Slide 0

Lecture 2.4

Committees of Neural Networks and Boosting

Jonathan Shapiro

Department of Computer Science, University of Manchester

February 2, 2003

Slide 1

Committees of Trained Neural Networks

Suppose you have trained a number of neural network on the same problem, but from different initial conditions.

(For example, you did cross-validation, or tried to create the network several times.)

If you take one network and discard the rest,

- The computational effort used to make the others is wasted;
- If you take the one with the lowest validation error, it might not have the lowest true error.

What to do?

Slide 2

Using all of the networks

- You can combine them into one “committee machine”
- Combine the outputs of all networks to produce output.
- Methods of computing the output of the committee

  Voting: choose the majority classification.

  Averaging: the output of the committee is the average of all of the outputs.

Slide 3

This always improves the expected performance

- Why this leads to improvement.
  - Variance reduction
  - Let ε_i be the expected deviation of the ith network from the true output. L networks in all.
  - Expected MSE if a network is chosen at random is the average of this,

$$\frac{1}{L}\sum_{i}^{L}\varepsilon_i^2$$

  - The expected MSE for the committee machine is

$$\left(\frac{1}{L}\sum_{i}^{L}\varepsilon_i\right)^2,$$

Slide 4

and by Cauchy's inequality,

$$\left(\frac{1}{L}\sum_i^L \varepsilon_i\right)^2 \leq \frac{1}{L}\sum_i^L \varepsilon_i^2$$

- Unfortunately, the improvement is typically small.
  - If there were no correlations in where the different networks made errors the reduction in expected error would be $1/L$.

$$\left(\frac{1}{L}\sum_i^L \varepsilon_i\right)^2 = \frac{1}{L^2}\sum_i^L \varepsilon_i^2.$$

  - Typically, however, this is not the case. The networks were trained on the same data and are similar networks, they tend to make errors on similar data.

We will explore two methods of improving this.

Slide 5

Boosting

Boosting involves training different components of the committee on different distributions or subsets of the data. Iterated:

- Learner A trained on data;
- data set for learner B contains more data which is hard for learner A
- Learner C contains most data which was hard for both A and B, less which was hard for only one, even less which both A and B got correct.
- etc.

Kearns and Valient (1988) posed the question, can a provable weak learner be "boosted" into an arbitrarily accurate strong learner.

Schapire (1990) and Freund (1991) showed the first algorithms.

Slide 6

Weak versus Strong Learning.

Strong learning: Often we look for a learning model which can achieve arbitrarily small error rates ε most of the time independently from the distribution of input patterns.

Weak learning: Weak learning models are only guaranteed to achieve results slightly better than a random guesser.

For boosting to work, we only need a weak learning model.

Slide 7

Boosting by Filtering

Assumptions: A large pool of examples. (An unrealistic assumption. We will remove it later.)

Three experts: Each can achieve an error rate of $\varepsilon < 0.5$.

Slide 8

Procedure

- Train expert one on N_1 examples.
- Build a training set for the second expert from the unused patterns by filtering through the first expert.
  - With probability 0.5, put in a pattern which the first expert correctly classifies;
  - with probability 0.5, put in a pattern which the first expert gets wrong.
  - Until you have a new training set of size N_1.
  - Note that expert 1 has an error rate of 0.5 on the second training set.
  - Building the second training set takes N_2 patterns, which depends on the error rate of expert 1.
- Train expert 2 on the second training set.

Slide 9

- Build a training set for the third expert from the unused patterns by filtering through the first two experts
  - Put a pattern into the third training set if the first two experts disagree on its classification.
  - Discard a pattern on which they agree.
  - Until you have a third training set of size N_1.
  - The committee of experts 1 and 2 have an error rate of 0.5 on training set 3.
  - Building training set three required N_3 patterns to pass through the filter.

Slide 10

Performance of the boosted learners

Performance of unboosted learner: is ε, by assumption.

Boosted committee: Each expert is making independent errors, by construction. Committee makes errors only if two out of three make a mistake.

$$\begin{aligned} \text{probability } 2/3 \text{ make an error} &= 3\varepsilon^2(1-\varepsilon) \\ \text{probability all } 3 \text{ make an error} &= \varepsilon^3 \\ \text{total error probability} &= 3\varepsilon^2 - 2\varepsilon^3 \\ &< \varepsilon \end{aligned}$$

For any $\varepsilon < 0.5$ the learning rate is improved.

