A ADDITIONAL BACKGROUND

A.1 CONTRASTIVE LEARNING

Contrastive learning refers to a class of methods that attempt to minimize the distance of data points that share some attribute or condition. Given some "anchor" datapoint, $\boldsymbol{x}$, one typically finds some "positive" datapoint $\boldsymbol{x}^+$ that meets a condition C and forms the pair $(\boldsymbol{x}, \boldsymbol{x}^+)$. "Negatives" $\{\boldsymbol{x}_k^-\}$ are then data points that do not meet condition C for $\boldsymbol{x}$. In order to learn an embedding function $\boldsymbol{z} = f_{\texttt{enc}}(\boldsymbol{x})$, a common objective used is of the form Arora et al. (2019)

$$\mathcal{L}_{\texttt{contrastive}} = \mathbb{E}_{\boldsymbol{x},\boldsymbol{x}^+,\boldsymbol{x}_1^-,\ldots,\boldsymbol{x}_N^-}\left[-\log\left(\frac{\exp\left(\boldsymbol{z}^\top\boldsymbol{z}^+\right)}{\exp\left(\boldsymbol{z}^\top\boldsymbol{z}^+\right)+\sum_k\exp\left(\boldsymbol{z}^\top\boldsymbol{z}_k^-\right)}\right)\right] \tag{9}$$

It is common to chose negatives randomly from one's available data. This objective then encourages the learner to learn f such that the inner product of $z^\top z^+$ is higher on average than for $z^\top z^-$.

In supervised learning, the condition C can be whether two data points share a label. In unsupervised learning, other ground-truth information may be used. This is often referred to as "self-supervised" learning. For example, Oord et al. (2018) define the condition C as whether x^+ occured within K time-steps of x. When ground-truth information is not available, a heuristic is typically used. This is what we do in our work.

A.1.1 A MORE NUMERICALLY STABLE VERSION

To give insight into some of the numerical stability issues that arise with the contrastive learning objective, we can introduce temperature τ and re-write it as follows Sohn (2016)

$$\begin{aligned}
\mathcal{L}_{\texttt{contrastive}} &= \mathbb{E}_{\boldsymbol{x},\boldsymbol{x}^+,\boldsymbol{x}_1^-,\ldots,\boldsymbol{x}_N^-}\left[-\log\left(\frac{\exp\left(\boldsymbol{z}^\top\boldsymbol{z}^+/\tau\right)}{\exp\left(\boldsymbol{z}^\top\boldsymbol{z}^+/\tau\right)+\sum_k\exp\left(\boldsymbol{z}^\top\boldsymbol{z}_k^-/\tau\right)}\right)\right] \\
&= \mathbb{E}_{\boldsymbol{x},\boldsymbol{x}^+,\boldsymbol{x}_1^-,\ldots,\boldsymbol{x}_N^-}\left[\log\left(\frac{\exp\left(\boldsymbol{z}^\top\boldsymbol{z}^+/\tau\right)+\sum_k\exp\left(\boldsymbol{z}^\top\boldsymbol{z}_k^-/\tau\right)}{\exp\left(\boldsymbol{z}^\top\boldsymbol{z}^+/\tau\right)}\right)\right] \\
&= \mathbb{E}_{\boldsymbol{x},\boldsymbol{x}^+,\boldsymbol{x}_1^-,\ldots,\boldsymbol{x}_N^-}\left[\log\left(1+\sum_k\frac{\exp\left(\boldsymbol{z}^\top\boldsymbol{z}_k^-/\tau\right)}{\exp\left(\boldsymbol{z}^\top\boldsymbol{z}^+/\tau\right)}\right)\right] \\
&= \mathbb{E}_{\boldsymbol{x},\boldsymbol{x}^+,\boldsymbol{x}_1^-,\ldots,\boldsymbol{x}_N^-}\left[\log\left(1+\sum_k\exp\left(\frac{\boldsymbol{z}^\top\boldsymbol{z}_k^- - \boldsymbol{z}^\top\boldsymbol{z}^+}{\tau}\right)\right)\right].
\end{aligned} \tag{10}$$

