(Towards) Statistical Machine Learning

2. A probabilistic formulation of supervised ANNs
Statistical Machine Learning?
Postgrad course at Carnegie Mellon U.

Course topics chosen from the following basic outline, subject to change …

- **Statistical theory**: Maximum likelihood, Bayes, minimax, Parametric versus Nonparametric Methods, Bayesian versus Non-Bayesian Approaches, classification, regression, density estimation.

- **Convexity and optimization**: Convexity, conjugate functions, unconstrained and constrained optimization, KKT conditions.

- **Parametric methods**: Linear Regression, Model Selection, Generalized Linear Models, Mixture Models, Classification (linear, logistic, support vector machines), Graphical Models, Structured Prediction, Hidden Markov Models.

- **Sparsity**: High Dimensional Data and Sparsity, Basis Pursuit and the Lasso Revisited, Sparsistency, Consistency, Persistency, Greedy Algorithms for Sparse Linear Regression, Sparsity in Nonparametric Regression. Sparsity in Graphical Models, Compressed Sensing.

- **Nonparametric methods**: Nonparametric Regression and Density Estimation, Nonparametric Classification, Boosting, Clustering and Dimension Reduction, PCA, Manifold Methods, Principal Curves, Spectral Methods, The Bootstrap and Subsampling, Nonparametric Bayes.  
  
  [...]cont...]
Statistical Machine Learning?
Postgrad course at Carnegie Mellon U.

(Cont.)

Course topics chosen from the following basic outline, subject to change.

- **Advanced theory**: Concentration of Measure, Covering numbers, Learning theory, Risk Minimization, Tsybakov noise, minimax rates for classification and regression, surrogate loss functions, boosting, sparsistency, Minimax theory.

- **Kernel methods**: Mercer kernels, reproducing kernel Hilbert spaces, relationship to nonparametric statistics, kernel classification, kernel PCA, kernel tests of independence.

- **Computation**: The EM Algorithm, Simulation, Variational Methods, Regularization Path Algorithms, Graph Algorithms.

- **Other learning methods**: Functional Data, Semi-Supervised Learning, Reinforcement Learning, Minimum Description Length, Online Learning, The PAC Model, Active Learning.
Statiscal Machine Learning?
SML research group at Berkeley, CA.

Research statement

• Statistical machine learning merges statistics with the computational sciences: computer science, systems science and optimization.

• Much of the agenda in statistical machine learning is driven by applied problems in science and technology, where data streams are increasingly large-scale, dynamical and heterogeneous, and where mathematical and algorithmic creativity are required to bring statistical methodology to bear.

• Fields such as bioinformatics, artificial intelligence, signal processing, communications, (etc.) are all being heavily influenced by developments in statistical machine learning.

• The field of statistical machine learning also poses some of the most challenging theoretical problems in modern statistics, chief among them being the general problem of understanding the link between inference and computation.

• Research in statistical machine learning at Berkeley builds on Berkeley's world-class strengths in probability, mathematical statistics, computer science and systems science. Moreover, by its interdisciplinary nature, statistical machine learning helps to forge new links among these fields.
## Glossary

<table>
<thead>
<tr>
<th><strong>Machine learning</strong></th>
<th><strong>Statistics</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>network, graphs</td>
<td>model</td>
</tr>
<tr>
<td>weights</td>
<td>parameters</td>
</tr>
<tr>
<td>learning</td>
<td>fitting</td>
</tr>
<tr>
<td>generalization</td>
<td>test set performance</td>
</tr>
<tr>
<td>supervised learning</td>
<td>regression/classification</td>
</tr>
<tr>
<td>unsupervised learning</td>
<td>density estimation, clustering</td>
</tr>
<tr>
<td>large grant = $1,000,000</td>
<td>large grant = $50,000</td>
</tr>
<tr>
<td>nice place to have a meeting: Snowbird, Utah, French Alps</td>
<td>nice place to have a meeting: Las Vegas in August</td>
</tr>
</tbody>
</table>

