Dropout是指在深度学习网络的训练过程中，**对于神经网络单元，按照一定的概率将其暂时从网络中丢弃**。注意是暂时，对于随机梯度下降来说，由于是随机丢弃，故而每一个mini_batch都在训练不同的网络。

dropout是CNN中防止过拟合提高效果的一个大杀器，但对于其为何有效，却众说纷纭。在下读到两篇代表性的论文，代表两种不同的观点，特此分享给大家。

1.dropout和bagging

dropout的思想继承自bagging方法.

bagging的**最基本的思想**是通过分别训练几个不同分类器，最后对测试的样本，每个分类器对其进行投票。在机器学习上这种策略叫model averaging。 model averaging 之所以有效，是因为并非所有的分类器都会产

生相同的误差，只要有不同的分类器产生的误差不同就会对减小泛化误差非常有效。

我们可以把dropout类比成将许多大的神经网络进行集成的一种bagging方法。

bagging：取多组训练数据，用相同的算法训练不同的模型；

dropout：每个batch随机抑制一部分神经元，相当于做了不同的模型；

bagging与dropout训练的对比

- 在bagging中，所有的分类器都是独立的，而在dropout中，所有的模型都是共享参数的。
- 在bagging中，所有的分类器都是在特定的数据集下训练至收敛，而在dropout中没有明确的模型训练过程。网络都是在一步中训练一次（输入一个样本，随机训练一个子网络）
- （相同点）对于训练集来说，每一个子网络的训练数据是通过原始数据的替代采样得到的子集。

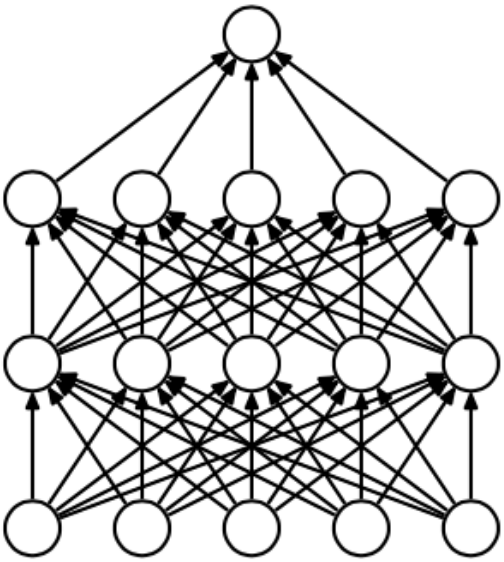
1.组合派

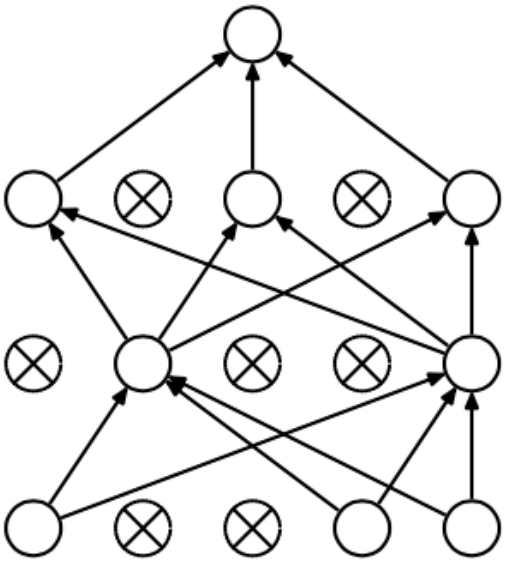
> [1]. Srivastava N, Hinton G, Krizhevsky A, et al. Dropout: A simple way to prevent neural networks from overfitting[J]. The Journal of Machine Learning Research, 2014, 15(1): 1929-1958.

