Dropout

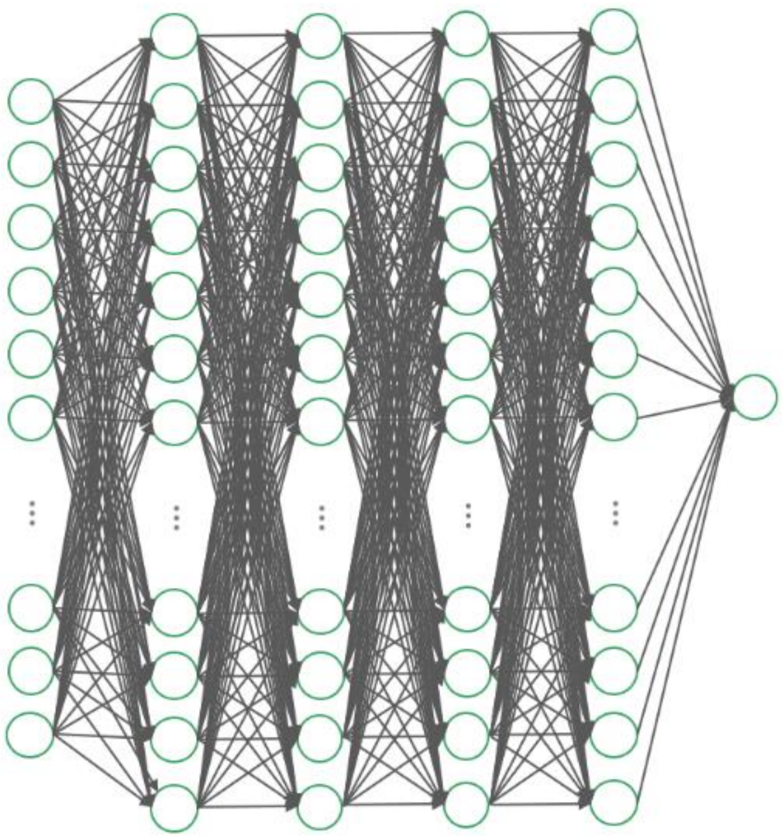
赵洲

浙江大学计算机学院

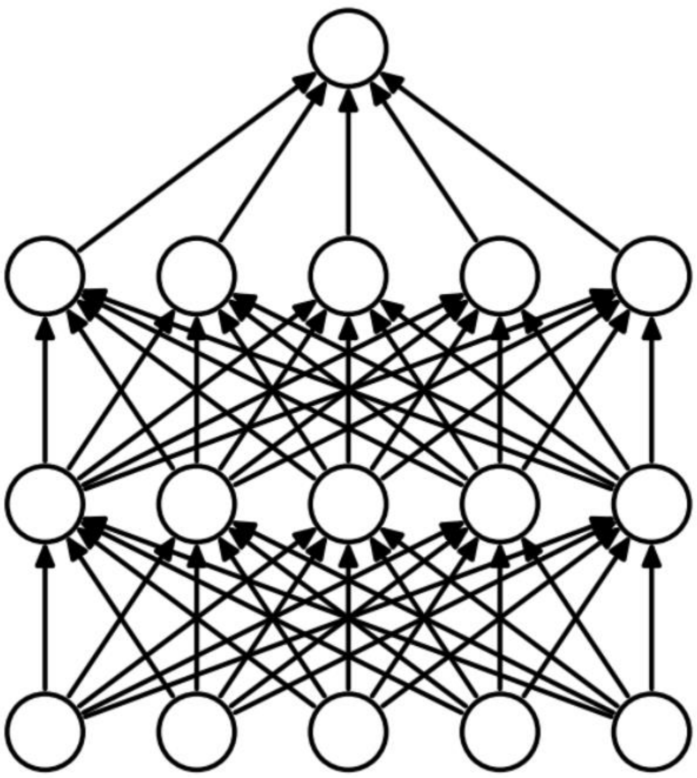
Dropout研究动机

- 解决神经网络中的co-adaptation问题。

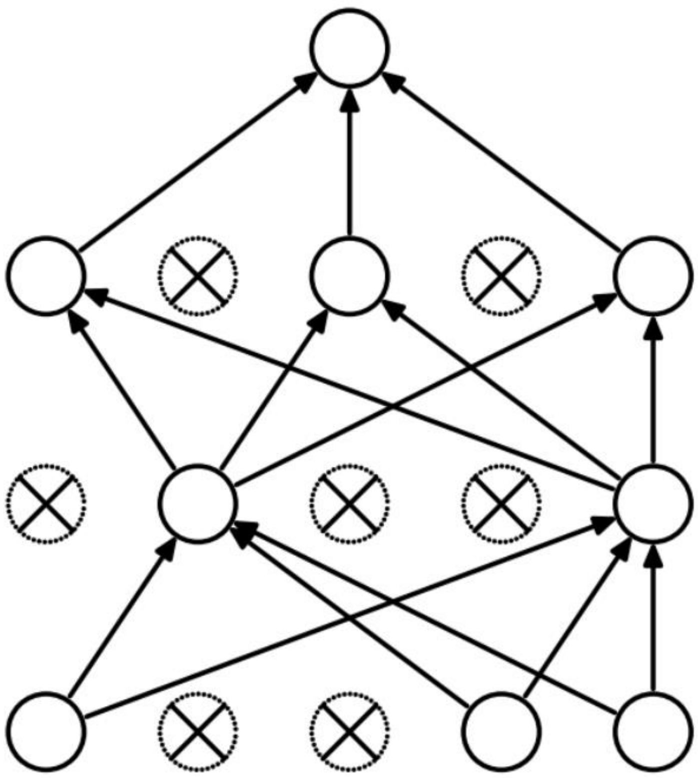
  “ co-adaptation refers to when different hidden units in a neural networks have highly correlated behavior”



