Reinforcement Learning for Mapping Instructions to Actions

S.R.K. Branavan

Joint work with: Harr Chen, Luke Zettlemoyer, Regina Barzilay

MIT

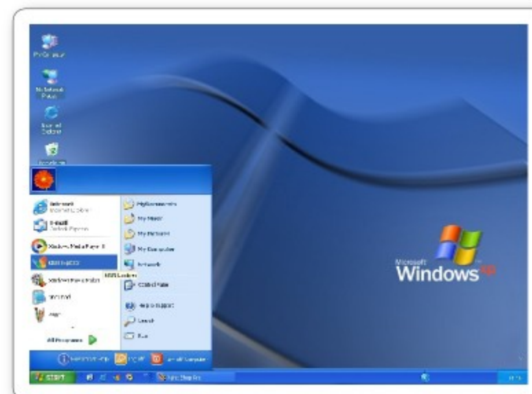
Mapping Instructions to Actions

Input

Instructions:
step-by-step
descriptions of actions

1. Click **Start**, point to **Search**, and then click **For Files or Folders.**
2. In the **Search for** box, type "msdownld.tmp"
3. In the **Look in** list, click **My Computer**, and then click **Search Now**.
4. ...

Target environment:
where actions
need to be executed



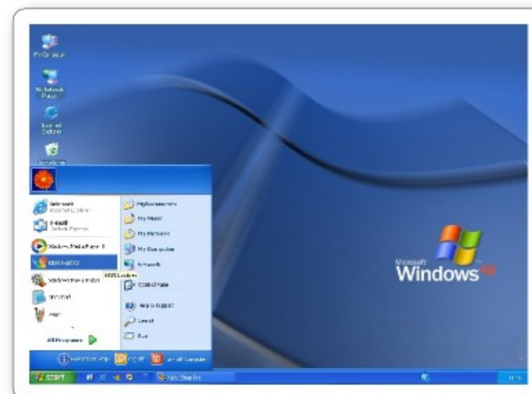
Mapping Instructions to Actions

Input

Instructions:
step-by-step
descriptions of actions

1. Click **Start**, point to **Search**, and then click **For Files or Folders.**
2. In the **Search for** box, type "msdownld.tmp"
3. In the **Look in** list, click **My Computer**, and then click **Search Now**.
4. ...

Target environment:
where actions
need to be executed



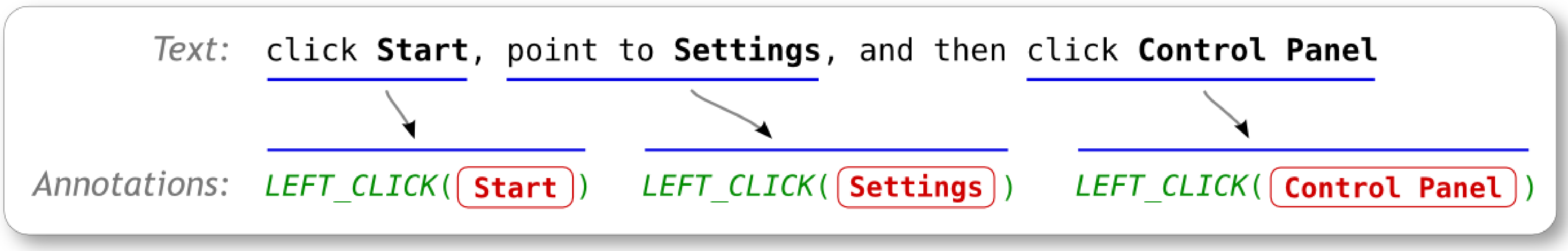
Output

Action sequence
executable in the environment

LEFT_CLICK(**Start**)
LEFT_CLICK(**Search**)
...
TYPE_INTO(**Search for:** , *"msdownld.tmp"*)
...

The Conventional Approach: Supervised Learning

1. Annotate training documents.

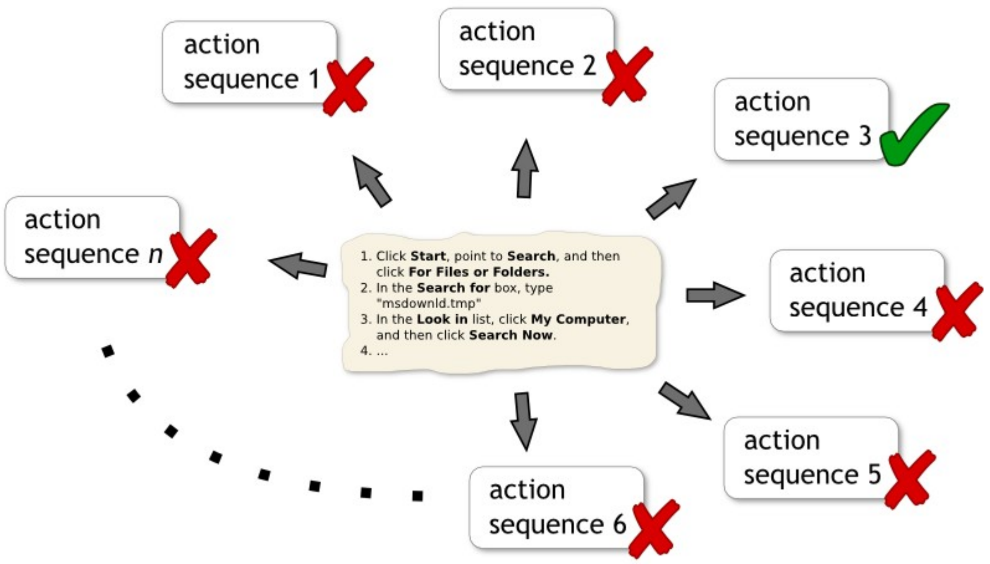


2. Use CRF to learning the mapping.

Our Approach

Learn through trial and error

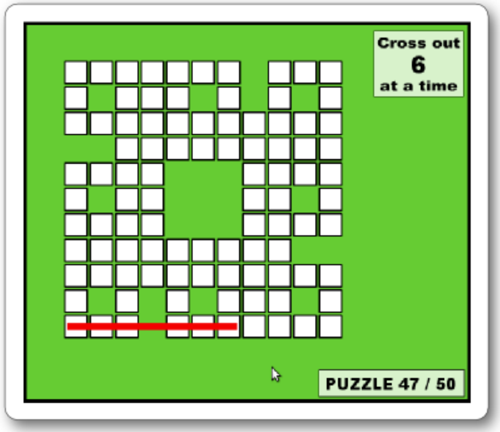
1. *Map instructions to candidate actions*
2. *Execute candidate actions in the environment*
3. *Check how well we do (reward signal)*
4. *Update model parameters based on reward*



Key hypothesis: Reward signal is sufficient supervision

Example Application 1: An Online Puzzle

Target environment
online flash puzzle



Instructions
puzzle solution

Clear the left six from the bottom row, the bottom six from the far-right column, the right six from the top row, the top six from the left column, the second row, the second-to-bottom row, the second from the left column and the second from the right column, the top six from columns three and four, the top two rows of six, the two columns of six, then the two rows.

Reward signal

Check if we won the puzzle !