1.1 观点

大规模的神经网络有两个缺点：费时和容易过拟合

Dropout的出现很好的可以解决这个问题，每次做完dropout，相当于从原始的网络中找到一个更瘦的网络，如下图所示：





因而，对于一个有N个节点的神经网络，有了dropout后，就可以看做是2^n个模型的集合了，但此时要训练的参数数目却是不变的，这就解脱了费时的问题。

1.2 动机

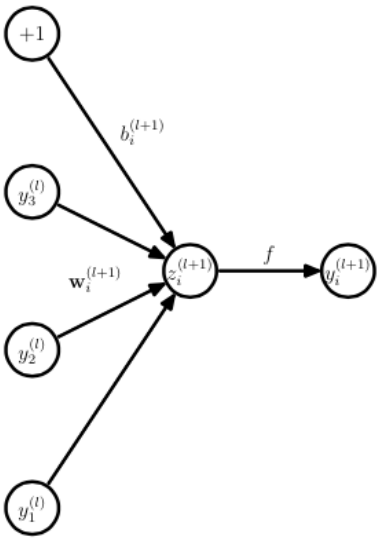
虽然直观上看dropout是ensemble在分类性能上的一个近似，然而实际中，dropout毕竟还是在一个神经网络上进行的，只训练出了一套模型参数。那么他到底是因何而有效呢？

Dropout强迫一个神经元和随机挑选出来的其他神经元共同工作，达到好的效果。消除减弱了神经元节点间的联合适应性，增强了泛化能力。

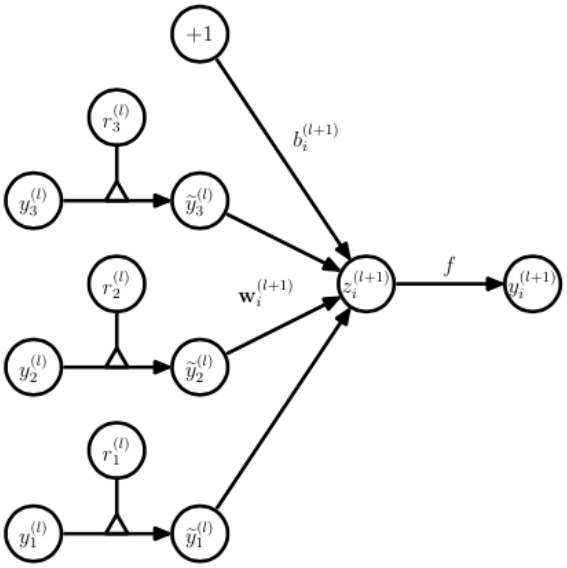
1.3 dropout带来的模型的变化

1. 训练层面

无可避免的，训练网络的每个单元要添加一道概率流程



(a) Standard network



(b) Dropout network

对应的公式变化如下：

- 没有dropout

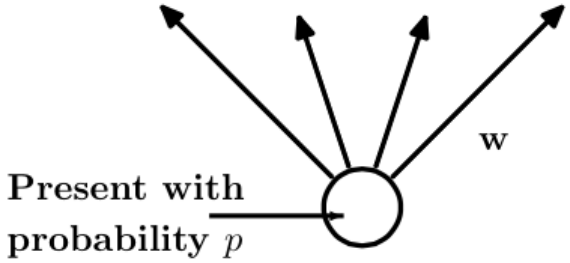
$$
\begin{aligned}
z_i^{(l+1)} &= \mathbf{w}_i^{(l+1)}\mathbf{y}^l + b_i^{(l+1)}, \\
y_i^{(l+1)} &= f(z_i^{(l+1)}),
\end{aligned}
$$

- 有dropout

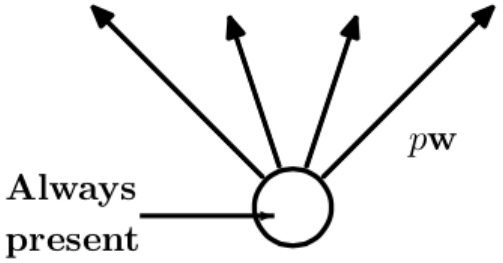
$$
\begin{aligned}
r_j^{(l)} &\sim \text{Bernoulli}(p), \\
\tilde{\mathbf{y}}^{(l)} &= \mathbf{r}^{(l)} * \mathbf{y}^{(l)}, \\
z_i^{(l+1)} &= \mathbf{w}_i^{(l+1)}\tilde{\mathbf{y}}^l + b_i^{(l+1)}, \\
y_i^{(l+1)} &= f(z_i^{(l+1)}).
\end{aligned}
$$

2. 测试层面

预测的时候，每一个单元的参数都要预乘以p



(a) At training time



(b) At test time

1.4.为什么说Dropout可以解决过拟合?

