Visual Semantic Navigation using Scene Priors

Wei Yang[1] , Xiaolong Wang[2] , Ali Farhadi[4,5] ,
Abhinav Gupta[2,3] & Roozbeh Mottaghi[5]

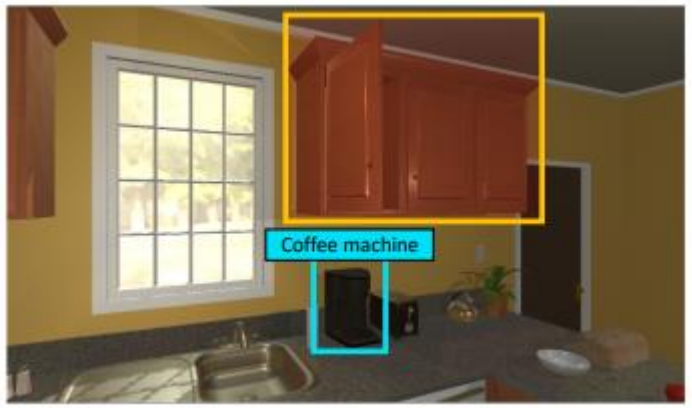
1 The Chinese University of Hong Kong 2 Carnegie Mellon University 3 Facebook AI Research
4 University of Washington 5 Allen Institute for AI