This is lower-bounded by 0 when the inner term $\exp(\cdot) \to 0$, which occurs when $\boldsymbol{z}^\top\boldsymbol{z}^+ = -\boldsymbol{z}^\top\boldsymbol{z}_k^- = \infty$. In practice, τ helps ensure that the norm of the resultant encodings is not too large. Even when $||\boldsymbol{z}||$ is not large, the exponential term can still grow to ∞, causing numerical instability. In practice, we constrain it by upper-bounding the term inside the exponential by m (see table 4 for exact numbers):

$$\mathcal{L}_{\texttt{contrastive}} = \mathbb{E}_{\boldsymbol{x},\boldsymbol{x}^+,\boldsymbol{x}_1^-,\ldots,\boldsymbol{x}_N^-}\left[\log\left(1+\sum_k\exp\left(\max\left(\frac{\boldsymbol{z}^\top\boldsymbol{z}_k^- - \boldsymbol{z}^\top\boldsymbol{z}^+}{\tau}, m\right)\right)\right)\right]. \tag{11}$$

This leads to the final contastive learning loss we used in practice in place of equation 6:

$$\mathcal{L}_{\texttt{model}} = \mathbb{E}_{s_t,a_t,s_{t+1}}\left[-\sum_{i\in v_{t+1}}\log p(\rho_{t+1,i}^g|s_t^o,a_t).\right] \tag{12}$$

$$p(\rho_{t+1,i}^g|s_t^o,a_t) = \left(1+\sum_k\exp\left(\max\left(\frac{F(s_t^o,\rho_{t,i}^g,a_t)^\top\boldsymbol{z}_{k,-}^o - F(s_t^o,\rho_{t,i}^g,a_t)^\top\boldsymbol{z}_+^{o,i}}{\tau}, m\right)\right)\right) \tag{13}$$

Networks	Parameters
	Relational Object-DQN
Activation fn. (AF)	Leaky ReLU (LR)
$f^{\texttt{ego}}_{\texttt{enc}}$	Conv(32-8-4)-AF-Conv(64-4-2)-AF-Conv(64-3-1)-AF-MLP(9216-512)-AF
$f^{o}_{\texttt{enc}}$	Conv(32-4-2)-AF-Conv(64-4-2)-AF-Conv(64-4-2)-AF-MLP(4096-512)-AF
$f^{\texttt{loc}}_{\texttt{enc}}$	MLP(6-256)-AF-MLP(256-256)-AF
$\widehat{Q}_{\texttt{int}}(o_i)$	MLP(1280-256)-AF-MLP(256-256)-AF-MLP(256-8)
$\widehat{Q}_{\texttt{nav}}$	MLP(768-256)-AF-MLP(256-256)-AF-MLP(256-8)
$\mathcal{R}(\boldsymbol{z}^{o,i}, \boldsymbol{Z}^{o}) : W^k_1, W^q_1$	MLP(512-64), MLP(512-64)
$\mathcal{R}(\boldsymbol{z}^{\kappa}, \boldsymbol{Z}^{o}) : W^k_2, W^q_2$	MLP(768-64), MLP(512-64)
	Object-centric model
$f_{\texttt{model}}$	MLP(1088-256)-AF-MLP(256-512)
$\boldsymbol{z}^a : W^o, W^b$	MLP(512-64), MLP(8-64)
	Scene-centric model
$f^{\kappa}_{\texttt{model}}$	MLP(832-512)
$\boldsymbol{z}^a : W^o, W^b$	MLP(512-64), MLP(8-64)
	VAE
$f_{\texttt{recon}}$	MLP(4096-512)-AF-Conv(64-4-2)-AF-Conv(64-4-2)-AF-Conv(32-3-2)

Table 3: Architectures used across all experiments.

B Agent Details

B.1 Architectures and objective functions

We present the details of the architecture used for all models in table 3. All models shared the Relational Object-DQN as their base. We built the Relational Object-DQN using the rlkit open-source reinforcement-learning library.

Relational Object-DQN. This is our base architecture. Aside from the details in the main text, we note that $f^{\kappa}_{\texttt{enc}}$ is the concatenation of one function which encodes image information and another that encodes location information: $f^{\kappa}_{\texttt{enc}}(s^{\kappa}) = f^{\kappa}_{\texttt{enc}}(s^{\texttt{ego}}, s^{\texttt{loc}}) = [f^{\texttt{ego}}_{\texttt{enc}}(s^{\texttt{ego}}), f^{\texttt{loc}}_{\texttt{enc}}(s^{\texttt{loc}})]$.

