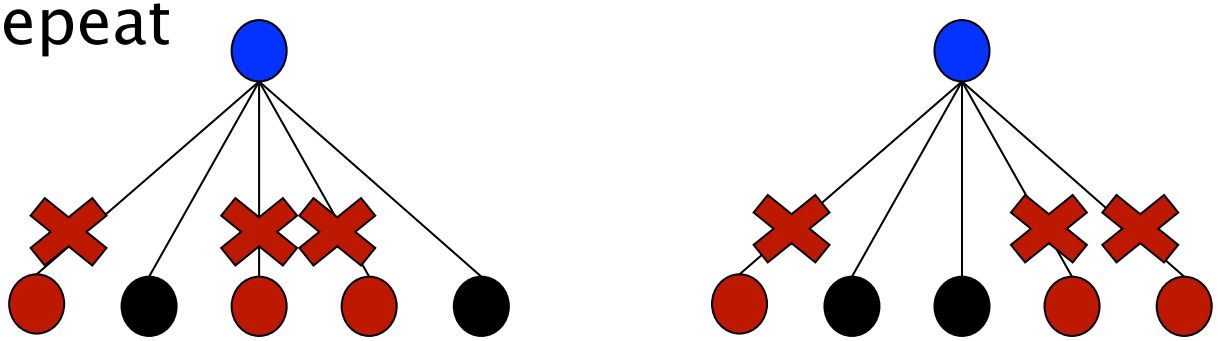
Fast dropout training

ICML 2013
Sida Wang, Chris Manning

What is dropout training

- Introduced by Hinton et al. in “Improving neural networks by preventing co-adaptation of feature detectors”
- Randomly select some inputs for each unit
  - zero them
  - compute the gradient, make an update
  - repeat



Dropout training is promising

- Won the ImageNet challenge by a margin
- George Dahl et al. won the Merck challenge
- Dropout seems to be an important ingredient

In this work…

- Sampling is inefficient – we integrate
  - 50% random dropout, after 5 passes of the data, $1/32 \approx 3\%$ of the data is still unseen

- Dropout as Bayesian model selection, equivalence to making the weights random

CLT and the Gaussian approximation

- Sample from the matching Gaussian

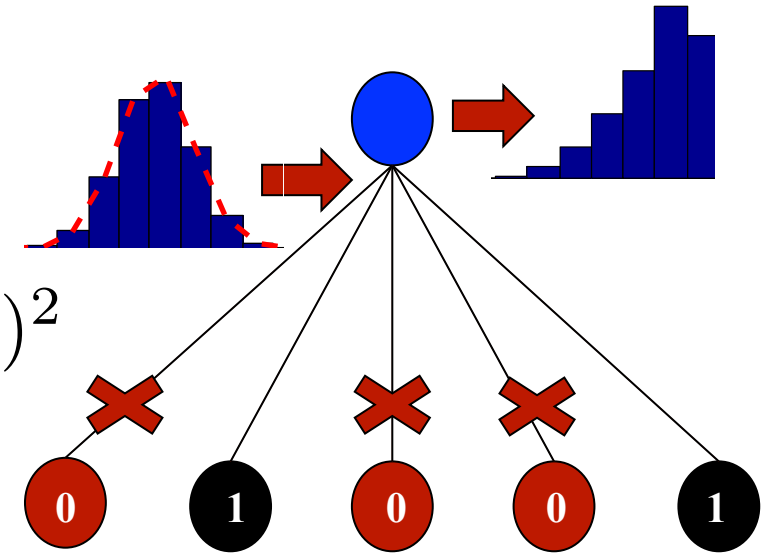
$$Y(z) = w^t D_z x = \sum_i^m w_i x_i z_i$$

