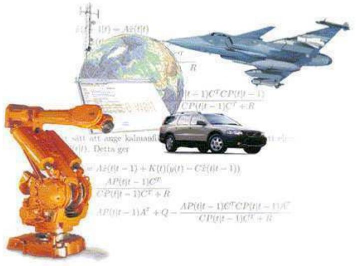
Machine Learning, Lecture 6
Boosting and Introduction to Graphical Models

"Boosting is one of the most powerful learning ideas introduced in the last twenty years." **HTF**



Thomas Schön

Division of Automatic Control
Linköping University
Linköping, Sweden.

Email: schon@isy.liu.se,
Phone: 013 - 281373,
Office: House B, Entrance 27.

Outline Lecture 6

1. Summary of Lecture 5
2. Boosting
   - Motivation
   - The AdaBoost algorithm
   - An interpretation of AdaBoost as additive modelling
   - An illustrative example
   - Two recent applications of AdaBoost
3. Introducing Graphical Models
   - Motivation and some basic facts
   - An example - Linear dynamic models

(Chapter 14.3 and Chapter 8.1)

Summary of Lecture 5 (I/III)

The Suppor vector machine (SVM) is a discriminative classifier providing a decision boundary, $\{x | w^T\phi(x) + b = 0\}$ that gives the maximum margin.

The margin is the perpendicular distance from the closest point to the decision boundary.

The decision boundary that maximizes the margin is given as the solution to the quadratic program (QP)

$$\min_{w,b} \frac{1}{2}\|w\|^2$$
$$\text{s.t.} \quad t_n(w^T\phi(x_n) + b) - 1 \geq 0, \quad n = 1, \dots, N$$

Summary of Lecture 5 (II/III)

Variational Inference is a type of approximate Bayesian inference where factorial approximations like

$$p(Z|X) \approx q(Z) = \prod_i q_i(Z_i)$$

are made on the form of the posteriors.

The Kullback-Leibler (KL) Divergence is used to find optimal approximations for the posteriors in two different forms.

Variational Bayes is a form of variational inference where KL$(q||p)$ is used for the optimization. We fix all but one of the factors and optimize as follows.

$$\hat{q}_j(Z_j) = \arg\min_{q_j} \left(q_j(Z_j) \prod_{i \neq j} \hat{q}_i(Z_i) \middle|\middle| p(X, Z) \right)$$

Summary of Lecture 5 (III/III)

Expectation Propagation is another form of variational inference where $\mathrm{KL}(p||q)$ is used for the optimization. We fix all but one of the factors and optimize.

$$\hat{q}_j(Z_j) = \arg\min_{q_j} \left(f_j(Z_j) \prod_{i\neq j} \hat{q}_i(Z_i) \middle|\middle| q_j(Z_j) \prod_{i\neq j} \hat{q}_i(Z_i) \right)$$

where f_j is the correct factor in $p(X, Z)$.

The AdaBoost Algorithm

Algorithm (*AdaBoost (Adaptive boosting)*)

1. *Initialize training data weights* $w_n^{(1)} = 1/N, n = 1, \ldots, N.$
2. *for m=1:M*
   (a) *Learn a classifier by minimizing the weighted cost function*
   $$J_m = \sum_{n=1}^{N} w_n^{(m)} I\left(y_m(x_n) \neq t_n\right).$$
   (b) *Compute* $\epsilon_m = \sum_{n=1}^{N} w_n^{(m)} I\left(y_m(x_n) \neq t_n\right) / \sum_{n=1}^{N} w_n^{(m)}$.
   (c) *Compute* $\alpha_m = \ln\left((1-\epsilon_m)/\epsilon_m\right)$.
   (d) *Compute new weights*
   $$w_n^{(m+1)} = w_n^{(m)} \exp\left(\alpha_m I(y_m(x_n) \neq t_n)\right).$$
3. *Result:* $Y_M(x) = \mathrm{sign}\left(\sum_{m=1}^{M} \alpha_m y_m(x)\right)$.