什么是Dropout?

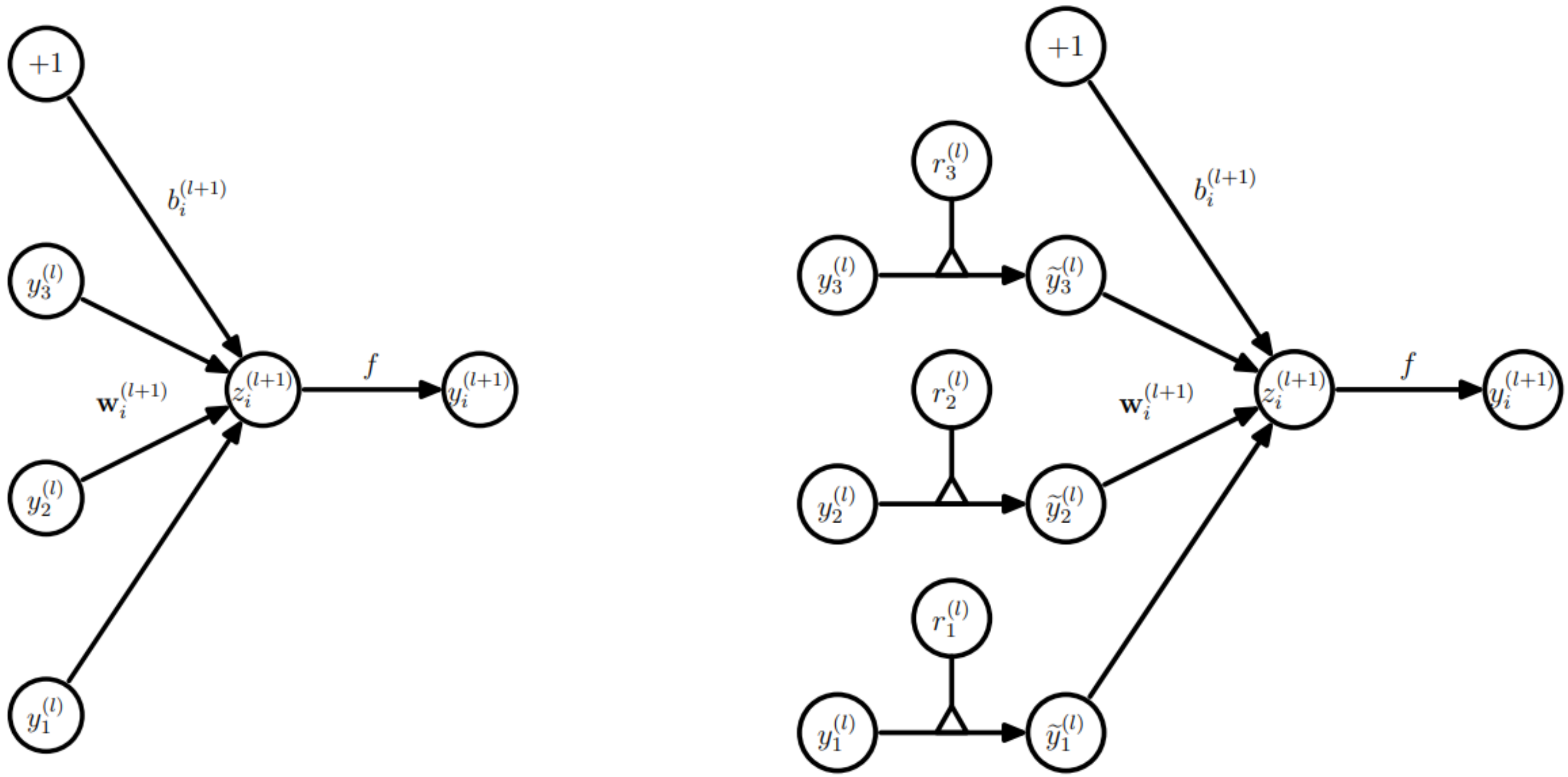


(a) Standard Neural Net



(b) After applying dropout.

