Modeling Human Play in Games:

From Behavioral Economics to Deep Learning

Kevin Leyton-Brown

Based on work with James R. Wright and Jason Hartford

a place of mind
THE UNIVERSITY OF BRITISH COLUMBIA

If we didn't have game theory, we'd need to invent it

- A general mathematical approach for reasoning about **arbitrary strategic situations**
- Given predictions about counterfactual play, we can **design mechanisms** that optimize properties of interest

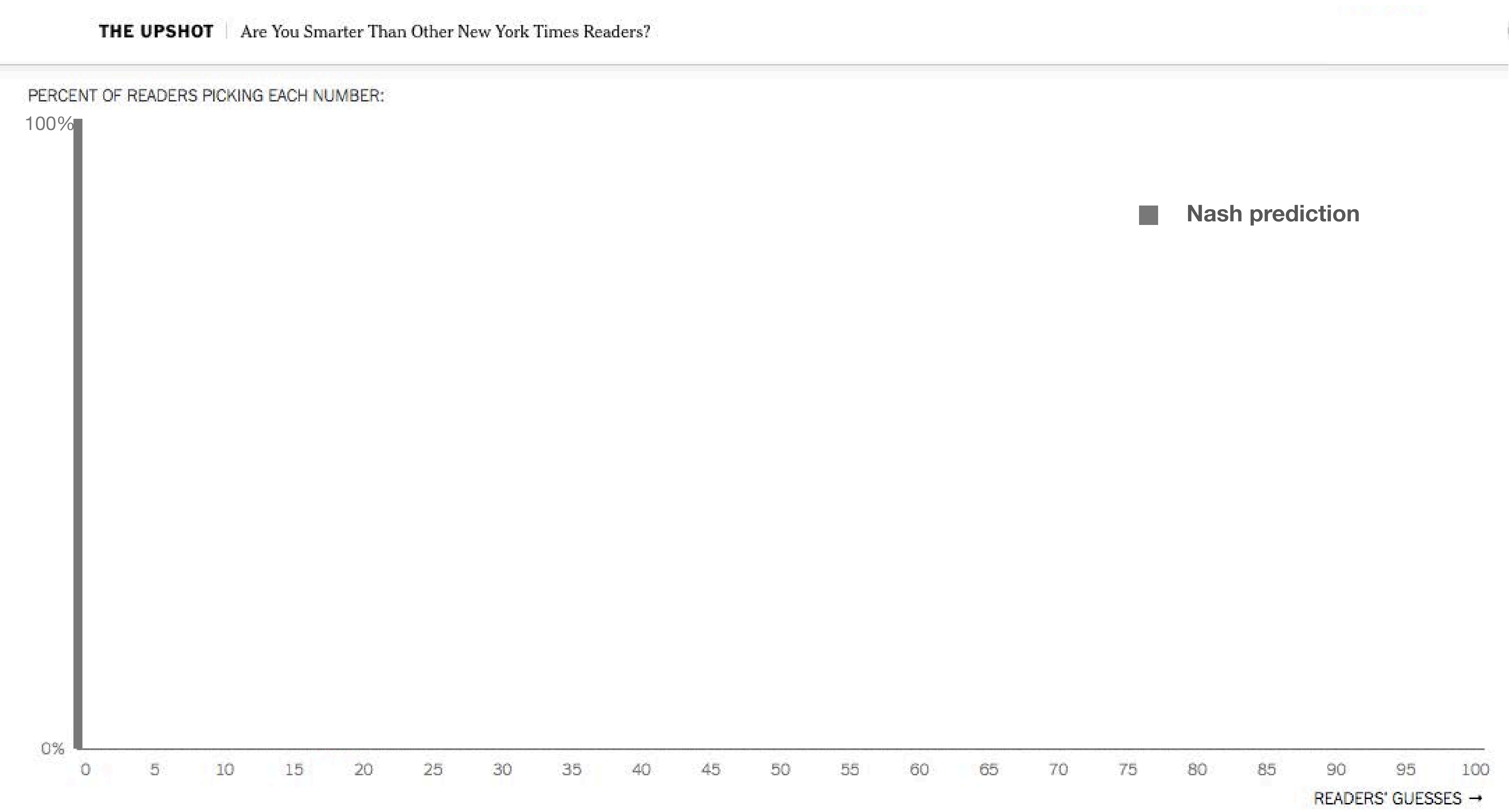
- The catch: design quality depends on **accuracy of the predictions**
- Let's consider a prediction that is among the strongest made by game theory: **unique, dominance-solvable Nash equilibrium**

Example: Beauty Contest Game

Pick to pick a number from 0 to 100

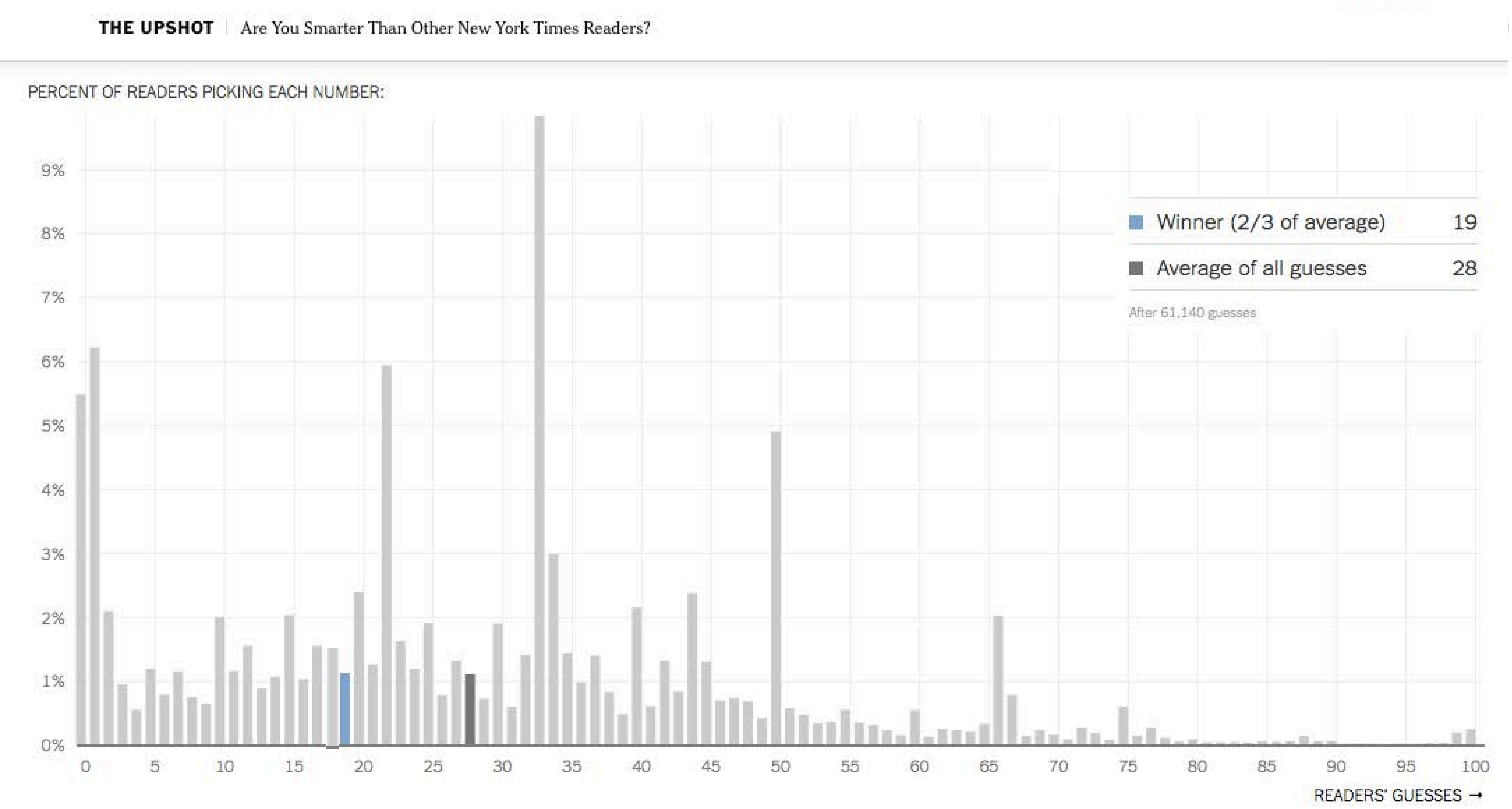
The integer closest to **two-thirds of the average of all numbers picked** wins

“Are You Smarter Than 61,140 Other New York Times Readers?”



Source: http://www.nytimes.com/interactive/2015/08/13/upshot/are-you-smarter-than-other-new-york-times-readers.

"Are You Smarter Than 61,140 Other New York Times Readers?"

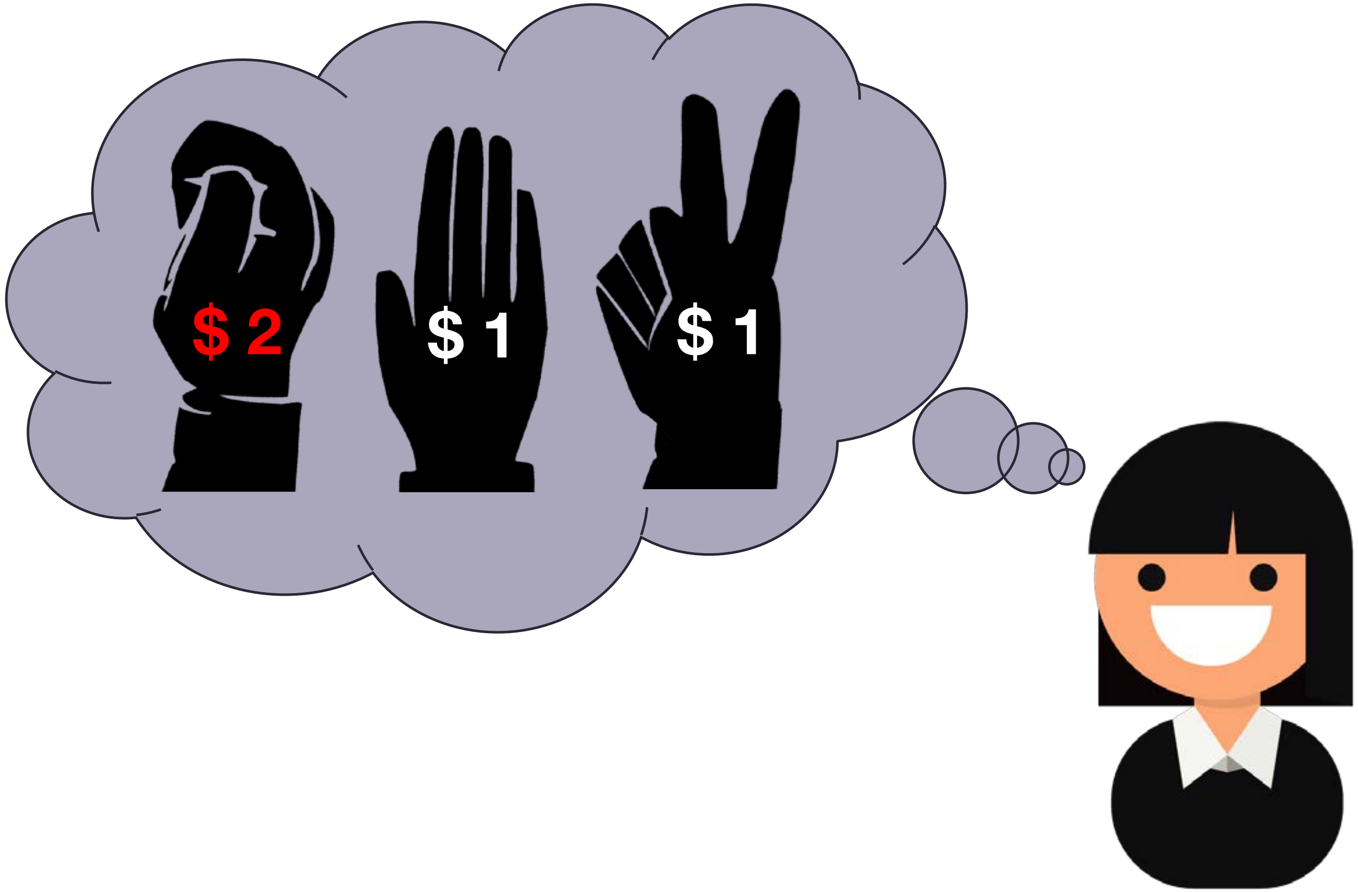


Source: http://www.nytimes.com/interactive/2015/08/13/upshot/are-you-smarter-than-other-new-york-times-readers.

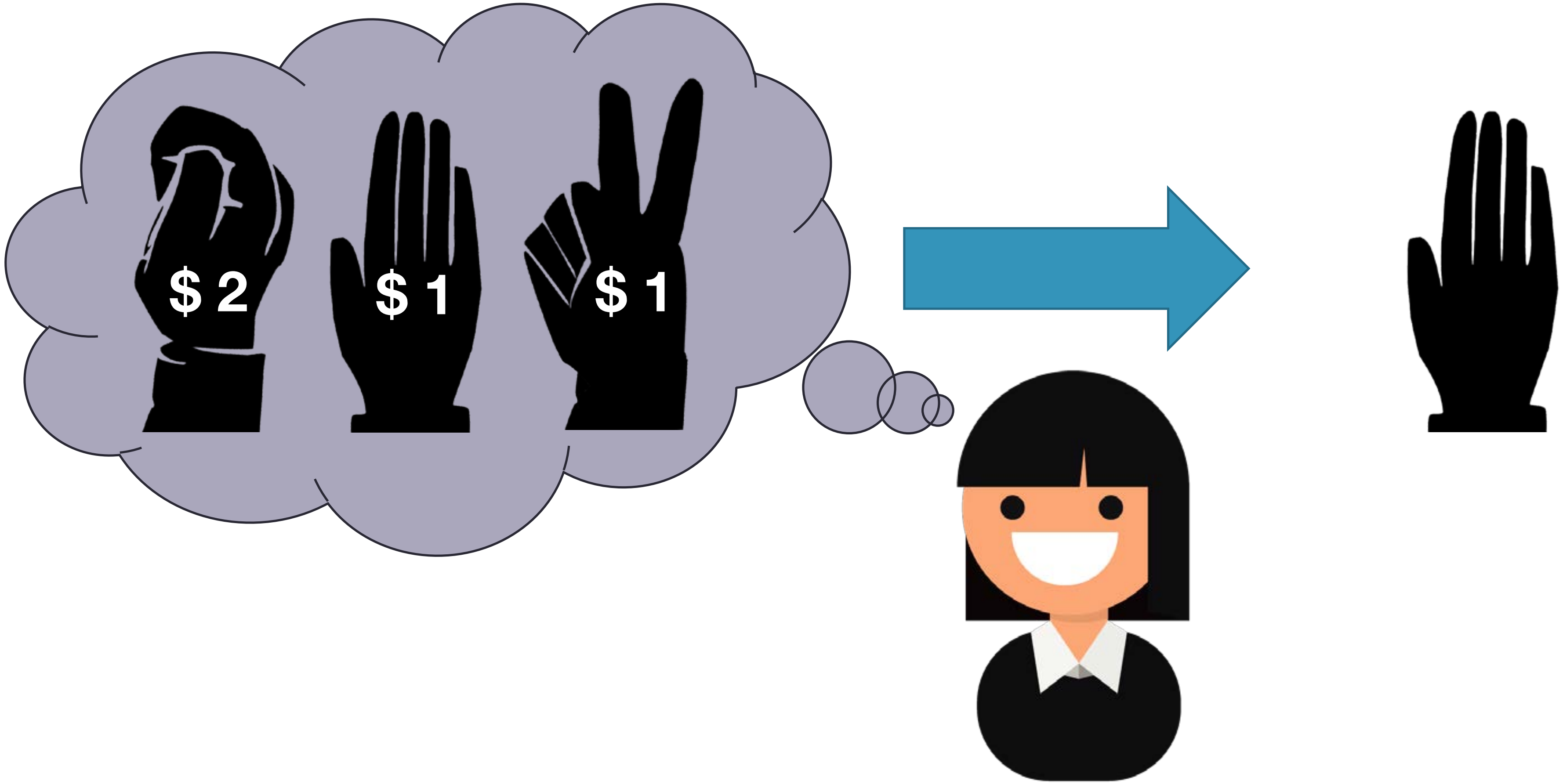
Limitations of perfect rationality

- Many of game theory's recommendations are **counterintuitive**
- Clearly the world is not populated only by **perfectly rational agents**
- To make good predictions about the play of unsophisticated humans (and hence, e.g., to design mechanisms they will use), we need a model of **human behavior**

Two player simultaneous-move games



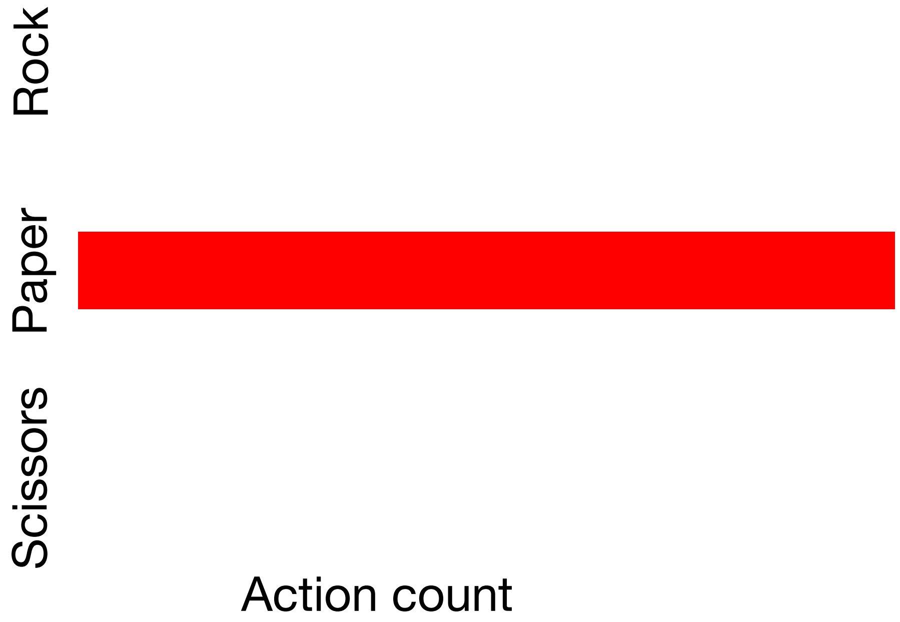
Two player simultaneous-move games



Two player simultaneous-move games

Column player's actions; Row player's actions

	Rock	Paper	Scissors
Rock	0, 0	-1, 1	2, -2
Paper	1, -1	0, 0	-1, 1
Scissors	-2, 2	1, -1	0, 0

