Lecture 4

Analytical Learning Presentation (2 of 4): Iterated Phantom Induction

Wednesday, January 26, 2000
(Given Friday, January 28, 2000)

Steve Gustafson
Department of Computing and Information Sciences, KSU
http://www.cis.ksu.edu/~steveg

Readings:
"Iterated Phantom Induction: A Little Knowledge Can Go a Long Way", Brodie and DeJong

Presentation Overview

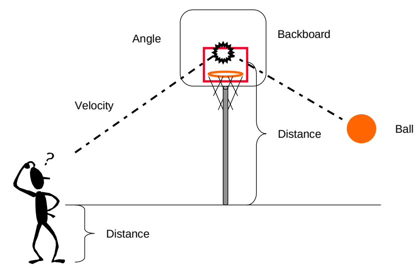
- Paper
  - "Iterated Phantom Induction: A Little Knowledge Can Go a Long Way"
  - Authors: Mark Brodie and Gerald DeJong, Beckman Institute, University of Ilinois at Urbana-Champaign
- Overview
  - Learning in failure domains by using phantom induction
    - Goals: *don't need to rely on positive examples or as many examples as needed by conventional learning methods.*
  - Phantom Induction
    - Knowledge representation: Collection of points manipulated by Convolution, Linear regression, Fourier methods or Neural networks
    - Idea: Perturb failures to be successes, train decision function with those "phantom" successes
- Issues
  - Can phantom points be used to learn effectively?
  - Key strengths: Robust learning method, convergence seems inevitable
  - Key weakness: Domain knowledge for other applications?

Outline

- Learning in Failure Domains
  - An example - basketball "bank-shot"
  - Conventional methods versus Phantom Induction
  - Process figure from paper
- The Domain
  - Air-hockey environment
- Domain Knowledge
  - Incorporating prior knowledge to explain world-events
  - Using prior knowledge to direct learning
- The Algorithm
  - The Iterated Phantom Induction algorithm
  - Fitness measure, inductive algorithm, and methods
- Interpretation
  - Results
  - Interpretation graphic - explaining a phenomenon
- Summary

Learning in Failure Domains

- Example - Learning to make a "bank-shot" in basketball - We must fail to succeed



Learning in Failure Domains

- Conventional learning methods
  - Using conventional learning methods in failure domains can require many, many examples before a good approximation to the target function is learned
  - Failure domains may require prior domain knowledge, something which may be hard to encode in conventional methods, like neural networks and genetic algorithms
- Phantom Decision method
  - Propose a problem, generate a solution, observe the solution, explain the solution and develop a "fix". (assumes the solution resulted in a failure)
  - The "fix" added to the previous solution creates a "phantom" solution, which should lead the original problem to the goal
  - Domain knowledge is used to explain the solution's results, and only perfect domain knowledge will lead to a perfect phantom solution.
  - After collecting phantom points, an INDUCTIVE algorithm is used to develop a new decision strategy
  - Another problem is proposed and a new solution is generated, observed, phantom decision found and decision strategy is again updated.

Learning in Failure Domains

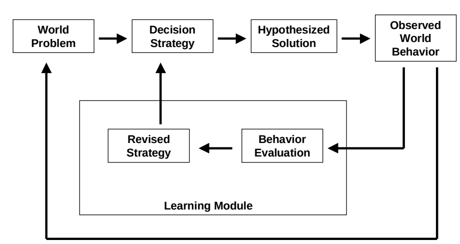


Figure 1: Interactive Learning Recreated from "Iterated Phantom Induction: A Little Knowledge Can Go a Long Way", Brodi and DeJong, AAAI, 1998

The Domain

- Air hockey table
  - Everything is fixed except angle at which puck is released
  - Paddle moved to direct puck to the goal
  - Highly non-linear relationship between puck's release angle and paddle's offset (does this have to do with the effort to simulate real world?)

