

Regularization

EE 5179: Deep Learning for Imaging

Instructor: Kaushik Mitra

Generalization

The central challenge of machine learning is to perform well on the - *unseen test* data, not just the *training data*

While training the model

Train err $\frac{1}{m^{(\text{train})}} \|X^{(\text{train})}w - y^{(\text{train})}\|_2^2,$

What we actually want

**Test err (or)
Generalization err** $\frac{1}{m^{(\text{test})}} \|X^{(\text{test})}w - y^{(\text{test})}\|_2^2.$

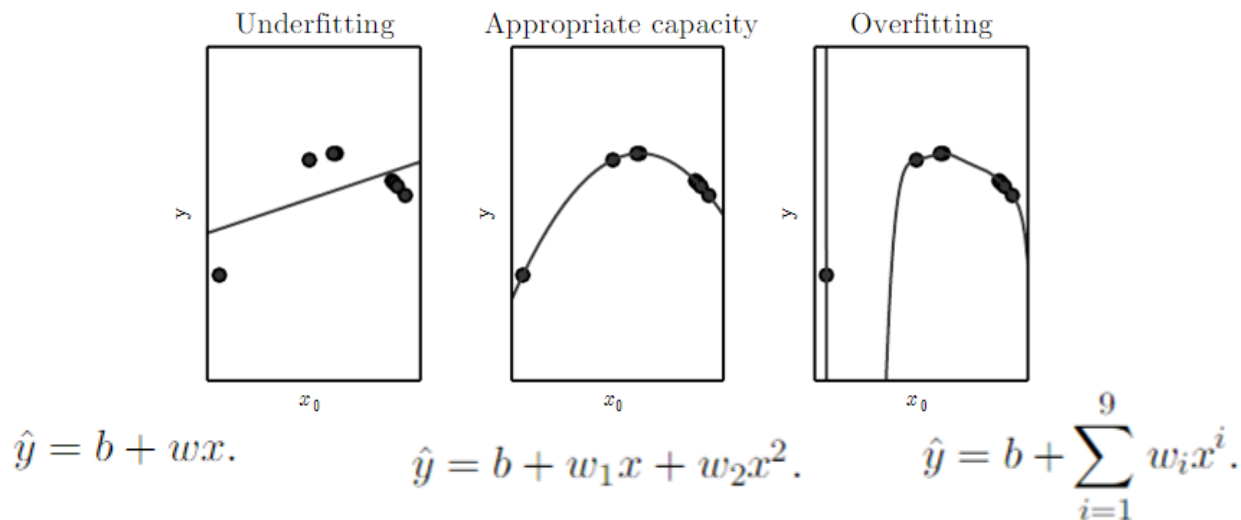
How can we say something about the test data by only seeing the train data?

Statistical learning theory

- Training and Test sets are not arbitrary
- Underlying *data generating distribution* is same