Example Application 2: Windows Help Instructions

Target environment
Windows 2000
graphical
user interface

Instructions
Microsoft help
document

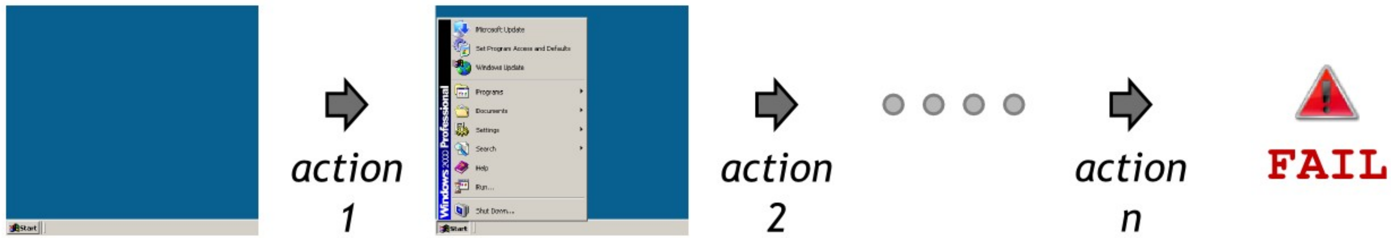
1. Click **Start**, point to **Search**, and then click **For Files or Folders**.
2. In the **Search Results** dialog box, on the **Tools** menu, click **Folder Options**.
3. In the **Folder Options** dialog box, on the **View** tab, under **Advanced Settings**, click **Show hidden files and folders**, and then click to clear the **Hide file extensions for known file types** check box. Click **Apply**, and then click **OK**.
4. In the **Search for files or folders named** box, type **Msdownld.tmp**.
5. In the **Look in** list, click **My Computer**, and then click **Search Now**.
6. In the **Search Results** pane, right-click **Msdownld.tmp** and then click **Delete** on the shortcut menu.
   A "Confirm Folder Delete" message appears.
7. Click **Yes**.

Reward signal

Check if we hit a dead-end
(check for overlap between
sentence words & GUI labels)

Learning Using Reward Signal: Challenges

1. Reward can be delayed



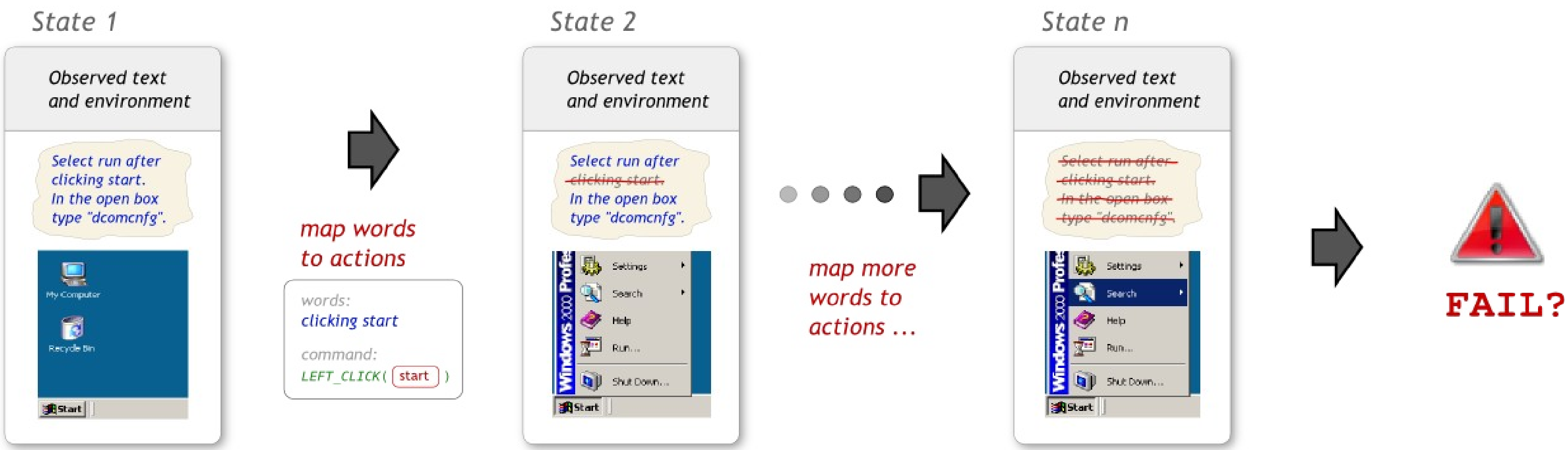
⇒ *How can reward be propagated to individual actions*

2. Number of candidate action sequences is very large

⇒ *How can this space be effectively searched?*

Use Reinforcement Learning

Reinforcement Learning: A Sketch



- *Repeat:*
  - *Observe current state of text + environment*
  - *Select action based on a probabilistic model*
  - *Execute action*
- *Receive reward and update parameters*

Reinforcement Learning: Representation

State s = Observed Text + Observed Environment

Action a = Word Selection + Environment Command

State 1

Observed text and environment

Select run after clicking start. In the open box type "dcomcnfg".

Action 1

words:
clicking start

command:
LEFT_CLICK(start *)*

Policy function

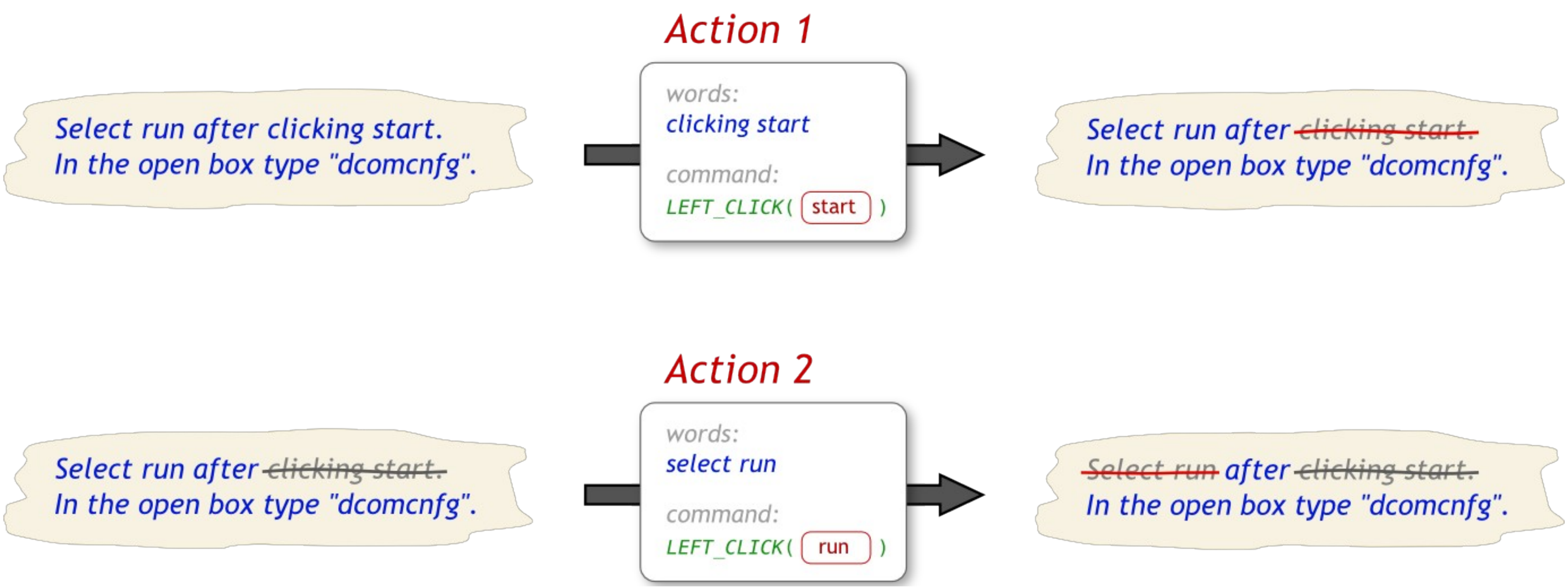
$$p(\, a \mid s \,)$$

State 2

Observed text and environment

Select run after ~~clicking start.~~ In the open box type "dcomcnfg".

