Learning and Generalization in linear and non-linear models

Finn Årup Nielsen
Informatics and Mathematical Modelling
Technical University of Denmark
DK-2800 Lyngby, Denmark

Email: fn@imm.dtu.dk
WWW: http://www.imm.dtu.dk/~fn
OVERVIEW

• Models
  – Linear and Non-linear models

• Learning
  – Optimization, iterative
    * Gradient
    * Hessian
  – Probability based learning
    * Regression
    * Classification

• Generalization
  – Learning curve
  – Bias variance
  – Regularization
  – Pruning
  – Committee of networks
  – Generalization assessment
    * Validation
    * Complexity criteria
MODEL

- A (mathematical) model is relation between a set of observables and parameters, e.g., a simple linear model

\[ y = \sum_{i=1}^{d} w_i x_i \]  

(1)

- A statistical model incorporate stochastic elements, which can be characterized by a probability distribution, e.g, a simple linear model with noise

\[ t = y + \epsilon = \sum_{i=1}^{d} w_i x_i + \epsilon, \]

where \( \epsilon \sim p(\epsilon) \).

- Input, output, target, noise
LINEAR AND NON-LINEAR MODELS

<table>
<thead>
<tr>
<th>Parameters / Input</th>
<th>Linear</th>
<th>Nonlinear</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linear</td>
<td>GLM with only linear terms</td>
<td>RBF</td>
</tr>
<tr>
<td>Nonlinear</td>
<td>Gamma convolution model</td>
<td>Neural network</td>
</tr>
</tbody>
</table>

Table 1: Parameter and input/output linear and nonlinear models. Partly from (Larsen 1996, table 1.1).

- Linear/linear, e.g., the general linear model (GLM)
  \[ Y = XB + U. \] (3)

- Linear/nonlinear, e.g., radial basis function networks with fixed basis functions (Bishop 1995, eq. 5.14), “generalized additive model”
  \[ y_k = \sum_{j=1}^{M} w_{kj} \phi_j(x) \] (4)

- Nonlinear/linear
  \[ y = \sum_{i=1}^{M} \exp(\beta_i) x_i \] (5)

- Nonlinear/nonlinear, e.g., two-layer neural network (Bishop 1995, eq. 4.7), where \( g \) is nonlinear
  \[ y = \tilde{g} \left[ \sum_{j=0}^{M} w_{kj}^{(2)} g \left( \sum_{i=0}^{d} w_{ji}^{(1)} x_i \right) \right] \] (6)
(Bishop 1995, p. 10) distinguishes between
- Supervized, involving a target, e.g., regression.
- Unsupervised, e.g., probability density estimation
- Reinforcement, target not known, but cost function is

Define a *cost function* that is large when the discrepancy between the model out and the target is large.

Batch/online
- Batch, all examples are used in the optimization.
- Online, one pattern at a time a “window” (some patterns are used). Stochastic, might escape, learning rate should be decreased as more examples have been presented.
OPTIMIZATION

Continuous valued smooth multidimensional functions with no constraints

- Taylor expansion of cost function around \( \hat{\mathbf{w}} \) (Bishop 1995, eq. 7.6)

\[
E(\mathbf{w}) = E(\hat{\mathbf{w}}) + (\mathbf{w} - \hat{\mathbf{w}})^T \mathbf{b} + \frac{1}{2} (\mathbf{w} - \hat{\mathbf{w}})^T \mathbf{H}(\mathbf{w} - \hat{\mathbf{w}}) + \ldots,
\]

where the gradient and Hessian is defined as

\[
(b)_i \equiv \frac{\partial E}{\partial w_i} \bigg|_{\hat{\mathbf{w}}}
\]

\[
(H)_{ij} \equiv \frac{\partial E}{\partial w_i \partial w_j} \bigg|_{\hat{\mathbf{w}}}
\]

- Appropriate when the cost function is \textit{smooth}.
- Minimum of the cost function is at a stationary point

\[
\frac{\partial E}{\partial \mathbf{w}} \bigg|_{\mathbf{w}^*} = 0.
\]

- Optimization by iterations \( \tau \)

\[
\mathbf{w}^{(\tau+1)} = \mathbf{w}^{(\tau)} + \Delta \mathbf{w}^{(\tau)}
\]
OPTIMIZATION — GRADIENT BASED

- Gradient descent (steepest descent, for neural networks: backpropagation) (Bishop 1995, sect. 7.5)

\[ \Delta \mathbf{w}^{(\tau)} = \eta \left. \frac{\partial E}{\partial \mathbf{w}} \right|_{\mathbf{w}^{(\tau)}} \]  

(12)

Increase step size \( \eta \) if successful decrease of cost function, decrease if not.