Capacity, Overfitting and Underfitting

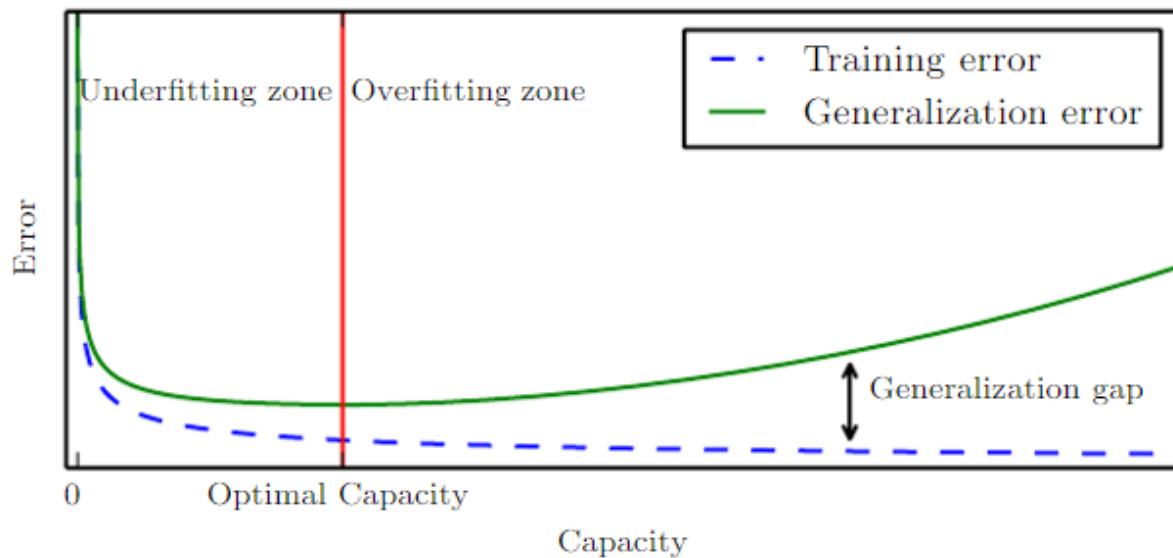
The central challenge of machine learning is to perform well on the *unseen test* data, not just the *training data*



Occam's razor: This principle states that among competing hypotheses that explain known observations equally well, one should choose the “simplest” one.

Capacity, Overfitting and Underfitting

The central challenge of machine learning is to perform well on the *unseen test* data, not just the *training data*

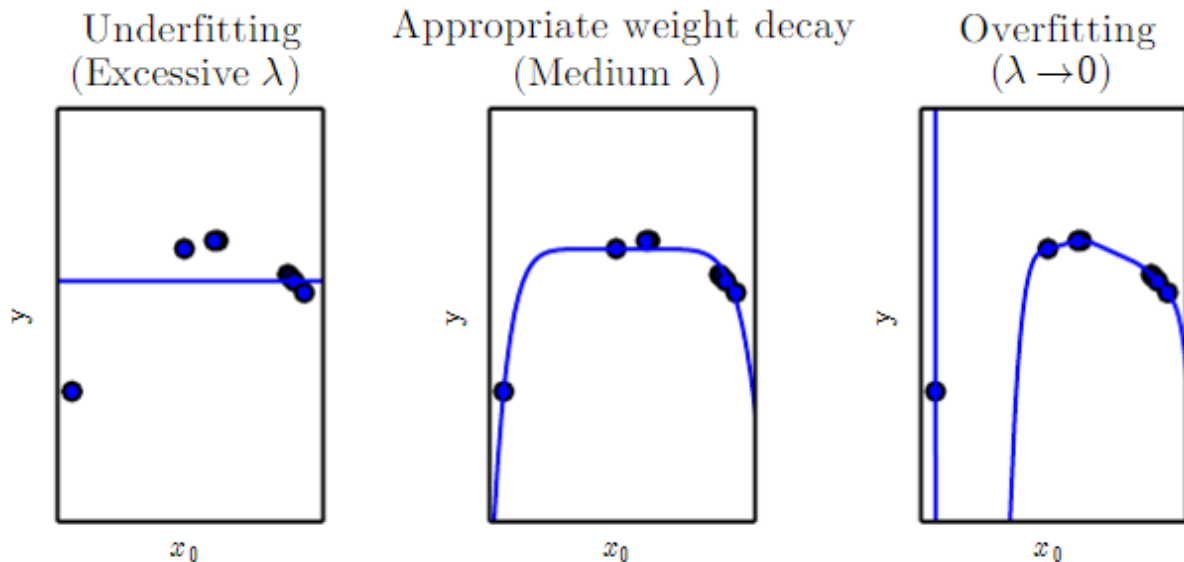


Regularization

Regularization is any modification we make to a learning algorithm that is intended to reduce its generalization error but not its training error

$$J(\mathbf{w}) = \text{MSE}_{\text{train}} + \lambda \mathbf{w}^\top \mathbf{w},$$