Relational Object-DQN + COBRA Object-Model: each agent predicts the latent factors that have generated each individual object-image-patch. This requires an additional reconstruction network for the object-encoder, $f_{\texttt{recon}}(\boldsymbol{z}^{o,i}_t)$, which produces an object-image-patch back from an encoding and a prediction network $f_{\texttt{cobramodel}}$ that produces the object-encoding for $\boldsymbol{z}^{o,i}_t$ at the next time-step. The objective function is:

$$\mathcal{L}_{\texttt{recon}} = \mathbb{E}_{s_t}\left[\sum_{i \in v_t} ||f_{\texttt{recon}}(\boldsymbol{z}^{o,i}_t) - o_{t,i}||^2_2 - \beta^{\texttt{kl}}\,\mathrm{KL}(p(\boldsymbol{z}^{o,i}_t|o_{t,i})||p(\boldsymbol{z}^{o,i}_t))\right] \tag{14}$$

$$\mathcal{L}_{\texttt{pred}} = \mathbb{E}_{s_t}\left[\sum_{i \in v_t} ||f_{\texttt{recon}}(f_{\texttt{cobramodel}}(\boldsymbol{z}^{o,i}_t)) - o_{t+1,i}||^2_2\right] \tag{15}$$

$$\mathcal{L}_{\texttt{cobra}} = \mathcal{L}_{\texttt{recon}} + \mathcal{L}_{\texttt{pred}} - \mathbb{E}_{s_t}\left[\sum_{i \in v_t} \beta^{\texttt{kl}}\,\mathrm{KL}(p(\boldsymbol{z}^{o,i}_t|o_{t,i})||p(\boldsymbol{z}^{o,i}_t))\right] \tag{16}$$

where KL is the Kullback-Leibler Divergence and $p(\boldsymbol{z}^{o,i}_t)$ is an isotropic, unit gaussian. We also model $p(\boldsymbol{z}^{o,i}_t|o_{t,i})$ as a gaussian. We augment the Relational Object-DQN so that $\boldsymbol{z}^{o,i}_t$ is the mean of the gaussian and so that a standard deviation is also computed. Please see Higgins et al. (2017) for more.

Relational Object-DQN + OCN: the agent tries to learn encodings of object-image-patches such that patches across time-steps corresponding to the same object are grouped nearby in latent space, and patches corresponding to different objects are pushed apart. This also relied on contrastive learning, except that it uses it on image-pairs across time-steps. Following Pirk et al. (2019), the anchor is defined as the object-encoding $\boldsymbol{z}_t^{o,i} = f(o_{t,i})$, which we will refer to as f. The positive is defined as the object-image-patch encoding at the next time-step with lowest $L2$ distance in latent space, $f^+ = \arg\min_{z_{t+1}^{o,j}} ||\boldsymbol{z}_t^{o,i} - \boldsymbol{z}_{t+1}^{o,j}||^2$. We then set negatives $\{f_k^-\}$ as the object-image-patches that did not correspond to the match. We note that augmenting Pirk et al. (2019) so that their objective function had temperature τ was required for good performance. For a unified perspective with our own objective function, we write their n-tuplet-loss with a softmax (see Sohn (2016) for more details on their equivalence). The objective function is:

$$\mathcal{L}_{\texttt{ocn}} = \mathbb{E}_{s_t, s_{t+1}} \left[-\log \left(\frac{\exp\left(f^\top f^+ / \tau\right)}{\exp\left(f^\top f^+ / \tau\right) + \sum_k \exp\left(f^\top f_k^- / \tau\right)} \right) \right] \tag{17}$$

Relational Object-DQN + Ground Truth Object Info: the agent doesn't have an auxilliary task and doesn't encode object-images. Instead it encodes object-information. For each object, we replace object-image-patches with the following information available in Thor:

1. the object's category (i.e. its type index)
2. the object's unique instance index (i.e. its token index)
3. the object this object was in (e.g. if this object is a cup in the sink, this would correspond to the sink index)
4. distance to object (in meters)
5. whether object is visible (boolean)
6. whether object is toggled (boolean)
7. whether object is broken (boolean)
8. whether object is filledWithLiquid (boolean)
9. whether object is dirty (boolean)
10. whether object is cooked (boolean)
11. whether object is sliced (boolean)
12. whether object is open (boolean)
13. whether object is pickedUp (boolean)
14. object temperature (cold, room-temperature, hot)