- Gradient descent with momentum

\[ \Delta \mathbf{w}^{(\tau)} = \eta \left. \frac{\partial E}{\partial \mathbf{w}} \right|_{\mathbf{w}^{(\tau)}} + \mu \Delta \mathbf{w}^{\tau-1} \]  

(13)
OPTIMIZATION — HESSION BASED

- Quadratic approximation, present point \( \hat{w} \) and optimal point \( w \) (Bishop 1995, eq 7.90)

\[
\frac{\partial E}{\partial w} = 0 = b + H(w - \hat{w})
\]

\( w = \hat{w} - H^{-1}b \)  \( (14) \)

"Newton method". Hessian not necessarily positive definite: Uphill step to maximum or saddle point.

- Make the Hessian positive definite:

\[
\tilde{H} = H + \lambda I
\]

\( (16) \)

with \( \lambda \) with a larger magnitude than the smallest negative eigenvalue of \( H \). This approximates the negative gradient as \( \lambda \to \infty \)

\[
\Delta w = - (H^{-1} + \lambda I)^{-1} b = - \frac{1}{\lambda} b
\]

\( (17) \)
PROBABILISTIC-BASED LEARNING

- Establish the probability density function for the stochastic element(s) in the model: \( p(t|x, w) \)
- Fix the data: The likelihood: \( \mathcal{L} = p(t|x, w) \)
- Cost function as the negative log-likelihood
  \[
  E = -\ln \mathcal{L}
  \]  
  (18)
- If the patterns are independent (Bishop 1995, eq. 6.5)
  \[
  E = -\ln \prod_{n=1}^{N} p(t^n|x^n) = -\sum_{n=1}^{N} \ln p(t^n|x^n)
  \]  
  (19)
- Maximum a posteriori (MAP). Likelihood augmented with a prior on the weights
  \[
  E_{\text{MAP}} = -\ln \prod_{n=1}^{N} [p(t|x, w) \, p(w)]
  \]  
  (20)
  \[
  E_{\text{MAP}} = -\sum_{n=1}^{N} \ln p(t|x, w) + \ln p(w)
  \]  
  (21)
Figure 2: Distribution of $p(t|x)$ (Bishop 1995, figure 6.1): $y$ should be “sufficiently general”, optimized completely and $N$ should be large.

- Gaussian error for the noise $\epsilon \sim \mathcal{N}(0, \sigma^2 I)$

$$p(t_k|x) = (2\pi\sigma^2)^{-\frac{1}{2}} \exp \left[ -\frac{(y_k(x; t) - t_k)^2}{2\sigma^2} \right]$$ (22)

Cost function for all outputs and all patterns

$$E = \frac{1}{2\sigma^2} \sum_n \sum_{k=1}^c (y_k^n - t_k^n)^2 + Nc \ln \sigma + \frac{Nc}{2} \ln(2\pi)$$ (23)
PROB. — CLASSIFICATION

- $y_k$ as the probability for belong to class $k$.
- Multiple attributes (multivariate Bernoulli) (Bishop 1995, sect. 6.8). With $t_k \in \{0, 1\}$

$$p(t|x) = \prod_{k=1}^{c} p(t_k|x) = \prod_{k=1}^{c} y_k^{t_k} (1 - y_k)^{1-t_k}$$

Normalization to range $[0; 1]$ with logistic function

$$y_k = \frac{1}{1 + \exp(a_k)}$$

- Multiple exclusive classes (multinomial)

$$p(t|x) = \prod_{k=1}^{c} y_k^{t_k}$$

Cross-entropy error function

$$E = - \sum_n \sum_{k=1}^{c} t_k^n \ln y_k^n$$

Normalization of range to $[0; 1]$ with the softmax function

$$y_k = \frac{\exp(a_k)}{\sum_{k'} \exp(a_{k'})}$$
• No target, density estimation of $p(x)$ with $p(x|\theta)$
• “Joint modeling”: Unsupervised/unsupervised distinction might not always be appropriate, e.g., regression can be done with unsupervised methods, where the joint probability for input and output is modeled

• Dependent on noise assumptions (noise on the input).

• A related model in (Bishop 1995, fig 6.7)
Figure 4: Overfitting and underfitting: Example of overfitting and underfitting for one-dimensional curve fitting (Bishop 1995, eq 1.4). $h(x) = 0.5 + 0.04 \sin(2\pi x)$. Blue the “true curve”. Red is estimated models.