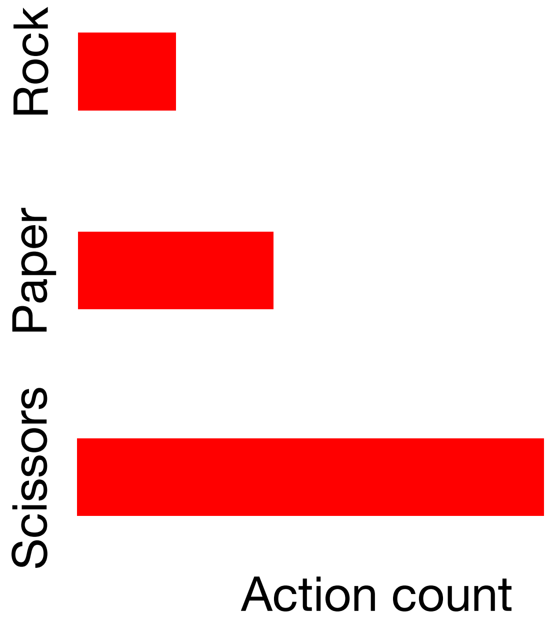


Two player simultaneous-move games

Column player's actions

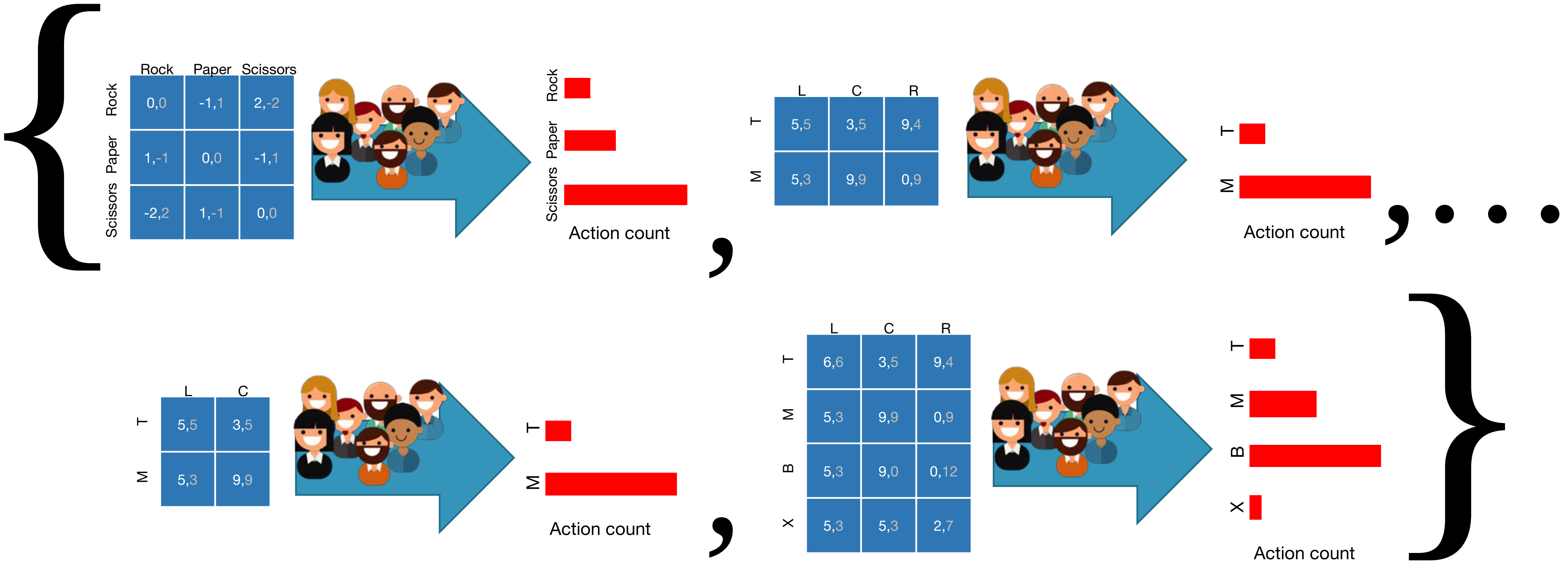
Row player's actions

	Rock	Paper	Scissors
Rock	0, 0	-1, 1	2, -2
Paper	1, -1	0, 0	-1, 1
Scissors	-2, 2	1, -1	0, 0



Learning problem

Given a dataset of **games**, each with observed **action counts**:

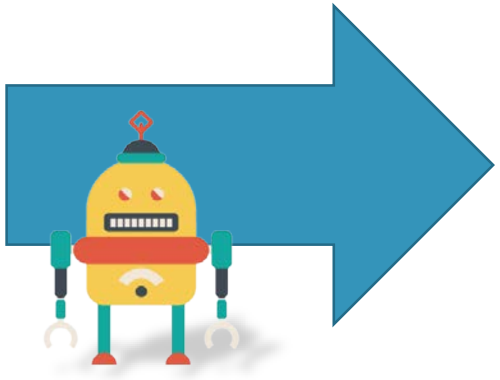


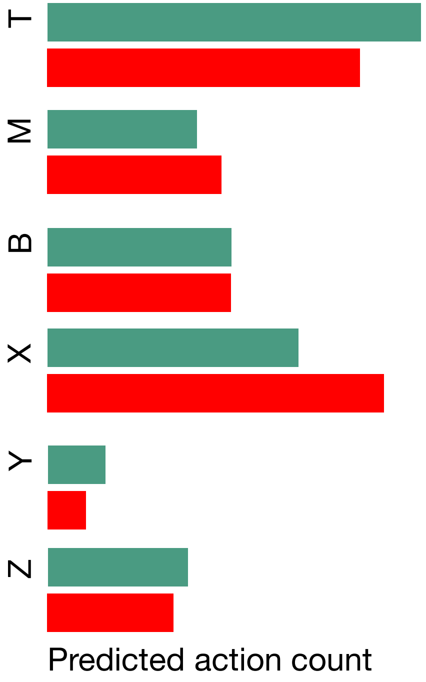
…learn a model that predicts players' **distribution** over actions

Learning problem

We will evaluate a learned model by
assessing how well it **predicts the distribution of play**
across human players from the same population
on arbitrary games not previously seen when fitting the model

	L	C	R
T	6,6	3,5	9,4
M	5,3	9,9	0,9
B	5,3	9,0	0,12
X	5,3	5,3	2,7
Y	0,-1	10,-8	0,0
Z	7,12	9,-8	0,0





Scoring models

- We randomly partition our data into **two different data sets**:

$$\mathcal{D} = \mathcal{D}_{\text{train}} \cup \mathcal{D}_{\text{test}}$$

- We choose parameter value(s) that **maximize the likelihood** of the training data:

$$\theta^* = \operatorname{argmax}_\theta \Pr(\mathcal{D}_{\text{train}} | \mathcal{M}, \theta)$$

- We score the performance of a model by likelihood of the **test data**:

$$\Pr(\mathcal{D}_{\text{test}} | \mathcal{M}, \theta^*)$$

- To reduce variance, we **repeat this process multiple times** with different random partitions, averaging the results

Data

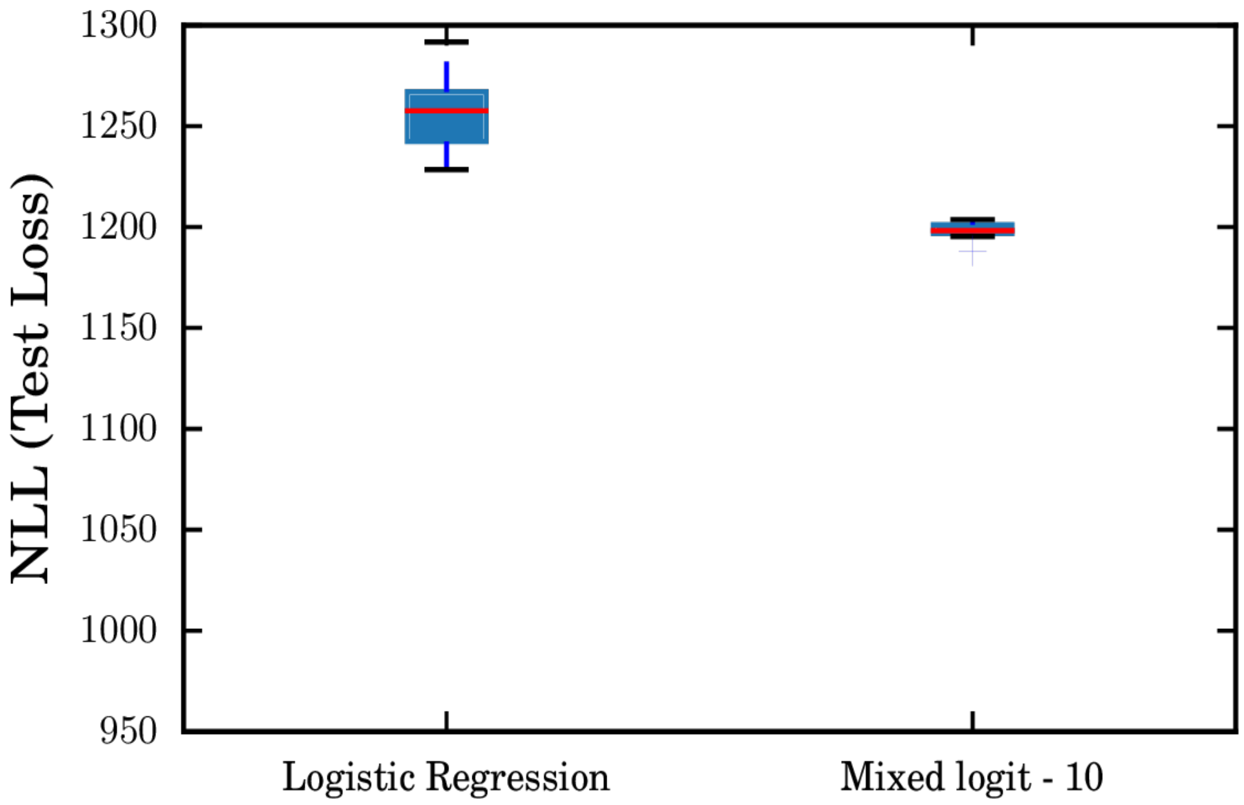
Name	Source	Games	n
SW94	[Stahl and Wilson, 1994]	10	400
SW95	[Stahl and Wilson, 1995]	12	576
CGCB98	[Costa-Gomes et al., 1998]	18	1296
GH01	[Goeree and Holt, 2001]	10	500
CVH03	[Cooper and Van Huyck, 2003]	8	2992
RPC09	[Rogers et al., 2009]	17	1210
HSW01	[Haruvy et al., 2001]	15	869
HS07	[Haruvy and Stahl, 2007]	20	2940
SH08	[Stahl and Haruvy, 2008]	18	1288
Combo9	400 samples from each	128	3600

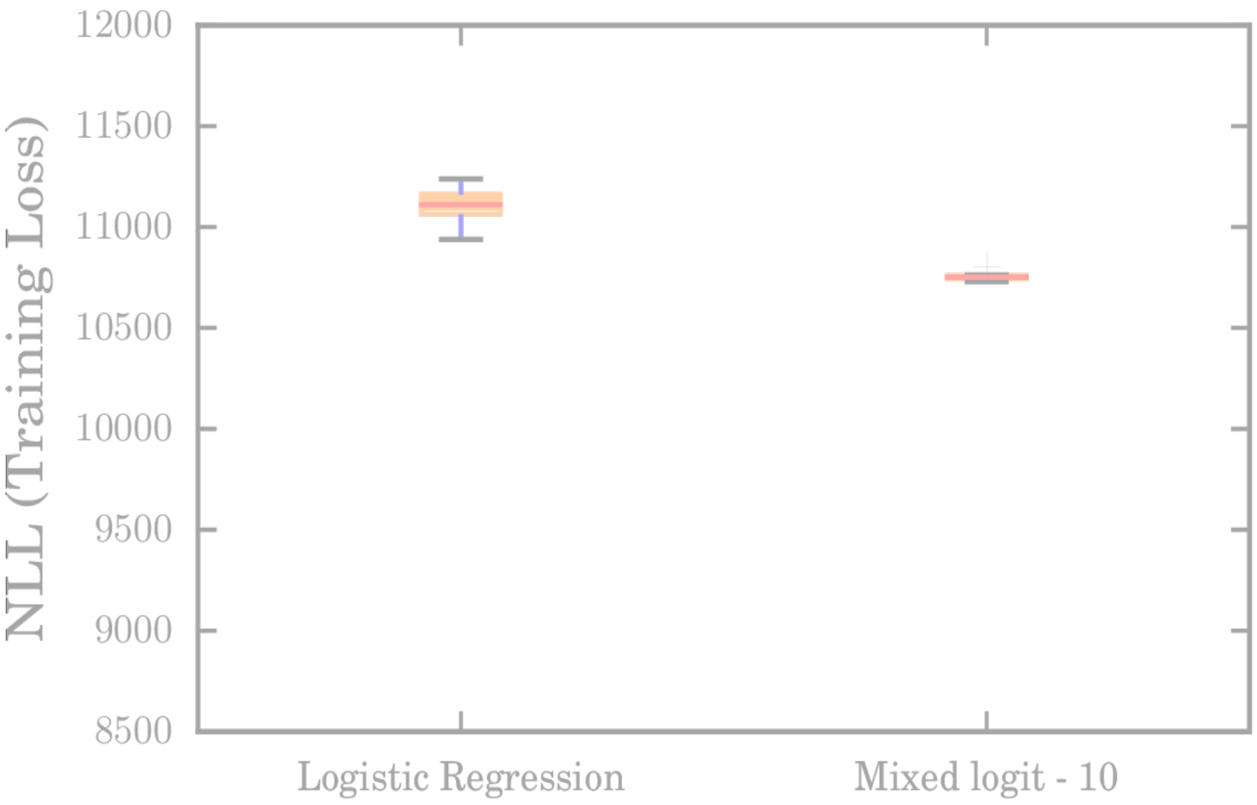
Is this a standard supervised learning problem?

- Challenges:
  - not simple classification: must return a **probability distribution**
  - not straightforward density estimation: **distribution size** varies with input
  - ...models are **mappings** from games to probability distributions
- One off-the-shelf idea: **discrete choice**
  - set of choices = row player's actions
  - features = payoffs
  - **logistic regression:** $P(a_i) = \dfrac{e^{\alpha + \sum_j \beta x_{i,j}}}{\sum_i e^{\alpha + \sum_j \beta x_{i,j}}}$
  - **mixed logit model:** $P(a_i) = \sum_{c=1}^{10} s^{(c)} \dfrac{e^{\alpha^{(c)} + \sum_j \beta^{(c)} x_{i,j}}}{\sum_i e^{\alpha^{(c)} + \sum_j \beta^{(c)} x_{i,j}}}, \qquad \sum_{c=1}^{10} s^{(c)} = 1$

    (10 latent classes)

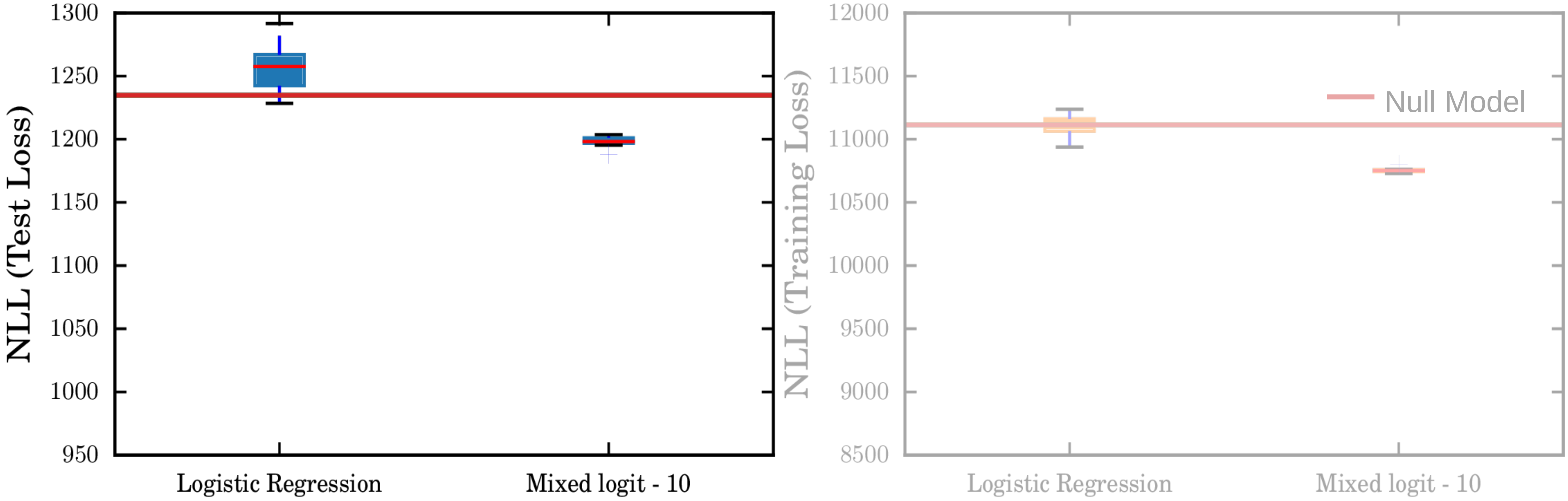
Mixed-logit performance





Is this any good?

