Language grounding in simulated environments by deep reinforcement learning

Haonan Yu

Baidu Research

07/20/2018
Personal background

• B.S., Computer Science, Peking University, 2011

• Ph.D., Computer Engineering, Purdue University, 2016
  • Language grounding in realistic video via supervised learning

• Research Scientist, Baidu, now
  • Language grounding in vision and control in virtual environments with very weak supervision (reward signals)
  • Engineering efforts on reinforcement learning algorithms and frameworks
Language grounding

• Wikipedia (*symbol grounding problem*)
  • “… the problem of how words (symbols) get their meanings, and hence to the problem of what meaning itself really is.”

• Associations with *sensory-motor* experiences

• *Not* to explain word meanings with ungrounded tokens from some dictionary, e.g.
  • Word embedding / Language modeling: explain a word with its context
  • Machine translation: explain source language with target language
  • Semantic parsing: explain sentences with symbolic representations
  • …
Sensory-motor experiences

• Sensor: visual (image & video), audio, depth, liquid flow, etc

• Motor: control
  • Continuous: angle, speed, force, etc
  • Discrete: left, right, forward, backward, turn left, turn right, etc

• Language grounding: to understand language in terms of sensory-motor features
Dynamic data

• Traditional Supervised Learning on a stationary dataset
  \[ \mathbb{E}_{Z \sim P^*(Z)}[f_{\theta'}(Z)] \]

• The existence of control leads to a dynamic dataset
  \[ \mathbb{E}_{Z \sim P_\theta(Z)}[f_{\theta'}(Z)] \]

• Strong supervision is no longer practical because of the non-stationarity

• Strong -> weak (reward signals)
  • How good the taken action is instead of what a good action should be taken
RL as weighted SL

Supervised Learning (cross entropy)
\[ \nabla_{\theta} \log P_{\theta}(a_i^*|s_i) \]

Reinforcement Learning with policy gradient (weighted cross entropy)
\[ \nabla_{\theta} \log P_{\theta}(a_i|s_i) \text{ can simply be weighted by the accumulated reward } \sum_{i' = i}^{\infty} r_{i'} \]
Starting with virtual environments

- Control and reward collection in realistic environments are both difficult
- Most current research work turns to virtual environments
Baidu’s XWorld for language grounding

(https://github.com/PaddlePaddle/XWorld)

Inputs: Raw pixel inputs + unstructured commands + sparse rewards
Projects based on XWorld

1. Guided feature transformation for language grounding

2. Zero-shot sentence understanding
   “Interactive Grounded Language Acquisition and Generalization in a 2D World”, Haonan Yu, Haichao Zhang, and Wei Xu, ICLR 2018
GFT – introduction

- Problem: navigate under a language command
- The key is how to fuse language and images to produce a representation for downstream control, i.e., language grounding

Gated network, [Chaplot et al, 2017]
Concatenation, [Hermann et al, 2017]
GFT – motivation

• How can we improve upon the gated network?

\[ g = (l \otimes 1^T) \odot C \]

• \( C \in \mathbb{R}^{D \times N} \) is a feature matrix given by a CNN

• \( l \in [0,1]^D \) is a sentence gated vector generated by an RNN/LSTM

• The assumption is that \( l \) is able to select which feature maps to be used for control

• In the same feature space for all possible commands!

• Feasible but large \( D \) required as problem scales!
GFT – method

• Why not rotate and scale the feature space given a specific command?

\[ \mathbf{C}^{[j]} = g(\mathbf{T}_j \begin{bmatrix} \mathbf{C}^{[j-1]} \\ 1^T \end{bmatrix}), \quad 1 \leq j \leq J, \]

• \( \{\mathbf{T}_j \in \mathbb{R}^{D \times (D+1)}\} \) is a sequence of transformation matrices generated by a command

• GFT is a generalization of:
  • FiLM [Perez et al., 2018] \( \mathbf{c}_d^{[j]} = g(\lambda_d \mathbf{c}_d^{[j-1]} + b_d) \)
  • Gated network [Chaplot et al., 2017]
GFT – agent framework

