Regularization focus on minimizing testing error (generalization error)

- Ideal model is often large but regularized.

Parameter Norm Penalties

Form: $\tilde{J}(\theta, x, y) = J(\theta, x, y) + \boxed{\alpha \Omega(\theta)}$

parameter norm penalties

- L_2 regularization

Form: $\tilde{J}(w, x, y) = J(w, x, y) + \frac{1}{2}\alpha w^T \cdot w$

Discussion 1: $\nabla_w \tilde{J}(w, x, y) = \alpha w + \nabla_w J(w, x, y)$

Updating gradients $w \leftarrow w - \epsilon(\alpha w + \nabla_w J(w, x, y))$

$w \leftarrow \boxed{(1-\epsilon\alpha)w} - \epsilon \nabla_w J(w, x, y)$

let w to be a smaller value.

Discussion 2: $\hat{J}(\theta) = J(w^*) + \frac{1}{2}(w - w^*)^T H (w - w^*)$

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

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

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

$H = Q \Lambda Q^T$

$\tilde{w} = Q(\Lambda + \alpha I)^{-1} \Lambda Q^T w^*$

L_2 Regularization affects little on the larger value of H eigenvalues, which means affect the one who affects the result least.

Discussion 3: Linear Regression with L_2 regularization means adding αI to co-variance matrix XX^T

- L_1 Regularization

Form: $\tilde{J}(w, x, y) = J(w, x, y) + \alpha \|w\|_1$

$\nabla_w \tilde{J}(w, x, y) = \nabla_w J(w, x, y) + \alpha \, sgn(w)$

Feature selection: $\tilde{J}(w, x, y) = J(w^*, x, y) + \sum_i \left[\frac{1}{2} H_{i,i} (w_i - w_i^*)^2 + \alpha |w_i|\right]$

Then: $w_i = \text{sign}(w_i^*) \max\{|w_i^*| - \frac{\alpha}{H_{ii}}, 0\}$

The one who is not significantly affecting the J. will be eliminated.

L1 regularization is equal to log-prior when Bayesian. + Laplace Approx.

Norm Penalties as Constrain Optimization

Construct Langrange function for constrain optimization of Norm Penalties (Explicit constrain, instead of enforcing with penalties)

$\mathcal{L}(\theta, \alpha, X, y) = J(\theta, X, y) + \alpha(\Omega(\theta) - K) \Rightarrow$ Explicit Constrain.

- Regularization can also be seen as solution to underdetermined issues.
- Data set Augmentation: subjective
  - Application: Preprocessing (Rotate, scale)
  - General: Algorithm (Adding noise)
- Noise Robustness
  - Adding noise to input/weight: Adding additional regularization
  - Adding noise to output/target: Hard to soft.
- Semi-Supervised Learning: Learning engaged with P(x)
- Multi-task learning: Learning relative tasks using shared weights.

Early stopping

Here we regard training steps as a Hyper-parameter.

Algorithms:

- Training the model with a stopping criterion

  After j steps of validation error goes up, we stop training and we save the optimal training steps i^*

- Retrain the model with optimal i^*
  1° Training from scratch with all training data, and train exactly i^* steps.
  2° Continue training with all data, until the validation error falls down below a threshold.

Early stopping can be seen as a L2 regularizer:

- L2 Regularizer revisit:

$$Q^T\tilde{w} = (\Lambda+\alpha I)^{-1}\Lambda Q^T w^*$$

- Early stopping when $w^{(0)}=0$:

$$w^{(\tau)} = w^{(\tau-1)} - \epsilon \nabla_w \hat{J}(w^{(\tau-1)})$$

$$Q^T(w^{(\tau)} - w^*) = (I-\epsilon\Lambda)Q^T(w^{(\tau-1)} - w^*)$$

$$\downarrow w^{(0)}=0$$

$$Q^T w^{(\tau)} = [I-(I-\epsilon\Lambda)^\tau]Q^T w^*$$

Build up connections

⇒ If we Taylor expanse J, and set $w^{(0)}=0$, we see in some specific value of α and τ, we can equal a $\tilde{w}$ got by early stopping equal to a $\tilde{w}$ got by L_2 regularization.

- Parameter trying and sharing: CNN a good example.
- Sparse Representations: Force some part of h to be zero
- Bagging and other ensemble methods:

$$E\left[\left(\frac{1}{k}\sum \epsilon_i\right)^2\right] = \frac{1}{k}V + \frac{k-1}{k}C$$

Model independent Covariance between model

Dropout

Dropout can be seen as bagging with partly shared weights

Algorithms:

Training: No matter forward or backward, we randomly abandon units with a probability P_u, by multiplying units with binary variable $\mu \in \{0, 1\}$.

Testing / Inference:

$w \leftarrow w \cdot p(\text{hi})$ or $h \leftarrow \frac{1}{p(\text{hi})} \cdot h \iff$ when training $\mathbb{E}_{p(\text{hi})}(0)$

This is for alignment with the expectation of input in training.

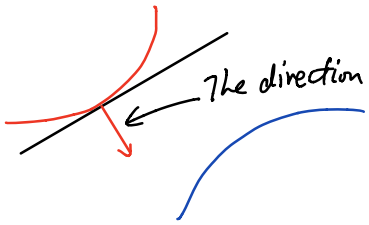
Notice:

- Non-linear model cannot prove weight scaling inference but linear model can.
- Dropout needs more complicated model and training steps.
- May be not effective when data is less or with semi-supervised learning.
- Linear Regression: Dropout = Weight decay
- Dropout in stochasticity has no guarantee to be optimal.
- μ sometimes can be any form (Continuous $N \sim (1, 1)$)
- Dropout used in Hidden Units is more common
  $\Rightarrow$ more stable / general noise: Deconstructing features.

- Adversial Training: Local Invariant
  - Semi-supervised adversial training
- Tangent Distance, Tangent Prop and Manifold Tagent Classifier

$$\Omega(f) = \sum_i \left((\nabla_x f(x))^T v_i \right)^2 \quad \text{i for each class}$$

we prefer the update direction out of the manifold of a class.



⇒
- Data Augmentation: Explicit manifold
- Relu: Linear units cannot always shrink.
- Auto-encoder: A way of finding tagent manifold

⇒ 1) Using auto-encoder to find tagent manifold
2) Using tagent regularization to train a model (tangent prop)