Constructing Mappings



Constructing Mappings

Mapping process allows us to:

- *Segment text to chunks that describe individual commands*
- *Learn translation of words to environment commands*
- *Reorder environment commands*

Select run *after* *clicking start,* *then* *in the open box, type "dcomcnfg".*

Constructing Mappings

Mapping process allows us to:

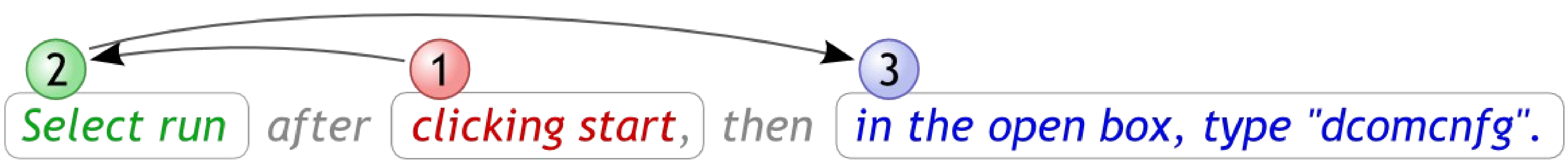
- *Segment text to chunks that describe individual commands*
- *Learn translation of words to environment commands*
- *Reorder environment commands*

"select run" ➡ *LEFT_CLICK* $\boxed{\text{run}}$

Constructing Mappings

Mapping process allows us to:

- *Segment text to chunks that describe individual commands*
- *Learn translation of words to environment commands*
- *Reorder environment commands*

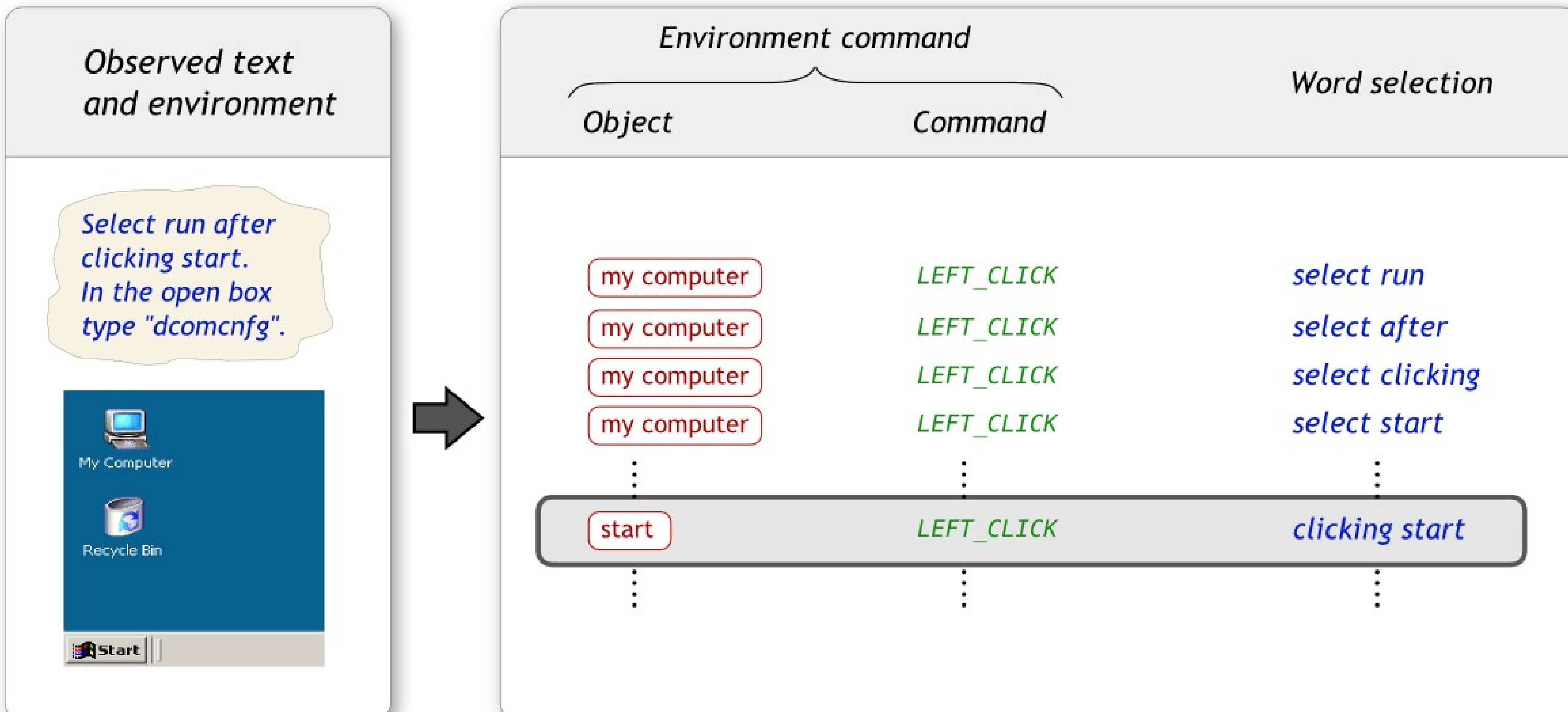


Generating Possible Actions

State s = Observed Text + Observed Environment

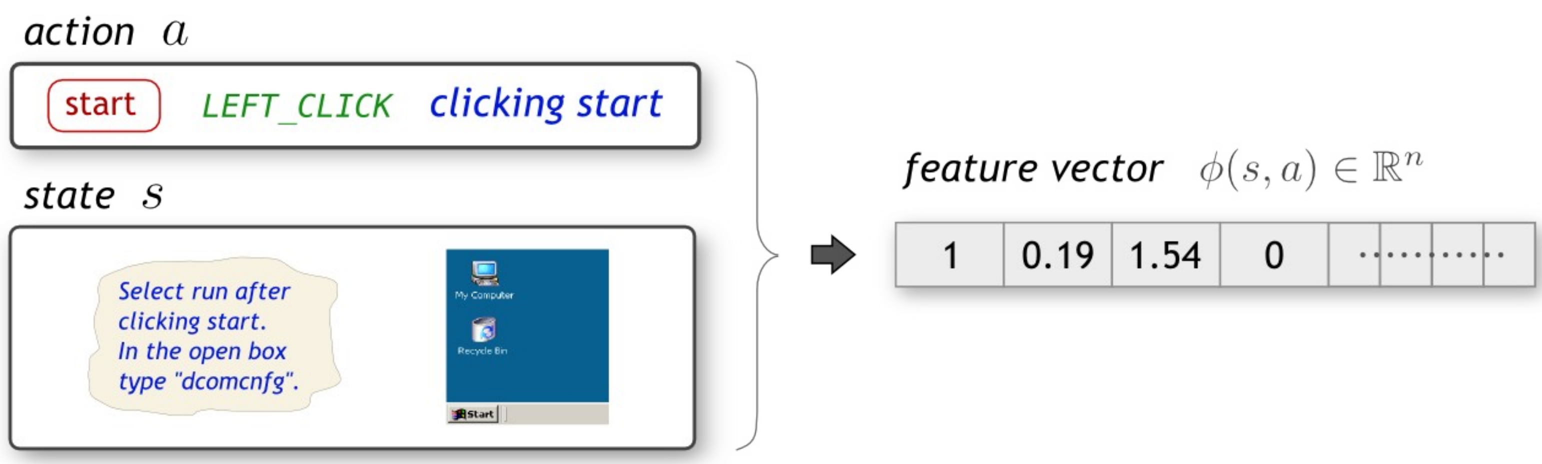
Action a = Word Selection + Environment Command

