Lecture 23: Evaluation of Multi-Agents systems

Start Recording!

Reminders

- Office Hours tomorrow with Adrien (11-12AM)
- Last lecture today.
- Papers presentation the 27th
- Final Reports the 28th

Talk on StarCraft II by Wojciech M. Czarnecki

On Friday 23th **at noon**

References for this lecture:

1. Balduzzi, David, et al. "Re-evaluating evaluation." arXiv preprint arXiv:1806.02643 (2018).

Today: Empirical Games

First Part: Agents Vs. Agents

Today

Last time: many questions about how to estimate ELO.

This time:

- Estimate Elo !
- Why sometimes we should not only consider Elo.
- Beyond Elo !

AntiSymmetric (zero-sum) Game (Functional Form)

Anti-symmetric Payoff:

$$\varphi : \boxed{W \times W} \to \mathbb{R}$$

Players (example: RL policies)

$$\varphi(u, w) = -\varphi(w, u)$$

Intuition: Switching the roles switches the results.
Example: Chess, Go, Poker (need to randomize who starts)

NB: Can generalize to non-zero sum (just heavier because of the two losses)

AntiSymmetric (zero-sum) Game (Functional Form)

Anti-symmetric Payoff:

$$\varphi : \boxed{W \times W} \to \mathbb{R}$$

Players (example: RL policies)

$$\varphi(u, v) = \operatorname{logit}(\mathbb{P}(u \succ v))$$

Intuition: Switching the roles switches the results.

Example: Chess, Go, Poker (need to randomize who starts)

NB: Can generalize to non-zero sum (just heavier because of the two losses)

Example: Elo Rating

$$\mathbb{P}(u \succ w) = \frac{1}{1 + \exp(\alpha \cdot (f(w) - f(u)))}$$

f(u) : Elo Rating of u

$$\varphi(u, v) = \mathrm{logit}(\mathbb{P}(u \succ v)) = \alpha \cdot (f(u) - f(w))$$

Antisymetric payoff!!! :-)

Online Estimation of the Elo

Target: p_{ij}

Estimated proba. $\hat{p}_{ij} = \sigma(f_i - f_j)$

$$\ell(\hat{p}_{ij}, p_{ij}) = -p_{ij}\log(\hat{p}_{ij}) - (1 - p_{ij})\log(1 - \hat{p}_{ij})$$

Cross-entropy loss

True proba.

Score of a Match-up

(stochastic) Gradient Descent on that loss: $\ell(\hat{p}_{ij}, p_{ij}) = \mathbb{E}_{S_{ij}}[\ell(\hat{p}_{ij}, S_{ij})]$

$$f_i^{t+1} = f_i^t - \eta \nabla_{f_i} \ell(\hat{p}_{ij}, S_{ij}^t)$$

Exercice: derive this gradient

Online Estimation of the Elo

Target:

Estimated proba.

$$\hat{p}_{ij} = \sigma(f_i - f_j)$$

$$\ell(\hat{p}_{ij}, p_{ij}) = -p_{ij} \log(\hat{p}_{ij}) - (1 - p_{ij}) \log(1 - \hat{p}_{ij})$$

Cross-entropy loss

True proba.

Score of a Match-up

(stochastic) Gradient Descent on that loss: $\ell(\hat{p}_{ij}, p_{ij}) = \mathbb{E}_{S_{ij}}[\ell(\hat{p}_{ij}, S_{ij})]$

$$f_i^{t+1} = f_i^t + \eta(S_{ij}^t - \hat{p}_{ij}^t)$$

Take-away

- Optimization perspective on the ELO:
  Stochastic gradient descent with constant step-size

$$\min_{f_i} \mathbb{E}_{f_j \sim \text{pop}} \ell(\sigma(f_i - f_j), \mathbb{P}(i \succ j))$$

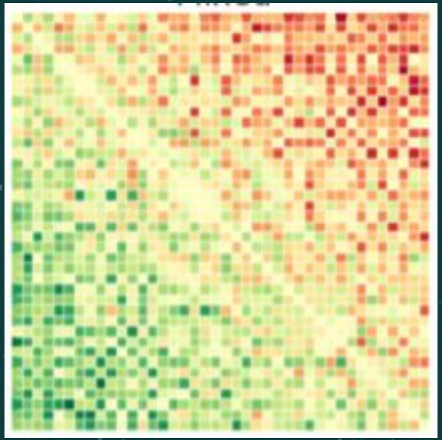
SGD with constant step-size does not converge.
(It only converges to a neighborhood proportional to the variance times the step-size)
Question: try to think why?