"Please move to the bike."
GFT – experiments

• 6 comparison methods: same CNN configuration, same control module, same optimization hyperparameters, only different language grounding modules

• Five types of navigation tasks

<table>
<thead>
<tr>
<th>Type</th>
<th>Navigation target</th>
<th>Example command</th>
</tr>
</thead>
<tbody>
<tr>
<td>nav</td>
<td>the specified object</td>
<td>“Please go to the chair.”</td>
</tr>
<tr>
<td>nav_nr</td>
<td>an object near the specified one</td>
<td>“Move to the object near the chair.”</td>
</tr>
<tr>
<td>nav_bw</td>
<td>the location between the two objects</td>
<td>“Go to the location between the chair and the table?”</td>
</tr>
<tr>
<td>nav_avoid</td>
<td>any object but the specified one</td>
<td>“Avoid the chair.”</td>
</tr>
<tr>
<td>nav_dir</td>
<td>an object specified by a relative direction w.r.t.</td>
<td>“Navigate to the object left of the chair.”</td>
</tr>
<tr>
<td></td>
<td>another object</td>
<td></td>
</tr>
</tbody>
</table>

• Commands are generated by pre-defined rules and grammar
GFT – experiments

• XWorld3D
  • 88 objects, 8 spatial-relation words, 136 words in total, 709k distinct sentences, lengths ranging from 1 to 15
  • 8 x 8 random maps, each containing 4 objects and 16 obstacles

• XWorld2D
  • 115 objects, 8 spatial-relation words, 163 words in total, 1.18m distinct sentences, lengths ranging from 1 to 15
  • 8 x 8 random maps, each containing 4 objects and 16 obstacles

• Data are very balanced; no good chance performance
• Both environments are partially observed
GFT – experiment results

(Trained with A2C [Wang et al. 2016] with 32 agents in parallel)
Guided Feature Transformation (GFT): A Neural Language Grounding Module for Embodied Agents

https://www.youtube.com/watch?v=bOBb1uhuJxg
GFT – limitations

• More parameters to learn compared to its simplified version FiLM (~3:2)
  • However, $T_j$ is not the definitive factor that affects the overall model size.
  • The projection matrix from CNN to a hidden vector
• Slower in computation depending on the sequence length $J$ of transformations
ZS understanding – in XWorld2D

• ZS1: new word combinations never seen during training
• ZS2: words learned exclusively in one use case, but applied to another use case without retraining
ZS understanding – motivations

• How can we achieve ZS1 and ZS2 (with a good chance) at the same time?

• **Compositionality:** understanding a sentence consists of understanding individual words and combining the individual understanding results

• **Transfer learning:** shared word representation between the two different use cases to enable the transfer
  - Use case 1: language grounding (QA questions, NAV commands)
  - Use case 2: language prediction (QA answers)
ZS understanding – transfer learning

• Simplify: transfer between
  • word prediction given a *single* feature (location in vocabulary), and
  • *single-word* grounding (location in images)

• Key observation: computing an inner product of a word embedding and a feature vector

• Grounding a single word:
  • Perform 1x1 convolution on the CNN output with the word embedding to obtain a score/saliency map

• Predicting a word given a location of an image:
  • Multiply the CNN feature vector with the embedding table to obtain a score table
ZS understanding – transfer learning

Single-word grounding

Feature vector

Image CNN output

Loc_0

Loc_1

...

