Explorations in Computer Go, Web Search, and Online Advertising

Thore Graepel
Microsoft Research, Cambridge

Thanks to Joaquin Quiñonero Candela, David Stern, Ralf Herbrich, Mykel Kochenderfer, Leon Bottou and other colleagues for their input
Overview

• Introduction

• Five Vignettes
  – $Q$-Learning and $\epsilon$-exploration for Fighting
  – User guided exploration for Racing
  – Monte-Carlo Tree Search for Computer Go
  – Bandits for Web Search
  – Exploration in Online Advertising

• Conclusion
Overview

• Introduction

• Five Vignettes
  – $Q$-Learning and $\epsilon$-exploration for Fighting
  – User guided exploration for Racing
  – Monte-Carlo Tree Search for Computer Go
  – Bandits for Web Search
  – Exploration in Online Advertising

• Conclusion
Maximising Long-Term Reward is Hard

- Marshmallow experiment
- Delayed gratification and self control
- Longitudinal study showed relation with SAT scores and drug use
- Walter Mischel, 1968, 1998
- Video
**Exploration-Exploitation Trade-Off**

- Given an unknown environment the agent faces a dilemma
  - **Exploit** current knowledge and optimise short-term reward or
  - **Explore** the environment to discover opportunities for a better policy

- Too little exploration may result in poor local policy optima.
- Too much exploration may result in random, undirected behaviour.
- Find sweet spot!
Overview

• Introduction

• Five Vignettes
  – $Q$-Learning and $\epsilon$-exploration for Fighting
  – User guided exploration for Racing
  – Monte-Carlo Tree Search for Computer Go
  – Bandits for Web Search
  – Exploration in Online Advertising

• Conclusion
## Tabular Q-Learning

### Q-Table

<table>
<thead>
<tr>
<th>game states</th>
<th>1ft / GROUND</th>
<th>2ft / GROUND</th>
<th>3ft / GROUND</th>
<th>4ft / GROUND</th>
<th>5ft / GROUND</th>
<th>6ft / GROUND</th>
<th>1ft / KNOCKED</th>
<th>2ft / KNOCKED</th>
<th>3ft / KNOCKED</th>
<th>4ft / KNOCKED</th>
<th>5ft / KNOCKED</th>
<th>6ft / KNOCKED</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>actions</strong></td>
<td>THROW</td>
<td>KICK</td>
<td>STAND</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1ft / GROUND</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2ft / GROUND</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>3ft / GROUND</strong></td>
<td>13.2</td>
<td>10.2</td>
<td>-1.3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4ft / GROUND</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5ft / GROUND</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6ft / GROUND</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1ft / KNOCKED</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2ft / KNOCKED</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3ft / KNOCKED</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4ft / KNOCKED</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>5ft / KNOCKED</strong></td>
<td>3.2</td>
<td>6.0</td>
<td>4.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6ft / KNOCKED</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Q-Learning Update Equation

\[
Q(s, a) \leftarrow Q(s, a) + \alpha \left[ r + \gamma \max_{a'} Q(s', a') - Q(s, a) \right]
\]
Results

• Game state features
  • Separation (5 binned ranges)
  • Last action (6 categories)
  • Mode (ground, air, knocked)
  • Proximity to obstacle
• Available Actions
  • 19 aggressive (kick, punch)
  • 10 defensive (block, lunge)
  • 8 neutral (run)
• Q-Function Representation
  • One layer neural net (tanh)

Reinforcement Learner

In-Game AI Code
Learning Aggressive Fighting

Reward for decrease in Wulong Goth’s health

Early in the learning process...

... after 15 minutes of learning
Learning “Aikido” Style Fighting

Punishment for decrease in either player’s health

Early in the learning process ...

... after 15 minutes of learning
Lesson 1

Simple $\epsilon$-greedy exploration is hard to beat!
(or shall I say kick?)

