CS 561 Artificial Intelligence
Lecture #19

Bayesian Approach to ANN

Rashmi Dutta Baruah

Department of Computer Science & Engineering

IIT Guwahati

Outline

- Bayesian Inference - Review
- Bayesian Learning of network weights
- Bayesian Model comparison

Bayesian Inference

- $P(\theta)$: prior probability of a parameter θ
- $P(D|\theta)$: Likelihood - the probability of the data D given θ
- Using Bayes' rule the posterior probability of θ can be determined given the data D $$P(\theta|D) = \frac{P(D|\theta)P(\theta)}{P(D)}$$
- In general, entire distribution over all possible values is obtained.

How Bayesian Inference useful for ANN?

- It can be applied to neural networks to obtain various probability distributions:
  - distribution over the network weights **w** given the training data D, and a fixed model H, $\mathbf{P}(\mathbf{w}|\mathrm{D}, \mathrm{H})$
  - distribution over network outputs given the data and the hypothesis (model) H, $\mathbf{P}(\mathbf{y}|\mathrm{D}, H)$ (for regression problems)
  - distribution over predicted class labels given the data and the hypothesis (model) H, $\mathbf{P}(\mathbf{y}|\mathrm{D}, H)$ (for classification problems)
  - distribution over models given the data, $\mathbf{P}(\mathbf{H}|\mathrm{D})$
  - distribution over network outputs given the data, $\mathbf{P}(\mathbf{y}|\mathrm{D})$

Bayesian Learning of NN weights

- Finding neural network weights by minimizing some error function (equivalent to maximum likelihood) returns a single set of values for network weights.
- Bayesian approach considers a probability distribution function over weight space.
- We find posterior distribution over weights assuming that the structure of ANN (number of layers, number of hidden units, activation function) is given

$$\mathbf{P}(\mathbf{w}|D) = \frac{\mathbf{P}(D|\mathbf{w})\mathbf{P}(\mathbf{w})}{\mathbf{P}(D)} \qquad \mathbf{P}(\mathbf{D}) = \int \mathbf{P}(D|\mathbf{w})\mathbf{P}(\mathbf{w})d\mathbf{w},$$ is the normalizing factor

  - where $\mathbf{w} = w_1, w_2, .., w_W$ denotes the weight vector, and D is the target data from the training set .

Bayesian Learning of NN weights

- In the Bayesian formalism, learning the weights means changing our prior belief $\mathbf{P}(\mathbf{w})$ to the posterior, $\mathbf{P}(\mathbf{w}|D)$ as a consequence of observing the data.
- Prior for the weights: Gaussian Prior $N(0,1)(w)$

$$\mathbf{P}(\mathbf{w}) = \frac{1}{Z_W(\alpha)} exp(-\alpha E_w)$$

$$\mathrm{E_w} = \frac{1}{2}\|\mathbf{w}\|^2 = \frac{1}{2}\sum_{i=1}^{W} w_i^2$$

$$Z_w(\alpha) = \int exp(-\alpha E_w)\ dw$$

$$P(w) = \frac{1}{\sqrt{2\pi}} exp\left(-\frac{1}{2}w^2\right)$$

(smaller weights generalize better)

$\frac{1}{\sqrt{2\pi}}$ is the normalizing constant

$$\mathbf{P}(\mathbf{w}) = \frac{1}{Z_W(\alpha)} exp\left(-\frac{\alpha}{2}\|\mathbf{w}\|^2\right)$$

$$\|\mathbf{w}\|^2 = \sum_{i=1}^{W} w_i^2$$

$$Z_W(\alpha) = \left(\frac{2\pi}{\alpha}\right)^{W/2}$$

- Where W (uppercase) is the number of parameters (weights and biases), and α is a hyperparameter (for variance) since α itself controls the distribution of other parameters (weights and biases).

