Learning to Win by Reading Manuals in a Monte-Carlo Framework

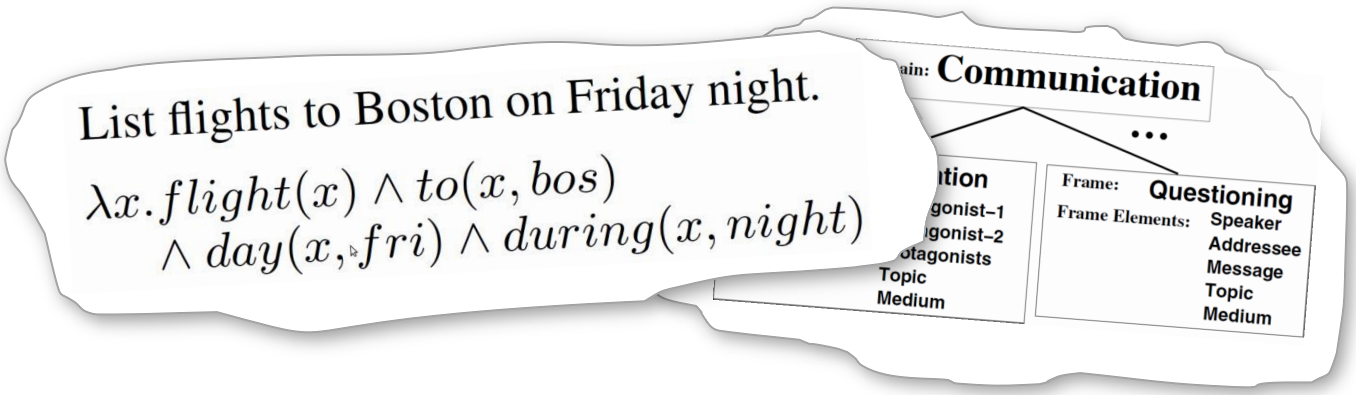
S.R.K. Branavan, David Silver, Regina Barzilay

MIT

Semantic Interpretation

Traditional view:

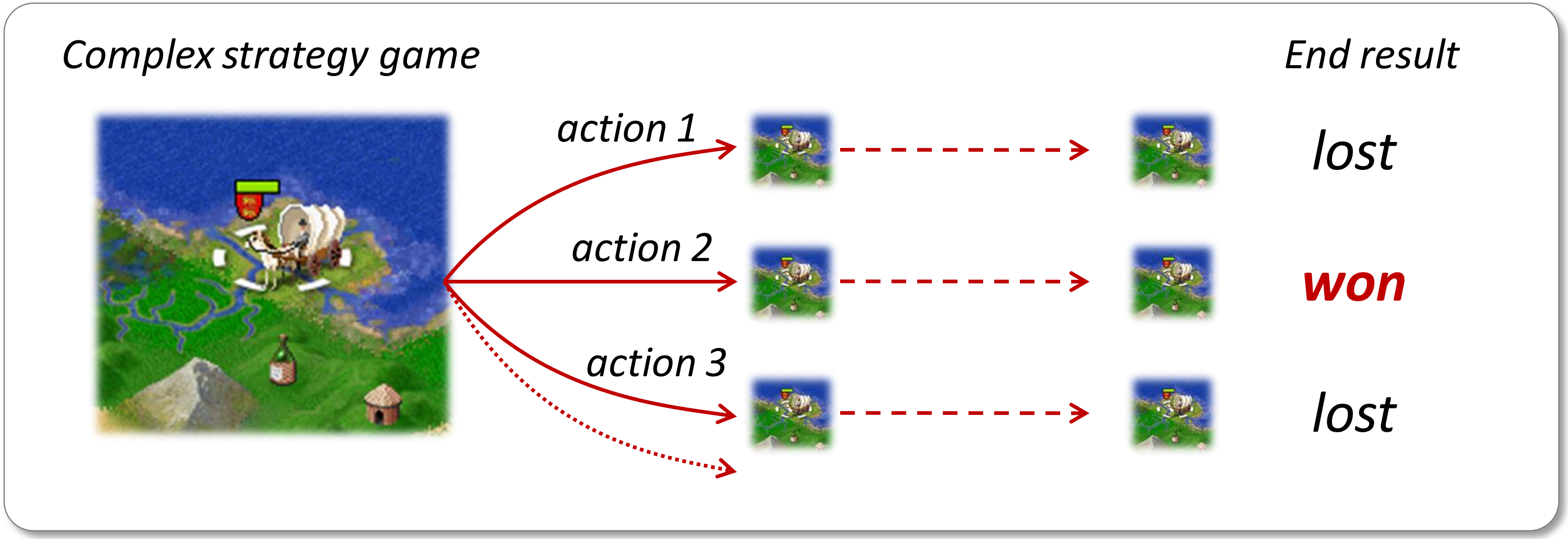
Map text into an abstract representation



Alternative view:

Map text into a representation which helps performance in a control application

Semantic Interpretation for Control Applications



Traditional approach:

Learn action-selection policy from game feedback.

Our contribution:

Use textual advice to guide action-selection policy.

Leveraging Textual Advice: Challenges

1. Find sentences relevant to given game state.

Game state

Strategy document

You start with two settler units. Although settlers are capable of performing a variety of useful tasks, your first task is to move the settlers to a site that is suitable for the construction of your first city. Use settlers to build the city on grassland with a river running through it if possible. You can also use settlers to irrigate land near your city. In order to survive and grow …

Leveraging Textual Advice: Challenges

1. Find sentences relevant to given game state.

Game state

Strategy document

*You start with two settler units. Although settlers are capable of performing a variety of useful tasks, your first task is to move the settlers to a site that is suitable for the construction of your first city. Use settlers to build the city on grassland with a river running through it if possible. You can also use settlers to **irrigate land near your city**. In order to survive and grow …*

Leveraging Textual Advice: Challenges

1. Find sentences relevant to given game state.

Game state

Strategy document

*You start with two settler units. Although settlers are capable of performing a variety of useful tasks, your first task is to move the settlers to a site that is suitable for the construction of your first city. Use settlers to **build the city on grassland with a river** running through it if possible. You can also use settlers to irrigate land near your city. In order to survive and grow ...*

Leveraging Textual Advice: Challenges

2. Label sentences with predicate stucture.

Move the settler to a site suitable for building a city, onto grassland with a river if possible.

`move_settlers_to()` ?

`settlers_build_city()` ?

Move the settler to a site suitable for building a city, onto grassland with a river if possible.

`move_settlers_to()`

Label words as *action*, *state* or *background*

Leveraging Textual Advice: Challenges

3. Guide action selection using relevant text

Build the city on plains or grassland with a river running through it if possible.