State *Possible actions*



Model Parameterization

Represent each action with a feature vector:



$\phi(s,a) \in \mathbb{R}^n$ - *real valued feature function on state s and action a*

Define policy function as a log-linear distribution:

$$p(a \mid s; \theta) = \frac{e^{\theta \cdot \phi(s,a)}}{\sum_{a'} e^{\theta \cdot \phi(s,a')}}$$

θ - *parameters of model*

Example Features

Features on words and environment command

Edit distance between word and object label

Binary feature on each (*word*, *command*) pair

Binary feature on each (*word*, *object type*) pair

Features on environment objects

Object is visible

Object is in foreground

Object was previously interacted with

Object became visible after last action

Features on words

Word type

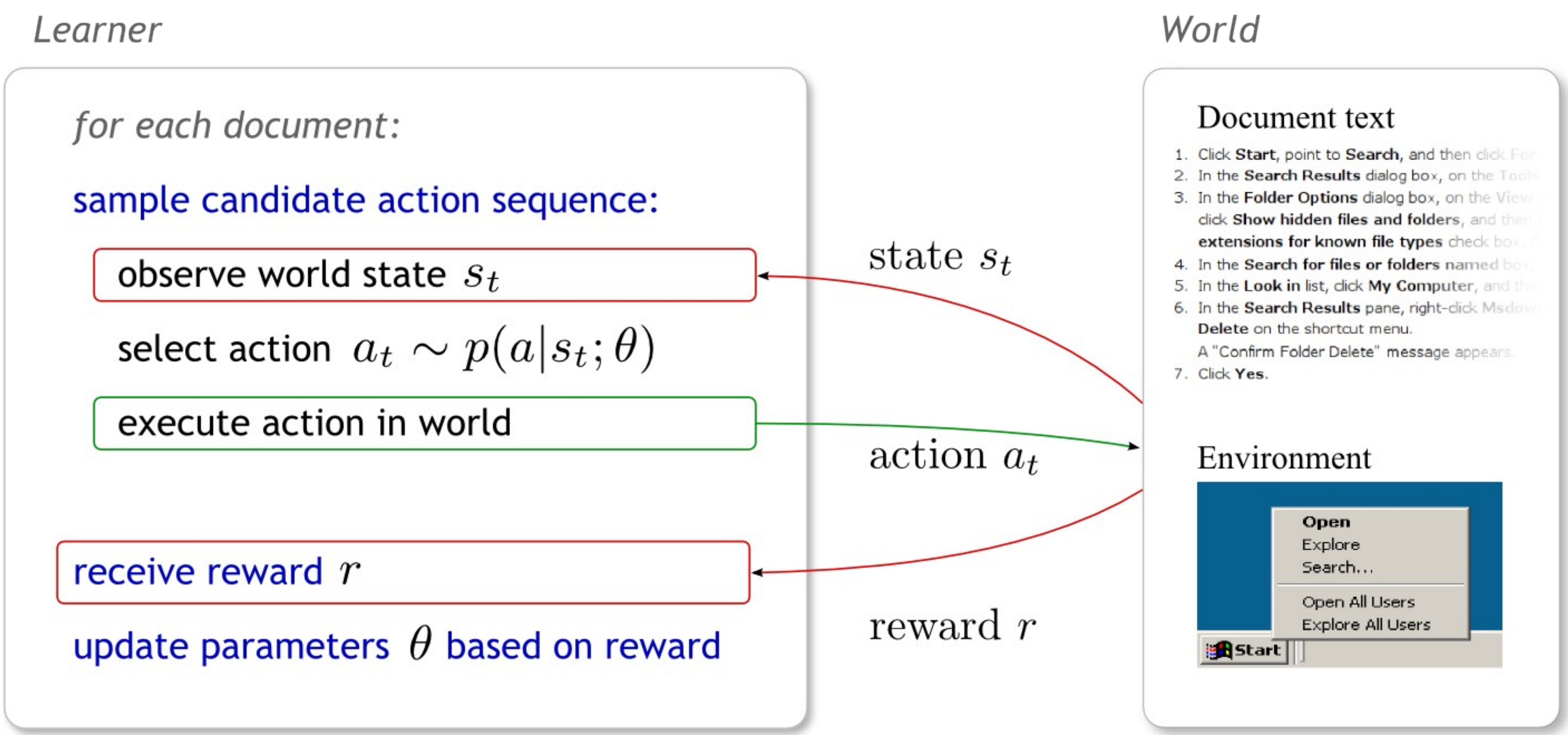
Distance from last used word

Total number of features: 4438

Learning Algorithm

Goal: *Find θ that maximizes the expected reward*

Method: *Policy gradient algorithm (stochastic gradient ascent on θ)*



Learning Algorithm

Parameter update:

$$\theta \leftarrow \theta + r \sum_t \underbrace{\left[\phi(s_t, a_t) - \sum_a \phi(s_t, a) \; p(a \mid s_t; \theta) \right]}_{\textit{Gradient of log-linear model}}$$

r – reward
s_t – state at time t
a_t – action taken at time t

Incorporating Annotation in Reinforcement Learning

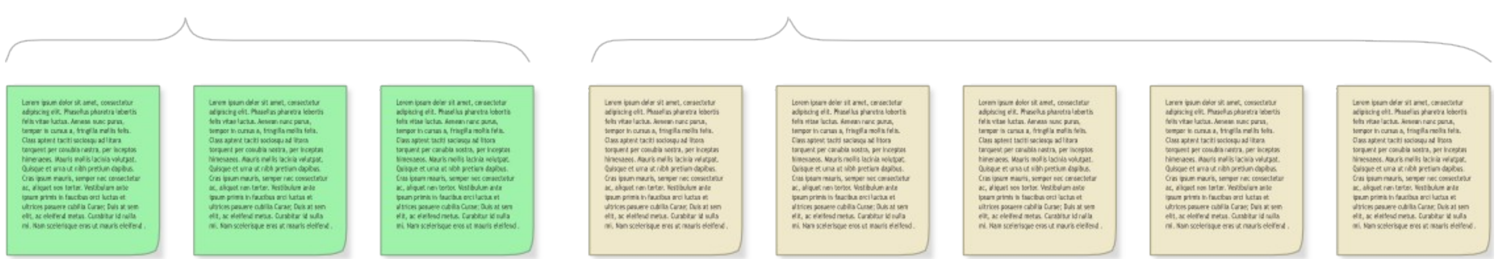
Reward can be based on annotations if available

$$\text{Reward } r(h) = \begin{cases} +1 & \text{if actions \textbf{match} annotations} \\ 0 & \text{if actions \textbf{don't match} annotations} \end{cases}$$

Reinforcement learning allows a mix of annotation and environment based reward signals

Annotated
Use annotation reward

Unannotated
Use environment reward



Incorporating Annotation in Reinforcement Learning

Reward can be based on annotations if available

$$\text{Reward } r(h) = \begin{cases} +1 & \text{if actions match annotations} \\ 0 & \text{if actions don't match annotations} \end{cases}$$

If all documents are annotated: Equivalent to stochastic gradient ascent with a maximum-likelihood objective

