COEN281 -- Introduction to Pattern Recognition and Data Mining

Lecture 8: Neural Networks (Part 2)

Instructor: Dr. Giovanni Seni
GSeni@scu.edu

Department of Computer Engineering
Santa Clara University

Fall/18

Syllabus

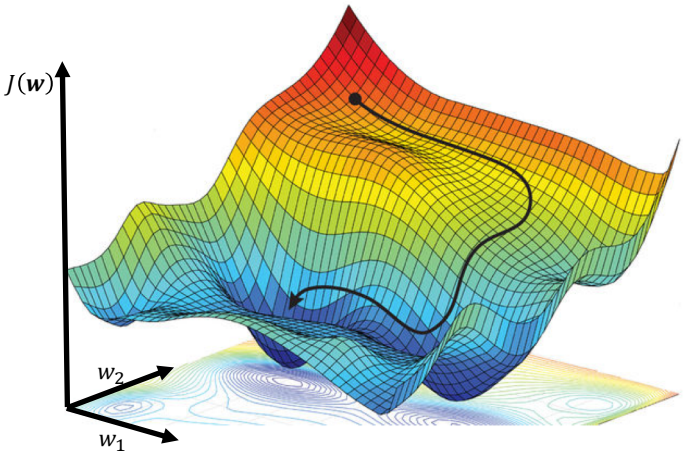
Week 1	Introduction; R (Ch.1)
Week 2	Bayesian Decision Theory (Ch.2; DHS: 2.1-2.6, 2.9) Parameter Estimation (DHS: 3.1-3.4)
Week 3	Linear Discriminant Functions (Ch.3&4; DHS: 3.8.2, 5.1-5.8) Regularization (Ch.6; SE: Ch.3)
Week 4	**Neural Networks** (DHS: 6.1-6.6, 6.8); Deep Learning
Week 5	Support Vector Machines (Ch.9)
Week 6	Decision Trees (Ch. 8.1; DHS: 8.3; Ch 2 SE)
Week 7	Ensemble Methods (Ch. 8.2; SE: Ch 4, 5)
Week 8	Clustering (Ch. 10; DHS: 10.6, 10.7) Clustering (DHS: 10.9); How many clusters are there? (DHS: 10.10)
Week 9	Non-metric: Association Rules Collaborative Filtering
Week 10	Text Retrieval; Other topics

Overview

- Error Surfaces
  - Minimums and Plateaus
- Practical Considerations
  - Momentum
  - Scaling input
  - Target values
  - Number of hidden units
  - Learning rate
  - Minibatch

Error Surfaces

Minimas



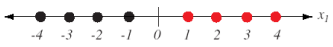
- Single global min exists; nice but not essential!

Error Surfaces

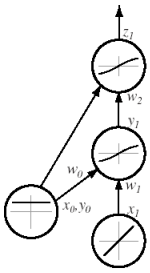
Plateaus

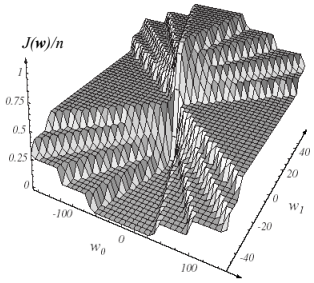
Plateaus – regions where weight change does not lead to a change in error

- Training can be slow
- Example: separable 2-class problem:



  - At $\mathbf{w} \approx \mathbf{0}$ error is high and slope is large





Practical Considerations

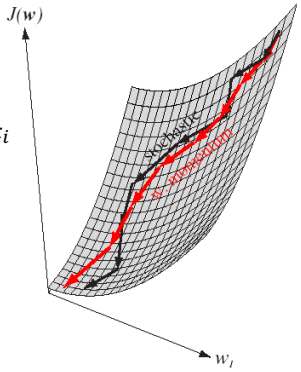
Momentum

- At plateaus, $J'(\mathbf{w}) \approx 0$ and learning can be very slow
- Physics' notion: keep moving unless acted upon by outside forces
- Modified learning rule ($\alpha \approx 0.9$):

$$w_{ji} = w_{ji} + \Delta w \quad \text{where } \Delta w = -\eta \nabla J(w) = \eta \delta_j x_i$$

$$\Downarrow$$

$$\Delta w_t = \alpha \cdot \Delta w_{t-1} + (1 - \alpha) \cdot \eta \nabla J(w)$$



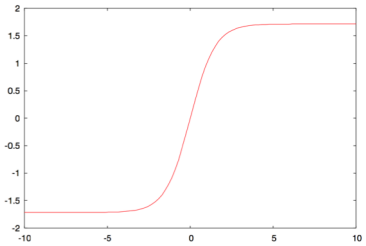
Practical Considerations
Scaling Input

- If one attribute is much larger than another attribute, the weights will be adjusted to represent such differences; that is not desirable
  - Example: mass (grams) and length (meters) of a fish
- Solution:
  - Standardize all features before training
  - E.g., "whitening" transform: $x_{new} = (x - \mu)/\sigma$
    - The mean of each feature is set to zero and the variance to 1.0
  - Also, convergence is usually faster if the average of each input variable over the training set is close to zero