Mixed-logit performance



Logistic regression applied to raw payoffs is **worse** than always predicting the **uniform** distribution. **Mixed logit** is not much better…

Lessons from behavioral economics

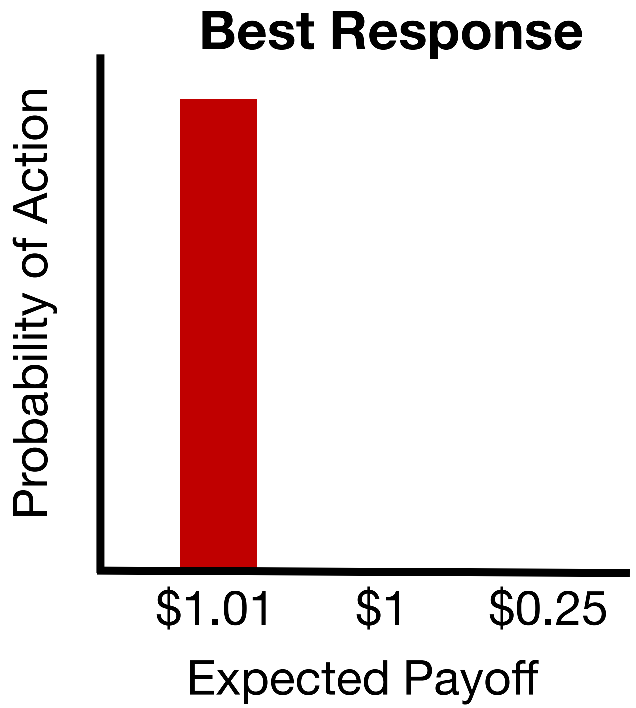
Behavioral Game Theory has proposed hand-tuned models based on psychological insights:

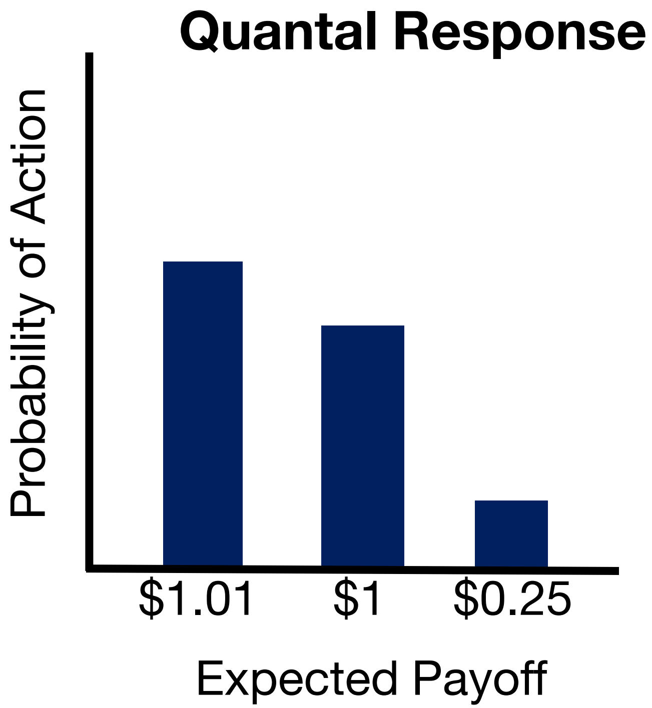
- Quantal Response Equilibrium [McKelvey & Palfrey 1995]
- Level-k [Costa-Gomes et al. 2001]
- Cognitive Hierarchy [Camerer et al. 2004]
- Noisy introspection [Goeree & Holt 2004]
- Quantal Lk, Quantal CH [Stahl & Wilson 1994; Camerer et al.]

Two key ideas underlie the best performing models [Wright, Leyton-Brown 2010; forthcoming]:

- **Quantal** utility maximization instead of utility maximization
- **Iterative strategic reasoning** instead of equilibrium

Quantal utility maximization





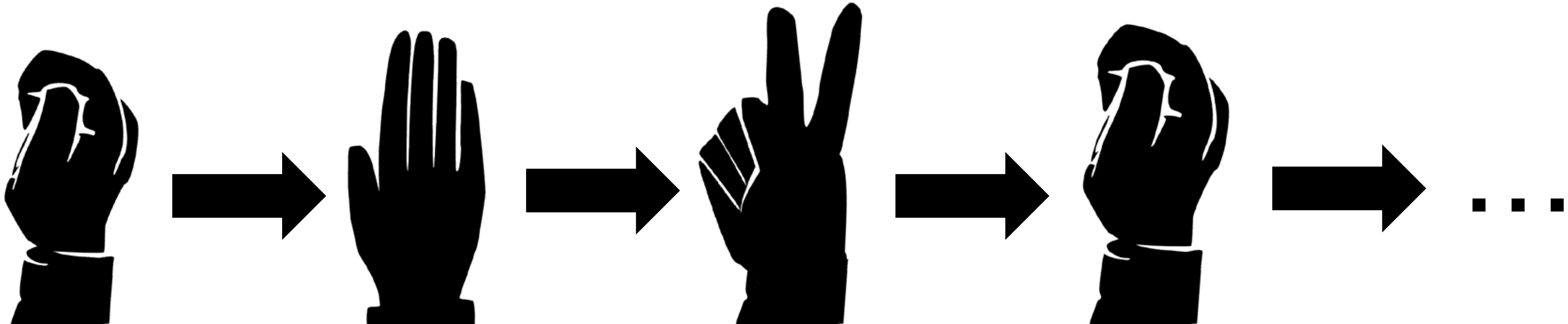
- **Best response**: Maximum utility action is always played
- **Quantal** (“softmax”) **response**: High-utility actions played often, low-utility actions played rarely

Iterative Strategic Reasoning

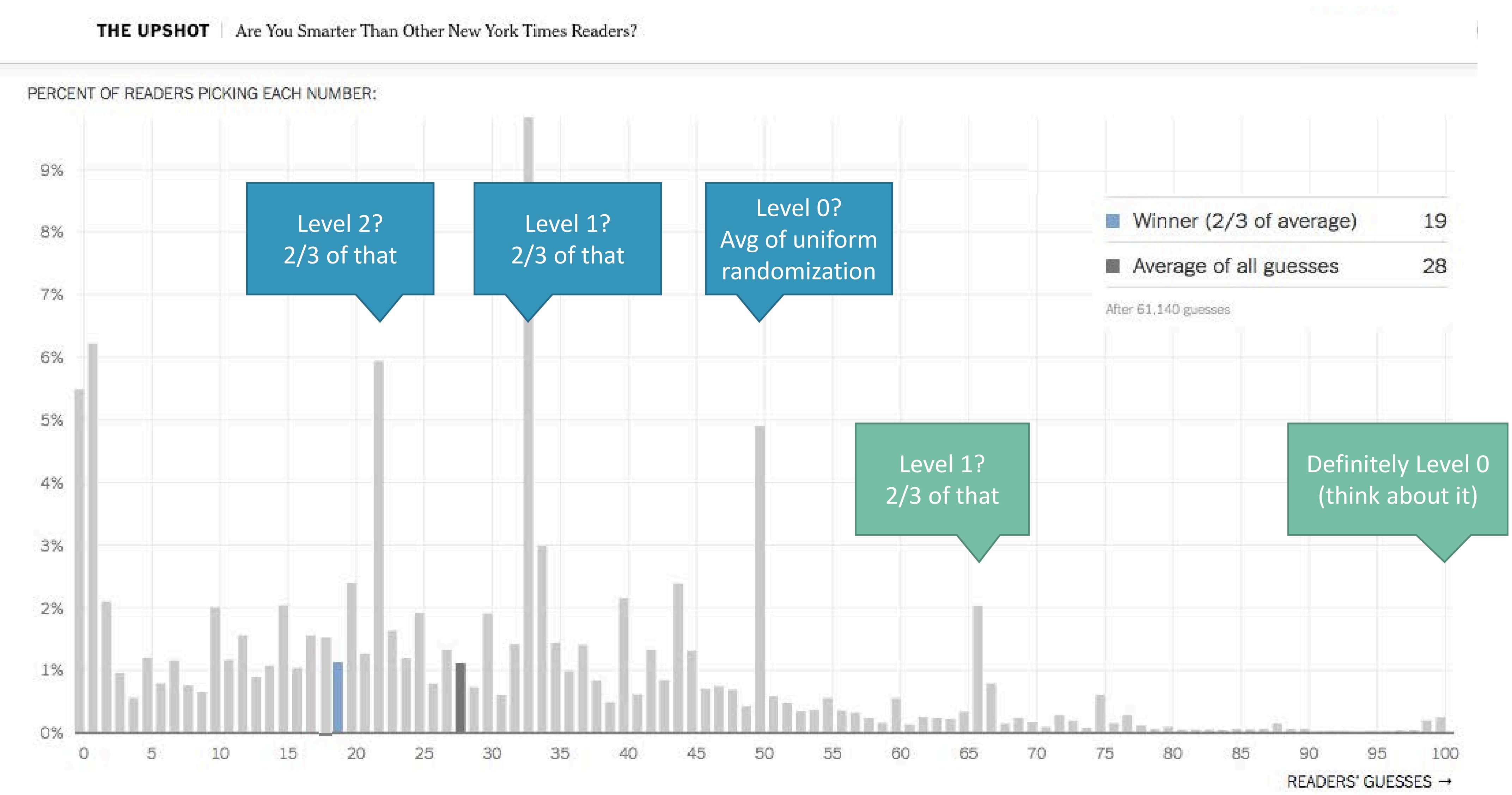
- **Level-0:** Some **nonstrategic** distribution of play (often uniform distribution)
- **Level-1:** Respond to level-0 players
- **Level-2:** Respond to level-0, or levels 0, 1

⋮

- **Level-k:** Respond to level $k-1$, or levels $\{0, \ldots, k-1\}$

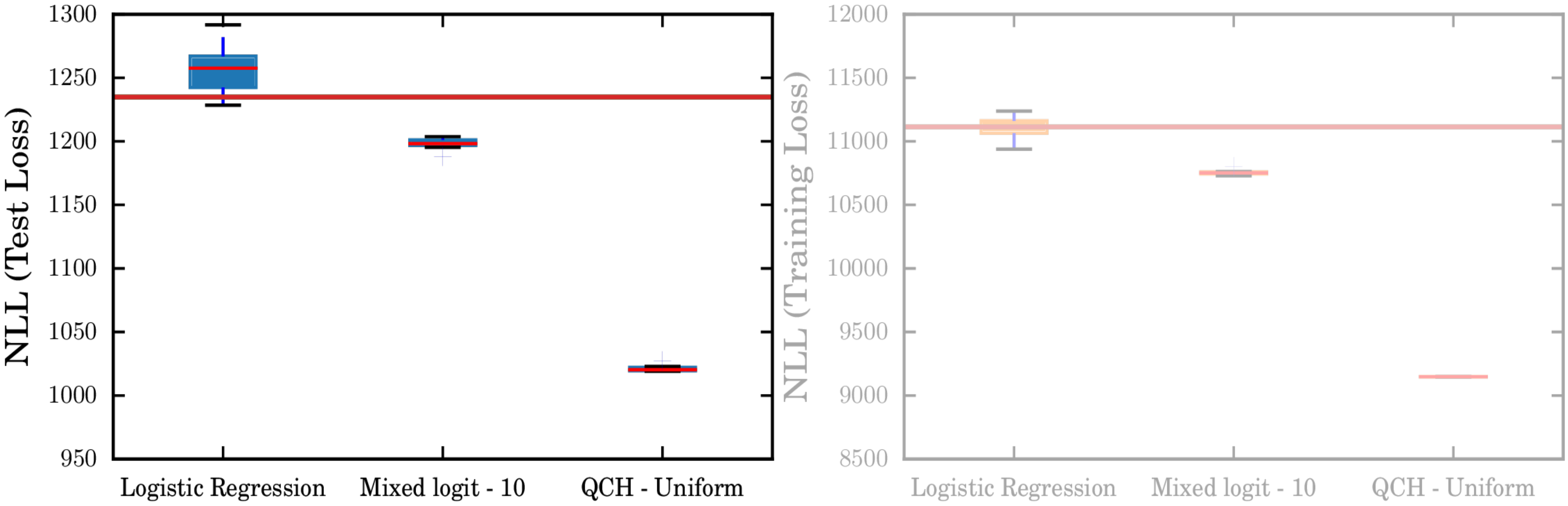


“Are You Smarter Than 61,140 Other New York Times Readers?”



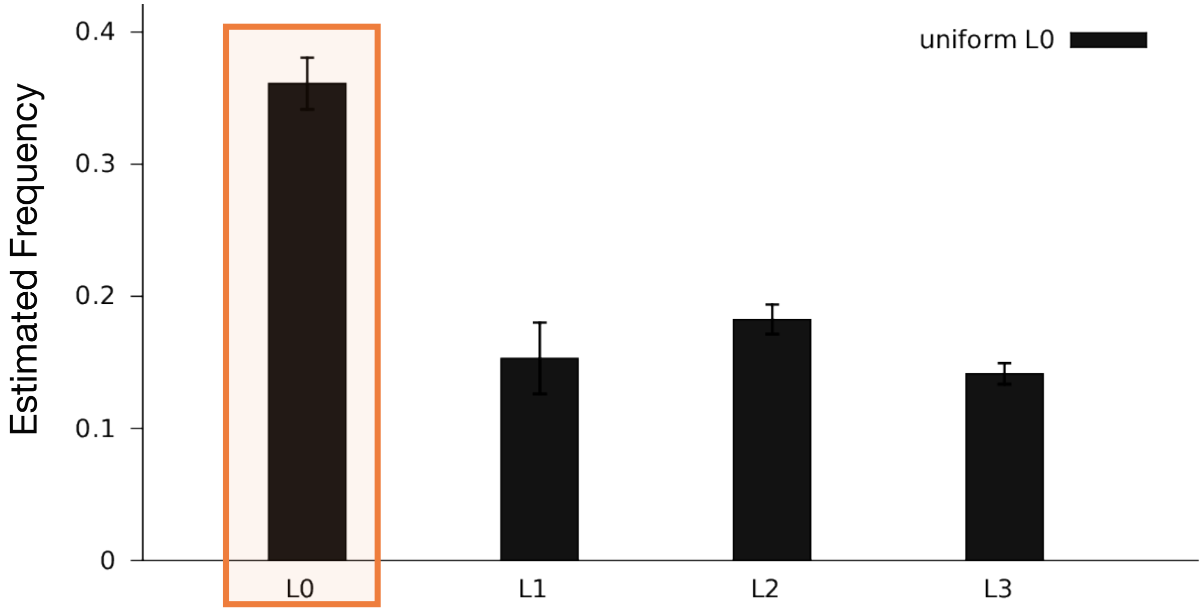
Source: http://www.nytimes.com/interactive/2015/08/13/upshot/are-you-smarter-than-other-new-york-times-readers.

Behavioral model performance



Level-0 agents

- **Bayesian analysis of parameters** shows something strange:



- The best performing models are quite certain that a large number of players **randomize uniformly**
  - Evidence of a misspecified model?

Let’s model Level-0 behavior explicitly

Five binary features:

- **Maxmin payoff (“Pessimistic”):** Is this action best in the (deterministic) worst case?
- **Maxmax payoff (“Optimistic”):** Does this action contribute to my own highest-payoff outcome?
- **Total (“Efficiency”):** Does this action contribute to the social-welfare-maximizing outcome?
- **Fairness:** Does this action contribute to the least unfair outcome?
- **Minimax regret:** Does this action minimize maximum regret?

Weighted linear model

- A feature f is **informative for game G** if f can distinguish at least one pair of actions in G
- For each action, compute a **sum of weights for features that are both informative and that "fire"**, plus a noise weight

$$\text{prediction for } a_i \propto w_0 + \sum_{f \in F} \mathbb{I}[f \text{ is informative}] \cdot \mathbb{I}[f(a_i) = 1] \cdot w_f$$

A 3×3 example; consider player 1

Every action starts out with **weight w_0**

	A	B	C	
Z	100, 20	10, 67	30, 40	w_0
Y	40, 35	50, 49	90, 70	w_0
X	41, 21	42, 22	40, 23	w_0

A 3×3 example; consider player 1

Maximize the **minimum** payoff

	A	B	C	
Z	100, 20	10, 67	30, 40	w_0
Y	40, 35	50, 49	90, 70	$w_0 + w_{\text{minmin}}$
X	41, 21	42, 22	40, 23	$w_0 + w_{\text{minmin}}$

A 3 × 3 example; consider player 1

Maximize the **best-case** payoff

	A	B	C	
Z	100, 20	10, 67	30, 40	$w_0 + w_{\text{maxmax}}$
Y	40, 35	50, 49	90, 70	$w_0 + w_{\text{minmin}}$
X	41, 21	42, 22	40, 23	$w_0 + w_{\text{minmin}}$

A 3 × 3 example; consider player 1

Maximize the **sum of** both players' payoffs

	A	B	C	
Z	100, 20	10, 67	30, 40	$w_0 + w_{\text{maxmax}}$
Y	40, 35	50, 49	90, 70	$w_0 + w_{\text{minmin}} + w_{\text{total}}$
X	41, 21	42, 22	40, 23	$w_0 + w_{\text{minmin}}$

A 3 × 3 example; consider player 1

Fairest outcome

	A	B	C	
Z	100, 20	10, 67	30, 40	$w_0 + w_{\text{maxmax}}$
Y	40, 35	50, 49	90, 70	$w_0 + w_{\text{minmin}} + w_{\text{total}} + w_{\text{fairness}}$
X	41, 21	42, 22	40, 23	$w_0 + w_{\text{minmin}}$

A 3×3 example; consider player 1

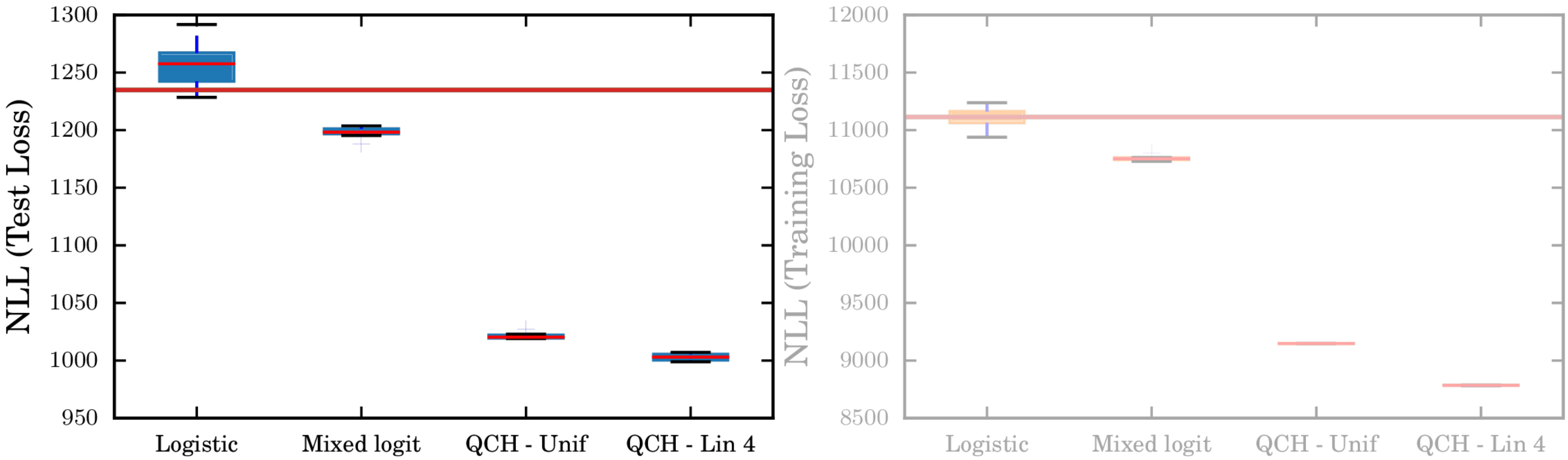
Minimax Regret isn't informative

(it's 60 for all actions; e.g., when Player 1 plays X, if Player 2 plays C, his regret is 60)

	A	B	C	
Z	100, 20	10, 67	30, 40	$w_0 + w_{\mathrm{maxmax}}$
Y	40, 35	50, 49	90, 70	$w_0 + w_{\mathrm{minmin}} + w_{\mathrm{total}} + w_{\mathrm{fairness}}$
X	41, 21	42, 22	40, 23	$w_0 + w_{\mathrm{minmin}}$

