A Short Course in Computational Learning Theory: ICML '97 and AAAI '97 Tutorials

Michael Kearns

AT&T Laboratories

Outline

- Sample Complexity/Learning Curves: finite classes, Occam's Razor, VC dimension
- Best Experts/Multiplicative Update Algorithms
- Statistical Query Learning and Noisy Data
- Boosting
- Computational Intractability Results
- Fourier Methods and Membership Queries

The Basic Model

- **Target function** $f : X \to Y$ ($Y = \{0, 1\}$ or $\{+1, -1\}$), may come from class F
- Input distribution/density P over X, may be known or arbitrary
- Class of **hypothesis** functions H from X to Y
- Random sample S of m pairs $\langle x_i, f(x_i) \rangle$
- **Generalization Error** $\epsilon(h) = \Pr_P[h(x) \neq f(x)]$

Measures of Efficiency

- Parameters $\epsilon, \delta \in [0,1]$: ask for $\epsilon(h) \leq \epsilon$ with probability at least $1 - \delta$
- Input dimension n
- Target function "complexity" $s(f)$
- **Sample** and **computational** complexity: scale nicely with $1/\epsilon, 1/\delta, n, s(f)$

Variations Ad Infinitum

- Target f from known class F, distribution P arbitrary: **PAC/Valiant model**
- Fixed, known P: **Distribution-specific PAC model**
- Target f arbitrary: **Agnostic model**
- Target f from known class F, add black-box access to f: **PAC model with membership queries**

Sample Complexity/Learning Curves

- Algorithm-specific vs. **general**
- Assume $f \in H$
- How many examples does an **arbitrary consistent** algorithm require to achieve $\epsilon(h) \leq \epsilon$?

The Case of Finite H

- Fix unknown $f \in H$, distribution P
- **Fix** "bad" $h_0 \in H$ $(\epsilon(h_0) \geq \epsilon)$
- Probability h_0 survives m examples $\leq (1-\epsilon)^m$ (**Independence**)
- Probability **some** bad hypothesis survives $\leq |H|(1-\epsilon)^m$ (**Union Bound**)
- Solve $|H|(1-\epsilon)^m \leq \delta$, $m = \Omega((1/\epsilon)\log(|H|/\delta))$ suffices
- Example: $|H| = 2^{n^\alpha}$, $m = \Omega((n^\alpha/\epsilon)\log(1/\delta))$ suffices
- Independent of distribution P

Occam's Razor

- Assume $f \in H_0$, but given m examples h is chosen from H_m, $H_0 \subseteq H_1 \subseteq \cdots \subseteq H_m$
- Same argument establishes that failure probability is bounded by $|H_m|(1-\epsilon)^m$
- Example: $|H_m| = 2^{n^\alpha m^\beta}$, $m = \Omega((n^\alpha m^\beta/\epsilon)\log(1/\delta))$ suffices; or $m = \Omega(((n^\alpha/\epsilon)\log(1/\delta))^{1/(1-\beta)})$
- **Compression** ($\beta < 1$) implies **Learning**

An Example: Covering Methods

- Target f is a conjunction of boolean attributes chosen from $x_1, \ldots, x_n$
- Eliminate any x_i such that $x_i = 0$ in a positive example
- Any surviving x_i "covers" or explains all negative examples with $x_i = 0$
- Greedy approach: always is an x_i covering $(1 - 1/k_{opt})$ of remainder, so cover all negatives in $O(k_{opt} \log(m))$

Infinite H and the VC Dimension

- Still true that probability that bad $h \in H$ survives is $\leq (1-\epsilon)^m$, but now $|H|$ is infinite
- H **shatters** $x_1, \ldots, x_d$ if all 2^d labelings are realized by H
- **VC dimension** d_H: size of the **largest** shattered set

Examples of the VC Dimension

- Finite class H: need 2^d functions to shatter d points, so $d_H \leq \log(|H|)$
- Axis-parallel rectangles in the plane: $d_H = 4$
- Convex d-gons in the plane: $d_H = 2d + 1$
- Hyperplanes in n dimensions: $d_H = n + 1$
- d_H usually "nicely" related to number of parameters, number of operations

The Dichotomy Counting Function

- For any set of points S, define $\Pi_H(S) = \{h \bigcap S : h \in H\}$ and $\Phi_H(m) = \max_{|S| \leq m}\{|\Pi_H(S)|\}$
- $\Phi_H(m)$ counts maximum number of labelings realized by H on m points, so $\Phi_H(m) \leq 2^m$ always
- **Important Lemma:** for $m \geq d_H$, $\Phi_H(m) \leq m^{d_H}$
- Proof is by double induction on m and d_H