If that is not low enough error rate, take the three learners as an expert, and do it all again.

Slide 11

Data use

The size of the data set must be arbitrarily large.

- Although, only $3N_1$ data used to make the three experts, $N_1 + N_2 + N_3$ examples required to make the training sets, and this is much larger.
- If we want to boost further, much more data is required.
- This makes the method of limit utility.

Now we look at a boosting method which does not require so much data.

Slide 12

Problems 1

problem 1.1 *Let $N_1 = 100$. Suppose all experts have error rates of 25% on their respective data sets. Compute N_2 and N_3 to find the total amount of data used. Compute the final error rate of boosted machine.*

Slide 13

AdaBoost

Freund and Schapire, 1996 Basic idea

- Introduce a probability distribution on the training examples. It will be $\rho_n(x)$ in the *n*th step [$\rho_1(x)$ is a uniform distribution, say].
- Use a sample of training data to train the *n*th expert.
- Update the distribution, downweighting examples which this expert correctly classifier. Go to step 2, and repeat for a fixed number of iterations.
- The final model is a weighted sum of all of the experts, weighted by their accuracy on the training data.

Slide 14

The algorithm (two class case)

Input: N training examples; weak learning model, integer T specifying number of iterations,

Initialize: $\rho_1(x) = 1/N$

Computation: For $n = 1, \ldots, T$, do

1. Generate expert E_n by learning on a sample from $\rho_n(x)$
2. Find the error e_n of expert E_n on distribution ρ_n
3. Set $\alpha_n = \frac{1}{2} \log\left(\frac{1-e_n}{e_n}\right)$.
4. Update the distribution

$$\rho_{n+1}(x) \propto \rho_n(x) \begin{cases} \exp \alpha_n & \text{if } E_n \text{ is incorrect on } x \\ \exp -\alpha_n; & \text{if } E_n \text{ is correct on } x \end{cases}$$

This must be normalized.

Slide 15

Coefficient of each expert: each expert is weighted by α.

Output: The final model is a weighted majority of the experts.

- Simplest if the learner outputs are $y_n = \pm 1$. Then,

$$Y = \sum_n \alpha_n y_n; \quad (1)$$

$$\text{classification} = \text{sign} H. \quad (2)$$

- If learner outputs are $h = 0, 1$,

$$H = \sum_n \alpha_n h_n; \quad (3)$$

$$\text{class } 1 \text{ if } \quad Y > \frac{1}{2} \sum_n \alpha_n. \quad (4)$$

Slide 16

Note

If you are following in Haykin, he defines in terms of a parameter $\beta_n = \exp -2\alpha$.

Slide 17

Problems 2

problem 2.2 *Show that the training error of expert* E_n *on distribution* ρ_{n+1} *is exactly* 0.5.

Slide 18

Example 1: Perceptrons learning XOR

- Train a perceptron on XOR using Perceptron Learning Algorithm. Will always get one pattern wrong.
- Notation: P_1 is perceptron 1; W_1 is the one it gets wrong. Assume that is $(0,0)$.
- $e_1 = 1/4$; Prob of misclassified data:$3/4$; prob. of correctly classified data: $1/4$.
- ρ_1 has 3 copies of $(0,0)$. Next perceptron gets another one wrong. Assume it is $(1,0)$.
- $e_2 = 1/6$; prob of misclassified data gets multiplied by $5/6$; prob of correctly classified data gets multiplied by $1/6$.
- ρ_2 has 3 copies of $(0,0)$, 5 copies of $(1,0)$. Next perceptron gets another one wrong. Assume it is $(1,1)$.

Slide 19

- $e_3 = 1/10$; prob of misclassified data gets multiplied by $8/9$; prob of correctly classified data gets multiplied by $1/9$.

Stop here. What is the classification?

Pattern $(0,0)$**:** weight of patterns which say output 0 is $(\log 5 + log 9)/2$; weight of patterns which say output is 1 is $(\log 3)/2$. Thus, output is 0. *correct*.

Pattern $(1,0)$**:** weight of patterns which say output 0 is $(\log 5)/2$; weight of patterns which say output is 1 is $(\log 3 + \log 9)/2$. Thus, output is 1. *correct*.

Etc. All other patterns are correctly classified.

