SIR: Structured Image Representations for Explainable Robot Learning

Supplementary Material

7. Additional information

Source code will be made publicly available upon acceptance. Our GNN implementation employs residual connections and normalization to mitigate oversmoothing, a phenomenon to which small, fully-connected graphs are particularly susceptible.

7.1. Pre-Trained Models

The pre-trained models for generating Cropped-Image-Features and Point-Cloud-Features were trained using human demonstration data from RoboCasa [19]. We extracted all objects from the scene at every 10th time step to train these vision networks using a reconstruction loss. For Cropped-Image-Features alone, we employ a ResNet18 [9] backbone with an embedding dimension of 256. However, when combining Cropped-Image-Features with BB-Coordinates, we use a smaller ResNet8 backbone with an embedding dimension of 37. This lower dimension was chosen specifically to match the size of the object label vectors (one-hot encoded length of 37), allowing the visual features to serve as a direct replacement for symbolic labels.

7.2. Graph Sparsification

Usually, graph sparsification methods are based on the Information Bottleneck (IB) principle [40] or an L0-regularization (or L1/L2) on the edge weights [39] (in our case: node weights). However, we empirically find that such methods are not applicable in our case, as they either keep all or zero nodes in the sub-graph. Regarding our finding that the dataset is biased, we attribute this phenomenon to the lack of discriminability of the information content of the node features. I.e., since there are strong correlations between objects' positions in the scene (or the position is not relevant at all), there is no extreme advantage of choosing one node over another, which makes it difficult to select one node over another to keep in the sub-graph. This is further complicated by the necessity for hard masks, which can hinder differentiability and optimizations, but are required in interpretability-seeking settings. This demonstrates the necessity for our soft histogram loss, which encourages an explicit ranking for the nodes, and the TopK-selection mechanism to ensure that there are neither all nor zero nodes in the sub-graphs.

8. Additional Evaluation Results

The following sections present results for SIR using BC, generated graphs and the CALVIN benchmark [18]. Additionally, we provide fine-grained performance metrics for all 24 RoboCasa tasks and tabular results for the distractor experiments. Table 15 displays the average results of all models using MDT as the action generator.

Table 4. Ablation results on RoboCasa using generated graphs, which are predicted using a fine-tuned DETR model. Models are not trained on these generated graphs, these are only swapped for ground truth information during rollout.

Task	Cropped-Img	BB + Crop-Img
Pick/Place (8)	0.06	0.06
Doors (4)	11.75	18.13
Drawers (2)	34.00	28.75
Knobs (2)	9.00	12.00
Levers (3)	39.67	35.83
Buttons (3)	12.33	10.50
Insert (2)	3.25	3.25
Avg (24)	12.33	12.50

8.1. Generated Graphs Results

This approach is called *generated graphs* and relies on a fine-tuned DETR for graph generation. The DETR model is used during rollout to predict the objects in a given scene and their segmentation masks, which the bounding boxes can be derived from. We evaluated using generated graphs on FC-Graph on RoboCasa using the Split-View approach. Models were not trained on the generated graphs, but on the ground truth information. This introduces a distribution shift, because the generated graphs will be imperfect, which is also visible in the performance drop, seen in Table 4. Compared to using ground truth graphs, the models drop over 4 percent for only using Cropped-Image-Features and 3.4 percent for using BB-Coordinates and Cropped-Image-Features. These results could have two causes, either the model can not handle the distribution shift from ground truth to generated graphs or generated graphs in general will lead to a worse performance. In future experiments, we also want to **train** networks on generated graphs to have a clear answer to this observation.

8.2. RoboCasa - Behaviour Cloning Results

The results for the BC-Transformer as action generation model in RoboCasa can be seen in Table 5. Using images as observation input only decreases the average result slightly compared to using MDT as the action generation model. In comparison, using graphs and BC-Transformer, performance drops heavily, for all approaches. But using SIR still results in the highest average success rate of 16.27

Table 5. Results for using BC-Transformer as the action generator on RoboCasa across all 24 tasks.

Feature Input	Pick/Place (8)	Doors (4)	Drawers (2)	Knobs (2)	Levers (3)	Buttons (3)	Insert (2)	**Avg (24)**
Baselines								
Image	**0.94**	25.63	36.50	5.75	22.33	**24.00**	**7.00**	14.48
Split-View Graph - Fully-Connected								
Cropped-Img	0.13	12.63	31.50	11.75	34.50	6.00	5.75	11.25
BB-Coordinates + Label	0.00	11.88	23.00	**16.25**	15.00	0.00	0.25	7.33
BB-Coordinates + Cropped-Img	0.06	9.00	24.75	6.50	36.33	5.33	3.00	9.58
Fusion Graph - Fully-Connected								
BB-Coordinates + Cropped-Img	0.25	18.13	34.5	11.25	31.83	5.0	0.3	11.77
Sparse Graph Methods								
TopK	0.31	17.25	36.75	10.0	37.83	6.67	2.25	12.63
SIR (Ours)	0.38	**29.63**	**42.25**	14.5	**43.33**	6.83	2.5	**16.27**

Table 6. Completion scores on CALVIN using only the static camera as observation D $\rightarrow$ D for 100 Rollouts using two seeds for each model. The maximum completion number is 5.

Model	**Task Completion (Max 5)**
Baseline	
Image	$\mathbf{1.5 \pm 0.2}$
Fully-Connected Graphs	
Cropped-Img	1.3 ± 0.02
BB-Coordinates + Cropped-Img	1.2 ± 0.03
Sparse Graph Methods	
TopK	1.4 ± 0.02

percent. These observation lead to the conclusion that graph observation can be better utilized by diffusion-based methods compared to BC-based methods, but overall achieve a higher result compared to image-based models regardles of training objective.

8.3. CALVIN - MDT Results

CALVIN [18] serves as our second evaluation benchmark. The results, presented in Table 6, report the average number of tasks completed out of a possible five per rollout. Image-based models achieve an average of 1.5 tasks, slightly outperforming the sparse TopK graph method, which completes 1.4 tasks. It is important to note that our experiments on the CALVIN environment are still preliminary. We hypothesize that further fine-tuning of the sparsification methods could yield performance gains similar to those observed in RoboCasa.

8.4. Distractor Objects

The introduction of distractor objects in RoboCasa leads to a clear decline in performance for both the image baseline and the TopK sparsification approach, displayed in Table 7. While success rates for the *Pick and Place* task appear to increase for all models except the image baseline, the absolute scores are too low to be considered conclusive. In the *Insert* task, performance decreases across the board, though graph-based models exhibit significantly smaller degradation.

8.5. Per Task Results

High average reward for the grouped tasks from the RoboCasa [19] paper does not mean that the model performs well in each subtask. Therefore, we included all single task results in the following tables: *Pick and Place* in Table 8, *Doors* in Table 9, *Drawers* in Table 10, *Knobs* in Table 11, *Levers* in Table 12, *Buttons* in Table 13 and *Insert* in Table 14. Each task grouping (except *Pick and Place*) includes tasks which are easier solvable and harder tasks. SIR only performs best on 4 single atomic tasks, which indicates that the higher average performance is distributed across all single tasks.

9. Additional Explainability Results

We further present additional explainability results for SIR on 6 tasks in RoboCasa, not present in the main paper, as well as a Grad-CAM visualization of the image baseline models.