Note that *biases* in general are not penalized



Regularization - parameter norm penalties

L_2 norm regularization

(weight decay, ridge regression)

$$\tilde{J}(\theta; X, y) = J(\theta; X, y) + \alpha \Omega(\theta)$$

$$\tilde{J}(w; X, y) = \frac{\alpha}{2} w^\top w + J(w; X, y),$$

Parameter update:

$$\nabla_w \tilde{J}(w; X, y) = \alpha w + \nabla_w J(w; X, y).$$

$$w \leftarrow w - \epsilon (\alpha w + \nabla_w J(w; X, y)).$$

$$w \leftarrow (1 - \epsilon \alpha) w - \epsilon \nabla_w J(w; X, y).$$

Regularization - parameter norm penalties

L_2 norm regularization $\tilde{J}(w; X, y) = \frac{\alpha}{2} w^\top w + J(w; X, y),$

Consider:

$w^* = \arg \min_w J(w).$ *Unregularized solution*

- Taylor expansion of J at w^* ,

$$\hat{J}(\theta) = J(w^*) + \frac{1}{2} (w - w^*)^\top H (w - w^*),$$

H is hessian of J wrt w at w^* ; **linear term?**

$$\nabla_w \hat{J}(w) = H(w - w^*)$$

- Add the L_2 regularization grad

$$\alpha \tilde{w} + H(\tilde{w} - w^*) = 0$$

$$(H + \alpha I) \tilde{w} = H w^*$$

$$\tilde{w} = (H + \alpha I)^{-1} H w^*.$$

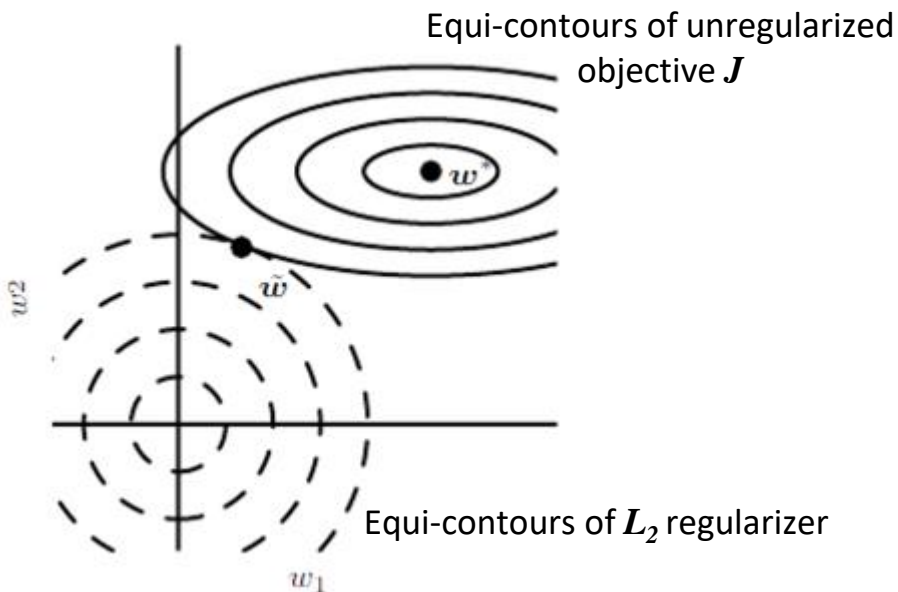
- Since H is real and symmetric

$$\begin{aligned} \tilde{w} &= (Q \Lambda Q^\top + \alpha I)^{-1} Q \Lambda Q^\top w^* \\ &= \left[Q (\Lambda + \alpha I) Q^\top \right]^{-1} Q \Lambda Q^\top w^* \\ &= Q \underbrace{(\Lambda + \alpha I)^{-1} \Lambda}_{\lambda_i / (\lambda_i + \alpha)} Q^\top w^*. \end{aligned}$$