（1）取平均的作用：　先回到标准的模型即没有dropout，我们用相同的训练数据去训练5个不同的神经网络，一般会得到5个不同的结果，此时我们可以采用 “5个结果取均值”或者“多数取胜的投票策略”去决定最终结果。例如3个网络判断结果为数字9,那么很有可能真正的结果就是数字9，其它两个网络给出了错误结果。这种“综合起来取平均”的策略通常可以有效防止过拟合问题。因为不同的网络可能产生不同的过拟合，取平均则有可能让一些“相反的”拟合互相抵消。**dropout掉不同的隐藏神经元就类似在训练不同的网络，随机删掉一半隐藏神经元导致网络结构已经不同，整个dropout过程就相当于对很多个不同的神经网络取平均。而不同的网络产生不同的过拟合，一些互为“反向”的拟合相互抵消就可以达到整体上减少过拟合。**

（2）减少神经元之间复杂的共适应关系：　因为dropout程序导致两个神经元不一定每次都在一个dropout网络中出现。这样权值的更新不再依赖于有固定关系的隐含节点的共同作用，阻止了某些特征仅仅在其它特定特征下才有效果的情况 。**迫使网络去学习更加鲁棒的特征 ，这些特征在其它的神经元的随机子集中也存在。换句话说假如我们的神经网络是在做出某种预测，它不应该对一些特定的线索片段太过敏感，即使丢失特定的线索，它也应该可以从众多其它线索中学习一些共同的特征。**从这个角度看dropout就有点像L1，L2正则，减少权重使得网络对丢失特定神经元连接的鲁棒性提高。

（3）Dropout类似于性别在生物进化中的角色：物种为了生存往往会倾向于适应这种环境，环境突变则会导致物种难以做出及时反应，性别的出现可以繁衍出适应新环境的变种，有效的阻止过拟合，即避免环境改变时物种可能面临的灭绝。

1.5 论文中的其他技术点

- 防止过拟合的方法:

  提前终止（当验证集上的效果变差的时候）

  L1和L2正则化加权

  soft weight sharing

  dropout

- dropout率的选择

经过交叉验证，隐含节点dropout率等于0.5的时候效果最好，原因是0.5的时候dropout随机生成的网络结构最多。

dropout也可以被用作一种添加噪声的方法，直接对input进行操作。输入层设为更接近1的数。使得输入变化不会太大（0.8）

- 训练过程

对参数w的训练进行球形限制(max-normalization)，对dropout的训练非常有用。

球形半径c是一个需要调整的参数。可以使用验证集进行参数调优

dropout自己虽然也很牛，但是dropout、max-normalization、large decaying learning rates and high momentum组合起来效果更好，比如max-norm regularization就可以防止大的learning rate导致的参数blow up。

使用pretraining方法也可以帮助dropout训练参数，在使用dropout时，要将所有参数都乘以1/p。

- 部分实验结论

该论文的实验部分很丰富，有大量的评测数据。

1. maxout 神经网络中得另一种方法，Cifar-10上超越dropout

2. 文本分类上，dropout效果提升有限，分析原因可能是Reuters-RCV1数据量足够大，过拟合并不是模型的主要问题

3. dropout与其他standerd regularizers的对比

L2 weight decay

lasso

KL-sparsity

max-norm regularization

dropout

4. 特征学习

标准神经网络，节点之间的相关性使得他们可以合作去fix其他节点中得噪声，但这些合作并不能在unseen data上泛化，于是，过拟合，dropout破坏了这种相关性。在autoencoder上，有dropout的算法更能学习有意义的特征（不过只能从直观上，不能量化）。