9.1. Sub-Graph Statistics

The generated explanations by the different models are presented in Tab. 16, Tab. 17, Tab. 18, Tab. 19, Tab. 20, Tab. 21, Tab. 22. Each column represents one of the three possibilities for the network to extract the desired sub-graph. The results show that the four examples in Fig. 5 are not cherry-picked, but that over all tasks a specific trend can be observed.

9.2. Sub-Graph Visualizations

The observed explanations fall into two distinct categories based on the model's adherence to pre-defined important

Table 7. Distractor objects: Success rate over 100 rollouts. Higher values indicate better performance.

Feature Input	Pick/Place (8)	Doors (4)	Drawers (2)	Knobs (2)	Levers (3)	Buttons (3)	Insert (2)	**Avg (24)**
Baseline								
Image	**1.19**	25.13	**49.75**	7.25	23.67	17.00	**4.75**	14.81
Image w/ Distractors	0.56	18.25	46.25	3.25	17.17	14.50	2.50	11.52
Fully Connected Graph								
FC-Graph	0.06	28.62	39.25	14.00	40.00	18.83	**4.75**	16.98
FC-Graph w/ Distractors	0.38	21.12	40.00	17.75	41.50	18.67	3.75	16.29
Sparse Graph Methods								
Threshold	0.00	28.38	47.25	11.00	39.50	19.17	3.00	17.17
Threshold w/ Distractors	0.40	22.50	44.50	12.75	40.33	22.17	2.50	16.67
TopK	0.12	**31.37**	45.00	19.50	41.17	18.17	4.50	18.44
TopK w/ Distractors	0.38	22.75	44.75	14.25	36.50	14.50	4.00	15.54
SIR	0.12	30.25	46.25	16.50	48.50	**21.83**	**4.75**	**19.50**
SIR w/ Distractors	0.56	23.12	47.50	**21.75**	**51.83**	20.67	4.25	19.23

Table 8. The results represent success rate over 100 rollouts with 2 models trained on different seeds for Pick and Place tasks.

Feature Input	PnPCab ToCounter	PnPCounter ToCab	PnPMicrowave ToCounter	PnPCounter ToMicrowave	PnPSink ToCounter	PnPCounter ToSink	PnPStove ToCounter	PnPCounter ToStove	**Average**
Baselines									
Image	3.0	1.5	2.5	0.0	0.0	0.5	0.5	1.5	1.19
Image + FiLM	1.0	0.0	0.0	0.5	0.0	0.25	0.5	0.5	0.34
Own Pretrained Image	0.5	0.0	0.0	0.0	0.0	0.5	0.0	0.0	0.13
Point Clouds	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Image + Point Clouds	1.5	0.0	0.0	0.5	0.5	1.0	0.5	0.5	0.56
Split-View Graph - Fully-Connected									
Cropped-Img	1.0	0.0	0.5	0.0	0.0	0.0	0.5	0.0	0.25
BB-Coordinates + Label	0.5	0.0	0.0	0.0	0.0	0.5	0.0	0.0	0.13
BB-Coordinates + Cropped-Img	1.5	0.0	0.0	0.0	0.5	0.0	0.0	0.0	0.25
Point Clouds	0.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.06
Label + Point Clouds	1.0	0.5	0.0	0.0	0.5	0.67	0.5	0.0	0.40
Cropped-Img + Point Clouds	1.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.25
Fusion Graph - Fully-Connected									
BB-Coordinates + Label	0.0	0.0	0.0	0.0	0.0	0.5	0.0	0.0	0.06
BB-Coordinates + Cropped-Img	0.0	0.0	0.0	0.0	0.5	0.0	0.0	0.0	0.06
Sparsification Methods									
Random Node Removal	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Naive NR (no soft histogram loss)	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Threshold	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
TopK	0.5	0.0	0.0	0.0	0.0	0.5	0.0	0.0	0.12
SIR (Ours)	0.5	0.0	0.0	0.0	0.5	0.0	0.0	0.0	0.12

objects. The first group (Figs. 6 to 8) demonstrates consistent reliance on the designated important nodes. In contrast, the second group (Figs. 9 to 11) exhibits significant variance across training seeds. While *PandaGripper* and *PandaMobile* remain constant, other object selections appear arbitrary. This suggests that in tasks where the model diverges from the expected nodes yet maintains high performance, it is exploiting underlying dataset biases to solve the task. Crucially, this inference is valid only for tasks where the model outperforms baselines, as arbitrary focus in low-performing models likely indicates a failure to learn the task.

9.3. Grad-CAM Visualizations

We visualize four steps of a rollout using the image baseline on the *CloseDrawer* task in Figure 12. The model's focus shifts between the robot arm, gripper, and counter/drawer. Over the whole rollout the focus includes nearly the entire image. This lack of precision renders such visualizations ineffective for interpreting model behaviour or gaining insights into the dataset.

Table 9. Door tasks: success rate over 100 rollouts with 2 models trained on different seeds.

Feature Input	Close Single Door	Open Single Door	Close Double Door	Open Double Door	**Average**
Baselines					
Image	63.0	6.0	30.0	1.5	25.13
Image + FiLM	64.5	7.5	36.0	3.0	27.75
Own Pretrained Image	55.5	4.0	8.0	0.5	17.00
Point Clouds	0.0	0.0	0.0	0.0	0.00
Image + Point Clouds	59.0	6.0	33.5	2.5	25.25
Split-View Graph - Fully-Connected					
Cropped-Img	48.5	16.0	17.5	1.5	20.88
BB-Coordinates + Label	54.0	3.5	18.5	0.5	19.13
BB-Coordinates + Cropped-Img	59.0	14.5	18.0	1.0	23.13
Point Clouds	21.5	0.0	24.0	0.0	11.38
Label + Point Clouds	24.5	6.5	31.5	1.0	15.88
Cropped-Img + Point Clouds	39.5	10.0	26.0	2.0	19.38
Fusion Graph - Fully-Connected					
BB-Coordinates + Label	47.5	3.0	6.0	0.0	14.12
BB-Coordinates + Cropped-Img	64.0	18.5	29.0	3.0	28.62
Sparsification Methods					
Random Node Removal	9.0	7.0	6.5	0.0	5.62
Naive NR (no soft histogram loss)	33.5	0.5	0.0	0.0	8.50
Threshold	61.5	17.5	32.0	2.5	28.38
TopK	60.0	25.0	35.0	5.5	31.37
SIR (Ours)	65.5	22.0	33.0	0.5	30.25

Table 10. Drawer tasks: success rate over 100 rollouts with 2 models trained on different seeds.

Feature Input	Close Drawer	Open Drawer	**Average**
Baselines			
Image	85.5	14.0	49.75
Image + FiLM	80.5	10.0	45.25
Own Pretrained Image	68.5	10.0	39.25
Point Clouds	2.0	0.0	1.0
Image + Point Clouds	82.0	7.5	44.75
Split-View Graph - Fully-Connected			
Cropped-Img	67.5	7.0	37.25
BB-Coordinates + Label	61.0	2.0	31.5
BB-Coordinates + Cropped-Img	63.0	7.5	35.25
Point Clouds	51.0	7.5	29.25
Label + Point Clouds	57.5	7.0	32.25
Cropped-Img + Point Clouds	70.0	11.5	40.75
Fusion Graph - Fully-Connected			
BB-Coordinates + Label	55.0	8.0	31.5
BB-Coordinates + Cropped-Img	75.0	3.5	39.25
Sparsification Methods			
Random Node Removal	18.5	0.5	9.5
Naive NR (no soft histogram loss)	46.0	1.0	23.5
Threshold	87.5	7.0	47.25
TopK	81.5	8.5	45.0
SIR (Ours)	80.5	12.0	46.25

Table 11. Stove tasks: success rate over 100 rollouts with 2 models trained on different seeds.