…and **normalize** to get the distribution over actions

Effect of modeling nonstrategic play

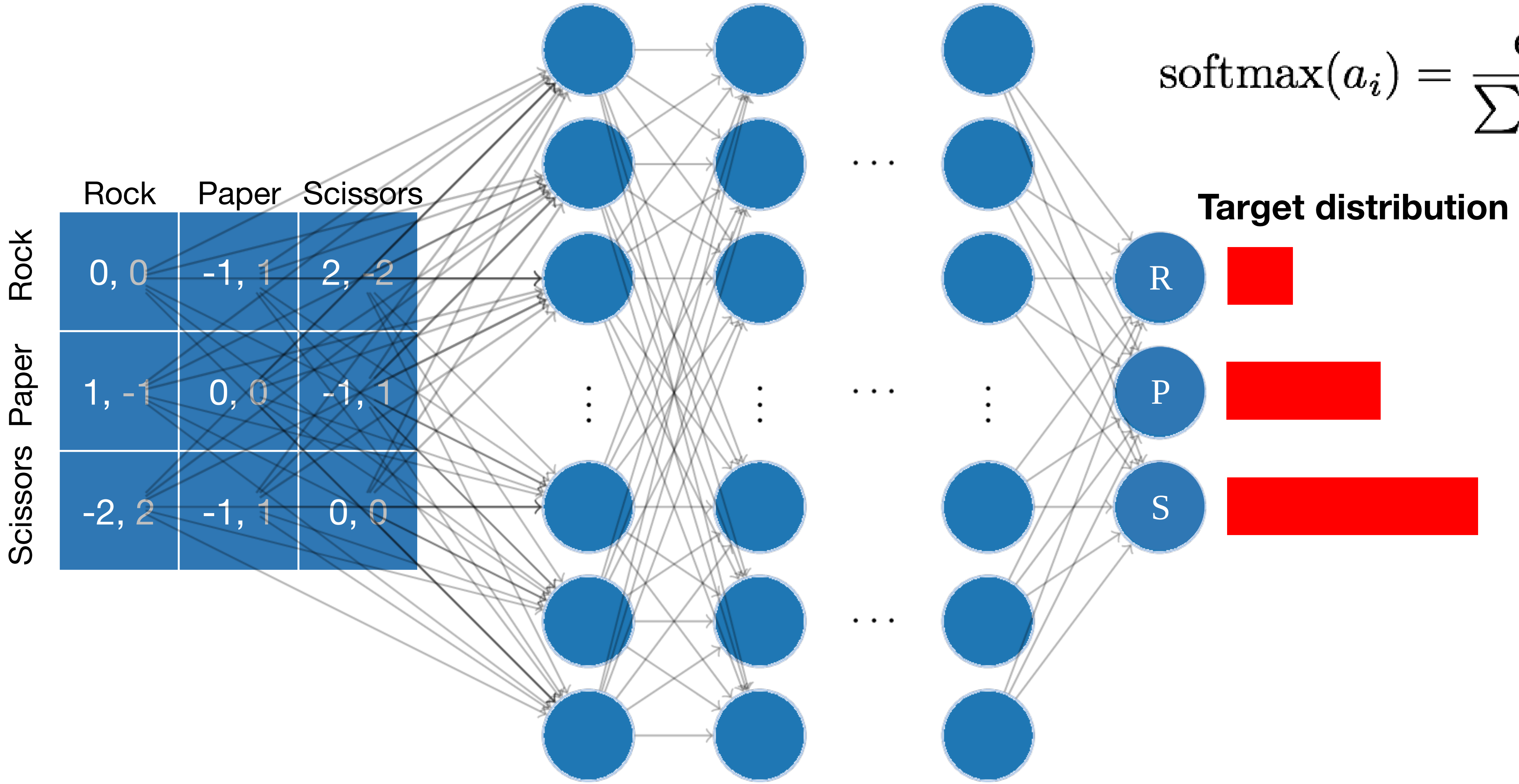


Beyond Feature Engineering

- A better model of **nonstrategic play** made a big difference
- But, it's hard to know if we've got the model right:
  - have we included the right **features**?
  - do our models have the right **functional form**?

- Deep learning has demonstrated the possibility of stunning predictive performance via **learning features**
- Could we **automatically search** for behavioral models?

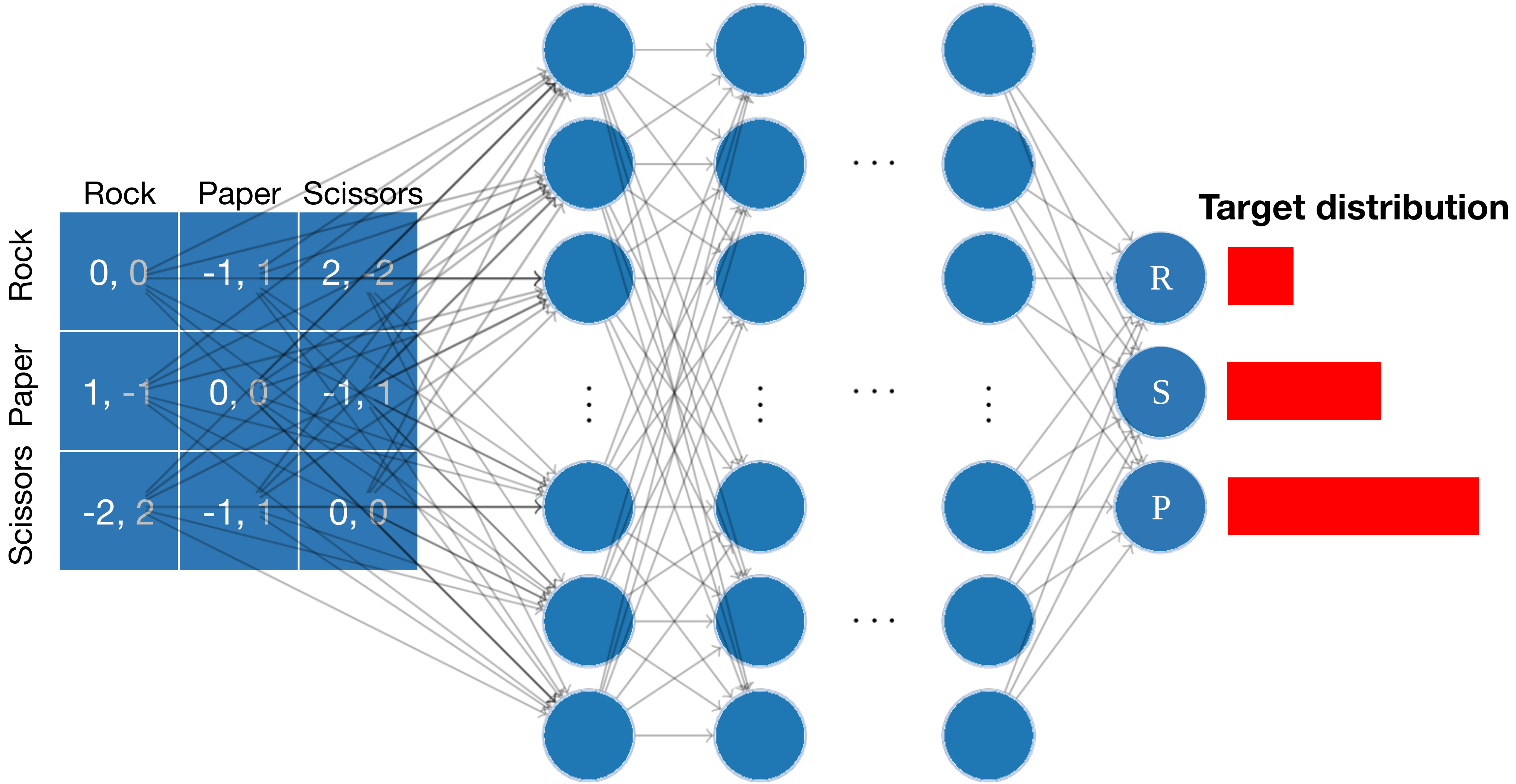
Direct application of a feed-forward net



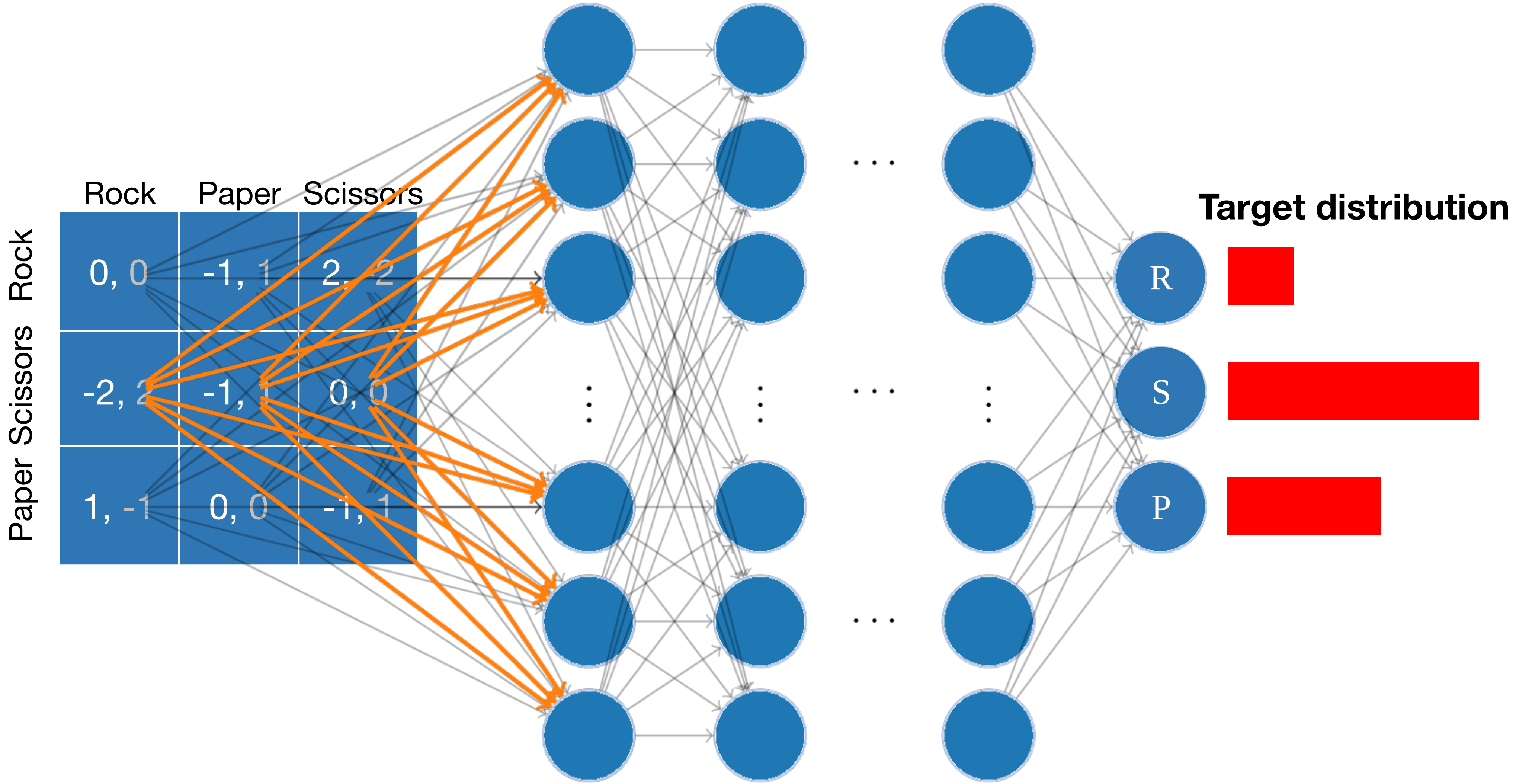
$$\mathrm{softmax}(a_i) = \frac{\exp(a_i)}{\sum_j \exp(a_j)}$$

$$\mathrm{relu}(x) = \max(0, x)$$

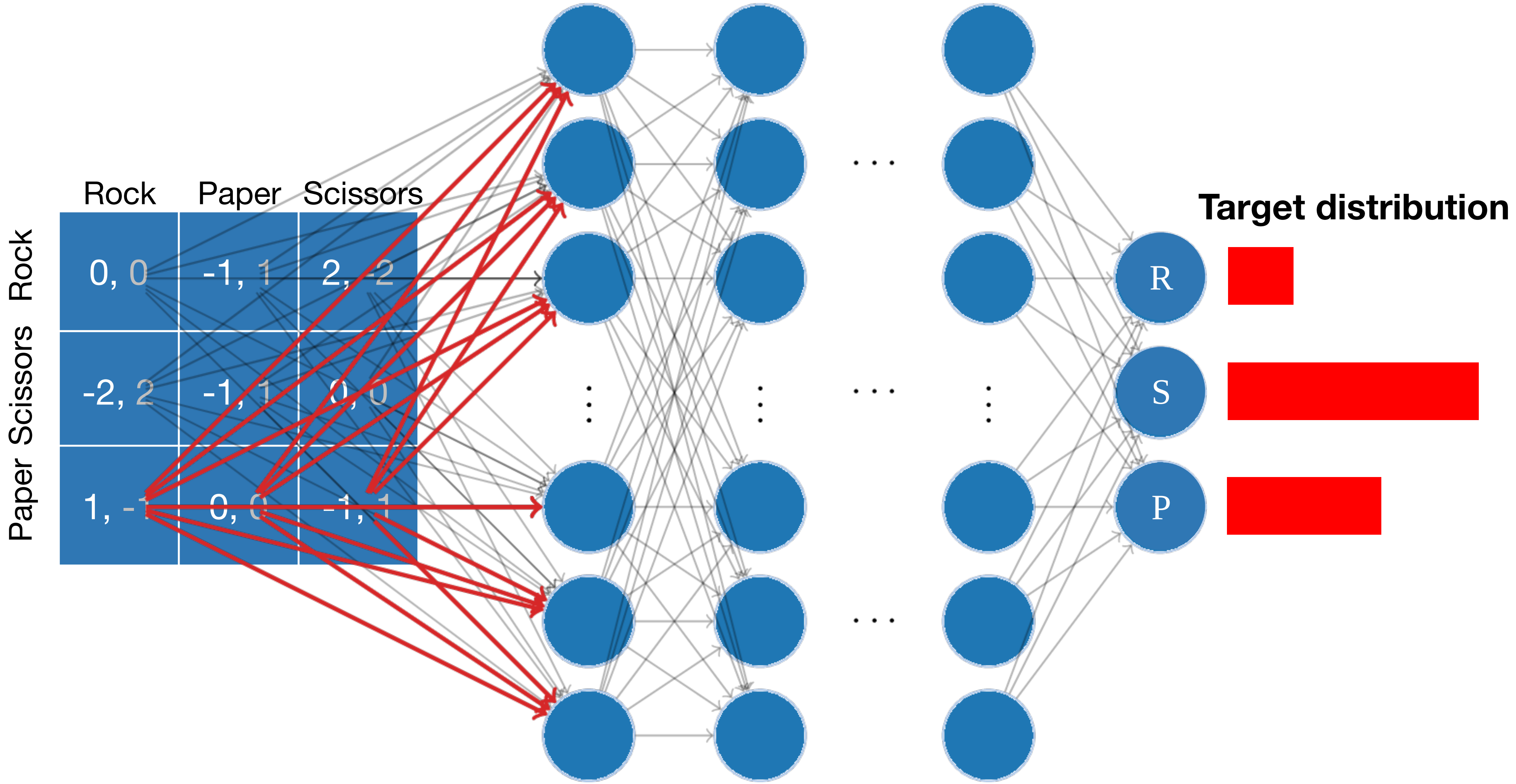
Direct application of a feed-forward net



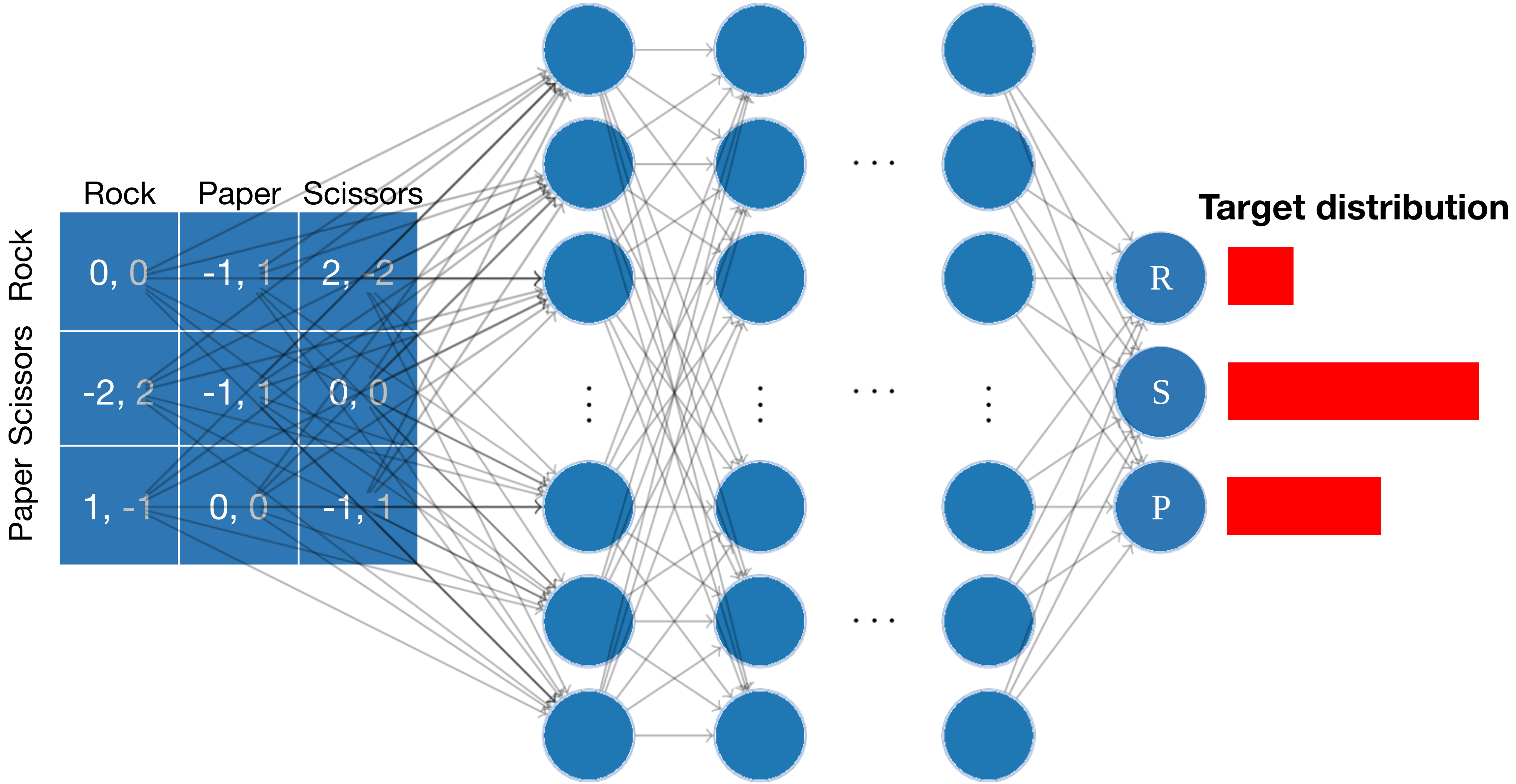
Direct application of a feed-forward net