Some Comments on the AdaBoost Algorithm

1. I denotes the indicator function,
$$I(y_m(x_n) \neq t_n) = \begin{cases} 1 & y_m(x_n) \neq t_n \\ 0 & \text{otherwise} \end{cases}$$
2. The cost function J_m minimized in step 2a of the algorithm makes use of the weights $w_n^{(m)}$ to assign a *greater cost* to the training samples that were *previously misclassified*.
3. In step 2b, ϵ_m represents the weighted measure of error rates for each of the weak classifiers. Via α_m in step 2c, this is used to find the final combination of the weak classifiers, resulting in a strong classifier.
4. In step 2d the weights $w_n^{(m)}$ corresponding to misclassified data are increased and weights $w_n^{(m)}$ corresponding to correctly classified data are not changed.

Commonly Used Weak Classifier

A commonly used weak classifier is the so called decision stump (decision tree with one node),

$$y_m(x_n) = \begin{cases} 1 & p x_n^j < p\lambda \\ -1 & \text{otherwise} \end{cases}$$

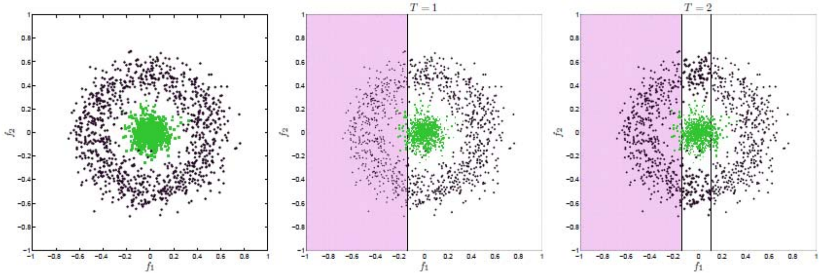
where $j \in \{1, \ldots, J\}$ indexes x_n, $p \in \{+1, -1\}$ is the "polarity" and $\lambda \in \mathbb{R}$ is a threshold. Hence, in step 2a we need to estimate $\theta = \begin{pmatrix} j & p & \lambda \end{pmatrix}^T$.

This corresponds to a partitioning of the input space into two half spaces by a decision boundary that is parallel to one of the input axes.

An Illustrative Example (I/III)

Generate a data set by sampling $\varphi \sim \mathcal{U}[0, 2\pi]$ and

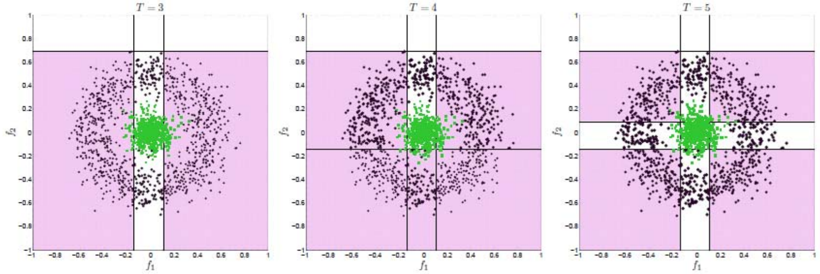
$$r = \begin{cases} \mathcal{N}(r; 0, 0.1) & \text{positive class} \\ \mathcal{N}(r; 0.5, 0.1) & \text{negative class} \end{cases} \qquad \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} r\cos(\varphi) \\ r\sin(\varphi) \end{pmatrix}$$



(a) Polar data (b) $T = 1$ (c) $T = 2$

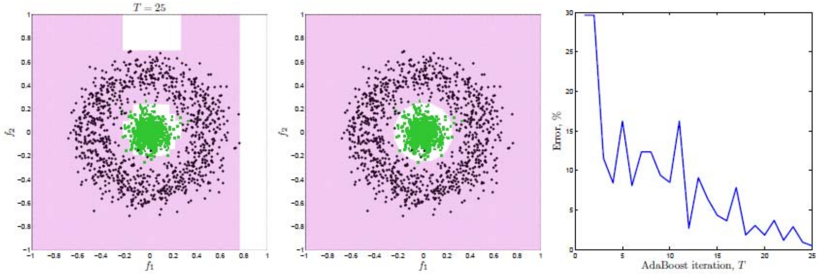
The data and the first two weak classifiers.