Feature Input	TurnOn Stove	TurnOff Stove	**Average**
Baselines			
Image	12.5	2.0	7.25
Image + FiLM	17.0	5.0	11.00
Own Pretrained Image	9.5	5.0	7.25
Point Clouds	1.0	1.5	1.25
Image + Point Clouds	9.0	2.0	5.50
Split-View Graph - Fully-Connected			
Cropped-Img	25.0	5.0	15.00
BB-Coordinates + Label	21.5	5.5	13.50
BB-Coordinates + Cropped-Img	21.5	7.0	14.25
Point Clouds	28.0	9.0	18.50
Label + Point Clouds	23.5	6.5	15.00
Cropped-Img + Point Clouds	12.0	5.0	8.50
Fusion Graph - Fully-Connected			
BB-Coordinates + Label	17.0	9.0	13.00
BB-Coordinates + Cropped-Img	22.5	5.5	14.00
Sparsification Methods			
Random Node Removal	8.0	2.5	5.25
Naive NR (no soft histogram loss)	8.5	5.0	6.75
Threshold	15.5	6.5	11.0
TopK	26.0	13.0	19.50
SIR (Ours)	26.5	6.5	16.50

Table 12. Sink tasks: success rate over 100 rollouts with 2 models trained on different seeds.

Feature Input	TurnOn Sink Faucet	TurnOff Sink Faucet	Turn Sink Spout	**Average**
Baselines				
Image	22.0	24.0	25.0	23.67
Image + FiLM	32.0	32.5	34.0	32.83
Own Pretrained Image	11.5	25.5	19.5	18.83
Point Clouds	27.5	2.5	29.5	19.83
Image + Point Clouds	21.5	18.5	22.0	20.67
Split-View Graph - Fully-Connected				
Cropped-Img	36.0	64.5	44.5	48.33
BB-Coordinates + Label	12.0	17.5	34.0	21.17
BB-Coordinates + Cropped-Img	30.5	51.5	39.0	40.33
Point Clouds	15.5	24.0	51.5	30.33
Label + Point Clouds	26.0	31.5	53.0	36.83
Cropped-Img + Point Clouds	24.5	55.0	43.0	40.83
Fusion Graph - Fully-Connected				
BB-Coordinates + Label	11.5	25.5	44.5	27.17
BB-Coordinates + Cropped-Img	23.0	54.5	42.5	40.00
Sparsification Methods				
Random Node Removal	5.0	21.5	24.5	17.00
Naive NR (no soft histogram loss)	15.0	33.0	39.5	29.17
Threshold	20.5	54.5	43.5	39.5
TopK	22.0	55.0	46.5	41.17
SIR (Ours)	31.5	69.0	45.0	48.50

Table 13. Button tasks: success rate over 100 rollouts with 2 models trained on different seeds.

Feature Input	TurnOff Microwave	TurnOn Microwave	Coffee PressButton	**Average**
Baselines				
Image	22.0	15.0	14.0	17.00
Image + FiLM	27.5	14.5	8.5	16.83
Own Pretrained Image	9.5	9.5	11.5	10.17
Point Clouds	24.0	5.5	3.0	10.83
Image + Point Clouds	17.5	15.0	13.0	15.17
Split-View Graph - Fully-Connected				
Cropped-Img	22.5	13.5	15.5	17.17
BB-Coordinates + Label	12.0	8.5	2.5	7.67
BB-Coordinates + Cropped-Img	25.5	14.5	13.0	17.67
Point Clouds	12.0	13.5	3.5	9.67
Label + Point Clouds	11.5	10.5	3.0	8.33
Cropped-Img + Point Clouds	15.0	15.0	18.0	16.00
Fusion Graph - Fully-Connected				
BB-Coordinates + Label	12.0	15.5	5.5	11.00
BB-Coordinates + Cropped-Img	30.5	17.5	8.5	18.83
Sparsification Methods				
Random Node Removal	13.5	7.5	4.0	8.33
Naive NR (no soft histogram loss)	21.5	9.5	10.5	13.83
Threshold	31.5	15.0	11.0	19.17
TopK	25.0	18.5	11.0	18.17
SIR (Ours)	38.5	14.0	13.0	21.83

Table 14. Coffee tasks: success rate over 100 rollouts with 2 models trained on different seeds.

Feature Input	Coffee Serve Mug	Coffee Setup Mug	**Average**
Baselines			
Image	9.5	0.0	4.75
Image + FiLM	3.0	0.0	1.50
Own Pretrained Image	4.0	0.0	2.00
Point Clouds	2.5	0.0	1.25
Image + Point Clouds	4.0	0.5	2.25
Split-View Graph - Fully-Connected			
Cropped-Img	12.0	1.0	6.50
BB-Coordinates + Label	9.0	0.5	4.75
BB-Coordinates + Cropped-Img	12.5	1.5	7.00
Point Clouds	4.5	0.0	2.25
Label + Point Clouds	3.0	0.5	1.75
Cropped-Img + Point Clouds	11.0	1.5	6.25
Fusion Graph - Fully-Connected			
BB-Coordinates + Label	4.5	0.5	2.50
BB-Coordinates + Cropped-Img	8.5	1.0	4.75
Sparsification Methods			
Random Node Removal	3.5	0.0	1.75
Naive NR (no soft histogram loss)	7.0	0.0	3.50
Threshold	6.0	0.0	3.0
TopK	8.5	0.5	4.50
SIR (Ours)	9.0	0.5	4.75

Table 15. Success rate on RoboCasa across all 24 tasks over 100 rollouts with 2 models trained on different seeds.

Feature Input	Pick/Place (8)	Doors (4)	Drawers (2)	Knobs (2)	Levers (3)	Buttons (3)	Insert (2)	**Avg (24)**
Baselines								
Image	**1.19**	25.13	**49.75**	7.25	23.67	17.00	4.75	14.81
Image + FiLM	0.34	27.75	45.25	11.00	32.83	16.83	1.50	15.85
Own Pretrained Image	0.13	17.00	39.25	7.25	18.83	10.17	2.00	10.11
Point Clouds	0.00	0.00	1.00	1.25	19.83	10.83	1.25	4.13
Image + Point Clouds	0.56	25.25	44.75	5.50	20.67	15.17	2.25	13.25
Split-View Graph - Fully-Connected								
Cropped-Img	0.25	20.88	37.25	15.00	48.33	17.17	6.50	16.65
BB-Coordinates + Label	0.13	19.13	31.50	13.50	21.17	7.67	4.75	10.98
BB-Coordinates + Cropped-Img	0.25	23.13	35.25	14.25	40.33	17.67	**7.00**	15.90
Point Clouds	0.06	11.38	29.25	18.50	30.33	9.67	2.25	11.08
Label + Point Clouds	0.40	15.88	32.25	15.00	36.83	8.33	1.75	12.50
Cropped-Img + Point Clouds	0.25	19.38	40.75	8.50	40.83	16.00	6.25	15.04
Fusion Graph - Fully-Connected								
BB-Coordinates + Label	0.06	14.12	31.50	13.00	27.17	11.00	2.50	11.06
BB-Coordinates + Cropped-Img	0.06	28.62	39.25	14.00	40.00	18.83	4.75	16.98
Sparsification Methods								
Random Node Removal	0.00	5.62	9.50	5.25	17.00	8.33	1.75	5.48
Naive NR (no soft histogram loss)	0.00	8.50	23.50	6.75	29.17	13.83	3.50	9.60
Threshold	0.0	28.38	47.25	11.0	39.5	19.17	3.0	17.17
TopK	0.12	**31.37**	45.00	**19.50**	41.17	18.17	4.50	18.44
SIR (Ours)	0.12	30.25	46.25	16.50	**48.50**	**21.83**	4.75	**19.50**

Table 16. Five most relevant nodes per task and model – PnP.