- The variance on the parameters should be small: Not applicable to non-parametric models, such as a neural network because parameter space symmetries, e.g., sign-flip and hidden units permutations, (Bishop 1995, sect. 4.4).

- The prediction of $y$ on the training set should be small: Problem with overfitting.

- The prediction of $y$ on a new dataset should be small.
Figure 5: Learning curves for three different models: 1st order polynomial, 10th order polynomial (blue) and a neural network (green) with 10 hidden units \((W = 31)\). The target function is (Bishop 1995, eq 1.4).

- **Generalization as a function of training set size** \(N\)
- **Complex models should benefit more than simple models:**
  - Simple linear model (red): constant error with no benefit of extra training data
  - Complex models (blue/green): Decreasing test set error.
- **Select the model according to the number of training examples.**
Figure 6: Bias variance decomposition.

- Ensemble of finite training sets, $y$ a stochastic variable dependent on the training set $D$: $p(y|D)$
- Bias variance decomposition, (Bishop 1995, eq. 9.7)

$$
\mathcal{E}_D \left[ \{y(\mathbf{x}) - \langle t|\mathbf{x} \rangle \}^2 \right] = \begin{cases} 
\mathcal{E}_D [y(\mathbf{x})] - \langle t|\mathbf{x} \rangle 
\end{cases}^2 + \mathcal{E}_D \left[ \{y(\mathbf{x}) - \mathcal{E}_D [y(\mathbf{x})] \}^2 \right]
$$
CONTROLLING THE EFFECTIVE COMPLEXITY

Bishop (1995, p. 332) distinguishes between:

• Structural stabilization, changing the number of parameters
  – *Optimal Brain Damage* (OBD) pruning
  – *Optimal Brain Surgeon* (OBS) pruning
  – Node pruning
• Regularization, “Adding a penalty term $\Omega$ to the cost function”
  – Weight decay, (Bishop 1995, sect. 9.2.1) also called ridge regression
    \[ \Omega = 1/2 \sum_i w_i^2. \] (31)
  – Soft weight sharing, Weights generated from a mixture of Gaussians.
  – Early stopping. Stop the optimization when the validation set is lowest.
  – Training with noise, perturbing the data points in the training set with noise.
Figure 7: Two-layer neural network curvefitting with least squares an with 40 hidden units and \( N = 10 \) training examples. “x” is training set and “o” is validation set.

- **Effective optimization** (Levenberg–Marquardt): fast learning and fast overfitting.
- **Slow optimization** (gradient descent with adaptive step size): Slow convergence, but no overfitting (yet!).
REGULARIZATION FROM PROBABILISTIC ASSUMPTIONS

• Bayes formula, (Bishop 1995, eq. 10.3)

\[ p(w|D) = \frac{p(D|w)p(w)}{p(D)} \]  \hspace{1cm} (32)

where \( w \) is the parameters and \( D \equiv (t^1, \ldots, t^N) \) is the training set of the target.

• \( p(D) \) is constant for a fixed data set

\[ p(w|D) \propto p(D|w)p(w) \]  \hspace{1cm} (33)

• Cost function

\[ E = - \ln p(D|w) - \ln p(w) \]  \hspace{1cm} (34)

• If Gaussian prior (independent) on the weights

\[ p(w) \propto \exp\left(-\lambda \sum_i w_i^2\right) \]

\[ E = - \ln p(D|w) + \lambda \sum_i w_i^2 \]  \hspace{1cm} (35)

which is \textit{weight decay}.
Figure 8: Pruning by OBD in a two-layer neural network curve fitting with least squares and with 40 hidden units and \( N = 50 \) training examples and \( N_{val} = 50 \) validation examples.

- Optimal Brain Damage (OBD) considers the saliency of weights: The change in the cost function when a small perturbation is made on a weight (Bishop 1995, eq. 9.66)

\[
\delta E = \sum_i \frac{\partial E}{\partial w_i} \delta w_i + \frac{1}{2} \sum_i \sum_j H_{ij} \delta w_i \delta w_j + \ldots \\
\text{Ignore if optimized}
\]

and diagonal approximation to the Hessian. Erase (set to zero) the weights associated with low effect (saliency).
**COMMITTEE OF NETWORKS**

![Committee network for model with two outputs.](image)

- **Consensus model**, e.g., average output of $L$ models, (Bishop 1995, eq. 9.83)

  $$y_{\text{COM}}(x) = \frac{1}{L} \sum_{i=1}^{L} y_i(x)$$  

- **This prediction is better than the average error of the individual models** ($E_{\text{COM}} < E_{\text{AV}}$), if
  - Errors are uncorrelated. Fully correlated (the same model) $E_{\text{COM}} = E_{\text{AV}}$.
  - Error function is convex, e.g., Gaussian
COMMITTEE OF NETWORKS

Figure 10: Committee network for model with two outputs.