网络对比



(a) Standard network

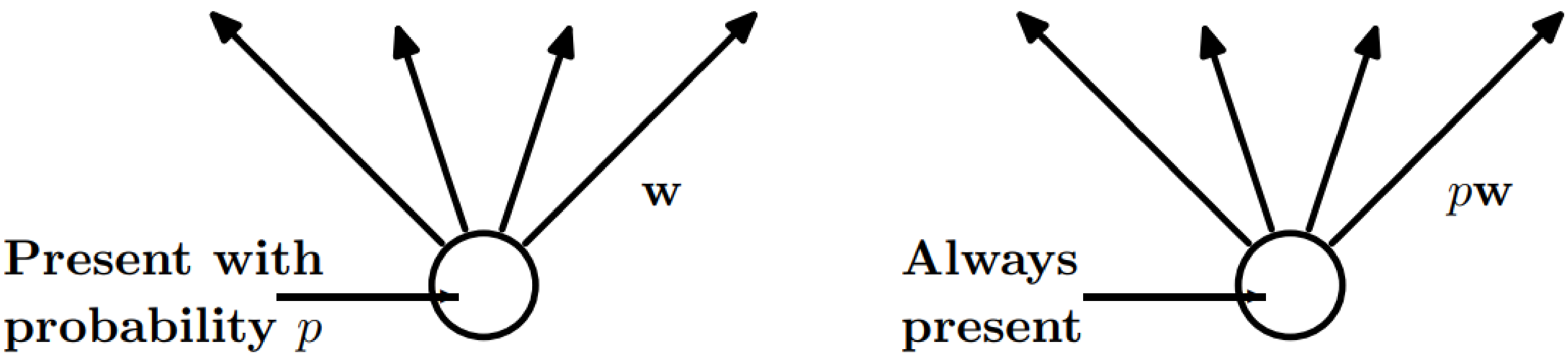
(b) Dropout network

模型描述

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

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

训练与推理



(a) At training time (b) At test time

横向比较

Method	Unit Type	Architecture	Error %
Standard Neural Net (Simard et al., 2003)	Logistic	2 layers, 800 units	1.60
SVM Gaussian kernel	NA	NA	1.40
Dropout NN	Logistic	3 layers, 1024 units	1.35
Dropout NN	ReLU	3 layers, 1024 units	1.25
Dropout NN + max-norm constraint	ReLU	3 layers, 1024 units	1.06
Dropout NN + max-norm constraint	ReLU	3 layers, 2048 units	1.04
Dropout NN + max-norm constraint	ReLU	2 layers, 4096 units	1.01
Dropout NN + max-norm constraint	ReLU	2 layers, 8192 units	0.95
Dropout NN + max-norm constraint (Goodfellow et al., 2013)	Maxout	2 layers, (5×240) units	0.94
DBN + finetuning (Hinton and Salakhutdinov, 2006)	Logistic	500-500-2000	1.18
DBM + finetuning (Salakhutdinov and Hinton, 2009)	Logistic	500-500-2000	0.96
DBN + dropout finetuning	Logistic	500-500-2000	0.92
DBM + dropout finetuning	Logistic	500-500-2000	**0.79**

Table 2: Comparison of different models on MNIST.