Direct application of a feed-forward net



Direct application of a feed-forward net

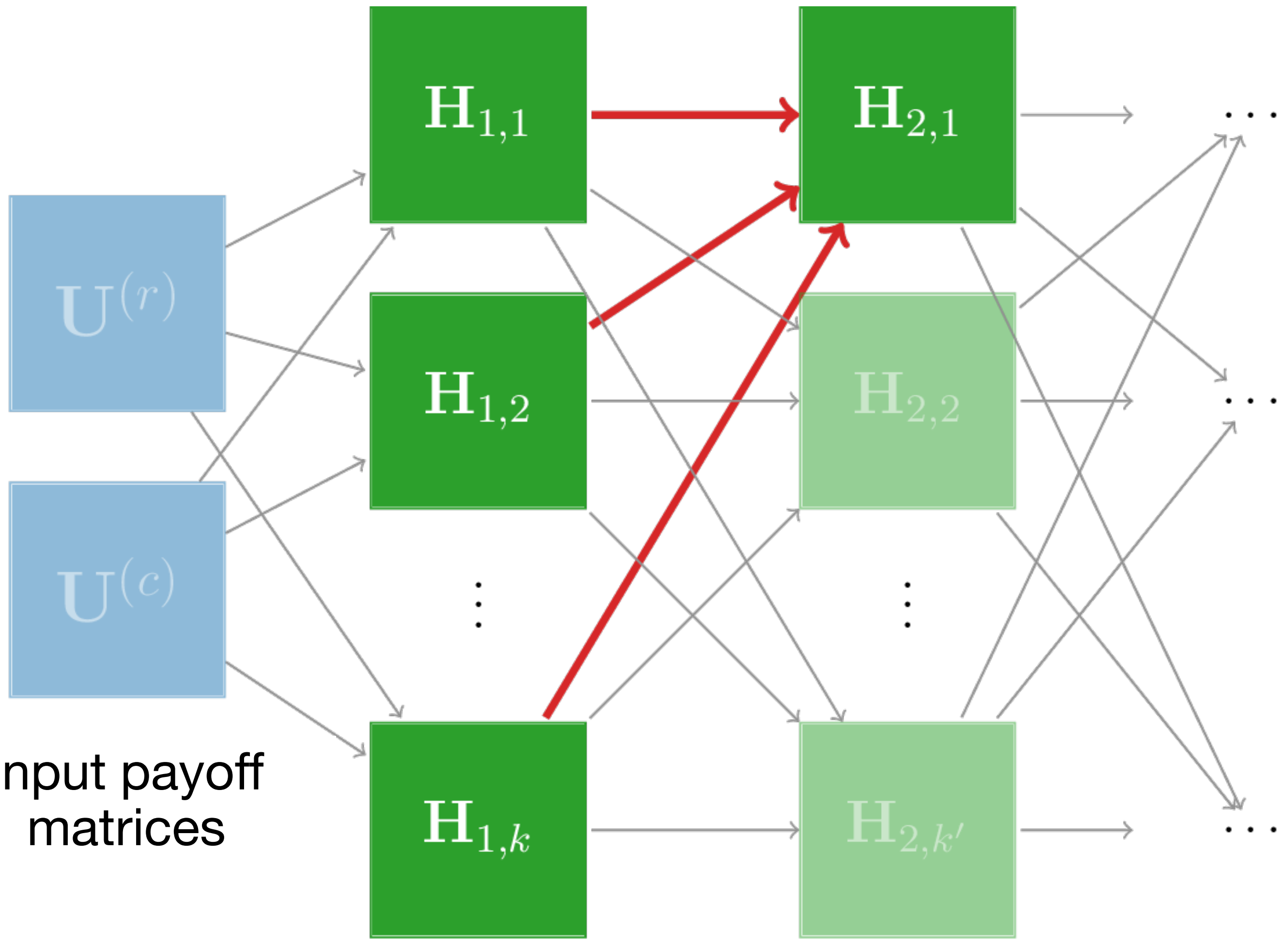


Game-Theoretic Wish List

1. Invariance to game **permutations**
2. Output is be a probability distribution of **size** = player 1's action space
3. Models allow **rich comparisons** between actions and outcomes
4. Models **iterative strategic reasoning**

Invariance-preserving hidden units

GT Wish List
1. **Permutation**
2. Size
3. Comparison
4. Iterative Reasoning

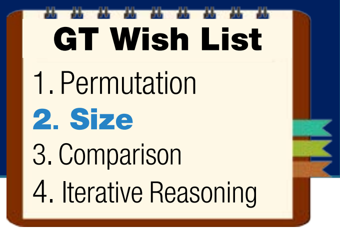


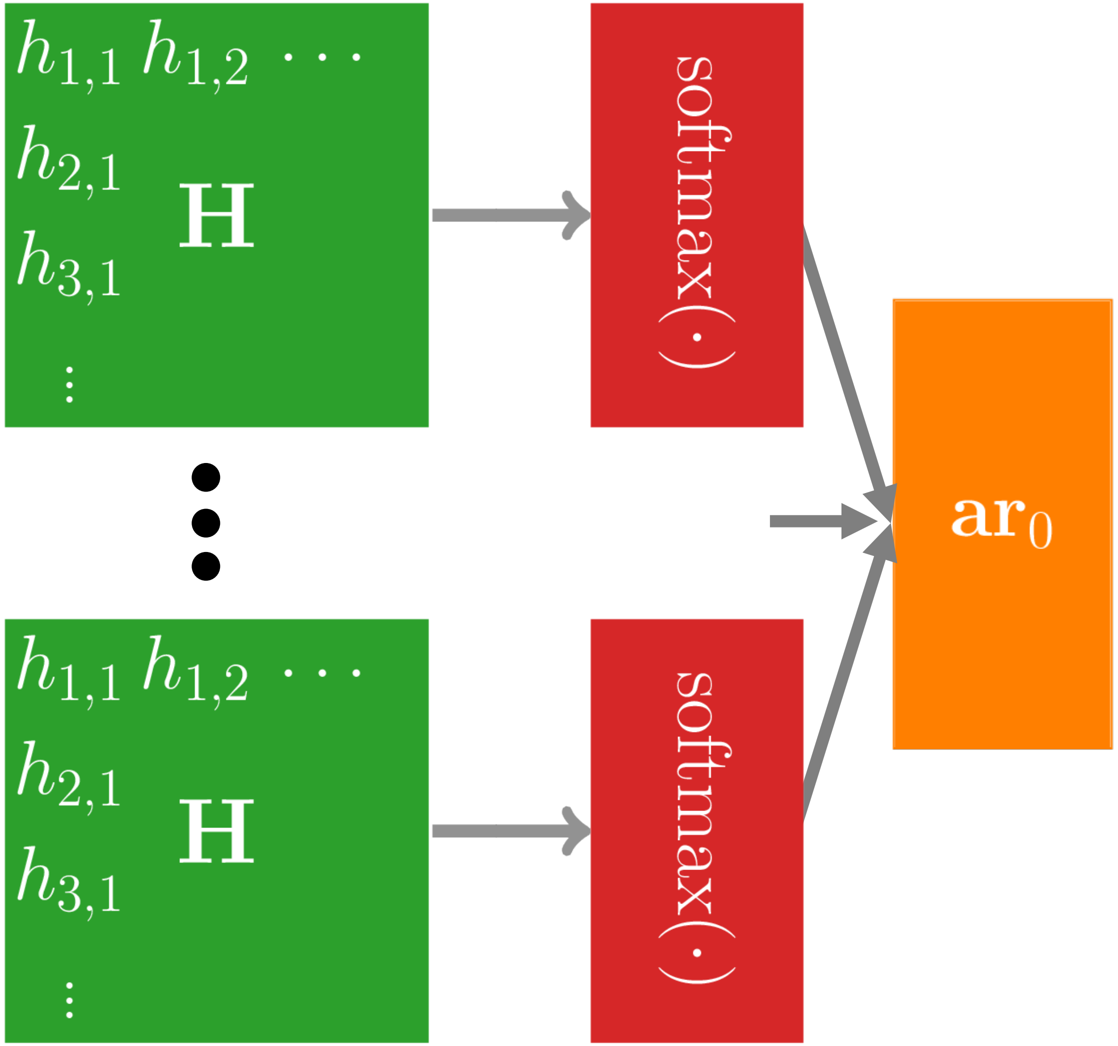
Input payoff matrices

$$H_{2,1} = \phi\left(\Sigma_k\, w_{2,k}\, H_{1,k}\right)$$

- Let each node be a **matrix** computing a **weighted sum** of the **matrices** in the preceding layer
- Apply an element-wise **activation function**, ∅
  - again, we use $\emptyset(x) = \mathrm{relu}(x)$

Predicting a distribution over actions





We want a distribution over player 1's actions with **size = P1's action space**

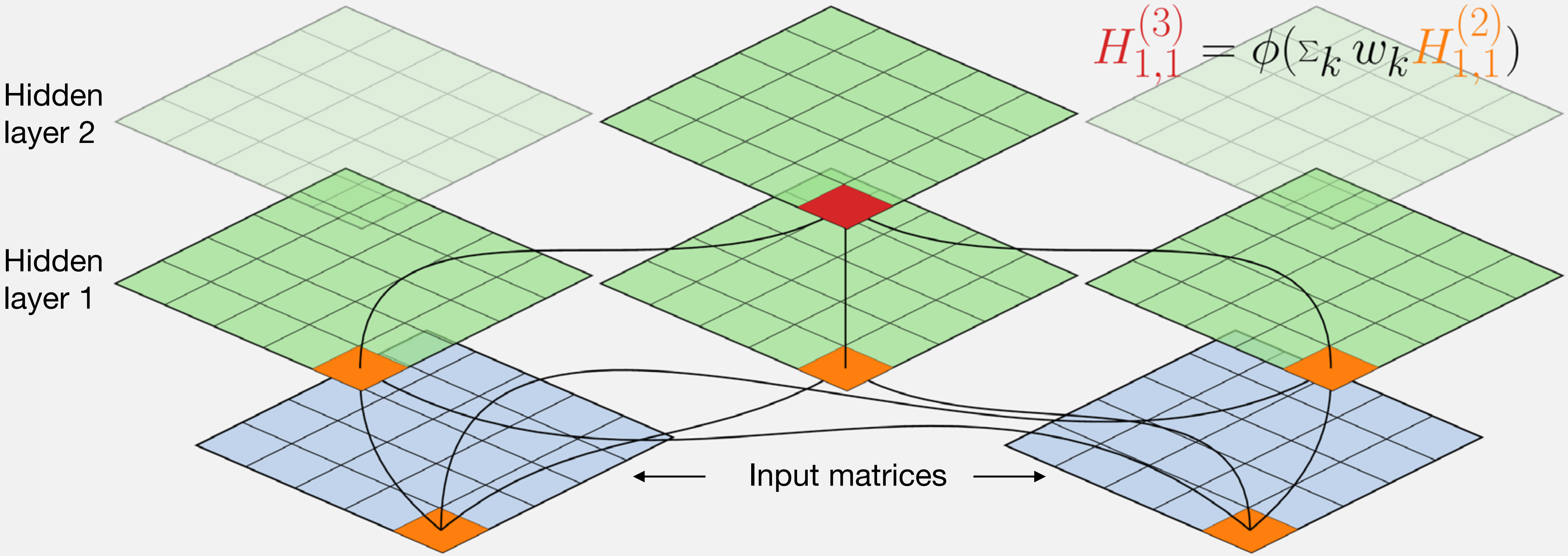
- Sum **uniformly** over the **column player's** actions & apply a **softmax** function to the resulting vectors:

$$\mathrm{softmax}(a_i) = \frac{\exp(a_i)}{\sum_j \exp(a_j)}$$

- Take **weighted sum** to construct our **output**

Comparing outcomes

GT Wish List
1. Permutation
2. Size
3. **Comparison**
4. Iterative Reasoning

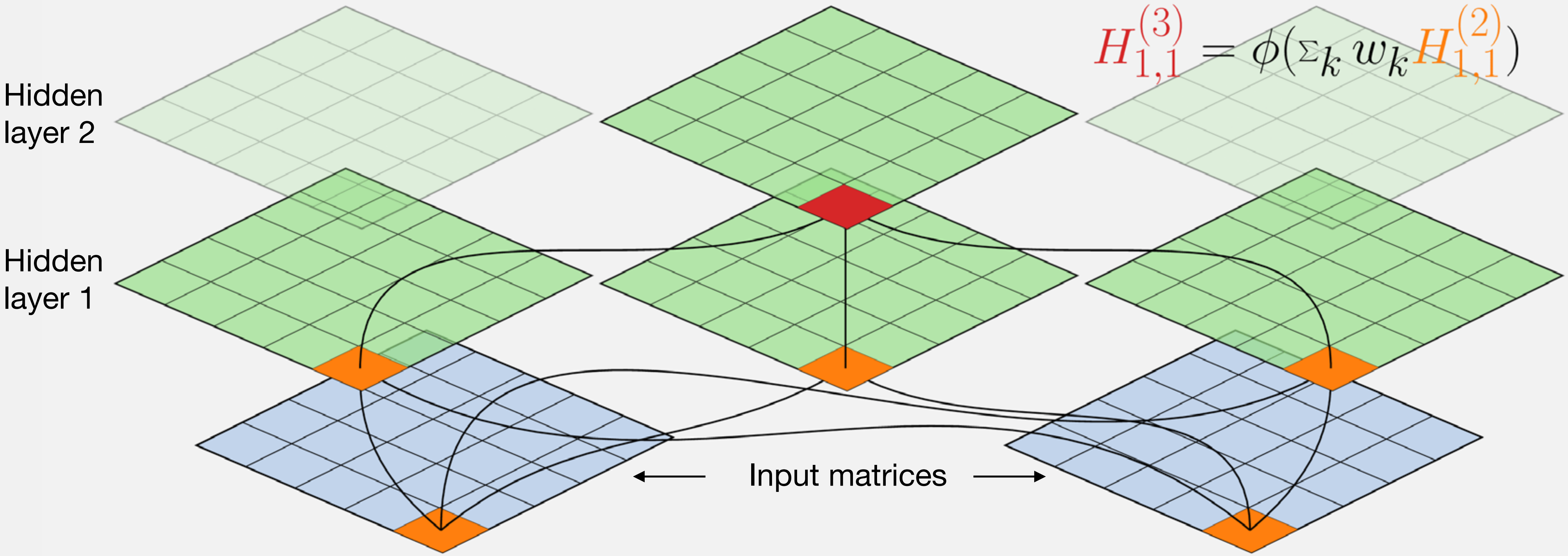


Each element of a given matrix depends only on the **corresponding elements** from input matrices

Can't learn functions that **relate elements** between cells

Comparing outcomes

GT Wish List
1. Permutation
2. Size
3. **Comparison**
4. Iterative Reasoning



Each element of a given matrix depends only on the **corresponding elements** from input matrices

Can't learn functions that **relate elements** between cells

Comparing outcomes

GT Wish List
1. Permutation
2. Size
3. **Comparison**
4. Iterative Reasoning

	Rock	Paper	Scissors
Rock	0, 0	-1, 1	2, -2
Paper	1, -1	0, 0	-1, 1
Scissors	-2, 2	-1, 1	0, 0
Nuclear	-5, -5	-5, -5	-5, -5

Probability of action

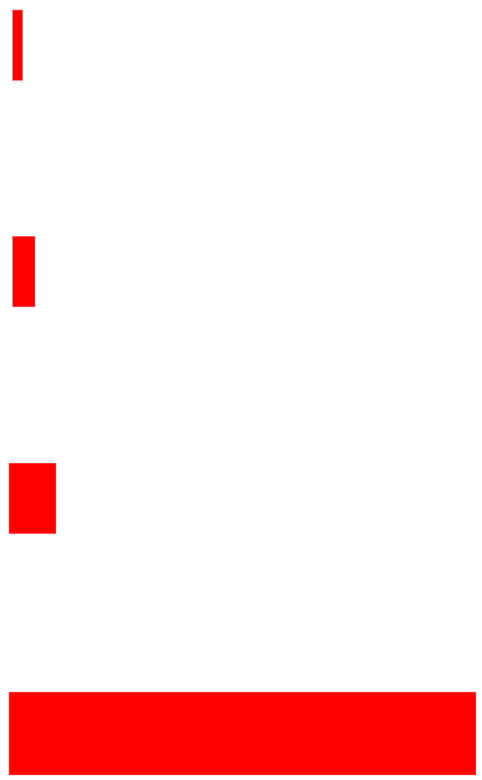
Comparing outcomes

GT Wish List
1. Permutation
2. Size
3. **Comparison**
4. Iterative Reasoning

(RPS × 1000) - 10

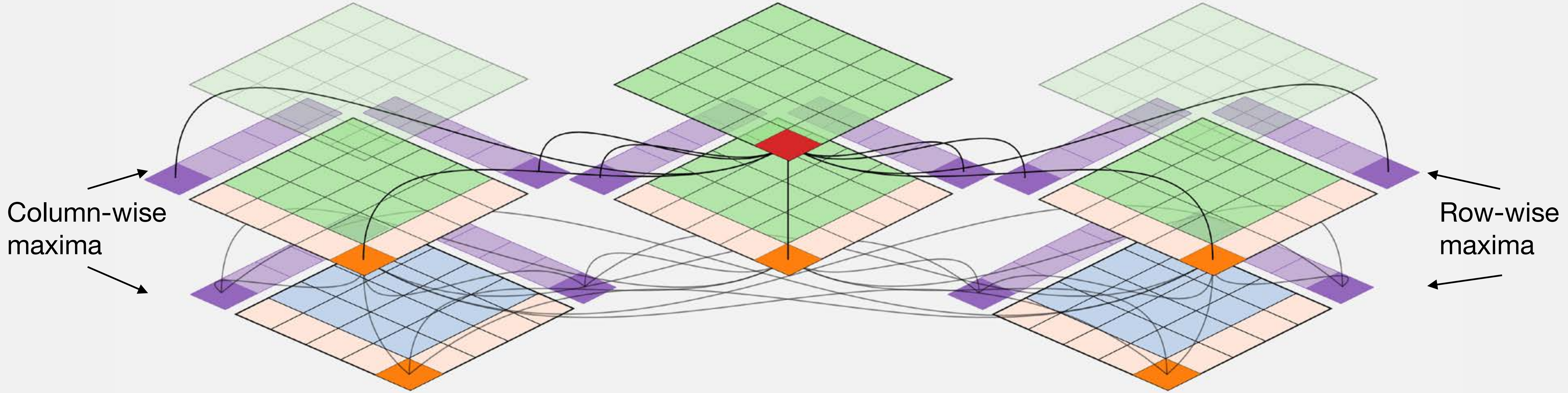
	Rock	Paper	Scissors
Rock	-10, -10	-1010, 990	1990, -2010
Paper	990, -1010	-10, -10	-1010, 990
Scissors	-2010, 1990	-1010, 990	-10, -10
Nuclear	-5, -5	-5, -5	-5, -5

