Machine Learning in Games

The Magic of Research in Microsoft Products

Joaquin Quiñonero Candela, Ralf Herbrich and Thore Graepel
Online Services and Advertising
Microsoft Research Cambridge
Overview

Why Machine Learning and Games?

Machine Learning in Video Games
- Drivatars™
- Reinforcement Learning

Machine Learning in Online Games
- TrueSkill™
- Halo 3

The Path of Go

Conclusions
Test Beds for Machine Learning

- Perfect instrumentation and measurements
- Perfect control and manipulation
- Reduced cost
- Reduced risk
- Great way to showcase algorithms

Improve User Experience

- Create adaptive, believable game AI
- Compose great multiplayer matches based on skill and social criteria
- Mitigate network latency using prediction
- Create realistic character movement
Games can be very hard!

- Partially observable stochastic games
  - States only partially observed
  - Multiple agents choose actions
  - Stochastic pay-offs and state transitions depend on state and all the other agents’ actions
  - Goal: Optimise long term pay-off (reward)

Just like life: complex, adversarial, uncertain, and we are in it for the long run!
Approximations

From single player’s perspective
- Partially Observable Markov Decision Process (POMDP)

Approximate Solutions
- Reinforcement Learning
- Unsupervised Learning
- Supervised Learning

What is the best AI?
- Always takes optimal actions
- Delivers best entertainment value
Overview

Why Machine Learning and Games?
Machine Learning in Video Games
  - Drivatars™
  - Reinforcement Learning
Machine Learning in Online Games
  - TrueSkill™
  - Halo 3
The Path of Go
Conclusions
Drivatar™

- Adaptive avatar for driving
- Separate game mode
- Basis of all in-game AI
- Basis of “dynamic” racing line
Demo: Forza Motorsport

XBOX Game

• Dynamic Racing Line
• Learning a Drivatar
• Using a Drivatar
The Racing Line Model
Two phase process:

1. Pre-generate possible racing lines prior to the race from a (compressed) racing table.
2. Switch the lines during the race to add variability.

- Compression reduces the memory needs per racing line segment
- Switching makes smoother racing lines.
Racing Tables

<table>
<thead>
<tr>
<th>Segments</th>
<th>$a_1$</th>
<th>$a_2$</th>
<th>$a_3$</th>
<th>$a_4$</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1" alt="Segment" /></td>
<td><img src="image2" alt="Red Line" /></td>
<td><img src="image2" alt="Red Line" /></td>
<td><img src="image2" alt="Red Line" /></td>
<td><img src="image2" alt="Red Line" /></td>
</tr>
<tr>
<td><img src="image3" alt="Segment" /></td>
<td><img src="image2" alt="Red Line" /></td>
<td><img src="image2" alt="Red Line" /></td>
<td><img src="image2" alt="Red Line" /></td>
<td><img src="image2" alt="Red Line" /></td>
</tr>
<tr>
<td><img src="image4" alt="Segment" /></td>
<td><img src="image2" alt="Red Line" /></td>
<td><img src="image2" alt="Red Line" /></td>
<td><img src="image2" alt="Red Line" /></td>
<td><img src="image2" alt="Red Line" /></td>
</tr>
<tr>
<td><img src="image5" alt="Segment" /></td>
<td><img src="image2" alt="Red Line" /></td>
<td><img src="image2" alt="Red Line" /></td>
<td><img src="image2" alt="Red Line" /></td>
<td><img src="image2" alt="Red Line" /></td>
</tr>
</tbody>
</table>
Minimal Curvature Lines
Overview

- Why Machine Learning and Games?
- Machine Learning in Video Games
  - Drivatars™
  - Reinforcement Learning
- Machine Learning in Online Games
  - TrueSkill™
  - Halo 3
- The Path of Go
- Conclusions
Reinforcement Learning

Agent

Learning Algorithm

Game

parameter update

action

reward / punishment

game state

action

game state
### Tabular Q-Learning

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

**actions**

$$Q(s, a) \leftarrow Q(s, a) + \alpha [r + \gamma \max_{a'} Q(s', a') - Q(s, a)]$$
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)
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
Overview