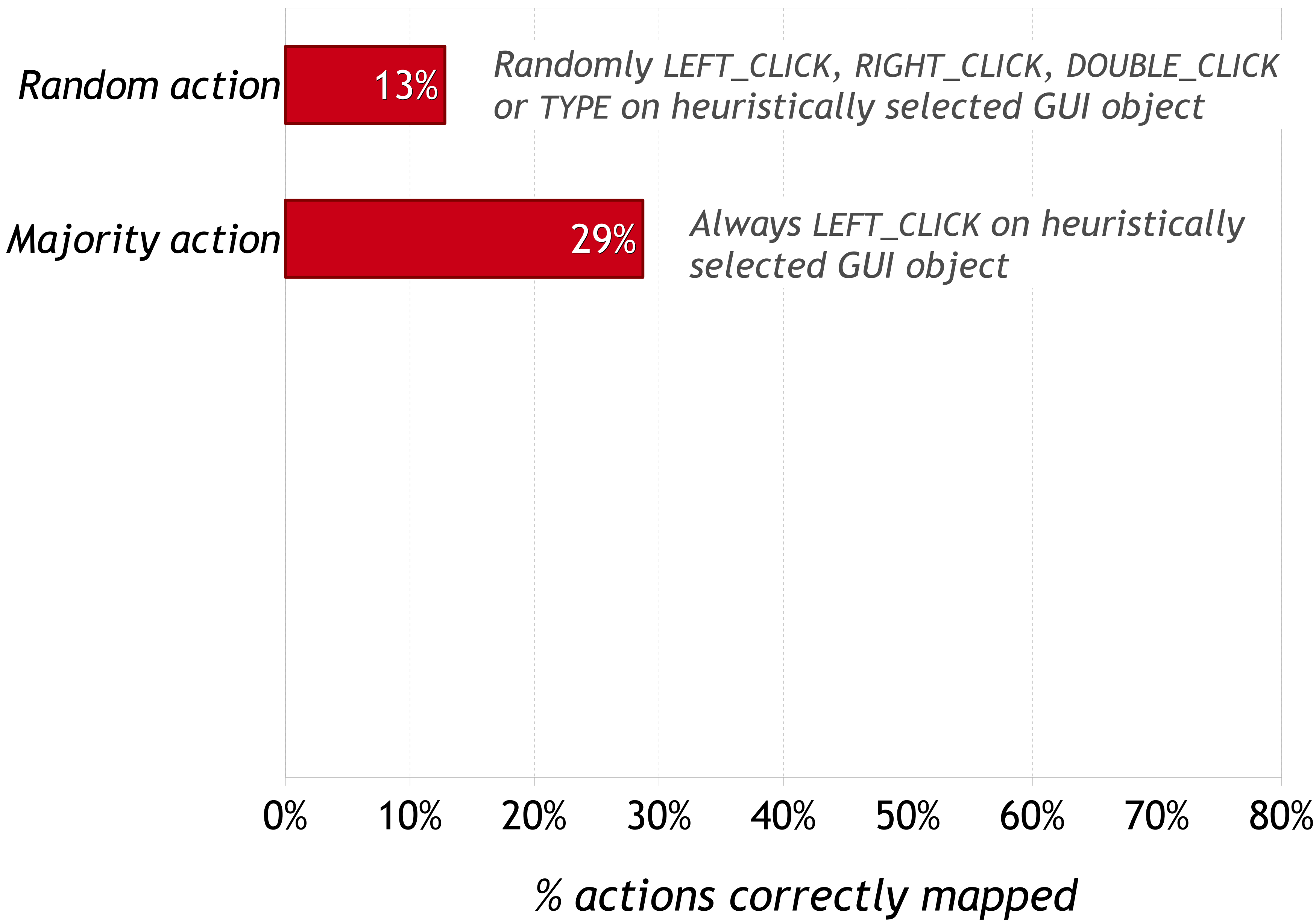
Windows Configuration Application

Windows 2000 help documents
from *support.microsoft.com*

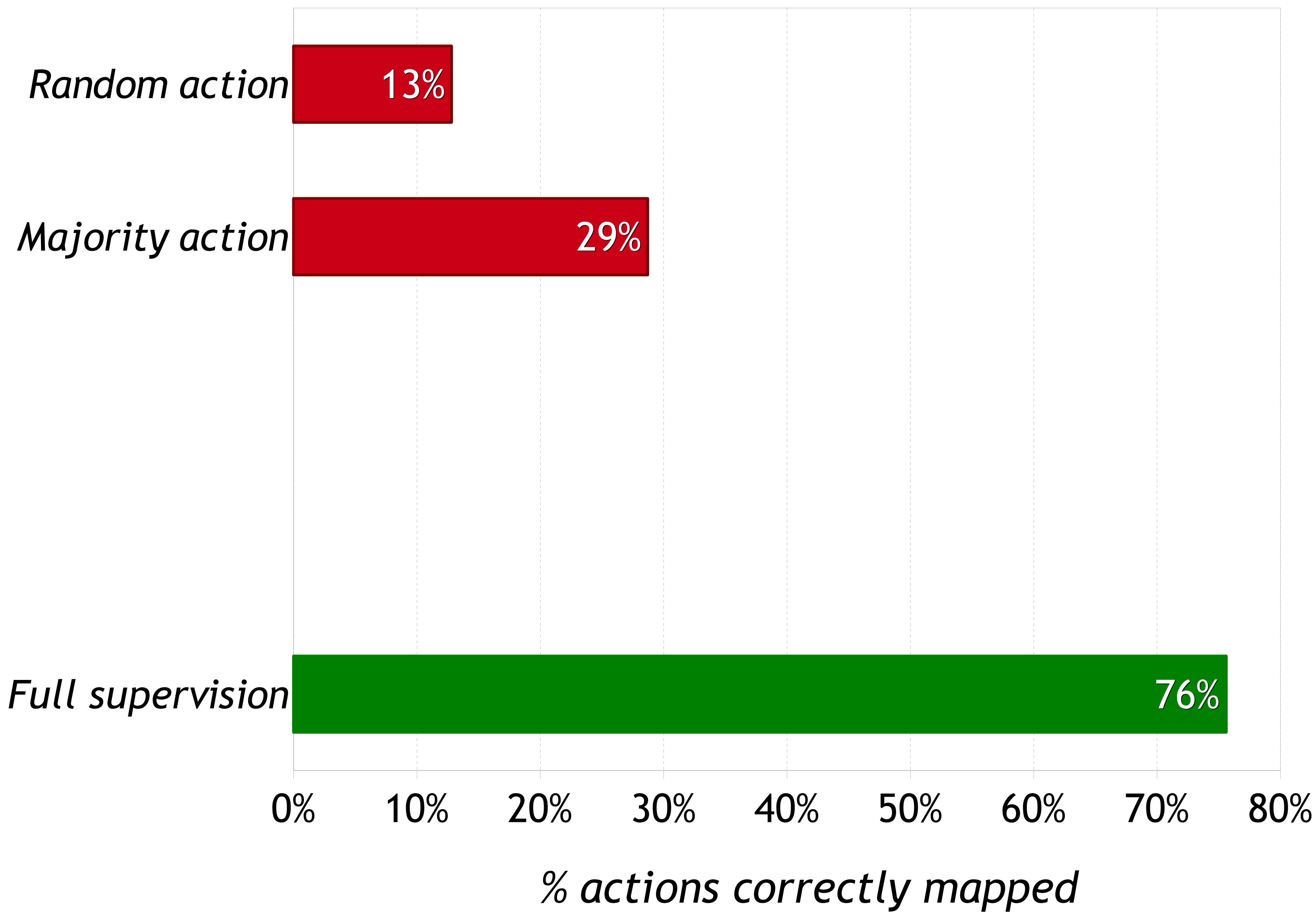
Total # of documents	128
Train/development/test	70 / 18 / 40
Total # of words	5562
Vocabulary size	610
Avg. words per sentence	9.93
Avg. sentences per document	4.38
Avg. actions per document	10.37