Estimation of the ELO at a given time!

Population of agents $\mathcal{B} = (u_i)$

Payoff matrix of the group: $A_{\mathcal{B}}$

$$[A_{\mathcal{B}}]_{ij} = \varphi(u_i, u_j)$$

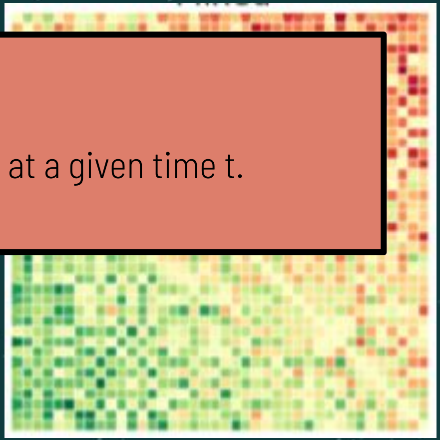


From last time

Population of agents $\mathcal{B} = (u_i)$

Payoff matrix of the group: $A_{\mathcal{B}}$

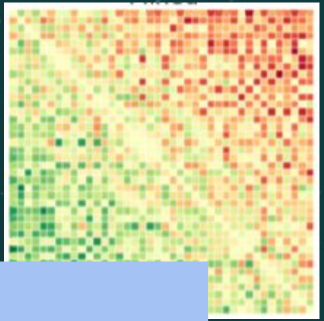
$[A_{\mathcal{B}}]$



The matrix contains 'simultaneous match-ups'

Question: How can we use that matrix to estimate Elo at a given time t.

Getting Elo From A



Intuition:

- If $A_{ij} = f_i - f_j$
- Then

Question (Simon):
We've seen that in a hypothetical tournament featuring all possible matchups, what you can calculate is [f_i - f_avg]. This emphasizes that to calculate f_i you need prior knowledge of f_avg.

- Drift of the ELo score?

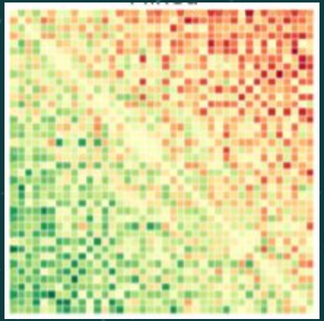
$A =$ ... $f^\top$

Average Elo

Individual Elo

$$f_i - \bar{f} = \frac{1}{n} \sum_{j=1}^{n} A_{ij}$$

Getting Elo From A



Intuition:

- If $A_{ij} = f_i - f_j$
- Then

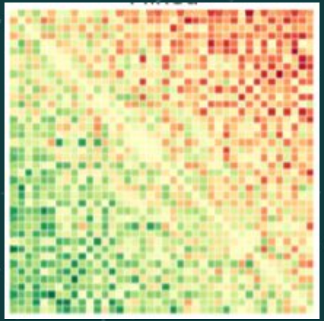
$$A = \begin{pmatrix} f_1 & \cdots & f_1 \\ \vdots & \ldots & \vdots \\ f_n & \cdots & f_n \end{pmatrix} - \begin{pmatrix} f_1 & \cdots & f_n \\ \vdots & \ldots & \vdots \\ f_1 & \cdots & f_n \end{pmatrix} = f\mathbf{1}^\top - \mathbf{1}f^\top$$

Average Elo

Individual Elo

$$f_i - \bar{f} = \frac{1}{n}\sum_{j=1}^{n} A_{ij}$$

Getting Elo From A



Theorem:

- If $\cancel{A_{ij} = f_i - f_j}$
- We have:

$$A = \underbrace{f\mathbf{1}^\top - \mathbf{1}f^\top}_{\text{Transitive component}} + \underbrace{B}_{\text{Cyclic component: } B\mathbf{1} = \mathbf{0}}$$

