## Recent AI Breakthroughs

### Images
- Image recognition, reconstruction, generation, super-resolution,

### Molecules
- Protein folding, molecule design,

### Games
- Super-human play

### Time-series Data
- Speech recognition, forecasting

### Natural Language
- Text generation, translation, chatbots, text embeddings,
A Dawn of Multi-Agent Applications

Multi-player Game-Playing:
- Superhuman GO, Poker, Gran Turismo
- Human-level Starcraft, Diplomacy

• Multi-robot interactions
• Autonomous driving
• Automated Economic policy design

Generative Adversarial Networks (GANs)

\[ Z \sim \mathcal{N}(0, I) \rightarrow \text{neural network} \rightarrow \sim P_{\text{target}} \]

boring randomness \hspace{1cm} \text{neural network} \hspace{1cm} \text{interesting randomness}

synthetic data generation

Adversarial Training

robustifying models against adversarial attacks

"Stop Sign" + Adversarial Perturbation = "Yield Sign"
Example: Deep Generative Models

\[ Z \sim \mathcal{N}(0, I) \quad \Rightarrow \quad G_\theta(\cdot) \]

Deep Neural Network (DNN)
with well-tuned parameters \( \theta \)

\[ G_\theta(Z) \sim P_{\text{interesting}} \]
Example: Deep Generative Models

How to train a Deep Generative Model?

[Goodfellow et al’14]: Set up a two-player zero-game between a player tuning the parameters $\theta$ of a Deep Neural Network (called the “generator”) and a player tuning the parameters $\omega$ of a Deep Neural Network (called the “discriminator”)

$$Z \sim \mathcal{N}(0, I) \rightarrow G_\theta(\cdot) \sim P_{\text{interesting}}$$

Real or Hallucinated?

**Discriminator**: DNN w/ parameters $\omega$

- Reward discriminator for *distinguishing* real from fake images
- Reward generator for *fooling* the discriminator

**Generator**: DNN w/ parameters $\theta$

$$Z_1, Z_2, \ldots, Z_N \sim \mathcal{N}(0, I)$$

Hallucinated Images (from generator)  
Real Images (from training set)
A Dawn of Multi-Agent Applications

Multi-player Game-Playing:
- Superhuman GO, Poker, Gran Turismo
- Human-level Starcraft, Diplomacy

Multi-robot interactions
- Autonomous driving
- Automated Economic policy design

$Z \sim \mathcal{N}(0, I) \rightarrow$ neural network $\rightarrow$ $\sim P_{\text{target}}$

boring randomness interesting randomness

Generative Adversarial Networks (GANs)
- synthetic data generation

Adversarial Training
- robustifying models against adversarial attacks

Important notes and caveats...
(I) Strategic Behavior does not emerge from standard training
(I) Strategic Behavior does not emerge from standard training (cont’d)

I am the x player in a game of tic-tac-toe, the other player is o. I am supposed to play next, and the current board configuration looks as follows. Where should I put x?

```
x | x |
---+---
o | o | o
---+---
    |   |
```

Based on the current board configuration, you should put an "x" in the middle-right cell to block your opponent from winning the game.