An Illustrative Example (II/III)



(d) $T = 3$ (e) $T = 4$ (f) $T = 5$

An Illustrative Example (III/III)



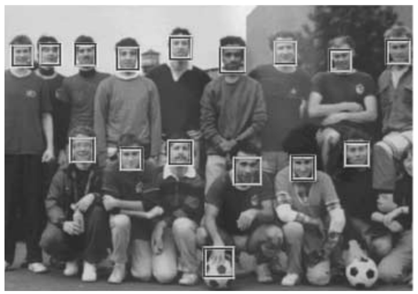
(g) Decision region (h) Probabilistic decision region. (i) Error

This example was borrowed from Karl Granström's licentiate thesis (more examples are provided in Chapter 4)

Karl Granström, *Loop detection and extended target tracking using laser data*, Linköping Studies in Science and Technology, Thesis No. 1465, February, 2011.

`http://liu.diva-portal.org/smash/record.jsf?pid=diva2:389565`

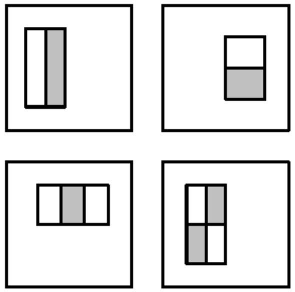
AdaBoost Example 1 - Face Detection (I/III)



Viola, P. A. and Jones, M. J. (2004) Robust real-time face detection. *International Journal of Computer Vision*, 57(2):137-154.

Recall example from lecture 1 and another solution using neural networks briefly discussed during lecture 4.

AdaBoost Example 1 - Face Detection (II/III)



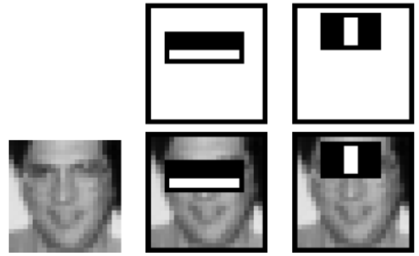
Copyright Springer (2004)

Very simple features (inputs) x_n^j are used. The feature is simply computed by adding the pixel values in the gray area and subtracting the pixel values in the white areas.

This can be done extremely fast using so called integral images (a.k.a. summed area table).

Drawback: the training time can be quite long.

AdaBoost Example 1 - Face Detection (III/III)



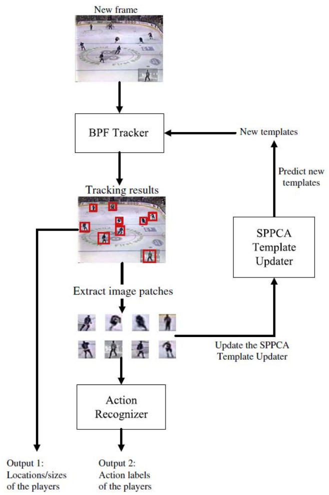
Copyright Springer (2004)

Interpretation of the two first features selected by AdaBoost:

The first feature measures the intensity difference between the eyes and the cheeks.

The second feature measures the intensity difference between the eyes and the bridge of the nose.

AdaBoost Example 2 - Multiple Target Tracking



Copyright Springer (2009)

Makes use of many of the methods we have learned about in this course (such as AdaBoost, EM, PPCA, probabilistic graphical models, logistic regerssion) and particle filtering.

Lu, W.-L., Okuma, K. and Little, J. J. Tracking and Recognizing Actions of Multiple Hockey Players using the Boosted Particle Filter *Image and Vision Computing*, 27(1–2):189–205, 2009.

Probabilistic Graphical Models - Motivation

"Graphical models bring together graph theory and probability theory in a powerful formalism for multivariate statistical modeling." [1]

We can of course always handle probabilistic models using pure algebraic manipulation. Some reasons for using probabilistic graphical models,

1. A simple way to visualize the structure of a probabilistic model.
2. Knowledge about model properties directly from the graph.
3. A different way of performing and structuring calculations.

[1] Wainwright, M. J. and Jordan, M. I. Graphical Models, Exponential Families, and Variational Inference, *Foundations and Trends in Machine Learning*, 1(1-2):1–305, 2008.