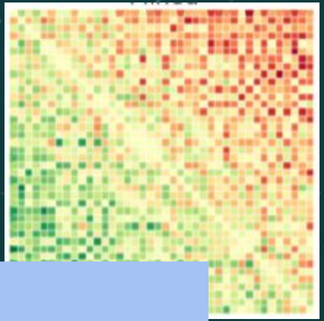
Take-away:

- ELO = f
- Meaningful if B << f

Cyclic component:

There exists cycles: **P1** beats P2, P2 beats P3, P3 beats **P1**

Getting Elo From A



Theorem:

- If $\cancel{A_{ij} = f_i - f_j}$
- We hav

Question (Semih):

- What's the intuition (or rather, theorem) behind the fact that a matrix A can be decomposed into transitive and cyclic components?
- What are the assumptions required such that such a decomposition exists?

Answer: It is more about identifying what is cyclic and what is transitive.

There exists cycles: **P1** beats P2, P2 beats P3, P3 beats **P1**

Why do we care about that

Elo is useful to predict win-loss probability:

- Under the assumption that the game is transitive

$$\mathbb{P}(i \succ j) = \sigma(f_i - f_j)$$

Assuming we 'know' f_i and f_j we can predict who will win.

We need a "higher-order" ELO in non-transitive games.

Higher Order Elo

Idea: "PCA" on B.

- B is skew-symmetric -> NO PCA but Schur decomposition!

$$A = f\mathbf{1}^\top - \mathbf{1}f^\top + B$$

$$B = O \begin{pmatrix} 0 & \lambda_1 & \\ -\lambda_1 & 0 & \\ & & \ddots \end{pmatrix} O^\top \quad \lambda_1 \geq \ldots \geq \lambda_p$$

Orthogonal matrices

Estimate the principal components of B.

Higher Order Elo

Idea: "PCA" on B.

First-K components: best rank-K estimate of B

$$\min_{rk(B_K)=K} \|B - B_K\|_2$$

Orthogonal matrices

Estimate the principal components of B.

Higher Order Elo

$$B \approx \lambda_1 O_{n\times 2} \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} O_{n\times 2}^\top$$

Perf of player i depends on two quantities:

- Skills (ELO): f_i
- Strategy (cyclic vector) : (O_{i1}, O_{i2})

Says how much the game is cyclic

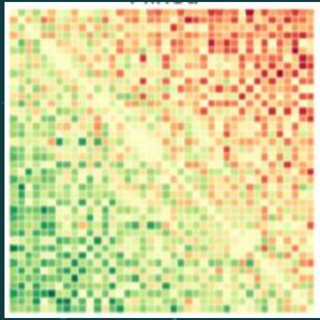
$$\hat{p}_{ij} = \sigma(A_{ij}) \approx \sigma(\underbrace{f_i - f_j}_{\text{Difference of skills}} + \boxed{\lambda_1}\underbrace{(O_{i1}O_{j2} - O_{i2}O_{j1})}_{\text{Cyclic component}})$$

Higher Order Elo

Perf of player i depends on two quantities:

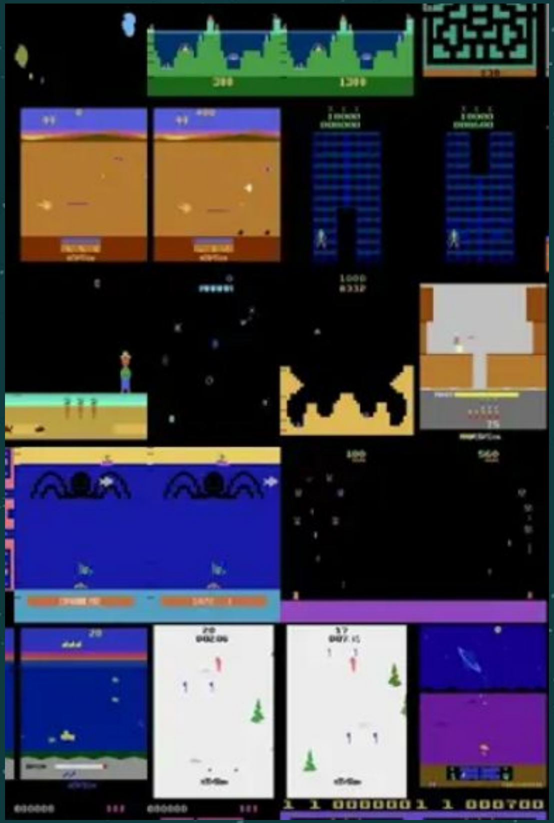
- Skills (ELO): f_i
- Strategy (cyclic vector) : (O_{i1}, O_{i2})