Two Clever Tricks

- Idea: in expression $|H|(1-\epsilon)^m$, try to replace $|H|$ by $\Phi_H(2m)$
- **Two-Sample Trick**: probability some bad $h \in H$ survives m examples $\approx$ probability some $h \in H$ makes NO mistakes on first m examples and $\geq \epsilon m/2$ mistakes on second m examples
- **Incremental Randomization Trick**: draw $2m$ sample $S = S_1 \bigcup S_2$ **first**, randomly split into S_1 and S_2 **later**
- Fix one of the $\Phi_H(2m)$ labelings of S which makes at least $\epsilon m/2$ mistakes; probability (wrt split) all mistakes end up in S_2 is exponentially small in m
- Failure probability $\Phi_H(2m)2^{-\epsilon m/2}$, sufficient sample size is $m = \Omega((d_H/\epsilon)\log(1/\epsilon) + (1/\epsilon)\log(1/\delta))$

Extensions and Refinements

- Not all $h \in H$ with $\epsilon(h) \geq \epsilon$ have $\epsilon(h) = \epsilon$; compute distribution-specific **error shells** (Energy vs. Entropy, Statistical Mechanics)
- Replace $\Phi_H(m)$ with distribution-specific **expected** number of dichotomies
- **“Uniform Convergence Happens”**: unrealizable f, squared error, log-loss, ...

Best Experts, Multiplicative Updates, Weighted Majority...

- Assume **nothing** about the data
- Input sequence $x_1, \ldots, x_m$ **arbitrary**
- Label sequence $y_1, \ldots, y_m$ **arbitrary**
- Given x_{m+1}, want to predict y_{m+1}
- What could we hope to say?

A Modest Proposal

- Only compare performance to a fixed collection of “expert advisors” $h_1, \ldots, h_N$
- Expert h_i predicts y_j^i on x_j
- Goal: for **any** data sequence, match the number of mistakes made by the **best** advisor on that sequence
- Idea: to punish us, force adversary to punish all the advisors
- As in sample complexity analysis for probabilistic assumptions, expect performance to degrade for large N

A Simple Algorithm

- Maintain weight w_i for advisor h_i, $w_i = 1/N$ initially
- Predict using weighted majority at each trial
- If we err, and h_i erred, $w_i \leftarrow w_i/2$

A Simple Analysis

- If $W = \sum_{i=1}^{N} w_i$, then when we err $W' \leq (W/2) + (1/2)(W/2) = (3/4)W$
- After k errors, $W \leq 1 \cdot (3/4)^k$
- If expert i makes ℓ errors, then $w_i \geq (1/N)(1/2)^\ell$
- Total weight $\geq w_i$ gives $(3/4)^k \geq (1/N)(1/2)^\ell$
- $k \leq (1/\log(4/3))(\ell + \log(N)) \leq 2.41(\ell + \log(N))$
- Sparse vs. distributed representations

Statistical Query Learning

- Model algorithms that use a random sample **only** to compute statistics
- Replace source of random examples of target f drawn from distribution P by an oracle for **estimating probabilities**
- Learning algorithm submits a **query** $\chi : X \times Y \to \{0, 1\}$
- Example: $\chi(x, y) = 1$ if and only if $x_5 = y$
- Response is an **estimate** of $P_\chi = \Pr_P[\chi(x, f(x)) = 1]$
- Demand that **complexity** of queries and **accuracy** of estimates permit **efficient** simulation from examples
- Captures almost all natural algorithms

Noise Tolerance of SQ Algorithms

- Let source of examples return $\langle x, y \rangle$, where $y = f(x)$ with probability $1 - \eta$ and $y = \neg f(x)$ with probability η
- Define $X_1 = \{x : \chi(x, 0) \neq \chi(x, 1)\}$, $X_2 = X - X_1$, and $p_1 = \Pr_P[x \in X_1]$
- $P_\chi = (p_1/(1 - 2\eta)) \Pr_{P,\eta}[\chi(x, y) = 1 | x \in X_1] + \Pr_{P,\eta}[(\chi(x, y) = 1) \wedge (x \in X_2)]$
- Can **efficiently** estimate P_χ from source of **noisy** examples
- Only known method for noise tolerance; other P_χ decompositions give tolerance to other forms of corruption

Limitations of SQ Algorithms

- How many queries χ are required to learn?
- **SQ dimension** $d_{F,P}$: largest number of pairwise (almost) uncorrelated functions in F with respect to P
- $\Omega\left(d_{F,P}^{1/3}\right)$ queries required for nontrivial generalization (Easy case: queries are functions in F)
- Also lower bounded by VC dimension of F
- Application: no SQ algorithm for small decision trees, uniform distribution (including C4.5/CART)

Boosting

- Replace source of random examples with an oracle accepting **distributions**
- On input P_i, oracle returns a function $h_i \in H$ such that $\Pr_P[h_i(x) \neq h(x)] \leq 1/2 - \gamma$, $\gamma \in (0, 1/2]$ (**weak learning**)
- Each successive P_i is a **filtering** of true input distribution P, so can simulate from random examples
- Oracle models a mediocre but nontrivial heuristic
- Goal: combine the h_i into h such that $\Pr_P[h(x) \neq f(x)] \leq \epsilon$

How is it Possible?

