Environment-Driven Lexicon Induction for High-Level Instructions

By Dipendra K. Misra, Kejia Tao, Percy Liang, and Ashutosh Saxena

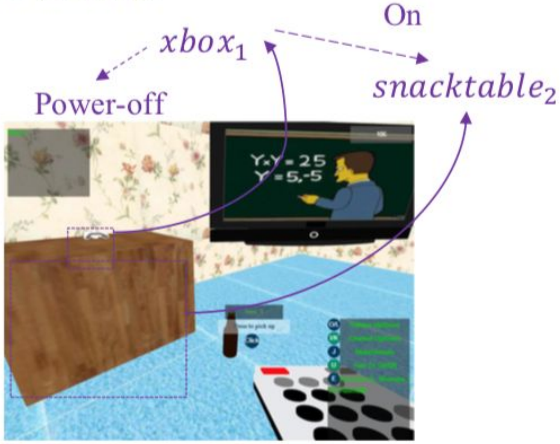
Presentation by Rishub Jain

Problem Statement

- Given an environment and text, predict a set of actions the text dictates

Text: *"Turn on xbox. Take Far Cry Game CD and put in xbox. Throw out beer, coke and sketchy stuff in bowl.⋯ "*

Action Sequence: $moveto(xbox_1); grasp(xbox_1); press(power\_button_1);$
$moveto(cd_2); grasp(cd_2); insert(cd_2, xbox_1)\ \cdots$



Environment:

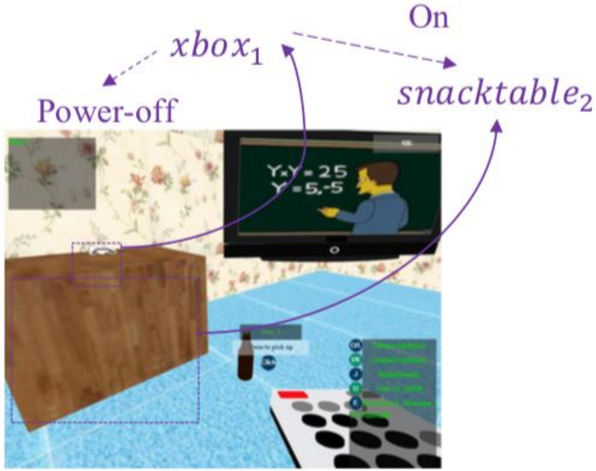
Previous work

- Previous work was always missing one of the following:
  - Able to correctly handle new actions in the test set
  - Able to handle complex actions (in a reasonable amount of time)
    - Microwaving a cup requires 10-15 sub-actions

- This work tries to do all of these things

Environment

- Represented as a graph
- Each vertex is an object, and has:
  - Instance ID (e.g. $xbox_1$)
  - Category name (e.g. xbox)
  - Set of properties (e.g. graspable)
  - Set of binary states (e.g. power-off)
- Each edge is a relationship between two objects
  - five basic spatial relations: near, grasping, on, in and below



Environment:

Actions

- Each action name in the sequence is one of 15 values (grasp, moveto, wait, etc.)
- Each action can contain an object ($xbox_1$), a spatial relation (keep($ramen_2$, in, $kettle_1$)), or a postcondition (wait(state($kettle_1$, boiling)))

Action Sequence: $moveto(xbox_1); grasp(xbox_1); press(power\_button_1);$
$moveto(cd_2); grasp(cd_2); insert(cd_2, xbox_1)\ \cdots$

Postconditions

- Instead of trying to predict actions, we predict post conditions, and infer actions
- Postcondition: A conjunction of atoms
- An atom can be:
  - A spatial relation (on($book_9$, $shelf_3$))
  - A state and value (state($kettle_1$, boiling))
- Represented as a logical form:
  - Each logic form has a set of parameterized post conditions, and a mapping from variables to objects

Logical Form $z = (\ell, \xi)$

$$\ell : put \Rightarrow [\lambda\vec{v}.\,\mathtt{state}(\mathtt{v}_1, \mathtt{has\text{-}cd}) \wedge \mathtt{near}(\mathtt{v}_1, \mathtt{v}_2), \xi']$$

$$\xi : \{v_1 \rightarrow xbox_1; v_2 \rightarrow robot_1\}$$

ξ': old mapping

Why use postconditions?

