Opponent Modeling In Interesting Adversarial Environments

Brett Borghetti

22 Jul 2008
Opponent Modeling in Interesting Adversarial Environments

• Opponent: An entity known to be attempting to maximize his own well being in an environment where that usually implies he is minimizing ours.

• Modeling: Building a capability to predict something about a target based on observations of the target in its environment.

• Interesting Environments: Where some agents are unlikely to have access to – or desire to use - perfect equilibrium solutions.
Motivation

• Systems with opponent modeling capability can learn to beat approximations of equilibrium strategies.
• Leaving the equilibrium opens an agent to exploitation.
• So we must decide whether to model our opponent
  – How good can we do in a given environment?
  – How good are we doing with a given model?
  – What can we do to improve if we don’t know our opponent yet?
Overview

• Contributions and Domains
• Related Work
• Opponent Models and Predictions
• Measuring Model Quality
• The Environment-Value of a Model
• Improving Opponent Models
• Conclusions & Future Work
Contributions

• Developed a fast-learning model for poker and showcased a method for quantifying the contribution made by the learning component.
• Created a new domain-independent measure of performance prediction quality for opponent models.
• Defined the environment-value of an opponent model and showed how to calculate it.
• Presented new techniques for improving models when the opponents are unknown and observations of their behavior are unavailable.
### Domains of Interest

<table>
<thead>
<tr>
<th>Information</th>
<th>Action</th>
<th>Length</th>
<th>Use</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simultaneous</td>
<td>Hidden</td>
<td>Simult.</td>
<td>Perf. Bounds</td>
</tr>
<tr>
<td>High Card Draw</td>
<td></td>
<td>Single</td>
<td></td>
</tr>
<tr>
<td>Full Scale Poker</td>
<td>Hidden</td>
<td>Seq.</td>
<td>Measurement; Improving</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Ext.</td>
<td>Models</td>
</tr>
<tr>
<td>Simultaneous</td>
<td>Perfect</td>
<td>Simult.</td>
<td>Perf. Bounds</td>
</tr>
<tr>
<td>Move Strategy</td>
<td></td>
<td>Ext.</td>
<td></td>
</tr>
<tr>
<td>Game</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Related Work

• Agent Interaction
  – Game Theory
  – Opponent Modeling

• Opponent Model Quality
  – Desired Properties
  – Measuring Quality
## Related Work: Game Theory

<table>
<thead>
<tr>
<th>Category</th>
<th>Reference</th>
</tr>
</thead>
</table>
| Rationality and Equilibrium      | Nash, Aumann, Brandenberger: Nash, Eq. & Correlated Eq.; unbounded rationality  
|                                  | Simon: Bounded Rationality                                               |
| Representing Environments as Games | Nash: two person games, non-cooperative games                            
|                                  | Koller & Megiddio: Mapping environments to extensive form games         |
| Solving Games                    | Koller & Megiddio: Direct solution for extensive-form games              
|                                  | Allis: Levels of solution; solving Connect 4                             
|                                  | Schaeffer: solving checkers                                             |
| Intractable Solutions            | Billings; Zinkevitch: State Abstraction                                  
|                                  | Gilpin & Sandholm: Automated Abstraction                                 |
## Related Work: Opponent Modeling

| Representations: | Carmel & Markovitch: DFA  
| Gal & Pfeffer: MAID, NID  
| Steffens: Case Based Reasoning  
| Riley & Veloso: Archetypes |
| Learning: | Bowling & Veloso: WoLF (GD)  
| Jensen: ELPH (pred. entropy)  
| Hoehn: Exp3 |
| Using: | Carmel & Markovitch: $M^*$  
| Luckhardt & Irani: $max^n$  
| Sturtevant & Bowling: Soft-$max^n$ |
# Related Work: Model Quality

| Desired properties: | Bowling & Veloso: *Rationality, Convergence*  
|                     | Jensen: *Fast reaction (tracking)*  
| Performance Measurements | Carmel & Markovitch: *Error*  
|                        | Rogowski: *Accuracy*  
|                        | Blaylock & Allen; Cox & Kerkez: *Precision and Recall*  
|                        | Davidson; Sukthankar & Sycara: *Confusion Matrix*  

Related Work: What’s Missing?