Loc_N

×

Grounding score map

Word embedding

Word prediction

Embedding vector

CNN feature vector

word_0

word_1

...

word_M

Transposed embedding table

Driven by navigation rewards

Driven by classification cost
ZS understanding – compositionality

• Two issues to be addressed to extend this transferring ability to *multi-word* commands and *multi-location* features:
  • Find out which words to ground given QA questions or NAV commands
    • Multi-hop word attention to learn sentence structures
    • Every time (softly) extract a word for grounding
    • Finally combine the individual score maps as the grounding result
  
  • Find out which image locations to predict for QA answers
    • Compute the grounding map of the question to (softly) select locations
ZS understanding – compositionality

- Multi-hop word attention to learn sentence structures

Word attention: \( o^i_l \propto \exp \left[ S_{\text{cos}}(p^{i-1}_l, \overline{w}_l) \right] \)

Attended context: \( \overline{w}^i = \sum_l o^i_l \overline{w}_l \)

Attended word: \( s^i = \sum_l o^i_l \overline{w}_l \)

Interpreter state: \( p^i = \text{GRU}(p^{i-1}_l, \overline{w}^i) \)
ZS understanding – compositionality

• Combine the individual grounding results

Detection: \[ y' = \text{softmax}(\phi(h, 1, s^i)) \]

Map transform: \[ x_{\text{loc}}^i = y' \ast y^{i-1} \]

Map update gate: \[ \rho^i = \sigma(Wp^i + b) \]

Map update: \[ y^i = \rho^i x_{\text{loc}}^i + (1 - \rho^i) y^{i-1} \]

• Why is ‘∗’ (2D convolution) performed on two 2D maps?
ZS understanding – compositionality

Made possible by appending a trainable feature cube to the image CNN cube

• * is used for grounding spatial-relation words!
ZS understanding – agent framework

NAV: “Move to the red apple.”
QA: “Say the color of avocado.”
ZS understanding – experiments

Four types of NAV commands
- `nav_obj`
- `nav_col_obj`
- `nav_nr_obj`
- `nav_bw_obj`

Twelve types of QA questions
- `rec_col2obj`
- `rec_obj2col`
- `rec_loc2obj`
- `rec_loc2col`
- `rec_obj2loc`
- `rec_loc_obj2obj`
- `rec_loc_obj2col`
- `rec_col_obj2loc`
- `rec_col_obj2col`
- `rec_bw_obj2obj`
- `rec_bw_obj2col`

XWORLD sentences
- NAV command (~570k): “Please arrive at the location between the cat and the dog.”
- QA question (~1m): “What is the object that is west of the orange?”
- QA answer (136): “Dragon”
ZS understanding – experiments

• 185 words in total, with 9 spatial relations, 8 colors, and 119 objects
• Sentences are generated with pre-defined rules and grammar
• Lengths range from 2 to 13
• 7x7 randomly generated maps with 5 objects and 15 obstacles
• Each map is fully observable
• All methods trained by off-policy AC with experience replay
ZS understanding – experiment results

Not so impressive compared to CA and SAN in normal (no ZS understanding) settings!!
ZS understanding – ZS settings

• ZS1: randomly hold out $X\%$ of word pairs (object, spatial relation), (object color), (object, object) during training
• ZS2: randomly hold out $X\%$ of object words during training
• $X = 12.5, 20.0, 50.0, 66.7, 90.0$ five different values
• Only report test performance on ZS sentences
Conclusion: ZS1 is not that difficult. Deep neural nets seem to inherently have the ability of handling it (able to interpolate to some extent). But our method is more robust.
Conclusion: ZS2 is much more difficult. The compared deep learning models cannot easily handle it. Our method is very robust to the held-out ratio change.
ZS understanding – ZS2 experiment results

• So far we’ve spent lots of efforts to achieve the ZS2 ability. Is it really worth it?

• Imagine that a child learns a new object word from a classification/recognition task, he/she probably won’t need to retrain all the other tasks that have language containing that new word

• Yes, we believe that it is important!
ZS understanding – limitations

- The framework is dedicated to a fully-observed 2D grid world
- The interpreter is still quite limited in its formulation; can only handle a small subset of language
- As the number of attention hops increases, it becomes difficult to train the whole framework end to end
Interesting ideas for future language grounding research

• ZS (especially ZS2) sentence understanding in 3D environments!
• Lifelong interactive language learning without session breaks
• Actively ask questions if new scenes/concepts encountered
• Multi-agent cooperation via language in 3D environments
• ...
Thank you! Questions?