s

a_1 – `move_settlers_to(7,3)`

a_2 – `settlers_build_city()`

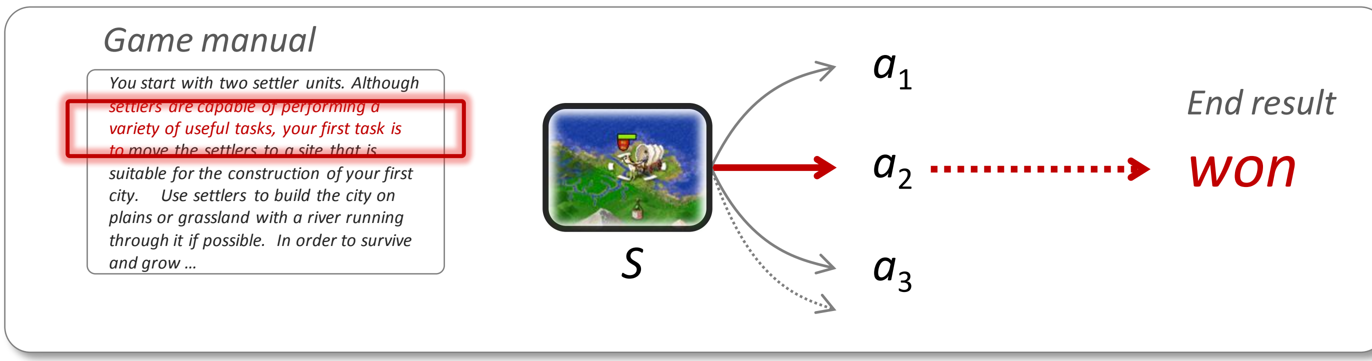
a_3 – `settlers_irrigate_land()`

Learning from Game Feedback

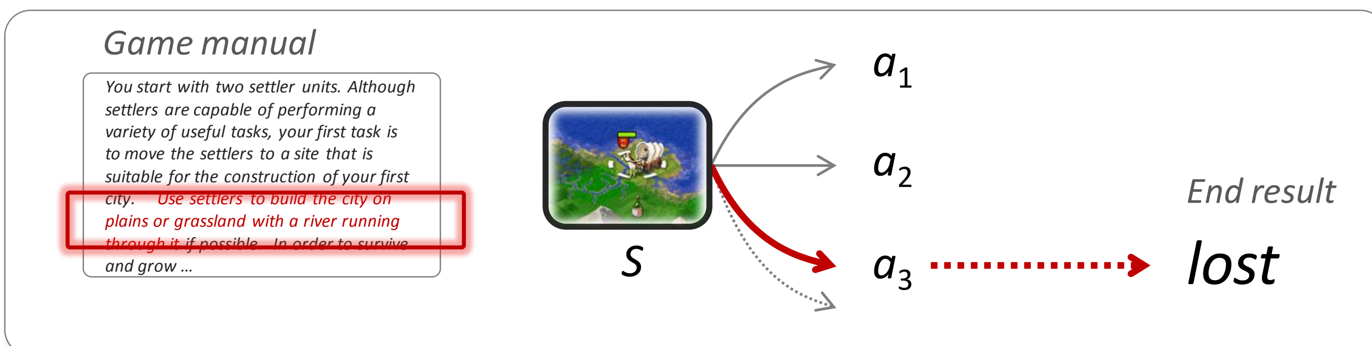
Goal: *Learn from game feedback as only source of supervision.*

Key idea: *Better parameter settings will lead to more victories.*

Model params: θ_1



Model params: θ_2



Model Overview

Monte-Carlo Search Framework

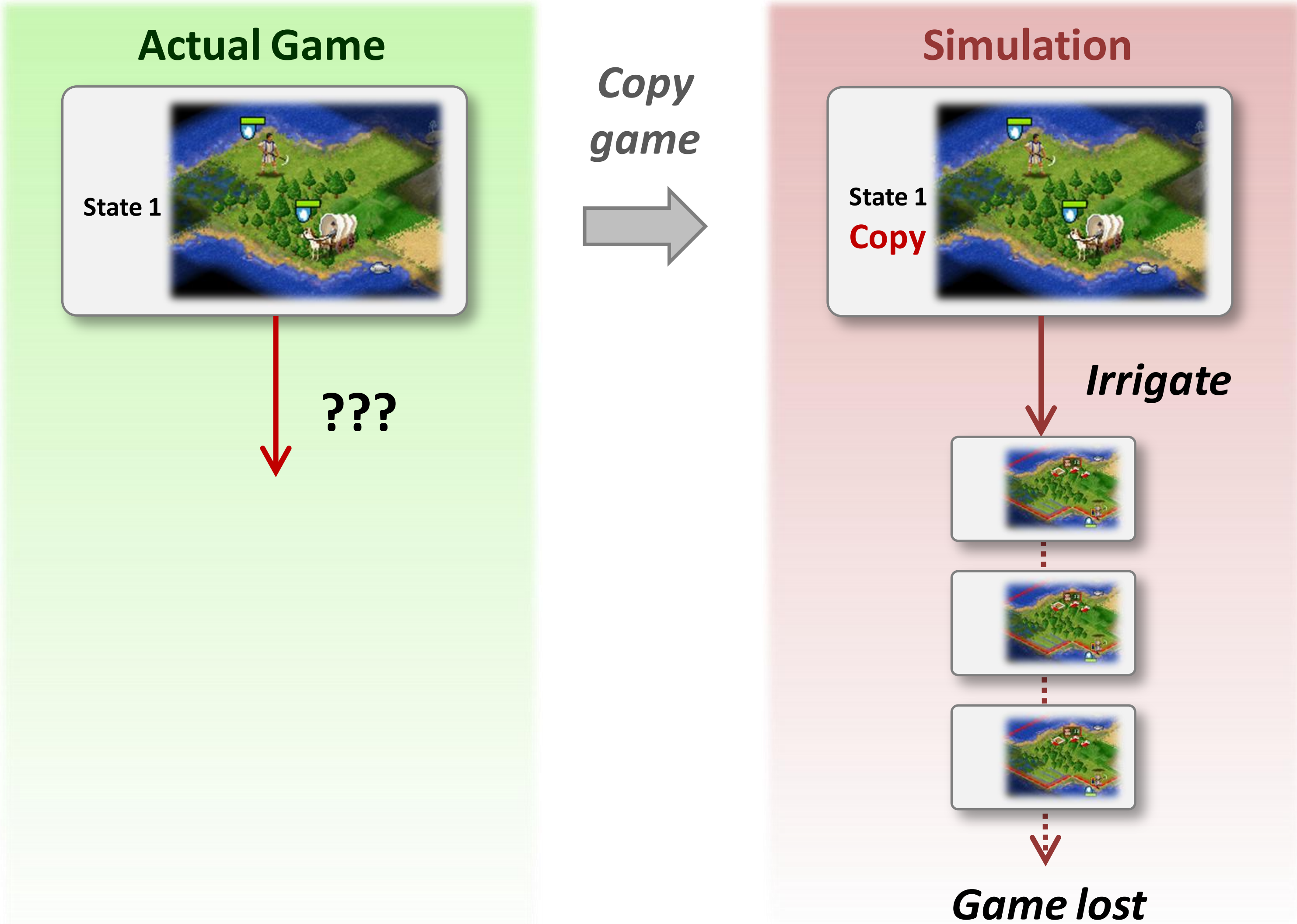
- *Learn action selection policy from simulations*
- *Very successful in complex games like Go and Poker.*

Our Algorithm

- *Learn text interpretation from simulation feedback*
- *Bias action selection policy using text*