Bayes & supervised ANNs
Recap: The Reverend and the Pope

\[
P(H|Bene) = \frac{P(Bene, H)}{P(Bene)} = \frac{P(Bene, H)}{P(Bene, H) + P(Bene, A)} = \frac{P(Bene|H)P(H)}{P(Bene|H)P(H) + P(Bene|A)P(A)}
\]

\[
P(Bene) = P(Bene, H) + P(Bene, A)
\]

\[
P(Bene, H) = P(Bene|H)P(H)
\]

\[
P(Bene, A) = P(Bene|A)P(A)
\]
Recap: The Reverend and the Pope

Bayes' theorem for probability densities

\[ f_X(x | Y = y) = \frac{f_{X,Y}(x, y)}{f_Y(y)} = \frac{f_{X,Y}(x, y)}{\int_{-\infty}^{+\infty} f_Y(y | X = \xi) f_X(\xi) d\xi} = \frac{f_Y(y | X = x) f_X(x)}{\int_{-\infty}^{+\infty} f_Y(y | X = \xi) f_X(\xi) d\xi} \]

\[ f_Y(y) = \int_{-\infty}^{+\infty} f_Y(y | X = \xi) f_X(\xi) d\xi \]

\[ f_{X,Y}(x, y) = f_Y(y | X = x) f_X(x) \]
Bayesian Neural Networks

“Probability theory should lay at the foundation any learning algorithm, otherwise risking that the reasoning performed in it be inconsistent in some cases.” (heuristics)

“Embedding probability theory into machine learning techniques requires modeling assumptions to be made explicit; it also automatically satisfies the likelihood principle and provides a natural framework to handle uncertainty.” (what type of uncertainty?)

“Probability theory is ideally suited as a theoretical foundation for pattern recognition.”
Bayesian Neural Networks (2)

**Advantages** of a Bayesian formulation for ANNs?

- The **OVERFITTING** problem can be handled using adaptive control of the complexity of the obtained model (parameters).
- The uncertainty with regard to models, parameters and results can often be explicitly calculated (e.g., error bars).
- All assumptions concerning the model and its parameters can (and have to!) be declared explicitly and integrated in the model calculations.
- It provides a **probabilistic interpretation** of the error function.
- Different models can be integrated probabilistically (**mixture of experts / committee networks / ensemble learning**) or compared using only training data.
But before getting under the bonnet ...
Step back: plain Neural Networks

- From a simple **linear discriminant** function to **logistic discrimination** and the **perceptron**

\[
y(x) = wx + w_0
\]

\[
y(x) = g(wx + w_0)
\]

\[
g(a) = \frac{1}{1 + \exp(-a)}
\]

\[
P(C_1|x) = \frac{p(x|C_1)p(C_1)}{p(x|C_1)p(C_1) + p(x|C_2)p(C_2)}
\]
Step back: plain Neural Networks

From a simple **perceptron** to the **MLP**

**Perceptron:** Rosemblatt (1962); Widrow and Hoff (1960)

\[
y = g \left( \sum_{j=0}^{M} w_j \phi_j(x) \right) = g(w \Phi)
\]

REMEMBER LOGISTIC DISCR.

\[
y(x) = g(wx + w_o)
\]
Step back: plain Neural Networks

- From a simple perceptron to the MLP

**MLP:** Two- and three-layered networks as universal approximators.

\[
y_k = \tilde{g}(a_k) = \tilde{g} \left( \sum_{j=0}^{M} \tilde{w}_{kj} z_j(x) \right) = \tilde{g} \left( \sum_{j=0}^{M} \tilde{w}_{kj} g \left( \sum_{i=0}^{D} w_{ji} x_i \right) \right)
\]
Step back: plain Neural Networks

\[ P(\text{data} | \text{class}) \propto e^{-\frac{(x-\mu)^2}{2\sigma^2}} \]

\[ P(\text{data} | \sim \text{class}) \propto e^{-\frac{(x-\mu')^2}{2\sigma'^2}} \]

\[ P(\text{class} | \text{data}) = \frac{P(\text{data} | \text{class}) \cdot P_c}{P(\text{data} | \text{class}) \cdot P_c + P(\text{data} | \sim \text{class}) \cdot (1 - P_c)} \]

\[ P(\text{class} | \text{data}) = \frac{1}{1 + e^{-\frac{x(\mu - \mu')}{2\sigma^2} \cdot (1 - P_c)}} \cdot \frac{1}{P_c} = \frac{1}{1 + e^{-w_{\text{data}} - \theta}} \]
Back to Bayesians …
The Bayesian formalism framework works at different levels of the development of a supervised neural network:

1. **Weight Selection**
   
   \[ p(\omega|D) = \frac{p(D|\omega)p(\omega)}{p(D)} \]

2. **Hyperparameters Selection**
   
   \[ p(\alpha, \beta|D) = \frac{p(D|\alpha, \beta)p(\alpha, \beta)}{p(D)} \]

3. **Model Selection**
   
   \[ p(M_i|D) = \frac{p(D|M_i)p(M_i)}{p(D)} \]
The Bayesian formalism framework works at **different levels** of the development of a supervised neural network:

\[ p(M_i|D) = \frac{p(D|M_i)p(M_i)}{p(D)} \]

- Out of a number of different models, we’ll choose that with **largest** conditional probability… or not?...
- This is equivalent to choosing the model with the **highest** evidence
Bayesian Neural Networks (5)

The Bayesian formalism framework works at different levels of the development of a supervised neural network:

$$p(M_i|D) = \frac{p(D|M_i)p(M_i)}{p(D)}$$

The evidence of an ANN model, in practice, can be approximated ...

REMEMBER: $f_Y(y) = \int_{-\infty}^{+\infty} f_Y(y|X = \xi)f_X(\xi)d\xi$

$$p(D|M_i) = \int p(D|\omega, M_i)p(\omega|M_i)d\omega$$

$$\equiv p(D|\omega_{MP}, M_i)p(\omega_{MP}|M_i)\Delta\omega_{posterior} \equiv p(D|\omega_{MP}, M_i)\frac{\Delta\omega_{posterior}}{\Delta\omega_{prior}}$$

Occam factor
The Bayesian formalism framework works at different levels of the development of a supervised neural network:

\[ p(M_i|D) = \frac{p(D|M_i)p(M_i)}{p(D)} \]

… But, actually, the recommended approach is using a “committee of networks”…

\[ P(O|D) = \sum_i P(O|M_i, D)P(M_i|D) \]

Given the limitations of some of the approximations that are required, the most sensible approach entails the use of the models’ evidences as weighting coefficients in a (non)linear combination of the models’ outputs.
The Bayesian formalism framework works at different levels of the development of a supervised neural network:

1. **Weight Selection**
   \[ p(\omega|D) = \frac{p(D|\omega)p(\omega)}{p(D)} \]

2. **Hyperparameter Selection**
   \[ p(\alpha, \beta|D) = \frac{p(D|\alpha, \beta)p(\alpha, \beta)}{p(D)} \]

3. **Model Selection**
   \[ p(M_i|D) = \frac{p(D|M_i)p(M_i)}{p(D)} \]
Bayesian Neural Networks (7)

The Bayesian formalism framework works at different levels of the development of a supervised neural network:

\[
p(\omega|D) = \frac{p(D|\omega)p(\omega)}{p(D)}
\]

1. **WEIGHT SELECTION**

- We start from a *prior distribution of the weights*

\[
p(\omega) = \frac{1}{C_{norm}} \exp(-\alpha E_w)
\]

- ... where \( E_w = \frac{1}{2} \sum_{i=1}^{W} \omega_i^2 = \frac{1}{2} \|w\|^2 \) ... This corresponds to the use of a **weight decay regularizer**
Bayesian Neural Networks (8)

The Bayesian formalism framework works at different levels of the development of a supervised neural network:

\[ p(\omega|D) = \frac{p(D|\omega)p(\omega)}{p(D)} \]

...starting from an expression for the evidence or likelihood

\[ p(D|\omega) = \frac{1}{C_{\text{norm}}} \exp(-\beta E_D) \]

...where

\[ E_D = \frac{1}{2} \sum_{n=1}^{N} \{y(x_n; w) - t_n\}^2 \]

...and assuming that the targets have been generated by a smooth function with added Gaussian noise (beta!)...
Bayesian Neural Networks (9)

- The Bayesian formalism framework works at different levels of the development of a supervised neural network:

\[ p(\omega|D) = \frac{p(D|\omega)p(\omega)}{p(D)} \]

1 WEIGHT SELECTION

- … then, the conditional a posteriori probability is proportional to:

\[ p(\omega|D) \propto \frac{1}{C_{\text{norm}}} \exp\left(- (\beta E_D + \alpha E_w)\right) \]

- … Notice (!!!) that maximization of this expression is almost equivalent to the ML solution for large data sets …
Bayesian Neural Networks (10)

The Bayesian formalism framework works at different levels of the development of a supervised neural network:

\[
p(\omega|D) = \frac{p(D|\omega)p(\omega)}{p(D)}
\]

\[
p(\omega|D) \propto \frac{1}{C_{\text{norm}}} \exp(- (\beta E_D + \alpha E_w))
\]

1

WEIGHT SELECTION

… In what sense do we say that the maximization is almost equivalent to the ML solution for large data sets?

