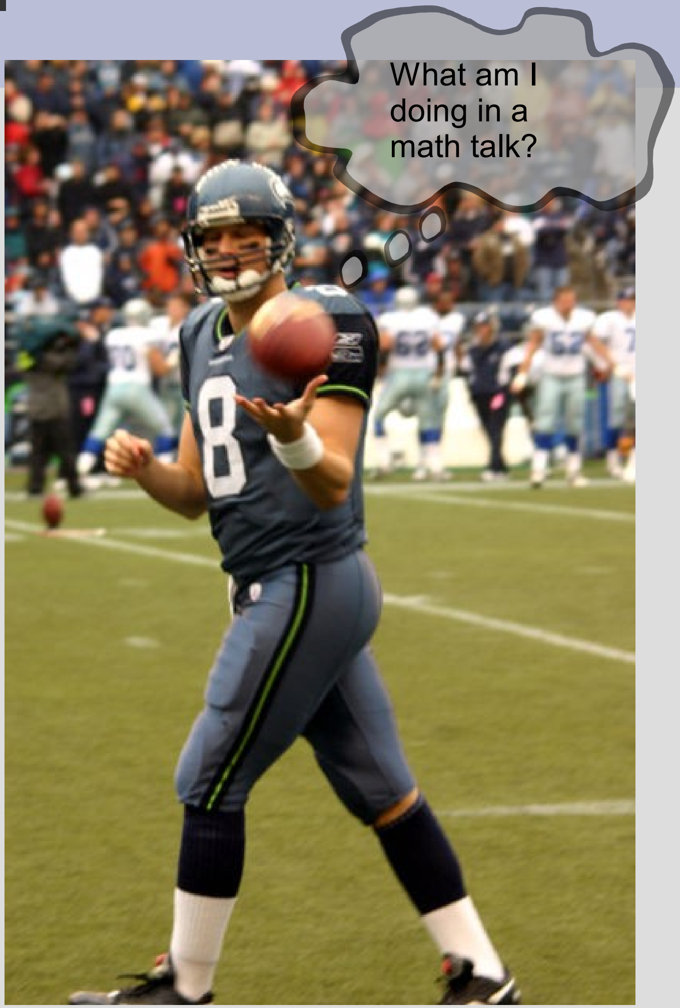
Automatic Relevance Detection And Bayesian Neural Networks

Christopher Kain

Automatic Relevance Detection

- Gain insight into the meaning of network!
- Figure out what inputs are important!
- Prevent over-fitting of data!



An instance where relevance detection would have been useful

How Do We Get There?

- Bayesian Neural Networks
  - Bayesian Learning
- Automatic Relevance Detection

Bayesian Learning

$$P(\theta|D) = \frac{P(D|\theta)P(\theta)}{P(D)}$$

- Generate a model, *P(θ),* called the prior
- Observe dataset *D*
- Find probability of the model given the dataset, *P(θ|D)*, the posterior distribution

Bayesian Neural Nets

- The parameters for a neural net model *θ* are the weights and biases
- The prior *P(θ)* is assumed to be Gaussian with mean zero and variance 1/*α*
- Error function is data error E_D plus a "weight error" term

$$\alpha E_W = \frac{\alpha}{2} \sum_{i=1}^{W} \omega_i^2$$

$$E = \beta E_D + \alpha E_W$$

Bayesian Neural Nets

- α is a “hyperparameter” that controls the size of the weights
  - Better to break α down into smaller, α_i
  - α_i Covers weights fanning out from node I

Fitting the Model

- Alternately minimize the error function

$$E = \beta E_D + \alpha E_W$$

- And refit the posterior distribution *p(θ|D)* with the evidence method (too long to explain here)
  - This refits the *α* parameter