B.2 Hyperparameter Search

Relational Object-DQN. All models are based on the same Relational Object-DQN agent and thus use the same hyperparameters. We searched over these parameters using "Relational Object-DQN + Ground-Truth Object-Information". We searched over tuples of the parameters in the "DQN" portion of table 4. In addition to searching over those parameters, we searched over "depths" and hidden layer size of the multi-layer perceptrons $f_{\texttt{enc}}^{\texttt{loc}}$, $\widehat{Q}_{\texttt{int}}(o_i)$, and $\widehat{Q}_{\texttt{nav}}$. For depths, we searched uniformly over $[0, 1, 2]$ and for hidden later sizes we searched uniformly over $[128, 256, 512]$. We searched over 12 tuples on the "Fill Cup with Water" task and 20 tuples on the "Place Apple on Plate & Both on Table" task. We found that task-performance was sensitive to hyperparameters and choose hyperparameters that achieved a $90\%+$ success rate on both tasks. We fixed these settings and searched over the remaining values for each auxilliary task.

Attentive Object-Model. In addition to experimenting with using a dot-product as our kernel $\boldsymbol{u}^\top \boldsymbol{w}$, due to the inherent numerical instability of equation 9, we also experimented with using cosine similarity and applying an L2 norm on the vectors. While the latter has been recommended by prior work Sohn (2016), we found that applying temperature τ as recent work has He et al. (2019); Warde-Farley et al. (2018) was superior. We experimented with the number of negative examples used for the contrastive loss and found no change in performance. We performed a search over 4

Hyperparameter	Final Value	Values Considered
Max gradient norm	0.076	log-uniform$(10^{-4}, 10^{-1})$
DQN		
Learning rate η_1	1.8×10^{-5}	log-uniform$(10^{-6}, 10^{-2})$
Target Smoothing Coeffecient η_2	0.00067	log-uniform$(10^{-6}, 10^{-3})$
Discount γ	0.99	
Training ϵ annealing	$[1, .1]$	
Evaluation ϵ	.1	
Replay Buffer Size	200000	
Batchsize	50	
Attentive Object-Model		
upper-bound m	85	-
Number of Negative Examples	20	-
temperature τ	8.75×10^{-5}	log-uniform$(10^{-6}, 10^{-3})$
Loss Coeffecient $\beta^{\texttt{model}}$	10^{-3}	-
Cobra Object-Model		
KL Coeffecient $\beta^{\texttt{kl}}$	26	log-uniform$(10^{-1}, 10^{2})$
Loss Coeffecient $\beta^{\texttt{cobra}}$	0.0032	log-uniform$(10^{-4}, 1)$
OCN		
temperature τ	5×10^{-5}	log-uniform$(10^{-6}, 10^{-3})$
Loss Coeffecient $\beta^{\texttt{ocn}}$	0.0047	log-uniform$(10^{-4}, 10^{-2})$

Table 4: Hyperparameters shared across all experiments.

tuples from the values in table 4. We chose the loss-coefficient as the the coefficient which put the object-centric model loss at the same order of magnitude as the DQN loss.

COBRA Object-Model, OCN. For each auxilliary task, we performed a search over 6 tuples from the values in table 4. For each loss, we chose loss coefficients that scaled the loss so they were between an order of magnitude above and below the DQN loss.

C THOR IMPLEMENTATION DETAILS

C.1 THOR SETTINGS

Environment. While AI2Thor has multiple maps to choose from, we chose "Floorplan 24". To reduce the action-space, we restricted the number of object-types an agent could interact with so that there were 10 distractor types beyond task relevant object-types. We defined task-relevant object-types as objects needed to complete the task or objects they were on/inside. For example, in "Place Apple on Plate & Both on Table", since the plate is on a counter, counters are task object-types. We provide a list of the object-types present in each task with the task descriptions below.

Observation. Each agent observes an 84×84 grayscale image of the environment, downsampled from a 300×300 RGB image. They can detect up to 20 obects per time-step within its line of sight, if they exceed 50 pixels in area, regardless of distance. Each object in the original 300×300 scene image is cropped and resized to a 32×32 grayscale image[1]. Each agent observes its (x, y, z) location, and its pitch, yaw, and roll body rotation $(\varphi_1, \varphi_2, \varphi_3)$ in a global coordinate frame.

