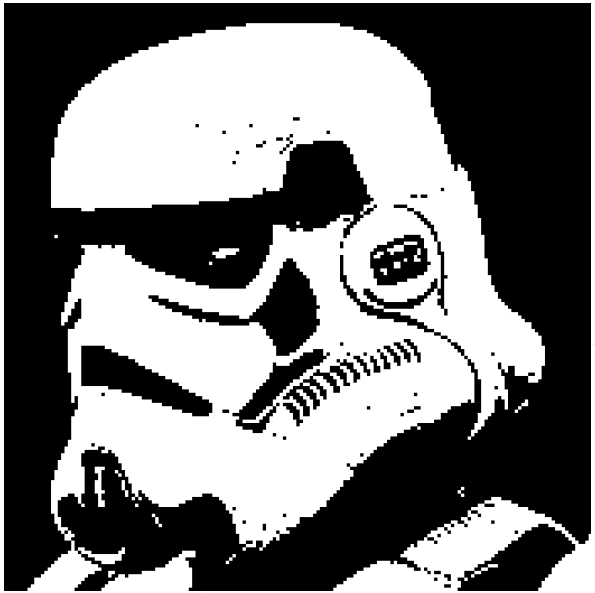
丢弃法

可能比 Weight Decay 效果更好

动机

- 一个好的模型需要对输入数据的扰动鲁棒
  - 使用有噪音的数据等价于 Tikhonov 正则
  - 丢弃法：在层之间加入噪音



无偏差的加入噪音

- 对 $\mathbf{x}$ 加入噪音得到 $\mathbf{x}'$，我们希望

$$\mathbf{E}[\mathbf{x}'] = \mathbf{x}$$

虽然加了噪音，但不要改变我的期望

- 丢弃法对每个元素进行如下扰动 保持期望不变

$$x_i' = \begin{cases} 0 & \text{with probablity } p \\ \frac{x_i}{1-p} & \text{otherise} \end{cases}$$

$$E[x_i'] = p \cdot 0 + (1-p) \cdot \frac{x_i}{1-p} = x_i$$

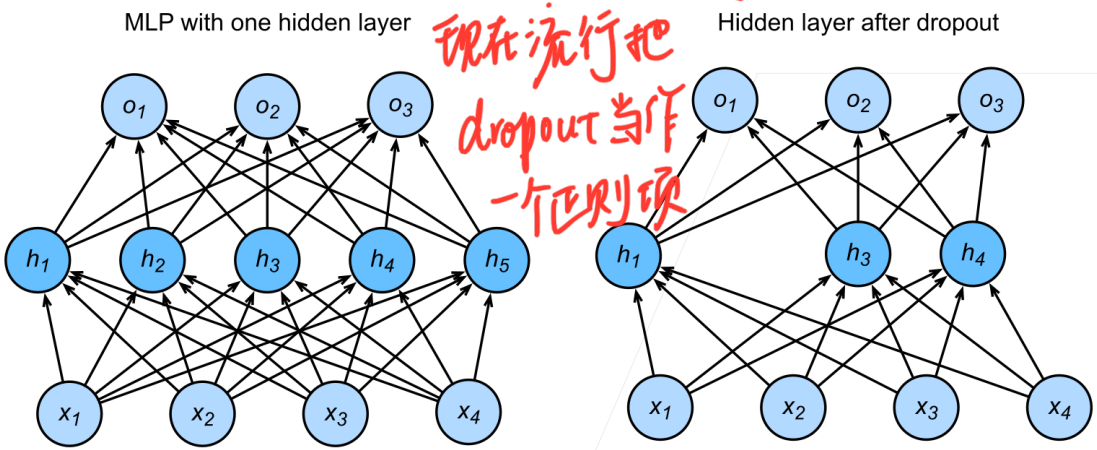
使用丢弃法

作者最早的解释是每次随机采样一个小的子网络，并综合多个子网络做平均，所以模型比较好。

- 通常将丢弃法作用在隐藏全连接层的输出上

后来人们觉得从实验上来说，它和正则效果是一样的，所以现在流行把dropout当作一个正则项

$$\mathbf{h} = \sigma(\mathbf{W}_1\mathbf{x} + \mathbf{b}_1)$$
$$\mathbf{h}' = \text{dropout}(\mathbf{h})$$
$$\mathbf{o} = \mathbf{W}_2\mathbf{h}' + \mathbf{b}_2$$
$$\mathbf{y} = \text{softmax}(\mathbf{o})$$



推理中的丢弃法

- 正则项只在训练中使用：他们影响模型参数的更新
- 在推理过程中，丢弃法直接返回输入

$$\mathbf{h} = \text{dropout}(\mathbf{h})$$

  - 这样也能保证确定性的输出

总结

- 丢弃法将一些输出项随机置0来控制模型复杂度
- 常作用在多层感知机的隐藏层输出上
- 丢弃概率是控制模型复杂度的超参数