IJCAI Tutorial

Solving Games with Complex Strategy Spaces

Part II: Integrating Learning with Game Theory for Strategic Decision Making

Fei Fang
Carnegie Mellon University
feifang@cmu.edu
Outline

- Games with Human Players for Real-world Applications
  - Wildlife Conservation

- End-to-End Learning and Decision Making in Games
  - A differentiable learning framework for learning game parameters

- Learning-Powered Strategy Computation in Large Games
  - Leveraging Deep Reinforcement Learning

- Other Applications and Summary
Societal Challenges

Security & Safety

Environmental Sustainability

Mobility
Solution Approaches

Artificial Intelligence

Machine Learning / Reinforcement Learning

Game Theory
Recap: Security Games

- **Strong Stackelberg Equilibrium**
  - Defender: mixed strategy
  - Attacker: best response, break tie in favor of defender

<table>
<thead>
<tr>
<th>Adversary</th>
<th>Target #1</th>
<th>Target #2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Target #1</td>
<td>5, -3</td>
<td>-1, 1</td>
</tr>
<tr>
<td>Target #2</td>
<td>-5, 4</td>
<td>2, -1</td>
</tr>
</tbody>
</table>
Quiz

- How to get the defender’s mixed strategy in SSE in this problem?

<table>
<thead>
<tr>
<th>Adversary</th>
<th>Target #1</th>
<th>Target #2</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Target #1</strong></td>
<td>5, -3</td>
<td>-1, 1</td>
</tr>
<tr>
<td><strong>Target #2</strong></td>
<td>-5, 4</td>
<td>2, -1</td>
</tr>
</tbody>
</table>

Defender

- 55.6%
- 44.4%
How to get the defender’s mixed strategy in SSE in this problem?

- \( \text{AttEU}_1 = p \cdot (-3) + (1 - p) \cdot 4 = p \cdot 1 + (1 - p) \cdot (-1) = \text{AttEU}_2 \)
- Equilibrium: \( \text{DefStrat} = (0.556, 0.444), \text{AttStrat} = (1, 0) \)

<table>
<thead>
<tr>
<th>Adversary</th>
<th>Target #1</th>
<th>Target #2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Target #1</td>
<td>5, -3</td>
<td>-1, 1</td>
</tr>
<tr>
<td>Target #2</td>
<td>-5, 4</td>
<td>2, -1</td>
</tr>
</tbody>
</table>
Recap: SSE vs NE

- **Zero-sum**
  - SSE = NE = minimax = maximin
  - Approach 1: Single LP (minimax or maximin strategy)
  - Approach 2: Greedy allocation for security games

- **General-sum**
  - SSE $\geq$ NE
  - Computing NE: PPAD Complete, LCP (linear complementarity problem) formulation, Gambit solver
  - Computing SSE
    - Approach 1: Multiple LPs (each solve a subproblem)
    - Approach 2: A single MILP that combines all the LPs
    - Approach 3: Extended greedy allocation algorithm $O(n\log n)$ for security games
Example: Protecting Staten Island Ferry
Outline

- Games with Human Players for Real-world Applications
  - Wildlife Conservation

- End-to-End Learning and Decision Making in Games
  - A differentiable learning framework for learning game parameters

- Learning-Powered Strategy Computation in Large Games
  - Leveraging Deep Reinforcement Learning

- Other Applications and Summary
Wildlife Conservation
Human Behavior in Games

- Not always perfectly rational or behave as expected!
- Task: Predict where the poachers place snares
Learn from Human Subject Experiments

Debarun Kar, Fei Fang, Francesco Maria Delle Fave, Nicole Sintov, Milind Tambe. In AAMAS-15
Learn from Real-World Data

- Raw Dataset for Queen Elizabeth National Park
  - Covers 2520 sq. km
  - Patrol and poaching recorded

Learn from Real-World Data

Each data point represents a 1km×1km area in a season.
Challenge 1: Data Uncertainty
Challenge 2: Lack of Recorded Attacks

**Patrolled Cells** (Year)

<table>
<thead>
<tr>
<th>Year</th>
<th>2010</th>
<th>2011</th>
<th>2012</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>61.7</td>
<td>67.7</td>
<td>66.0</td>
</tr>
</tbody>
</table>

**Per 100 cells**

<table>
<thead>
<tr>
<th>Year</th>
<th>2013</th>
<th>2014</th>
<th>2015</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>54.6</td>
<td>59.0</td>
<td>61.0</td>
</tr>
</tbody>
</table>

**Not Attacked Patrolled Cells**

<table>
<thead>
<tr>
<th>Year</th>
<th>2010</th>
<th>2011</th>
<th>2012</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>86.0</td>
<td>91.9</td>
<td>89.5</td>
</tr>
</tbody>
</table>

**Per 100 cells**

<table>
<thead>
<tr>
<th>Year</th>
<th>2013</th>
<th>2014</th>
<th>2015</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>92.3</td>
<td>93.0</td>
<td>88.4</td>
</tr>
</tbody>
</table>

**Attacked Patrolled Cells**

<table>
<thead>
<tr>
<th>Year</th>
<th>2010</th>
<th>2011</th>
<th>2012</th>
</tr>
</thead>
<tbody>
<tr>
<td>14</td>
<td>8.1</td>
<td>10.5</td>
<td>7.7</td>
</tr>
</tbody>
</table>

**Per 100 cells**

<table>
<thead>
<tr>
<th>Year</th>
<th>2013</th>
<th>2014</th>
<th>2015</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>7</td>
<td>11.6</td>
<td></td>
</tr>
</tbody>
</table>
Quantal Response Model

- Classical model in behavioral game theory
- Probability of attacking target \( j \)

\[
q_j = \frac{e^{\lambda \cdot \text{AttEU}_j(x)}}{\sum_i e^{\lambda \cdot \text{AttEU}_i(x)}}
\]

- \( \lambda \): represents error level (=0 means uniform random)
  - Maximal likelihood estimation (\( \lambda = 0.76 \))
  - \( \max_{\lambda} f(\lambda) = \sum_j N_j \log(q_j) \)
  - Solved through gradient ascent \( \lambda \leftarrow \lambda + \alpha \nabla_{\lambda} f(\lambda) \)

