Lecture 7

Analytical Learning Discussion (3 of 4): Learning and Knowledge

Wednesday, February 2, 2000

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

Readings:
Chown and Dietterich

Lecture Outline

- Paper
  - Paper: "A Divide-and-Conquer Approach to Learning from Prior Knowledge"
  - Authors: E. Chown and T. G. Dietterich
- Overview
  - Using prior knowledge as an aid to learning
    - Model calibration problem
    - Role of prior knowledge in analytical and inductive learning
  - Hierarchical learning system: *MAPSS*
    - Analytical learning to decompose prediction learning problem sequentially
    - Idea: choose hypothesis language (parameters), examples for subproblems
- Topics to Discuss
  - How to choose prediction target(s)?
  - Local versus global optimization: how can knowledge make difference?
  - How does hierarchical decomposition implement *bias shift* (search for *H*)?
  - Empirical improvements using prior knowledge? Ramifications for KDD?
- Next Paper: Towell, Shavlik, and Noordewier, 1990 (KBANN)

Background AI and Machine Learning Material

- Parameter Estimation
  - Russell and Norvig
    - Chapter 18: inductive learning (version spaces, decision trees)
    - Chapter 21: learning with prior knowledge
  - Mitchell
    - Chapter 2: inductive learning (basics, inductive bias, version spaces)
    - Chapter 6: Bayesian learning
- Topics to Discuss
  - Muddiest points
    - Inductive learning: learning as search (in *H*)
    - Data preprocessing in KDD
    - Model calibration: parameter estimation (inductive learning application)
    - Local versus global optimization
  - Example questions to ask when writing reviews and presentations
    - *How is knowledge represented?*
    - *Exactly how is prior knowledge applied to improve learning?*

MAPSS: Issues Brought Up by Students in Paper Reviews

- Key MAPSS-Specific Questions
  - How to choose prediction target(s)? (prefilter using "relevance knowledge")
  - Learning by local vs. global optimization
    - Global (e.g., simulated annealing): "no prior assumptions" about P(*h*)
    - Role of knowledge? (preference, representation bias)
  - *How does hierarchical decomposition implement bias shift (search for H)?*
    - Bias shift: change of representation (aspect of inductive bias)
    - References: [Fu and Buchanan, 1985; Jordan *et al*, 1991; Ronco *et al*, 1995]
  - Empirical improvements using prior knowledge? (better convergence in training)
  - Ramifications for KDD? (better parametric models for prediction; scalability)
- Key General Questions
  - *How is knowledge base (KB) represented?* (programmatic classification model)
  - *Exactly how is prior knowledge applied to improve learning?* (prefiltering *D*)
- Important Question: *What Kind of Analytical/Inductive Hybrid Is This?*
- Applications to KDD (Model Calibration in Simulators, etc.)

Key Strengths of MAPSS Learning Technique

- Strengths
  - Prior knowledge led to training convergence
    - Previously, could only calibrate 12 of 20 parameters of model (Section 2.2)
    - Prior knowledge made it possible to calibrate rest (Section 3.3)
  - Idea: *analysis of code to produce prior knowledge*
    - Knowledge-based software engineering (KBSE) concept
    - Implement classification model as program
    - Use partial evaluation of program to find $x \in D$ for which *few* θ_i are unknown
  - Idea: *bootstrapped (interleaved inductive, analytical) learning*
    - Training: "short runs" of global optimization, interleaved with prefiltering of *D*
    - Produces filter models and one example per model (batch of 40)
  - Idea: *decomposing problems into locally relevant sets of parameters*
    - Scalability (through divide-and-conquer): relative to θ_i (65 attributes)
    - Partitioning problem by partitioning attributes [Hsu, Ray, and Wilkins, 2000]
- Applications to KDD
  - Can express many KBs as programs: simulators, classification systems
  - Methods for estimating (e.g., EM) missing values in data
  - Breaking problem into more tractable pieces (more in Paper 8!)

Key Weaknesses of MAPSS Learning Technique

- Weaknesses
  - Still took 3+ months (even using prior knowledge)!
    - 750K evaluations took 6 CPU weeks (SPARC 2)
    - 1.5M evaluations in final version
  - Generality not well established
    - *Under what conditions can we express prediction rules in the imperative programming language used?*
    - Ramifications for general-case learning applications (e.g., KDD?)
  - Typos in section 3.2?
- Unclear Points
  - *What form of partial evaluation is appropriate for prediction task?*
  - *How to choose the right architecture of committee machine? (e.g., filter models)*
  - *Can technique scale up calibration of broad class of scientific models?*
  - How to use prior relevance knowledge in KDD?
    - Acquisition (automatic relevance determination, *aka* ARD) – "20 important θ_i"
    - Automatic application (stay tuned…)
  - How to apply other forms of prior knowledge (constraints, etc.)? – Paper 4