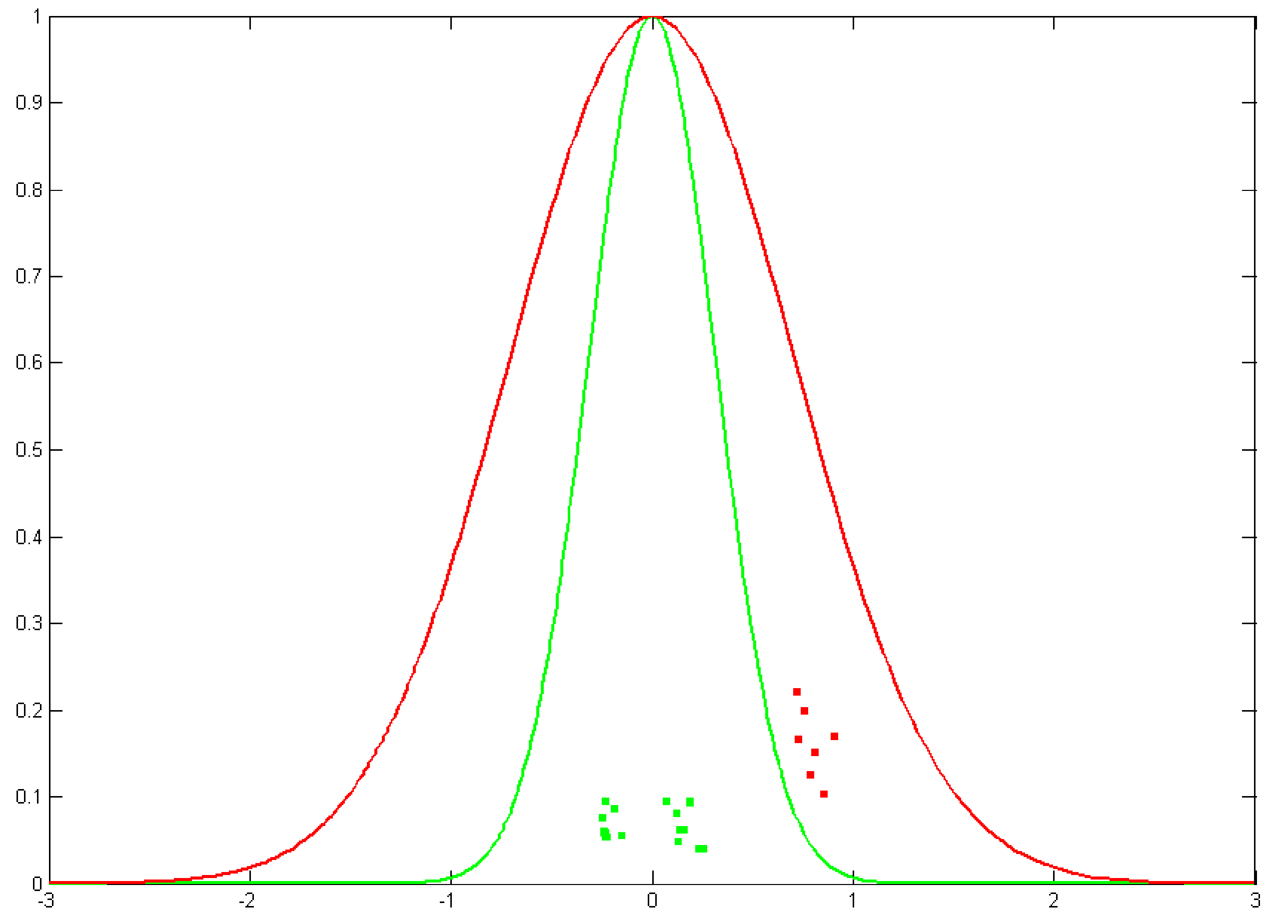
Automatic Relevance Detection

- Fitting the weights to the data and *α* results in smoother functions
  - Prevents over-fitting
- Fitting *α* to the data and the weights results in low influence nodes being further marginalized
  - Nodes with mostly small weights get assigned a smaller standard deviation
    - When fitting weights, more error from large weights

Automatic Relevance Detection

- α_i is the inverse variance of the Gaussian distribution for the weights leading from node i
  - Variance increases with weights
  - *Therefore α_i provides a measure of the relevance for the node i.*

Automatic Relevance Detection



Demo

- X1 = sin(2*pi*x1) + .002*randn
- X2 = X1 - .02*randn
- X3 = 0.5 + 0.2*randn