Lecture 2

Analytical Learning Presentation (1 of 4): Explanation-Based and Inductive Learning in ANNs

Friday, January 21, 2000

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

Readings:
"Integrating Inductive Neural Network Learning and Explanation-Based Learning", Thrun and Mitchell

Presentation Outline

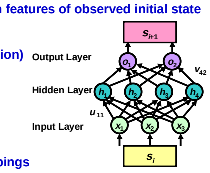
- **Paper**
  - "Integrating Inductive Neural Network Learning and Explanation-Based Learning"
  - Authors: S. B. Thrun and T. M. Mitchell
  - Thirteenth *International Joint Conference on Artificial Intelligence* (IJCAI-93)
- **Overview**
  - Combining analytical learning (specifically, EBL) and inductive learning
    - Spectrum of domain theories (DTs)
    - Goals: *robustness, generality, tolerance for noisy data*
  - Explanation-Based Neural Network (EBNN) learning
    - Knowledge representation: artificial neural networks (ANNs) as DTs
    - Idea: track changes in goal state with respect to query state (*bias derivation*)
- **Application to Decision Support Systems (DSS): Issues**
  - Neural networks: good substrate for integration of analytical, inductive learning?
  - How are goals of robustness and generality achieved? Noisy data tolerance?
  - Key strengths: approximation for EBL; using domain theory for bias shift
  - Key weakness: how to *express* prior DT, *interpret* explanations?

Inductive Learning versus Analytical Learning

- **Inductive Learning**
  - Given
    - Instances X
    - Target function (concept) c: $X \rightarrow H$
    - Hypotheses (i.e., hypothesis language *aka* hypothesis space) H
    - Training examples D: positive, negative examples of target function c
  - Determine
    - Hypothesis $h \in H$ such that $h(x) = c(x)$ for all $x \in D$
    - Such h are consistent with training data
- **Analytical Learning**
  - Given
    - X, c: $X \rightarrow H$, H, $D = \{<x_i, c(x_i)>\}$
    - Domain theory T for explaining examples
  - Determine
    - $h \in H$ such that $h(x) = c(x)$ for all $x \in D$ as can be proven deductively using T
    - h is consistent with D and T

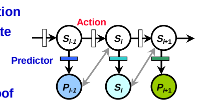
Analytical Learning and Inductive Learning: Integration for Problem Solving using ANNs

- **Analytical Learning in Problem-Solving Frameworks**
  - Target function: ANN or lazy representation (e.g., *k*-nearest neighbor)
  - Instances: labeled episodes ("will doing a_1 in s_1 lead to $s_n \in$ Goals?")
  - Domain theory: expressed as sequence of ANNs
  - Explanations
    - *Post-facto* prediction of observed episode using domain knowledge
    - Shows how achieving final goal depended on features of observed initial state
- **Inductive Learning using ANNs**
  - Purpose: to acquire DT (*components* of explanation)
  - Each multi-layer perceptron (MLP) trained
    - Input: features (attributes) of state s_i
    - Target: features of state $s_{i+1} = Do\,(a_{i-1}, s_{i-1})$
    - *How do weights capture domain theory?*
  - Instances: state/action-to-state (predictive) mappings
  - Objective: produce sequence of mappings from s_1 to goal feature of s_n



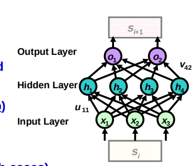
Analytical and Inductive Learning: Principles of Integration in Decision Support

- **Generalizing over ANN Representations**
  - Prediction
    - Target at each step: to identify features of next state
    - Predictor: any inductive learner (H, L) capable of expressing this mapping
    - Sequence (chain) of predictions forms explanation
  - Pattern matching: *unify* predicted features with state
- **Ways to Integrate (Section 3)**
  - Analytical, then inductive (e.g., EBNN)
    - EBG: prove that "x is a $c(x)$" and generalize proof
    - Apply inductive generalization (e.g., version spaces) to proofs, examples, attributes
  - Inductive, then analytical
    - Find empirical (statistical) regularities (predicates) by simple induction
    - Explain, generalize associations (NB: *idea recurs in uncertain reasoning*)
  - Interleaved inductive and analytical processes
    - Explain D, not x_i; inductively complete explanations, abstract over DT
    - EBL with systematic or opportunistic induction steps to improve DT



Domain Theories

- **Symbolic EBL Domain Theory**
  - Stereotypically, knowledge base of arbitrary Horn clauses
    - Atomic inference step: resolution (sequent rule: $P \vee L, L \rightarrow R \vdash P \vee R$)
    - Inductive learning: rule acquisition (FOIL, inductive logic programming, etc.)
  - Due to inferential completeness (and decidability limitations), usually restricted
  - Atomic (deductive) learning step: variabilization, pruning proof tree (see Section 21.2, Russell and Norvig)
- **EBNN Domain Theory**
  - All "inference rules" expressed as ANNs
    - Atomic inference step: feeding example forward
    - Corresponds to many floating-point operations
    - Inductive learning: ANN training (e.g., backprop)
  - Atomic (deductive) learning step: next
- **Application**
  - Direct application: classification, prediction (in both cases)
  - Indirect application: control, planning, design, other optimization