鲁棒性结果

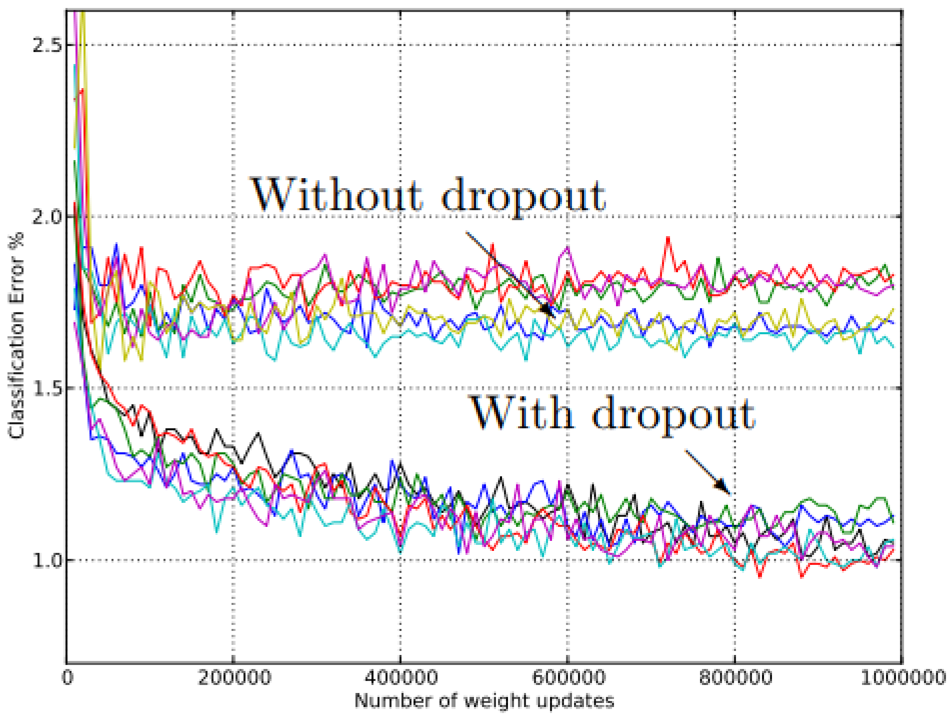
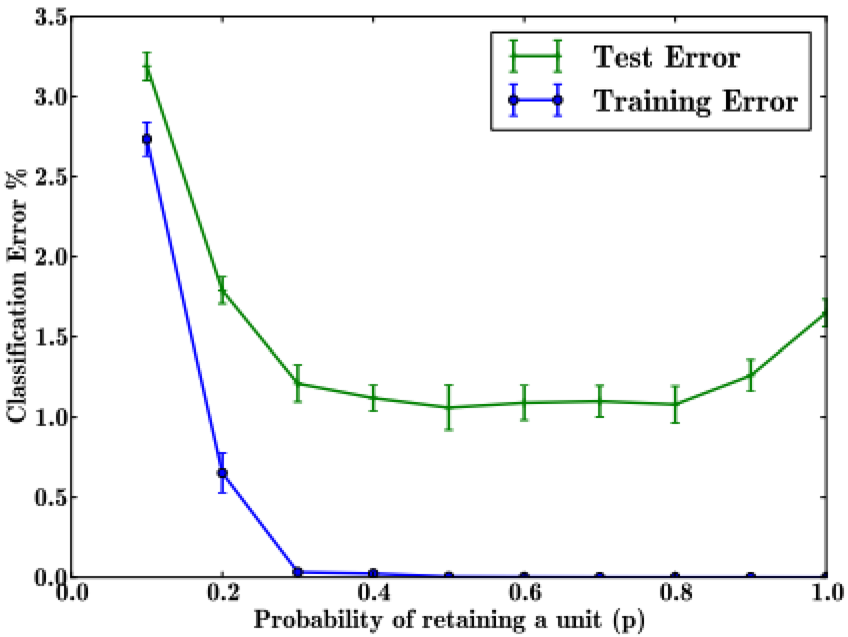
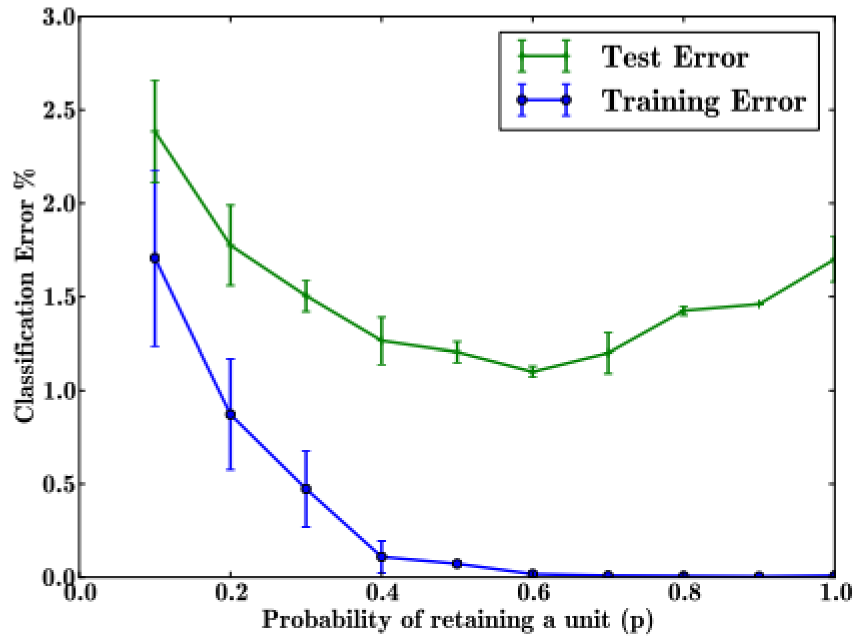


Figure 4: Test error for different architectures with and without dropout. The networks have 2 to 4 hidden layers each with 1024 to 2048 units.

P值影响



(a) Keeping n fixed.



(b) Keeping pn fixed.