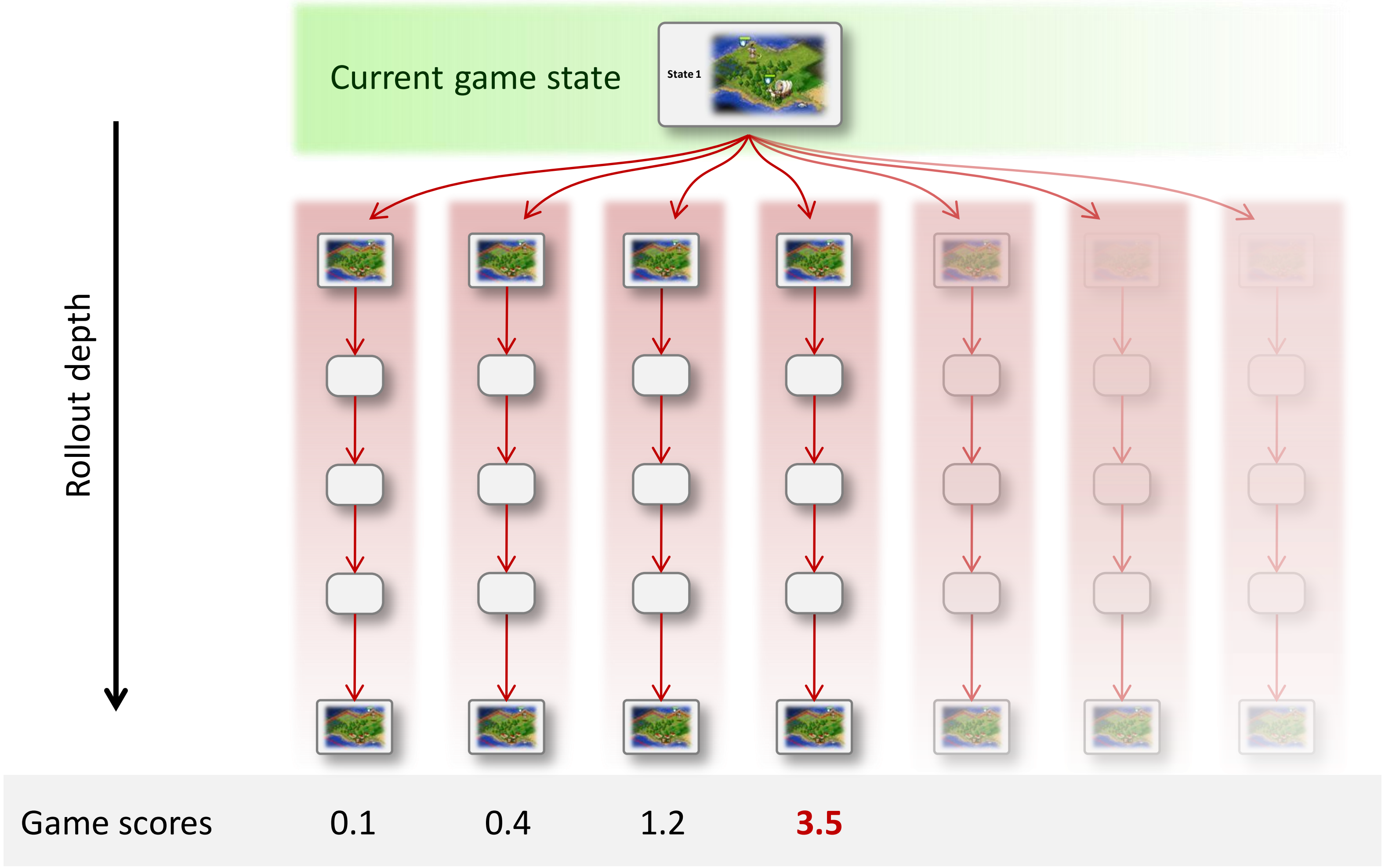
Monte-Carlo Search

Select actions via simulations, game and opponent can be stochastic



Monte-Carlo Search

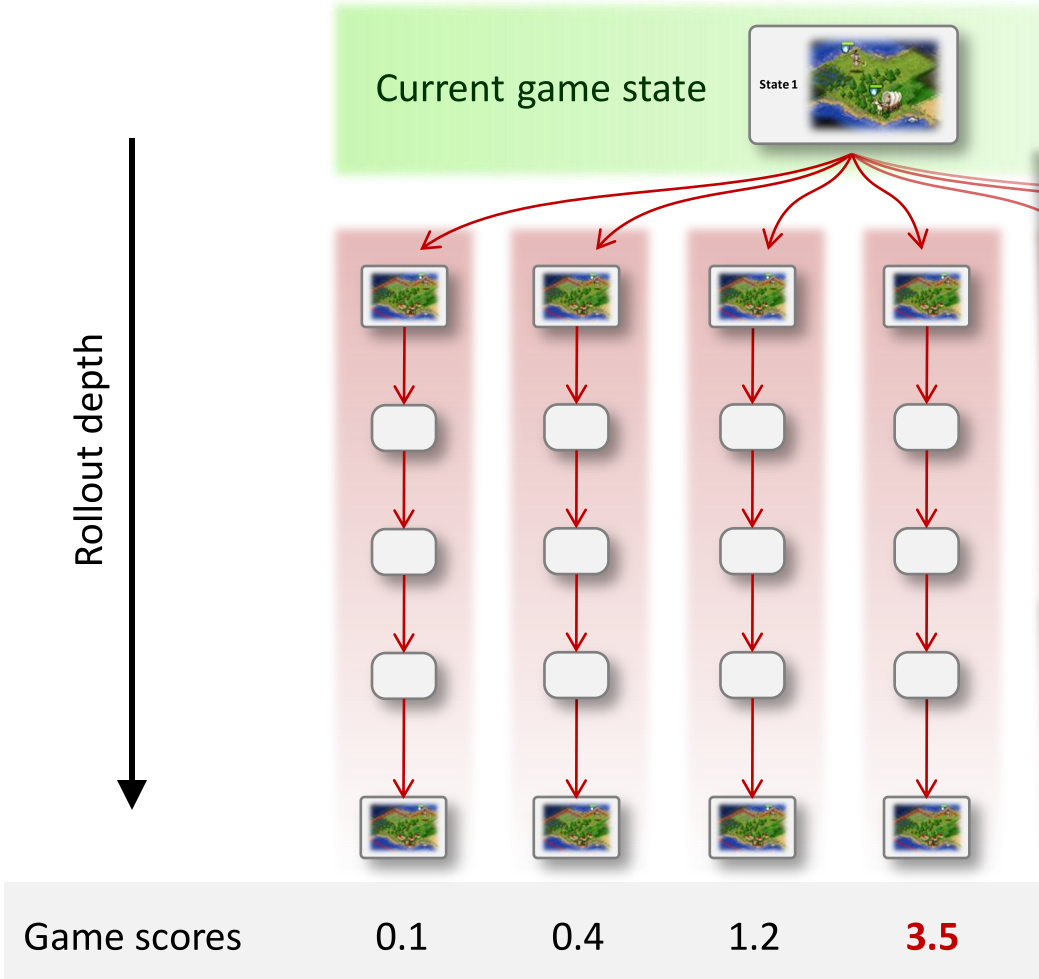
Try many candidate actions from current state & see how well they perform.



Monte-Carlo Search

Try many candidate actions from current state & see how well they perform.

Learn feature weights from simulation outcomes



5	1	0	1	1	0	1	**= 0.1**
15	0	1	0	0	1	0	**= 0.4**
37	1	0	1	0	0	0	**= 1.2**

. . . .

$$Q(s,a) \;\propto\; \vec{\theta} \cdot \vec{\phi}(s,a)$$

$\vec{\phi}(s,a)$ - *feature function*

$\vec{\theta}$ - *model parameters*

Model Overview

Monte-Carlo Search Framework

- *Learn action selection policy from simulations*

Our Algorithm

- *Bias action selection policy using text*
- *Learn text interpretation from simulation feedback*

Modeling Requirements

- *Identify sentence relevant to game state*

Build cities near rivers or ocean.

- *Label sentence with predicate structure*

Build cities near rivers or ocean.

Build cities *near* ***rivers or ocean.***

- *Estimate value of candidate actions*

Build cities *near* ***rivers or ocean.***

Irrigate :	*-10*
Fortify :	*-5*
. . . .	
Build city :	*25*

Sentence Relevance

1
2
3

Identify sentence relevant to game state and action

State s, *candidate action* a, *document* d

$$p(y = y_i | s, a, d) \propto e^{\vec{u} \cdot \vec{\phi}(y_i, s, a, d)}$$

Sentence y_i *is selected as relevant*

Log-linear model:

$\vec{u}$ - *weight vector*

$\vec{\phi}(y_i, s, a, d)$ - *feature function*

Predicate Structure

1
2
3

Select word labels based on sentence + dependency info

E.g., *"**Build cities** near **rivers or ocean.**"*

Word index j, sentence y, dependency info q

$$p(e_j \mid j, y, q) \propto e^{\vec{v} \cdot \vec{\psi}(e_j, j, y, q)}$$

Predicate label e_j = { ***action***, ***state***, ***background*** }

Log-linear model:

$\vec{v}$ - *weight vector*

$\vec{\psi}(e_j, j, y, q)$ - *feature function*

Final Q function approximation

1
2
3

Predict expected value of candidate action

State s, candidate action a

$$Q(s, a, d, y_i, \vec{e}_i) = \vec{w} \cdot \vec{f}(s, a, d, y_i, \vec{e}_i)$$

Document d, relevant sentence y_i, predicate labeling $\vec{e}_i$

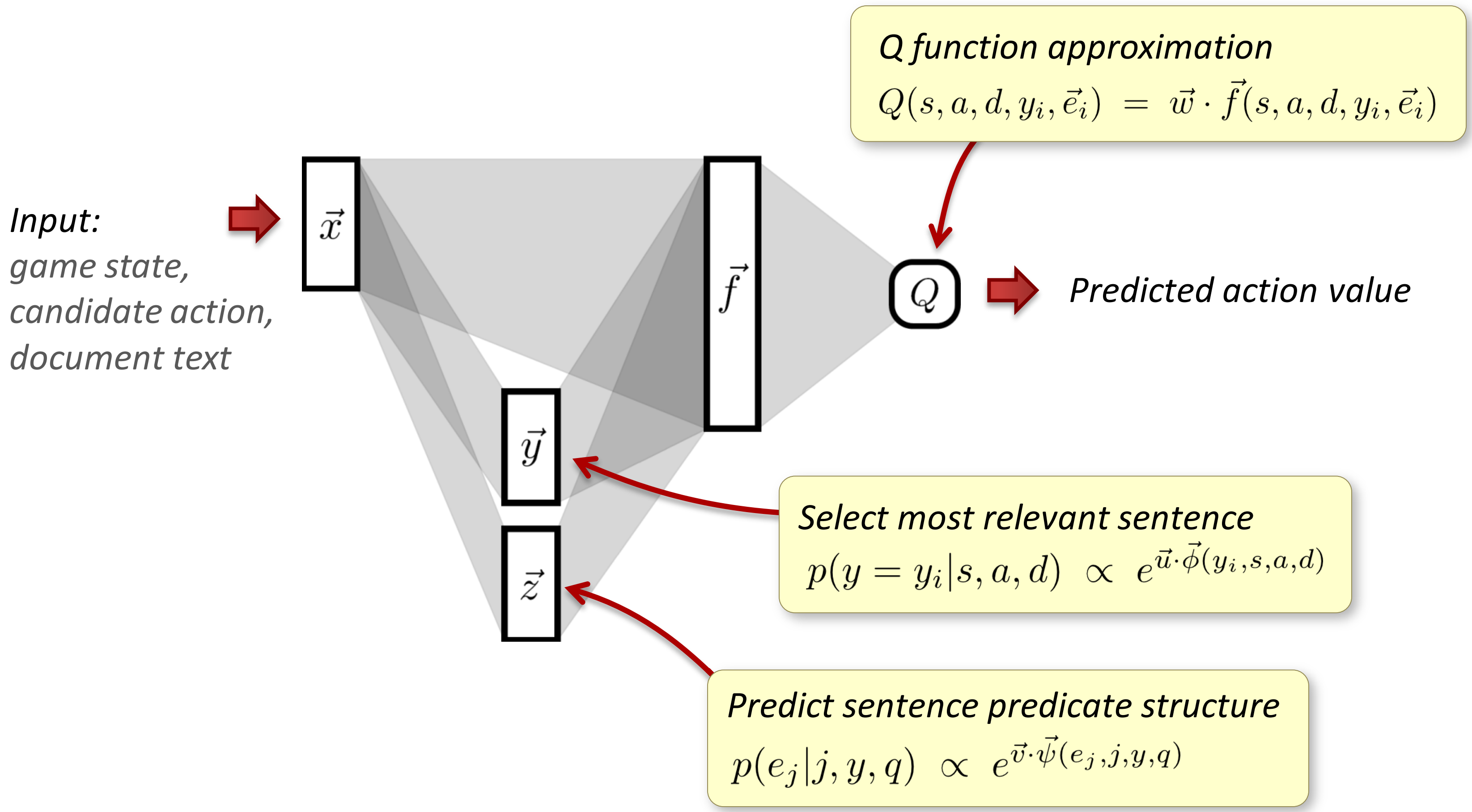
Linear model:

$\vec{w}$ - *weight vector*

$\vec{f}(s, a, d, y_i, \vec{e}_i)$ - *feature function*

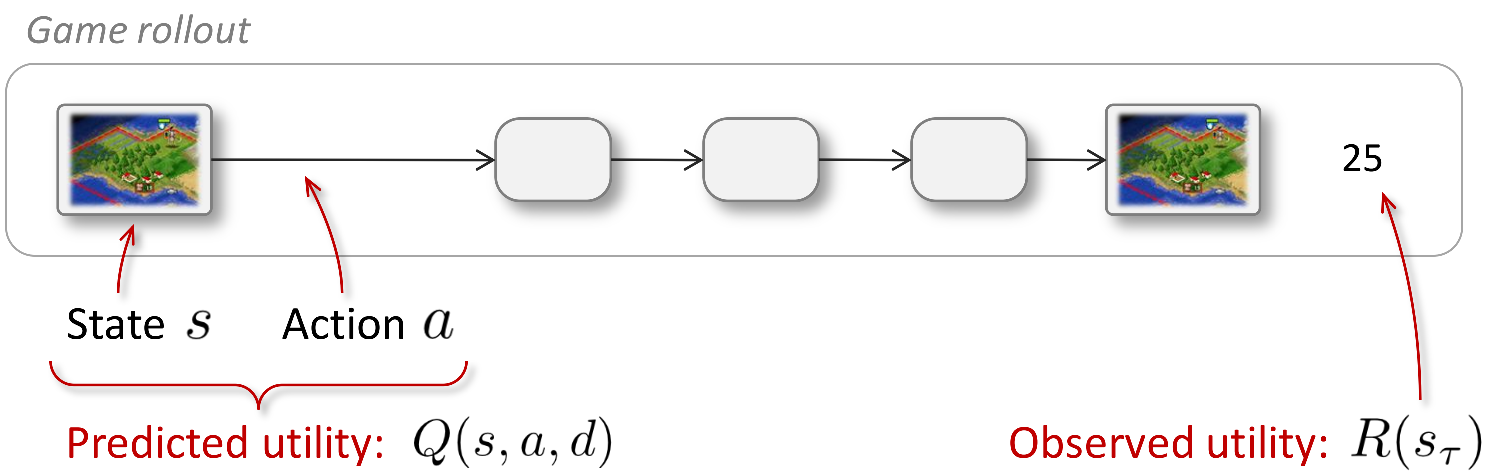
Model Representation

Multi-layer neural network: *Each layer represents a different stage of analysis*



Parameter Estimation

Objective: Minimize *mean square error* between predicted utility $Q(s, a, d)$ and observed utility $R(s_\tau)$



Parameter Estimation

Method: Gradient descent – i.e., Backpropagation.

Parameter updates:

$$\vec{u}_i \leftarrow \vec{u}_i + \alpha_u \,[Q - R(s_\tau)]\; Q\, \vec{x}\, [1 - p(y_i|\cdot)]$$

$$\vec{v}_i \leftarrow \vec{v}_i + \alpha_v \,[Q - R(s_\tau)]\; Q\, \vec{x}\, [1 - p(e_i|\cdot)]$$

$$\vec{w} \leftarrow \vec{w} + \alpha_w \,[Q - R(s_\tau)]\; \vec{f}(s, a, d, y_i, z_j)$$

Features

State features:

- *Amount of gold in treasury*
- *Government type*
- *Terrain surrounding current unit*

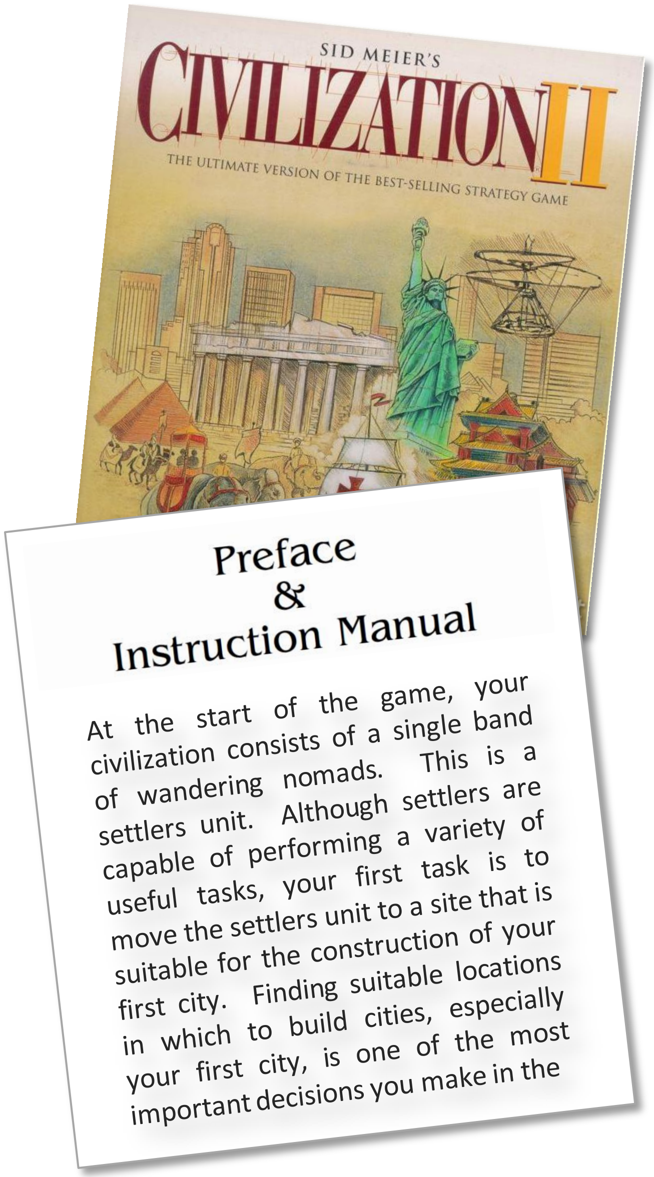
Action features:

- *Unit type (settler, worker, archer, etc)*
- *Unit action type*

Text features:

- *Word*
- *Parent word in dependency tree*
- *Word matches text label of unit*

Experimental Domain



Game:

- *Complex, stochastic turn-based strategy game Civilization II.*
- *Branching factor:* 10^{20}

Document:

- *Official game manual of Civilization II*

Text Statistics:

Sentences:	***2083***
Avg. sentence words:	***16.7***
Vocabulary:	***3638***

Experimental Setup

Game opponent:

- *Built-in AI of Game.*
- *Domain knowledge rich AI, built to challenge humans.*

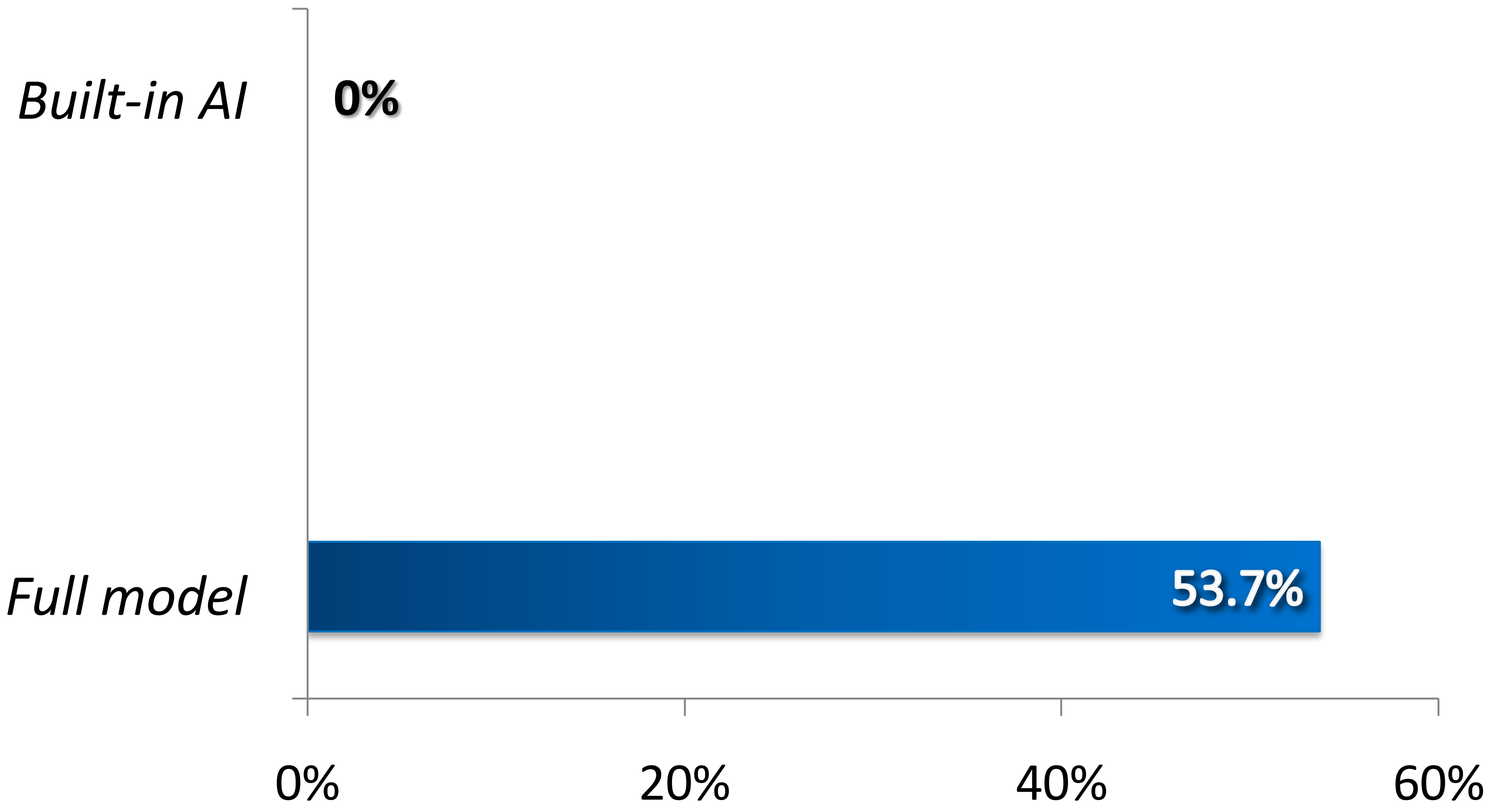
Primary evaluation:

- *Games won within first 100 game steps.*
- *Averaged over 200 independent experiments.*
- *Avg. experiment runtime: 1.5 hours*

Secondary evaluation:

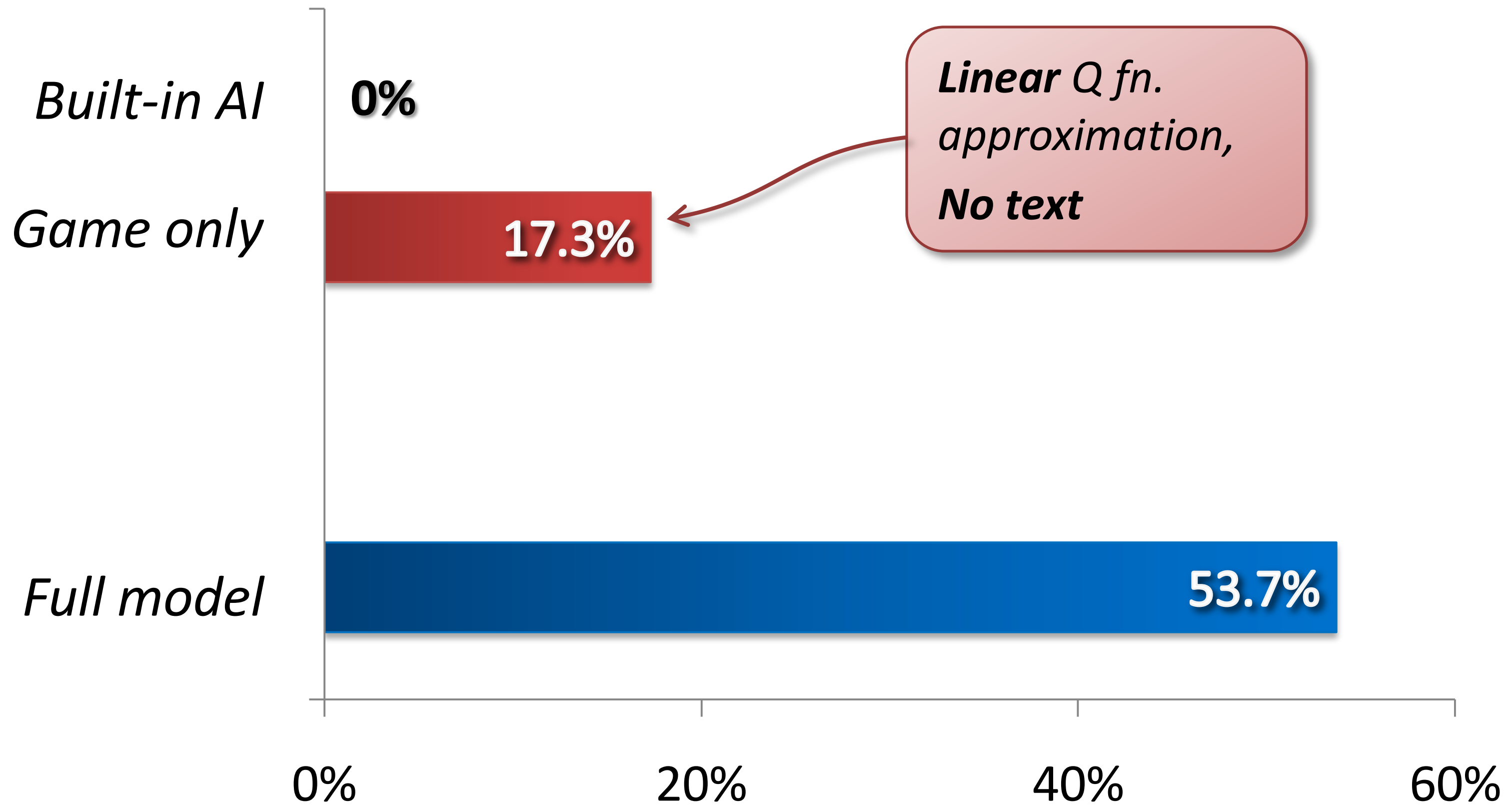
- *Full games won.*
- *Averaged over 50 independent experiments.*
- *Avg. experiment runtime: 4 hours*