The updated board would look like this:

```
x | x |
---+---
o | o | o
---+---
    |   |
```

```
(II) Naively trained models can be manipulated

"pig" + 0.005 x = "airliner"

[Athalye, Engstrom, Ilyas, Kwok ICML’18]

[Engstrom et al. 2019]
(III) Combining agents that were trained in isolation can lead to undesirable behavior

Example: AI for dynamic pricing

Setting: Duopoly w/ two symmetric firms

Independent Learning: firms cannot communicate other than setting prices, observing their profit and adjusting their price using some standard AI algorithm

(III) Combining agents that were trained in isolation can lead to undesirable behavior

Example: AI for dynamic pricing

Setting: Duopoly w/ two symmetric firms

Independent Learning: firms cannot communicate other than setting prices, observing their profit and adjusting their price using some standard AI algorithm

(IV) The optimization workhorse of Deep Learning struggles in multi-agent settings

\[ \min_{\theta} \ell(\theta) \]

- \( \theta \): high-dimensional
- \( \ell \): nonconvex

essentially only accessible through \( \ell(\theta) \) and \( \nabla \ell(\theta) \) queries

\[ \theta_{t+1} = \theta_t - \eta \cdot \nabla \ell(\theta_t) \]

Gradient Descent

**Theoretical Guarantee:** Even if \( \ell \) nonconvex, Gradient Descent efficiently computes *local minima* 

[Lee et al 2017, Ge et al ’15]

**Empirical Finding:** *Local minima* are good enough

Prominent Paradigm:

\[ \theta_{t+1} \leftarrow \theta_t - \nabla_{\theta} \ell(\theta_t) \]
(IV) The optimization workhorse of Deep Learning struggles in multi-agent settings

Practical Experience: GD vs GD (vs GD...) is cyclic or chaotic, and it is a hard engineering challenge to make it identify a good solution.
(IV) The optimization workhorse of Deep Learning struggles in multi-agent settings

GAN training on MNIST Data:
Target:

GAN training on Gaussian Mixture Data:
Target:

Simultaneous Gradient Descent (GD) Dynamics:
\[
\begin{align*}
\theta_{t+1} &= \theta_t - \eta \cdot \nabla_{\theta} \ell_G(\theta_t, \omega_t) \\
\omega_{t+1} &= \omega_t - \eta \cdot \nabla_{\omega} \ell_D(\theta_t, \omega_t)
\end{align*}
\]

\[\ell_D(\theta, \omega) = -\ell_G(\theta, \omega)\]
\(\ell_G, \ell_D\): nonconvex in \(\theta\) & \(\omega\) resp.;
\(\theta, \omega\): high-dimensional

pictures from [Metz et al ICLR’17]
(V) Finally Game Theory May Break

In applications involving DNNs, agents’ losses are non-convex (a.k.a. their utilities are non-concave).

Without further structure, this is trouble for Game Theory, in that standard ways to solve the game are not applicable.
Summary so far...

• (I) Strategic Behavior does not emerge from standard training
• (II) Naively trained models can be manipulated
• (III) Combining agents that were trained in isolation can lead to undesirable (e.g. collusive) behavior
• (IV) The optimization workhorse of Deep Learning (namely gradient descent) struggles in multi-agent settings
• (V) Finally Game Theory (namely standard models and solutions) are inadequate to address non-convexity
This class: lay modern foundations of multi-agent learning

- Game Theory: to incorporate models of strategic behavior
- Learning: to incorporate models of uncertainty, and learning
- Computation: to tackle complexity
- Humans: to incorporate models of strategic behavior
Today’s Menu

• Motivation
• Administrivia
• Course Overview
Administrivia

• Course website: https://web.mit.edu/~gfarina/www/6S890/
  • We will use the website as the public face of the course, and to post lecture notes and slides
  • Private discussions, questions, grading, etc. will be arranged on Canvas

• Lecturers: Costis Daskalakis, Gabriele Farina
  • TA? tbd

• Attendance: Everyone is welcome! If just auditing please register as a listener

• Office hours are flexible: email us to schedule
If Registered for Credit:

• Solve problem sets: 2-3 problems sets, two weeks to solve each (weight: 40%)

• Project: proposal due end of September (weight: 50%)
  • Project brainstorming class on 9/26
  • Project break on the week of 11/13-17
  • Project presentations starting 11/30
  • Project can be theoretical, practical, or a mix
  • We encourage creativity!
  • Feel free to run your ideas by us
  • Feel free to apply ideas from this class to your own area of interest
  • Project can be done in teams

• No exams
Any questions regarding logistics and projects?
Today’s Menu

- Motivation
- Administrivia
- Course Overview
Our goal in this class, revisited

*How can we, and machines, systematically reason about the behavior, incentives, and outcomes of multiagent systems?*

*And as computer scientists, how can we teach machines to compute, predict, or learn such behavior?*
The Concept of **Game**

*Games* are thought experiments to help us learn how to *predict rational behavior* in *situations of conflict*.

**Situation of conflict:** Everybody's actions affect others.

**Rational Behavior:** The players want to maximize their own expected utility. No altruism, envy, masochism, or externalities (if my neighbor gets the money, he will buy louder stereo, so I will hurt a little myself...).

**Predict:** We want to know what happens in a game. Such predictions are called *solution concepts* (e.g., Nash equilibrium).
Situations Modeled as Games

- Recreational games
  - Rock paper scissors
  - Diplomacy
  - Poker
  - Go
  - ...

- Non-recreational settings
  - Auctions
  - Markets
  - Logistics
  - Budget allocation (e.g., political campaigns)
  - Generative networks