- They generalize better
  - To fill a cup with water, postcondition = “cup is full”, while action = “fill cup using tap”. During testing, you may fill the cup using a bucket
- Much less number of atoms to represent complex task
  - Microwaving requires 10-15 actions, but just 2 atoms in its postcondition: in(cup_2, $microwave_1$) $\wedge$ state(microwave, is-on)

Approach

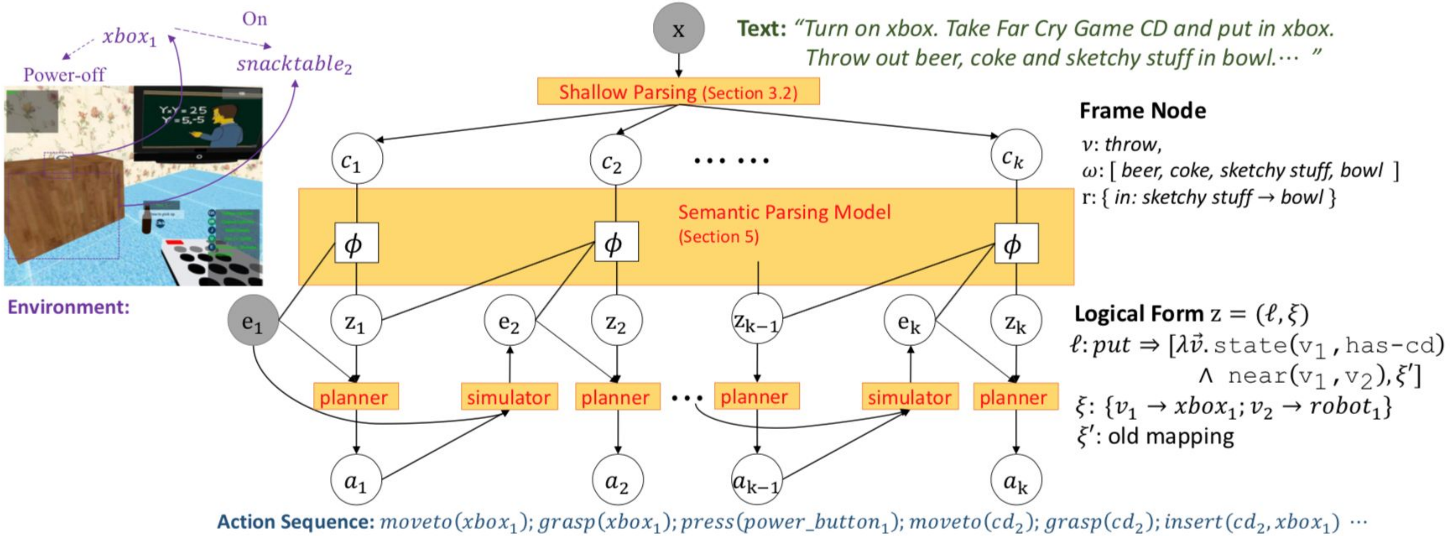
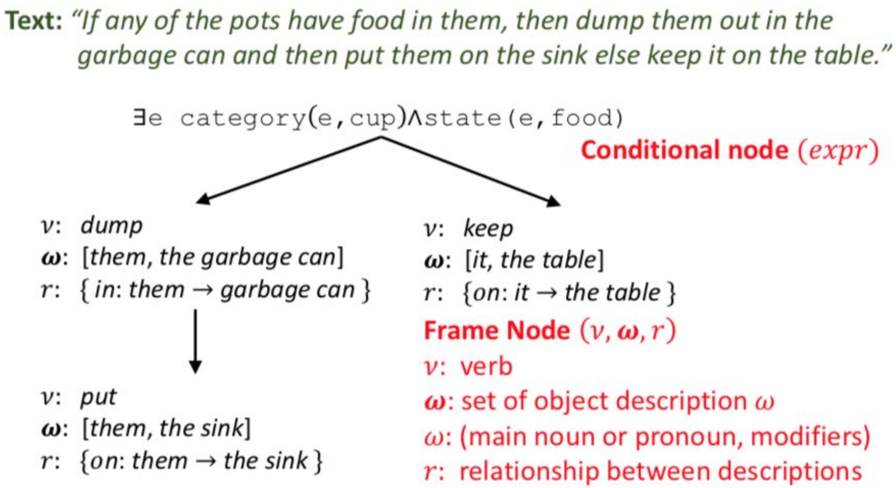