1. Model prediction quality is usually measured in terms of number of correct predictions made.
2. Nothing regarding performance bounds on opponent models as a function of the game.
3. Lacking in dedicated improvement techniques for opponent modeling.
Definition of Opponent Model

• Two types of models:
  – Predict future action based on current state
  – Explain past state based on current action

• Can be expressed in terms of probability distribution over possible actions or states
Prediction

- Opponent is going to make a decision
- Model’s job is to predict the conditional probability that the opponent will take each action
Calculating Expected Value (EV) of a Branch

- If we are deciding what to do next we would like to know the EV of each action
- $EV = \sum (P_i V_i)$
- $EV$ depends on our opponent
Measuring Model Quality

• Total Performance: How well did the agent perform overall?
• Differential Performance: How much better is the agent when it can learn?
• Prediction Performance: How well did the model predict the opponent?
Measuring Model Quality: Total Performance Measurement

• PokeMinnLimit1 vs. Top 3 agents (2007)
  – Equilibrium/Fixed Strategy opponents
• Smoothed payouts over 10 matches/3000 hands each
• Evidence of learning against very strong fixed strategies
Measuring Model Quality: Differential Performance

Differential Performance (PokeMinnLimit1-PokeMinnLimit1NoLearn) versus Hyperborean07LimitEq1

- Normalized Cumulative Performance
- Double Exponential Smoothed Performance, Alpha = 0.005
Measuring Model Quality: Prediction Performance

• Expected value calculations require correct distributions over actions

• If our predictions are distributions over opponent behaviors, current measures are not well suited.
  – Precision/recall don’t make sense when the number of underlying classes is large

• We need a scalar measure of the distance between two distributions
Measuring Model Quality: Prediction Performance

• Desired traits of a distribution-distance measurement
  – Result = 0 if distributions are the same
  – Result >= 0 for all pairs of distributions
  – Result is symmetric
  – Result obeys triangle inequality
  – Result obeys the transitive property
  – Result is bounded
Measuring Model Quality: Prediction Performance

Given distributions $p$ (prediction) and $q$ (truth) over a set of choices $\chi$, the *Jensen-Shannon Divergence* is:

$$JSD(p, q) = D \left( p \| \frac{p + q}{2} \right) + D \left( q \| \frac{p + q}{2} \right)$$

Where $D$ is Relative Entropy / KL-Divergence:

$$D(a \| b) = \sum_{x \in \chi} a(x) \log \frac{a(x)}{b(x)}$$
Measuring Model Quality: Prediction Performance

\[
WPD(N) = \frac{\sum_{k \in N} W(k) \sqrt{JSD(P_k, Q_k)}}{\sum_{k \in N} W(k)}
\]

- Weighted Prediction Divergence: For all nodes \( k \) in the set \( N \), given the calculations for each node’s JSD and a weighting scheme \( W \), calculate the weighted prediction divergence for the set of nodes
Measuring Model Quality: Prediction Performance

• Weighting Schemes
  – **Uniform**: When no particular biases are required, set all weights to 1.
  – **Frequency weighting**: When some nodes occur more frequently, weight them according to frequency of occurring
  – **Utility Weighting**: based on the relative value of outcomes for trajectories passing through this node.
  – **Risk (reward) weighting**: Multiplying a node’s frequency by its utility yields the node’s risk.
Measuring Model Quality: Prediction Performance

• Empirical Results from Poker
  – Collected data from PokeMinnLimit1 vs HyperboreanLimitEq1 from 2007 Computer Poker Competition
  – Compared two opponent models predictions to truth data
  – Training Set Size: 20 observations/hand
  – Test Set Size: 660 observations/hand
Measuring Model Quality: Prediction Performance

- While Learning performed well during the pre-flop and flop portions of the game, it grew worse over time during the turn and river.
Measuring Model Quality: Discussion

• While agent-level performance of the learning model was better than the fixed model, WPD analysis reveals quality issues with the performance of the learning model.

• WPD paves the way for reasoning over which types of models might do well during a specific interaction.
Bounding Performance

• Goal: Determine the relationship between the structure of an environment and the potential value of an opponent model within that environment.
  – Helps us make decisions on whether, and what to model

• Method
  – Determine equilibrium behavior (and value) in original game
  – Transform the environment to provide perfect predictions
  – Recalculate the behavior and value in the new game

• Domains
  – Simultaneous-Bet High Card Draw
  – Simultaneous-move Strategy Game
Bounding Performance: Simultaneous-Bet High Card Draw

- High Card Draw With Simultaneous Betting
  - Ante = $1
  - Bet = $1
- Strategy = Threshold Card
  - Opponent = X axis
  - Me = Y axis
- Expected Earnings from our perspective (Z axis)
- Pure Strategy Equilibrium (PSE) at EIGHT,EIGHT
Bounding Performance:
Simultaneous-Bet High Card Draw

- When we know the opponent’s State (Card) what strategy should we choose?
  - If we have higher card, bet, otherwise follow our strategy
- PSE = ACE,EIGHT
- Guaranteed at least $0.24 per game
Bounding Performance:
Simultaneous-Bet High Card Draw

- When we know the opponent’s Policy, what strategy should we use?
- If opponent policy EIGHT, our payoff = $0.0
- But could be higher if opponent is not using that policy
Bounding Performance:
Simultaneous-Bet High Card Draw