Estimated with an empirical payoff matrix



Caveat: We need all the pairwise matchups!!!
(not always the case... think about chess)

Agents Vs Tasks



How Tasks are Combined?

	Task 1	Task 2	Task 3	AVG	Rank
Agent 1	89	93	76	**86**	**1**
Agent 2	85	85	85	**85**	**2**
Agent 3	79	74	99	**84**	**3**

Table from NeurIPS tutorial on learning dynamics by Marta Garnelo, Wojciech Czarnecki and David Balduzzi

How Tasks are Combined?

	Task 1	Task 2	Task 3	Task 3'	AVG	Rank
Agent 1	89	93	76	77	**83.75**	**3**
Agent 2	85	85	85	84	**84.75**	**2**
Agent 3	79	74	99	98	**87.5**	**1**

Averaging is a dangerous game.

Table from NeurIPS tutorial on learning dynamics by Marta Garnelo, Wojciech Czarnecki and David Balduzzi

Desired properties

Desired properties:

1. **Invariant:** adding redundant copies of an agent or task to the data should make no difference.

2. **Continuous:** the evaluation method should be robust to small changes in the data.

3. **Interpretable:** hard to formalize, but the procedure should agree with intuition in basic cases

Maxent Nash Evaluation Method

Meta-Agent

Meta-Task

$$\min_{p} \max_{q} p^{\top} \begin{array}{c|ccc} & \text{Task 1} & \text{Task 2} & \text{Task 3} \\ \hline \text{Agent 1} & 89 & 93 & 76 \\ \text{Agent 2} & 85 & 85 & 85 \\ \text{Agent 3} & 79 & 74 & 99 \end{array} \; q$$

Theorem: There is unique (p^*,q^*) that maximize the entropy H(p^*) + H(q^*)

Best Agents

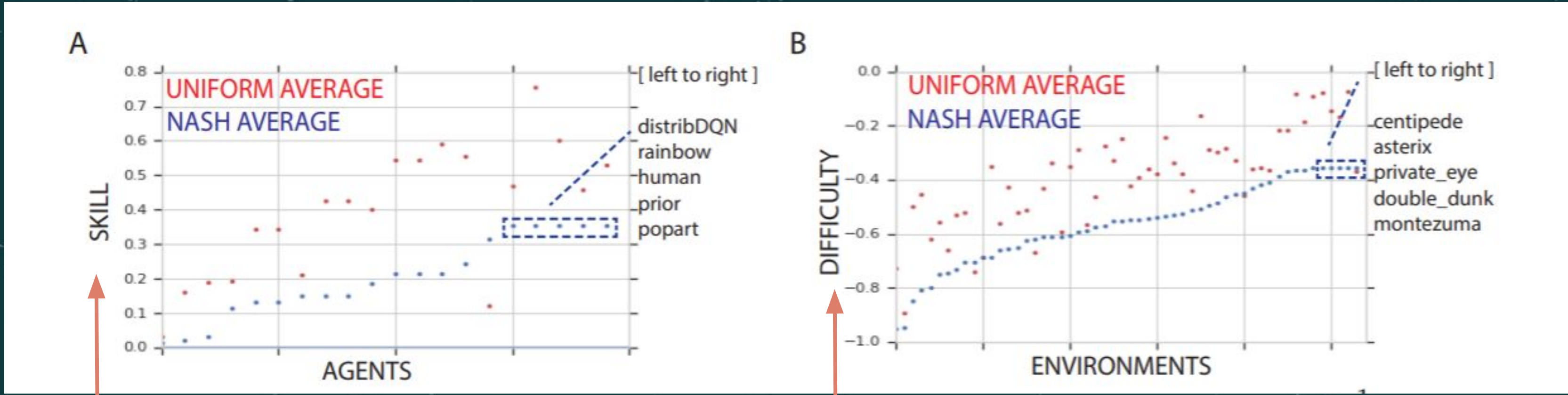
Best Agents are the ones in the MaxEnt Nash

P1. **Invariant:** *Nash averaging, with respect to the maxent NE, is invariant to redundancies in* $\mathbf{A}$.