Regularization - parameter norm penalties

L_2 norm regularization

$$\tilde{J}(w; X, y) = \frac{\alpha}{2} w^\top w + J(w; X, y),$$



- Since H is real and symmetric

$$\begin{aligned}\tilde{w} &= (Q\Lambda Q^\top + \alpha I)^{-1} Q\Lambda Q^\top w^* \\ &= \left[Q(\Lambda + \alpha I)Q^\top \right]^{-1} Q\Lambda Q^\top w^* \\ &= Q \underbrace{(\Lambda + \alpha I)^{-1} \Lambda}_{\lambda_i / (\lambda_i + \alpha)} Q^\top w^*.\end{aligned}$$

Regularizer effectively **rescales** w^* along eigenvectors of H

- $\lambda_i \gg \alpha$, regularizer effect is relatively less
- $\lambda_i \ll \alpha$, weights shrink to zero

Regularization - parameter norm penalties

L_1 norm regularization

$$\tilde{J}(\theta; X, y) = J(\theta; X, y) + \alpha \Omega(\theta)$$

$$\Omega(\theta) = \|w\|_1 = \sum_i |w_i|,$$

$$\tilde{J}(w; X, y) = \alpha \|w\|_1 + J(w; X, y),$$

$$\nabla_w \tilde{J}(w; X, y) = \alpha \text{sign}(w) + \nabla_w J(X, y; w)$$

Regularization - parameter norm penalties

L_1 norm regularization

- Taylor expansion of J at w^* ,

$$\hat{J}(\theta) = J(w^*) + \frac{1}{2}(w - w^*)^\top H(w - w^*),$$

H is diagonal, with $h_{ii} > 0$, for all i

- Revisiting L_2 with diagonal H

$$\tilde{w}_i = \frac{H_{i,i}}{H_{i,i} + \alpha} w_i^*$$

Still it only scales the weights - **not sparsity**

- Now, the L_1 regularized objective

$$\hat{J}(\theta) = J(w^*; X, y) + \sum_i \left[\frac{1}{2} H_{i,i} (w_i - w_i^*)^2 + \alpha |w_i| \right]$$

$$w_i = \text{sign}(w_i^*) \max \left\{ |w_i^*| - \frac{\alpha}{H_{i,i}}, 0 \right\} \text{ . solution}$$

- Let $w_i^* > 0$, for all i

$$\text{ - if } w_i^* \leq \frac{\alpha}{H_{ii}} ; w_i = 0$$

$$\text{ - else } w_i^* \geq \frac{\alpha}{H_{ii}}$$

$$w_i \text{ moves towards } 0 \text{ by } \frac{\alpha}{H_{ii}}$$

L_1 regularization enforces **sparsity** in the solution

Regularization - other methods

Bagging - bootstrap aggregating

- Say k regression models
 - Say each of them makes ϵ_i error
 - Error is drawn from Multivariate normal distribution