- If we know the opponent’s next action what strategy should we choose?
  - Equivalent to turn-based betting
- No PSE
- Guarantee of at least $0.10 if I play fixed (SEVEN)
  - Mixed Strategy (SIX and SEVEN, 50% each) guarantees $0.1124
- New expected value \(-0.78 < EV < +0.85\)
Bounding Performance: Simultaneous-Bet High Card Draw

• Normal expectation is ±$0.78
• If we know perfectly the opponent’s:
  – State: Guaranteed at least $0.24
  – Policy: Guaranteed at least $0.00
  – Action: Guaranteed at least $0.10 ($0.11 mixed)
• Helps us make modeling decisions
Bounding Performance: Simultaneous-Bet High Card Draw

- Examine class of games formed when
  - Ante=1-Bet
  - Bet ranges from [0,2]
- Where do the guarantees occur?

![Perfect Model Guarantee Comparisons](image-url)
Bounding Performance:
Simultaneous-Move Strategy Game

• 2-Player Perfect Information Environment
  – 4 locations
  – 3 unit cap per player, 4 base limit per player

• Multiple phases per turn
  – Order/Move/Battle/Generate/Destroy/Build

• Solvable with iterated equilibrium calculation (2136 state isomorphisms with up to 2304 joint action sets per state)
Bounding Performance: Simultaneous-Move Strategy Game

- Transform environment to new environment where the action of the opponent is always revealed
  - Equivalent to a turn-based game
- Calculate equilibrium in the new game
- Subtract values to obtain performance improvement attributable to the model
Bounding Performance: Simultaneous-Move Strategy Game

- When oracle is used on every turn our improvement is significant
  - Fair Starting State mean improvement = 0.0808
  - All State mean improvement = 0.0692
Bounding Performance: Simultaneous-Move Strategy Game

- When oracle is used only once our improvement is small
  - Fair Starting State mean improvement = 0.0041
  - All State mean improvement = 0.0129
Improving Opponent Models:

• Goal: Formalize methods for assisting developers wishing to improve their modeling algorithms when actual opponents are unknown or unavailable for testing.

• Method: Use co-evolution and near-self-play to generate strong agents
Improving Opponent Models:
Tools

• Instrumentation
  – Identify and track measurable performance points where quality can be assessed (i.e. WPD)

• Advanced Parametric Control
  – Identify $p$ key features of the code (functions or variable settings) and rewrite the software to allow the features to be chosen at run time

$$\prod_{n=1..p} \phi_n$$
Improving Opponent Models: Techniques

• How do we reduce the size of the search?
  – Preprocess the $k$-parameter space to determine parameters with the most effect on outcome

\[
\sum_{i=1..k} \binom{n_i}{2}
\]

• Rank order performance
  – Test all pairs of parameter settings after preprocessing

• Techniques are amenable to automation
Future Work

• Equilibrium deviance detection (JSD)
  – If opponent is *not* playing correct distribution

• Automatic Performance Improvement
  – Genetic Algorithm Parameter Search (offline)
  – Metareasoning (online)
Future Work (cont.)

• Bounds calculation for model value
  – Private oracles
  – Distribution of opponent types
  – Non-Zero-Sum Environments
• Non-stationary opponents (anticipation)
Questions?
Precision and Recall

• **Precision** = correctly predicted / total predictions
  – What percentage of my predictions were correct?

• **Recall** = correctly predicted / total in that prediction class
  – What percentage of the total class was I able to correctly classify?

• **F-measure** = \(2 \times P \times R / (P + R)\)
Simultaneous Turn Strategy Game
Iterated Equilibrium Calculation

• Calculate the set of transition probabilities from every state to every other state, conditional on the orders given by each player

• Initialize values of states for terminal nodes to +1, 0 or -1 for win/tie/loss. Allow all other state values to float (initialize to zero)

• For each subgame of length $L=1..N$, compute:
  – Mixed NE distribution over orders from each player in each state
  – Value of playing the NE for each player, using game $(L-1)$’s values for the values of the states

• Repeat until convergence
Metareasoning Goal

• For each prediction required within an expected value calculation, estimate:
  – What is the prediction quality and cost of each model at this node?

• Choose the model for each prediction:
  – Lowest cost that meets the minimum quality
  – Highest quality within a fixed cost budget
Proposed System: Overview

Ground Level

Object Level

Meta-Level

Doing

Reasoning

Metareasoning

Action Selection

Perception

Control

Monitoring
Proposed System: Object Level Component

Object Level
Proposed System: Meta-Level Component

- Control
- Utility Calculation
- Activity Repository
- Predictions
- Prediction Cost
- Observations
- Context
- Monitoring
- World State
- Meta-Level

22 July 2008 Brett Borghetti