CSC2515 Fall 2008
Introduction to Machine Learning

Lecture 11a
Boosting and Naïve Bayes

All lecture slides will be available as .ppt, .ps, & .htm at
www.cs.toronto.edu/~hinton

Many of the figures are provided by Chris Bishop
from his textbook: "Pattern Recognition and Machine Learning"

A commonsense way to use limited computational resources

- First train a model on all of the data
  - Lets assume it get the great majority of the cases right.
- Then train another model on all the cases the first model got wrong plus an equal number that it got right.
  - This focusses the resources on modelling the hard cases.
- Train a third model focusssing on cases that either or both previous models got wrong.
  - Then use a simple committee of the three models
- This is quite effective for learning to recognize handwritten digits, but it is also very heuristic.
  - Can we give it a theoretical foundation?

Making weak learners stronger

- Suppose you have a weak learning module (a "base classifier") that can always get 0.5+epsilon correct when given a two-way classification task
  - That seems like a weak assumption but beware!
- Can you apply this learning module many times to get a strong learner that can get close to zero error rate on the training data?
  - Theorists showed how to do this and it actually led to an effective new learning procedure (Freund & Shapire, 1996)

Boosting (ADAboost)

- First train the base classifier on all the training data with equal importance weights on each case.
- Then re-weight the training data to emphasize the hard cases and train a second model.
  - How do we re-weight the data? Maybe some theory can help.
- Keep training new models on re-weighted data
- Finally, use a weighted committee of all the models for the test data.
  - How do we weight the models in the committee? Maybe some theory can help.

How to train each classifier

input is $\mathbf{x}$, *output is* $y(\mathbf{x}) \in \{1,-1\}$

tarrget is $t \in \{1,-1\}$,

weight on case n for classifier m is w_n^m

Cost function for classifier m:

$$J_m = \sum_{n=1}^{N} w_n^m \left[y_m(\mathbf{x}_n) \neq t_n \right] = \sum \textit{weighted errors}$$

1 if error,
0 if correct

How to weight each training case for classifier m

$$\textit{Let} \quad \varepsilon_m = \frac{J_m}{\sum_n w_n^m}$$

← weighted error rate of classifier

$$\textit{Let} \quad \alpha_m = \ln\left(\frac{1-\varepsilon_m}{\varepsilon_m}\right)$$

← This is the quality of the classifier. It is zero if the classifier has weighted error rate of 0.5 and infinity if the classifier is perfect

$$w_n^{m+1} = w_n^m \exp\{ \alpha_m [y(x_n) \neq t_n] \}$$

How to make predictions using a committee of classifiers

- Weight the binary prediction of each classifier by the quality of that classifier:

$$Y_M(\mathbf{x}) = sign\left(\sum_{m=1}^{M} \alpha_m \, y_m(\mathbf{x}) \right)$$

This completely ignores the confidence of the classifier which is very un-bayesian and probably inefficient, but may make it easier to prove things.

An alternative derivation of ADAboost

- Just write down the right cost function and let each classifier minimize it greedily (Freidman et. al. 2000).

$$E = \sum_{n=1}^{N} \exp\{ -t_n \, f_m(\mathbf{x}_n) \}$$

← the exponential loss for classifier m

$$f_m(\mathbf{x}) = \frac{1}{2} \sum_{l=1}^{l=m} \alpha_l \, y_l(\mathbf{x})$$

← real-valued prediction by committee of models up to m

Learning classifier m using exponential loss

$$E = \sum_{n=1}^{N} \exp\{-t_n \; f_m(\mathbf{x}_n)\}$$

$$f_m(\mathbf{x}) = \frac{1}{2}\sum_{l=1}^{l=m} \alpha_l y_l(\mathbf{x}) = \frac{1}{2}\alpha_m \; y_m(\mathbf{x}) \; + \; \frac{1}{2}\sum_{l=1}^{l=m-1} \alpha_l y_l(\mathbf{x})$$

$$E_{relevant} = \sum_{n=1}^{N} \exp\left\{-t_n \; f_{m-1}(\mathbf{x}_n) - \frac{1}{2} t_n \; \alpha_m \; y_m(\mathbf{x}_n)\right\}$$

$$= \sum_{n=1}^{N} w_n^m \exp\left\{-\frac{1}{2} t_n \; \alpha_m \; y_m(\mathbf{x}_n)\right\}$$

Re-writing the part of the exponential loss that is relevant when fitting classifier m

$$E_{relevant} - = \sum_{n=1}^{N} w_n^m \exp\left\{-t_n \frac{\alpha_m}{2} \; y_m(\mathbf{x}_n)\right\}$$

$$= e^{-\frac{\alpha_m}{2}} \sum_{right} w_n^m \; + \; e^{+\frac{\alpha_m}{2}} \sum_{wrong} w_n^m$$

unmodifiable

$$= \left(e^{\frac{\alpha_m}{2}} - e^{-\frac{\alpha_m}{2}}\right) \sum_{n} w_n^m [t_n \neq y_m(\mathbf{x}_n)] \; + \; e^{-\frac{\alpha_m}{2}} \sum_{n} w_n^m$$

multiplicative constant — wrong cases

An impressive example of boosting

- Viola and Jones created a very fast face detector that can be scanned across a large image to find the faces.
- The base classifier just compares the total intensity in two rectangular pieces of the image.
  - There is a neat trick for computing the total intensity in a rectangle in a few operations.
    - So its easy to evaluate a huge number of base classifiers and they are very fast at runtime.
  - We add classifiers greedily based on their quality on the weighted training cases.