- With no model bias against the true output

\[ E_{\text{COM}} = \frac{1}{L} E_{\text{AV}} \]  

\[ E_{\text{AV}} = \frac{1}{L} \sum_{i=1}^{L} E_i = \frac{1}{L} \sum_{i=1}^{L} \mathcal{E} \left[ \epsilon_i^2 \right] \]  

\[ E_{\text{COM}} = \mathcal{E} \left[ \left( \frac{1}{L} \sum_{i=1}^{L} y_i (x - h(x)) \right)^2 \right] = \mathcal{E} \left[ \left( \frac{1}{L} \sum_{i=1}^{L} \epsilon_i \right)^2 \right] \]
COMMITTEE OF NETWORKS

![Figure 11: Committee neural network prediction.](image)

- Models should be heterogenous, e.g., a linear model will fit the same curve.
- Averaging over models mostly reduces the variance rather than the bias.
- Neural network committee example (permuting training and test set, different seed, very little regularization) with validation set and early stop. 10 networks in committee.

\[
E_{AV, \text{ Test set}} = 0.0394 \quad (41)
\]
\[
E_{COM, \text{ Test set}} = 0.0084 \quad (42)
\]

2 individual models were better, 8 worse. Empirical observation: Errors are not necessarily Gaussian
MODEL ORDER SELECTION, VALIDATION

- Validation-based (Bishop 1995, sect. 9.8.1), test-set should be independent “Hold out method”) and from the same distribution
  - Single-set validation. A finite size validation set will be “noisy”.
  - Cross-validation, partition the data set in $S$ distinct segments. $S$ times larger computation

![Cross-validation partitioning diagram](image)

Figure 12: Cross-validation partitioning (Bishop 1995, figure 9.17). With $N = 50$ examples and $S = 5$ distinct partitions of the data.

- Leave-one-out. Only one example in the validation set.
- “Overvalidation”: If the validation set is applied too much the model might not generalize (Bishop 1995, p. 364–365), e.g., consider random models picked by the validation set.
Figure 13: Bias variance decomposition on a 20th order polynomial with a weight decay hyperparameter varied with $N = 10$ examples and 100 runs (Bishop 1995, fig. 9.16, eqs. 9.109 and 9.110).

- **Bias variance as a function of model complexity** (Bishop 1995, figure 9.16): $\mathcal{E}_D[y] \approx \bar{y} = \sum_{i=1}^{100} y_i$ and $\langle t|x \rangle$ assessed by large validation set

- **Simple models**: High bias, low variance, e.g., a constant model $y = 0$ have no variance and bias as $\langle t|x \rangle^2$.

- **Complex model**: Might have low bias and high variance, e.g., a model that fits the data points perfectly.
• Complexity criteria, (Bishop 1995, sect. 9.8.3). One of the forms (Bishop 1995, eq. 9.111)

\[
\text{PE} = \text{training error} + \text{complexity term} \quad (43)
\]

For sum-of-squares error, \( E = \frac{1}{2} \sum_{n=1}^{N} [y(x, w) - t]^2 \)

- Final prediction error (FPE), (Bishop 1995, eq. 9.112)

\[
\text{FPE} = \frac{2E}{N} + \frac{W}{N} \sigma^2 \quad (44)
\]

where \( W \) is the number of free parameters.

- Generalized prediction error (GPE)

\[
\text{GPE} = \frac{2E}{N} + \frac{2\gamma}{N} \sigma^2 \quad (45)
\]

where \( \gamma \) is an effective number of parameters.
CONCLUSION

- Learning can be performed in a variety of ways: gradient, Hessian-based.
- Learning problems can be based on probabilistic models: regression, classification, ...
- Model should generalize well: It should not only consider presented data (training) but fit new data as well.
- Complexity of model can be adjusted by varying the number of free parameters, by regularization, pruning or by not training the model “well”.
- Combining models (“consensus models”, “committee of network”) might improve generalization.
- Generalization can be assessed by validation set or by complexity criteria.
References