Probabilistic Graphical Models - Basic Facts (I/II)

A graph $\mathcal{G} = (\mathcal{V}, \mathcal{L})$ consists of

1. a set of nodes $\mathcal{V}$ (a.k.a. vertices) representing the random variables and
2. a set of links $\mathcal{L}$ (a.k.a. edges or arcs) containing elements $(i, j) \in \mathcal{L}$ connecting a pair of nodes $(i, j) \in \mathcal{V}$.

The links describes the probabilistic relations between the random variables (nodes).

Probabilistic graphical model representations,

1. **Bayesian networks** represents a set of random variables and their conditional dependencies via a directed acyclic graph (DAG).
2. **Markov random fields** represents a set of random variables having a Markov property by an undirected graph.

Probabilistic Graphical Models - Basic Facts (II/II)

Define

$$\mathcal{P}(j) \triangleq \{i \in \mathcal{V} \mid (i, j) \in \mathcal{E}\}$$

denoting the set of parents to node j.

The directed graph describes how the joint distribution $p(x)$ factors into a product of factors $p(x_i \mid x_{\mathcal{P}(i)})$ only depending on a subset of the variables,

$$p(x_{\mathcal{V}}) = \prod_{i \in \mathcal{V}} p(x_i \mid x_{\mathcal{P}(i)}),$$

where $x_{\mathcal{A}}$ denotes the set $\{x_i \mid i \in \mathcal{A}\}$.

Hence, node's value conditioned on its parents is independent of all other ancestors.

Application: Image De-noising

- Suppose we have a noisy image.
- Model the true pixel values as $x_{i,j}$.
- Model the measured image pixel values as

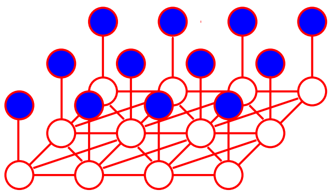
$$y_{i,j} = x_{i,j} + v_{i,j}$$

where $v_{i,j} \sim \mathcal{N}(0, \beta^2)$.

- Choose the energy functions as

$$E_y(x_{i,j}, y_{i,j}) = \frac{1}{\beta^2}(y_{i,j} - x_{i,j})^2$$

$$E_x(x_{i_1,j_1}, x_{i_2,j_2}) = \min\left(\frac{1}{\alpha^2}(x_{i_1,j_1} - x_{i_2,j_2})^2, \gamma\right)$$

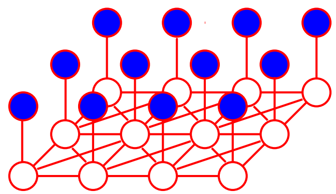


Application: Image De-noising

- The density is then

$$\begin{aligned} -\log p(x_{1:N_x,1:N_y}, y_{1:N_x,1:N_y}) = \sum_{i,j} & E_y(x_{i,j}, y_{i,j}) \\ & + E_x(x_{i,j}, x_{i+1,j+1}) + E_x(x_{i,j}, x_{i-1,j-1}) \\ & + E_x(x_{i,j}, x_{i-1,j+1}) + E_x(x_{i,j}, x_{i+1,j-1}) + C \end{aligned}$$

- If the image is 8 bit grayscale, maximization in general requires the calculation of $256^{(N_x \times N_y)}$ different combinations.
- We instead maximize w.r.t. only one pixel keeping the others fixed at their last values.
- This is called Iterative Conditional Modes (ICM).



A Few Concepts to Summarize Lecture 6

Weak classifier: (a.k.a. base classifier) A classifier that is just slightly better than random guessing.

Boosting: Trains a sequence of M weak classifiers (models), where the error function used to train a certain model depends on the performance of the previous weak classifiers. All weak learners are then combined to a final strong classifier.

Probabilistic graphical model: Offers a compact way of encoding the conditional dependency structure of a set of random variables.

Bayesian network: A probabilistic graphical model that represents a set of random variables and their conditional dependencies via a directed acyclic graph (DAG).

Markov random field: A probabilistic graphical model that represents a set of random variables having a Markov property by an undirected graph.