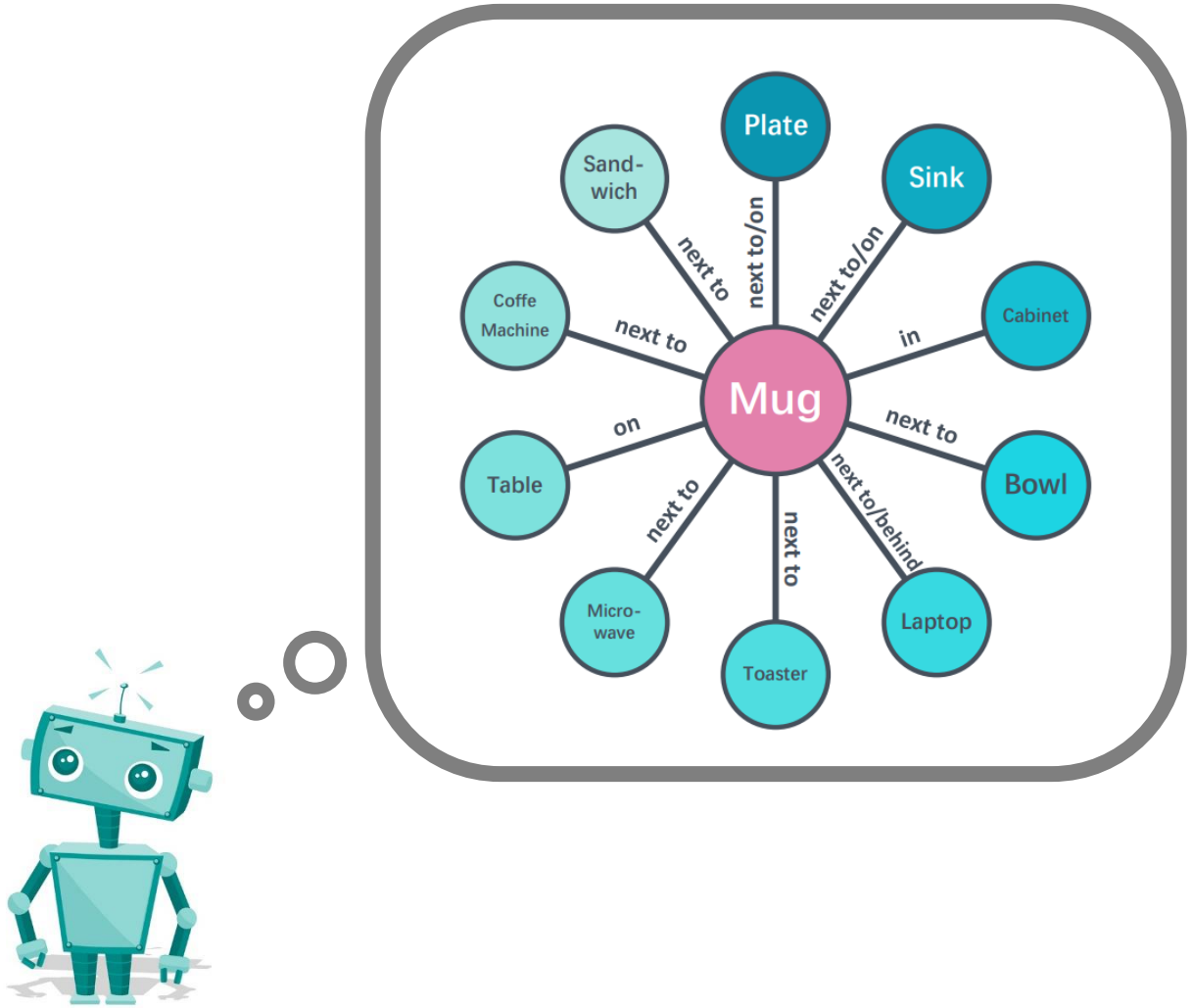
ICLR 2019

Presentation : **Obin Kwon** (2018-29096)

Object Search using Semantic priors

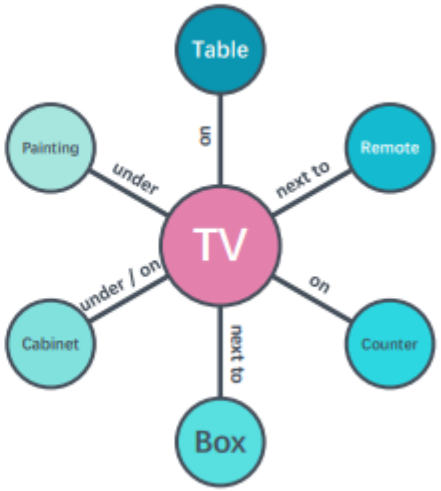
"Find a mug"





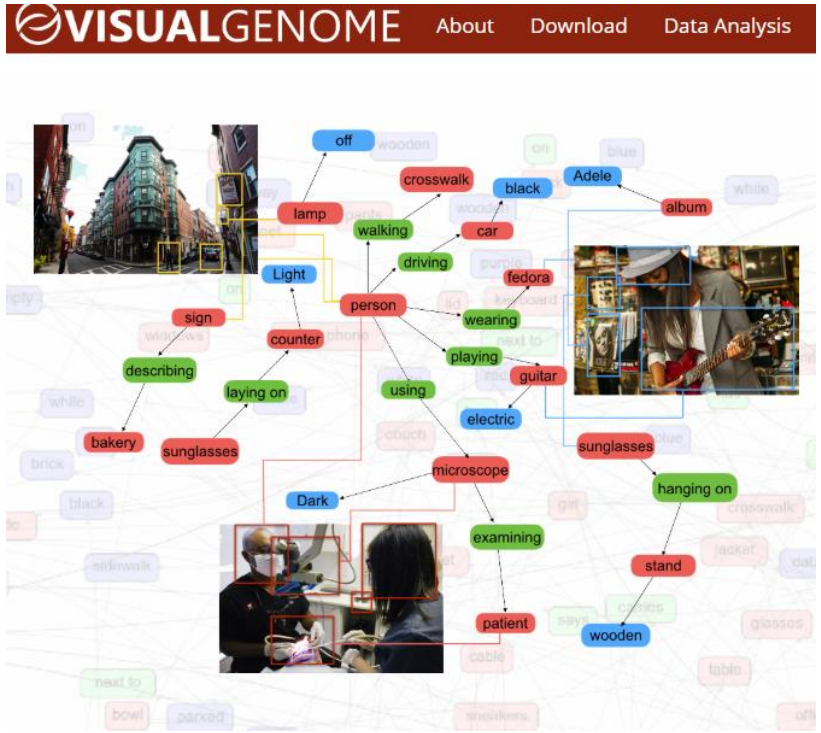
Object Search using Semantic priors

- Human use strong priors about the functional and semantic structure of the world to develop efficient navigation strategies
- This paper models the relationship between objects into graphs and incorporate it into the navigation policy.