  - Multi-robot interactions
  - Fraud detection systems
  - ...

Scenic Tour
Part I: Normal-Form Games

These are “matrix” games
- Simultaneous actions
- Single move per player

Simple model but already captures several important aspects

Rock-paper-scissors

<table>
<thead>
<tr>
<th></th>
<th>Deny (cooperate)</th>
<th>Confess (betray)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deny (cooperate)</td>
<td>-1, -1</td>
<td>-3, 0</td>
</tr>
<tr>
<td>Confess (betray)</td>
<td>0, -3</td>
<td>-2, -2</td>
</tr>
</tbody>
</table>

Prisoner’s dilemma
(-1 = 1 year in jail)
Despite their simplicity, normal-form games will provide the ground to start looking into the following key concepts in multiagent settings:

• Solution concepts and equilibria (Nash, maxmin, correlated, ...)

• Learning from repeated play
  • Learning enables iteratively refining strategies to become stronger and stronger, and it has been a key component in all recent game AI breakthroughs
  • Local learning of each agent can often be connected to global notion of equilibrium
  • ≈ Mental model: “reinforcement learning but also works in nonstationary settings”

• Deep connection between equilibria and other important concepts in computer science

• After that, we will move on to notions of games that capture more interesting / real-world phenomena, especially:
  • Sequential moves, Imperfect information, nonconvexity
What is “rational play” for the agents?

Example: What should happen in prisoner’s dilemma?

• From blue player’s point of view, Confess dominates Deny, i.e. no matter what orange plays blue is better off by playing Confess.

• Likewise, From orange’s point of view, Confess dominates Deny.

• So the rational strategy for both is to play Confess.

• It is a dominant strategy equilibrium.

• It is worse for both compared to (Deny, Deny)...

<table>
<thead>
<tr>
<th></th>
<th>Deny (cooperate)</th>
<th>Confess (betray)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deny (cooperate)</td>
<td>-1, -1</td>
<td>-3, 0</td>
</tr>
<tr>
<td>Confess (betray)</td>
<td>0, -3</td>
<td>-2, -2</td>
</tr>
</tbody>
</table>

Benefit of dominant strategy equilibrium: requires no "counterspeculation"
In general, counterspeculation cannot be avoided

- Dominant strategy equilibrium is the exception, not the norm: no strategies in general might be dominated

Furthermore, this example shows that it is generally a bad idea for any player to stick to a single action

Instead, players should play from a distribution over actions

General principle: randomization is generally required to play “optimally”
• Idea: Ideally, we would want a strategy that we are comfortable playing over and over
  • Even if the opponent learned about our distribution, and computed an optimal counterstrategy, we would not want to change our strategy

• In zero-sum games are lucky: if we settle on a strategy \( x \), we know that an “expert” will maximize their own utility, which is the opposite of ours. So, if we play against an expert we can predict that our utility will be

\[
\nu(x) = \min_y E_{a \sim x, b \sim y} [u(a, b)]
\]

• We then want to select a strategy \( x \) with maximum return:

\[
x^* \in \arg \max \nu(x)
\]
Maxmin strategies

• The notion we just introduced is called a maxmin strategy
• Natural when playing against a strong player in a two-player zero-sum game
  • E.g. was used to beat top poker pros in Head’s Up No Limit Hold’em
  • Does not require human data
  • Might not be the most natural choice against a weak opponent though
Maxmin strategies

• Computation of maximin strategies is tractable
  • Convex optimization problem
  • In fact, linear -> We can use the simplex algorithm or interior point methods

• Even better: there exist very attractive learning algorithm
  • Start by playing the game
  • After every round, refine the strategy according to the learning algorithm
  • Repeat
  • Many algorithms known: hedge, optimistic multiplicative weights, regret matching, online gradient ascent, ...
  • Extremely scalable and practical algorithm
Nash equilibria

- An assignment of maxmin strategies for each player in two-player zero-sum games forms a **Nash equilibrium**
  - That is, an assignment of independent strategies for each player, so that no player has any incentive to unilaterally deviate
  - Each player is best responding to each of the other players
  - Concept generalizes to any number of players and beyond zero-sum games

Since maxmin strategies can be computed in polynomial time, this means that a Nash equilibrium in two-player zero-sum games always exists and can be computed in polynomial time
Beyond Zero-Sum?
Nash’s Theorem

[John Nash ’50]: A Nash equilibrium exists in every finite game. Deep influence in Economics, enabling other existence results. Proof highly non-constructive (uses Brouwer’s fixed point thm) No simpler proof has been discovered

[Daskalakis-Goldberg-Papadimitriou’06]: no simpler proof exists i.e.

\[ \text{Nash Equilibrium} \quad \leftrightarrow \quad \text{Brouwer’s Fixed Point Theorem} \]
• In practice, people have been successful applying learning methods beyond two-player zero-sum settings and achieving human or even superhuman performance
  • Example: six-player poker was solved this way

• In general-sum games, learning dynamics also provably converge to relaxations of the Nash equilibrium (e.g., correlated and coarse-correlated equilibria), which are interesting on their own
# Part I: Normal-Form Games

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>9/12</td>