Subjective Utility Quantal Response Model

SEU_j = \sum_k w_k f^k_j, \quad q_j = \frac{e^{\lambda \cdot \text{SEU}_j(x)}}{\sum_i e^{\lambda \cdot \text{SEU}_i(x)}}

- Past Success/Failure Induced Features +
- Coverage Probability + Reward/Penalty
- Attack Probability

Adapted Behavioral Game Theory Models

- **CAPTURE**
  - Real-world Data
  - Dynamic Bayes Net: Time Dependency & Imperfect Observation

---

**Limited Data, Predicting Everywhere, Slow Learning**

- Distance to rivers / roads / villages
- Ranger observation
- ...
Decision Tree

- **PROS**
  - High speed
  - Learn global poachers behavior
  - Learn nonlinearity in geo-spatial predictor

- **CONS**
  - No explicit temporal dimension
  - No aspect for label uncertainty
Markov Random Field

**PROS**
- Explicit spatial dimension
- Explicit temporal dimension
- Addresses label uncertainty

**CONS**
- Low speed
- Data greedy
Hybrid Model

Static Covariates

Spatial Coordinates

Gaussian Mixture Model

Geo-clusters
Hybrid Model

Taking it for a Test Drive: A Hybrid Spatio-temporal Model for Wildlife Poaching Prediction Evaluated through a Controlled Field Test. Shahrzad Gholami, Benjamin Ford, Fei Fang, Andrew Plumptre, Milind Tambe, Margaret Driciru, Fred Wanyama, Aggrey Rwetsiba, Mustapha Nsubaga, Joshua Mabonga. In ECML-PKDD 2017
Augment Dataset With Expert Knowledge

- Negative sampling: sample from unpatrolled regions
- Positive sampling: Estimate from rangers’ estimated scores
  - Collect answers for several sets of clusters $C^1, C^2$
  - Compute aggregated score $s = \min\{s_1(C^1_i), s_2(C^1_j), \ldots\}$, add unlabeled points as positive points if $s \geq 6$
Field Test 1 in Uganda (1 month)

- **Trespassing**
  - 19 signs of litter, ashes, etc.

- **Poached animals**
  - 1 poached elephant

- **Snaring**
  - 1 active snare
  - 1 cache of 10 antelope snares
  - 1 roll of elephant snares

- **Snaring hit rates**
  - Outperform 91% of months

---

Cloudy with a Chance of Poaching: Adversary Behavior Modeling and Forecasting with Real-World Poaching Data. Debarun Kar, Benjamin Ford, Shahrzad Gholami, Fei Fang, Andrew Plumptre, Milind Tambe, Margaret Driciru, Fred Wanyama, Aggrey Rwetsiba. In AAMAS-17
Field Test 1 in Uganda: Base rate comparison
Field Test 1 in Uganda: % Months Exceeded Historical

- Animal Commercial
- Animal Noncommercial
- Fishing
- Plant Noncommercial
- Trespassing

Percentile
Field Test 2 in Uganda (8 months)

- 27 areas (9-sq km each)
- 454 km patrolled in total
- No point > 5 km from patrol post
- No area patrolled too much/rarely
- No overlapping areas
- <= 2 areas per patrol post
Field Test 2 in Uganda (8 months)

- 2 experiment groups
  - 1: $\geq 50\%$ attack prediction rate
    - 5 areas
  - 2: $< 50\%$ attack prediction rate
    - 22 areas

- Catch Per Unit Effort (CPUE)
  - Unit Effort = km walked
Field Test in China

- Two-day field test in October 2017: 22 snares
- 34 patrols from November 2017 to February 2018
  - 7 snares
Where to place snares?

Where to patrol?

From Prediction to Prescription

Machine Learning

Game Theoretic Reasoning

Route Planning
Game Theoretic Reasoning Based on Learned Model

- Find optimal patrol strategy given poachers respond to the patrol strategy according to learned model

- Challenges
  - Learned model is hard to represent using closed form function (e.g., decision tree)
  - Hard to scale up when considering scheduling constraints
Game Theoretic Reasoning Based on Learned Model

- **Input:** A machine learning model that predicts snares
- **Output:** an optimal patrolling strategy
- **Goal:** maximize catches of snares

Current period

Previous period

happened

To be planned

Optimal Patrol Planning for Green Security Games with Black-Box Attackers. Haifeng Xu, Benjamin Ford, Fei Fang, Bistra Dilkina, Andrew Plumptre, Milind Tambe, Margaret Driciru, Fred Wanyama, Aggrey Rwetsiba, Mustapha Nsubaga, Joshua Mabonga. In GameSec-17: The 8th Conference on Decision and Game Theory for Security
Game Theoretic Reasoning Based on Learned Model

For each cell $i$:

$x_i$: Current patrol effort at $i$

$y_i$: Prob. of detecting a snare at $i$ in current period

$g_i$

Optimization problem: $\max \sum_i g_i(x_i)$

However…

Patrol post (one patroller)
Game Theoretic Reasoning

- **Observe**: a pure strategy = a path from $v_{11}$ to $v_{1T}$
- **Claim**: a mixed strategy $\iff$ one-unit fractional flow from $v_{11}$ to $v_{1T}$
- **Patrol effort at cell $i$**: = the aggregated flow through cell $i$
- Build a mixed integer linear program