Graph Construction

- Encodes spatial relationships between different object categories
- **Visual Genome** Dataset
  - 100K natural images annotated with objects, attributes and the relationships between objects.



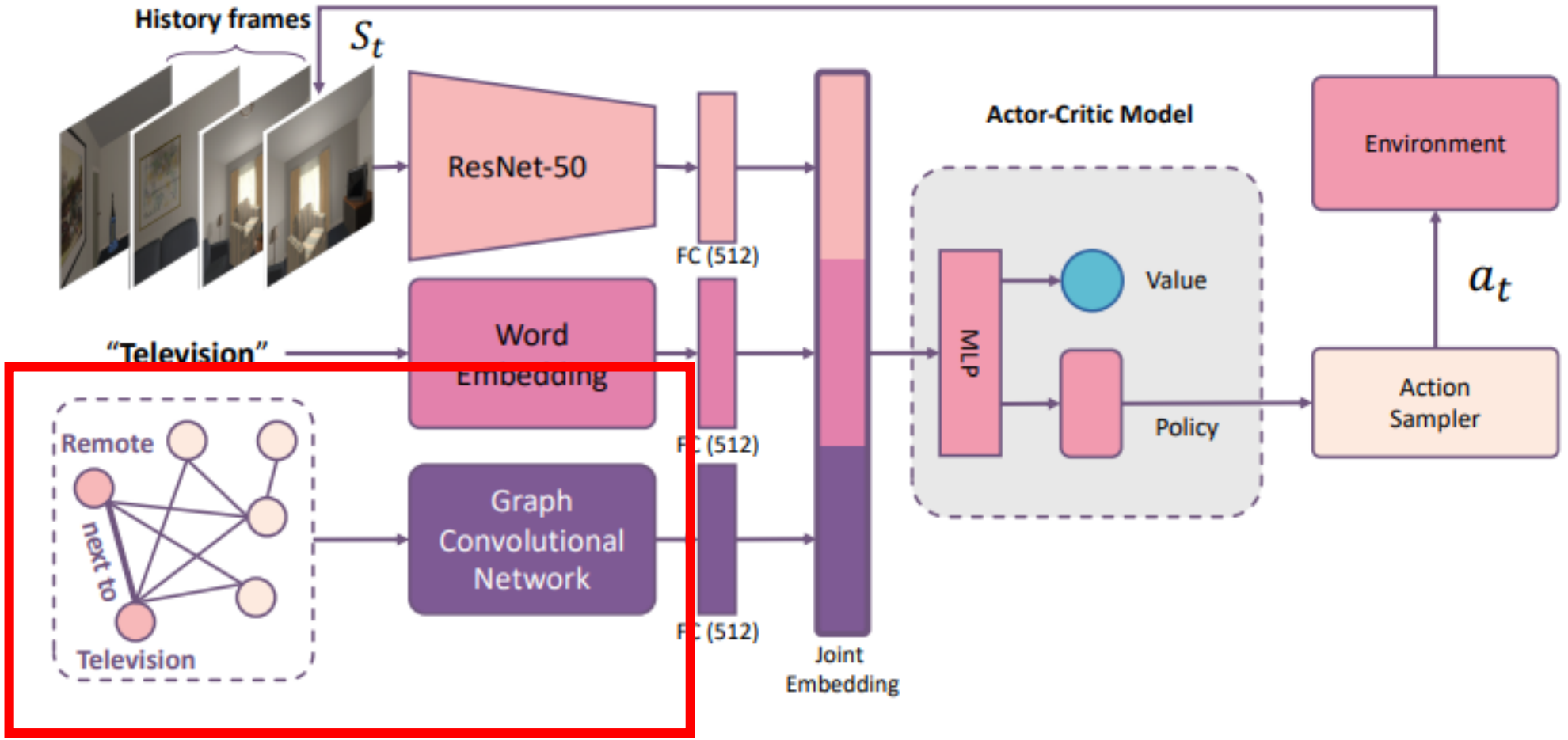
Using this dataset, build the knowledge graph by including all the object categories that appear in the simulator.