$$\mathbb{E}[\epsilon_i^2] = v; \quad \mathbb{E}[\epsilon_i \epsilon_j] = c$$

- Avg. error by k models

$$(1/k) \sum_i \epsilon_i$$

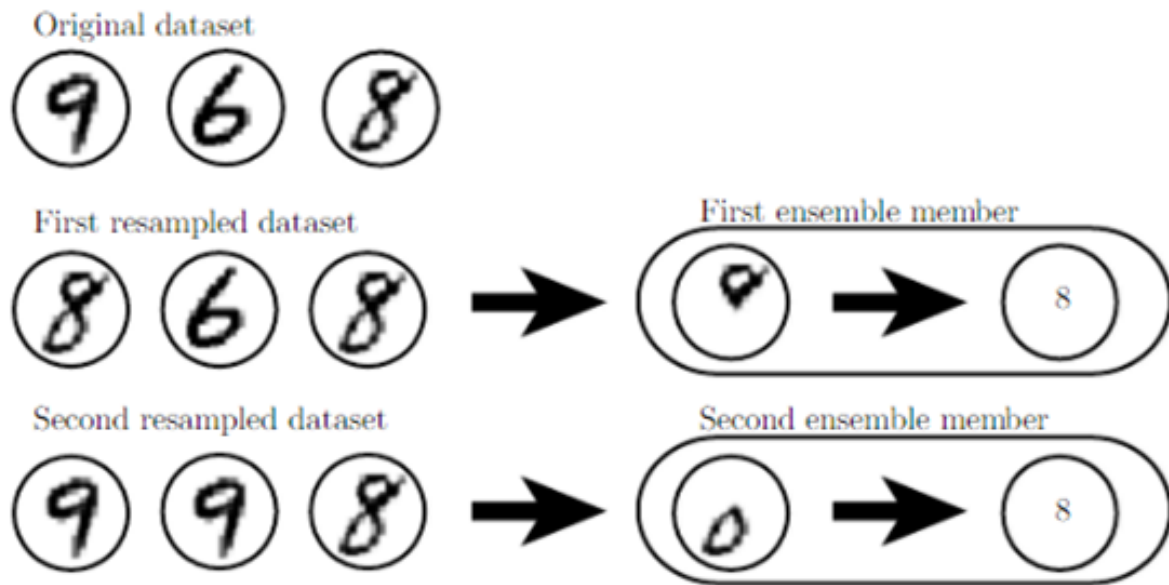
- Expected squared error of the ensemble predictor

$$\begin{aligned} \mathbb{E} \left[\left(\frac{1}{k} \sum_i \epsilon_i \right)^2 \right] &= \frac{1}{k^2} \mathbb{E} \left[\sum_i \left(\epsilon_i^2 + \sum_{j \neq i} \epsilon_i \epsilon_j \right) \right] \\ &= \frac{1}{k} v + \frac{k-1}{k} c. \end{aligned}$$

- If the models are perfectly correlated and $c = v$, error reduces to v
- If perfectly uncorrelated, $c = 0$, error reduces to v/k

Regularization - other methods

Bagging - bootstrap aggregating (Breiman, 1994)



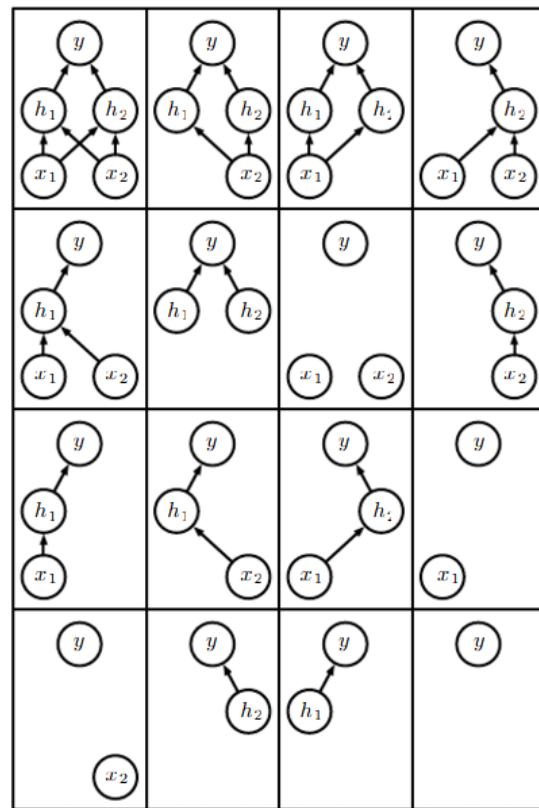
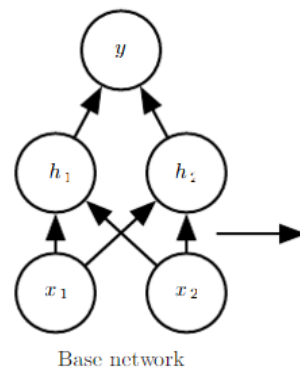
Regularization - other methods