Time-unrolled Graph
Game Theoretic Reasoning Based on Learned Model

- A MILP formulation

\[
\begin{align*}
\text{maximize} & \quad \sum_{i=1}^{N} \left( g_i(0) + \sum_{j=1}^{m} z_i^j \cdot [g_i(j) - g_i(j-1)] \right) \\
\text{subject to} & \quad x_i \geq \sum_{j=1}^{m} z_i^j \cdot [\alpha_j - \alpha_{j-1}], \\
& \quad x_i \leq \alpha_1 + \sum_{j=1}^{m} z_i^j \cdot [\alpha_{j+1} - \alpha_j], \\
& \quad z_i^1 \geq z_i^2 \ldots \geq z_i^m, \\
& \quad z_i^j \in \{0, 1\}, \\
& \quad x_i = \sum_{t=1}^{T} \left[ \sum_{e \in \sigma^+(v_{t,i})} f(e) \right], \\
& \quad \sum_{e \in \sigma^+(v_{t,i})} f(e) = \sum_{e \in \sigma^-(v_{t,i})} f(e), \\
& \quad \sum_{e \in \sigma^+(v_{T,i})} f(e) = \sum_{e \in \sigma^-(v_{1,i})} f(e) = 1 \\
& \quad 0 \leq x_i \leq 1, \quad 0 \leq f(e) \leq 1,
\end{align*}
\]

\[\approx \max_{x_i} \sum_{i} g_i(x_i)\]

\[x_i = z_i^1 + z_i^2 + \ldots\]

Patrol effort at cell \(i\) = the aggregated flow through cell \(i\)

\(f\) is a unit flow
Complex Terrain
Complex Terrain

Patrol Route (2D)

Patrol Route (3D)
Trial Patrol in the Field

- 8-hour patrol in April 2015: patrolling is not easy!
Spatial Constraint
Spatial Constraint

- Grid based → Route based
- Hierarchical modeling: Focus on terrain features
- Build virtual street map
Spatial Constraint

- Hierarchical model: Focus on terrain feature
Patrol Route Design
Field Test in Malaysia

- In collaboration with Panthera, Rimba
- Regular deployment since July 2015 (Malaysia)
Real-World Deployment

Grid Based

Route Based

Fei Fang
Real-World Deployment

Animal Footprint

Tree Mark

Tiger Sign

Camping Sign

Lighter
Real-World Deployment

![Bar chart showing human and animal activity sign per kilometer for different patrols.](chart)

- **Human Activity Sign/km**
  - Previous Patrol
  - PAWS Patrol
  - Explorative PAWS Patrol

- **Animal Sign/km**
  - Previous Patrol
  - PAWS Patrol
  - Explorative PAWS Patrol

Fei Fang

8/12/2019
PAWS: Protection Assistant for Wildlife Security

- Past Patrolling and Poaching Information
- Protected Area Information
- Learn Behavior Model
- Game-theoretic Reasoning
- Route Planning
- Patrol Routes
- Poaching Data Collected
PAWS: Protection Assistant for Wildlife Security

- PAWS is deployed in the field
  - Saved animals!
Outline

- Games with Human Players for Real-world Applications
  - Wildlife Conservation
- End-to-End Learning and Decision Making in Games
  - A differentiable learning framework for learning game parameters
- Learning-Powered Strategy Computation in Large Games
  - Leveraging Deep Reinforcement Learning
- Other Applications and Summary
What game are we/they playing?

- Common criticism: game parameters are fully known
  - E.g. target importance
- How to learn parameters of 2-player zero sum games from opponents’ or players’ actions?
Forward Problem: Game Solving

Solve

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>−1</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>−1</td>
</tr>
<tr>
<td>−1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Equilibrium strategies

\[ u^* = v^* = \left[ \frac{1}{3}, \frac{1}{3}, \frac{1}{3} \right] \]
# Inverse Problem: Game Learning

i.i.d samples from equilibrium strategies

\[
\begin{align*}
\alpha^{(1)} &= (\text{hand}, \text{hand}) \\
\alpha^{(2)} &= (\text{hand}, \text{fist}) \\
\alpha^{(3)} &= (\text{fist}, \text{hand}) \\
&\quad \ldots
\end{align*}
\]

<table>
<thead>
<tr>
<th></th>
<th>Hand</th>
<th>Hand</th>
<th>Fist</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rock</td>
<td>??</td>
<td>??</td>
<td>??</td>
</tr>
<tr>
<td>Paper</td>
<td>??</td>
<td>??</td>
<td>??</td>
</tr>
<tr>
<td>Scissors</td>
<td>??</td>
<td>??</td>
<td>??</td>
</tr>
</tbody>
</table>
What game are we/they playing?

- Previous work on this topic
  - Directly learn good strategies from data (e.g. Letchford et al., 2009)
  - Rely on special game structures (Vorobeychik et al., 2007)
  - Computational Rationalization framework (Waugh et al., 2011)
Differentiable Learning

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>$-b_1$</th>
<th>$-b_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>$b_1$</td>
<td>0</td>
<td>$-b_3$</td>
</tr>
<tr>
<td>$-b_1$</td>
<td>$b_3$</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

- i.i.d samples from equilibrium strategies
  - $a^{(1)} = (\text{ }, \text{ })$
  - $a^{(2)} = (\text{ }, \text{ })$
  - $a^{(3)} = (\text{ }, \text{ })$

- Guess the value of $b_i$
- Compute equilibrium of guessed game
- Check if the computed equilibrium consistent with data
- Adjust the value of $b_i$ to increase consistency
- Repeat until satisfied

\[ \text{Update } b_i := b_i - \frac{\partial L}{\partial b_i} \]
NE and QRE in Zero-Sum Games

Recall LP for computing NE

\[
\begin{align*}
\min_{x, u} & \quad x \\
\text{s.t.} & \quad x \geq \sum_i u_i P_{ij}, \forall j \\
& \quad \sum_i u_i = 1, u_i \geq 0, \forall i
\end{align*}
\]

Nash Equilibrium

- Assumes perfect rationality
- May have multiple equilibria
- Discontinuous w.r.t. \( P \)

Recall Quantal Response

\[
q_j = \frac{e^{\lambda \cdot \text{AttEU}_j(x)}}{\sum_i e^{\lambda \cdot \text{AttEU}_i(x)}}
\]