Episodes. The episode terminates either after 500 steps or when a task is complete. The agent's spawning location is randomly sampled from the 81 grid positions facing North with a body angle

[1]For "Slice" tasks and "Make Tomato & Lettuce Salad", we used an object image size of 64×64 to facilitate recognition of smaller objects. We decreased the replay buffer to have 120000 samples.

$(0°, 0°, 0°)$. Each agent recieves reward 1 if a task is completed successfully and a time-step penalty of -0.04.

Setting	Values
Observation Size	300×300
Downsampled Observation Size	84×84
Object Image Size	32×32
Min Bounding Box Proportion	$\frac{50}{300 \times 300}$
Max Interaction Distance	1.5m

Table 5: Settings used in Thor across experiments.

C.2 Task Details

For each task, we describe which challenges were present, what object types were interactable, and the total Key Semantic Actions available. We chose objects that were evenly spaced around the environment. The challenges were:

Challenge A: the need for view-invariance (e.g. recognizing a knife across angles),

Challenge B: the need to reason over ≥ 3 objects,

Challenge C: the need to recognize and use *combined* objects (e.g. filling a cup with water in the sink or toasting bread in a toaster).

Slice Bread.
Challenges:
A: recognizing the knife across angles.
Interactable Object Types: 15
- CounterTop: 3, DiningTable: 1, Microwave: 1, Plate: 1, CoffeeMachine: 1, Bread: 1, Fridge: 1, Egg: 1, Cup: 1, Pot: 1, Pan: 1, Tomato: 1, Knife: 1

Key Semantic Actions:
1. Go to Knife
2. Pickup Knife
3. Go to Bread
4. Slice Bread

Slice Lettuce and Tomato. (order doesn't matter)
Challenges:
A: recognizing the knife across angles.
B: ecognizing and differentiate 3 task objects: the knife, lettuce, and tomato. As each object is cut, the agent needs to choose from more objects as it can select from the object-slices.
Interactable Object Types: 17
- CounterTop: 3, DiningTable: 1, Microwave: 1, Plate: 1, CoffeeMachine: 1, Bread: 1, Fridge: 1, Spatula: 1, Egg: 1, Cup: 1, Pot: 1, Pan: 1, Tomato: 1, Lettuce: 1, Knife: 1

Key Semantic Actions:
1. Go to Knife
2. Pickup Knife
3. Go to Table
4. Slice Lettuce
5. Slice Tomato

Slice Lettuce and Apple, and Potato. (order doesn't matter)
Challenges:
A: recognizing the knife across angles.
B: recognizing and differentiate 4 task objects: the knife, lettuce, and apple, and potato. As each object is cut, the agent needs to choose from more objects as it can select from the object-slices.
Interactable Object Types: 18

- CounterTop: 3, DiningTable: 1, Microwave: 1, Plate: 1, CoffeeMachine: 1, Bread: 1, Fridge: 1, Potato: 1, Egg: 1, Cup: 1, Pot: 1, Pan: 1, Tomato: 1, Lettuce: 1, Apple: 1, Knife: 1

Key Semantic Actions:

1. Go to Knife
2. Pickup Knife
3. Go to Table
4. Slice Lettuce
5. Slice Apple
6. Slice Potato

Cook Potato on Stove.

Challenges:

A: recognizing the stove across angles.
B: needs to differentiate 3 objects: the stove knob, pot, and potato.
C: recognizing the potato in the pot.

Interactable Object Types: 21

1. StoveBurner: 4, StoveKnob: 4, DiningTable: 1, Microwave: 1, Plate: 1, CoffeeMachine: 1, Bread: 1, Fridge: 1, Potato: 1, Egg: 1, Cup: 1, Pot: 1, Pan: 1, Tomato: 1, Knife: 1

Key Semantic Actions:

1. Go to Potato
2. Pickup Potato
3. Go to Stove
4. Put Potato in Pot
5. Turn on Stove Knob

Fill Cup with Water.

Challenges:

A: recognizing the cup across angles and backgrounds.
B: recognizing the cup in the sink.
C: the need to recognize and use *combined* objects (e.g. filling a cup with water in the sink or toasting bread in a toaster).