Bayesian Learning of NN weights

- Example of prior

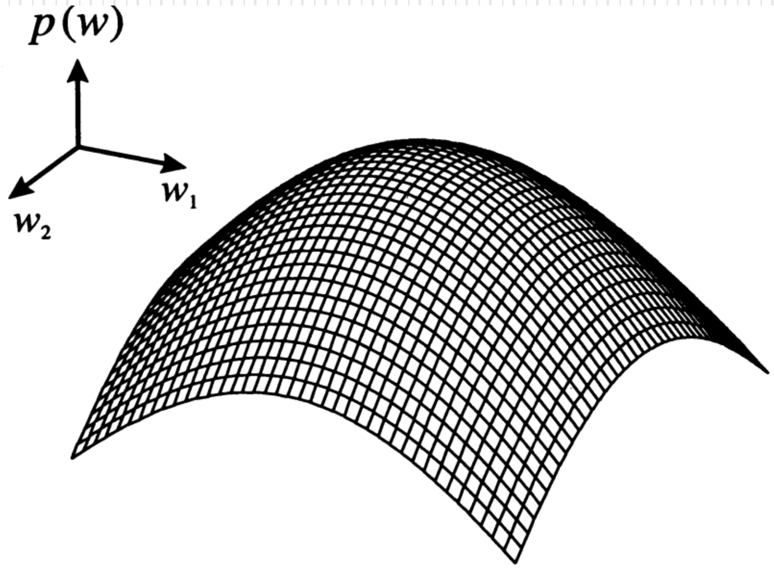


Fig. (a) Prior over two weights

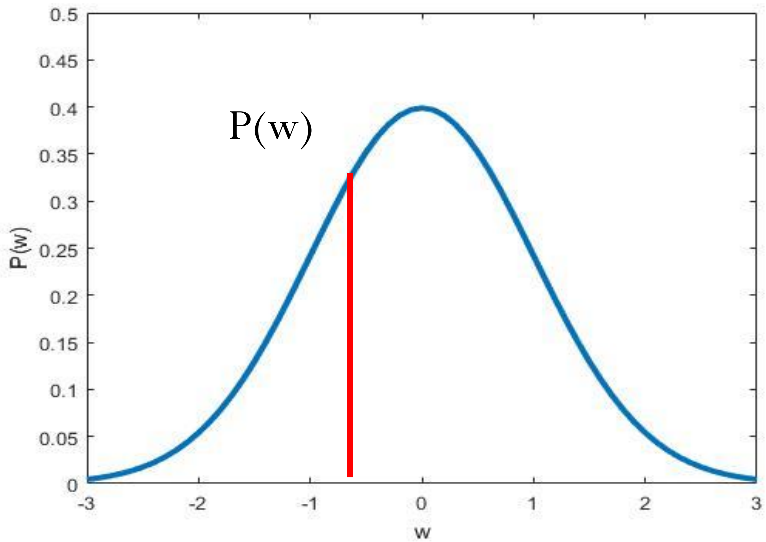


Fig. (b) Prior over one weight

Bayesian Learning of NN weights

- Likelihood of the data
  - If we assume that after training the target data $t \in D$, a Gaussian distribution is obtained with mean $y(\mathbf{x}; \mathbf{w})$, then the likelihood function is given by

$$\mathbf{P}(\mathrm{D}|\mathbf{w}) = \frac{1}{Z_{\mathrm{D}}(\beta)} exp(-\beta E_D)$$

$$\mathbf{P}(D|\mathbf{w}) = \prod_{\boldsymbol{i=1}}^{\boldsymbol{N}} \mathrm{P}(t^i|\mathbf{w}) = \frac{1}{Z_{\boldsymbol{D}}(\beta)} exp\left(-\frac{\beta}{2}\sum_{i=1}^{n}\left(y^i - t^i\right)^2\right)$$

  - where β is another hyperparameter and $\boldsymbol{Z}_D(\beta)$ is the normalization factor