(See also Vermorel and Mohri, 2005)
Overview

• Introduction
• Five Vignettes
  – $Q$-Learning and $\epsilon$-exploration for Fighting
  – User guided exploration for Racing
  – Monte-Carlo Tree Search for Computer Go
  – Bandits for Web Search
  – Exploration in Online Advertising
• Conclusion
Reinforcement Learning for Car Racing: AMPS (Kochenderfer, 2005)

1. Collect Experience
2. Learn transition dynamics $D$ and rewards $R$ (counting)
3. Revise value function $V$ and policy $\pi$ (prioritised sweep)
4. Revise state-action abstraction
5. Return to 1 and collect more experience
Balancing Abstraction Complexity

Too Coarse

Just Right!

Too Fine

Representational Complexity
Project Gotham Racing 3

Real time racing simulation.

Goal: as fast lap times as possible.
State Representation and Reward

Laser Range Finder Measurements as Features

Progress along Track as Reward
Actions

- Coast
- Accelerate
- Brake
- Hard-Left
- Hard-Right
- Soft-Left
- Soft-Right
Project Gotham Racing

XBOX 360 Integration

• Efficient Implementation
• 60 fps
• video
Lesson 2

Your exploration policy need not be (or look) smart to be effective!
Overview

• Introduction

• Five Vignettes
  – $Q$-Learning and $\epsilon$-exploration for Fighting
  – User guided exploration for Racing
  – Monte-Carlo Tree Search for Computer Go
  – Bandits for Web Search
  – Exploration in Online Advertising

• Conclusion
The Game of Go

• Started about 4000 years ago in ancient China.
• About 60 million players worldwide.
• 2 Players: Black and White.
• Board: 19×19 grid (here 9x9 for simplification).
• Rules:
  – Turn: stone placed on vertex.
  – Capture.
• Aim: Gather territory
Computer Go

• 5th November 1997: Gary Kasparov beaten by Deep Blue.

• Best Go programs cannot beat strong amateurs.
Computer Go

• Minimax search defeated.

• High **Branching Factor**.
  – Go: ~200
  – Chess: ~35

• Complex **Position Evaluation**.
  – Stone’s value derived from configuration of surrounding stones.
Key Insights for Monte-Carlo Go

1. **Random play-outs provide weak evaluation:**
   Even under random play, the better position wins more often (Bruegmann, 1993)

2. **Bias play-outs towards better moves:**
   Bandit problem at each node of game tree with optimistic exploration (UCB/UCT, Kocsis & Szepesvari 2006)

3. **Speed-up estimate by ignoring move order:**
   All-moves-as-first or rapid value estimate (RAVE), (Gelly & Silver 2007)

4. **Use Prior knowledge to improve estimates:**
   Pattern libraries and influence functions to tune priors or prune tree (e.g., Stern et al 2006)
Monte Carlo Go

 Territory Hypothesis
Monte Carlo Go
Monte Carlo Go
Monte Carlo Go
Monte Carlo Go

This node
Seen: 3 times
Win: 2/3 times
Upper Confidence Intervals

- How to decide which move to try next?
- Use UCB1 algorithm (Auer et al, 2002) for every internal node.
  - Choose move $j$ to maximise $\bar{x}_j + \sqrt{\frac{2 \ln n}{n_j}}$
  - $\bar{x}_j$: average payoff of move $j$ in node
  - $n$: number of visits to node
  - $n_j$: number of times move $j$ played in node
- UCT paper: this leads to sufficient exploration despite drifting pay-offs from down stream tree
Success of Monte-Carlo Go

- MoGo first program to use UCT (Gelly, Teytaud et al, 2006)
  - Beats 8 dan professional with 9 handicap stones
  - On par with professional players on 9x9
  - Running on “Huygens” (800 proc., 4.7 GHz = 15 Tflops)
- Other strong programs:
  - Valkyria, CrazyStone, Zen, Fuego, Many Faces, GnuGo
- Commercial Applications
  - Many Faces of Go
  - The Path of Go on Xbox Live Arcade
- BTW, The best programs just use $\bar{x}_j$ and no UCB now!
Lesson 3

Theory can be a powerful guide, but must not stop us from finding the best practical solution!
Overview

• Introduction

• Five Vignettes
  – $Q$-Learning and $\epsilon$-exploration for Fighting
  – User guided exploration for Racing
  – Monte-Carlo Tree Search for Computer Go
  – Bandits for Web Search
  – Exploration in Online Advertising

• Conclusion
Traditional Web Search Paradigm

• Standard techniques for Web search engines
  – Boolean Retrieval
  – Static Rank (PageRank etc.)
  – Machine learning of ranking models from human labels of query-URL pairs.