Slide 20

Generalization properties

- AdaBoost seems to decrease the generalization continuously during iterations.
- No overtraining. (see figure)
- Remarkable since model complexity is ever increasing.

Slide 21

Why it works

Boosting does not necessarily decrease the training error. What can be shown in the following. Using ± 1 representation, the following is a bound for the training error,

$$J(H_m) = \frac{1}{N} \sum_{i=1}^{N} \exp(-y_i H_m(x_i)), \tag{5}$$

and that this bound always decreases

$$J(H_m) \geq J(H_{m+1}). \tag{6}$$

Thus, the *margin* of the classifier decreases which each iteration.

Slide 22

Problems 3

problem 3.3 *Using $y = \pm 1$ representation for the classifier outputs, show that the formula for α_k can be derived by finding the k which minimizes the following function,*

$$G(\alpha) = \log \left[\sum_i \rho_k(i) \exp\left(-\alpha y_i h_k(x_i)\right) \right]. \tag{7}$$

To do this, set the derivative to zero and show that the resulting equation is equivalent to the condition that the training error of classifier k with regard to distribution ρ_{k+1} is 0.5.

problem 3.4 *Use the result from above to show that the log of the bound*

Slide 23

decreases with each iteration. I.e, with

$$G(H_m) = \log \left[\sum_{i=1}^{N} \exp\left(-y_i H_m(x_i)\right) \right], \tag{8}$$

$$G(H_m) \geq G(H_{m+1}). \tag{9}$$

Since the $\log$ *function is monotonic, the decrease of the* $\log$ *implies the decrease of the function itself.*

Slide 24

Mixture of Experts Model

(Jacobs and Jordan, 1992)

In the final approach, we consider a model in which

- The outputs are a weighted sum of the outputs of the expert.
- Those weights learn during learning of the experts.
- The weights depend on the pattern being classified. So the system tries to learn which expert is good at which pattern.

Slide 25

Architecture

Experts: Ordinary neural networks which would learn to minimize MSE. Each expert produces output y_k (from kth expert).

gating network: A softmax output. One output for each expert (output g_k gates the kth output).

Total output of committee: $y = g_1 y_1 + \ldots + g_n y_n$.

Slide 26

Learning Rule

Let,

$$h_i = \frac{g_i \exp\left[-\frac{1}{2}|y_i - d|^2\right]}{\sum_j g_i \exp\left[-\frac{1}{2}|y_j - d|^2\right]}$$

where y_i is the output from the ith expert, d is the desired output, and g_i is the ith output from the gate.

- h_i is like the extent to which expert i is responsible for the current pattern. It is small if this expert gets a large error on this pattern, or if the gate network is sending the responsibility elsewhere (i.e. g_i is small).
- Learning rules:

Slide 27

Expert i: learns via backpropagation, but with a learning rate proportional to h_i. Those with smallest errors and largest assignment from the gating network learn the most.

Gating network: learns to set g_i towards the current h_i. Gives responsibility to experts which have a low error on the current pattern, and have current responsibility for the current pattern.

Like an EM algorithm.

Slide 28

Results

1. The mixture of experts can partition the space up, so committees of simple experts can solve problems too complex for any one of the experts.
   - E.g. can adapt local linear models, using simple linear networks.
2. Can make a mixture of very different experts, capable of different types of tasks. The gating network learns to appropriately partition the task to the expert in some examples. ‘
3. Can make a hierarchical mixture of experts; experts are mixtures of experts.
   - Subdivides the input space hierarchically.

Slide 29

Application to task distribution

One view of this is that if the experts are networks with different biases, this approach will allocate the task to the appropriate expert.

- Example: “What-where” task — a system which simultaneously finds an object and identifies it. (Find a face in a scene; identify the person from the face).
  - What task: global, synthetic, requires long-range, coarse connections;
  - where task: local, analytic, requires local, fine-scaled connections.
  - Related to hemisphere lateralization in mammalian visual tasks by Jacobs and Kosslyn (1994).
- Example: Hemisphere lateralization in human language. (See book, “Rethinking innateness”).

Slide 30

Conclusions

- Simple committees always give some improvement.
- Improvement can be substantial when efforts are made to insure that the experts make errors on uncorrelated pattern sets.
  - Boosting can insure this.
  - Allowing the committee to consist of weighted sums of experts and allowing the weights to adapt allows the system to match the input patterns with the appropriate expert.