Learning to Interpret Natural Language
Navigation Instructions from Observations

David L. Chen and Raymond J. Mooney
Presented by: Madhumitha Sakthi
Problem Statement

- To transform natural-language navigation instructions into executable formal plans
- System learns by observing how humans follow navigation instructions

**Evaluation:**
- Three complex virtual indoor environments with numerous objects and landmarks

**Solution:**
- A learned lexicon to refine inferred plans.
- A supervised learner to induce a semantic parser with no prior linguistic knowledge.
Virtual World

Figure 1: This is an example of a route in our virtual world. The world consists of interconnecting hallways with varying floor tiles and paintings on the wall (butterfly, fish, or Eiffel Tower.) Letters indicate objects (e.g. ‘C’ is a chair) at a location.
Challenges

- People may describe routes using **landmarks** (e.g. yellow floored hall) or **specific actions** (e.g. walk forward once)
- Describe the same object differently *coat rack* vs. *hatrack*
- Amount of detail given
- Spelling and grammatical errors as well as logical errors (e.g. confusing left and right)
Training data

- Learn syntax, semantics and Lexicans
- Formally, the system is given training data in the form: \{(e_1, a_1, w_1), (e_2, a_2, w_2), \ldots, (e_n, a_n, w_n)\},

- \textbf{e}_i - natural language instruction
- \textbf{a}_i - observed action sequence
- \textbf{w}_i - current state of the world including the patterns of the floors and walls and positions of any objects

- A system that can produce the correct \textbf{a}_j given a previously unseen (\textbf{e}_j, \textbf{w}_j) pair
Training data & challenges

- No direct correspondence between $e_i$ and $a_i$

- Infer $p_i$ from training data and build semantic parser
- So, $e_i$ can be converted to $p_i$
Data MacMohan et al. (2006)

- Three different virtual worlds with interconnecting hallways
- Several short concrete hallways and seven long hallways
- Floor patterns: grass, brick, wood, gravel, blue, flower, and yellow octagons
- Three areas contain butterfly, fish, and Eiffel tower each
- Furnitures at various intersections
- Seven chosen positions
- Three worlds have the same elements but in different configs

Figure 1: This is an example of a route in our virtual world. The world consists of interconnecting hallways with varying floor tiles and paintings on the wall (butterfly, fish, or Eiffel Tower.) Letters indicate objects (e.g., 'C' is a chair) at a location.
Data MacMohan et al. (2006) conti..

- Each instructions were 5 sentences long
- They manually split the data action sequences and aligned with each sentence
- All actions were discrete with **left, right** and **move from one intersection to another**
System overview
Steps

- Constructing navigation plans
  - Basic plan: direct instructions
  - Landmark plan: complex instructions
- Plan refinement
  - Learning lexicon
  - Refined navigation $\pi_i$ for each $\epsilon_i$ using lexicon
- Learning semantic parser
  - Conversion of $\epsilon_i$ to $\pi_i$ given Context Free Grammar
  - turn to face the sofa $\rightarrow$ Turn(), Verify(front: SOFA)
- Executing Instructions

Instruction: “Go away from the lamp to the intersection of the red brick and wood.”

Basic:
- Turn(1), Travel(steps: 1)

Landmarks:
- Turn(1), Verify(left: WALL, back: LAMP, back: HATRACK, front: BRICK HALL),
  Travel(steps: 1),
  Verify(side: WOOD HALL)

Figure 3: Examples of automatically generated plans.
Learning Lexicon

Algorithm 1 \textsc{Lexicon Learning}

\textbf{input} Navigation instructions and the corresponding navigation plans \((e_1, p_1), \ldots, (e_n, p_n)\)

\textbf{output} \textit{Lexicon}, a set of phrase-meaning pairs

1: main
2: for n-gram \(w\) that appears in \(e = (e_1, \ldots, e_n)\) do
3: \hspace{1em} for instruction \(e_i\) that contains \(w\) do
4: \hspace{2em} Add navigation plan \(p_i\) to \textit{meanings}(\(w\))
5: \hspace{1em} end for
6: repeat
7: \hspace{1em} for every pair of meanings in \textit{meanings}(\(w\)) do
8: \hspace{2em} Add intersections of the pair to \textit{meanings}(\(w\))
9: \hspace{1em} end for
10: \hspace{1em} Keep \(k\) highest-scoring entries of \textit{meanings}(\(w\))
11: until \textit{meanings}(\(w\)) converges
12: Add entries of \textit{meanings}(\(w\)) with scores higher than threshold \(t\) to \textit{Lexicon}
13: end for
14: end main

Figure 4: Example of computing the intersections of two graph representations of navigation plans.
Refining navigation plan using lexicon

- Lexicon learned remove extraneous information from landmark plans

1. Pick highest scored \((w,g)\) s.t. \(w \in e\) and \(g \in p\)
2. Remove \(w\) from \(e\), mark all \(g\) in \(pi\)
3. Remove all non-marked nodes in \(pi\)
4. Refined \(pi'\)

