Lecture 22

Combining Classifiers:
Boosting the Margin and Mixtures of Experts

Thursday, November 11, 1999

William H. Hsu
Department of Computing and Information Sciences, KSU
http://www.cis.ksu.edu/~bhsu

Readings:
"Bagging, Boosting, and *C4.5*", Quinlan
Section 5, "*MLC++* Utilities 2.0", Kohavi and Sommerfield

Lecture Outline

- Readings: Section 5, *MLC++* 2.0 Manual [Kohavi and Sommerfield, 1996]
- Paper Review: "Bagging, Boosting, and *C4.5*", J. R. Quinlan
- Boosting the Margin
  - Filtering: feed examples to trained inducers, use them as "sieve" for consensus
  - Resampling: *aka* subsampling ($S[i]$ of fixed size m' *resampled* from D)
  - Reweighting: fixed size $S[i]$ containing *weighted examples* for inducer
- Mixture Model, *aka* Mixture of Experts (ME)
- Hierarchical Mixtures of Experts (HME)
- Committee Machines
  - Static structures: *ignore input signal*
    - Ensemble averaging (single-pass: weighted majority, bagging, stacking)
    - Boosting the margin (some single-pass, some multi-pass)
  - Dynamic structures (multi-pass): *use input signal to improve classifiers*
    - Mixture of experts: training in combiner inducer (*aka* gating network)
    - Hierarchical mixtures of experts: hierarchy of inducers, combiners

Quick Review: Ensemble Averaging

- Intuitive Idea
  - Combine experts (*aka* prediction algorithms, classifiers) using combiner function
  - Combiner may be weight vector (WM), vote (bagging), trained inducer (stacking)
- Weighted Majority (WM)
  - Weights each algorithm *in proportion to its training set accuracy*
  - Use this weight in performance element (and on test set predictions)
  - Mistake bound for WM
- Bootstrap Aggregating (Bagging)
  - Voting system for collection of algorithms
  - Training set for each member: sampled with replacement
  - Works for unstable inducers (search for h sensitive to perturbation in D)
- Stacked Generalization (*aka* Stacking)
  - Hierarchical system for combining inducers (ANNs or other inducers)
  - Training sets for "leaves": sampled with replacement; combiner: validation set
- Single-Pass: Train Classification and Combiner Inducers Serially
- Static Structures: Ignore Input Signal

Boosting: Idea

- Intuitive Idea
  - Another type of static committee machine: can be used to improve *any* inducer
  - Learn set of classifiers from D, but reweight examples to *emphasize misclassified*
  - Final classifier $\leftarrow$ weighted combination of classifiers
- Different from Ensemble Averaging
  - WM: all inducers trained on same D
  - Bagging, stacking: training/validation partitions, i.i.d. *subsamples* $S[i]$ of D
  - Boosting: data sampled according to *different distributions*
- Problem Definition
  - Given: collection of multiple inducers, large data set or example stream
  - Return: combined predictor (trained committee machine)
- Solution Approaches
  - Filtering: use weak inducers in cascade to filter examples for downstream ones
  - Resampling: reuse data from D by subsampling (don't need huge or "infinite" D)
  - Reweighting: reuse $x \in D$, but measure error over weighted x

Boosting: Procedure

- Algorithm *Combiner-AdaBoost* (*D, L, k*) // Resampling Algorithm
  - $m \leftarrow D.size$
  - FOR $i \leftarrow 1$ TO m DO // initialization
    Distribution[i] $\leftarrow 1 / m$ // subsampling distribution
  - FOR $j \leftarrow 1$ TO k DO
    - $P[j] \leftarrow L[j]$.*Train-Inducer* (*Distribution, D*) // assume $L[j]$ identical; $h_j \equiv P[j]$
    - *Error*[j] $\leftarrow$ *Count-Errors*($P[j]$, *Sample-According-To* (*Distribution, D*))
    - $\beta[j] \leftarrow$ *Error*[j] / (1 - *Error*[j])
    - FOR $i \leftarrow 1$ TO m DO // update distribution on D
      Distribution[i] $\leftarrow$ *Distribution*[i] * (($P[j](D[i]) = D[i]$.*target*) ? $\beta[j]$: 1)
    - *Distribution.Renormalize* () // Invariant: *Distribution* is a pdf
  - RETURN (*Make-Predictor* (P, D, β))
- Function *Make-Predictor* (P, D, β)
  - // *Combiner*(x) = $\text{argmax}_{v \in V} \sum_{j:P[j](x) = v} \lg (1/\beta[j])$
  - RETURN (fn $x \Rightarrow$ *Predict-Argmax-Correct* (P, D, x, fn $\beta \Rightarrow \lg (1/\beta)$))