Figure 2: Graphical model overview: we first deterministically shallow parse the text x into a control flow graph consisting of shallow structures $\{c_i\}$. Given an initial environment e_1, our semantic parsing model maps these frame nodes to logical forms $\{z_i\}$ representing the postconditions. From this, a planner and simulator generate the action sequences $\{a_i\}$ and resulting environments $\{e_i\}$.

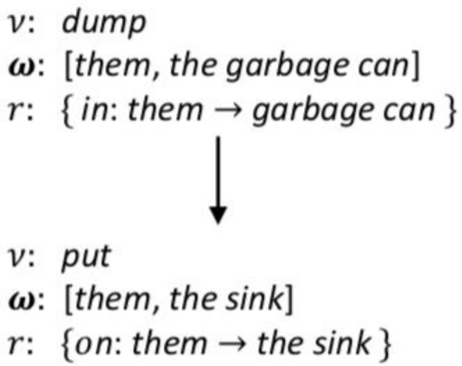
Shallow Parsing

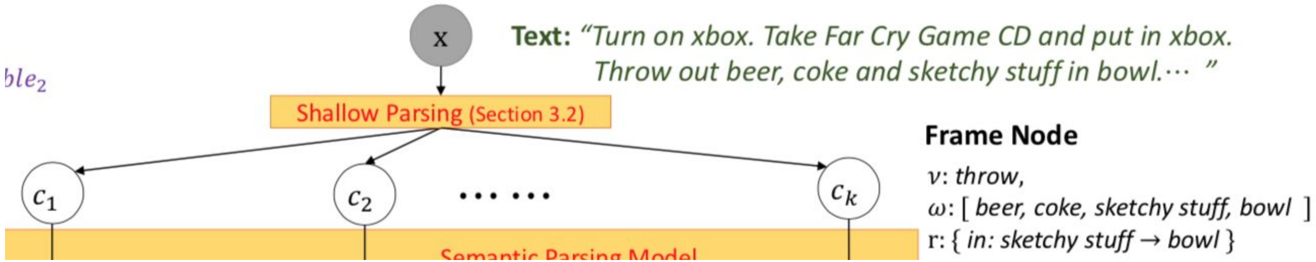
- Deterministically parse text into "Control Flow Graph"
- Frame node:
  - Verb
  - Object descriptions
  - Spatial relationships
- Conditional node:
  - Branching: two children separated by condition
  - Temporal: "until" statement
- Based on manual rules and the Stanford parser



Shallow Parsing

- Given environment, resolve all conditional branches, and return a sequence of frame nodes