Task	TopK	SIR	Threshold
PnPCabToCounter	PandaMobile (0.999) HingeCabinet (1.000) / Hood (0.985) Wall (0.947) / PandaGripper (0.893) Counter (0.918) / Toaster (0.763) SingleCabinet (0.844) / distr_counter (0.689)	SingleCabinet (1.000) / HingeCabinet (1.000) PandaMobile (1.000) HingeCabinet (1.000) / Counter (0.986) Counter (1.000) / PandaGripper (0.977) PandaGripper (0.985) / SingleCabinet (0.936)	PandaMobile (1.000) PandaGripper (0.964) / FramedWindow (0.950) SingleCabinet (0.909) / PandaGripper (0.912) Toaster (0.891) / Sink (0.690) FramedWindow (0.503) / Toaster (0.672)
PnPCounterToCab	PandaMobile (0.997) HingeCabinet (1.000) / Toaster (0.952) Wall (0.992) / obj (0.829) SingleCabinet (0.907) / PandaGripper (0.822) Counter (0.836) / distr_counter (0.747)	HingeCabinet (1.000) / Counter (1.000) PandaMobile (1.000) Counter (1.000) / PandaGripper (0.996) PandaGripper (0.995) / HingeCabinet (0.917) SingleCabinet (0.939)	PandaMobile (0.999) PandaGripper (0.927) SingleCabinet (0.924) / Toaster (0.672) Toaster (0.882) / FramedWindow (0.575) FramedWindow (0.870) / SingleCabinet (0.569)
PnPCounterToMicrowave	Microwave (0.999) / PandaMobile (0.988) PandaMobile (0.999) / obj_container (0.892) Oven (0.849) / obj (0.795) container (0.791) / distr_counter (0.768) Fridge (0.714) / PandaGripper (0.734)	PandaMobile (1.000) Microwave (0.999) PandaGripper (0.951) / Counter (0.982) Counter (0.951) / PandaGripper (0.980) obj (0.660)	Stovetop (1.000) / PandaMobile (0.998) PandaMobile (0.983) / PandaGripper (0.749) PandaGripper (0.829) / obj_container (0.679) obj_container (0.800) / FramedWindow (0.568) container (0.649) / Oven (0.526)
PnPCounterToSink	PandaMobile (0.999) Sink (0.974) / obj (0.815) CoffeeMachine (0.937) / distr_counter (0.803) Wall (0.915) / PandaGripper (0.709) Counter (0.874) / distr_sink (0.704)	PandaMobile (0.999) Counter (0.998) Sink (0.995) PandaGripper (0.953) obj (0.843)	PandaMobile (1.000) PandaGripper (0.832) / CoffeeMachine (0.912) FramedWindow (0.726) / PandaGripper (0.823) distr_counter (0.601) / WallAccessory (0.564) WallAccessory (0.509) / FramedWindow (0.461)
PnPCounterToStove	Microwave (1.000) / PandaMobile (0.982) PandaMobile (0.997) / obj_container (0.961) Wall (0.954) / obj (0.836) Drawer (0.943) / PandaGripper (0.819) Stovetop (0.939) / WallAccessory (0.444)	PandaMobile (1.000) PandaGripper (0.989) / Counter (0.999) Counter (0.984) / PandaGripper (0.999) Stove (0.989) obj (0.919)	PandaMobile (0.991) Fridge (0.957) / obj_container (0.892) PandaGripper (0.883) obj_container (0.823) / CoffeeMachine (0.752) WallAccessory (0.656)
PnPMicrowaveToCounter	PandaMobile (0.988) Microwave (1.000) / container (0.839) Oven (0.865) / obj (0.776) Fridge (0.805) / PandaGripper (0.769) Wall (0.799) / distr_counter (0.606)	PandaMobile (0.993) / Microwave (0.998) Microwave (0.976) / Counter (0.964) PandaGripper (0.945) / PandaMobile (0.961) Counter (0.940) / PandaGripper (0.951) obj (0.822)	Sink (1.000) / PandaMobile (0.988) PandaMobile (0.997) / PandaGripper (0.733) PandaGripper (0.887) / Oven (0.713) container (0.652) Oven (0.720) / obj (0.394)
PnPSinkToCounter	PandaMobile (0.994) Sink (0.997) / container (0.916) Wall (0.955) / PandaGripper (0.877) FramedWindow (0.842) / distr_counter (0.736) Counter (0.792) / obj (0.717)	obj (1.000) / PandaMobile (1.000) Sink (0.999) / Counter (0.981) PandaMobile (0.999) / Sink (0.970) Counter (0.991) / obj (0.947) PandaGripper (0.952)	PandaMobile (0.999) / Microwave (1.000) FramedWindow (0.949) / PandaMobile (1.000) PandaGripper (0.812) WallAccessory (0.685) / container (0.681) container (0.640) / FramedWindow (0.666)
PnPStoveToCounter	Microwave (1.000) / PandaMobile (0.991) PandaMobile (1.000) / container (0.984) Stovetop (0.888) / PandaGripper (0.880) Wall (0.881) / Dishwasher (0.621) Counter (0.638) / HingeCabinet (0.507)	PandaMobile (1.000) Stove (1.000) / PandaGripper (0.999) Counter (1.000) / Stove (0.998) PandaGripper (0.999) / Counter (0.990) obj (0.958)	PandaMobile (0.994) PandaGripper (0.845) / container (0.849) container (0.843) / PandaGripper (0.805) Stove (0.761) / WallAccessory (0.543) CoffeeMachine (0.667) / Stove (0.474)

Table 17. Five most relevant nodes per task and model – Door.