Boosting: Properties

- Boosting in General
  - Empirically shown to be effective
  - Theory still under development
  - Many variants of boosting, active research (see: references; current ICML, COLT)
- Boosting by Filtering
  - Turns weak inducers into strong inducer (committee machine)
  - Memory-efficient compared to other boosting methods
  - Property: improvement of weak classifiers (trained inducers) guaranteed
    - Suppose 3 experts (subhypotheses) each have error rate $\varepsilon < 0.5$ on $D[i]$
    - Error rate of committee machine $\leq g(\varepsilon) = 3\varepsilon^2 - 2\varepsilon 3$
- Boosting by Resampling (*AdaBoost*): Forces $Error_{\mathcal{D}}$ toward $Error_D$
- References
  - Filtering: [Schapire, 1990] - MLJ, 5:197-227
  - Resampling: [Freund and Schapire, 1996] - ICML 1996, p. 148-156
  - Reweighting: [Freund, 1995]
  - Survey and overview: [Quinlan, 1996; Haykin, 1999]

Mixture Models: Idea

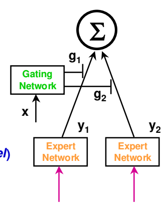
- **Intuitive Idea**
  - Integrate knowledge from multiple experts (or data from multiple sensors)
    - Collection of inducers organized into committee machine (e.g., modular ANN)
    - Dynamic structure: *take input signal into account*
  - References
    - [Bishop, 1995] (Sections 2.7, 9.7)
    - [Haykin, 1999] (Section 7.6)
- **Problem Definition**
  - Given: collection of inducers ("experts") *L*, data set *D*
  - Perform: supervised learning using inducers and self-organization of experts
  - Return: committee machine with trained gating network (combiner inducer)
- **Solution Approach**
  - Let combiner inducer be generalized linear model (e.g., threshold gate)
  - Activation functions: linear combination, vote, "smoothed" vote (softmax)

Mixture Models: Procedure

- **Algorithm *Combiner-Mixture-Model* (*D, L, Activation, k*)**
  - *m* ← *D.size*
  - FOR *j* ← 1 TO *k* DO // initialization
    w[*j*] ← 1
  - UNTIL the termination condition is met, DO
    - FOR *j* ← 1 TO *k* DO
      P[*j*] ← *L*[*j*].*Update-Inducer* (*D*) // single training step for *L*[*j*]
    - FOR *i* ← 1 TO *m* DO
      Sum[*i*] ← 0
      FOR *j* ← 1 TO *k* DO *Sum*[*i*] += P[*j*](*D*[*i*])
      Net[*i*] ← *Compute-Activation* (*Sum*[*i*]) // compute $g_j \equiv Net[i][j]$
      FOR *j* ← 1 TO *k* DO *w*[*j*] ← *Update-Weights* (*w*[*j*], *Net*[*i*], *D*[*i*])
  - RETURN (*Make-Predictor* (*P*, *w*))
- ***Update-Weights*: Single Training Step for Mixing Coefficients**

Mixture Models: Properties

- **Unspecified Functions**
  - *Update-Inducer*
    - Single training step for each expert module
    - e.g., ANN: one backprop cycle, *aka* epoch
  - *Compute-Activation*
    - Depends on ME architecture
    - Idea: smoothing of "winner-take-all" ("hard" max)
    - Softmax activation function (*Gaussian mixture model*)

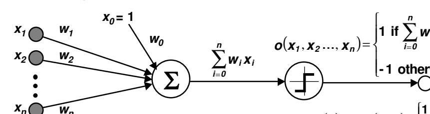


$$g_l = \frac{e^{w_l \cdot \vec{x}}}{\sum_{j=1}^{k} e^{w_j \cdot \vec{x}}}$$

- **Possible Modifications**
  - Batch (as opposed to online) updates: lift *Update-Weights* out of outer FOR loop
  - Classification learning (versus concept learning): multiple y_j values
  - Arrange gating networks (combiner inducers) in *hierarchy* (HME)