Drop-out (Srivastava et al., 2014)

Stochastically turn the activation of the hidden unit off with a certain probability

$$h^{(k)} = f(Wh^{(k-1)} + b^{(k-1)})$$
$$\hat{h}^{(k)} = \mu^{(k)} \odot h^{(k)}$$

- How to train it?
- Is this same as bagging?



Ensemble of subnetworks

Regularization - other methods

Drop-out (Srivastava et al., 2014)

Stochastically turn the activation of the hidden unit off with a certain probability

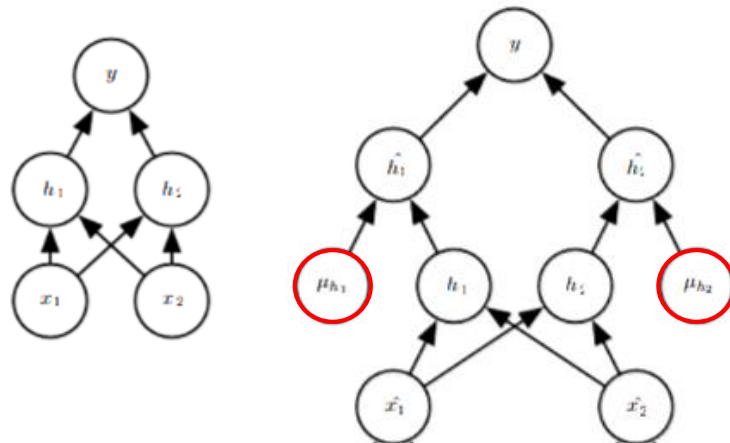
$$h^{(k)} = f(Wh^{(k-1)} + b^{(k-1)})$$

$$\hat{h}^{(k)} = \mu^{(k)} \odot h^{(k)}$$

– Inference, $p(y|x)$

Bagging

$$\frac{1}{k} \sum_{i=1}^k p^{(i)}(y | x).$$



Drop-out

$$\sum_{\mu} p(\mu) p(y | x, \mu)$$

$p(\mu)$ - Distribution used to sample μ

- Not easy to evaluate, **why?**
- Do sample averaging

Regularization - other methods

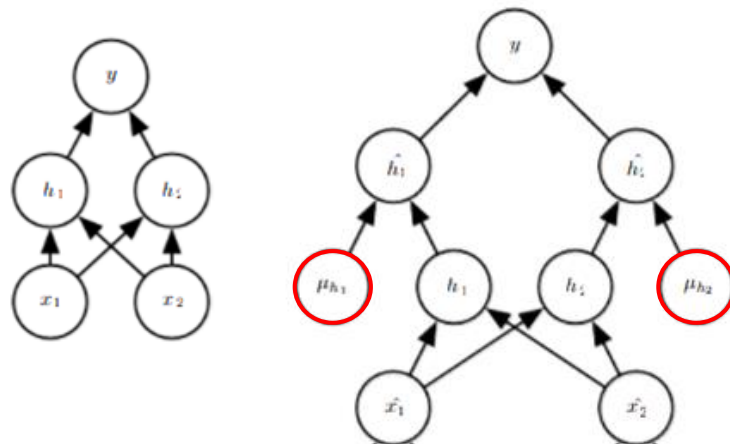
Drop-out (Srivastava et al., 2014)

Stochastically turn the activation of the hidden unit off with a certain probability

$$h^{(k)} = f(Wh^{(k-1)} + b^{(k-1)})$$

$$\hat{h}^{(k)} = \mu^{(k)} \odot h^{(k)}$$

We will look at a simple **weight scaling** result which **approximates** the **geometric mean** of models prediction in one **forward pass**