Explanation-Based Neural Network (EBNN) Methodology

- Goals (Section 1)
  - Robustness – ability to use different "strength" DTs, performing:
    - At least as well as inductive system even in "worst" case of no DT
    - Comparably to EBG with perfect DT
    - With graceful degradation in between
  - Generality – ability to incorporate DTs of different levels of completeness
  - Noise tolerance – able to learn from *D* with error in instances (x_i), labels ($c(x_i)$)
- Intuitive Idea (Section 2.1-2.2)
  - Given: sequence of descriptors (of state of problem-solving world and actions)
  - Train ANNs to predict next state
  - Use them to form explanation chain; analyze chain to train "top level" model
- Relation to KDD
  - Key contribution: method for hybrid learning of predictor functions for DSS
  - Possible direct application: synopsis
    - Wrappers for KDD performance optimization (data cleansing, queries)
    - Stay (around Lecture 30 or sooner if you want project topics…)

EBNN Learning Algorithm

- Given: Sequence of State-Action Pairs
- Methodology (Section 2.1-2.2)
  - Train ANNs to predict next state
    - Represent state descriptors as feature (attribute) vectors in training examples
    - Store trained ANN(s) as units of DT
  - Step 1 (Explain): use DT to form explanation chain
    - Given: "known" sequence of action, state pairs $<a_1, s_1>, <a_2, s_2>, \ldots, <a_n, s_n>$
    - Use DT to predict next state (*post-facto*) in each case (single or multiple-step lookahead)
  - Step 2 (Analyze): compute slope of target *wrt* initial state, action
    - Take derivatives of ANN weights in chain (recursively, using chain rule)
    - Rationale (Figure 3): $f'(x)$ helps in interpolating $f(x)$
  - Step 3 (Refine): use derivative to fit top-level curve
    - Partial derivative: $\partial\, s_n \text{ is-goal} \,/\, \partial a_1 \partial s_1$
    - Top-level curve is ANN or other model that maps $<a_1, s_1>$ to {+, –}

Robustness of EBNN

- Problem (Section 2.3)
  - *What if some ANNs (for DT, not for overall target) are wrong?*
  - Domain theory could be arbitrarily bad (inaccurate) over desired inference space (problem-solving world)!
  - Want to give proportionately less weight to poor slopes
  - But how to guess generalization quality over slopes?
- Solution Approach (Section 2.3)
  - Idea: assign credit (loss) in proportion to accuracy (error) on predictions
  - Assumption (LOB*): *prediction errors measure slope errors*
    - Ramifications: can propagate credit back through explanation chain (*n*-step estimate), weight analytical, inductive components accordingly
    - Open question: *For what inducers (inductive learning models, algorithms) does LOB* hold?*
- Experimental Goals (Section 2.4)
  - Determine role of knowledge quantitatively (measure improvement due to DT)
  - Test quality of lazy (nearest-neighbor) generalization

Experimental Method

- Experimental Results (Section 2.4)
  - Improvement using DT (Figure 5): pre-trained ANNs improve average and worst-case performance of learned control function
  - Improvement in proportion to DT strength (Figure 6): graphs showing gradual improvement as DT-learning ANNs get more training examples
  - Possible experimental issues
    - Highly local instance-based generalization: *k*-NN with $k = 3$
    - Small sample: average of 3 sets (but large *D* in each case)
    - Depends on how *D* was "randomly generated"
  - Visualization issue: would have helped to have graph of Figure 6 with one axis labeled "examples"
- Claims (Section 1, 4)
  - *EBNN is robust*: *n*-step accuracy estimate weights ANN predictions according to cumulative credit (product of prediction accuracy "down the chain"), improving tolerance for poor DTs
  - *EBNN is general*: can incrementally train ANNs to get partial DT

Using Integrated (Multi-Strategy) Learning for Decision Support

- Multi-Strategy Learning
  - Also known as integrated, hybrid learning
  - Methods for combining multiple algorithms, hypotheses, knowledge/data sources
- Role of Analytical-Inductive Multi-Strategy Learning in Problem Solving
  - "Differential" method: compatible with dynamic programming (DP) methods?
    - *Q*-learning [Watkins, 1989]
    - TD(λ) [Sutton, 1988]
  - Other numerical learning ("parametric", "model-theoretic") learning models
    - Hidden Markov Models (HMMs), Dynamic Bayesian Networks (DBNs)
    - See Lectures 17-19, CIS 798 (Fall 1999), especially 19
    - ADN approach more suited to analytical learning?
  - Methods for incorporating knowledge: stay tuned (next presentation)
- Applicability to Decision Support Systems (DSS) and KDD
  - Important way to apply *predictions* (e.g., output of business simulation) to DSS
  - *Q: Can we use this for KDD directly?*
    A: Perhaps, if sequence of states of data model can be explained

Summary Points

- Content Critique
  - Key contribution: simple, direct integration of inductive ANN learning with EBL
    - Significance to KDD: good way to apply predictive models in decision support
    - Applications: policy (control) optimization; DTs, explanations for wrappers?
  - Strengths
    - Generalizable approach (significant for RL, other learning-to-predict inducers)
    - Significant experiments: measure generalization quality, graceful degradation
  - Weaknesses, tradeoffs, and questionable issues
    - EBNN DT lacks some advantages (semantic clarity, etc.) of symbolic EBL DT
    - Other numerical learning models (HMMs, DBNs) may be more suited to EBG
- Presentation Critique
  - Audience: AI (learning, planning), ANN, applied logic researchers
  - Positive and exemplary points
    - Clear introduction of DT "spectrum" and treatment of integrative approaches
    - Good, abstract examples illustrating role of inductive ANN learning in EBNN
  - Negative points and possible improvements
    - Insufficient description of analytical ANN *hypothesis representations*
    - Semantics: still not clear how to *interpret* ANN as DT, explanations