产生的向量具有稀疏性。

保持隐含节点数目不变，dropout率变化；保持激活的隐节点数目不变，隐节点数目变化。

5. 数据量小的时候，dropout效果不好，数据量大了，dropout效果好

6. 模型均值预测

使用weight-scaling来做预测的均值化

使用mente-carlo方法来做预测。即对每个样本根据dropout率先sample出来k个net，然后做预测，k越大，效果越好。

7. Multiplicative Gaussian Noise

使用高斯分布的dropout而不是伯努利模型dropout

8. dropout的缺点就在于训练时间是没有dropout网络的2-3倍。

2.噪声派（没看懂 先不看）

Dropout as data augmentation. http://arxiv.org/abs/1506.08700

2.1 观点

就是对于每一个dropout后的网络，进行训练时，相当于做了Data Augmentation，因为，总可以找到一个样本，使得在原始的网络上也能达到dropout单元后的效果。 比如，对于某一层，dropout一些单元后，形成的结果是(1.5,0,2.5,0,1,2,0)，其中0是被drop的单元，那么总能找到一个样本，使得结果也是如此。这样，每一次dropout其实都相当于增加了样本。

2.2稀疏性

知识点A

首先，先了解一个知识点：

> When the data points belonging to a particular class are distributed along a linear manifold, or sub-space, of the input space, it is enough to learn a single set of features which can span the entire manifold. But when the data is distributed along a highly non-linear and discontinuous manifold, the best way to represent such a distribution is to learn features which can explicitly represent small local regions of the input space, effectively “tiling” the space to define non-linear decision boundaries.

大致含义就是：

在线性空间中，学习一个整个空间的特征集合是足够的，但是当数据分布在非线性不连续的空间中得时候，则学习局部空间的特征集合会比较好。

知识点B

假设有一堆数据，这些数据由M个不同的非连续性簇表示，给定K个数据。那么一个有效的特征表示是将输入的每个簇映射为特征以后，簇之间的重叠度最低。使用A来表示每个簇的特征表示中激活的维度集

合。重叠度是指两个不同的簇的Ai和Aj之间的Jaccard相似度最小，那么：

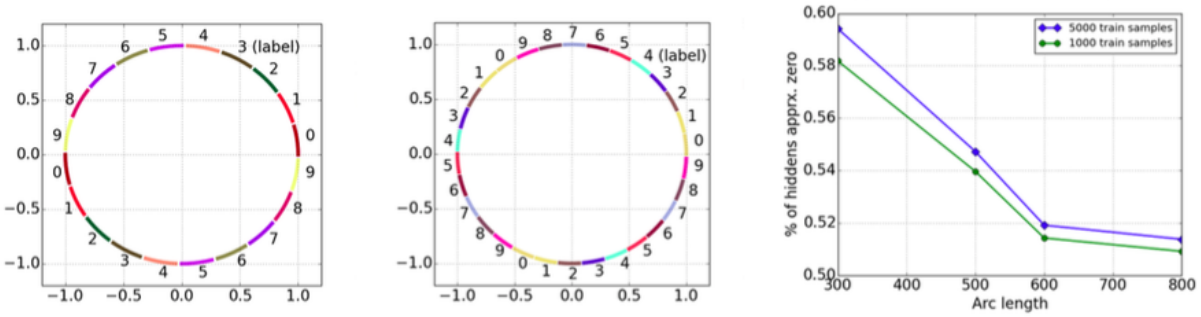
当K足够大时，即便A也很大，也可以学习到最小的重叠度

当K小M大时，学习到最小的重叠度的方法就是减小A的大小，也就是稀疏性。

主旨意思是这样，我们要把不同的类别区分出来，就要是学习到的特征区分度比较大，在数据量足够的情况下不会发生过拟合的行为，不用担心。但当数据量小的时候，可以通过稀疏性，来增加特征的区分度。

因而有意思的假设来了，使用了dropout后，相当于得到更多的局部簇，同等的数据下，簇变多了，因而为了使区分性变大，就使得稀疏性变大。

为了验证这个数据，论文还做了一个实验，如下图：



(a) Arclength=300　　(b) Arclength=500　　(c) Hidden layer sparsity

该实验使用了一个模拟数据，即在一个圆上，有15000个点，将这个圆分为若干个弧，在一个弧上的属于同一个类，一共10个类，即不同的弧也可能属于同一个类。改变弧的大小，就可以使属于同一类的弧变多。

实验结论就是当弧长变大时，簇数目变少，稀疏度变低。与假设相符合。

个人观点：该假设不仅仅解释了dropout何以导致稀疏性，还解释了dropout因为使局部簇的更加显露出来，而根据知识点A可得，使局部簇显露出来是dropout能防止过拟合的原因，而稀疏性只是其外在表现。