训练数据大小影响

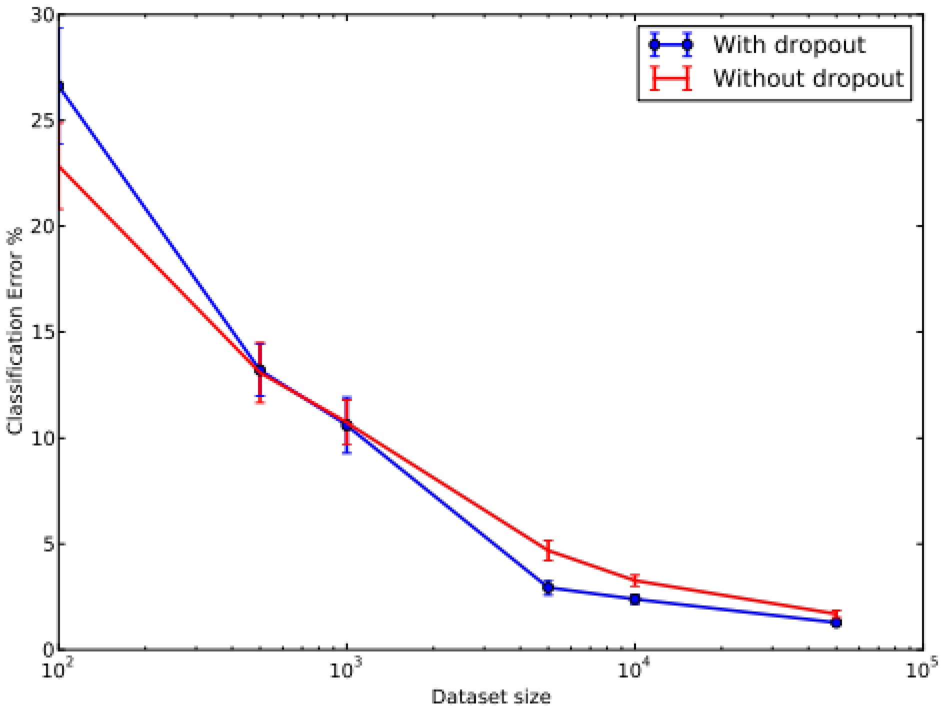
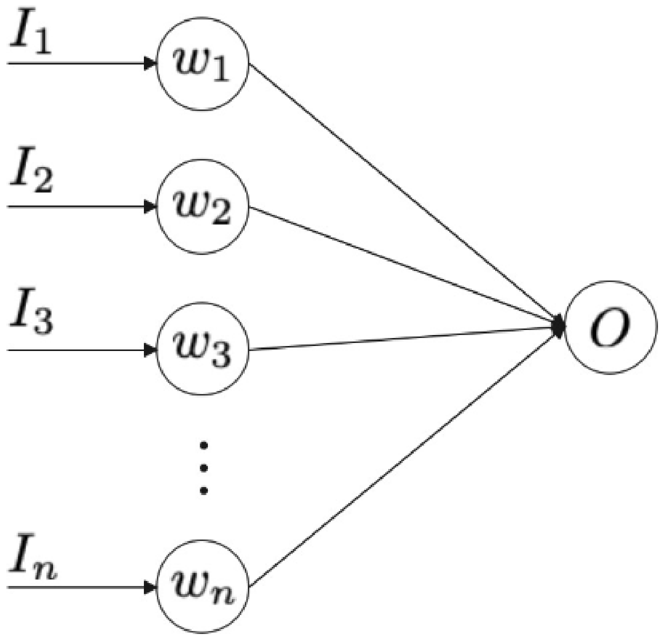


Figure 10: Effect of varying data set size.

Dropout的数学原理

$$O = \sum_{i}^{n} w_i I_i$$



$$E_N = \frac{1}{2}\left(t - \sum_{i=1}^{n} w_i' I_i\right)^2$$

$$E_N = \frac{1}{2}\left(t - \sum_{i=1}^{n} p_i w_i I_i\right)^2$$

$$\frac{\partial E_N}{\partial w_i} = -t p_i I_i + w_i p_i^2 I_i^2 + \sum_{j=1, j\neq i}^{n} w_j p_i p_j I_i I_j$$

Dropout网络的期望值是一种正则

$$E_D = \frac{1}{2}\left(t - \sum_{i=1}^{n} \delta_i w_i I_i\right)^2 \qquad \delta \sim \mathrm{Bernoulli}(p)$$

$$\frac{\partial E_D}{\partial w_i} = -t\delta_i I_i + w_i \delta_i^2 I_i^2 + \sum_{j=1, j\neq i}^{n} w_j \delta_i \delta_j I_i I_j$$

$$\begin{aligned}
E\left[\frac{\partial E_D}{\partial w_i}\right] &= -t p_i I_i + w_i p_i^2 I_i^2 + w_i \operatorname{Var}(\delta_i) I_i^2 + \sum_{j=1, j\neq i}^{n} w_j p_i p_j I_i I_j \\
&= \frac{\partial E_N}{\partial w_i} + w_i \operatorname{Var}(\delta_i) I_i^2 \\
&= \frac{\partial E_N}{\partial w_i} + w_i p_i (1 - p_i) I_i^2
\end{aligned}$$