Generalized Linear Models (GLIMs)

- **Recall: Perceptron (Linear Threshold Gate) Model**



$$\sum_{i=0}^{n} w_i x_i$$

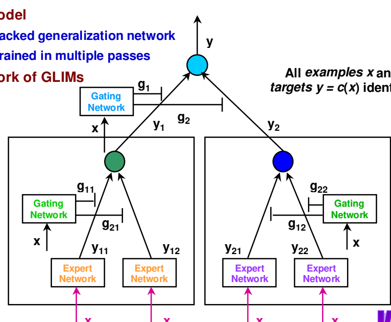
$$o(x_1, x_2 \ldots, x_n) = \begin{cases} 1 \text{ if } \sum_{i=0}^{n} w_i x_i > 0 \\ -1 \text{ otherwise} \end{cases}$$

$$\text{Vector notation: } o(\vec{x}) = sgn(\vec{x}, \vec{w}) = \begin{cases} 1 \text{ if } \vec{w} \cdot \vec{x} > 0 \\ -1 \text{ otherwise} \end{cases}$$

- **Generalization of LTG Model [McCullagh and Nelder, 1989]**
  - Model parameters: connection weights as for LTG
  - Representational power: depends on transfer (activation) function
- **Activation Function**
  - Type of mixture model depends (in part) on this definition
  - e.g., *o*(*x*) could be *softmax* (*x* · *w*) [Bridle, 1990]
    - *NB*: *softmax* is computed across *j* = 1, 2, …, *k* (cf. "hard" max)
    - Defines (*multinomial*) pdf over experts [Jordan and Jacobs, 1995]

Hierarchical Mixture of Experts (HME): Idea

- **Hierarchical Model**
  - Compare: stacked generalization network
  - Difference: trained in multiple passes
- **Dynamic Network of GLIMs**



All *examples x* and *targets y* = *c*(*x*) identical

Hierarchical Mixture of Experts (HME): Procedure

- **Algorithm *Combiner-HME* (*D, L, Activation, Level, k, Classes*)**
  - *m* ← *D.size*
  - FOR *j* ← 1 TO *k* DO *w*[*j*] ← 1 // initialization
  - UNTIL the termination condition is met DO
    - IF *Level* > 1 THEN
      FOR *j* ← 1 TO *k* DO
      P[*j*] ← *Combiner-HME* (*D*, *L*[*j*], *Activation*, *Level* - 1, *k*, *Classes*)
    - ELSE
      FOR *j* ← 1 TO *k* DO *P*[*j*] ← *L*[*j*].*Update-Inducer* (*D*)
    - FOR *i* ← 1 TO *m* DO
      Sum[*i*] ← 0
      FOR *j* ← 1 TO *k* DO
      Sum[*i*] += P[*j*](*D*[*i*])
      Net[*i*] ← *Compute-Activation* (*Sum*[*i*])
      FOR *l* ← 1 TO *Classes* DO *w*[*l*] ← *Update-Weights* (*w*[*l*], *Net*[*i*], *D*[*i*])
  - RETURN (*Make-Predictor* (*P*, *w*))

Hierarchical Mixture of Experts (HME): Properties

- Advantages
  - Benefits of ME: base case is single level of expert and gating networks
  - More combiner inducers ⇒ more capability to decompose complex problems
- Views of HME
  - Expresses divide-and-conquer strategy
    - Problem is distributed across subtrees "on the fly" by combiner inducers
    - Duality: data fusion ⇔ problem redistribution
    - Recursive decomposition: until good fit found to "local" structure of *D*
  - Implements soft decision tree
    - Mixture of experts: 1-level decision tree (decision stump)
    - Information preservation compared to traditional (hard) decision tree
    - Dynamics of HME improves on greedy (high-commitment) strategy of decision tree induction

Training Methods for Hierarchical Mixture of Experts (HME)

- Stochastic Gradient Ascent
  - Maximize log-likelihood function $L(\Theta) = \lg P(D \mid \Theta)$
  - Compute

$$\frac{\partial L}{\partial w_{ij}}, \frac{\partial L}{\partial a_j}, \frac{\partial L}{\partial a_{ij}}$$