- The most probable (likely) solution for the weights, or \( w_{MP} \), corresponds to the minimum of \( S(w) = (\beta E_D + \alpha E_w) \)

- … or minimum error

\[
S(w) = \frac{\beta}{2} \sum_{j=0}^{M} (y(x_n, w) - t_n)^2 + \frac{\alpha}{2} \sum_{i=1}^{W} w_i^2
\]

- … and maximum posterior probability

\[
p(\omega|D) \propto \frac{1}{C_{\text{norm}}} \exp(- (\beta E_D + \alpha E_w))
\]
(Another) step back: supervised models & function approximation & (the other) ML

Function fitting and **Maximum Likelihood**

- Suppose data arising from a statistical model: \( Y = f(X) + \varepsilon \)
- For this model: \( f(x) = \mathbb{E}(Y|X = x) \)
- In most cases, the input–output pairs \((X,Y)\) will not have a deterministic relationship \(Y = f(X)\)
- Any **supervised learning model** attempts to **learn** \(f\) **by example** through a teacher \((\text{target})\).
- A supervised learning algorithm produces outputs \(\hat{f}(x_i)\) in response to inputs.
- Adaptation of parameters **led by error** \(y_i - \hat{f}(x_i)\)
- The goal is obtaining an approximation to \(f(X)\) for all \(x\) in some region of \(\mathbb{R}^p\), given the representations in \(T=(X,Y)\).
(Another) step back: supervised models & function approximation & (the other) ML

- Function fitting and ML
  - Model approximations will have associated a set of parameters $\theta$ that can be adaptively modified to fit the available data.

Examples:
- Linear model $f(x) = x^T \beta$ has $\theta = \beta$
- Linear basis expansions:
- ... Or nonlinear ones
  $$h_k(x) = \frac{1}{1 + \exp(-x^T \beta_k)}$$
  - The adaptive parameters in these models can be estimated by maximum likelihood. The log-probability of observed sample is:

$$L(\theta) = \sum_{i=1}^{N} \log P_{\theta}(y_i)$$
(Another) step back: supervised models & function approximation & (the other) ML

Function fitting and **ML**

- **ML** assumes that the *most likely values for* $\theta$ *are those for which the probability of the observed sample is largest.*

- Conditional likelihood:

\[
\Pr(Y | X, \theta) = N(f_\theta(X), \sigma^2)
\]

- leading to **log-likelihood**:

\[
L(\theta) = -\frac{N}{2} \log(2\pi) - N \log \sigma - \frac{1}{2\sigma^2} \sum_{i=1}^{N} (y_i - f_\theta(x_i))^2
\]
Bayesian Neural Networks (11)

The Bayesian formalism framework works at **different levels** of the development of a supervised neural network:

\[ p(\omega|D) = \frac{p(D|\omega)p(\omega)}{p(D)} \]

**1. WEIGHT SELECTION**

- Distribution of the network *outputs*
  \[ p(t|x, D) = \int_w p(t|x, w)p(w|D)dw \]

**REMEMBER:** \[ p(D|M_i) = \int p(D|w, M_i)p(w|M_i)dw \]

- Error bars in regression
  \[ p(t|x, D) \approx C \exp \left( -\frac{(t - y_{MP})^2}{2\sigma_t^2} \right) \]

\[ \sigma_t^2 = \beta^{-1} + g^T A^{-1} g \]

Two terms: one from intrinsic noise, another from width of post dist of weights
The Bayesian formalism framework works at different levels of the development of a supervised neural network:

\[ p(\alpha, \beta | D) = \frac{p(D | \alpha, \beta) p(\alpha, \beta)}{p(D)} \]