Dropout网络的期望值是一种正则

- 最小化Dropout网络的损失等价于最小化带正则项的网络。

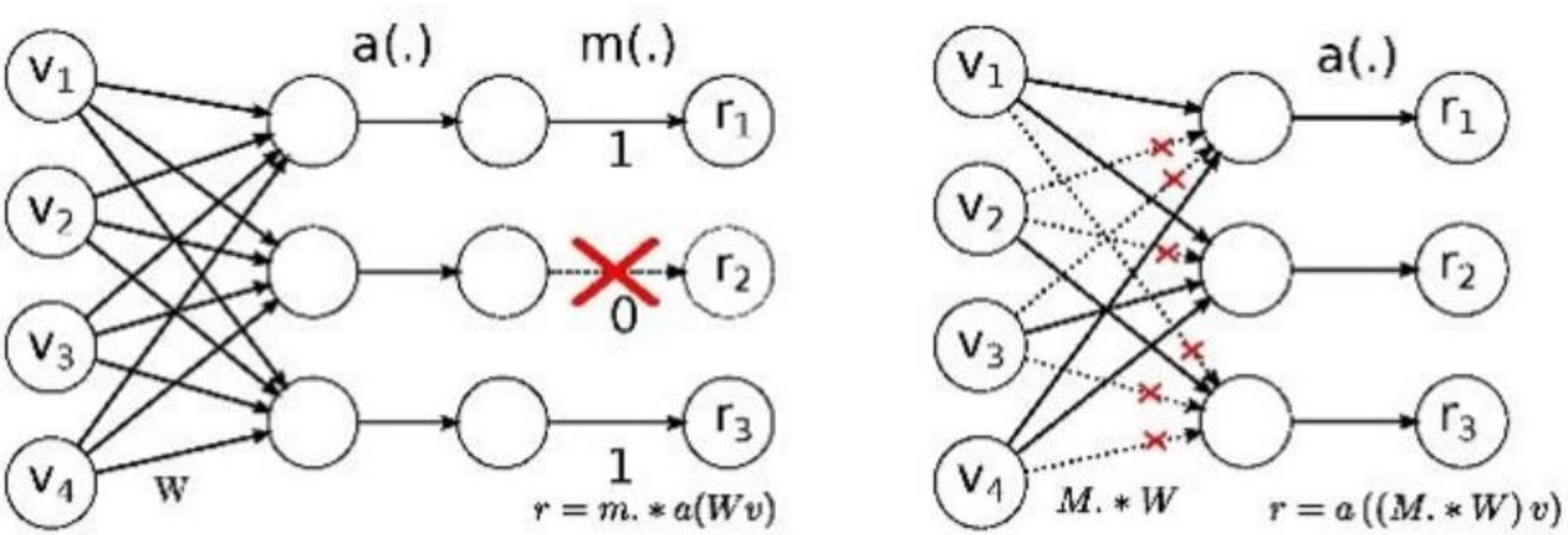
$$E_R = \frac{1}{2}\left(t - \sum_{i=1}^{n} p_i w_i I_i\right)^2 + \sum_{i=1}^{n} p_i \left(1 - p_i\right) w_i^2 I_i^2$$