Quantal Response Equilibrium

- Captures bounded rationality
- Unique
- Continuous w.r.t. \( P \)

\[
\begin{align*}
\min_u \max_v & \quad u^T P v - \sum_i v_i \log v_i + \sum_i u_i \log u_i \\
\text{s.t.} & \quad 1^T u = 1, u \geq 0 \\
& \quad 1^T v = 1, v \geq 0
\end{align*}
\]

\[
\begin{align*}
u_i^* &= \frac{\exp(Pv)_i}{\sum_q \exp(Pv)_q}, v_j^* &= \frac{\exp(P^T u)_j}{\sum_q \exp(P^T u)_q}
\end{align*}
\]
Learning of normal form games

- QRE = solution of min-max convex-concave problem

\[
\begin{align*}
\min_u \max_v u^T P v - \sum_i v_i \log v_i + \sum_i u_i \log u_i \\
1^T u = 1, \ 1^T v = 1
\end{align*}
\]

- KKT conditions:

\[
\begin{align*}
P v + \log(u) + 1 + \mu 1 &= 0 \\
P^T u - \log(v) - 1 + \nu 1 &= 0 \\
1^T u &= 1, \ 1^T v = 1
\end{align*}
\]

- Forward pass: Apply Newton’s Method

Recall: Newton’s Method for 1-D:

\[
x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}
\]

Generally, for nonlinear system

\[
J_F(x_n)(x_{n+1} - x_n) = -F(x_n)
\]

\[
\begin{bmatrix}
\text{diag}\left(\frac{1}{u}\right) & P \\
P^T & -\text{diag}\left(\frac{1}{v}\right)
\end{bmatrix}
\begin{bmatrix}
\Delta u \\
\Delta v \\
\Delta \mu \\
\Delta \nu
\end{bmatrix}
= -\begin{bmatrix}
P v + \log(u) + 1 + \mu 1 \\
P^T u - \log(v) - 1 + \nu 1 \\
1^T u - 1 \\
1^T v - 1
\end{bmatrix}
\]
Learning of normal form games

- Backward pass: Gradients of $P$ may be obtained via the implicit function theorem

\[ \nabla_P L = y_u v^T + u y_v^T, \]

where

\[
\begin{bmatrix}
y_u \\
y_v \\
y_\mu \\
y_\nu \\
\end{bmatrix} = \begin{bmatrix}
\text{diag}(\frac{1}{u}) \\
P^T \\
1^T \\
0 \\
\end{bmatrix} \begin{bmatrix}
P & 1 & 0 \\
-P & 0 & 1 \\
1^T & 0 & 0 \\
0 & 1^T & 0 \\
\end{bmatrix}^{-1} \begin{bmatrix}
-\nabla_u L \\
-\nabla_v L \\
0 \\
0 \\
\end{bmatrix}
\]
Learning in the presence of features

\[ b_1(x) \quad 0 \quad -b_3(x) \]

\[ b_2(x) \quad 0 \quad -b_3(x) \]

\[ -b_2(x) \quad b_3(x) \quad 0 \]

i.i.d samples from equilibrium strategies

\[ a^{(1)} = (\text{ }, \text{ }) \]
\[ a^{(2)} = (\text{ }, \text{ }) \]
\[ a^{(3)} = (\text{ }, \text{ }) \]

...  

Context

\[ x^{(1)} = [0.1, 0.5] \]
\[ x^{(2)} = [0.3, 0.7] \]

...
Learning in the presence of features

- Figure out which features attract/discourage attackers
  - Better understand attacker’s interests
  - Design better configurations which favor defenders

- Predict each player’s mixed strategy given an *new* environment
  - In practice, environment is changing over time
Learning in the presence of features

- Context (feature) $x^{(i)}$ and payoff matrix $P_\Phi(x^{(i)})$, parameterized by $\Phi$

- Each player acts according to a mixed strategy $(u, v)$ given by the QRE of $P_\Phi(x^{(i)})$, giving realizations $a^{(i)}$

- Objective: Learn $\Phi$ from $\{x^{(i)}, a^{(i)}\}$
End-to-end learning

Algorithm 1: Learning parameters $\Phi$ using SGD

Input: training data $\{(x^{(i)}, a^{(i)})\}$, learning rate $\eta$, $\Phi_{\text{init}}$

for $ep \in \{0, \ldots, ep_{\text{max}}\}$ do
    Sample $(x^{(i)}, a^{(i)})$ from training data;
    Forward pass: Compute $P_\Phi(x^{(i)})$, QRE $(u, v)$ and loss $L(a^{(i)}, u, v)$;
    Backward pass: Compute gradients $\nabla_u L, \nabla_v L, \nabla_P L, \nabla_\Phi L$;
    Update parameters: $\Phi \leftarrow \Phi - \eta \nabla_\Phi L$;
end

Main contribution

$\nabla_\Phi L$
Extensive form Games

- Let \((u, v)\) be strategies in sequence form
- Equilibrium is expressed as solution using \textit{dilated entropy regularization} (Equivalent to solving QRE for the reduced normal form)

\[
\min_u \max_v u^T P v - \sum_i \sum_a v_a \log \left( \frac{v_a}{v_{pi}} \right) + \sum_i \sum_a u_a \log \left( \frac{u_a}{u_{pi}} \right)
\]

\(Eu = e, Fv = f\)

\[
\nabla_P L = y_u v^T + u y_v^T,
\]

where

\[
\begin{bmatrix}
y_u \\
y_v \\
y_\mu \\
y_\nu
\end{bmatrix} = \begin{bmatrix}
-\Xi(u) & P & E^T & 0 \\
PT & \Xi(v) & 0 & F^T \\
E & 0 & 0 & 0 \\
0 & F & 0 & 0
\end{bmatrix}^{-1} \begin{bmatrix}
-\nabla_u L \\
-\nabla_v L \\
0 \\
0
\end{bmatrix}
\]
Resource Allocation Security Game