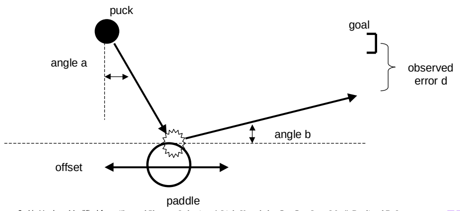


Figure 2: Air-Hockey Modified from "Iterated Phantom Induction: A Little Knowledge Can Go a Long Way", Brodi and DeJong, AAAI, 1998

Domain Knowledge

- Domain Knowledge
  - *f** is the ideal function which produces the paddle offset to put the puck in the goal, determined from the puck's angle a
  - The learning problem is to approximate *f**
  - *e** is the ideal function which produces the correct offset from the error, *d*, from *f(a)*
  - *e*(d,a) + f(a)* should place the puck in the goal
  - Both *f** and *e** are highly non-linear and require a perfect domain knowledge
  - So, the system needs to approximate *e* so that it can adequately approximate f**
  - What domain knowledge is needed to approximate *e** ?
    - As angle b increases, error d increases
    - As offset increases, b increases
  - System Inference: positive error = decrease offset proportional to size of error

The Algorithm

1. $f_0 = 0$
2. $j = 0$
3. for $i = 1$ to n
   - i: *generate* a_i [puck angle]
   - ii: $o_i = f_j(a_i)$ [apply current strategy to get offset]
   - iii: find d [observe error d from puck and goal]
   - iv: find $e(d_i)$ [decision error, using error function e]
   - v: find $o_i + e(d_i)$ [phantom offset that should puck with a_i in the goal]
   - vi: add (a_i, $o_i + e(d_i)$) to training points [phantom point]
4. $j = j + 1$
5. Find a new f_j from training points [use inductive algorithm]
6. Apply fitness function to f_j
7. If "fit" function, exit, otherwise go to step 3

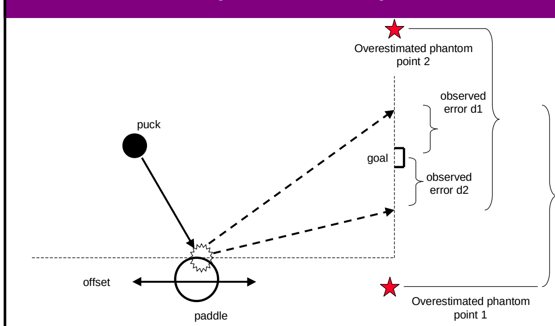
The Algorithm

- Performance Measure
  - 100 randomly generated points, no learning or phantoms produced, mean-squared error
- Inductive algorithm
  - instance-based, convolution of phantom points
  - Place a Gaussian point at center of puck angle
  - Paddle offset is weighted average of phantom points where the weights are come from the values of the Gaussian.
- Other Algorithms
  - Linear Regression, Fourier Methods, and Neural Networks
  - All yielded similar results
  - Initial divergence, but eventual convergence

The Experiments

- Experiment 1 - Best Linear Error Function
  - Similar to performance *e** - errors remains due to complexity target function
- Experiment 2 - Underestimating Error Function
  - slower convergence rate
- Experiment 3 - Overestimating Error Function
  - fails to oscillate as expected, converges after initial divergence
- Experiment 4 - Random Error Function
  - How can it fail?? Error function sometime underestimates error, sometimes overestimates error
- Interpretation
  - The successive strategies are computed by convoluting the previous phantom points, therefore, the following strategy passes through their average.
  - Hence, even large errors result in convergence

Interpretation Example



Summary Points

- **Content Critique**
  - Key contribution:
    - "iterated phantom induction converges quickly to a good decision strategy."
    - Straight-forward learning method which models real world.
  - Strengths
    - Robust - when doesn't this thing diverge!
    - Interesting possibilities for applications (failure domains)
  - Weaknesses
    - Domain knowledge is crucial. Unclear on how to determine sufficient domain knowledge given a problem
    - No comparison to other learning methods
- **Presentation Critique**
  - Audience: Artificial intelligence enthusiasts - robot, game, medical applications
  - Positive points
    - Good introduction, level of abstraction, and explanations
    - Understandable examples and results
  - Negative points
    - Some places could use more detail - inductive algorithm, fitness measure