G = (V,E)

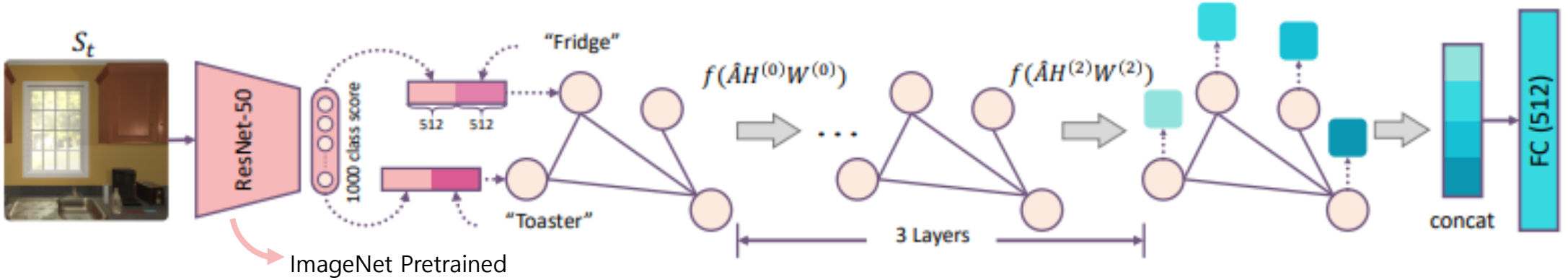
v : object category

e : relationship between a pair of object categories

Overall Architecture



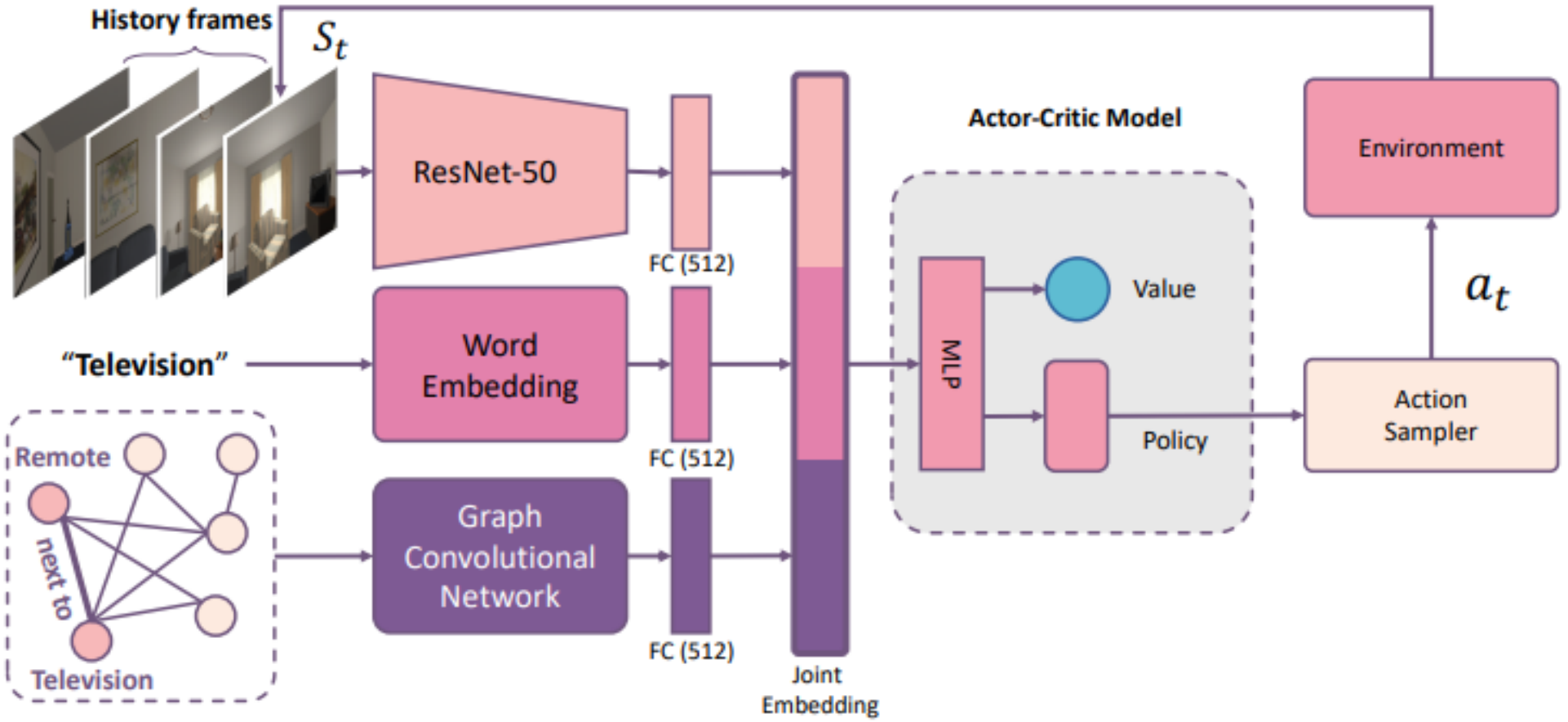
Incorporate knowledge graph into policy



- ChebNet [1]
  - $X = [x_1, \ldots, x_{|V|}] \in R^{|V| \times D}$ -> GCN -> $Z = [z_1, \ldots, z_{|V|}] \in R^{|V| \times D}$
  - Normalized Adjacency matrix $\hat{A}$
  - $H^{l+1} = f(\hat{A}, H^{(l)}W^{(l)})$
  - $H^{(0)} = X,\ H^{(L)} = Z$

[1] Thomas N. Kipf and Max Welling. *Semi-supervised classification with graph convolutional networks*. In ICLR, 2017.

Overall Architecture



Experiments

- Environment
  - AI2-THOR simulator
  - 120 indoor Scenes (kitchens, living rooms, bedrooms, bathrooms)
  - 53 objects ($|V| = 53$)
    - Split categories into known and unseen(novel) sets. Only the known set of object is used in training
  - A = {'Move Forward', Move Back', 'Turn Left', 'Turn Right', 'Stop'}
  - Reward
    - - 0.01 for every step
    - + 10 for success (reached to target within a certain number of steps)
  - A3C

Experiment Setup

- **Baselines**
  - Random actions
  - A3C
- **Navigation Scenario**
  1. **Seen scenes** with **known object** categories
  2. **Seen scenes** but **novel object** categories
  3. **Known object** categories but **unseen scenes**
  4. Both **unseen scenes** with **novel object** categories

- **Metrics**
  - Success Rate
  - SPL $\frac{1}{N}\sum_{i=1}^{N} S_i \frac{L_i}{\max(P_i, L_i)}$

S_i : $success\ indicator$
L_i : $the\ shortest\ path\ length$
P_i : $agent\ path\ length$