Task	TopK	SIR	Threshold
OpenSingleDoor	PandaMobile (0.979) / Microwave (0.989) Wall (0.855) / FramedWindow (0.972) Microwave (0.802) / PandaMobile (0.841) PandaGripper (0.759) / Oven (0.835) Counter (0.672) / PandaGripper (0.831)	PandaMobile (1.000) PandaGripper (0.985) CoffeeMachine (0.302) / HingeCabinet (0.324) Toaster (0.249) / Microwave (0.303) OmronMobileBase (0.215) / Box (0.267)	PandaMobile (0.989) FramedWindow (0.974) PandaGripper (0.844) distr_counter_3 (0.732) / Microwave (0.750) Oven (0.631) / distr_counter_2 (0.638)
OpenDoubleDoor	PandaMobile (0.996) / PandaGripper (0.904) Wall (0.990) / HingeCabinet (0.901) HingeCabinet (0.944) / PandaMobile (0.891) PandaGripper (0.861) / Sink (0.851) Counter (0.793) / Drawer (0.751)	PandaMobile (1.000) PandaGripper (0.989) HingeCabinet (0.403) / Drawer (0.437) SingleCabinet (0.299) / Oven (0.395) FramedWindow (0.289) / HingeCabinet (0.377)	PandaMobile (0.997) HingeCabinet (0.918) / PandaGripper (0.941) PandaGripper (0.881) / HingeCabinet (0.769) Toaster (0.660) WallAccessory (0.469) / Sink (0.705)
CloseSingleDoor	PandaMobile (0.989) / Microwave (0.885) SingleCabinet (0.780) / PandaMobile (0.872) Microwave (0.763) / Oven (0.828) PandaGripper (0.697) / SingleCabinet (0.775) Wall (0.695) / FramedWindow (0.750)	PandaMobile (1.000) PandaGripper (0.997) Dishwasher (0.209) / HingeCabinet (0.431) Fridge (0.198) / Box (0.351) CoffeeMachine (0.182) / Fridge (0.314)	PandaMobile (0.998) / Stovetop (1.000) FramedWindow (0.959) / PandaMobile (0.984) PandaGripper (0.879) / FramedWindow (0.907) SingleCabinet (0.805) / PandaGripper (0.888) Oven (0.648) / SingleCabinet (0.747)
CloseDoubleDoor	PandaMobile (0.994) / HingeCabinet (0.964) Wall (0.950) / PandaMobile (0.894) HingeCabinet (0.950) / PandaGripper (0.798) Counter (0.798) / SingleCabinet (0.657) PandaGripper (0.571) / FramedWindow (0.637)	PandaMobile (1.000) PandaGripper (0.991) Hood (0.381) / Oven (0.718) HingeCabinet (0.324) / Microwave (0.388) Microwave (0.291) / Fridge (0.355)	PandaMobile (0.997) HingeCabinet (0.913) / PandaGripper (0.852) PandaGripper (0.835) / Toaster (0.751) FramedWindow (0.747) / HingeCabinet (0.744) Toaster (0.571) / Sink (0.736)

Table 18. Five most relevant nodes per task and model – Drawer.

Task	TopK	SIR	Threshold
OpenDrawer	Oven (1.000) / PandaMobile (0.939) Microwave (1.000) / distr_counter_2 (0.900) FramedWindow (1.000) / distr_counter_1 (0.873) PandaMobile (0.970) / Toaster (0.782) Wall (0.879) / Stovetop (0.753)	PandaGripper (1.000) / PandaMobile (1.000) Counter (1.000) PandaMobile (0.998) / PandaGripper (0.999) Drawer (0.726) Stool (0.574) / Dishwasher (0.722)	Oven (1.000) / PandaMobile (0.989) PandaMobile (0.991) / Sink (0.885) FramedWindow (0.781) Sink (0.938) / OmronMobileBase (0.491) Toaster (0.681) / PandaGripper (0.484)
CloseDrawer	Oven (1.000) / distr_counter_2 (0.887) Microwave (1.000) / PandaMobile (0.842) FramedWindow (1.000) / PandaGripper (0.781) PandaMobile (0.997) / Toaster (0.718) Wall (0.852) / distr_counter_1 (0.712)	PandaMobile (1.000) Counter (1.000) Drawer (0.996) / PandaGripper (0.999) PandaGripper (0.994) / Drawer (0.987) Fridge (0.014) / Floor (0.013)	PandaMobile (0.980) FramedWindow (0.811) / Sink (0.789) Sink (0.797) / PandaGripper (0.590) Toaster (0.737) / FramedWindow (0.500) Oven (0.700) / OmronMobileBase (0.467)

Table 19. Five most relevant nodes per task and model – Knobs.

Task	TopK	SIR	Threshold
TurnOnStove	Microwave (1.000) / Sink (1.000) Stove (1.000) PandaMobile (1.000) / PandaGripper (0.905) Stovetop (0.965) / PandaMobile (0.893) Wall (0.941) / Dishwasher (0.813)	Stove (1.000) PandaMobile (1.000) PandaGripper (0.999) Hood (0.168) / OmronMobileBase (0.216) Floor (0.123) / SingleCabinet (0.167)	Fridge (1.000) / PandaMobile (0.994) PandaMobile (0.999) / PandaGripper (0.846) PandaGripper (0.912) / Toaster (0.701) Microwave (0.641) / WallAccessory (0.598) Stove (0.572)
TurnOffStove	Microwave (1.000) / Stove (0.955) PandaMobile (1.000) / Dishwasher (0.931) Wall (0.999) / PandaGripper (0.927) Stove (0.999) / PandaMobile (0.879) Stovetop (0.952) / Fridge (0.785)	Stove (1.000) / PandaMobile (1.000) PandaMobile (1.000) / PandaGripper (1.000) PandaGripper (0.993) / Stove (1.000) Floor (0.159) / Stovetop (0.086) Accessory (0.139) / WallAccessory (0.066)	PandaMobile (0.991) PandaGripper (0.928) WallAccessory (0.583) / Sink (0.865) Microwave (0.529) / Drawer (0.833) Stove (0.493) / CoffeeMachine (0.587)

Table 20. Five most relevant nodes per task and model – Lever.

Task	TopK	SIR	Threshold
TurnOnSinkFaucet	PandaMobile (0.965) FramedWindow (0.956) / Fridge (0.851) Sink (0.888) Wall (0.864) / distr_counter_1 (0.628) Counter (0.673) / WallAccessory (0.586)	Sink (1.000) PandaMobile (0.999) PandaGripper (0.995) Stool (0.016) Counter (0.002) / Wall (0.002)	PandaMobile (0.998) FramedWindow (0.839) / PandaGripper (0.607) Toaster (0.780) / FramedWindow (0.389) PandaGripper (0.735) / distr_counter_1 (0.331) WallAccessory (0.546) / distr_counter_0 (0.321)
TurnOffSinkFaucet	PandaMobile (0.995) Sink (0.956) / distr_counter_0 (0.737) FramedWindow (0.845) / distr_counter_1 (0.731) Counter (0.809) / PandaGripper (0.686) Stool (0.797) / Sink (0.658)	PandaMobile (1.000) Sink (1.000) PandaGripper (0.999) HingeCabinet (0.002) / Stool (0.005) Wall (0.000) / Floor (0.002)	PandaMobile (1.000) / Toaster (1.000) FramedWindow (0.971) / PandaMobile (1.000) Fridge (0.919) / PandaGripper (0.704) PandaGripper (0.610) / FramedWindow (0.560) WallAccessory (0.508) / Fridge (0.527)
TurnSinkSpout	PandaMobile (0.990) Sink (0.960) / distr_counter_0 (0.838) Wall (0.924) / PandaGripper (0.809) FramedWindow (0.879) / distr_sink (0.666) Counter (0.877) / distr_counter_2 (0.635)	Sink (1.000) / PandaMobile (1.000) PandaMobile (1.000) / Sink (0.995) PandaGripper (0.976) Counter (0.001) / Floor (0.056) Wall (0.001) / Counter (0.000)	PandaMobile (1.000) FramedWindow (0.911) / Fridge (1.000) PandaGripper (0.884) Sink (0.559) / FramedWindow (0.841) distr_counter_0 (0.527) / WallAccessory (0.651)

Table 21. Five most relevant nodes per task and model – Button.