Results



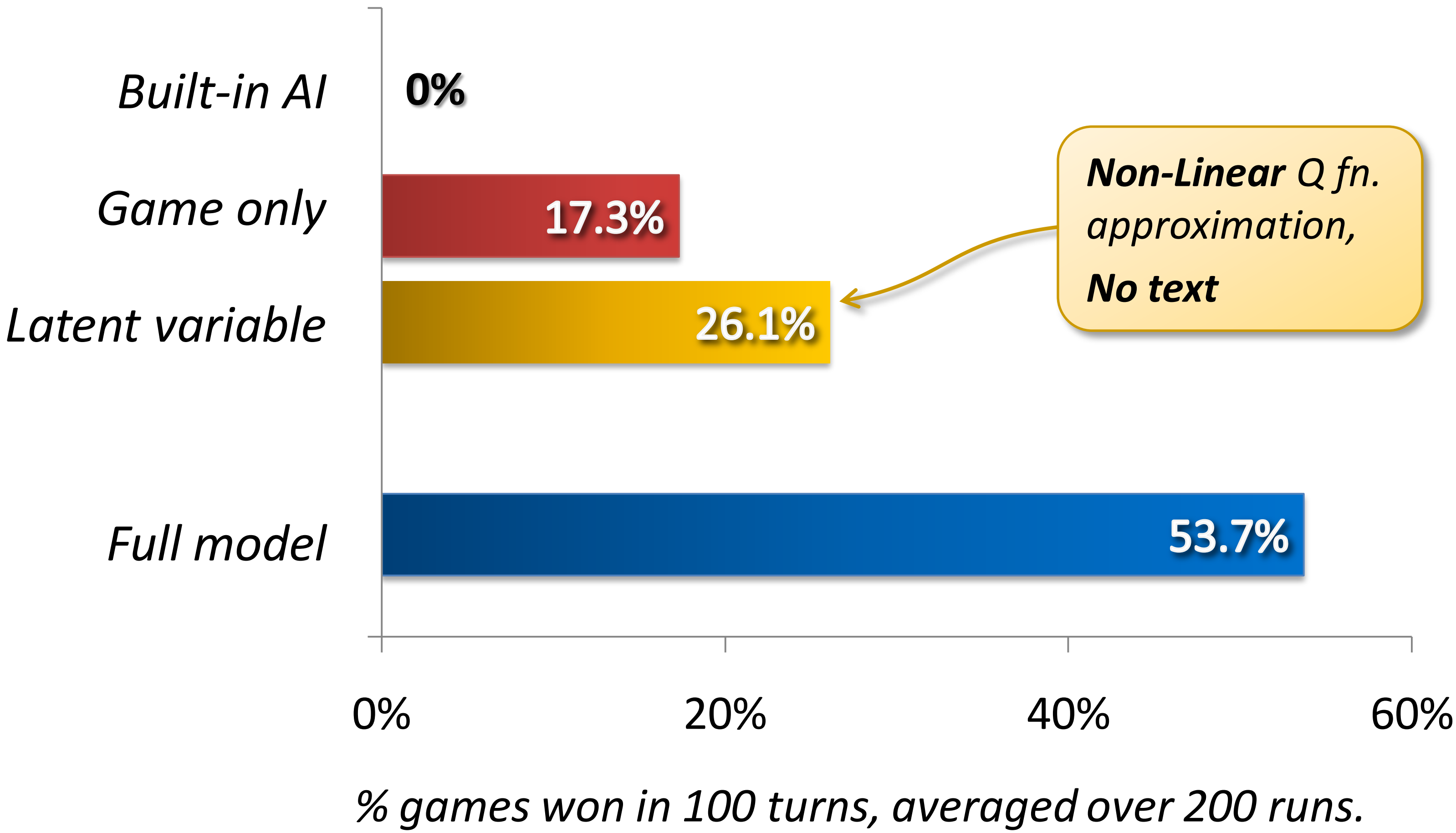
% games won in 100 turns, averaged over 200 runs.

Does Text Help ?

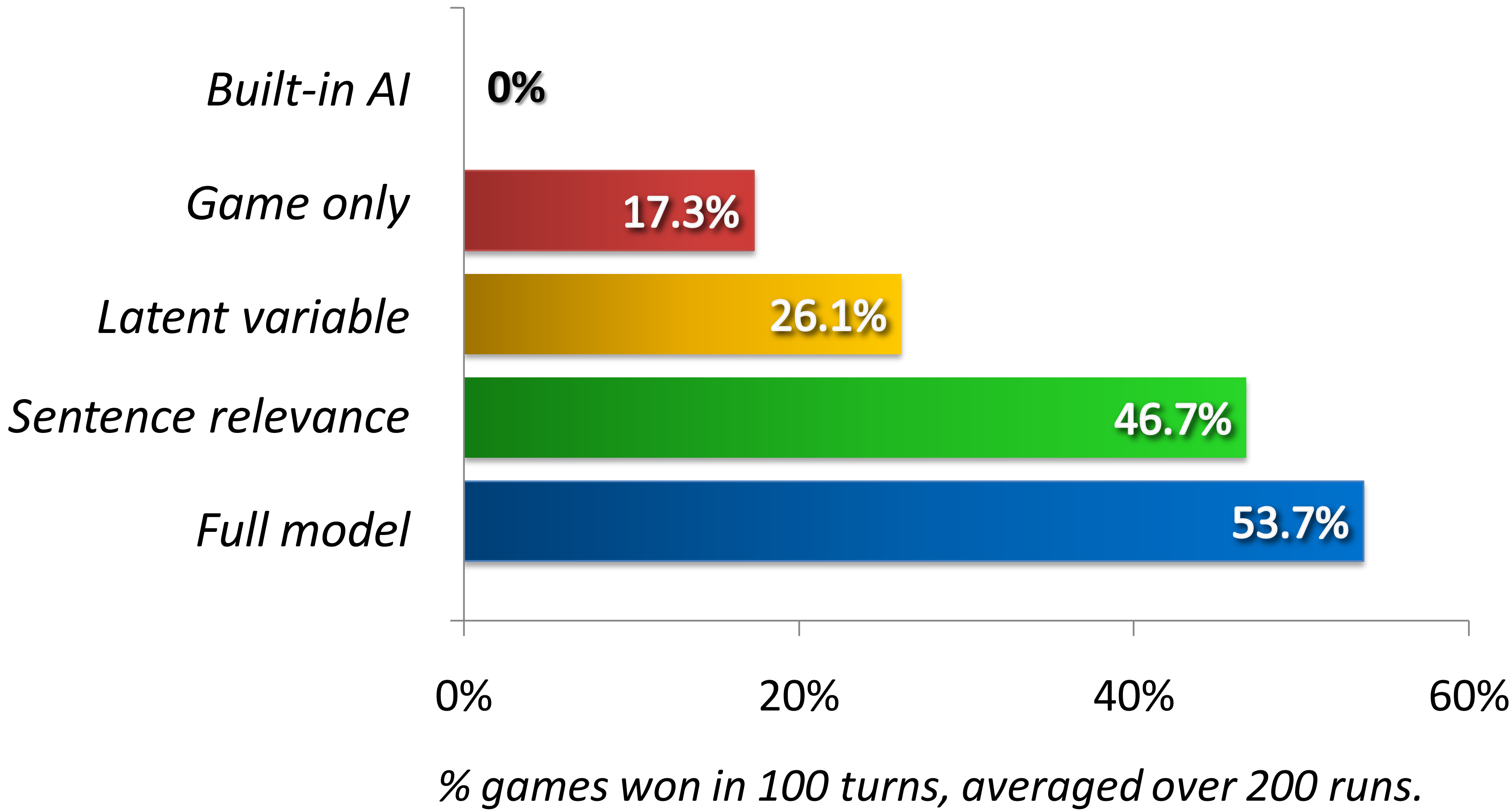


% games won in 100 turns, averaged over 200 runs.