Bayesian Learning of NN weights

- Posterior on weights
  - As we have now the prior and likelihood, we can use Bayes' theorem to find the posterior distribution of the weights

$$\mathbf{P}(\mathbf{w}|D) = \frac{1}{Z_{\boldsymbol{D}}(\beta)} exp\left(-\frac{\beta}{2}\sum_{i=1}^{n}\left(y^i - t^i\right)^2\right)\frac{1}{Z_{\mathrm{W}}(\alpha)} exp\left(-\frac{\alpha}{2}\|\mathbf{w}\|^2\right)$$
$$= \frac{1}{Z_{\boldsymbol{S}}} exp(-\beta E_D - \alpha E_W)$$
$$= \frac{1}{Z_{\boldsymbol{S}}} exp\big(-S(\mathbf{w})\big), \text{ where } S(\mathbf{w}) = \beta E_D + \alpha E_W$$

$$Z_{\boldsymbol{S}}(\alpha,\beta) = \int exp(-\beta E_D - \alpha E_W)$$

$$\mathbf{P}(\mathbf{w}|D) = \frac{\mathbf{P}(D|\mathbf{w})\mathbf{P}(\mathbf{w})}{\mathbf{P}(D)}$$

$$\mathbf{P}(\mathbf{D}) = \int \mathbf{P}(D|\mathbf{w})\mathbf{P}(\mathbf{w})d\mathbf{w}$$

- Maximum a posteriori (MAP) distribution
  - Finding the weight vector $\mathbf{w}_{MP}$ corresponding to the maximum of the posterior distribution
  - Minimizing the negative logarithm of $\mathbf{P}(\mathbf{w}|D)$

Bayesian Learning of NN weights

$$\mathbf{P}(\mathbf{w}|D) = \frac{1}{Z_S}\, exp(-S(\mathbf{w}))$$

$$-\log(\mathbf{P}(\mathbf{w}|D)) = -\log\left(\frac{1}{Z_S}\exp(-S(\mathbf{w}))\right) = S(\mathbf{w}) + constant$$

- Since the normalizing factor Z_S is independent of the weights, minimizing $-\log(\mathbf{P}(\mathbf{w}|D))$ is equivalent to minimizing $S(\mathbf{w})$

$$S(\mathbf{w}) = \frac{\beta}{2}\sum_{i=1}^{n}\left(y^i - t^i\right)^2 + \frac{\alpha}{2}\sum_{i=1}^{\mathrm{W}} w_i^2 = \sum_{i=1}^{n}\left(y^i - t^i\right)^2 + \frac{\alpha}{\beta}\sum_{i=1}^{\mathrm{W}} w_i^2$$

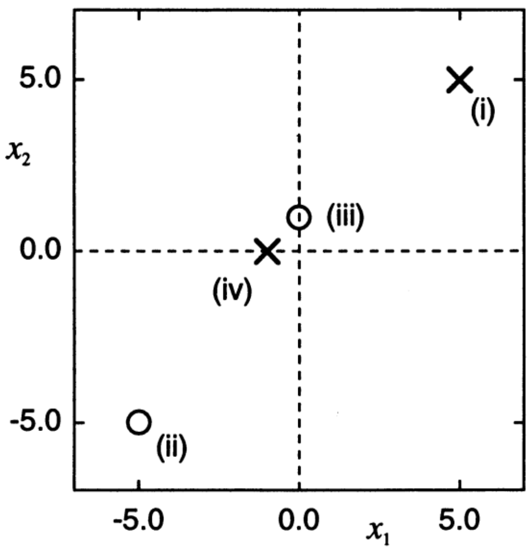
Sum of squares error function with a weight decay regularization term

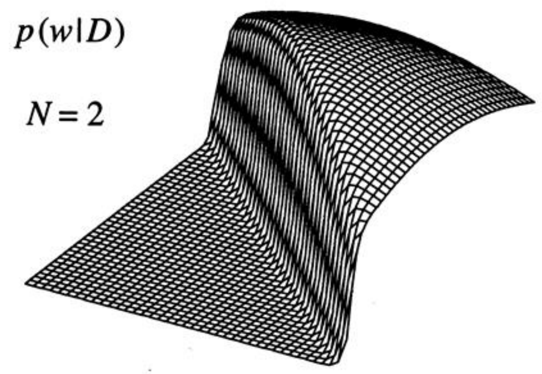
ratio of two variances- weight penalty that penalizes large weight