- Defender: \( r \) resources, \( n \) targets
  - Can allocate multiple resources to one target
- Attacker choose a target to attack
- Each target has value \( R_i \)
- If target \( i \) is protected by \( x \) resources and is attacked:
  \[
  U_a = \frac{R_i}{2x} = -U_d
  \]
- Attacker may learn \( R_i \) from observed defender actions
- Extend to \( T \)-stage game
Resource Allocation Security Game

\[ n = 2, r = 5 \]
One-Card Poker

- Learn players’ belief of card distribution
- Variant of Kuhn Poker with 4 cards, with non-uniform card distributions
- Observe actions of each player (e.g. raise, fold)
- Probabilities for chance nodes are embedded in $P_\Phi$
Featurized Rock Paper Scissors

\[ P = \begin{array}{ccc}
R & P & S \\
R & 0 & -b_1 & b_2 \\
P & b_1 & 0 & -b_3 \\
S & -b_2 & b_3 & 0 \\
\end{array} \]

\[ b = \Phi x, \]
\[ x \in [0, 1]^2 \]
\[ \Phi \in [0, 10]^{3 \times 2} \]

Objective is to learn \( \Phi \)
Recall in the basic approach, each step in the Newton’s method of each forward pass requires solving a linear system → Time consuming

\[
\begin{bmatrix}
\text{diag}(\frac{1}{u}) & P \\
p^T & -\text{diag}(\frac{1}{v}) \\
1^T & 0 \\
0 & 1^T
\end{bmatrix}
\begin{bmatrix}
\Delta u \\
\Delta v \\
\Delta \mu \\
\Delta \nu
\end{bmatrix} = -
\begin{bmatrix}
Pv + \log(u) + 1 + \mu_1 \\
P^T u - \log(v) - 1 + v_1 \\
1^T u - 1 \\
1^T v - 1
\end{bmatrix}
\]

Solution: Use first-order iterative method (FOM) to solve the forward pass directly

\[
\min_u \max_v u^T Pv - \sum_i v_i \log v_i + \sum_i u_i \log u_i
\]

\[
1^T u = 1, \quad 1^T v = 1
\]
The problem in the forward pass is a problem of the following min-max format, where the last two terms are strictly convex functions:

$$\min_{E x = x_0} \max_{F y = y_0} x^T P y + \mathcal{E}(x) - \mathcal{F}(y)$$

This problem can be solved using various FOMs.

**Input:** $x^{(0)}, y^{(0)}, P, \tau, \sigma$

**for** $i$ **in** {0, \ldots } **do**

- $\tilde{y} = y^{(i)}$
- $x^{(i+1)} = \text{BR}_x(x^{(i)}, \tilde{y}; P, \tau)$
- $\tilde{x} = 2x^{(i+1)} - x^{(i)}$
- $y^{(i+1)} = \text{BR}_y(y^{(i)}, \tilde{x}; P, \sigma)$

**BR is smoothed best response**

$$\text{BR}_x(\bar{x}, \tilde{y}) = \arg \min_{E x = x_0} x^T P \tilde{y} + \mathcal{E}(x) + \frac{1}{\tau} D_x(x, \bar{x})$$

$$\text{BR}_y(\bar{y}, \tilde{x}) = \arg \min_{F y = y_0} -\tilde{x}^T P y + \mathcal{F}(y) + \frac{1}{\sigma} D_y(y, \bar{y}).$$
Improve Scalability using FOM

Surprisingly, solving each step in the backward pass can also be converted to solving a problem with the min-max format. So same FOM can be applied.

\[
\begin{bmatrix}
y_u \\
y_v \\
y_\mu \\
y_\nu \\
\end{bmatrix} = \begin{bmatrix}
-\Xi(u) & P & E^T & 0 \\
P^T & \Xi(v) & 0 & F^T \\
E & 0 & 0 & 0 \\
0 & F & 0 & 0 \\
\end{bmatrix}^{-1} \begin{bmatrix}
-\nabla_u L \\
-\nabla_v L \\
0 \\
0 \\
\end{bmatrix}
\]

KKT Conditions

\[
\min_{x} \max_{y} \quad x^T P y + \frac{1}{2} x^T \Xi(u)x - \frac{1}{2} y^T \Xi(v)y + \nabla_u L^T x + \nabla_v L^T y
\]

subject to \( E x = 0 \) \( F y = 0 \).
Speedup in Forward Pass

Depth of game tree increase

#actions increase
Speedup in Backward Pass

Depth of game tree increase

#actions increase
Outline

- Games with Human Players for Real-world Applications
  - Wildlife Conservation

- End-to-End Learning and Decision Making in Games
  - A differentiable learning framework for learning game parameters

- Learning-Powered Strategy Computation in Large Games
  - Leveraging Deep Reinforcement Learning

- Other Applications and Summary
Solving Game through Learning from Self Play

AlphaGo vs AlphaGo

https://www.youtube.com/watch?v=Ue4A2Y_i3ZQ
More Complex Games: Patrol with Real-Time Information

- **Sequential interaction**
  - Players make flexible decisions instead of sticking to a plan
  - Players may leave traces as they take actions

- **Example domain: Wildlife protection**

![Footprints](image1)
![Lighters](image2)
![Old poacher camp](image3)
![Tree marking](image4)
Multi-Agent Reinforcement Learning

Defender’s view

Footprints of attacker

Destructive tools placed by the attacker

Footprints of defender

Features corresponding to the cell $\phi_{i,j}$ (animal density)

STRAT POINT

Attacker’s view
Compute Best Response by Training a Deep Q-Network

- **Q Network**: Game state $\rightarrow$ Q-value
- Use Deep Reinforcement learning to train the network and find optimal patrol policy (assuming fixed attacker)
Compute Best Response by Training a Deep Q-Network

DQN Defender vs Non-Adaptive Attacker

Attacker
Snares
Start from one of the corners

Defender
Start from Patrol Base

Start from
Patrol Base

8/12/2019
Compute Equilibrium: DQN + Double Oracle

Compute $\sigma^d, \sigma^a = \text{Nash}(G^d, G^a)$

Train $f^d = DQN(\sigma^a)$

Compute Nash / Minimax

Train $f^a = DQN(\sigma^a)$

Find Best Response to attacker’s strategy

Find Best Response to defender’s strategy

Add $f^d, f^a$ to $G^d, G^a$

Update basic strategy set
Enhancements

- Use local modes for efficient and parallelized training
- Start with domain-specific heuristic strategies
Solving Game through Learning from Self Play