Drop-out

$$\sum_{\mu} p(\mu) p(y | x, \mu)$$

$p(\mu)$ - Distribution used to sample μ

- Not easy to evaluate, **why?**
- Do sample averaging

Regularization - other methods

Drop-out (Srivastava et al., 2014)

Stochastically turn the activation of the hidden unit off with a certain probability

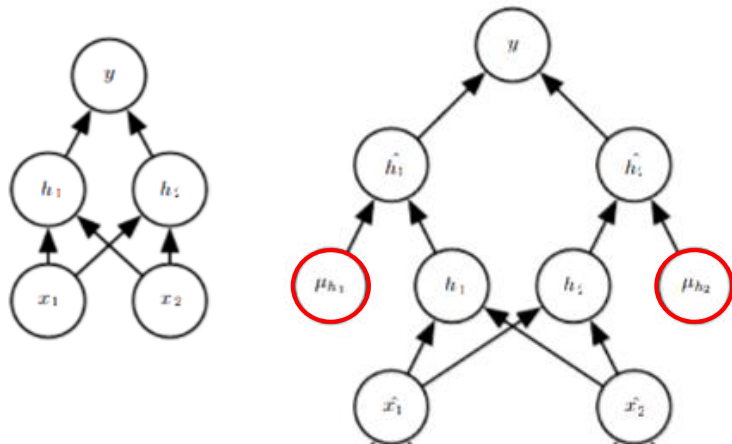
$$h^{(k)} = f(Wh^{(k-1)} + b^{(k-1)})$$

$$\hat{h}^{(k)} = \mu^{(k)} \odot h^{(k)}$$

Weight rescaling (Hinton et al., 2012)

To evaluate $p(y/x)$ with all units

- Multiply weights going out of unit i with probability of including unit i



Drop-out

$$\sum_{\mu} p(\mu) p(y | x, \mu)$$

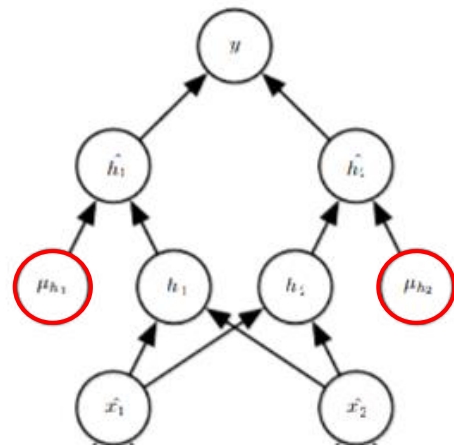
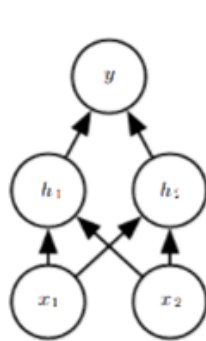
$p(\mu)$ - Distribution used to sample μ

- Not easy to evaluate, **why?**
- Do sample averaging

Regularization - other methods

Drop-out (Srivastava et al., 2014)

Stochastically turn the activation of the hidden unit off with a certain probability



unnormalized
probability

$$\tilde{p}_{\text{ensemble}}(y \mid \mathbf{x}) = \sqrt[2^d]{\prod_{\mu} p(y \mid \mathbf{x}, \mu)}$$

Uniform probability of masking

$$p_{\text{ensemble}}(y \mid \mathbf{x}) = \frac{\tilde{p}_{\text{ensemble}}(y \mid \mathbf{x})}{\sum_{y'} \tilde{p}_{\text{ensemble}}(y' \mid \mathbf{x})}.$$

Drop-out (Srivastava et al., 2014)

In case of linear hidden units, the weight scale inference is exact.