P2. **Continuous:** *If* $\mathbf{p}^*$ *is a Nash for* $\hat{\mathbf{A}}$ *and* $\epsilon = \|\mathbf{A} - \hat{\mathbf{A}}\|_{max}$ *then* $\mathbf{p}^*$ *is an* ϵ*-Nash for* $\mathbf{A}$.

P3. **Interpretable:** **(i)** *The maxent NE on* $\mathbf{A}$ *is the uniform distribution,* $\mathbf{p}^* = \frac{1}{n}\mathbf{1}$, *iff the meta-game is* cyclic, *i.e.* $\operatorname{div}(\mathbf{A}) = \mathbf{0}$. **(ii)** *If the meta-game is* transitive, *i.e.* $\mathbf{A} = \operatorname{grad}(\mathbf{r})$, *then the maxent NE is the uniform distribution on the player(s) with highest rating(s) – there could be a tie.*

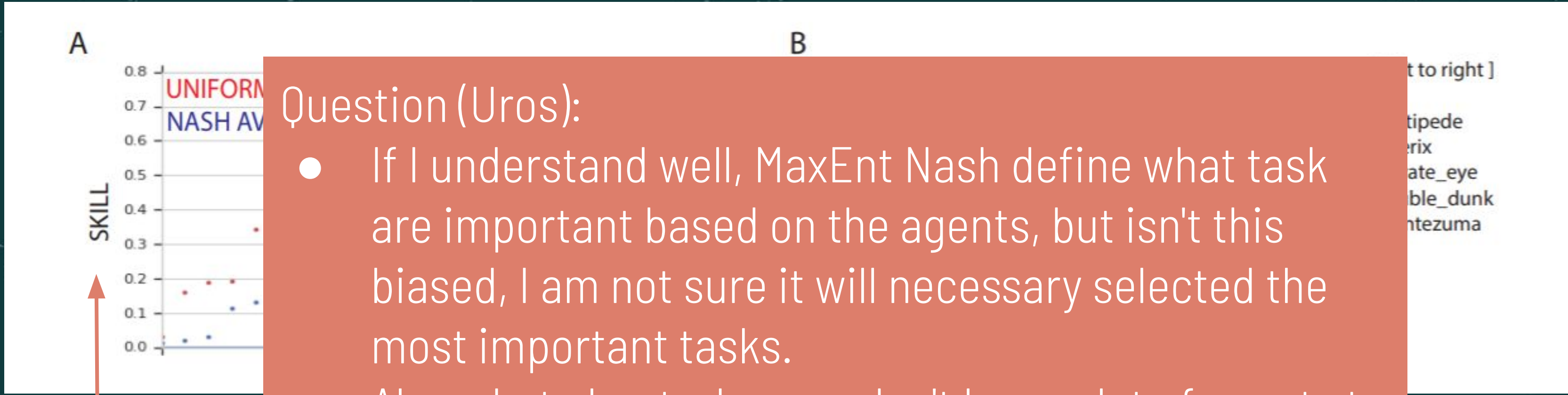
Application: Atari



Perf against the Env (uniform or Nash Avg)

Difficulty of the env against an avg player (uniform or Nash Avg)

Application: Atari



Question (Uros):

- If I understand well, MaxEnt Nash define what task are important based on the agents, but isn't this biased, I am not sure it will necessary selected the most important tasks.
- Also what about when we don't have a lot of agents to compare, is it still working well.

Perf against the Env (unifc

r Nash Avg)

Conclusion

- Two big settings for evaluations
  - Agents Vs. Agents
  - Agents Vs. Env

- For some games we may want to go beyond ELO **(estimate cyclic component of the game)**

- For Agents vs. Env we can use MaxEnt Nash to get a principled way to evaluate agents across envs.