- Green dots: Valuable trees
- Blue dots: Defender location
- Red dots: Logging locations
- Zero-sum game
- Goal: Find defender strategy or defender policy
Key idea 1: Represent mixed strategy using logit normal distribution in polar coordinate system

\[ r \sim P(N(\mu_r, \sigma_r^2)) \]
\[ \theta \sim P(N(\mu_\theta, \sigma_\theta^2)) \]
Solving Game through Learning from Self Play

- Key idea 2: Represent a “policy” with Convolutional Neural Network
  - Policy: mapping from game setting to strategy
  - CNN: Tree Distribution → Mean/Std of $r$ and $\theta$
Key idea 3: Approximate Fictitious Play

- Fictitious Play: Best responds to opponent's average strategy
- Average strategy → Random samples from history
- Best response → Update neural network
Solving Game through Learning from Self Play

Put them together

Algorithm 1: OptGradFP

Initialization. Initialize policy parameters $w_D$ and $w_O$, replay memory $mem$;

for $ep \in \{0, \ldots, ep_{max}\}$ do
  Simulate $n_s$ game play. Sample game setting and actions from current policy $\pi_D$ and $\pi_O$ $n_s$ times, save in $mem$;
  Replay for defender. Draw $n_b$ samples from $mem$, resample defender action from current policy $\pi_D$;
  Update parameter for defender. Update defender policy parameter
  $$w_D := w_D + \frac{\alpha_D}{1 + \epsilon_p \beta_D} \cdot \nabla w_D J_D;$$
  Replay for attacker. Draw $n_b$ samples from $mem$, resample attacker action from current policy $\pi_O$;
  Update parameter for attacker. Update attacker policy parameter
  $$w_O := w_O + \frac{\alpha_O}{1 + \epsilon_p \beta_O} \cdot \nabla w_O J_O$$
Solving Game through Learning from Self Play

- **Single game setting**

- **Multiple game setting**
  - Train on 1000 forest states, predict on unseen forest state
  - 7 days for training, Prediction time 90 ms
  - Shift computation from online to offline
Enhancement

- **DeepFP**
  - Generative network for approx. BR + game model network
  - Allow to use mathematical programming-based approach to compute BR for one or both players

**Data:** `max_games`, batch sizes `(m_1, m_2, m_G)`, memory size `E`, game simulator and oracle `BR_O_p` for players with no gradient

**Result:** Final belief densities \( \bar{\sigma}_p^* \) in `mem` \( \forall \) players `p`.

Initialize all network parameters \((\theta_1, \theta_2, \phi)\) randomly;
Create empty memory `mem` of size `E`;
for `game \in \{1, \ldots, max_games\}` do
  - Obtain best responses
  - Play game and update memory
  - Train shared game model net
  - Train best response nets

\[
L_{r_p}(\theta_p) = -\mathbb{E}_{(z_p \sim \mathcal{N}(0, I), u_{-p} \sim \bar{\sigma}_{-p})}[\hat{r}_p(BR_p(z_p; \theta_p), u_{-p}; \phi)]
\]
## Enhancement

<table>
<thead>
<tr>
<th>Forest structure</th>
<th>DeepFP</th>
<th>OptGradFP</th>
<th>DLP (approx. ground truth)</th>
</tr>
</thead>
<tbody>
<tr>
<td>F1</td>
<td>$\epsilon = 10.96 \pm 6.19$</td>
<td>$\epsilon = 21.72 \pm 4.47$</td>
<td>$\epsilon = 12.48 \pm 2.23$</td>
</tr>
<tr>
<td>F2</td>
<td>$\epsilon = 1.20 \pm 0.07$</td>
<td>$\epsilon = 1.28 \pm 0.07$</td>
<td>$\epsilon = 0.53 \pm 0.10$</td>
</tr>
<tr>
<td>F3</td>
<td>$\epsilon = 8.96 \pm 3.65$</td>
<td>$\epsilon = 14.58 \pm 0.20$</td>
<td>$\epsilon = 0.49 \pm 0.13$</td>
</tr>
<tr>
<td>F4</td>
<td>$\epsilon = 2.05 \pm 0.13$</td>
<td>$\epsilon = 2.18 \pm 0.09$</td>
<td>$\epsilon = 0.12 \pm 0.07$</td>
</tr>
</tbody>
</table>
Outline

- Games with Human Players for Real-world Applications
  - Wildlife Conservation

- End-to-End Learning and Decision Making in Games
  - A differentiable learning framework for learning game parameters

- Learning-Powered Strategy Computation in Large Games
  - Leveraging Deep Reinforcement Learning

- Other Applications and Summary
How Valuable is This Car?
Deception
Deception
Cyber Deception

- What can the defender do without “patrol boats”?
- Use deception to confuse the attackers!

Enterprise Network

Send probes to systems to gather information

Give information about systems on network

Attacker
Cyber Deception

- How should the defender disguise the systems to induce the adversary to attack the least valuable systems?

- **Cyber Domain Challenges:**
  - Intelligent adversary; could perceive deception occurring
  - Large number of system configurations and ways to disguise
  - Arbitrary deception may not be feasible or may affect performance
Cyber Deception Game: Setting

- $K$ systems, each has **true configuration** (TC) $f \in F$
- Successful attack on system with TC $f$ yields utility $U_f$ to attacker; defender loses $U_f$ (gains $-U_f$)
Cyber Deception Game: Setting

- Defender disguise the systems through deceptive responses
- Each system gets observed configuration (OC) $\tilde{f} \in \tilde{F}$
Cyber Deception Game: Defender

- Know true configuration (TC) $f$
- Need to decide observed configuration (OC) $\tilde{f}$
- Systems with same TC are indifferent to the defender
- $N_f = \text{Number of systems having TC } f \in F$
Cyber Deception Game: Defender

- Deception strategy encoded via integer matrix $\phi$
  - $\phi_{f,\tilde{f}} =$ number of systems with TC $f$ and OC $\tilde{f}$