Complex environment: 13088 observed states

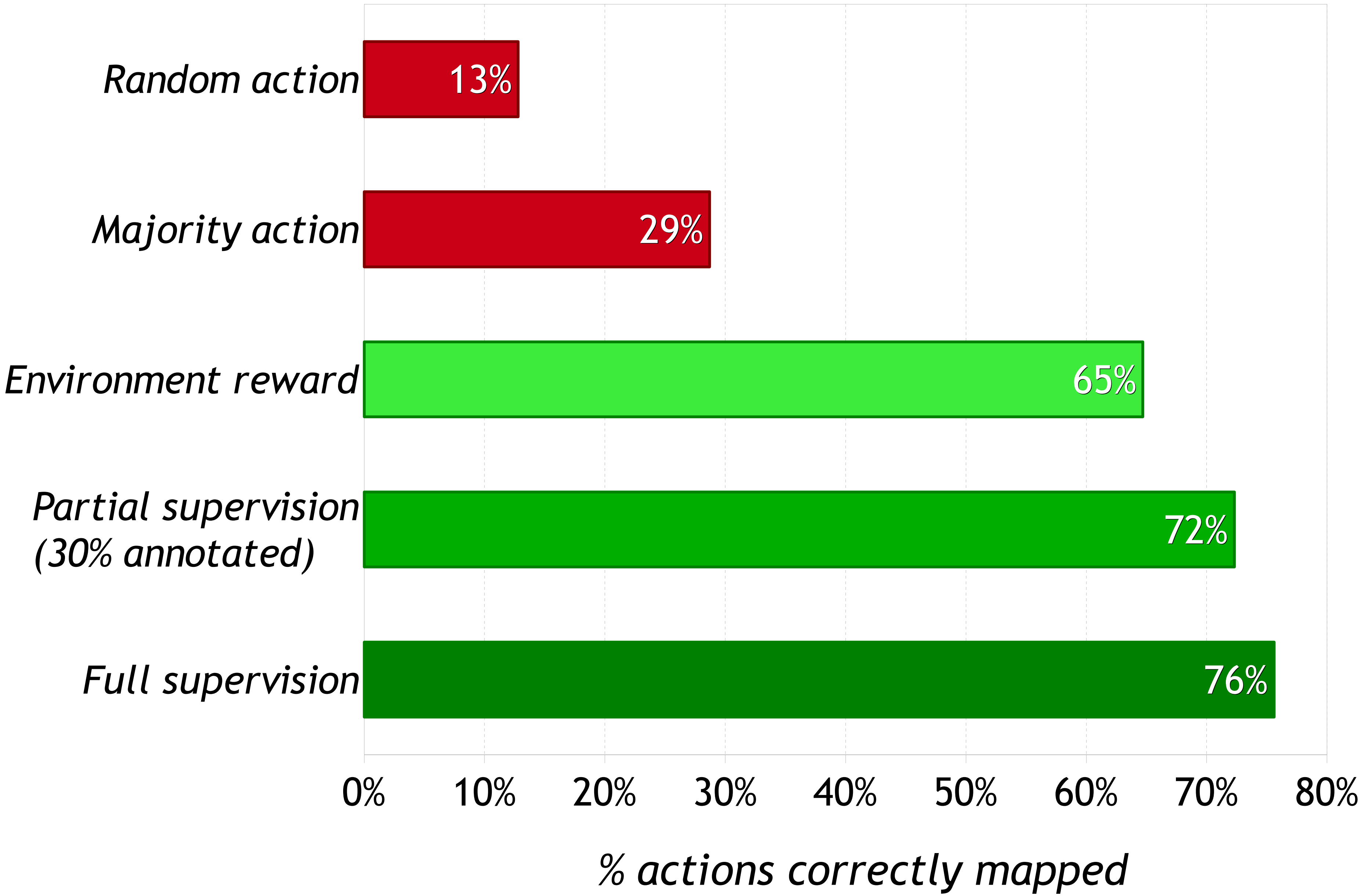
Results: Baselines



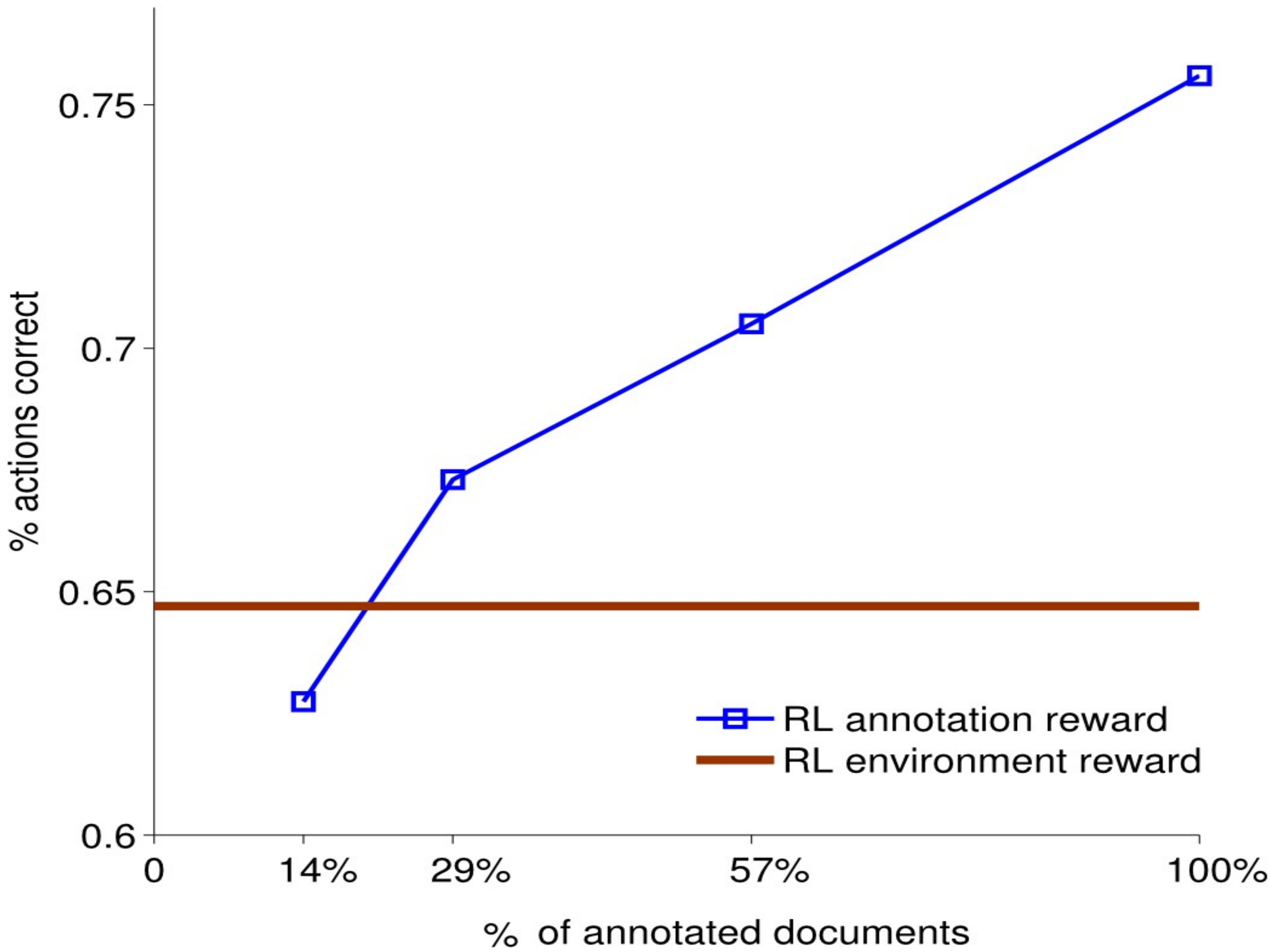
Results: Supervised



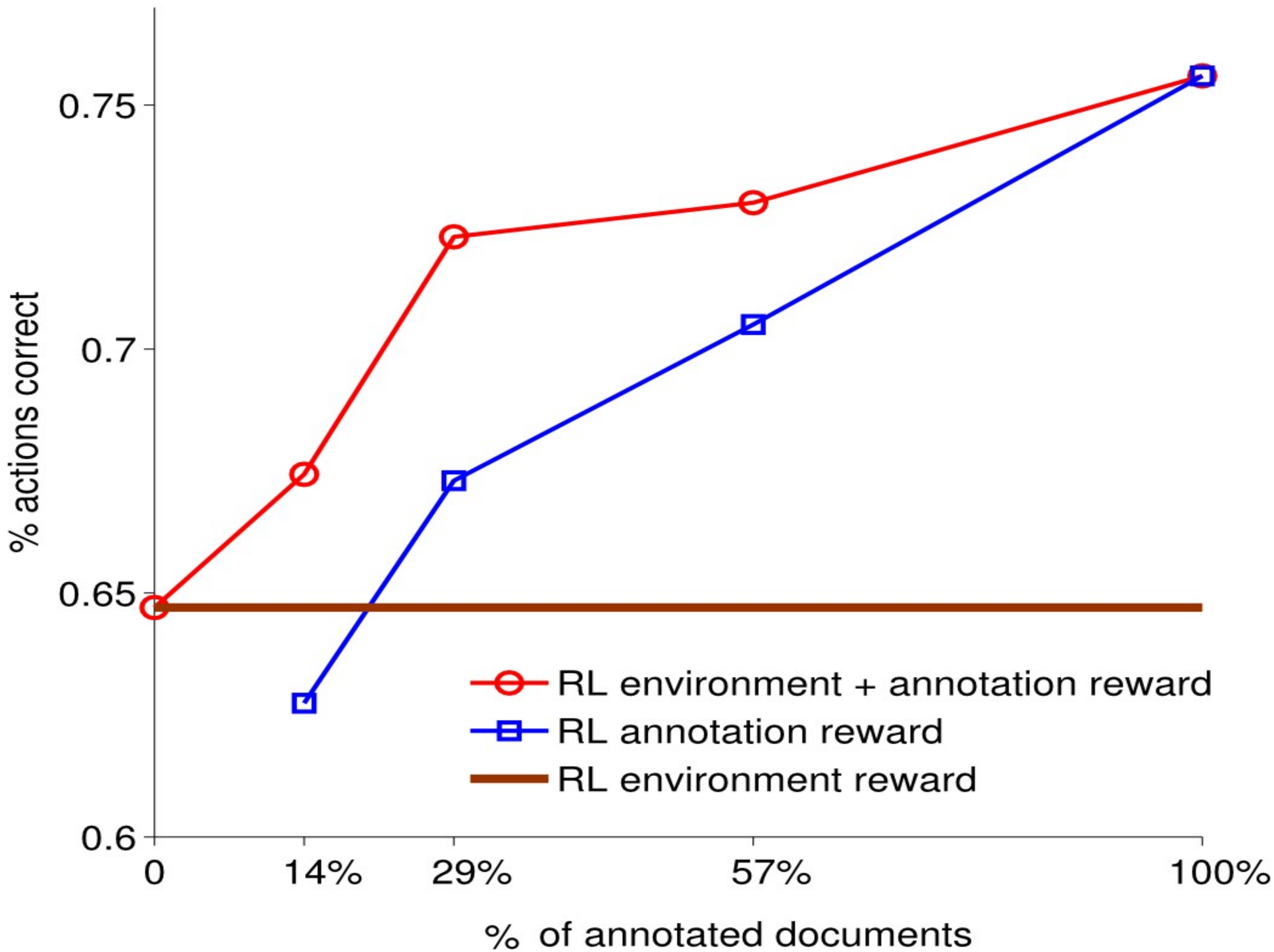
Results


Random action
13%
Majority action
29%
Environment reward
65%
Partial supervision
(30% annotated)
72%
Full supervision
76%
0%
10%
20%
30%
40%
50%
60%
70%
80%
% actions correctly mapped

Trade off between
Environment Reward and Manual Annotations



Trade off between Environment Reward and Manual Annotations