$$S = E_z[Y(z)] + \sqrt{\mathrm{Var}[Y(z)]}\epsilon$$

$$\epsilon \sim \mathcal{N}(0,1),$$

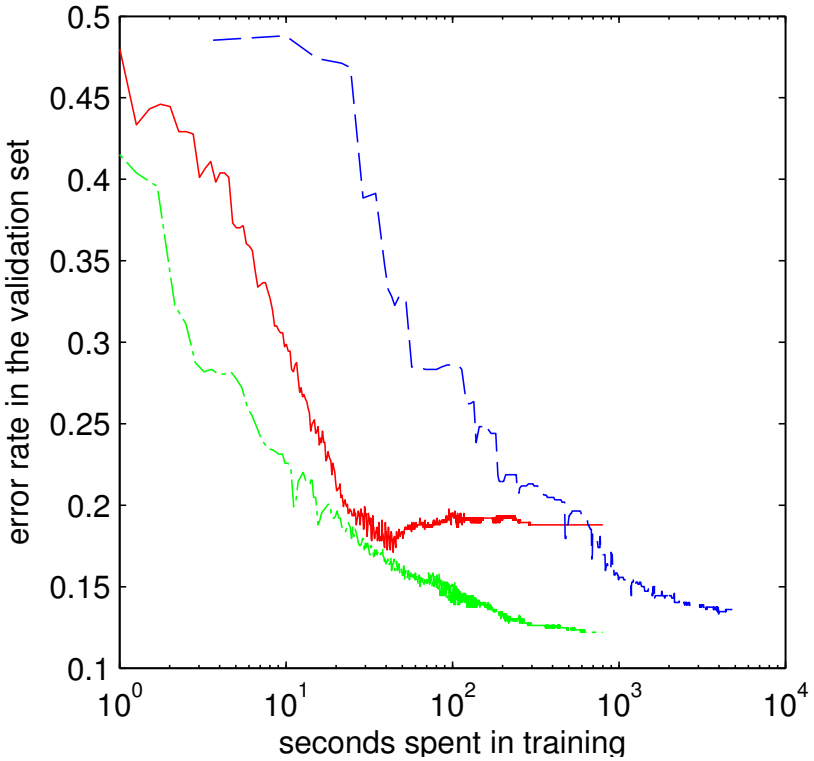
$$E_z[Y(z)] = \sum_i^m p_i w_i x_i$$

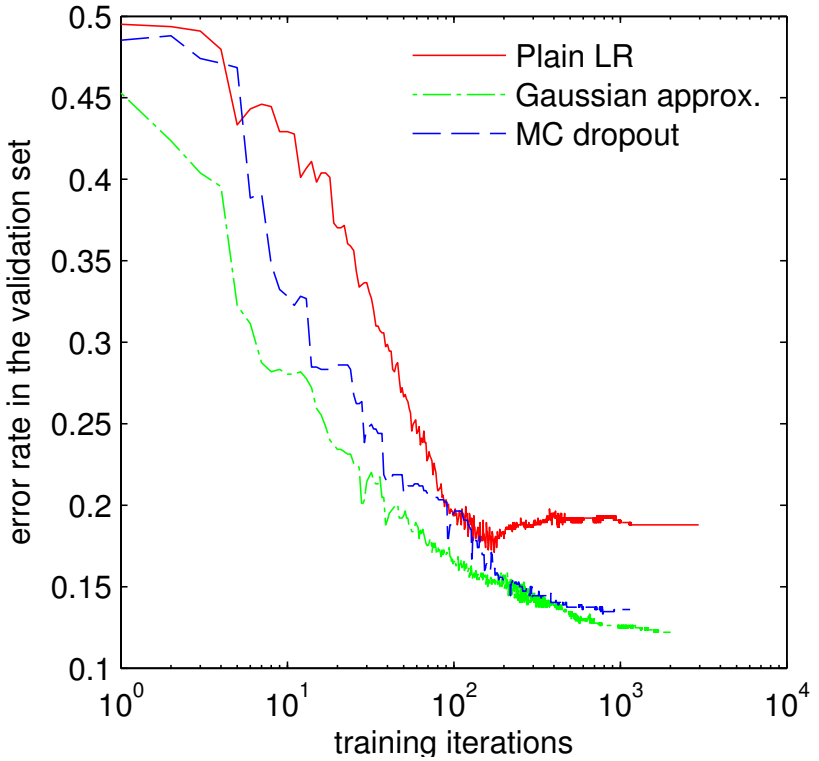
$$\mathrm{Var}\,[Y(z)] = \sum_i^m p_i(1-p_i)(w_i x_i)^2$$



This method works well

- Reduces complexity from O(Md) to O(M+d) for M samples and data dimension d.

