉Feature Map中不重要的样本，进一步减少参数数量。Pooling的方法中最常用的是Max Pooling。Max Pooling实际上就是在n\*n的样本中取最大值，下图就是2\*2 max pooling



常用的还有Mean Pooling，取各样本的平均值