Task	TopK	SIR	Threshold
CoffeePressButton	PandaMobile (0.988) Counter (0.974) / PandaGripper (0.941) Wall (0.897) / obj (0.901) PandaGripper (0.857) / CoffeeMachine (0.810) Fridge (0.741) / Toaster (0.724)	PandaMobile (1.000) PandaGripper (0.995) CoffeeMachine (0.992) obj (0.008) WallAccessory (0.000) / Wall (0.007)	PandaMobile (0.992) PandaGripper (0.930) / FramedWindow (0.961) Sink (0.820) / PandaGripper (0.838) obj (0.522) / CoffeeMachine (0.667) WallAccessory (0.495) / Toaster (0.644)
TurnOnMicrowave	PandaMobile (0.995) / Toaster (1.000) Microwave (0.992) / PandaGripper (0.978) Oven (0.983) / PandaMobile (0.754) PandaGripper (0.678) / Oven (0.728) Wall (0.607) / OmronMobileBase (0.717)	PandaGripper (0.997) / Microwave (1.000) PandaMobile (0.997) Microwave (0.994) / PandaGripper (0.997) Oven (0.014) / HingeCabinet (0.029) Counter (0.005)	PandaMobile (0.985) Toaster (0.940) / PandaGripper (0.619) PandaGripper (0.776) / Microwave (0.222) Oven (0.425) / OmronMobileBase (0.201) Microwave (0.249) / CoffeeMachine (0.163)
TurnOffMicrowave	Microwave (1.000) / Toaster (0.960) PandaMobile (0.999) / Microwave (0.872) Oven (0.991) / PandaGripper (0.841) PandaGripper (0.794) / Oven (0.815) Wall (0.622) / PandaMobile (0.713)	PandaGripper (1.000) / PandaMobile (1.000) Microwave (1.000) / PandaGripper (0.997) PandaMobile (1.000) / Microwave (0.996) Counter (0.000) / HingeCabinet (0.064) WallAccessory (0.000)	PandaMobile (1.000) PandaGripper (0.870) Toaster (0.768) / Microwave (0.256) Oven (0.493) / OmronMobileBase (0.176) Microwave (0.380) / Floor (0.132)

Table 22. Five most relevant nodes per task and model – Insertion.

Task	TopK	SIR	Threshold
CoffeeServeMug	PandaMobile (0.997) / CoffeeMachine (0.967) Wall (0.981) / PandaMobile (0.951) Counter (0.963) / obj (0.895) Sink (0.760) / PandaGripper (0.848) PandaGripper (0.744) / Toaster (0.760)	obj (1.000) / CoffeeMachine (1.000) CoffeeMachine (1.000) / Counter (1.000) Counter (0.999) / PandaMobile (1.000) PandaMobile (0.995) / obj (0.999) PandaGripper (0.981)	PandaMobile (0.994) Sink (0.924) / FramedWindow (0.998) PandaGripper (0.921) / CoffeeMachine (0.958) CoffeeMachine (0.846) / PandaGripper (0.935) WallAccessory (0.695) / Sink (0.872)
CoffeeSetupMug	FramedWindow (0.994) / PandaMobile (0.991) PandaMobile (0.992) / obj (0.989) Wall (0.981) / CoffeeMachine (0.964) Counter (0.969) / PandaGripper (0.917) PandaGripper (0.883) / Toaster (0.722)	Counter (1.000) / PandaMobile (1.000) CoffeeMachine (0.999) / Counter (1.000) PandaGripper (0.980) / CoffeeMachine (1.000) PandaMobile (0.974) / obj (0.995) obj (0.964) / PandaGripper (0.978)	FramedWindow (1.000) / PandaMobile (1.000) PandaMobile (0.993) / PandaGripper (0.991) PandaGripper (0.978) / CoffeeMachine (0.991) CoffeeMachine (0.797) / Toaster (0.973) WallAccessory (0.692) / Sink (0.655)

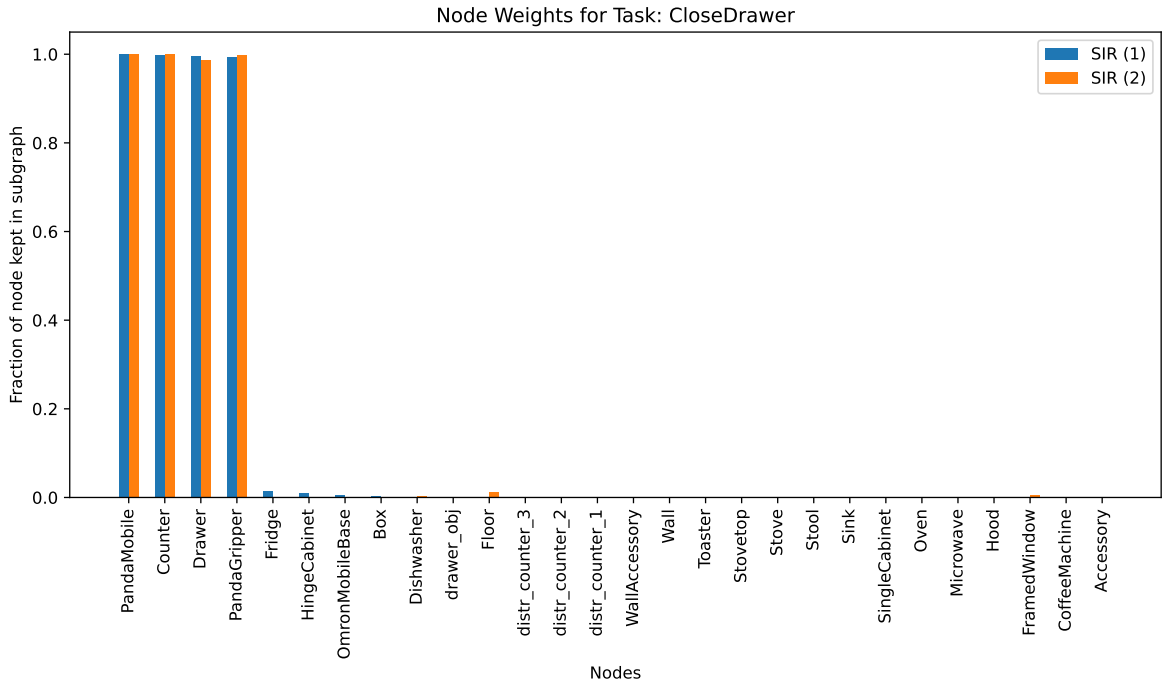

Figure 6. Percentage of nodes to be kept in the sub-graph per seeded model *CloseDrawer*. In contrast to Figure 5c, SIR is able to include the drawer in the graph. This stands in no conflict with our previous claim that the drawer is not important, as the instruction-grounded method is explicitly *encouraged* by the additional loss to include the drawer. This further stands in no conflict with our results, where even in the instruction-grounded approach, the relevant nodes are not included, as the instruction-grounded approach is not *forced* to use the important nodes.