• Challenges
  – Human labels are costly
  – Human labels may not reflect audience preference
  – New trends not picked up quickly enough
  – Difficult to model personal preferences
  – No relation to user engagement and page composition
User Feedback and Clicks

• Learn from usage data
  – More data available at less cost
  – Incorporates dynamics of users and documents
  – Useful for personalisation and targeting
  – Available even for small collections and niche scenarios if they are used

• Online learning scenario
  – Choose documents to show for immediate benefit (exploitation)
  – Choose documents to show to improve policy (exploration)
Generalisation across Documents

• Exponentially large or infinite action (i.e. documents) sets are intractable
• How can we generalise across documents, possibly based on a feature representation?
• Make smoothness assumptions on pay-offs dependent on metric distance (Lipschitz condition) → Metric Bandits (R. Kleinberg et al, 2008)
• Devise “adaptive” cover of action space to zoom in on high-payoff action to achieve optimal regret for any metric space.
Diversity of Search Results

• In web search we display 10 blue links to the user in ranked order

• With $n$ documents and $k$ slots, this gives $\binom{n}{k} k!$ possible actions

• Value of results page is not just the sum of values of individual results, but depends, e.g., on diversity.

• Use problem structure to keep problem tractable: One bandit per display position (slot)

• Ranked Bandit Algorithm (Radlinski et al, 2008) approaches optimal poly-time regret.
Dynamics and Mortality

• Search results may not be available permanently, but can disappear (Mortal MAPs, Chakrabarti et al, 2008)

• Algorithm needs to continuously explore and discover alternative documents to display (“arms to pull”)

• Make mortality assumption on arms, e.g., budgeted death or timed death.

• Algorithms must quickly find “good enough” arm and start exploiting, e.g., UCB1 on $k$-subset.
Lesson 4

Theory can be enriched to provide guidance in more differentiated practical scenarios!
Overview

• Introduction

• Five Vignettes
  – $Q$-Learning and $\epsilon$-exploration for Fighting
  – User guided exploration for Racing
  – Monte-Carlo Tree Search for Computer Go
  – Bandits for Web Search
  – Exploration in Online Advertising

• Conclusion
Display to users (expected bid): \[ b_1 \cdot p_1 \geq b_2 \cdot p_2 \geq \cdots \]

Charge advertisers (per click): \[ c_i = b_{i+1} \cdot \frac{p_{i+1}}{p_i} \]

Importance of accurate probability estimates:

- Increase user satisfaction by better targeting
- Provide better deal to advertisers
- Increase revenue by showing ads with high click-thru rate
AdPredictor: Bayesian Probit Regression

- Client IP:
  - 102.34.12.201
  - 15.70.165.9
  - 221.98.2.187
  - 92.154.3.86

- Match Type:
  - Exact Match
  - Broad Match

- Position:
  - ML-1
  - SB-1
  - SB-2

Impression Level Click-Through Rate Prediction

\[ p(p\text{Click}) \]
The Causal Loop

Click Probability Model

User
Query data

Ad selection. Ad filtering.

Ad ranking. Page layout.