  - Finds MAP values
    - Expert network (leaf) weights w_{ij}
    - Gating network (interior node) weights at lower level (a_{ij}), upper level (a_j)
- Expectation-Maximization (EM) Algorithm
  - Recall definition
    - Goal: maximize incomplete-data log-likelihood function $L(\Theta) = \lg P(D \mid \Theta)$
    - Estimation step: calculate *E*[*unobserved variables* | Θ], assuming current Θ
    - Maximization step: update Θ to maximize $E[\lg P(D \mid \Theta)]$, *D* ≡ *all variables*
  - Using EM: estimate with gating networks, then adjust $\Theta \equiv \{w_{ij}, a_{ij}, a_j\}$

Methods for Combining Classifiers: Committee Machines

- Framework
  - Think of collection of trained inducers as *committee of experts*
  - Each produces predictions given input ($s(D_{test})$, i.e., new *x*)
  - Objective: combine predictions by vote (subsampled D_{train}), learned weighting function, or more complex combiner inducer (trained using D_{train} or $D_{validation}$)
- Types of Committee Machines
  - Static structures: based only on *y* coming out of local inducers
    - Single-pass, same data or independent subsamples: WM, bagging, stacking
    - Cascade training: *AdaBoost*
    - Iterative reweighting: boosting by reweighting
  - Dynamic structures: take *x* into account
    - Mixture models (mixture of experts *aka* ME): one combiner (gating) level
    - Hierarchical Mixtures of Experts (HME): multiple combiner (gating) levels
    - Specialist-Moderator (SM) networks: partitions of *x* given to combiners

Comparison of Committee Machines

	Aggregating Mixtures			Partitioning Mixtures	
	Stacking	Bagging	SM Networks	Boosting	HME
Sampling Method	Round-robin (cross-validation)	Random, with replacement	Attribute partitioning/ clustering	Least squares (proportionate)	Linear gating (proportionate)
Splitting of Data	Length-wise	Length-wise	Length-wise	Width-wise	Width -wise
Guaranteed improvement of weak classifiers?	No	No	No	Yes	No
Hierarchical?	Yes	No, but can be extended	Yes	No	Yes
Training	Single bottom-up pass	N/A	Single bottom-up pass	Multiple passes	Multiple top-down passes
Wrapper or mixture?	Both	Wrapper	Mixture, can be both	Wrapper	Mixture, can be both

Terminology

- Committee Machines *aka* Combiners
- Static Structures
  - Ensemble averaging
    - Single-pass, separately trained inducers, common input
    - Individual outputs combined to get scalar output (e.g., linear combination)
  - Boosting the margin: separately trained inducers, *different input distributions*
    - Filtering: feed examples to trained inducers (weak classifiers), pass on to next classifier *iff* conflict encountered (consensus model)
    - Resampling: *aka* subsampling ($S[i]$ of fixed size m' *resampled* from *D*)
    - Reweighting: fixed size $S[i]$ containing *weighted examples* for inducer
- Dynamic Structures
  - Mixture of experts: training in combiner inducer (*aka* gating network)
  - Hierarchical mixtures of experts: hierarchy of inducers, combiners
- Mixture Model, *aka* Mixture of Experts (ME)
  - Expert (classification), gating (combiner) inducers (modules, "networks")
  - Hierarchical Mixtures of Experts (HME): multiple combiner (gating) levels

Summary Points

- Committee Machines *aka* Combiners
- Static Structures (Single-Pass)
  - Ensemble averaging
    - For improving weak (especially unstable) classifiers
    - e.g., weighted majority, bagging, stacking
  - Boosting the margin
    - Improve performance of any inducer: weight examples to emphasize errors
    - Variants: filtering (*aka* consensus), resampling (*aka* subsampling), reweighting
- Dynamic Structures (Multi-Pass)
  - Mixture of experts: training in combiner inducer (*aka* gating network)
  - Hierarchical mixtures of experts: hierarchy of inducers, combiners
- Mixture Model (*aka* Mixture of Experts)
  - Estimation of mixture coefficients (i.e., weights)
  - Hierarchical Mixtures of Experts (HME): multiple combiner (gating) levels
- Next Week: Intro to GAs, GP (9.1-9.4, Mitchell; 1, 6.1-6.5, Goldberg)