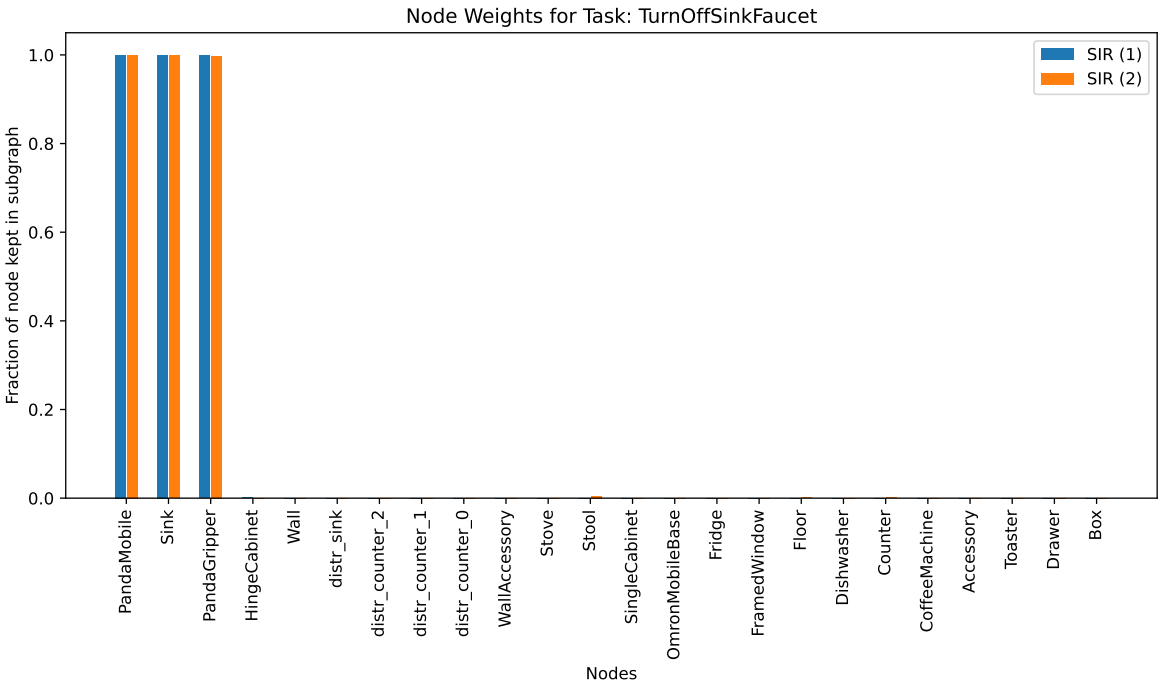

Figure 7. Percentage of nodes to be kept in the sub-graph per seeded model *TurnOffSinkFaucet*.

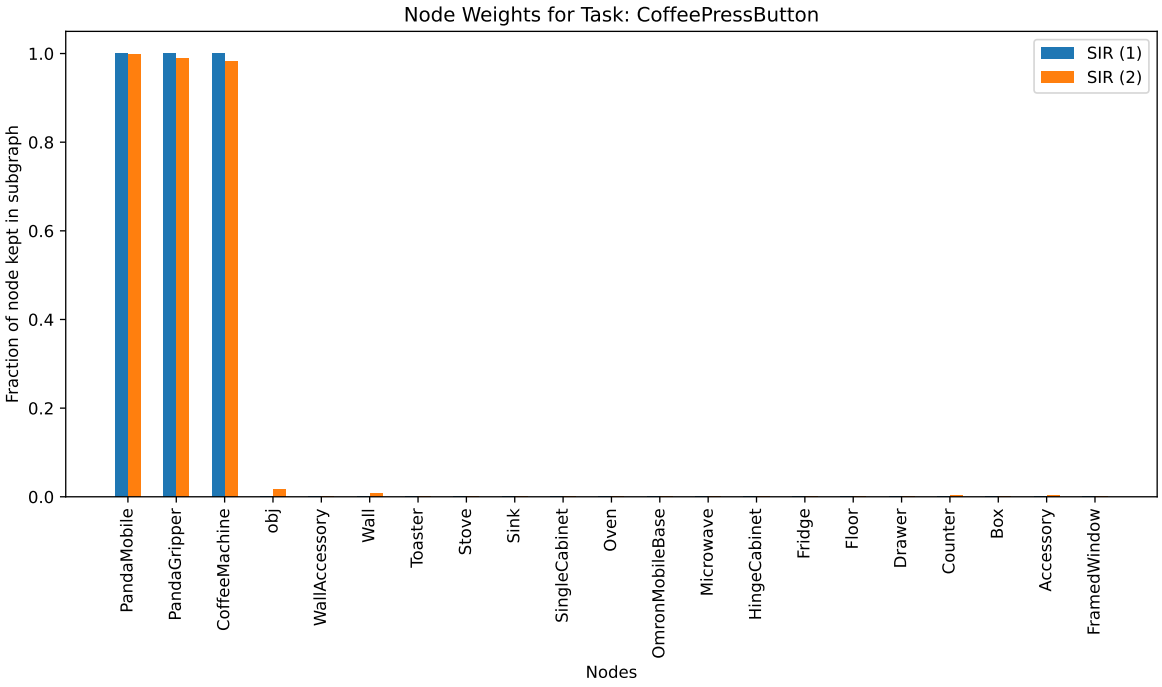

Figure 8. Percentage of nodes to be kept in the sub-graph per seeded model *CoffeePressButton*.

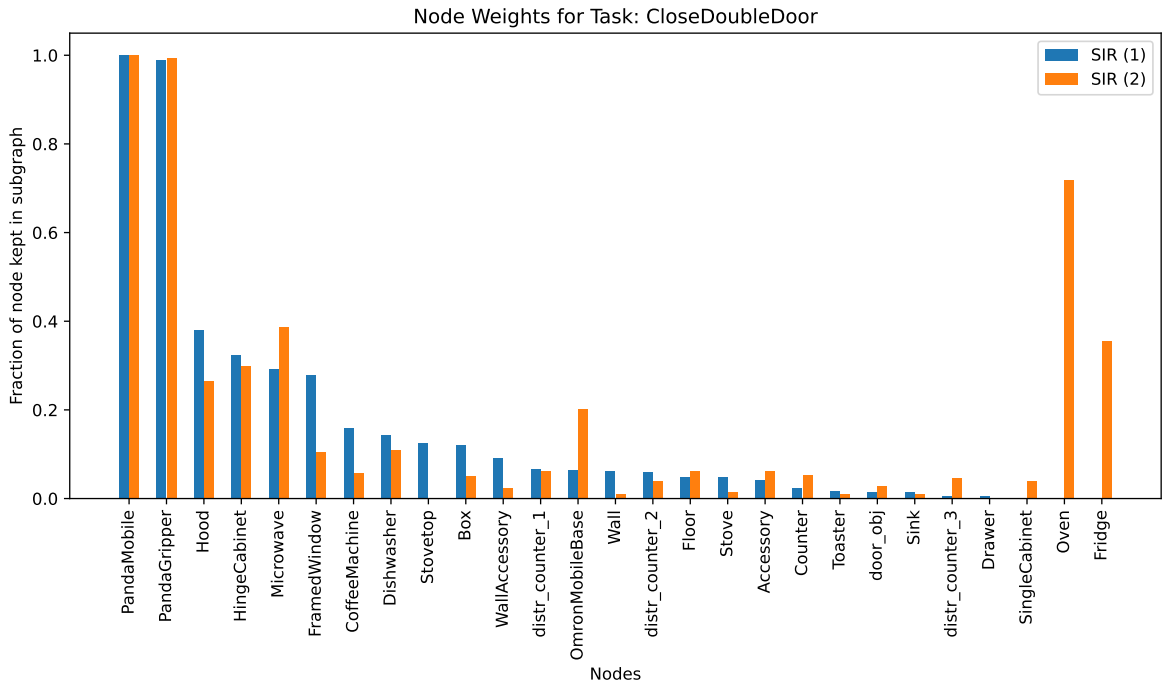

Figure 9. Percentage of nodes to be kept in the sub-graph per seeded model *CloseDoubleDoor*. Similar to Figure 5d, here also the object of which the door has to be close (HingeCabinet) is not always selected to be in the graph.