- P = 0.5, Dropout的正则最强

- P选择策略
  - 浅层网络
  - 深层网络

DropConnect网络

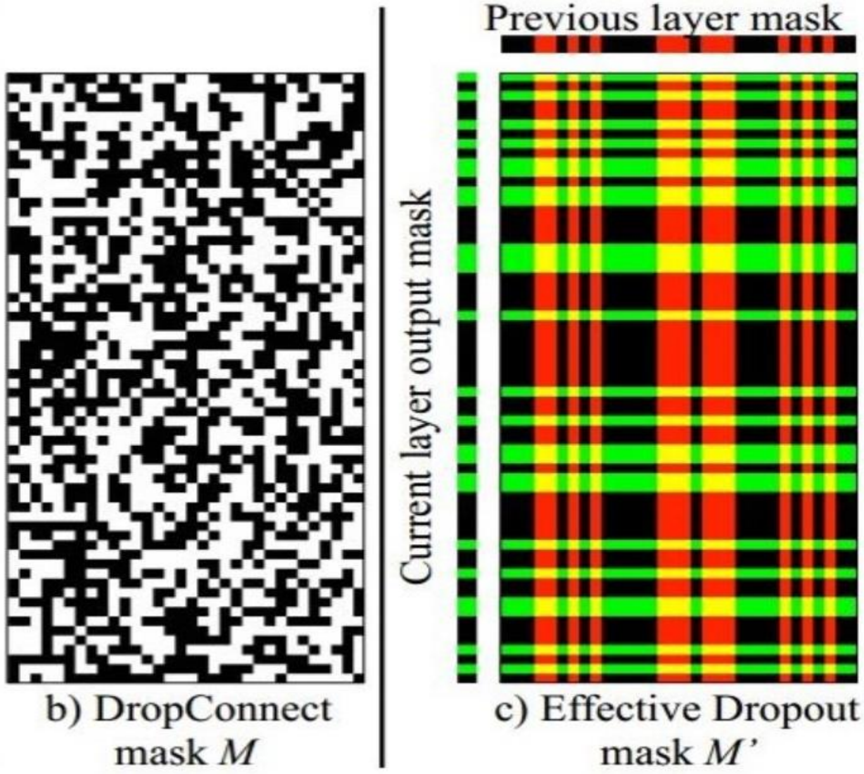
- Dropout是将输出随机置0，而DropConnect是将权重随机置0。



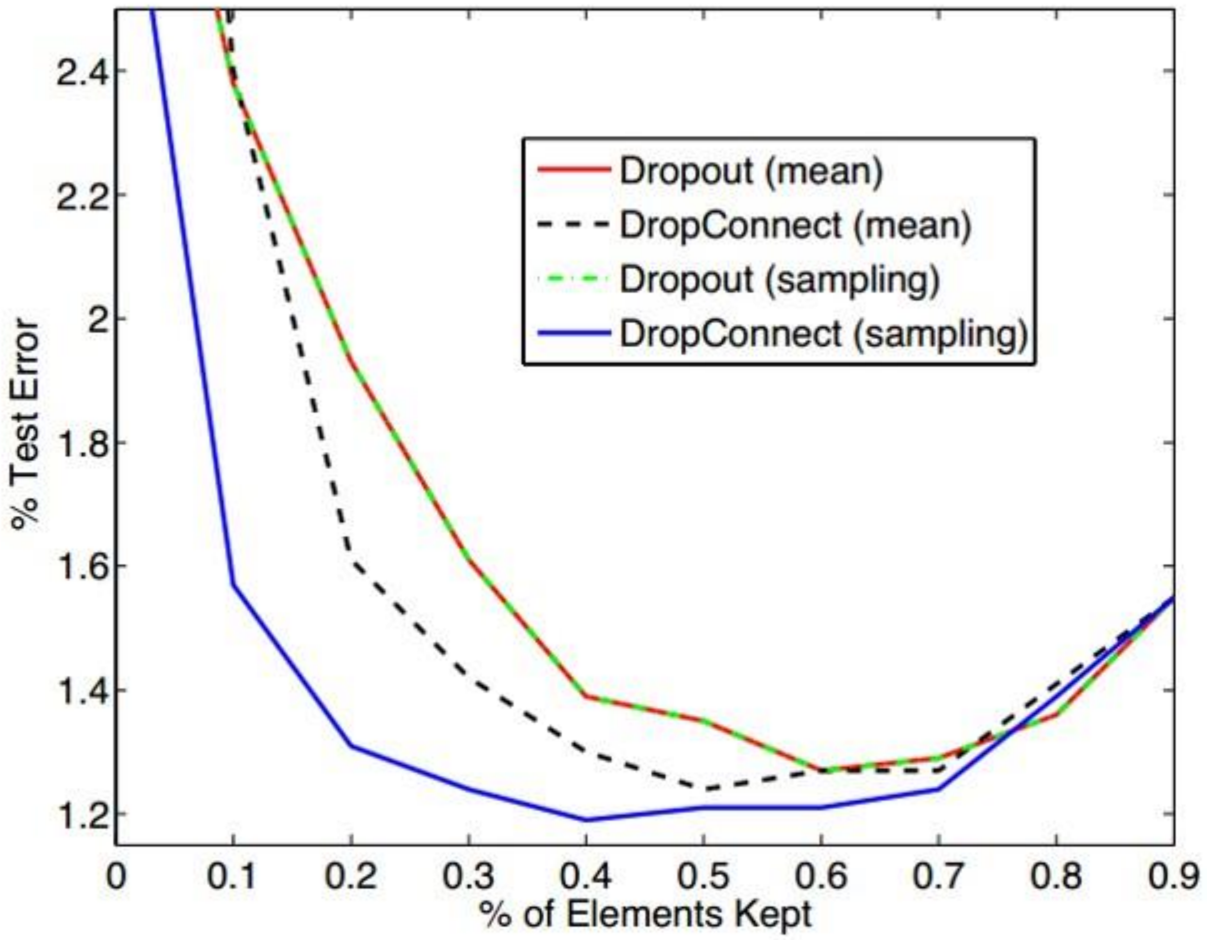
DropOut Network　　DropConnect Network

DropConnect网络的动机

- Dropout仅对输出进行随机置0，因此对掩码相当于随机的行和列进行置0。
- DropConnect由于直接对权重随机设置0，因此其掩码显得更加具有随机性。