For example, consider a softmax regression classifier

$$P(y = y \mid \mathbf{v}) = \text{softmax} \left(\mathbf{W}^\top \mathbf{v} + \mathbf{b} \right)_y$$

$$P(y = y \mid \mathbf{v}; \mathbf{d}) = \text{softmax} \left(\mathbf{W}^\top (\mathbf{d} \odot \mathbf{v}) + \mathbf{b} \right)_y$$

$$\tilde{P}_{\text{ensemble}}(y = y \mid \mathbf{v}) = \sqrt[2^n]{\prod_{\mathbf{d} \in \{0,1\}^n} P(y = y \mid \mathbf{v}; \mathbf{d})}$$

$$= \sqrt[2^n]{\prod_{\mathbf{d} \in \{0,1\}^n} \text{softmax} \left(\mathbf{W} (\mathbf{d} \odot \mathbf{v}) + \mathbf{b} \right)_y}$$

$$= \sqrt[2^n]{\prod_{\mathbf{d} \in \{0,1\}^n} \frac{\exp \left(\mathbf{W}_{y,:}^\top (\mathbf{d} \odot \mathbf{v}) + b_y \right)}{\sum_{y'} \exp \left(\mathbf{W}_{y',:}^\top (\mathbf{d} \odot \mathbf{v}) + b_{y'} \right)}}$$

$$= \frac{\sqrt[2^n]{\prod_{\mathbf{d} \in \{0,1\}^n} \exp \left(\mathbf{W}_{y,:}^\top (\mathbf{d} \odot \mathbf{v}) + b_y \right)}}{\sqrt[2^n]{\prod_{\mathbf{d} \in \{0,1\}^n} \sum_{y'} \exp \left(\mathbf{W}_{y',:}^\top (\mathbf{d} \odot \mathbf{v}) + b_{y'} \right)}}$$

$$\tilde{P}_{\text{ensemble}}(y = y \mid \mathbf{v}) \propto \sqrt[2^n]{\prod_{\mathbf{d} \in \{0,1\}^n} \exp \left(\mathbf{W}_{y,:}^\top (\mathbf{d} \odot \mathbf{v}) + b_y \right)}$$

$$= \exp \left(\frac{1}{2^n} \sum_{\mathbf{d} \in \{0,1\}^n} \mathbf{W}_{y,:}^\top (\mathbf{d} \odot \mathbf{v}) + b_y \right)$$

$$= \exp \left(\underbrace{\frac{1}{2} \mathbf{W}_{y,:}^\top \mathbf{v}}_{\text{Weight rescale}} + b_y \right).$$

Weight rescale

Regularization - other methods

Dataset augmentation



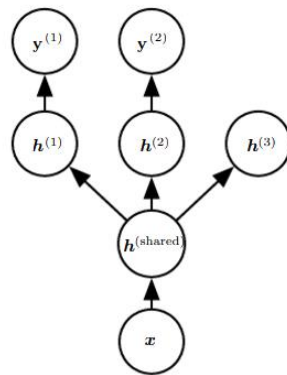
Flipping the image for classification

*pic courtesy, [web](#)

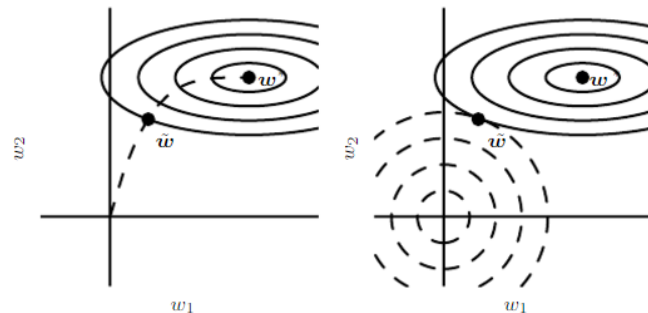
Parameter sharing and tying

Most extensively employed with Convolutional Neural Nets (CNN)

Multi-task learning



Early stopping



End