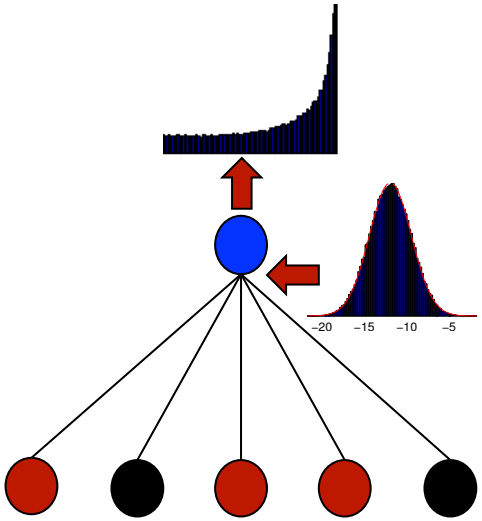


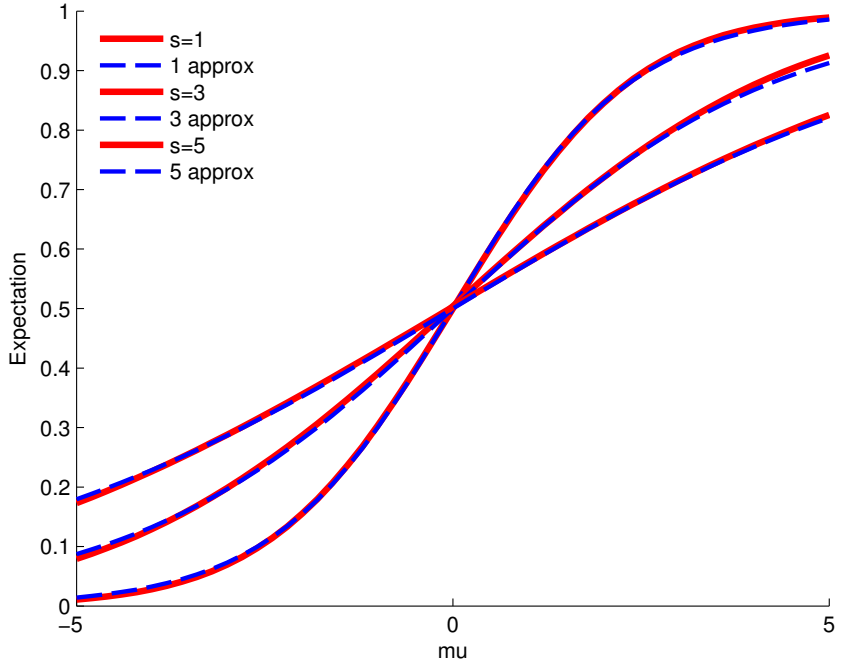


Integrating everything

- We need to compute several expectations

$$\int_{-\infty}^{\infty} \sigma(x)\mathcal{N}(x|\mu, s^2)dx \approx \sigma\left(\frac{\mu}{\sqrt{1+\pi s^2/8}}\right)$$





The objective function

- For LR, the objective is:

$$E_{Y \sim \mathcal{N}(\mu, s^2)}[\log(\sigma(Y))] = \int_{-\infty}^{\infty} \log(\sigma(x)) \mathcal{N}(x|\mu, s^2) dx$$
$$\approx \sqrt{1 + \pi s^2/8} \log \sigma\left(\frac{\mu}{\sqrt{1 + \pi s^2/8}}\right)$$

- Other losses can be integrated in a similar way

Document classification results

- Improvements over plain LR and most previous methods. Sometimes state-of-the-art.