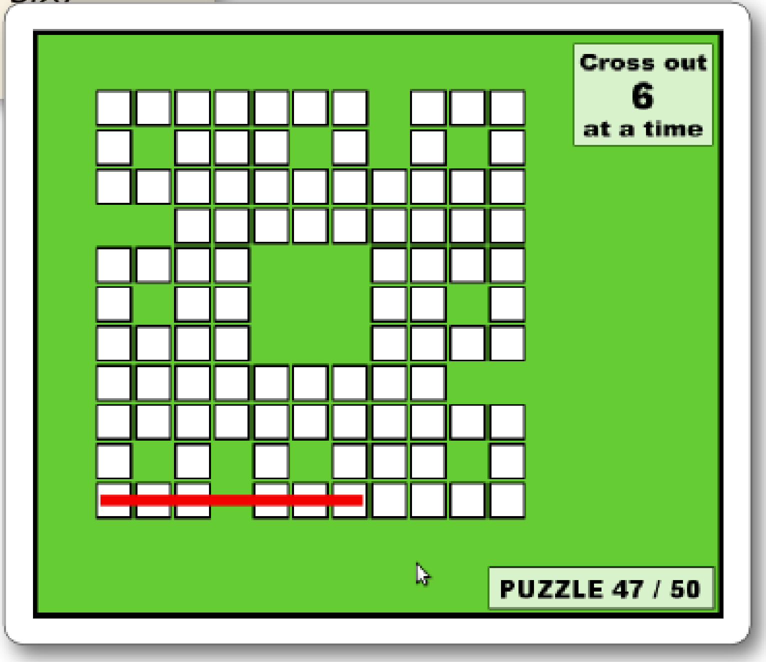
Puzzle Application

Walk-through documents from the *Crossblock* flash puzzle

Clear the left six from the bottom row, the bottom six from the far-right column, the right six from the top row, the top six from the left column, the second row, the second-to-bottom row, the second from the left column and the second from the right column, the top six from columns three and four, the top two rows of six, the two columns of six, then the two rows.