Text vs. Representational Capacity



Linguistic Complexity vs. Performance Gain



Results: Sentence Relevance

Problem: *Sentence relevance depends on game state.*

States are game specific, and not known a priori!

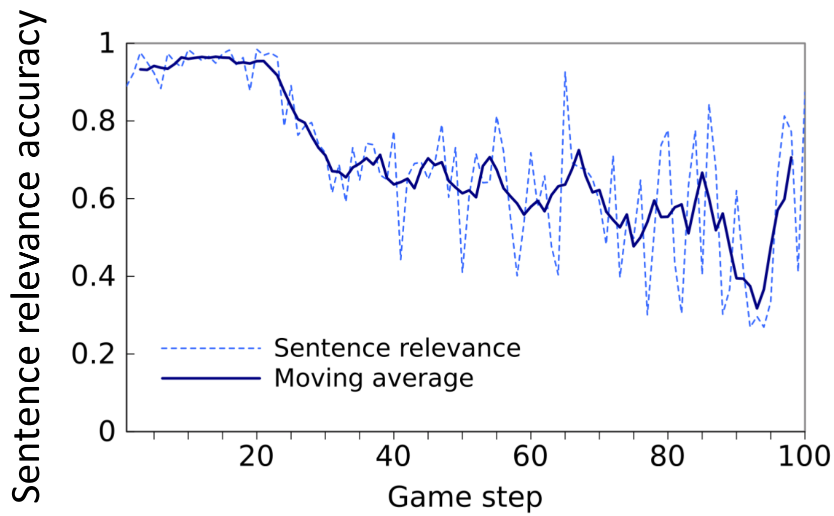
Solution: *Add known non-relevant sentences to text.*

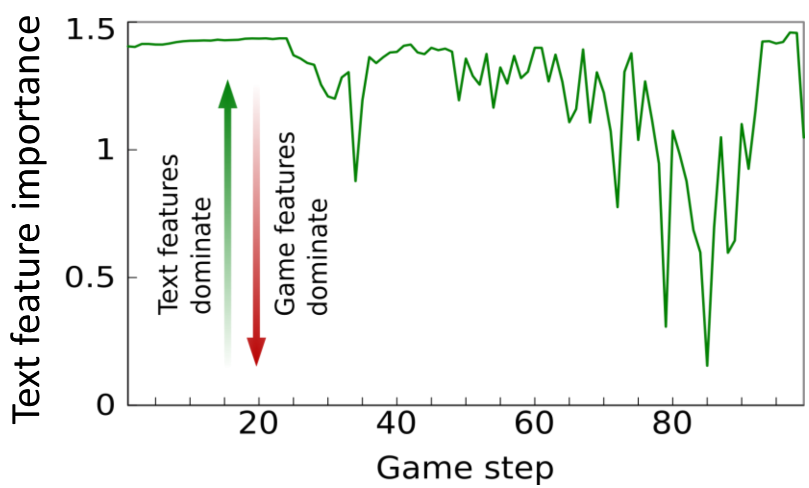
E.g., sentences from the Wall Street Journal corpus.

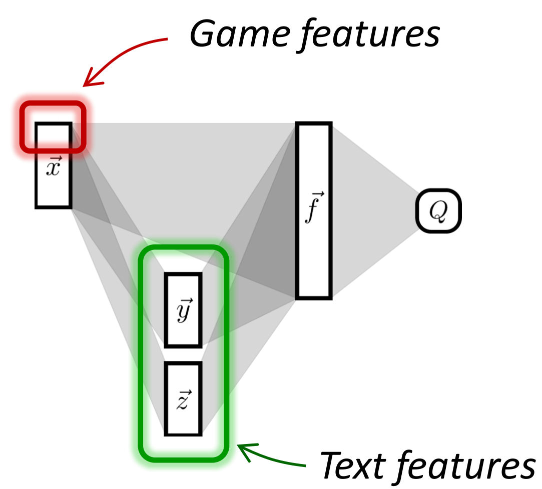
Results: ***71.8%*** *sentence relevance accuracy…*

Surprisingly poor accuracy given game win rate!

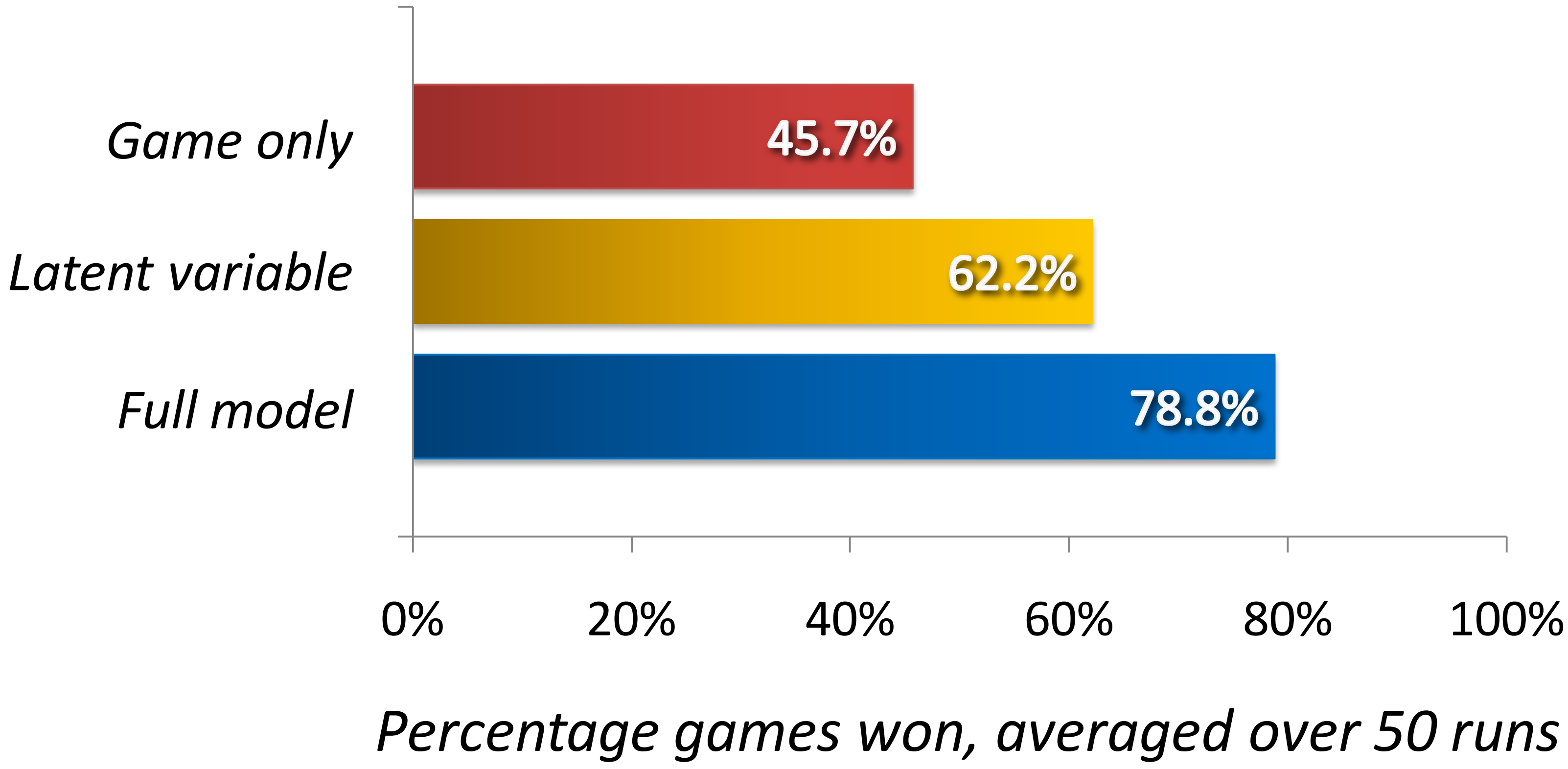
Results: Sentence Relevance







Results: Full Games



Related Work

Grounded Language Acquisition: Instruction Interpretation

Branavan et al. 2009, 2010, Vogel & Jurafsky 2010

- Imperative descriptions of action sequences
- Assume relevance of text to current world state