<table>
<thead>
<tr>
<th>$\phi$</th>
<th>$\tilde{f}_1$</th>
<th>$\tilde{f}_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$f_1$</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>$f_2$</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>$f_3$</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

- Diagram:
  - $f_1$: $U_{f_1} = 10$
  - $f_2$: $U_{f_2} = 0$
  - $f_3$: $U_{f_3} = 5$
Cyber Deception Game: Defender

- Deception strategy encoded via integer matrix $\phi$
  - $\phi_{f,\tilde{f}} = \text{number of systems with } \text{TC } f \text{ and } \text{OC } \tilde{f}$
  - $\text{TC } f$ may not be masked with $\text{OC } \tilde{f}$ ($\pi_{f,\tilde{f}} = 0$)
  - Showing deceptive responses incur costs $c(f, \tilde{f})$; budget $B$

![Diagram showing the game with nodes labeled $f_1$, $f_2$, and $f_3$ with their utilities and costs]
Cyber Deception Game: Attacker

- Can observe OC of each system
- Cannot differentiate systems with same OC
- Uniformly randomly attacks systems with most attractive OC

How much does the attacker know about the deception?
Cyber Deception Game: Attacker

- Powerful attacker: Knows deception strategy $\phi$
  - Computes expected payoff for all OCs and best-responds
  - Robust assumption to minimize worst-case loss

<table>
<thead>
<tr>
<th>OC</th>
<th>$\tilde{f}_1$</th>
<th>$\tilde{f}_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$f_1$</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>$f_2$</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>$f_3$</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

Expected Payoff

$$\tilde{U}_{\tilde{f}} = \frac{\sum_{f \in F} \phi_{f, \tilde{f}} U_f}{\sum_{f \in F} \phi_{f, \tilde{f}}}$$

$$\tilde{U}_{\tilde{f}_1} = \frac{10 + 2 \times 0}{3} = 3.33$$

$$\tilde{U}_{\tilde{f}_2} = \frac{5}{1} = 5$$
Cyber Deception Game: Attacker

- **Powerful attacker**: Knows deception strategy $\phi$
  - Computes expected payoff for all OCs and best-responds
  - Robust assumption to minimize worst-case loss

- **Naive attacker**: Not aware of deception
  - Believe what they observe
  - Preset preferences (utilities) for attacking OCs
Quiz

- With powerful attacker, when there are no budget constraint and feasibility constraint, what is the optimal defender strategy?
Quiz

- With powerful attacker, when there are no budget constraint and feasibility constraint, what is the optimal defender strategy?

- Trivial case (no constraints): assign to same OC
Against Powerful Attacker

- Powerful attacker: Knows deception strategy $\phi$
  - Computes expected payoff for all OCs and best-responds
  - Robust assumption to minimize worst-case loss

- When some masking infeasible or budget limited

  **Theorem**: NP-hard to compute optimal strategy for defender against powerful adversary.

  - Proven via reduction to Partition problem
  - NP-hard even with just feasibility or just budget constraint
Against Powerful Attacker

- Solve through mathematical programming

\[
\begin{align*}
\min_{u, \phi} & \quad u \\
\text{s.t.} & \quad u \geq \frac{\sum_{f \in F} \phi_{f, \tilde{f}} U_f}{\sum_{f \in F} \phi_{f, \tilde{f}}} \quad \forall \tilde{f} \in \tilde{F}
\end{align*}
\]

Non-linear

Feasibility Constraints

\[
\begin{align*}
\sum_{\tilde{f}} \phi_{f, \tilde{f}} &= N_f \\
\sum_{\bar{f}} \phi_{f, \bar{f}} &= N_{\bar{f}} \\
\phi_{f, \tilde{f}} &\leq \pi_{f, \tilde{f}} \\
\phi_{f, \bar{f}} &\in \mathbb{Z}_{\geq 0}
\end{align*}
\]

Budget Constraint

\[
\sum_{f} \sum_{\tilde{f}} \phi_{f, \tilde{f}} c_{f, \tilde{f}} \leq B
\]
Against Powerful Attacker

- Solve through mathematical programming
- Reformulate to MILP: Guaranteed to find optimal solution
  - Remove the non-linear constraint
  - Adds $|K| |\tilde{F}|$ auxiliary variables
  - Adds $4|K| |\tilde{F}|$ additional constraints

- Approximation algorithm: Solve sequential MILPs
- Heuristic algorithm: Greedy MiniMax (GMM)
  - A fast heuristic which greedily minimizes attacker utility
Naive attacker: Not aware of deception
- Simply believes OCs (or just not reasoning about the actual TC→OC mapping strategy used by the defender)
- Preset preferences (utilities) for attacking OCs

When no budget constraints; but just the feasibility constraints

Theorem: can be solved in $O(|F| |\tilde{F}|)$ time

When both budget and feasibility constraints present

Theorem: NP-hard to compute optimal strategy for defender against naïve adversary.
Simulation Results

- 20 TCs, 20 Systems
- Attacker Utility = 10 without deception
Simulation Results

- Attacker model and belief of attacker model matters

Against Powerful Attacker

- Graph showing Adversary's Utility against the number of systems.

Against Naive Attacker

- Graph showing Adversary's Utility against the number of systems.
Evolution of Surge Pricing

- Surge price interface

Demand is off the charts! Fares have increased to get more Ubers on the road.