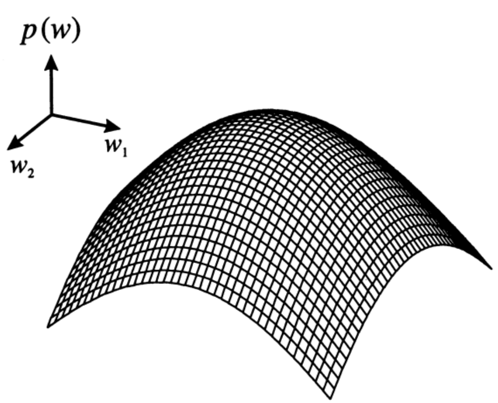
- The most probable value of the weight vector ($\mathbf{w}_{MP}$), corresponds to the maximum of the posterior probability, or equivalently to the minimum of the RHS of the above equation.

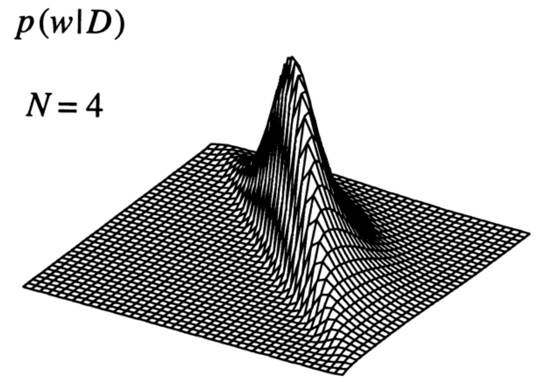
Bayesian Learning of NN weights

- A classification problem with two inputs and one logistic output









Bayesian Model Comparison

- So far, we have been dealing with the application of Bayesian methods to a neural network with a fixed number of units and a fixed architecture.
- Bayesian approach allows selection of appropriate model size and even model type.
- Consider a set of candidate models $\mathcal{H}_i$ that could include neural networks with different numbers of hidden units, RBF networks and linear models.
- With Bayes' theorem we can compute the posterior distribution over models, once we have observed training dataset and then pick the model with largest posterior.

Bayesian Model Comparison

- Consider three different models H1, H2, and H3 which are successively more complex.
- In each model we can vary the values of parameters to represent a range of input-output functions.
- More complex models can represent greater range of input-output functions.
- Using Bayes' theorem we can get the posterior probability of each of the models

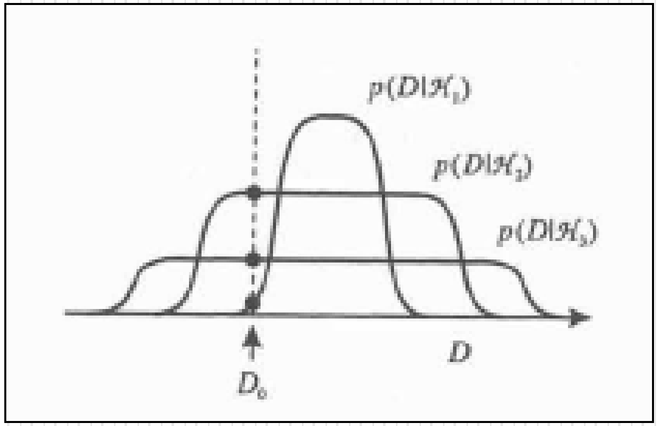


Figure: Schematic example of three models: H1, H2, H3 which are successively complex, showing the probability of the data sets D given each model

$$\mathbf{P}(\mathcal{H}_i|D) = \frac{\mathrm{P}(D|\mathcal{H}_i)\mathrm{P}(\mathcal{H}_i)}{\mathrm{P}(D)}$$

Bayesian approach can be used to select a particular model for which evidence is maximum.