Language Analysis in Games

Eisenstein et al. 2009

- Extract high-level semantic representation from text
- Learn game rules from labeled traces + extracted formulae

Gorniak & Roy 2005

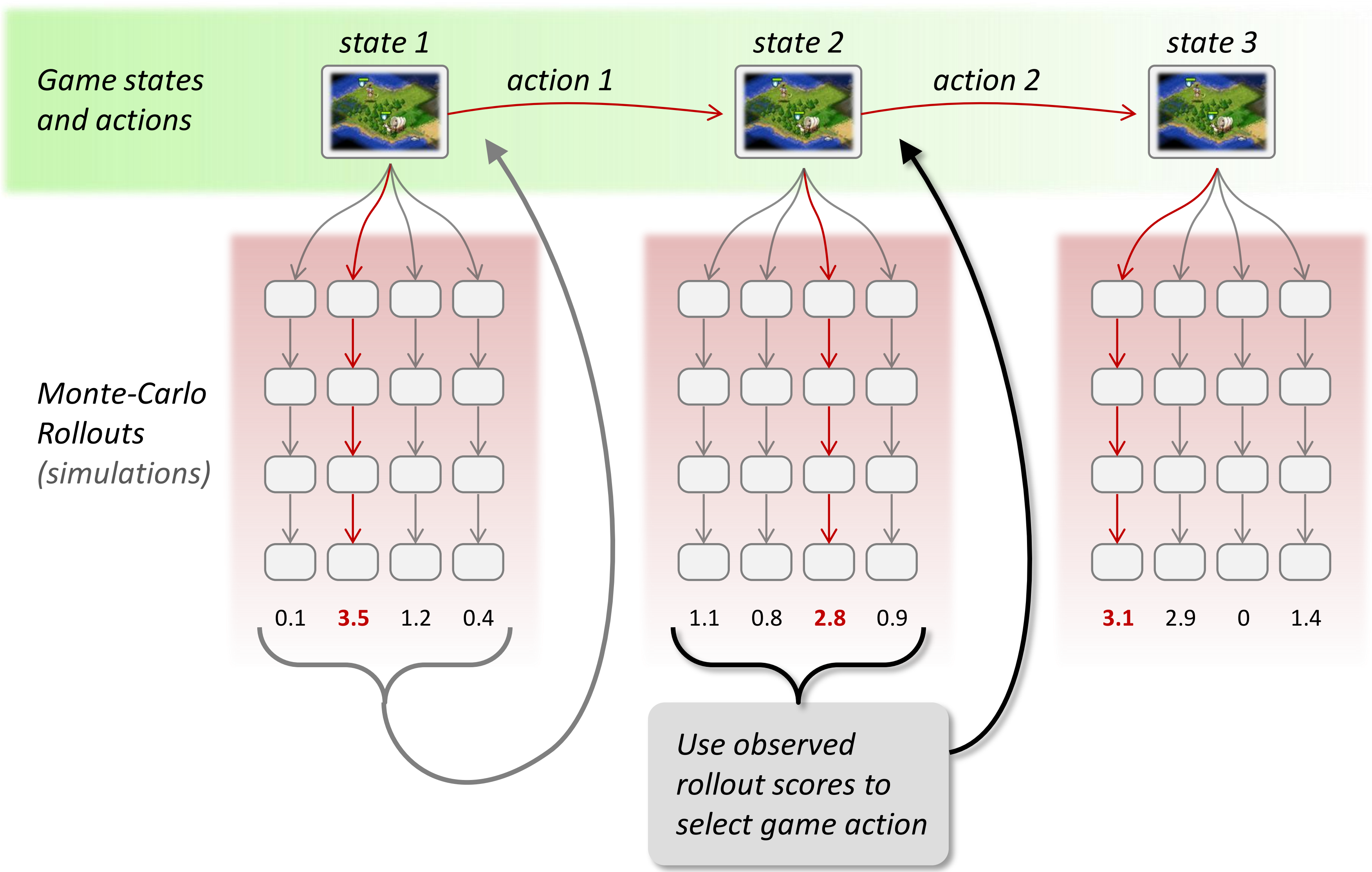
- Interpret spoken commands to control game character
- Learn from labeled parallel corpus

Conclusions

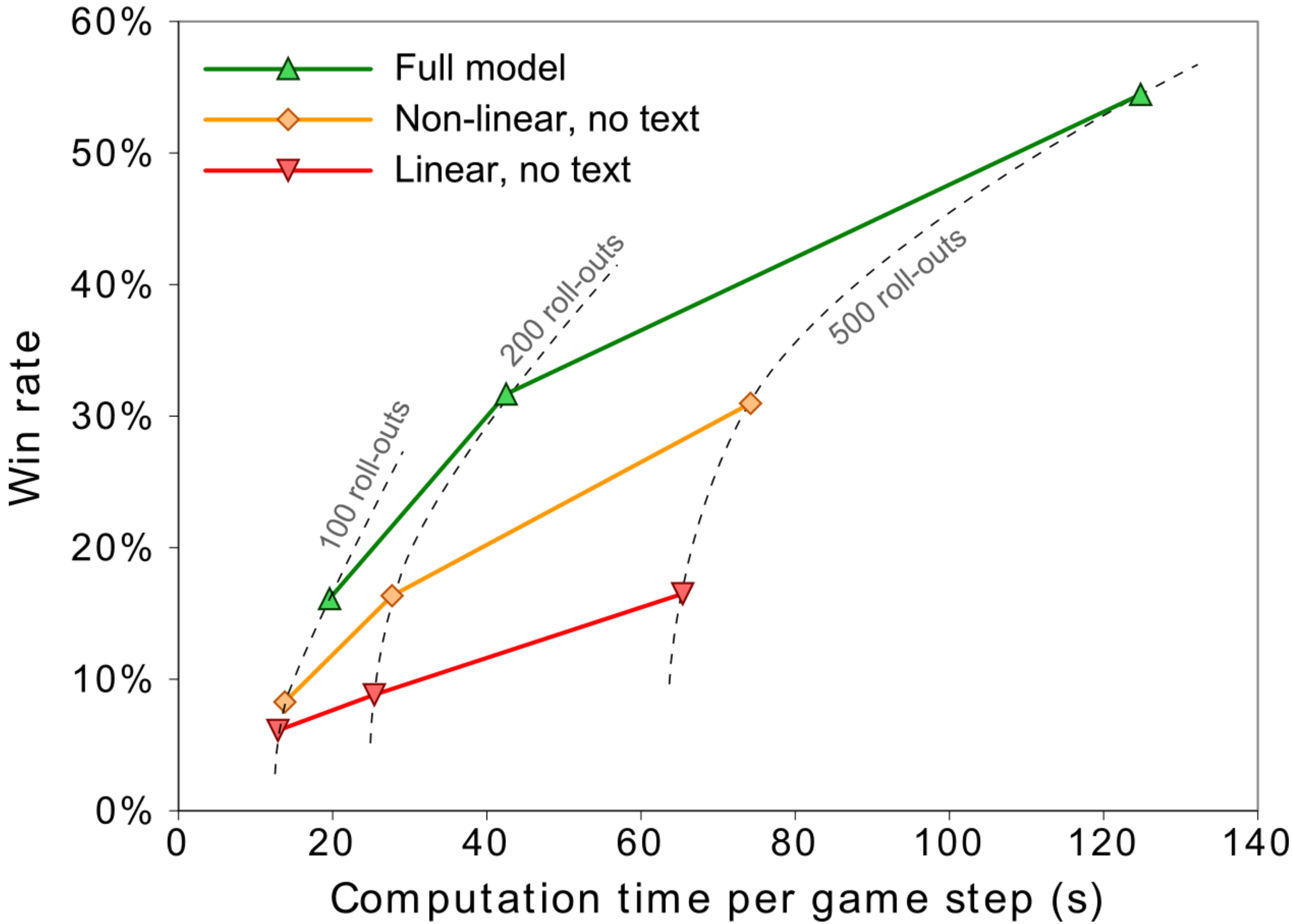
- Human knowledge encoded in natural language can be automatically leveraged to improve control applications.
- Environment feedback is a powerful supervision signal for language analysis.
- Method is applicable to control applications that have an inherent success signal, and can be simulated.

Code, data & experimental framework available at:
http://groups.csail.mit.edu/rbg/code/civ

Monte-Carlo Search: Summary



Model Complexity, Time and Performance



Dependency Information