Probability of action



Action pooling units

GT Wish List
1. Permutation
2. Size
3. **Comparison**
4. Iterative Reasoning



Pooling units output **aggregates** of the payoffs associated with particular actions by computing **row** and **column-wise maxima** for each hidden unit and providing them as inputs to subsequent layers

Action response layers

GT Wish List
1. Permutation
2. Size
3. Comparison
4. Iterative Reasoning

Dot product

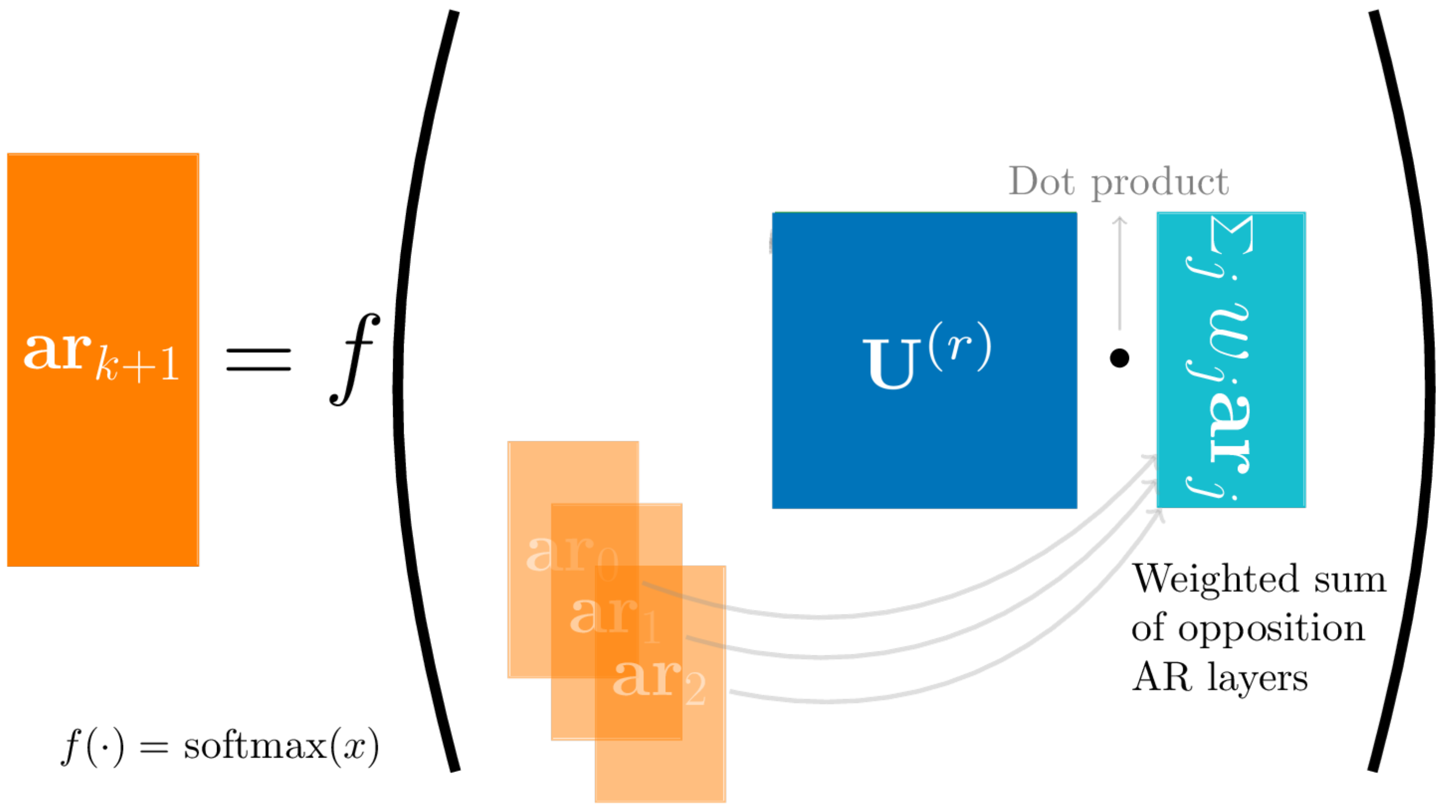
$$\mathbf{ar}_{k+1} = f\left(\mathbf{U}^{(r)} \bullet \mathbf{ar}_k \right)$$

$f(\cdot) = \text{softmax}(x)$

AR layers…

- Compute a **weighted** sum over P2 actions
- Weights are the model's predicted **distribution** over **P2's** actions
- Applied **recursively** to model multiple iterative reasoning steps

Action response layers



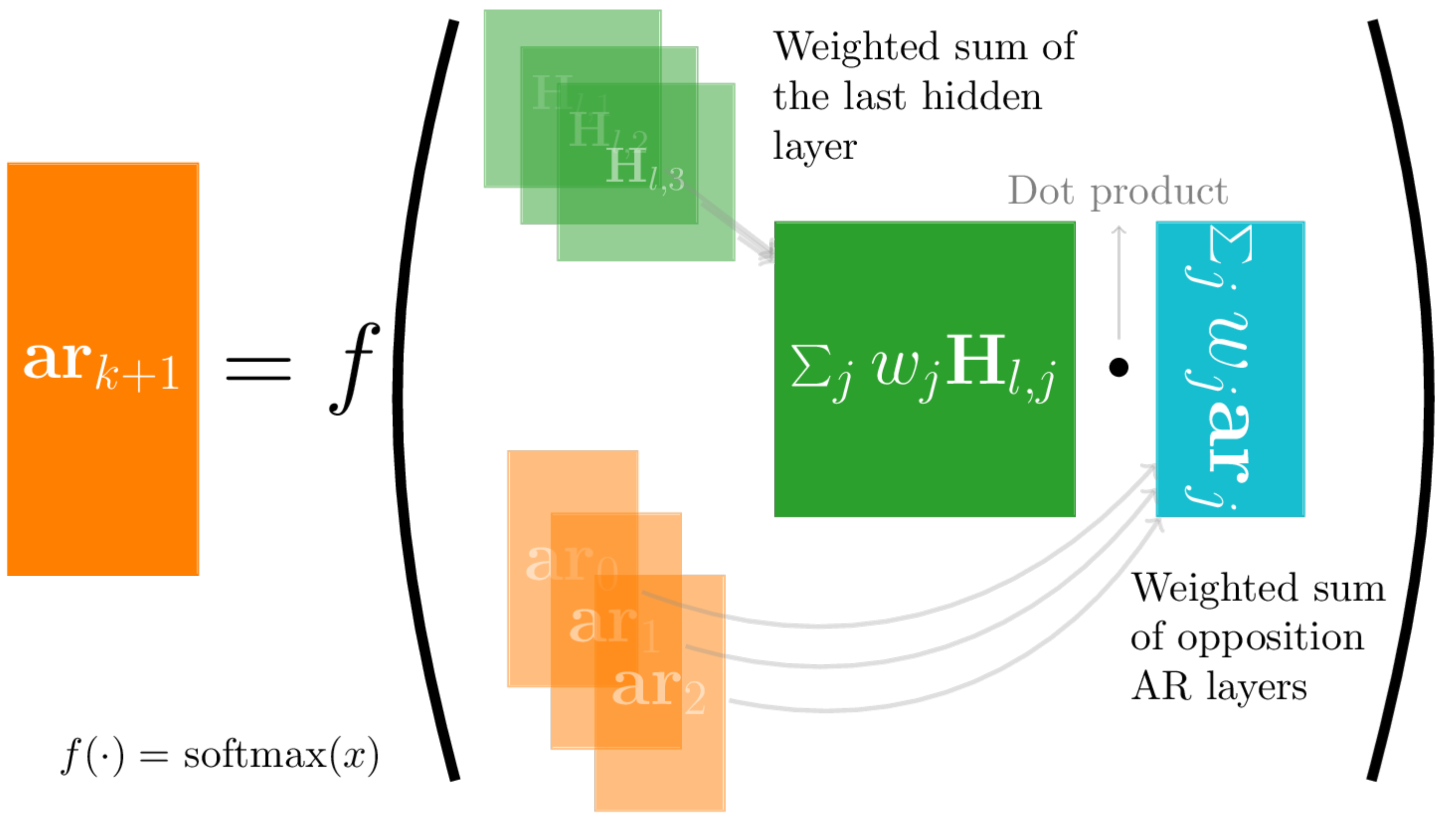
AR layers…

- Compute a **weighted** sum over P2 actions
- Weights are the model's predicted **distribution** over **P2's** actions
- Applied **recursively** to model multiple iterative reasoning steps

Action response layers

GT Wish List

1. Permutation
2. Size
3. Comparison
4. Iterative Reasoning

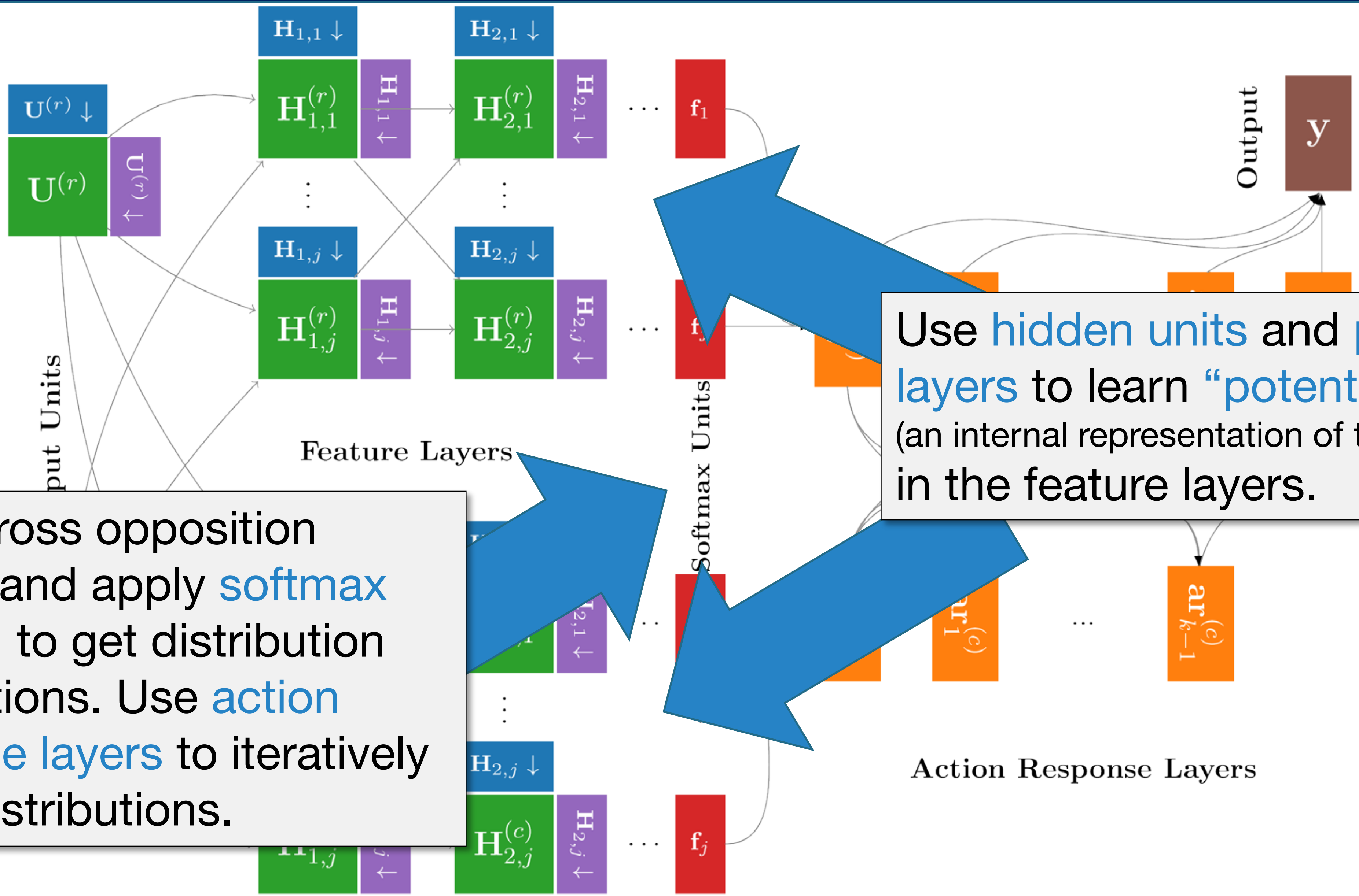


AR layers…

- Compute a **weighted** sum over P2 actions
- Weights are the model's predicted **distribution** over **P2's** actions
- Applied **recursively** to model multiple iterative reasoning steps

Full Architecture

GT Wish List
1. Permutation
2. Size
3. Comparison
4. Iterative Reasoning



Use hidden units and pooling layers to learn “potentials” (an internal representation of the input) in the feature layers.

Sum across opposition actions and apply softmax function to get distribution over actions. Use action response layers to iteratively refine distributions.

Representational generality

Our "**deep cognitive hierarchy**" subsumes previous approaches

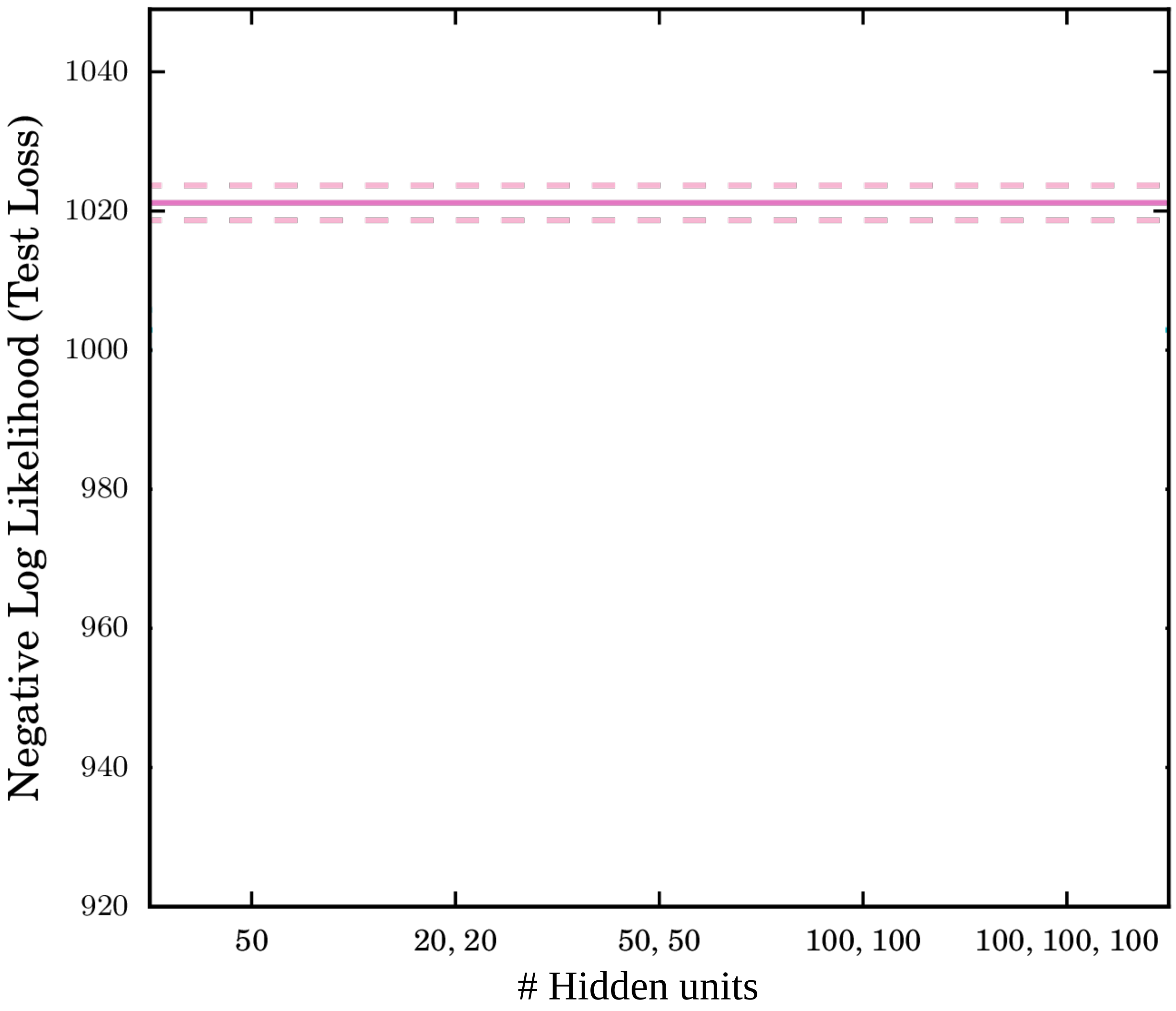
- Clearly, it generalizes **quantal cognitive hierarchy**
  - Action response layers can represent **level-k, cognitive hierarchy**
  - Agents can both **best respond** and **quantally respond**

- It also generalizes our **weighted linear level-0** extension:
  - Feature layers can represent **minmin unfairness, maxmax payoff, maxmin payoff, minimax regret, efficiency**

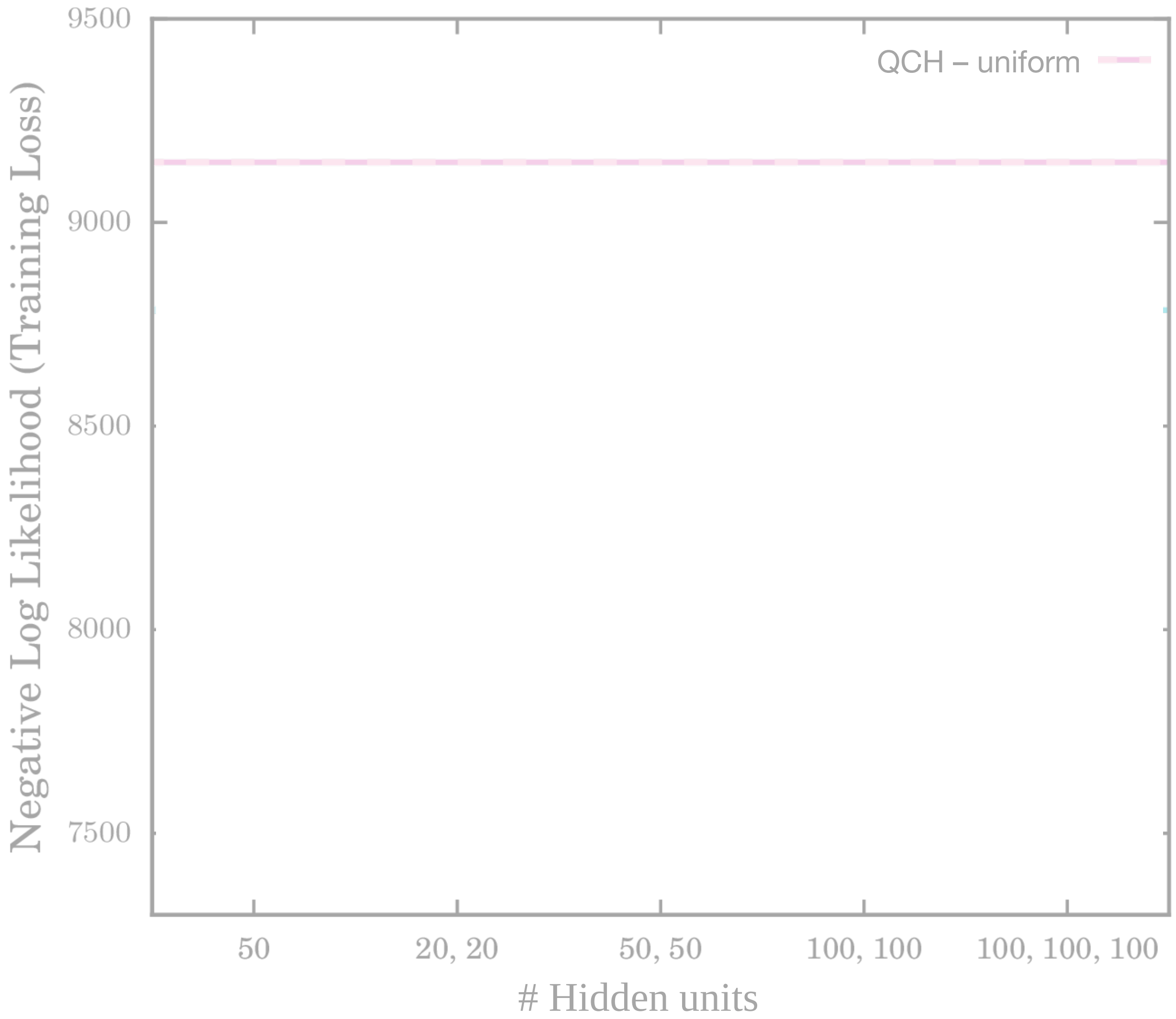
So…
does deep learning live up to the hype?