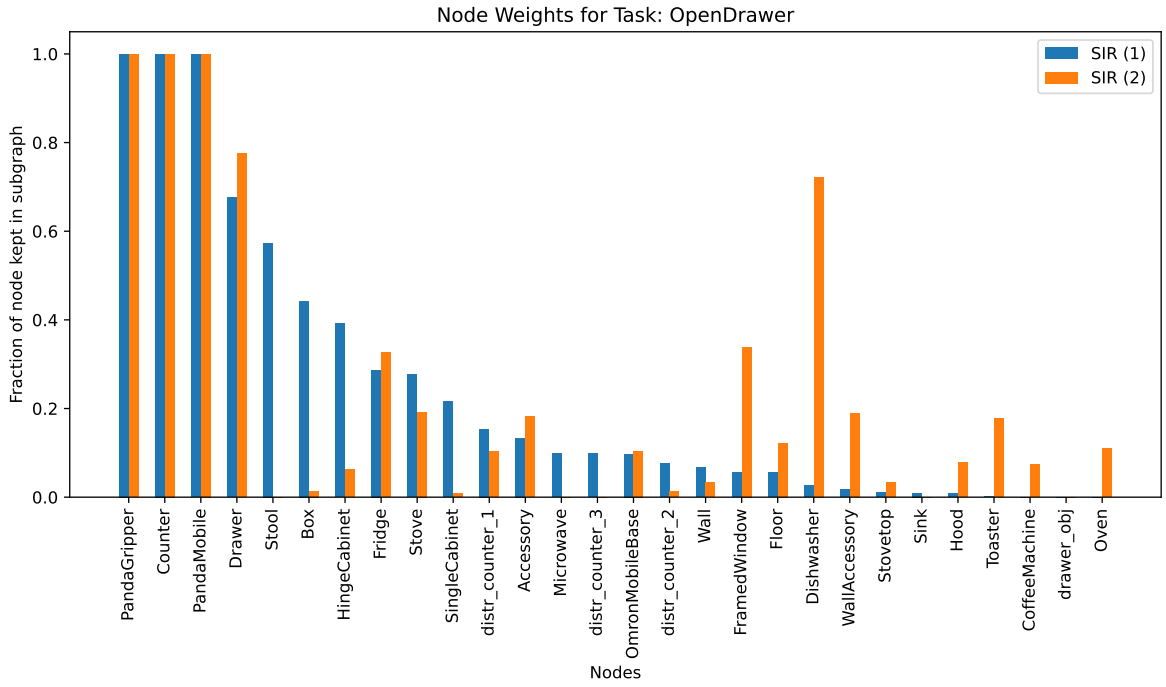

Figure 10. Percentage of nodes to be kept in the sub-graph per seeded model *OpenDrawer*. Although the important nodes are included, other nodes are not consistently removed.

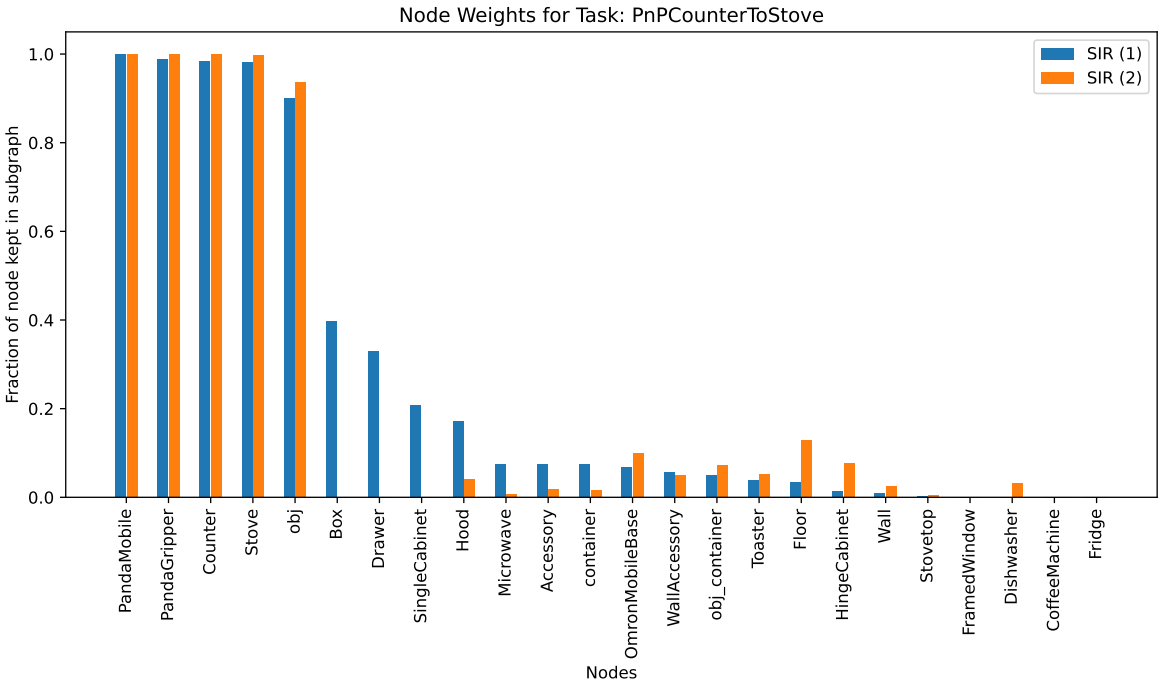

Figure 11. Percentage of nodes to be kept in the sub-graph per seeded model in *PnPCounterToStove*.

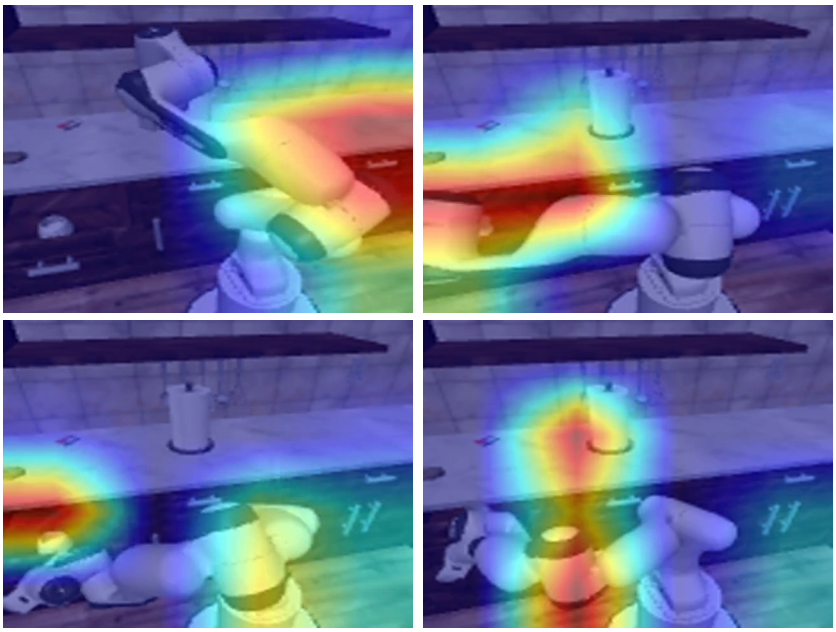

Figure 12. Grad-CAM explanations over time for a rollout of the *CloseDrawer* task (frames 20, 70, 120, 170). We find that the explanations produced by Grad-CAM are neither consistent, as the activated regions are jumping around a lot in the image, nor expressive, as the highlighted regions are not bound to objects.