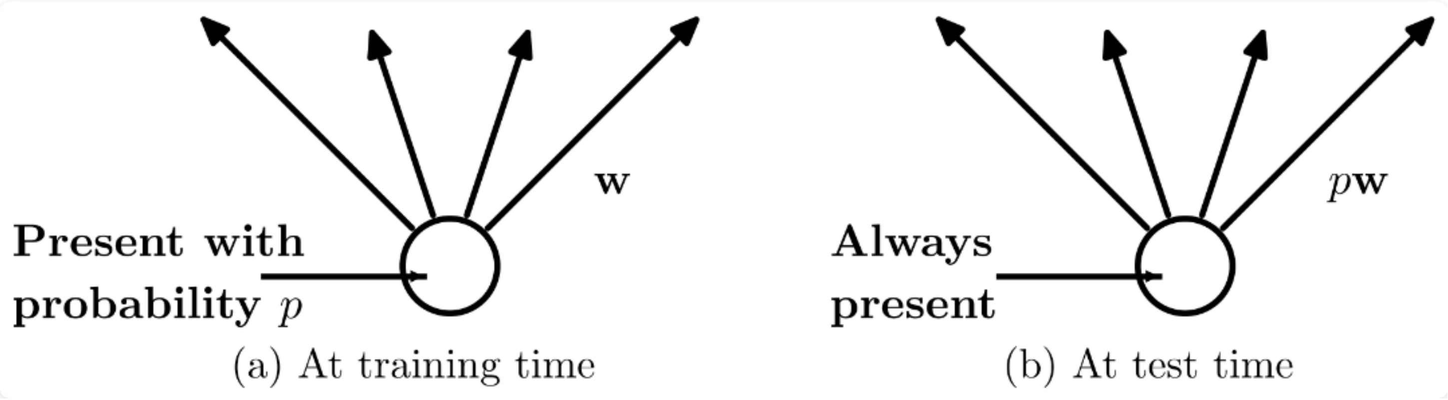
实验对比



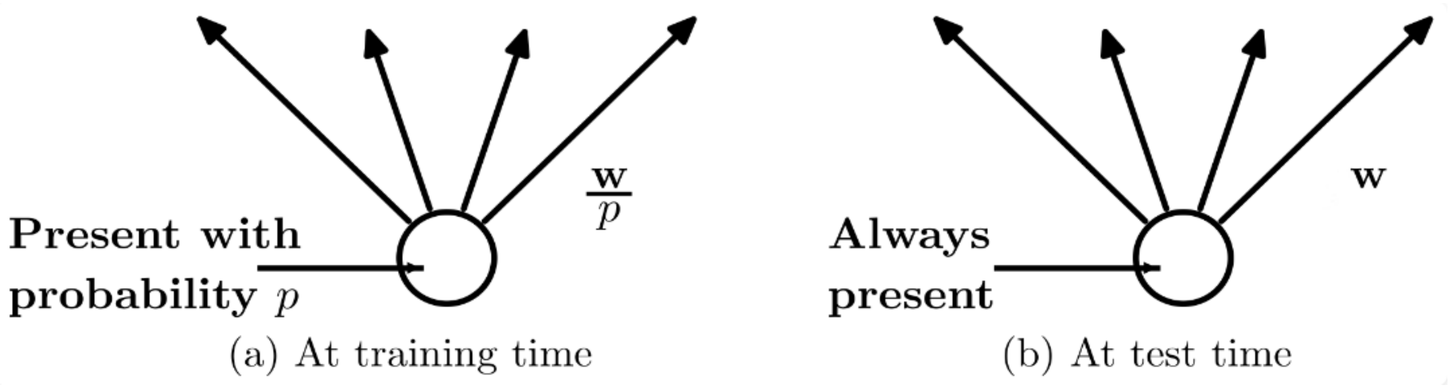
Dropout (mean)
DropConnect (mean)
Dropout (sampling)
DropConnect (sampling)
% Test Error
% of Elements Kept
2.4
2.2
2
1.8
1.6
1.4
1.2
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9

实现方式

- 原始实现：



- 目前实现：



实现原理

$$\min_{w} ||y - Xw||^2$$

$$\min_{w} ||y - p_{keep}Xw||^2 + p_{keep}(1 - p_{keep})||\Gamma w||^2$$

$$\min_{w} ||y - p_{keep}X\frac{w}{p_{keep}}||^2 + p_{keep}(1 - p_{keep})||\Gamma \frac{w}{p_{keep}}||^2$$

$$\min_{\frac{w}{p_{keep}}} ||y - p_{keep}X\frac{w}{p_{keep}}||^2 + p_{keep}(1 - p_{keep})||\Gamma \frac{w}{p_{keep}}||^2$$

函数实现

```python
import numpy as np

"""
discard_prob : 每个神经元被丢弃的概率，计算时要换成keep_prob
"""
def dropout(x, discard_prob=0.5, seed=None):
    if discard_prob < 0 or discard_prob > 1:
        raise ValueError('Dropout prob must be in interval [0,1].')

    keep_prob = 1 - discard_prob

    seed = np.random.seed(seed)

    random_tensor = np.random.binomial(n=1, p=keep_prob, size=x.shape)

    x *= random_tensor

    x /= keep_prob

    return x
```

总结

- **模型平均：Dropout是模型平均的一种**
  - **对于每次输入的样本，其对应的网络结构都是不同的，取平均会让过拟合和欠拟合结果互相抵消。**
  - **Dropout策略使得网络共享权重，因此训练测试时间代价低。**

- **减少单元共适应（co-adaptions）**
  - **Dropout策略使得每个单元不一定每次训练都在同一个网络，从而可以解耦原先单元们之间的以来关系，提高网络的鲁棒性。**