Experiment Results

		Kitchen	Living room	Bedroom	Bathroom	Avg.
Seen scenes,	Random	2.4 / 3.5	1.1 / 1.7	1.8 / 2.7	3.2 / 4.8	2.1 / 3.1
	A3C	38.5 / 51.0	9.7 / 15.1	6.8 / 11.5	69.1 / 81.0	31.1 / 39.6
Known objects	Ours	**58.6 / 72.7**	**12.4 / 18.6**	**41.6 / 52.4**	**71.3 / 83.0**	**46.0 / 56.7**
Seen scenes,	Random	0.9 / 1.3	0.8 / 1.2	2.3 / 3.4	1.4 / 2.1	1.4 / 2.0
	A3C	2.1 / 4.9	3.2 / 4.8	0.5 / 1.7	17.1 / 28.5	5.7 / 9.9
Novel objects	Ours	**3.2 / 6.1**	**9.8 / 16.2**	**6.2 / 8.6**	**24.7 / 37.3**	**11.0 / 17.1**
Unseen scenes,	Random	4.1 / 5.9	0.9 / 1.3	1.6 / 2.4	4.2 / 6.2	2.7 / 3.9
	A3C	11.5 / 18.8	0.5 / 2.5	2.2 / 3.8	8.6 / 18.7	5.7 / 10.4
Known objects	Ours	**12.7 / 20.5**	**1.0 / 4.0**	**4.5 / 11.0**	**8.7 / 21.1**	**6.7 / 13.4**
Unseen scenes,	Random	2.0 / 2.8	0.6 / 1.0	**2.0** / 2.8	2.7 / 3.9	1.8 / 2.6
	A3C	2.2 / 7.5	2.5 / 4.4	1.3 / 4.4	3.4 / 9.3	2.4 / 5.9
Novel objects	Ours	**3.3 / 12.7**	**2.8 / 5.3**	**2.0 / 6.3**	**4.1 / 12.2**	**3.1 / 8.5**

Table 1: **Results using termination (stop) action.** SPL / Success rate (%) is shown. We compare against a random baseline and A3C (Mnih et al., 2016).

Experiment Results – without STOP action

		Kitchen	Living room	Bedroom	Bathroom	Avg.
Seen scenes, Known objects	Random	17.9 / 33.1	12.1 / 30.5	16.8 / 51.2	24.5 / 34.6	17.8 / 37.3
	A3C	79.9 / 86.7	38.8 / 57.6	87.8 / 89.5	**93.7 /96.6**	75.0 / 82.5
	Ours	**83.5 / 88.2**	**46.4 /64.4**	**90.6 / 92.7**	93.6 / 96.5	**78.5 / 85.5**
Seen scenes, **Novel** objects	Random	10.0 / 23.1	8.0 / 18.5	17.3 / 35.2	11.2 / 32.2	11.6 / 27.2
	A3C	20.2 / 38.8	24.2 / 46.5	23.5 / 35.8	50.2 / 74.6	29.5 / 48.9
	Ours	**22.9 / 53.6**	**39.5 / 66.5**	**26.1 / 38.9**	**50.5 / 78.6**	**34.7 / 59.4**
Unseen scenes, Known objects	Random	27.3 / 45.2	5.6 / 16.6	13.1 / 34.5	36.0 / 49.1	20.5 / 36.3
	A3C	39.5 / 56.2	12.0 / 31.8	22.5 / 49.2	47.4 / 60.2	30.3 / 49.3
	Ours	**46.2 / 62.5**	**13.8 / 40.6**	**26.5 / 58.6**	**51.5 / 65.8**	**34.5 / 56.9**
Unseen scenes, **Novel** objects	Random	21.3 / 44.3	3.3 / 22.9	25.8 / 47.8	25.5 / 48.9	19.0 / 41.0
	A3C	26.1 / 56.3	9.4 / 25.1	28.2 / 54.0	33.8 / 90.7	24.4 / 56.5
	Ours	**38.5 / 62.5**	**13.7 / 40.3**	**30.1 / 63.1**	**39.2 / 93.6**	**30.4 / 64.9**

Table 2: **Results without termination (stop) action.** SPL / Success rate (%) is shown. We compare against a random baseline and A3C. This scenario is simpler than the case shown in Table 1.

Experiment Results – ablation study

- Kitchen, without stop action
- **Drop Nodes / Edges : (SPL)**

Drop %	0%	20%	40%	60%	80%
Objects	38.5	34.8	33.7	33.5	31.1
Relations	38.5	36.7	35.0	34.2	31.5

Table 3: Results of removing objects and relations in the knowledge graph.