- Why Machine Learning and Games?
- Machine Learning in Video Games
  - Drivatars™
  - Reinforcement Learning
- Machine Learning in Online Games
  - TrueSkill™
  - Halo 3
- The Path of Go
- Conclusions
Motivation

- Competition is central to our lives
  - Innate biological trait
  - Driving principle of many sports

- Chess Rating for fair competition
  - ELO: Developed in 1960 by Árpád Imre Élő
  - Matchmaking system for tournaments

- Challenges of online gaming
  - Learn from few match outcomes efficiently
  - Support multiple teams and multiple players per team
The Skill Rating Problem

Given:
- Match outcomes: Orderings among $k$ teams consisting of $n_1$, $n_2$, ..., $n_k$ players, respectively.

Questions:
- Skill $s_i$ for each player.
- Global ranking among all players.
- Fair matches between teams of players.
Two Player Match Outcome Model

- Latent Gaussian performance model for fixed skills
- Possible outcomes: Player 1 wins over 2 (and vice versa)

\[
P(y_{12} = (1, 2) | p_1, p_2) = \mathbb{I}(p_1 > p_2)
\]
Efficient Approximate Inference

Gaussian Prior Factors

Fast and efficient approximate message passing using Expectation Propagation

Ranking Likelihood Factors
Applications to Online Gaming

**Leaderboard**
- Global ranking of all players

**Matchmaking**
- For gamers: Most uncertain outcome
- For inference: Most informative
- Both are equivalent!
Xbox 360 & Halo 3

Xbox 360 Live
- Launched in September 2005
- Every game uses TrueSkill™ to match players
- > 35 million players
- > 4 million matches per day
- > 2 billion hours of gameplay / month

Halo 3
- Launched on 25ᵗʰ September 2007
- Largest entertainment launch in history
- > 200,000 player concurrently (peak: 1,000,000)
Demo: Halo 3

Halo 3 Game

• Matchmaking
• Skill Stats
• Tight Matches
Overview

Why Machine Learning and Games?

Machine Learning in Video Games
- Drivatars™
- Reinforcement Learning

Machine Learning in Online Games
- TrueSkill™
- Halo 3

The Path of Go

Conclusions
Learning to Play Go
The Game of Go

- Started about 4000 years ago in ancient China.
- About 60 million players worldwide.
- 2 Players: Black and White.
- Board: $19 \times 19$ grid.
- 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 amateurs.
Minimax search defeated.

High Branching Factor.
- Go: ~200
- Chess: ~35

Complex Position Evaluation.
- Stone’s value derived from configuration of surrounding stones.
Monte Carlo Go

Territory Hypothesis
Monte Carlo Go
Monte Carlo Go
Monte Carlo Go

This node
Seen: 3 times
Win: 2/3 times
Learning From the Experts with TrueSkill
Learning From the Experts with TrueSkill
Learning From the Experts with TrueSkill
Learning From the Experts with TrueSkill
Prune Away the Bad Moves
Machine Learning Assisted Monte Carlo Go

Monte Carlo Go

- Play random games ('rollouts').
- Each game gives a sample win/loss.
- Average estimates position value.
- Store game tree in memory.
- Bootstrap rollout policy.

Pattern Ranking System

- Learn pattern rankings using TrueSkill.
- Moves chosen over other moves by experts are inferred to have higher value.
- Training Data: 200,000 Expert Go Games.

Pattern Pruning

- Too many possible moves to evaluate.
- Pattern system estimates move quality.
- Prune bad moves from the game tree.
Demo: The Path of Go

The Path of Go

- MSRC Go AI (written in F#)
- TrueSkill Match Making
- XNA Game Studio
Overview

- Why Machine Learning and Games?
- Machine Learning in Video Games
  - Drivatars™
  - Reinforcement Learning
- Machine Learning in Online Games
  - TrueSkill™
  - Halo 3
- The Path of Go
- Conclusions
Conclusions

- Computer games can be used as test beds for research.
- Machine learning can be used to improve the user experience in computer games.
- Both research and applications are in their infancy and there are many open questions.
- XNA framework exists to plug in machine learning algorithms.
- For more questions, please drop us a line.

Joaquin Quiñonero Candela, Ralf Herbrich, Thore Graepel
Online Services and Advertising Group