- Average accuracy and time on 7 datasets:

Methods	Plain LR	Dropout	Gaussian	Determ.
Accuracy	84.97	86.88	86.99	86.87
Time	92	2661	325	120

Relation to Bayesian model selection – random weights

- We are in effect maximizing a lower bound on the Bayesian evidence:

$$M_\mu = \int p(\mathcal{D}|w)p(w|\mu)dw$$

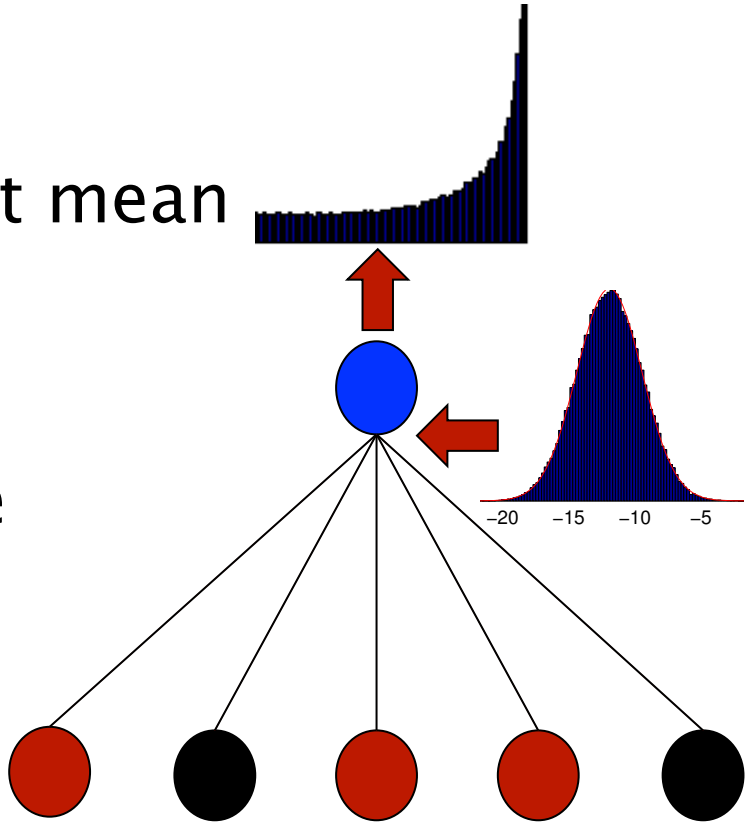
- Under the model $p(w_i|\mu_i) = \mathcal{N}(w_i|\mu_i, \alpha\mu_i^2)$

$$p(y|x,w) = \sigma(yw^T x)$$

$$\begin{aligned}
L(w) &= E_{z;z_i\sim\text{Bernoulli}(p_i)}[\log p(y|w^T D_z x)] \\
&\approx E_{Y\sim\mathcal{N}(E[w^T D_z x],\text{Var}[w^T D_z x])}[\log p(y|Y)] \\
&= E_{v:v_i\sim\mathcal{N}(\mu_i,\alpha\mu_i^2)}[\log p(y|v^T x)] \\
&\leq \log E_{v:v_i\sim\mathcal{N}(\mu_i,\alpha\mu_i^2)}[p(y|v^T x)] \\
&= \log(M_\mu)
\end{aligned}$$

Fast dropout in neural networks

- The fast dropout idea can be applied to neural networks
- Each hidden unit has an input mean and an input variance
- Outputs a mean and variance



Applies to different types of units

- Sigmoid unit:

$$\nu = \int_{-\infty}^{\infty} \sigma(x)\mathcal{N}(x|\mu, s^2)dx \approx \sigma\left(\frac{\mu}{\sqrt{1+\pi s^2/8}}\right)$$