- **Knowledge graph**
  - Fully connected graph : 32.5
  - Random graph : 30.1 ± 0.6

Proper Knowledge graph
38.5

Conclusion

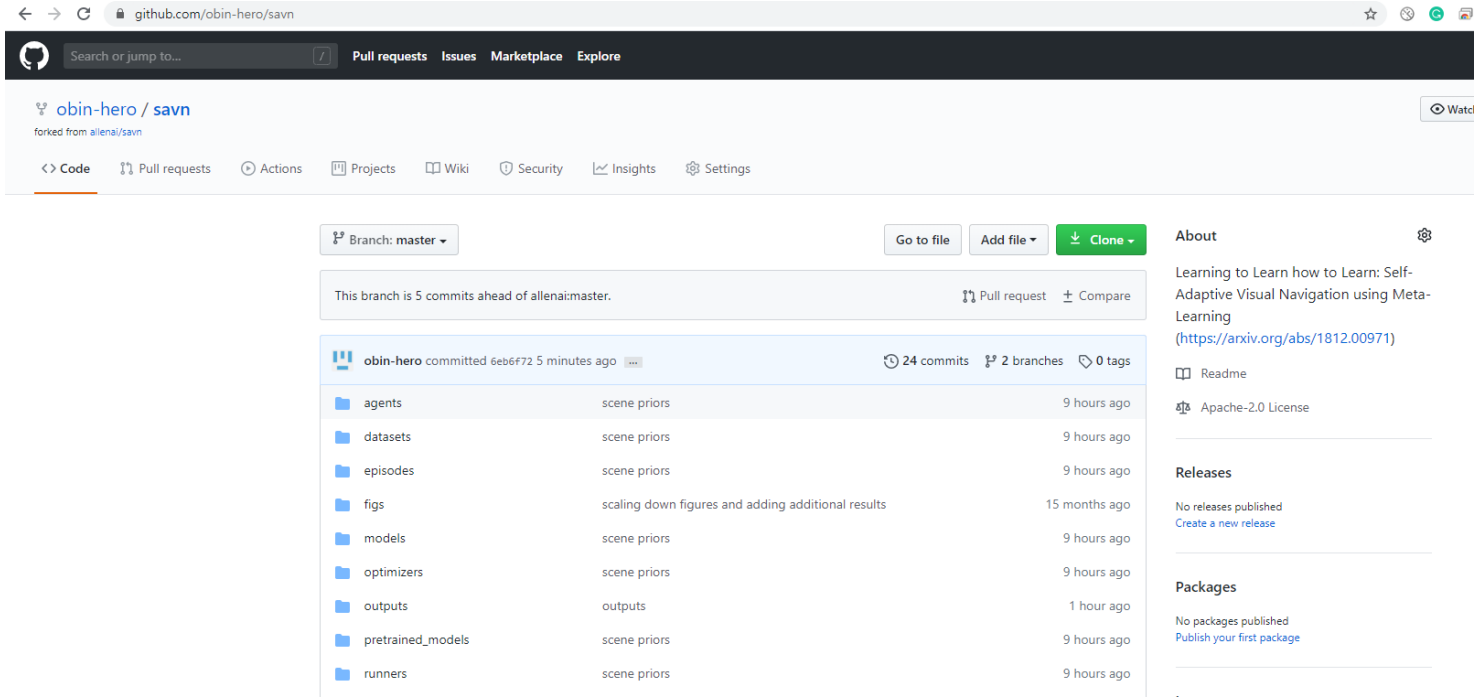
- Integrates semantic and functional priors with a deep reinforcement learning model for the task of navigation
- GCN for encode the prior knowledge and to update the knowledge according to the observations from the current scene
- The experiments show that prior knowledge improves generalization to unseen scenes or targets.