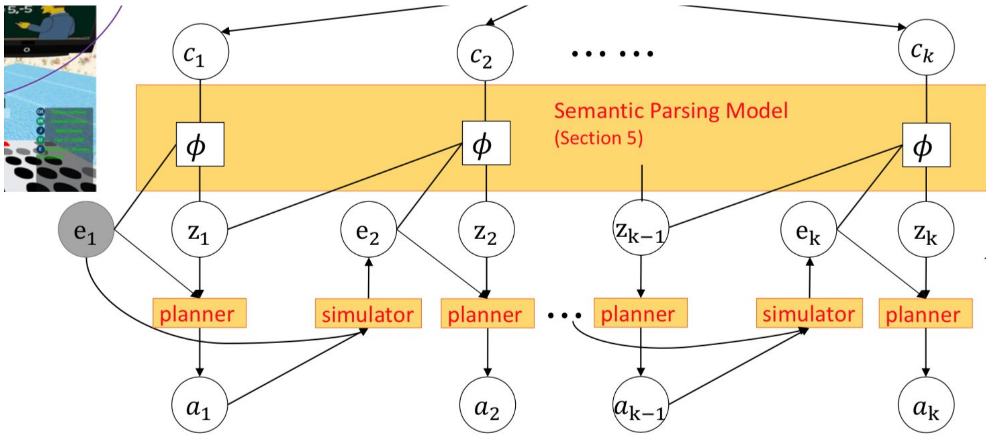




Semantic Parsing Model

- Given a sequence of frame nodes $c_{1:k}$ and initial environment e_1, define a joint distribution over all logical forms $z_{1:k}$ using a conditional random field (CRF)
- e_{i+1} = simulator(e_i, planner(e_i, z_i))

$$p_\theta(z_{1:k} \mid c_{1:k}, e_1) \propto \exp\left(\sum_{i=1}^{k} \phi(c_i, z_{i-1}, z_i, e_i) \cdot \theta\right)$$



Semantic Parsing Model - Feature Vector

- Given object descriptions (Far Cry Game CD, xbox) determine probability of objects they are referring to
  - Not just a text matching problem
    - For example, multiple CD's
    - "Get me a tank of water" requires you to use a "cup" object
  - Uses a combination of rules to determine this (Wordnet similarity, category matching, etc.)
- In z_i (grasping(robot, couch)), given the training distribution of postconditions, determine probability that it is reasonable
- Other less important features involving transition probabilities between z_{i-1} and z_i, etc.

$$\phi(c_i, z_{i-1}, z_i, e_i)$$

Lexicon Induction

- Remember: Logical form has parameterized postconditions and object mapping
- To get parametrized mapping, if the verb appears in the training set, you can just find the most likely mapping with:

Logical Form $\mathrm{z} = (\ell, \xi)$

$\ell: put \Rightarrow [\lambda\vec{v}.\,\texttt{state}(\texttt{v}_1,\texttt{has-cd})$
$\quad \wedge\ \texttt{near}(\texttt{v}_1,\texttt{v}_2), \xi']$

$\xi: \{v_1 \rightarrow xbox_1; v_2 \rightarrow robot_1\}$

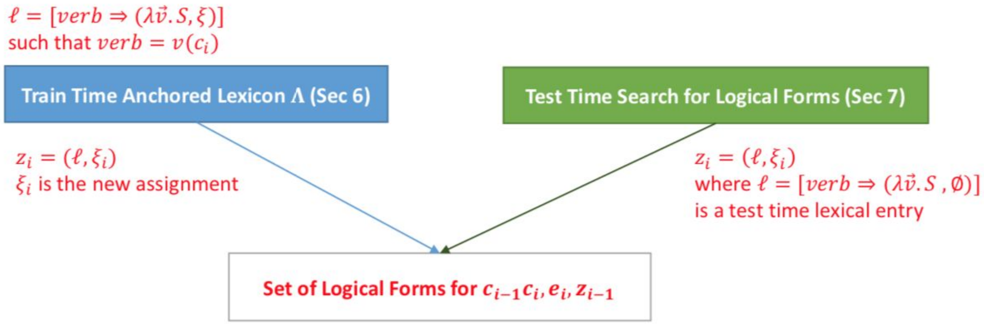
ξ': old mapping

$$\xi_i \;=\; \arg\max_{\xi'} \phi(c_i, z_{i-1}, (\ell_i, \xi'), e_i)$$

- Becomes an approximately quadratic programming problem

Environment-Driven Lexicon Induction

- If verb does not appear in training set, you cannot do regular lexicon induction
- (Using the approach from the Semantic Parsing Model), for each object description, select only the object with the highest probability that the description is referring to it
- To assign the objects to the variables, my guess: they do a beam search on the combinations of object assignments, since there are usually 1-4 variables