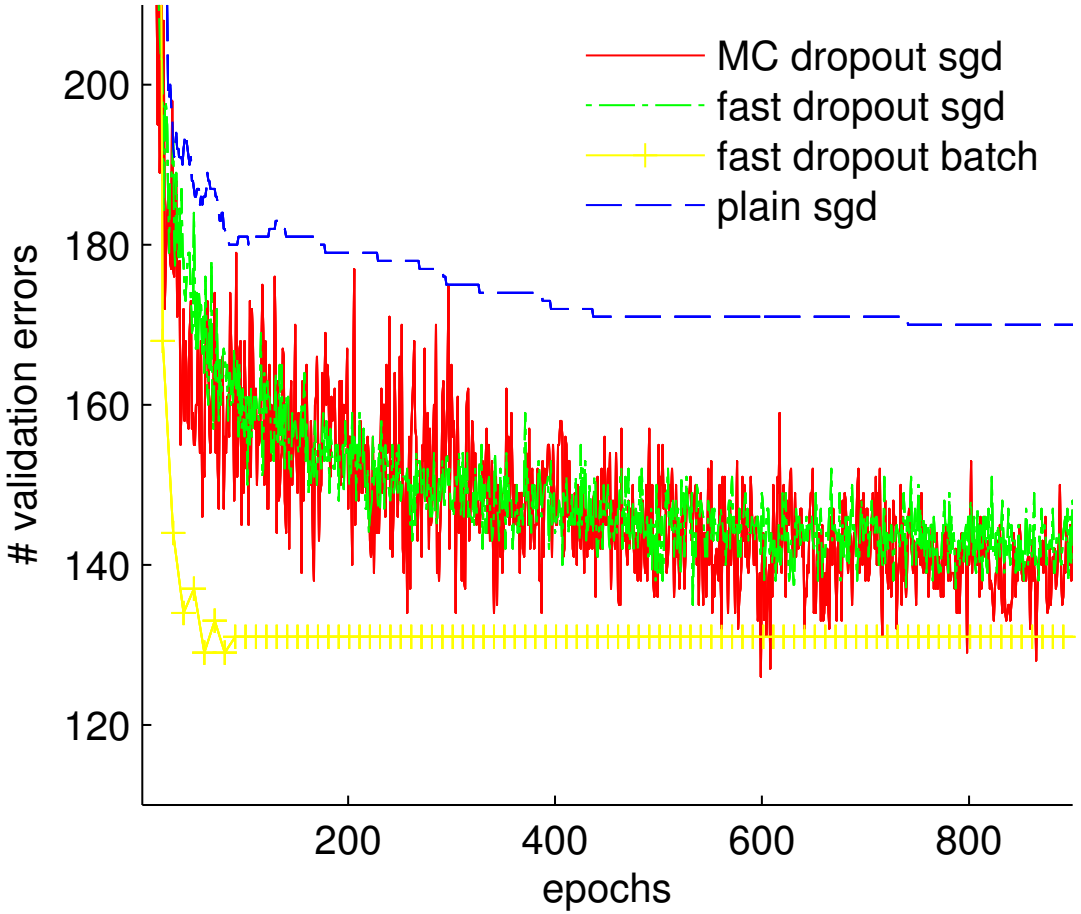
- Rectified linear unit: $f(x) = \max(0, x)$ $\quad r = \mu/s$

$$\nu = \int_{-\infty}^{\infty} f(x)\mathcal{N}(x|\mu, s^2)dx = \Phi(r)\mu + s\mathcal{N}(r|0, 1)$$

- And other objective functions: regression, MaxOut

Neural networks training curve

- Validation errors vs. training iterations



Performance on MNIST

- 2-layer neural networks on MNIST

Method name	Number of errors
NN-1200-1200 plain	182
NN-1200-1200 det. Dropout	134
NN-1200-1200 det. Dropout + Var	110
NN-300 MSE [LeCun 1998]	360
NN-800 [Simard 2003]	160
Real dropout [Hinton 2012]	105-120 79 with pretraining

Conclusions

- Dropout training seems promising
- But doing real dropout is slow, sampling is expensive
- Apply the Gaussian approximation
- Cheaper samples or completely deterministic approximations!

Related Recent Works

- Part of this work appeared in NIPS 2012 workshops
- Learning with Marginalized Corrupted Features [van der Maaten et al., 2013]
- DropConnect [Wan et al., 2013]