- Recent work with similar ideas

- Lv, Yunlian, et al. "**Improving Target-driven Visual Navigation with Attention on 3D Spatial Relationships**." *arXiv preprint arXiv:2005.02153* (2020).
- Nguyen, Tai-Long, Do-Van Nguyen, and Thanh-Ha Le. "**Reinforcement Learning Based Navigation with Semantic Knowledge of Indoor Environments**." *2019 11th International Conference on Knowledge and Systems Engineering (KSE)*. IEEE, 2019.
- Qiu, Yiding, Anwesan Pal, and Henrik I. Christensen. "**Target driven visual navigation exploiting object relationships**." *arXiv preprint arXiv:2003.06749* (2020).
- Mahdi Kazemi Moghaddam, M., et al. "**Utilising Prior Knowledge for Visual Navigation: Distil and Adapt**." *arXiv* (2020): arXiv-2004

Reproduce Experiments

- Code : https://github.com/obin-hero/savn
- I brought the code from other work as it use this paper as baseline.
- If you want to experiment the code, just follow the setup process in README.md
- I added visualization code to see how the agent moves



Code explanation

- Main operations are done in models/gcn.py

1. Environmental state if forwarded to model

```
159     def forward(self, model_input, model_options):
160         state = model_input.state.cuda()
161         (hx, cx) = model_input.hidden
162 
163         target = model_input.target_class_embedding.cuda()
164         action_probs = model_input.action_probs.cuda()
165         x, image_embedding = self.embedding(state, target, action_probs)
166         actor_out, critic_out, (hx, cx) = self.a3clstm(x, (hx.cuda(), cx.cuda()))
167 
168         return ModelOutput(
169             value=critic_out.detach().cpu(),
170             logit=actor_out.detach().cpu(),
171             hidden=(hx.detach().cpu(), cx.detach().cpu()),
172             embedding=image_embedding.detach().cpu(),
173         )
```

State embedding

The agent reasons about the action using embedded features

Code explanation

2. Embedding

```
def embedding(self, state, target, action_probs):
    if not self.resnet_built:
        self.resnet_build()
    action_embedding_input = action_probs

    glove_embedding = F.relu(self.embed_glove(target))
    glove_reshaped = glove_embedding.view(1, 64, 1, 1).repeat(1, 1, 7, 7)

    action_embedding = F.relu(self.embed_action(action_embedding_input))
    action_reshaped = action_embedding.view(1, 10, 1, 1).repeat(1, 1, 7, 7)
    img_tensor = resnet_input_transform(state.cpu().numpy().astype(np.uint8),224).cuda().unsqueeze(0)
    state = self.image_res18(img_tensor)
    image_embedding = F.relu(self.conv1(state))
    x = self.dropout(image_embedding)
    x = torch.cat((x, glove_reshaped, action_reshaped), dim=1)
    x = F.relu(self.pointwise(x))
    x = self.dropout(x)
    out = x.view(x.size(0), -1)

    out = torch.cat((out, self.gcn_embed(state)), dim=1)

    return out, image_embedding
```

Target object word embedding

Image embedding

GCN embedding

Code explanation

3. GCN Embedding

```
def gcn_embed(self, state):
    x = self.resnet18[0](state)
    x = x.view(x.size(0), -1)
    x = torch.sigmoid(self.resnet18[1](x))
    class_embed = self.get_class_embed(x)
    word_embed = self.get_word_embed(self.all_glove.detach())
    x = torch.cat((class_embed.repeat(self.n, 1), word_embed), dim=1)
    x = torch.mm(self.A, x)
    x = F.relu(self.W0(x))
    x = torch.mm(self.A, x)
    x = F.relu(self.W1(x))
    x = torch.mm(self.A, x)
    x = F.relu(self.W2(x))
    x = x.view(1, self.n)
    x = self.final_mapping(x)
    return x
```

Get each class score from the last layer of pretrained resnet

Turn into embedding

Get all the class word embedding

GCN layers

Repeating aggregation & embedding

Experiment Results

- Video results

- Quantitative Results

Success Rate	SPL
31.5	0.133