<td><strong>Setting and equilibria: Nash equilibrium</strong>&lt;br&gt;Definition of normal-form games. Properties of Nash equilibria. Nash equilibria in two-player games as linear and linear complementarity problems</td>
</tr>
<tr>
<td>3</td>
<td>9/14</td>
<td><strong>Setting and equilibria: Correlated equilibrium</strong>&lt;br&gt;Definition of Correlated and coarse correlated equilibria. Their relationships with Nash equilibria in two-player zero-sum games. Linear programming formulations</td>
</tr>
<tr>
<td>4</td>
<td>9/19</td>
<td><strong>Learning in games: Foundations</strong>&lt;br&gt;Regret and hindsight rationality. Phi-regret minimization and special cases. Connections with equilibrium computation and saddle-point optimization</td>
</tr>
<tr>
<td>5</td>
<td>9/21</td>
<td><strong>Learning in games: Algorithms</strong>&lt;br&gt;Regret matching, regret matching plus, hedge, FTRL and OMD</td>
</tr>
</tbody>
</table>

# Part Ib: Complexity of Equilibrium

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>9/28</td>
<td><strong>Nash equilibrium and PPAD complexity</strong>&lt;br&gt;Sperner's lemma, Brouwer's fixed point, and the PPAD complexity class. Nash's proof</td>
</tr>
<tr>
<td>8</td>
<td>10/3</td>
<td><strong>PPAD-completeness of Nash equilibria, and open problems.</strong></td>
</tr>
</tbody>
</table>
Parts II and III

Most interactions do not look like a normal-form games

Players can often make more than one move, and they often have imperfect information
Part II: Markov (aka Stochastic) Games

Markov games: many “normal-form” stage games played on a graph.
(≈ think: “MDP of normal-form games”)

Can be generally solved via **backward induction** (bottom up induction)

Surprising complexity of stationary equilibria

• Example: diplomacy
<table>
<thead>
<tr>
<th>Chapter</th>
<th>Date</th>
<th>Topic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Part II</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>10/5</td>
<td>Stochastic games</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Minimax theorem, and existence of equilibrium. Stationary Markov Nash equilibria.</td>
</tr>
<tr>
<td>10</td>
<td>10/12</td>
<td>Computation of equilibria in stochastic games</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Finite horizon vs infinite horizon (discounted) setting, the role of nonstationarity, and backward induction.</td>
</tr>
<tr>
<td>11</td>
<td>10/17</td>
<td>Minimax stationary Markov learning</td>
</tr>
<tr>
<td>12</td>
<td>10/19</td>
<td>Complexity of correlated equilibria in general-sum stochastic games</td>
</tr>
</tbody>
</table>
Part III: Imperfect-Information Extensive-Form Games

While stochastic games capture *sequential* moves, they do not address *imperfect information*.

Imperfect information extensive-form games: ≈ “stochastic games + imperfect information + tree structure (instead of general graph)”

• Example: poker
Difficulties with Imperfect Information

• Compared to normal-form games, imperfect-information extensive-form games bring many conceptual challenges

1. The number of (deterministic) strategies grows **exponentially** in the game tree

2. Imperfect information makes backward induction and local reasoning not viable

   Think about poker: need to reason about **misdirection**. **General principle**: you need to think about what the opponents don’t know about you and leverage that to your advantage

3. Other players have control over what part of the game tree is visited/explored

• Nonetheless: many positive results
  • In fact, we live in a world where machines bluff at poker better than humans
• As an example of a positive result, we will show that learning can be carried out efficiently in imperfect-information games
By the end of this part...
• By the end of this part, you should have acquired:
  • A **language** to think about and describe different equilibrium points of multiagent interactions (Nash equilibrium, maxmin strategies, correlated equilibria, ...)
  • An appreciation for what is **computationally tractable** in every case, and what only in special cases
  • The ability to **implement learning dynamics** to progressively refine strategies, including in imperfect-information domains
  • A general understanding of what techniques are used to push scalability, and what major areas of investigation remain **underexplored**
Part IV: Nonconvexity (preview)

<table>
<thead>
<tr>
<th></th>
<th></th>
<th>Aspects of nonconvex-nonconcave games (TBD)</th>
</tr>
</thead>
<tbody>
<tr>
<td>21</td>
<td>11/21</td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>11/28</td>
<td></td>
</tr>
</tbody>
</table>
Part IV: Nonconcave games

Emerging applications in Machine Learning involve multiple agents who:

- choose high-dimensional strategies $x_i \in X_i \subset \mathbb{R}^{d_i}$
- maximize utility functions $u_i(x_i ; x_{-i})$ that are typically nonconcave in their own strategy (a.k.a. minimize loss functions that are nonconvex in their own strategy)

Issue: Game Theory is fragile when utilities are nonconcave

- in particular, Nash equilibrium (and other types of equilibrium) may not exist
- so what is even our recommendation about reasonable optimization targets in the multi-agent setting?
And finally...

<table>
<thead>
<tr>
<th>Date</th>
<th>Date</th>
<th>Projects</th>
</tr>
</thead>
<tbody>
<tr>
<td>23</td>
<td>11/30</td>
<td>Projects</td>
</tr>
<tr>
<td>24</td>
<td>12/5</td>
<td>Projects</td>
</tr>
<tr>
<td>25</td>
<td>12/7</td>
<td>Projects</td>
</tr>
</tbody>
</table>