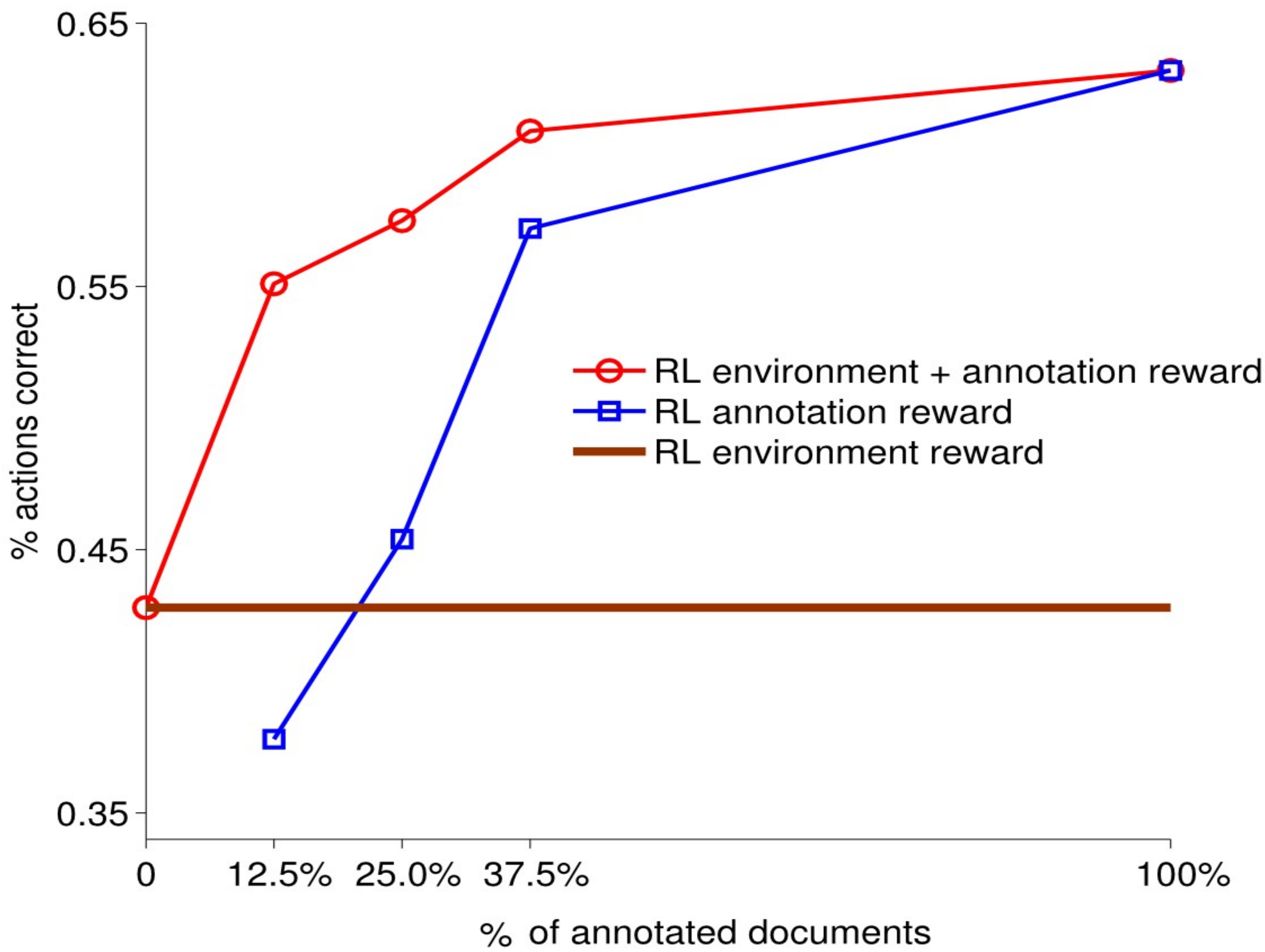
Instructions

Target environment

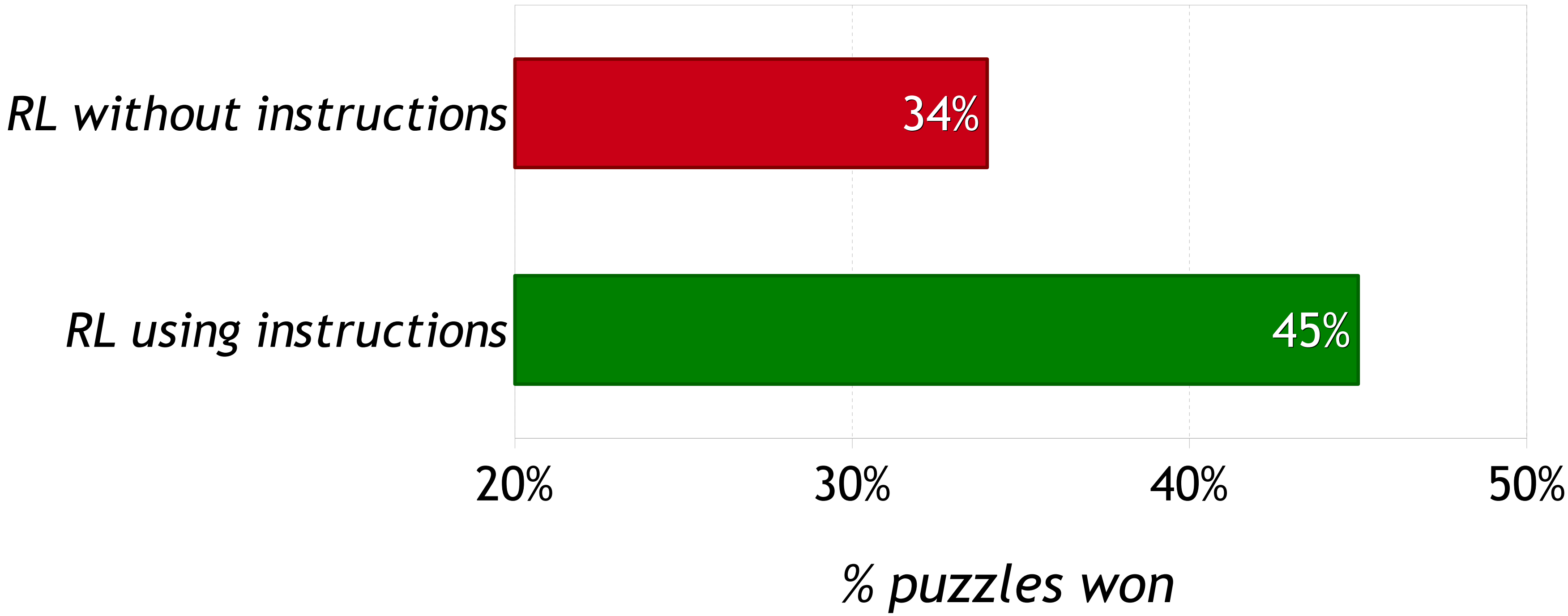


http://hexaditidom.deviantart.com/art/Crossblock-108669149

Results: Puzzle Game Application



Results: Puzzle Game Application



Our method can leverage knowledge encoded in natural language

Related Work

Reinforcement Learning for Dialogue Management:

Scheffler and Young (2002), Roy et al. (2000),
Litman et al. (2000), Singh et al. (1999)

Fundamentally different problems

Grounded Language Acquisition:

Chen and Mooney (2008), Roy and Pentland (2002),
Siskind (2001), Barnard and Forsyth (2001), Oates (2001)

Assume parallel corpus of text and semantic representations (e.g. database entries)

Conclusions

- *Environment feedback is an effective source of supervision*
  - *Reduces need for manual annotations*
- *Our method can leverage knowledge encoded in natural language*



Code and data available at:

`groups.csail.mit.edu/rgb/code/rl`

Results

	Windows				Puzzle		
	Action	Sent.	Doc.	Word	Action	Doc.	Word
Random baseline	0.128	0.101	0.000	——	0.081	0.111	——
Majority baseline	0.287	0.197	0.100	——	——	——	——
Environment reward	$*$ 0.647	$*$ 0.590	$*$ 0.375	0.819	$*$ 0.428	$*$ 0.453	0.686
Partial supervision	$\diamond$ 0.723	$*$ 0.702	0.475	0.989	0.575	$*$ 0.523	0.850
Full supervision	$\diamond$ 0.756	0.714	0.525	0.991	0.632	0.630	0.869