Practical Considerations
Target Values

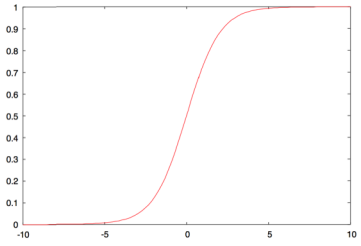
- *One-of-c* representation

$t(x) = [-1 \cdots +1 \cdots -1]$



$$f(net) = 1.716 \times \tanh(\frac{2}{3} \times net)$$

$t(x) = [0.1 \cdots 0.9 \cdots 0.1]$

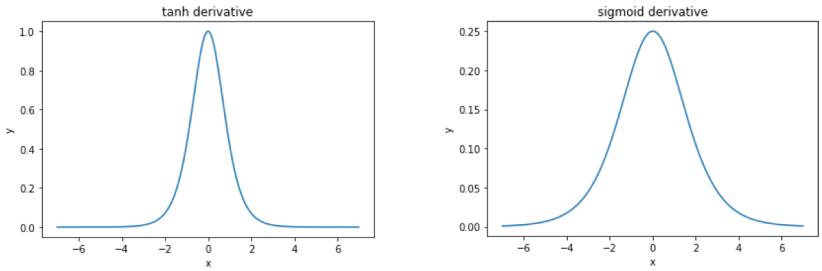


$$f(net) = \frac{1}{1+e^{-net}}$$

- Saturation values should be avoided

Practical Considerations
Tanh vs Logistic

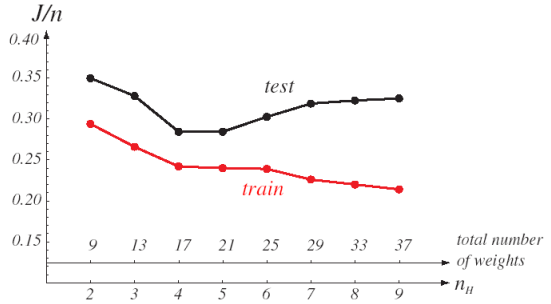
- *tanh* has larger derivatives



  - 0-centered output is also desirable (see scaling)

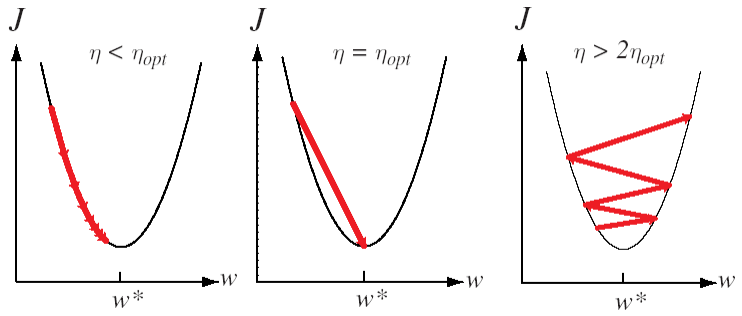
Practical Considerations
Number of Hidden Units

- n_H governs the expressive power of the net
  - Number of weights ≈ number of degrees of freedom
    - Should never have more weights than training patterns!
    - Rule of thumb: n/10



Practical Considerations
Learning Rate

- In principle, if small enough (e.g., $\eta \approx 0.001$), it only determines speed of convergence



- A principled method exists for rapid and uniform learning

$$\eta(w_k) = 1/J''(w_k)$$

Practical Considerations
Optimal Learning Rate

- 1-Dimensional Quadratic Example: $J(w) = aw^2 + bw + c$

$$\Longrightarrow J'(w) = 2aw + b \qquad J'(w) = 0 \implies 2aw + b = 0$$
$$J''(w) = 2a \qquad \hat{w} = -b/2a$$

- Now imagine that we are starting at some arbitrary initial point w_0 and we want to get to $\hat{w}$
  - Step $\Delta w = -\eta J'(w)$
  - Best η: get to $\hat{w}$ in one step – i.e., $\Delta w = \hat{w} - w_0$

$$\implies -\frac{b}{2a} - w_0 = \frac{2aw_0 + b}{2a} = \eta \cdot (2aw_0 + b)$$

$$\implies \eta = \frac{1}{2a} = \frac{1}{J''(w_0)}$$

Practical Considerations
Learning Rate Decay

- Constant

  (requires schedule for piecewise constant, tricky)

- Useful hack

  Keep learning rate constant until no improvement on validation data, then reduce rate.

- Polynomial decay

  Canonical choice for convex solvers $\eta(t) = \frac{\alpha}{\sqrt{\beta+t}}$

- Exponential decay

  not recommended

Practical Considerations
Learning Rate Decay (2)

- (locally) Adaptive

$$\eta_{ij}(t) = \frac{\eta_0}{\sqrt{K + \sum_{t'=t-\tau}^{t} J'^{2}_{ij}(t')}}$$

  – For directions with large gradient, decrease learning rate aggressively to avoid instability

  – If gradients start vanishing, learning rate decrease reduces, too

† Adaptive Subgradient Methods for Online Learning and Stochastic Optimization
John Duchi, Elad Hazan, Yoram Singer; 2011.

Practical Considerations

Minibatch

- The frequent updates of Stochastic Gradient Descent (SGD) can result in a noisy gradient signal
  - Weights can jump around ==> hard to settle on an error minimum

Aggregate gradients over a few instances before updating

- Reduces variance in gradients
- Better for vectorization (GPUs)
- Large minibatch may need lots of memory (and may slow updates)
- GPU considerations
  - 256GPUs and 20 samples/GPU ==> mini batch of 5120