HYPERPARAMETER SELECTION

- Variance \( \beta \) and regularization \( \alpha \) hyperparameters…

\[ S(w) = \frac{\beta}{2} \sum_{n=1}^{N} (y(x_n, w) - t_n)^2 + \frac{\alpha}{2} \sum_{i=1}^{W} w_i^2 \]

\[ p(D | \alpha, \beta) = \int p(D | w, \alpha, \beta) p(w | \alpha, \beta) dw = \int p(D | w, \beta) p(w | \alpha) dw \]

... can be calculated differentiating with respect to hyperparameters obtaining iterative formulae for their estimation
Over- and under-fitting

A STEP ASIDE: OVERFITTING

When fitting a model to noisy data (ALWAYS), we make the assumption that the data have been generated from some “TRUE” model by making predictions at given values of the inputs, then adding some amount of noise to each point, where the noise is drawn from a normal distribution with an unknown variance.

Our task is to discover both this model and the width of the noise distribution. In doing so, we aim for a compromise between bias, where our model does not follow the right trend in the data (and so does not match well with the underlying truth), and variance, where our model fits the data points too closely, fitting the noise rather than trying to capture the true distribution. These two extremes are known as underfitting and overfitting.

IMPORTANT! : the number of parameters in a model; the higher, the more complexly the model can fit the data. If the number of parameters in our model is larger than that the “true one”, then we risk overfitting, and if our model contains fewer parameters than the truth, we could underfit.
A STEP ASIDE: OVERFITTING

The illustration shows how increasing the number of parameters in the model can result in overfitting. The 9 data points are generated from a cubic polynomial which contains 4 parameters (the true model) and adding noise. We can see that by selecting candidate models containing more parameters than the truth, we can reduce, and even eliminate, any mismatch between the data points and our model. This occurs when the number of parameters is the same as the number of data points (an 8th order polynomial has 9 parameters).
Function fitting and ML

Models can be ‘restricted’ by explicitly penalizing $RSS(f)$ (residual sum of squares) with a roughness penalty term:

$$PRSS(f; \lambda) = RSS(f) + \lambda J(f)$$

The functional $J(f)$ will be large for functions $f$ that vary too rapidly over small regions of input space.

Penalty function methods for model regularization express our prior belief that the type of functions we seek exhibit a certain type of smooth behavior, and can usually be defined within a Bayesian framework. The penalty $J$ corresponds to a log-prior, and $PRSS(f; \lambda)$, to the log-posterior distribution to be minimized.

$$p(\alpha|D) \propto \frac{1}{C_{\text{norm}}} \exp\left(-\beta E_D + \alpha E_w\right)$$
What else does this Bayesian formalism offer us?

**Variable selection**: Automatic Relevance Determination (ARD): regularization terms are associated to each network input. The *a priori* distribution of the weights is defined as:

\[
p(w) = A \exp \left( - \sum_{c} \alpha_c \sum_{i} w_i^2 / 2 \right)
\]

**Automatic Regularization**: The regularization coefficients associated to irrelevant inputs are inferred high values that make the corresponding weights tend to 0. This can be interpreted as *soft input pruning*. Inspection of the final values \(\{\alpha_c\}\) indicates the relative relevance of each variable.
Bayesian Neural Networks (14)

The Bayesian formalism for ANNs in a nutshell

- **Initialization** of hiperparameters $\alpha$ and $\beta$. Initialization of weights with values obtained from $p(w)$
- **Network training** using standard optimization techniques to minimize $(E_d + E_w)$
- Insert, every few iterations of the algorithm, the **reevaluation of $\alpha$ and $\beta$**, which entails the evaluation of the Hessian of the error function $E_d$ and the extraction of its eigenvalues.
- [optionally, repeat the three previous steps to limit the negative effect of local minima of the error function]
- Repeat the previous steps for a sample of different possible models or, alternatively, use a “**committee of networks**” or select models with high evidence to define a “**mixture of experts**”.
Bayesians

- C.M. Bishop, 1995. *Neural Networks for Pattern Recognition* (Ch.10). Oxford University Press.
- Tom Loredo, 2000. Bayesian Inference, a practical primer (tutorial)
  - [http://astrosun.tn.cornell.edu/staff/loredo/bayes/tjl.html](http://astrosun.tn.cornell.edu/staff/loredo/bayes/tjl.html)