- 2.1x
  - THE NORMAL FARE
  - $16.80 MINIMUM FARE
  - $0.84 / MIN  $3.04 / KM

- I ACCEPT HIGHER FARE
- NOTIFY ME IF SURGE ENDS

- Economy
  - Fares are slightly higher due to increased demand
  - $4.99
    - 00:03
  - Premiu
  - $11.02
    - 23:58

REQUEST UBERX
Evolution of Surge Pricing

- Coarse → Fine grained in space
Quiz

- What are the potential strategic behavior of a driver (with old or new interface)?
Market Failure - 1
Bad draw dispatches: “after accepting, drivers are able to contact the rider. Some may [] learn [the] destination [] and canceling if [] the trip will not be worth the time.”
Competitive Equilibrium

- Competitive Equilibrium (CE)
  - Also called Walrasian equilibrium
  - Traditional concept in economics
  - Commodity markets with flexible prices and many traders
Competitive Equilibrium

- A very simple setting
  - A set of items \([n] = \{1, 2, \ldots, n\}\)
  - A set of buyers \([m] = \{1, 2, \ldots, m\}\)
  - Each buyer \(i\) has a valuation for each item \(j\): \(v_{ij}\)
  - Given a price vector \(p \in \mathbb{R}^n\), agent \(i\)'s utility is: 
    \[u_i(x; p) = v_i \cdot x - p \cdot x\]
    where \(x \in \{0,1\}^n\) indicates which items the agent gets
  - Each agent can get at most one item
Competitive Equilibrium

- A CE consists of:
  - A price vector \( p \in \mathbb{R}^n_+ \)
  - A valid allocation matrix \( x \)
    - \( x_{ij} \in \{0, 1\} \) indicates whether or not item \( j \) is allocated to agent \( i \)
    - Each item is allocated at most once \( \sum_i x_{ij} \leq 1, \forall j \)
    - Each buyer can get at most one item \( \sum_j x_{ij} \leq 1, \forall i \)
    - Use \( x_i \) to denote the binary vector for agent \( i \)
  - \( p \) and \( x \) satisfy the following constraints
    - Best response
      \[ x_i \in \arg\max_{x: x \in \{0, 1\}^n, \sum_j x_j \leq 1} u_i(x; p), \forall i \]
    - Market clearance
      \[ \forall j, \sum_i x_{ij} = 1 \text{ or } p_j = 0 \]
Myopic Pricing

- At current time $t$, each location has a sub-market
- Allocate cars to the riders with highest valuations
- Driver-pessimal price shown in black
With Myopic Pricing, at most, how much more can the purple driver earn if he deviates from the system’s assignment and all other drivers always follow the system’s assignment? (Options: $100, $90, $80, $0)
Purple driver rejects the assigned ride at 9:50am to earn more money.
Spatial-Temporal Pricing

- Model: Discrete time/location, Impatient riders, Anonymous origin-destination trip price

- One-shot assignment
  - Assignment plan: Decompose a min-cost flow
  - Pricing: Dual of flow LP
  - Form competitive equilibrium (CE)
    - Welfare optimal
    - Maximize total payment for each driver
    - Maximize utility for each rider
    - Envy free
    - All feasible driver payments in CE form a lattice
ILP for Computing Optimal Assignment Plan

\[
\begin{align*}
\text{max}_{x,y} & \quad \sum_{j \in R} x_j v_j - \sum_{i \in D} \sum_{k=0}^{\mid Z_i \mid} y_{i,k} \lambda_{i,k} \\
\text{s.t.} & \quad \sum_{j \in R} x_j 1\{(\omega_j, d_j, \tau_j) = (a, b, t)\} \leq \sum_{i \in D} \sum_{k=0}^{\mid Z_i \mid} y_{i,k} 1\{(a, b, t) \in Z_{i,k}\}, \\
& \quad \sum_{k=0}^{\mid Z_i \mid} y_{i,k} = 1, \quad \pi_i \\
x_j & \in \{0, 1\}, \\
y_{i,k} & \in \{0, 1\}, \\
x_j & \leq 1, \quad u_j \\
x_j & \geq 0 \\
y_{i,k} & \geq 0
\end{align*}
\]

Dual Variables

\[
p_{a,b,t} \quad \forall (a, b, t) \in T
\]

LP Relaxation

\[
\forall i \in D
\]

\[
\forall j \in R
\]

\[
\forall i \in D, \ k = 1, \ldots, \mid Z_i \mid
\]
Dual Problem to Compute CE Pricing

\[
\min \sum_{i \in \mathcal{D}} \pi_i + \sum_{j \in \mathcal{R}} u_j \\
\text{s.t. } \pi_i \geq \sum_{(a,b,t) \in \mathcal{Z}_{i,k}} p_{a,b,t} - \lambda_{i,k} \\
u_j \geq v_j - p_{o_j,d_j,r_j}, \\
p_{a,b,t} \geq 0, \\
u_j \geq 0,
\]

\[\forall k = 0,1,\ldots,|\mathcal{Z}_i|, \forall i \in \mathcal{D}\]

\[\forall j \in \mathcal{R}\]

\[\forall (a,b,t) \in \mathcal{T}\]

\[\forall j \in \mathcal{R}\]
Spatial-Temporal Pricing

- However…Drivers can deviate and trigger recomputation!

- Solution: Driver-Pessimal CE
  - Trip price = welfare gain difference
    \[ p_{a,b,t} = \Phi_{a,t} - \Phi_{b,t + dist(a,b)} \]
    \[ \Phi_{a,t} \triangleq W(D \cup \{(t, T, a)\}, R) - W(D, R) \]
  - Incentive compatible subgame perfect equilibrium
  - No driver want to deviate from assigned action!
Spatial-Temporal Pricing

- SPT vs Naïve surge
Summary

- Games with Human Players for Real-world Applications
- End-to-End Learning and Decision Making in Games
- Learning-Powered Strategy Computation in Large Games

Thank you!

Fei Fang
Carnegie Mellon University
feifang@cmu.edu
Security Challenges

Explosions near stadium
Restaurant shooting
Hostages at theater
Charlie Hebdo attack
Eiffel Tower

PARIS ATTACKS

Explosions in Brussels

Ansbach attack
A suicide bomb injured at least 12 in Germany’s Ansbach, near Nuremberg, on July 24. This is the fourth violent incident in Germany in a week.

Wurzbuerg
July 18
A Pakistan refugee injured five people with an axe near Wurzbuerg.

Ansbach
July 24
A suicide bomb injured at least 12.

Reutlingen
July 24
A Syrian refugee killed a pregnant woman.

Munich
July 22
A gunman shot nine people dead.

Source: Reuters
J. Wu, 25/07/2016
Sustainability Challenges

Today
≈ 3,200

100 years ago
≈ 60,000
Mobility Challenges