Data Gathering Algorithm

- Committee Machine
  - See
    - Chapter 7, Haykin
    - Chapter 7, Mitchell
    - Lectures 21-22, CIS798 (http://ringil.cis.ksu.edu/Courses/Fall-1999/CIS798)
  - Idea
    - Use experts to preprocess (filter) *D* or combine predictions
    - In this case, 40 experts prefilter *D* to get *n* = 40 examples; need 32-36 to agree
- Intuitive Idea
  - Want to use prior knowledge (in form of imperative program) to speed up learning
    - Analyze program: perform partial evaluation using current calibration
    - Prefilter data: find "good operating regions" (classification paths with "few enough" unknown parameters)
  - Algorithm: technical details
    - Need to reduce sensitivity (instability): 1 example per model (of 40)
    - Accumulate 40 "good" training examples

Scaling Up KDD Using Prior Knowledge

- MAPSS Problem
  - *n* = |*D*| = 40: considered "small" for this problem
    - Not clear how many candidates, but only 5 filter passes suffices
    - Nota Bene: Takes many experts (32-36 out of 40) to get good "consensus"!
  - *m* = 65 attributes: considered "medium" for this problem (given *n*)
  - 5 prediction targets
    - 3 leaf area index (LAI) predictions, 1 runoff prediction (numerical)
    - 1 biome classification (74 possible values)
- Prior Knowledge: Lessons Learned
  - Previous approaches
    - KBANN: backpropagation in feedforward ANNs using "compiled" constraints
    - FOCL: variant of FOIL (decision trees using first-order logic predicates)
    - Others: qualitative simulation, inductive logic programming (ILP), etc.
  - Problem: lack of scalability
    - Computational limitations of inference (semidecidability of resolution)
    - Intractability of even very restricted learning approaches

Course Project: Overview

- 3 Components
  - Project proposal (20%, 50 points)
  - Implementation (50%, 125 points)
  - Final report (30%, 75 points)
- Project Proposal (Due 02/14/2000)
  - 1-3 page description of project topic, plan
  - Guidelines: next (and suggested topics, tools on course web page)
- Implementation
  - Students choice of programming language
  - Guidelines: Friday (and on course web page)
- Final Report
  - 4-6 page report on implementation, experimental results, interpretation
  - Peer-reviewed (does not determine grade)
  - Reviews graded (short report worth 60 points, reviews worth 15 points)

Course Project: Proposal Guidelines

- Report Contents (1-3 Pages)
  - Scope: *What kind of data will you use?*
  - Problem: *What problem are you addressing?*
  - Methodology: *How are you addressing the problem?*
- Scope
  - *What data sets will you use?*
  - *What characteristics of the data are you trying to deal with / exploit?*
- Problem
  - Objective: *What KDD problem are you trying to solve?*
  - Performance element: *What is the problem-solving component of your KDD system?*
  - Evaluation: *How will you measure success?*
- Methodology
  - Implementation: *What will you implement?* (general statement, not specification)
  - Tools: *What programming languages and KDD tools will you use?*

Terminology

- Inductive Learning
  - Prior knowledge
    - Declarative: expressed in assertions (e.g., FOPC)
    - Procedural: expressed in imperative statements
    - Functional: expressed as functions (e.g., higher-order) and relations
    - Taxonomic: expressed as classification hierarchy
  - Inductive bias
    - Representation bias: expressed by *H*, hypothesis space (language)
    - Preference bias: expressed by *L*, learning algorithm
    - Change of representation: transformation from *H* into *H'* (form of bias shift)
    - Bias shift: change in inductive bias (representation or preferences)
- Divide-and-Conquer Approaches to Learning
  - Hierarchical learning systems: decompose problem according to attributes, examples, etc.
  - Committee machines: combine outputs of multiple expert "modules"

Summary Points

- Key Points Covered
  - Using prior (declarative) knowledge as an aid to learning
  - Hierarchical learning system: *MAPSS*
    - Bias shift through systematic problem decomposition
    - Idea: choose hypothesis language (parameters), examples for subproblems
- Discussion Topics
  - Local versus global optimization: knowledge as bias (control of search over *H*)
  - Scalable KDD: hierarchical decomposition using relevance knowledge
    - Prior knowledge in form of classification program
    - Developing relevance knowledge using partial evaluation
  - Choosing prediction targets in KDD: general filtering problem
- Next Paper
  - Towell, Shavlik, and Noordewier, 1990
  - "Knowledge-Based Artificial Neural Networks (KBANN)": constraints in feedforward ANN learning