- Intuition: each successive P_i should force oracle to learn something "new"
- Example: $P_1 = P$; P_2 balances $h_1(x) = f(x)$ and $h_1 \neq f(x)$; P_3 restricted to $h_1(x) \neq h_2(x)$
- $h(x)$ is majority vote of h_1, h_2, h_3
- Claim: if each h_i satisfies $\Pr_{P_i}[h_i(x) \neq f(x)] \leq \beta$, then $\Pr_P[h(x) \neq f(x)] \leq 3\beta^2 - 2\beta^3$
- Represent error algebraically, solve constrained maximization problem
- Now **recurse**

Measuring Performance

- How many rounds (filtered distributions) required to go from $1/2 - \gamma$ to ϵ?
- Recursive scheme: polynomial in $\log(1/\epsilon)$ and $1/\gamma$, hypothesis is ternary majority tree of weak hypotheses
- Adaboost: $(1/\gamma^2)\log(1/\epsilon)$, hypothesis is linear threshold function of weak hypotheses
- Top-down decision tree algorithms: $(1/\epsilon)^{1/\gamma^2}$, hypothesis is a decision tree over weak hypotheses
- Advantage γ is actually problem- and algorithm-dependent

Computational Intractability Results for Learning

- **Representation-dependent**: hard to learn the functions in F **using** hypothesis representation H
- **Representation-independent**: hard to learn the functions in F, **period**

A Standard Reduction

- Given examples of $f \in F$ according to any P, L outputs $h \in H$ with $\epsilon(h) \leq \epsilon$ with probability $\geq 1 - \delta$ in time polynomial in $1/\epsilon, 1/\delta, n, s(f)$
- Given sample S of m pairs $\langle x_i, f(x_i) \rangle$, run algorithm L using $\epsilon = 1/(2m)$ and drawing randomly from S to provide examples
- Then if there exists $h \in H$ consistent with S, L will find it with probability $\geq 1 - \delta$
- So: reduce hard combinatorial problem to **consistency problem** for (F, H), then learning is hard unless RP $=$ NP
- Note: converse from Occam's Razor

Examples of Representation-Dependent Intractability

- Consistency for (k-term DNF,k-term DNF) as hard as graph k-coloring
- Consistency for (k-term DNF,$2k$-term DNF) as hard as approximate graph coloring
- Consistency for (k-term DNF,k-CNF) is **easy**
- Approximate consistency for (conjunctions with errors, conjunctions) as hard as set covering

Representation-Independent Intractability

- Complexity theory: set of examples $\approx$ problem instance, simple hypothesis $\approx$ solution for instance
- "Read off" a k-coloring of original graph G from the **sytactic form** of a k-term DNF formula for the associated sample S_G
- **No** restrictions on form of h: **everything** is learnable
- At least ask that hypotheses be **polynomially evaluatable**: compute $h(x)$ in time polynomial in $|x|, s(h)$
- Want learning to be hard for **any** efficient algorithm

Public-Key Cryptography and Learning

- A sends many encrypted messages $F(x_1), \ldots, F(x_m)$ to B
- Efficient eavesdropper E may obtain (or generate) many pairs $\langle x_i, F(x_i) \rangle$
- Want it to be **hard** for E to decrypt a "new" $F(x')$ (generalization)
- Thus, E wants to "learn" inverse of F (decryption) from examples!
- Inverse is "simple" given "trapdoor" (fairness of learning)

Applications

- Given by PKC: encryption schemes F with polynomial-time inverses, E's task as hard as factoring ($F(x) = x^e \bmod p \cdot q$)
- Thus, learning (small) boolean circuits is intractable
- Also hard: boolean formulae, finite automata, small-depth neural networks
- DNF and decision trees?

The Fourier Representation

- Any function $f : \{0, 1\}^n \rightarrow \Re$ is a 2^n-dimensional real **vector**
- **Inner product**: $\langle f, h \rangle = (1/2^n) \sum_x f(x)h(x)$
- $\langle f, h \rangle = E_U[f(x)h(x)] = \Pr_U[f(x) = h(x)] - \Pr_U[f(x) \neq h(x)]$
- Set of 2^n orthogonal **basis** functions: $g_z(x) = +1$ if x has even parity on bits i where $z_i = 1$, else $g_z(x) = -1$
- Write $f(x) = \sum_z \alpha_z(f) \cdot g_z(x)$, coefficients $\alpha_z(f) = \langle f, g_z \rangle$

Fun Fourier Facts

- Parseval's Identity: $\langle f, f \rangle = \sum_z (\alpha_z(f))^2 = 1$
- Small DNF f: $\sum_{z : w(z) \leq c_0 \log(n)} (\alpha_z(f))^2 \approx 1$
- Small decision tree f: spectrum is **sparse**, only a small number of non-zero coefficients
- Tree-based spectrum estimation using membership queries