Performance

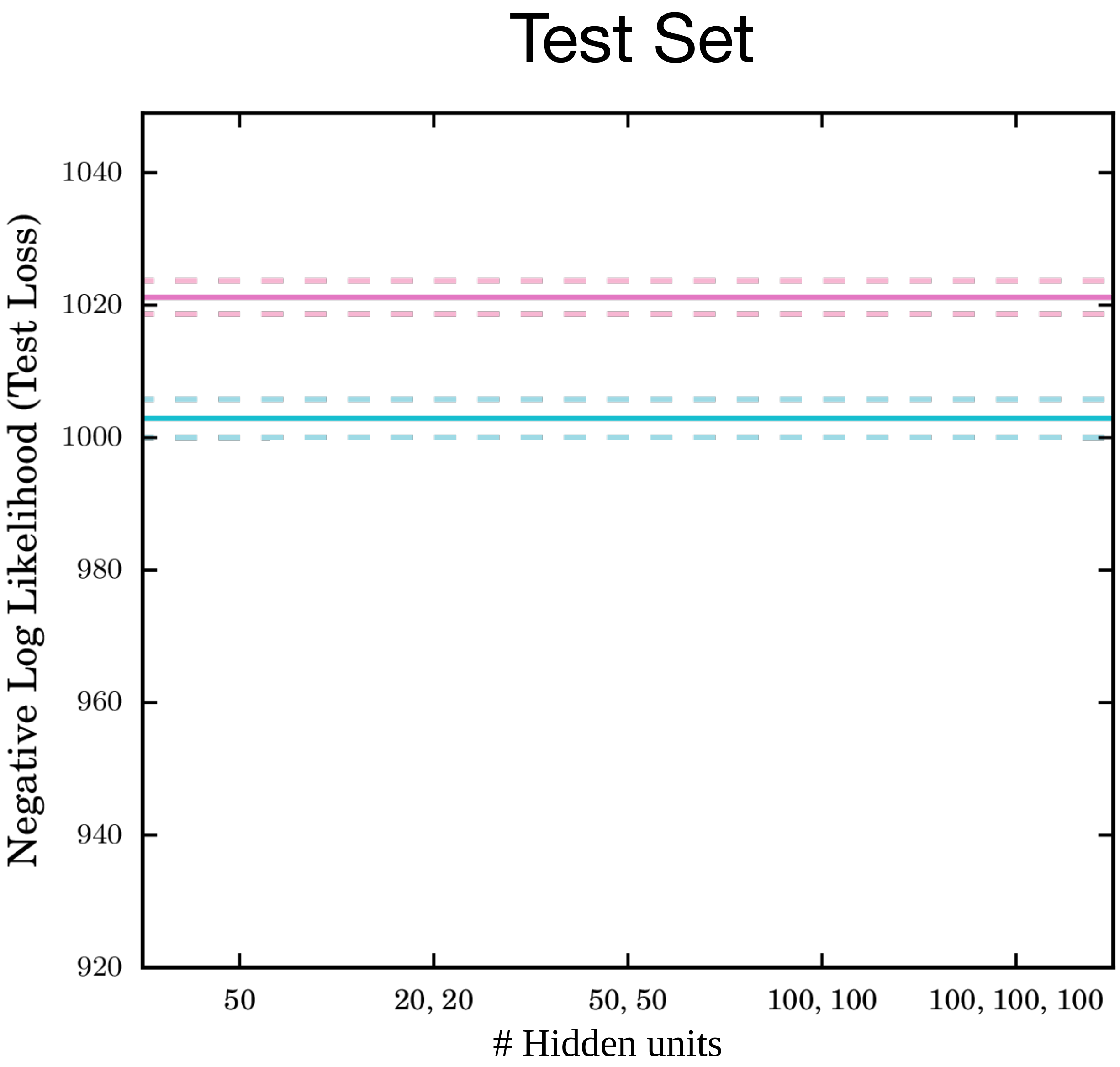
Test Set

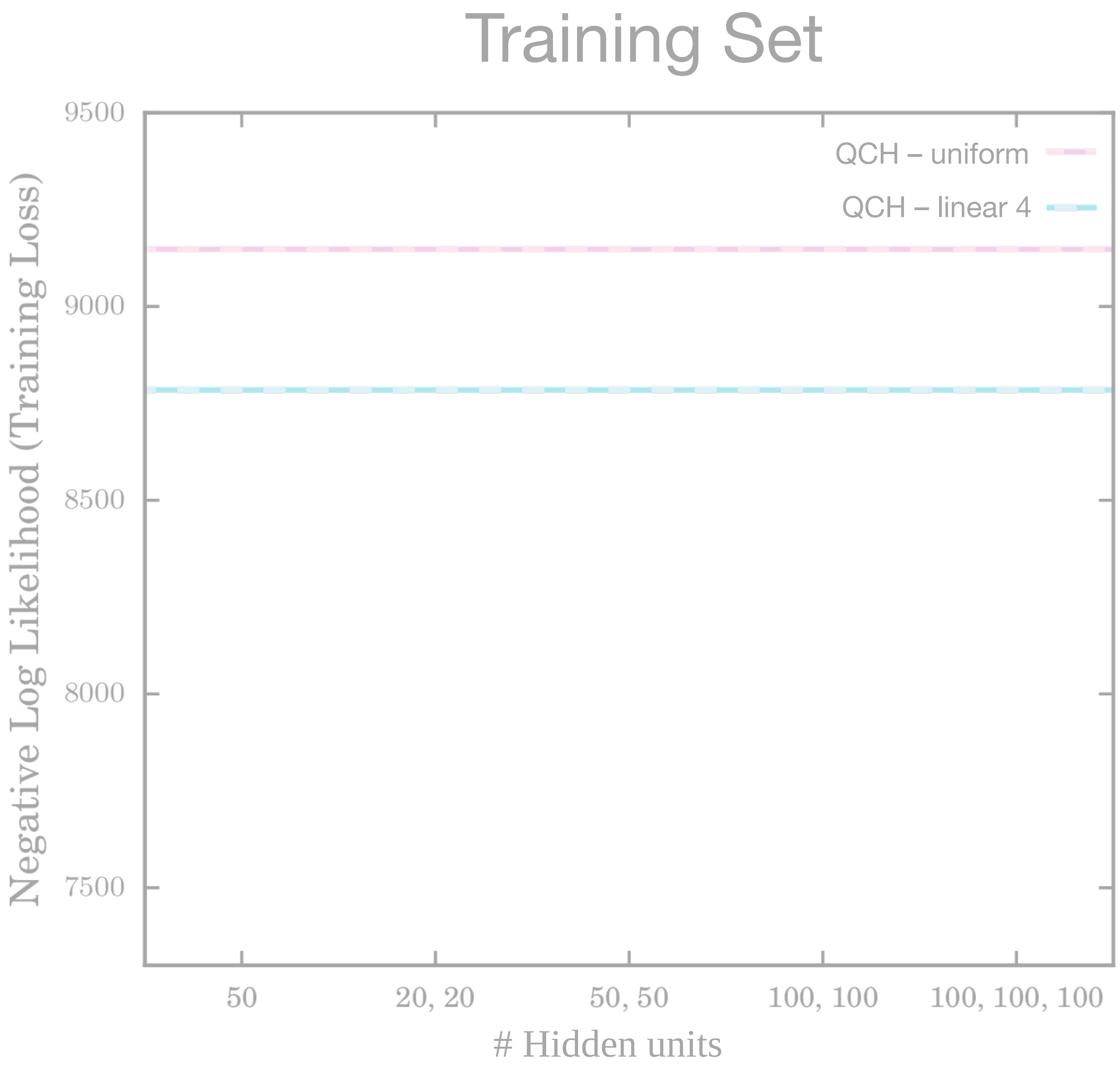


Training Set

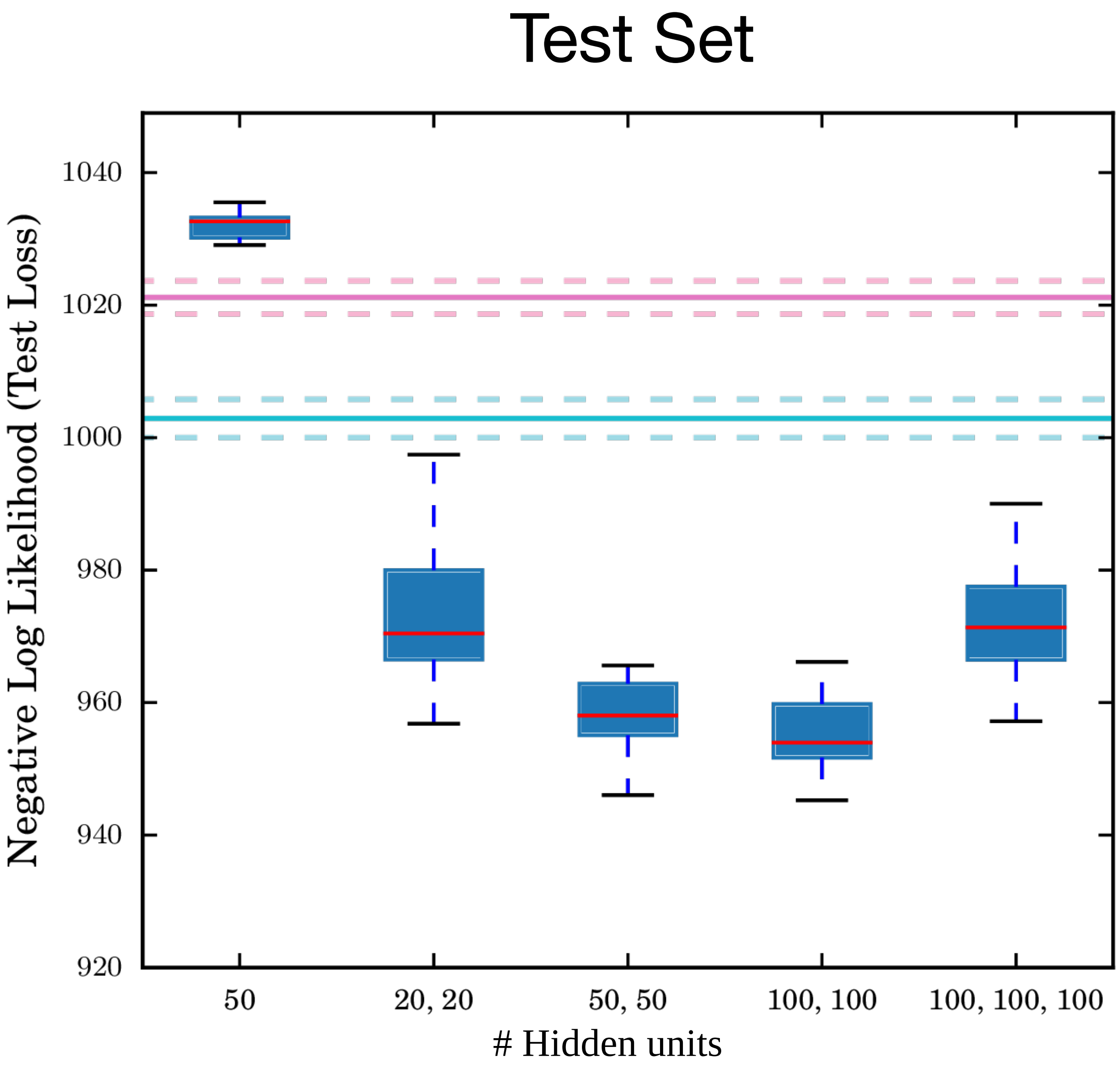


Performance



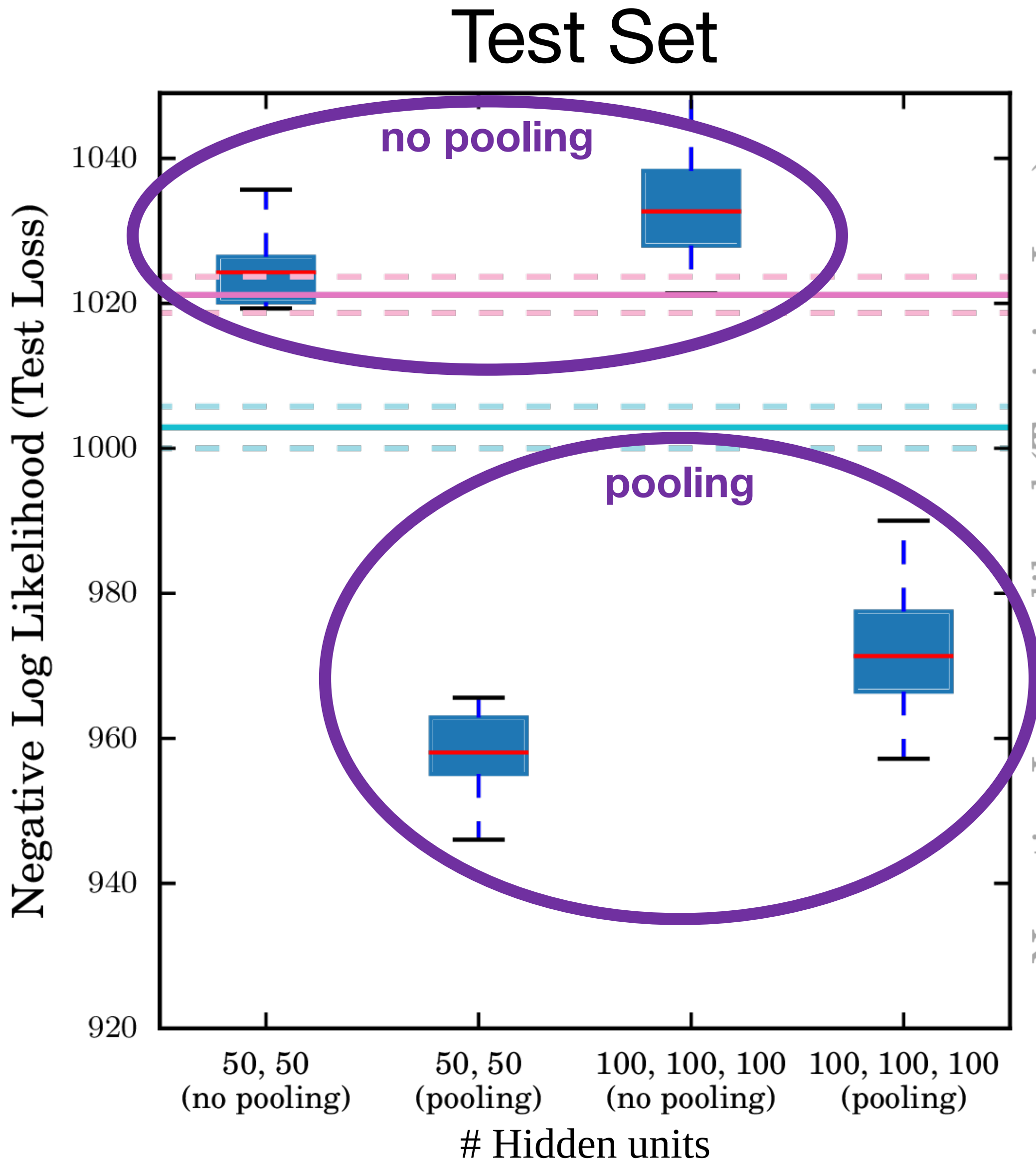


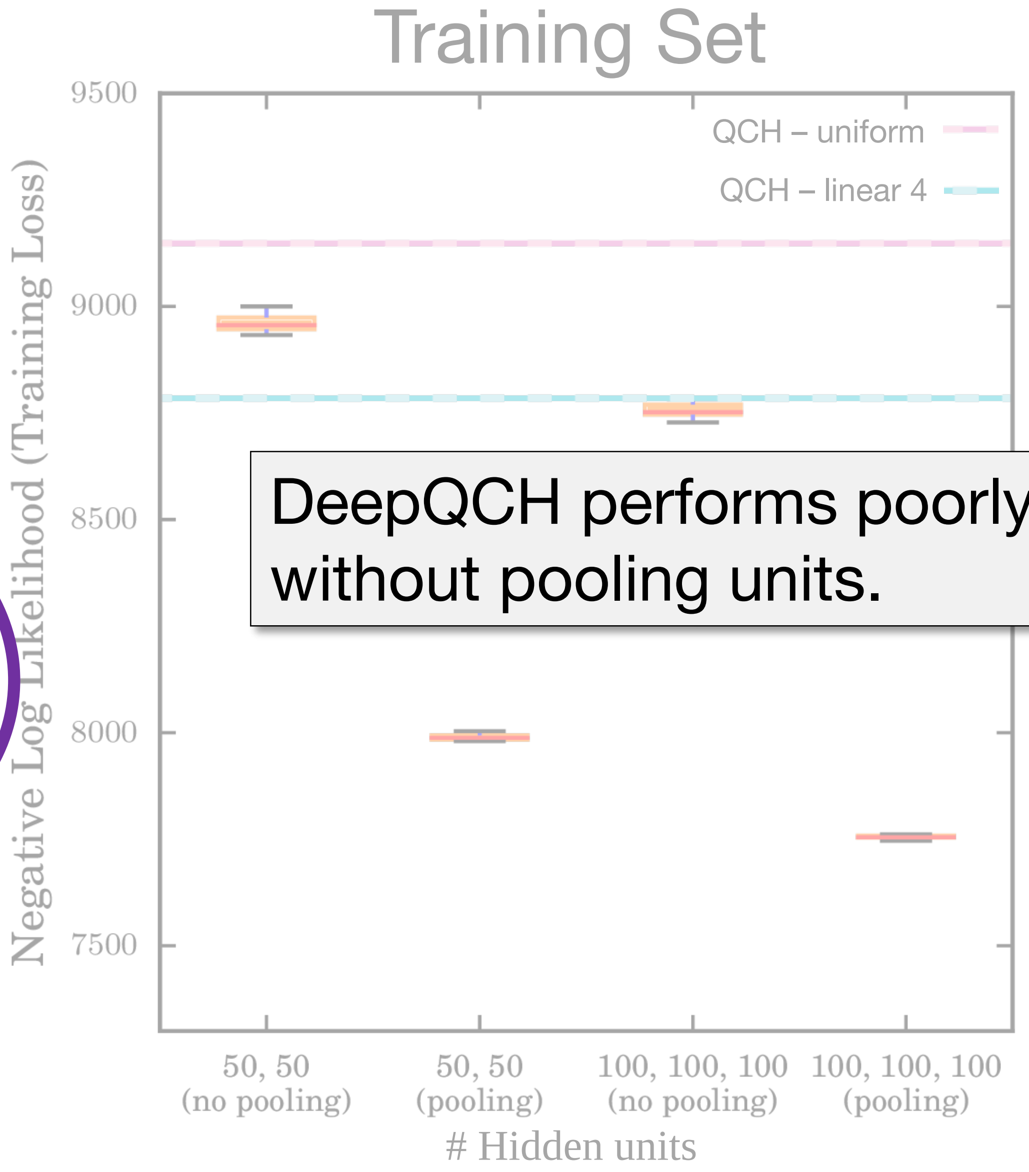
Overall performance



Large improvement in performance over the previous state of the art despite no access to hand-crafted features.

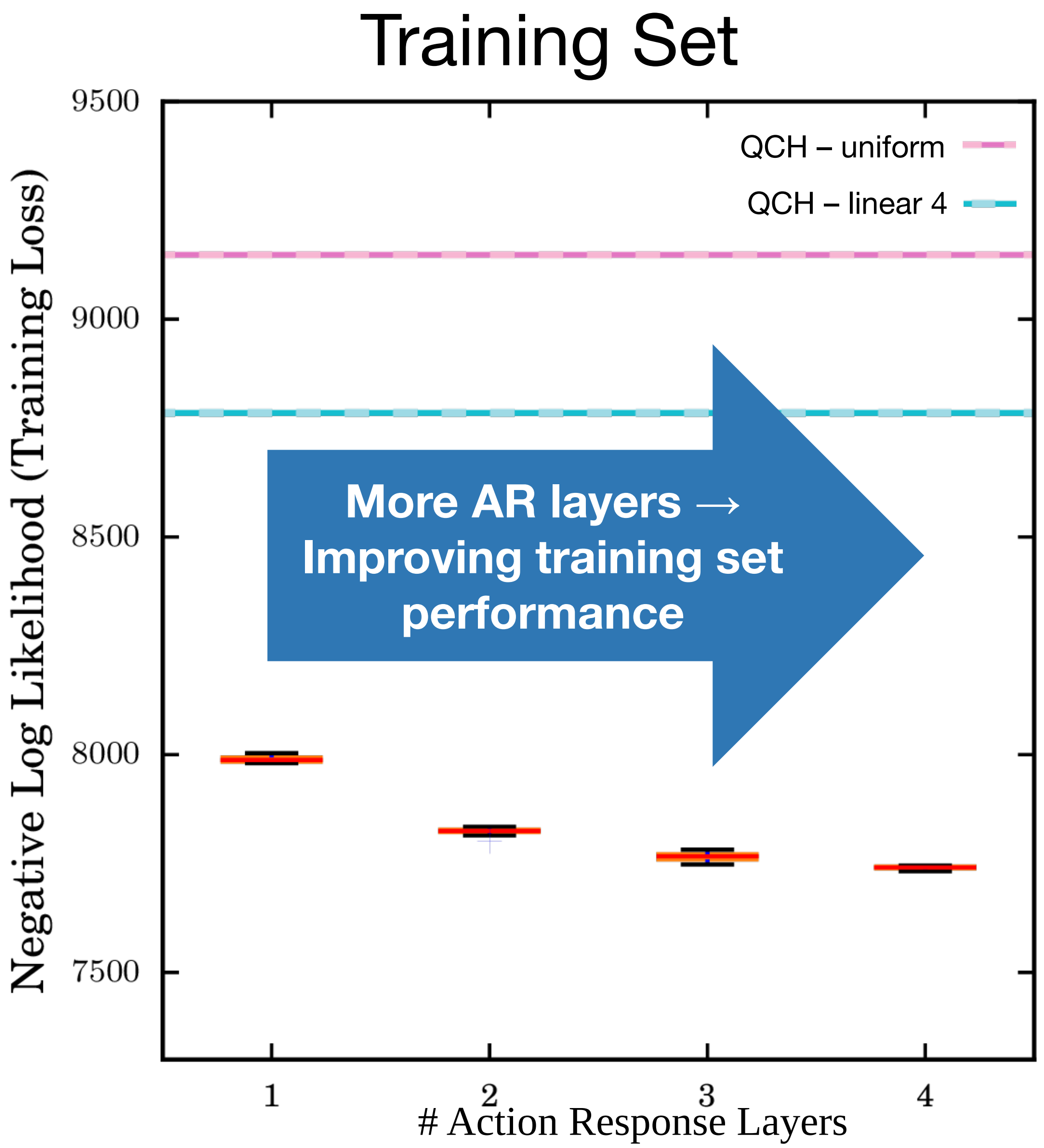
Performance – pooling layers



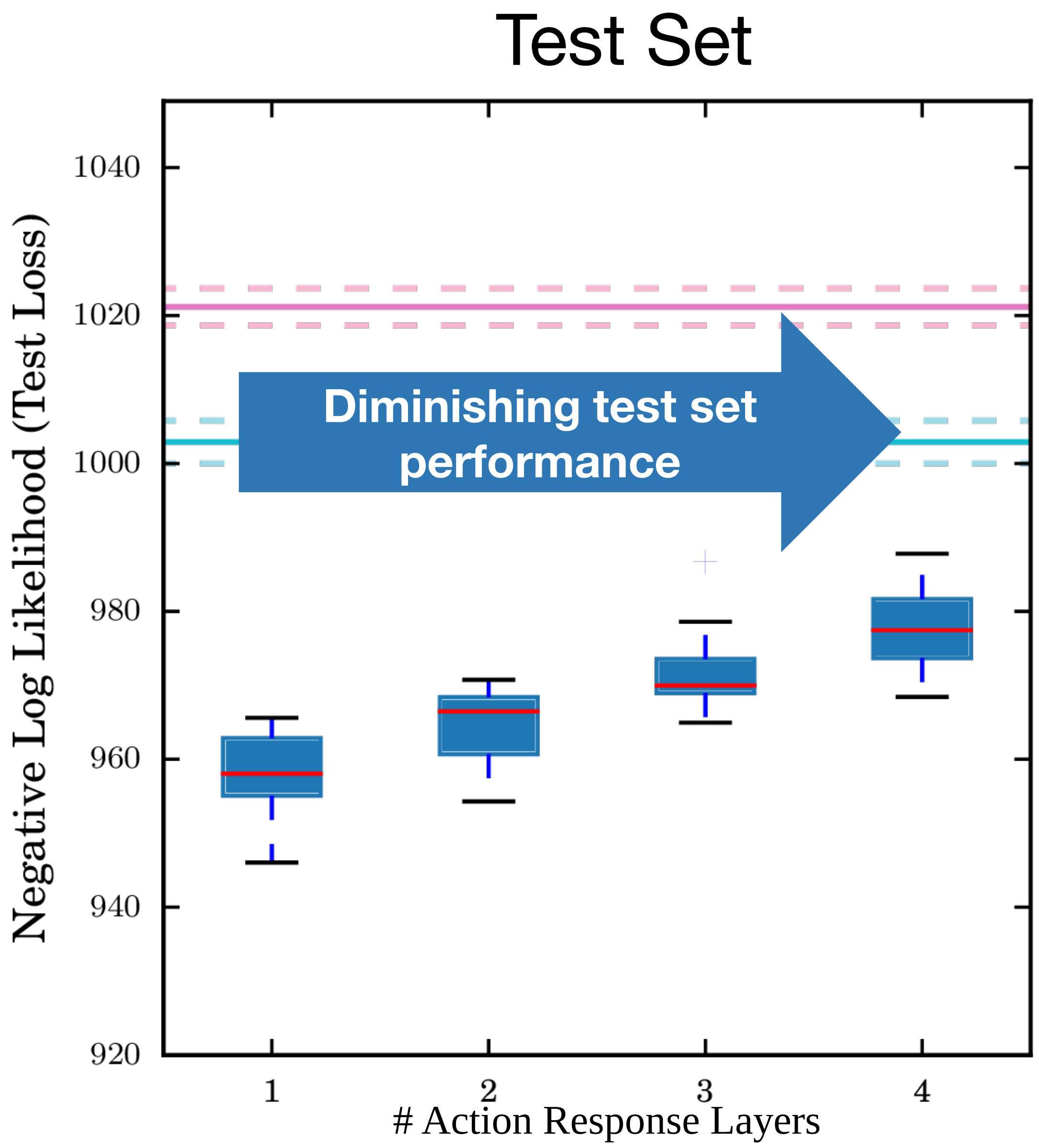


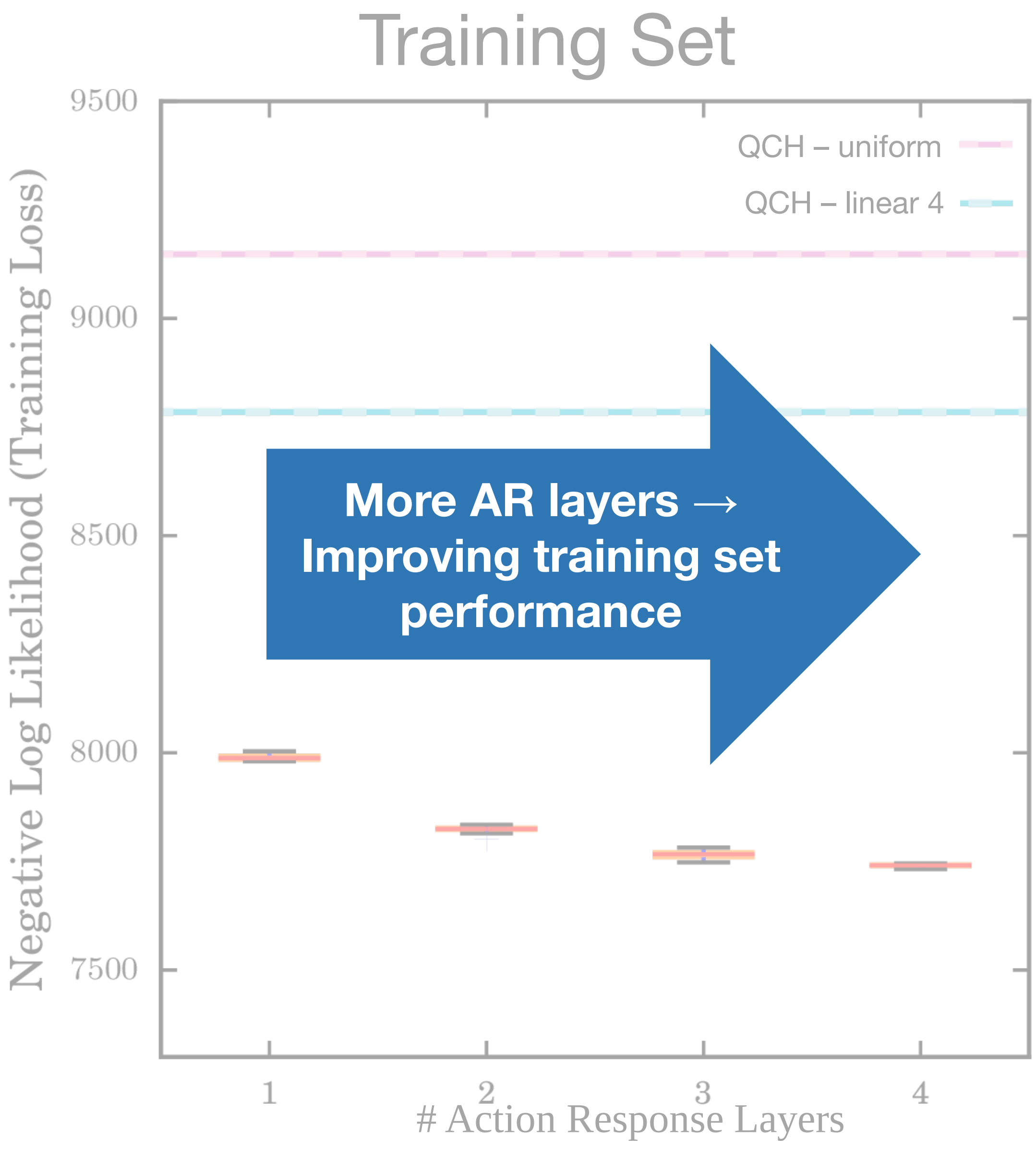
DeepQCH performs poorly without pooling units.

Performance – action response layers

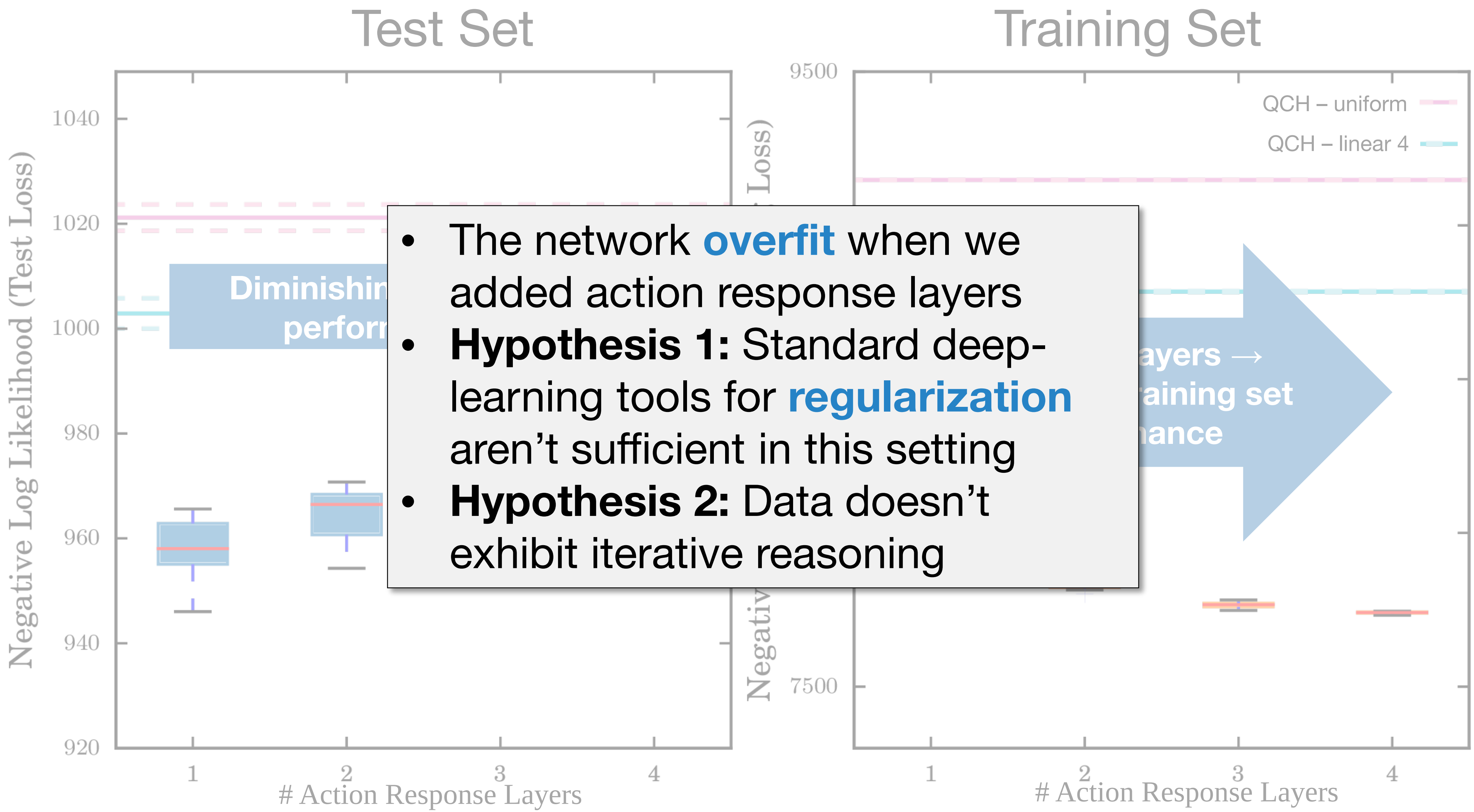


Performance – action response layers





Performance – action response layers



- The network **overfit** when we added action response layers
- **Hypothesis 1:** Standard deep-learning tools for **regularization** aren't sufficient in this setting
- **Hypothesis 2:** Data doesn't exhibit iterative reasoning

Conclusions and future directions

Architecture achieves **state-of-the-art** performance **predicting human behavior** in normal form games

- Generalizes to new games with **unseen number of actions**
- Model generalizes iterative response-style behavioral models
- …but the best-performing model didn't use action response layers

Future work:

- Build more flexible architecture to model **salience**; **dominance-style** features
- Explore the connection to **discrete choice**
  - in that setting, our model generalizes the latent class model