User

Bids

x_l

q_l

y_l

Training signal

$$$

Slides thanks to Leon Bottou
A toy example

• Two queries
  Q1: “cheap diamonds”  (50% traffic)
  Q2: “google”  (50% traffic)

• Three ads
  A1: “cheap jewelry”
  A2: “cheap cars”
  A3: “engagement rings”

• More simplifications
  - We show only one ad per query
  - All bids are equal to $1.
1- Showing random ads

• Observed click through rates (per ad/query)

<table>
<thead>
<tr>
<th></th>
<th>A1 (cheap jewelry)</th>
<th>A2 (cheap cars)</th>
<th>A3 (engagement rings)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q1 (cheap diamonds)</td>
<td>10%</td>
<td>2%</td>
<td>12%</td>
</tr>
<tr>
<td>Q2 (google)</td>
<td>2%</td>
<td>2%</td>
<td>2%</td>
</tr>
</tbody>
</table>

• Overall click through rate:

\[
CTR = \frac{1}{2} \left( \frac{10 + 2 + 12}{3} + \frac{2 + 2 + 2}{3} \right) = 5 \%
\]
2- Training a click model
(using data collected by showing random ads)

Our toy example has only one feature $F$:
$F = \text{#words appearing in both query and ad}$

<table>
<thead>
<tr>
<th></th>
<th>A1 (cheap jewelry)</th>
<th>A2 (cheap cars)</th>
<th>A3 (engagement rings)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q1 (cheap diamonds)</td>
<td>10%</td>
<td>2%</td>
<td>12%</td>
</tr>
<tr>
<td>Q2 (google)</td>
<td>2%</td>
<td>2%</td>
<td>2%</td>
</tr>
</tbody>
</table>

$F=1 : \quad \text{Pclick} = \frac{10 + 2}{2} = 6\%$

$F=0 : \quad \text{Pclick} = \frac{12+2+2+2}{5} = 3.6\%$

Pclick metrics: $\text{RIG} \approx 0.0014$ (Relative Information Gain)
3- Using the click model

<table>
<thead>
<tr>
<th></th>
<th>A1 (cheap jewelry)</th>
<th>A2 (cheap cars)</th>
<th>A3 (engagement rings)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q1 (cheap diamonds)</td>
<td>10%</td>
<td>2%</td>
<td>12%</td>
</tr>
<tr>
<td>Q2 (google)</td>
<td>2%</td>
<td>2%</td>
<td>2%</td>
</tr>
</tbody>
</table>

Overall click through rate:

$$\frac{1}{2} \left( \frac{10 + 2}{2} + \frac{2 + 2 + 2}{3} \right) = 4\%$$

- CTR is poor because our feature does not identify (Q1,A3).
- Adding better features would improve both RIG (*measured on random ads*) and CTR (*measured on ads shown according to the click model*).

→ Improving RIG (*measured on random ads*) does the right thing.
4- Retraining the click model (using data collected by showing ads according to pclick)

$$F=1 : \quad P_{click} = \frac{10+2}{2} = 6\%$$

$$F=0 : \quad P_{click} = \frac{2+2+2}{3} = 2\% \quad \text{(changed !)}$$

Pclick metrics: $RIG \approx 0.032\%$

- RIG has improved because the data is different.
- Adding a feature that pinpoint (Q1,A3) would not improve RIG.
- Adding a feature that pinpoints (Q1,A2) would improve RIG.

$\Rightarrow$ Improving RIG (measured on shown ads) can only eliminate ads.

$\Rightarrow$ We have created a black hole ...
The black hole

• Ads can be occasionally sucked by the black hole.

• New ads can be born in the black hole.
  – Example: the “Cold Start” problem.

• Solution: allowing ads to escape the black hole.
  – Better than trying to prevent ads to go there!
  – Use Exploration!!
Thompson Heuristic

average: 25% (3 clicks out of 12 impressions)

average: 30% (30 clicks out of 100 impressions)
Lesson 5

Studying exploration and the “causal loop” will be increasingly important as more and more machine learning algorithms use what they have learnt to make decisions and then continue learning.
Lessons Learnt

1. Simple $\epsilon$-greed hard to beat!
2. Effective exploration may look stupid!
3. Theory can be a powerful guide!
4. Real-world needs more differentiated theory!
5. Watch out for the causal loop!