Bayesian Model Comparison

$$\mathbf{P}(\mathcal{H}_i|D) = \frac{\mathrm{P}(D|\mathcal{H}_i)\mathrm{P}(\mathcal{H}_i)}{\mathrm{P}(D)}$$

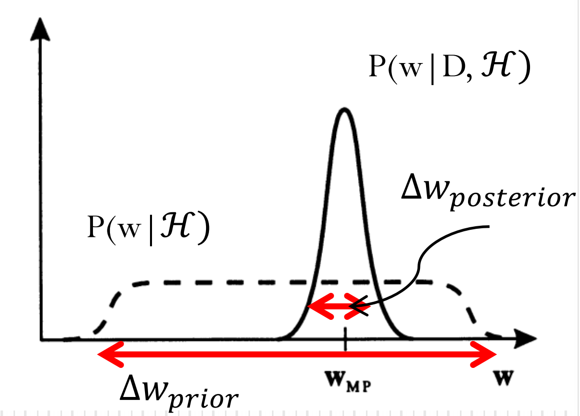
- The term $\mathrm{P}(D|\mathcal{H}_i)$ is referred to as the evidence for $\mathcal{H}_\mathrm{i}$, and is given as

$$P(D|\mathcal{H}_\mathrm{i}) = \int \mathrm{P}(D|\mathbf{w}, \mathcal{H}_i)\mathrm{P}(\mathbf{w}|\mathcal{H}_i)\mathrm{d}\mathbf{w}$$

$$\mathrm{P}(D|\mathcal{H}_i) = \mathrm{P}(\mathrm{D}, \mathcal{H}_i)/P(\mathcal{H}_i) = \frac{\int \mathrm{P}(D|\mathbf{w},\mathcal{H}_i)\mathrm{P}(\mathbf{w}|\mathcal{H}_i)P(\mathcal{H}_i)\,\mathbf{dw}}{P(\mathcal{H}_i)}$$

- The evidence term balances between fitting the data well and avoiding overly complex models.

- Consider a single weight w, if we assume that the posterior is sharply peaked around the most probable value, w_{MP}, with width $\Delta w_{posterior}$ then we can approximate the integral by the height of the peak of the integrand $\mathrm{P}(D|\mathbf{w}, \mathcal{H}_i)\mathrm{P}(\mathbf{w}|\mathcal{H}_i)$ times its width $\Delta w_{posterior}$

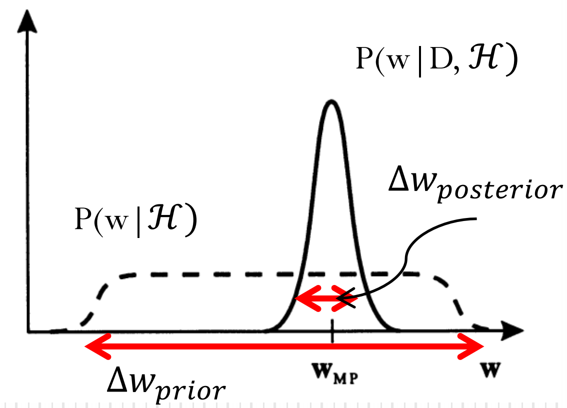


Bayesian Model Comparison

$$P(D|\mathcal{H}_\mathrm{i}) \approx P(D|w_{MP},\mathcal{H}_i)P(w_{MP}|\mathcal{H}_i)\,\Delta w_{posterior}$$

- If we consider that the prior $P(w|\mathcal{H}_i)$ is uniform on some large interval Δw_{prior}, then the evidence becomes

$$P(D|\mathcal{H}_\mathrm{i}) \approx P(D|w_{MP},\mathcal{H}_i)\frac{\Delta w_{posterior}}{\Delta w_{prior}}$$



- The first term on the RHS is the likelihood evaluated for the most probable weight values, while the second term, referred to as **Occam factor** and which has a value <1, penalizes the network for having this particular posterior distribution of weights.
- For a model with many parameters, each will generate a similar Occam factor and so the evidence will be correspondingly reduced.

What did we learn in L#19?

- Bayesian inference in ANN (Bayesian regularisation)
- Bayesian Model comparison