Till words in \(e\) or lexicon entries
Learning a semantic parser

- KRISP to convert natural language to formal language.
- turn to face the sofa $\rightarrow$ Turn(), Verify(front: SOFA)
- SVM to estimate probabilities of production rule in CFG.
- This compositionally build a complete string in the formal language for the sentence.
Executing Instructions

- **MARCO** uses compound action specifications
- Raw text is converted to compound action specifications
- Interleaves action and perception to execute action

- It can also handle *walk two steps to the chair* when the chair is three steps away
Evaluation dataset

- Each sentence was paired with an action sequence followed by majority of the followers action’s and map
- 300 sentences were not matched with an action
- “This is position n”
- Sentence level version for training and both for testing

<table>
<thead>
<tr>
<th></th>
<th>Original</th>
<th>Single-sentence</th>
</tr>
</thead>
<tbody>
<tr>
<td># instructions</td>
<td>706</td>
<td>3236</td>
</tr>
<tr>
<td>Vocabulary size</td>
<td>660</td>
<td>629</td>
</tr>
<tr>
<td>Avg. # sentences</td>
<td>5.0 (2.8)</td>
<td>1.0 (0)</td>
</tr>
<tr>
<td>Avg. # words</td>
<td>37.6 (21.1)</td>
<td>7.8 (5.1)</td>
</tr>
<tr>
<td>Avg. # actions</td>
<td>10.4 (5.7)</td>
<td>2.1 (2.4)</td>
</tr>
</tbody>
</table>
Generating Navigation Plans

- Gold-standard plans
  - Hand-annotated instructions in a sentence to navigation
- Refined landmark plans preserved high precision of basic plan and high recall of landmark.

<table>
<thead>
<tr>
<th></th>
<th>Precision</th>
<th>Recall</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basic plans</td>
<td>81.46</td>
<td>55.88</td>
<td>66.27</td>
</tr>
<tr>
<td>Landmarks plans</td>
<td>45.42</td>
<td>85.46</td>
<td>59.29</td>
</tr>
<tr>
<td>Refined landmarks</td>
<td>78.54</td>
<td>78.10</td>
<td>78.32</td>
</tr>
</tbody>
</table>

Table 2: Partial parse accuracy of how well the inferred navigation plans match the human annotations.
Building a Semantic parser

- Trained on both inferred plans and human-annotated plans
- Leave-one-out-map approach for training
- Clearing unnecessary information produced better results

<table>
<thead>
<tr>
<th></th>
<th>Precision</th>
<th>Recall</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basic plans</td>
<td>86.68</td>
<td>48.62</td>
<td>62.21</td>
</tr>
<tr>
<td>Landmarks plans</td>
<td>50.40</td>
<td>31.10</td>
<td>38.39</td>
</tr>
<tr>
<td>Refined landmarks plans</td>
<td>90.22</td>
<td>55.10</td>
<td>68.37</td>
</tr>
<tr>
<td>Human annotated plans</td>
<td>88.24</td>
<td>71.70</td>
<td>79.11</td>
</tr>
</tbody>
</table>

Table 3: Partial parse accuracy of how well the semantic parsers trained on the different navigation plans performed on test data.
Executing the navigation plan

- Leave-one-map-out cross-validation
- Executed the plan 10 times and averaged results are shown
- Compared with lower baseline (generative model)
- Three upper baseline
  - Trained on human annotated data
  - Full MARCO system
  - Human Followers

<table>
<thead>
<tr>
<th></th>
<th>Single-sentence</th>
<th>Complete</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simple generative model</td>
<td>11.08</td>
<td>2.15</td>
</tr>
<tr>
<td>Basic plans</td>
<td>56.99</td>
<td>13.99</td>
</tr>
<tr>
<td>Landmarks plans</td>
<td>21.95</td>
<td>2.66</td>
</tr>
<tr>
<td>Refined landmarks plans</td>
<td>54.40</td>
<td>16.18</td>
</tr>
<tr>
<td>Human annotated plans</td>
<td>58.29</td>
<td>26.15</td>
</tr>
<tr>
<td>MARCO</td>
<td>77.87</td>
<td>55.69</td>
</tr>
<tr>
<td>Human followers</td>
<td>N/A</td>
<td>69.64</td>
</tr>
</tbody>
</table>
Critique

- The virtual world and plans are limited vocabulary For eg. “make a roundabout”
- Hard to extend this system to real-world
- What about cases where the score(w,g) is <0.4 but were significat?
- How did they choose t=0.4?
- Partial accuracy is missing information about virtual world used for training and testing

- Mitigate mistakes made in the earlier steps
Future research directions

- Extending this work to multiple languages - no prior linguistic knowledge is required
- Minor: Try data size, test it on trigram / $n$-gram for lexicon learning