Interactable Object Types: 18

- CounterTop: 3, Faucet: 2, Sink: 1, DiningTable: 1, Microwave: 1, CoffeeMachine: 1, Bread: 1, Fridge: 1, Egg: 1, Cup: 1, SinkBasin: 1, Pot: 1, Pan: 1, Tomato: 1, Knife: 1

Key Semantic Actions:

1. Go to Cup
2. Pickup Cup
3. Go to Sink
4. Put Cup in Sink
5. Fill Cup

Toast Bread Slice.

Challenges:

A: recognizing the toaster across angles.
C: recognizing the bread slice in the toaster.

Interactable Object Types: 21

- BreadSliced: 5, CounterTop: 3, Bread: 2, DiningTable: 1, Microwave: 1, CoffeeMachine: 1, Fridge: 1, Egg: 1, Cup: 1, Pot: 1, Pan: 1, Tomato: 1, Knife: 1, Toaster: 1

Key Semantic Actions:

1. Go to Bread Slice
2. Pickup Bread Slice
3. Go to Toaster
4. Put Breadslice in Toaster
5. Turn on Toaster

Place Apple on Plate & Both on table.

Challenges:

A: recognizing the plate across occlusions.
B: needs to differentiate 3 objects: the apple, plate, and table.
C: recognizing the apple on the plate.

Interactable Object Types: 16

- CounterTop: 3, DiningTable: 1, Microwave: 1, Plate: 1, CoffeeMachine: 1, Bread: 1, Fridge: 1, Spatula: 1, Egg: 1, Cup: 1, Pot: 1, Pan: 1, Apple: 1, Knife: 1

Key Semantic Actions:

1. Go to Apple
2. Pickup Apple
3. Put Apple on Plate
4. Pickup Plate
5. Go to Table
6. Put Plate on Table

Make Tomato & Lettuce Salad.

Challenges:

A: recognizing the plate across occlusions.
B: needs to differentiate 3 objects: the tomato slice, lettuce slice, and plate.
C: recognizing the tomato slice or lettuce slice on the plate.

Interactable Object Types: 32

- TomatoSliced: 7, LettuceSliced: 7, CounterTop: 3, Bread: 2, DiningTable: 1, Microwave: 1, Plate: 1, CoffeeMachine: 1, Fridge: 1, Spatula: 1, Egg: 1, Cup: 1, Pot: 1, Pan: 1, Tomato: 1, Lettuce: 1, Knife: 1

Key Semantic Actions:

1. Go to table
2. Pickup tomato or lettuce slice
3. Put slice on Plate
4. Pickup other slice
5. Put slice on Plate

C.3 Interaction Dataset

In order to measure and analyze the quality of the object representations learned via each auxilliary task, we created a dataset with programmatically generated object-interactions and with random object-interactions. This enabled us to have a diverse range of object-interactions and ensured the dataset had many object-states present.

Programatically Generated object-interactions. This dataset contains programmatically generated sequences of interactions for various tasks. The tasks currently supported by the dataset include: *pickup X, turnon X, open X, fill X with Y, place X in Y, slice X with Y, Cook X in Y on Z.* For each abstract task type, we first enumerate all possible manifestations based on the action and object properties. For example, manifestations of open X include all objects that are openable. We exhaustively test each manifestation and identify the ones that are possible under the physics of the environment. We explicitly build the action sequence required to complete each task. Because we only want to collect object-interactions, we use the high level "TeleportFull" command for navigation to task objects. The TeleportFull command allows each agent to conveniently navigate to desired task objects at a particular location and viewing angle. For example, the sequence for place X in Y is: TeleportFull to X, Pickup X, TeleportFull to Y, and Put X in Y. An agent will execute each action until termination. We collect both successful and unsuccessful task sequences. There is a total of 156 unique tasks in the dataset and 1196 individual task sequences amounting to 2353 $(\text{state}, \text{action}, \text{next state})$ tuples.

Random object-interactions. The random interaction dataset consists of $(\text{state}, \text{action}, \text{next state})$ tuples of random interactions with the environment. An agent equipped with a random action policy interacts with the environment for episodes of 500 steps until it collects a total of 4000 interaction samples.

Training. We divided the data into an 80/20 training/evaluation split and trained for 2000 epochs. We reported the test data results.