Inference and Parameter Estimation

- Train CRF to find the parameters θ

$$\tilde{p}_\theta(z_i \mid z_{i-1}, c_i, e_i) \propto \exp(\phi(c_i, z_{i-1}, z_i, e_i)^\top \theta).$$

- Starting with the k most likely values for z_1, conduct a beam-search to find the resulting $z_{1:k}$, and then deterministically find $a_{1:k}$ using the deterministic planner: a_i = planner(e_i, z_i)

Dataset

- Created their own dataset by crowd-sourcing

- 20 3D environments had 40 objects on average
- 10 total high-level objectives (clean the room, etc.), 5 per scenario

- Asked one group of users to write the Text describing what to do
- Another group wrote the actual actions of the robot
- Total of 500 examples (469 after filtering)
- 148 different verbs, an average of 48.7 words per text, and an average of 21.5 actions per action sequence

Evaluation

- 2 metrics:
  - IED: Edit distance from ground-truth action sequence
  - END: Jaccard index of sets A (set of atoms whose truth values changed after simulating entire action sequence) and B (ground truth)

Algorithm	**IED**	**END**
Chance	0.3	0.5
Manually Defined Templates	2.5	1.8
UBL- Best Parse (Kwiatkowski et al., 2010)	5.3	6.9
VEIL (Misra et al., 2014)	14.8	20.7
Model with only train-time lexicon induction	20.8	26.8
Model with only test-time lexicon induction	21.9	25.9
Full Model	**22.3**	**28.8**

Evaluation

- 2 metrics:
  - IED: Edit distance from ground-truth action sequence
  - END: Jaccard index of sets A (set of atoms whose truth values changed after simulating entire action sequence) and B (ground truth)

Table 2: New verbs and concepts induced at test time (Section 7).

Text	**Postcondition represented by the learned logical form**	**# Log. forms explored**
"mix it with ice cream and syrup"	$\texttt{state}(\texttt{cup}_2, \texttt{ice-cream}_1) \wedge \texttt{state}(\texttt{cup}_2, \texttt{vanilla})$	15
"distribute among the couches"	$\wedge_{\texttt{j} \in \{1,3\}} \texttt{on}(\texttt{pillow}_\texttt{j}, \texttt{loveseat}_1) \wedge \texttt{on}(\texttt{pillow}_{\texttt{i}+1}, \texttt{armchair}_{\texttt{i}+1})$	386
"boil it on the stove"	$\texttt{state}(\texttt{stove}, \texttt{stovefire1}) \wedge \texttt{state}(\texttt{kettle}, \texttt{water})$	109
"change the channel to a movie"	$\texttt{state}(\texttt{tv}_1, \texttt{channel4}) \wedge \texttt{on}(\texttt{book}_1, \texttt{loveseat}_1)$	98

Appendix: Mapping Object Descriptions

Given an object description ω and a set of physical objects $\{o_j\}_{j=1}^{m}$; we want to find the correlation $\rho(\omega, o_j) \in [0, 1]$ of how well does the description ω describes the object o_j. When the description is not a pronoun, we take the following approach. We initialize $\forall_j \; \rho(\omega, o_j) = 0$ and then try the following rules in the given order, stopping after the first match:

- *category matching*: if there exists a set of objects $\{o'_j\}$ containing part of the description in its name then we define $\forall_j \rho(\omega, o'_j) = 1$.
- *containment (metonymy)*: for every object o_j; if the main noun in ω matches the state-name of a state of o_j which has value $True$ then we define $\rho(\omega, o_j) = 1$.
- *wordnet similarity*: for every object o_j we find $\rho(\omega, o_j)$ using a modified Lesk algorithm based on WordNet. If a similarity score greater than 0.85 is found then we return.
- *domain specific references*: We use giza-pp algorithm to learn translation probabilities between text and corresponding action sequences, using the training data. This gives us a probability table T[*words,object-name*] of words in text and object name in the sequence. We then initialize $\rho(\omega, o_j)$ by averaging the value of $T[w, o_j.name]$ for every word w in ω.