Human Hands as Probes for Interactive Object Understanding – Supplementary Material

S6. State Sensitive Feature Learning

S6.1. Model Details

S6.1.1 Baseline Models

We compare to six baseline models: ImageNet pre-trained model without any further training, three self-supervised models (SimCLR [6], TCN [53], and SimCLR+TCN), and two supervised models (action classification on EPIC-KITCHENS, and MIT States supervision). All models are initialized from ImageNet pre-training.

All models use the ResNet-18 backbone. We average pool the output after the second last layer to obtain a 512 dimensional representation. The self-supervised models use 3 projection layers with sizes 512, 512, 128. The 128-dimensional output from the last layer is used for computing the similarity. The semantically supervised models (*i.e*. those trained on MIT States dataset, or for action classification on the EPIC-KITCHENS dataset) only use a single linear classifier layer directly on top of ImageNet features. These additional layers were thrown out and just the ResNet-18 backbone is used for state classification experiment on EPIC-STATES. We only train a linear classifier on top of the learned ResNet-18 backbone for downstream state-classification on EPIC-KITCHENS dataset.

We use a batch size of 256 for pre-trainings. We optimize using Adam optimizer with a learning rate of 10^{-4}. All models (with the exception of the two supervised baselines) are trained for 200 epochs and the last checkpoint is selected as the final model, which eliminates any dependencies on the pre-training validation set. All models are trained on a single NVidia GPU (RTX A40 or equivalent). We next list method specific hyper-parameters.

1. **ImageNet Pre-trained Model.** This model has no additional hyper-parameters.

2. **SimCLR.** We use the standard SimCLR augmentations in the following order: random resized crop with scale (0.5, 1.0), random horizontal flip, random color jitter with parameters (0.8, 0.8, 0.8, 0.2) and 80% probability, random grayscale with 20% probability, Gaussian blur with 12 size kernel and sigma set to (0.1, 2.0), and finally ImageNet normalization.

3. **TCN.** Let w denote the length of the current track (number of frames). TCN's window size is set to $\lfloor w/4 \rfloor$ and the negative sample is guaranteed to be at least $\lceil w/2 \rceil$ (negative window) away from the positive examples. Sampling the positive and negatives on opposite ends of the track ensures a large distance between them. TCN is optimized with a triplet margin loss. Let us reuse o_i, o_i' as the positive pair and define o_i'' as the negative sample. Given an arbitrary margin α (in practice $\alpha = 2$), the triplet margin loss is as follows. We chose the hyper-parameters as suggested by [53].

$$||f_o(o_i) - f_o(o_i')||_2^2 + \alpha < ||f_o(o_i) - f_o(o_i'')||_2^2. \tag{1}$$

4. **SimCLR+TCN.** We combine SimCLR and TCN, by a) sampling negative from both within and across tracks, and b) using a NT-Xent loss from SimCLR [6]. We also adopt the image augmentations used in SimCLR.

5. **Action Classification.** We train ResNet-18 (initialized from ImageNet) on 32 action labels along with their temporal extent, available as part of the EPIC-KITCHENS dataset. These include: take, put, wash, open, close, insert, cut, pour, mix, turn-on, move, remove, turn-off, dry, throw, shake, squeeze, adjust, peel, scoop, empty, flip, fill, turn, check, spray, apply, pat, fold, scrape, sprinkle, break. The model samples two frames (in order) and uses them jointly to classify the action. This allows the model to disambiguate between open and close actions. The model is trained for 30 epochs and we select the model which performed the best on the action classification validation set. Validation performance peaked within 30 epochs.

6. **MIT States.** We train ResNet-18 (initialized from ImageNet) on the MIT-States attributes dataset [24]. The dataset consists of 115 classes and approximately 53K images. Examples of attributes include mossy, deflated, dirty, *etc*. This model is trained for 20 epochs and we select the model which performed the best on the MIT-States validation set. Validation performance peaked within 20 epochs.

S6.1.2 TSC and TSC+OHC Models

For TSC, object crops are selected by randomly sampling o_i' such that o_i' is no more than $\lfloor w/4 \rfloor$ away from o_i in the track, where w is the length of the track.

For TSC+OHC, we use two separate models, one each for the object and the hand. The object model itself has 2 heads, one is used for the object-object similarity for L_{temporal}, and another for object-hand similarity for L_{hand}. The hand model only has one head. The hand model has additional layers to produce and combine the positional encodings that represent motion. The positional encoding is generated by alternating sines and cosines over 12 frequencies for each element of h_i^m. It is concatenated with the output of the ResNet-18 backbone. These combined features are projected to 512 dimensions with another linear layer and finally fed through the hand model's loss head. Note that the object and hand crop fed through this model are not augmented with random horizontal flip to preserve handedness.

For the TSC+OHC model, tracks are independently sampled for computing the L_{temporal} and L_{hand} losses. Tracks with less than 4 frames of hands were filtered out to remove noise, which led to a pre-training dataset size of 53,661 tracks. The evaluation scheme remains the same as TSC since we only use the features learned by the object model. We throw out the hand model.

Loss Functions. Recall that h_i^a and h_i^m jointly represent the hand: h_i^a describes the appearance and h_i^m describes the motion. To detail h_i^m, consider the object bounding box for o_i defined as $(o_{i,x1}, o_{i,y1}, o_{i,x2}, o_{i,y2})$ where $(o_{i,x1}, o_{i,y1}), (o_{i,x2}, o_{i,y2})$ are the coordinates of the top left and bottom right corner of the bounding box, respectively. We define the hand crop bounding box similarly: $(h_{k,x1}, h_{k,y1}, h_{k,x2}, h_{k,y2})$ where k is uniformly sampled at random such that $|k-i| \leq 3$. $(*_H, *_W)$ represent the height and width of the bounding box and $(*_{xc}, *_{yc})$ refer to the center coordinates of the bounding box. For an arbitrary p, we calculate $h_{i,p}^m$ and h_i^m as follows:

$$h_{i,p}^m = \left[\frac{o_{i,xc} - h_{p,xc}}{o_{i,W}}, \frac{o_{i,yc} - h_{p,yc}}{o_{i,H}}, \frac{h_{p,W}}{o_{i,W}}, \frac{h_{p,H}}{o_{i,H}}\right] \tag{2}$$

$$h_i^m = \left[h_{i,k-1}^m, \quad h_{i,k}^m, \quad h_{i,k+1}^m\right] \tag{3}$$

Assuming that all crops (o_i, o_i', h_i^a) are transformed using the standard SimCLR augmentations, we describe our application of normalized temperature scaled cross-entropy loss (NT-Xent) [6] below:

$$s_{i,j}^{oo'} = {}^1\!/\!_\tau \cdot \text{sim}(f_o(o_i), f_o(o_j')) \tag{4}$$

$$s_{i,j}^{oh} = {}^1\!/\!_\tau \cdot \text{sim}(f_h(o_i), g_h(h_j)) \tag{5}$$

$$s_{i,j}^{hh} = {}^1\!/\!_\tau \cdot \text{sim}(g_h(h_i), g_h(h_j)) \tag{6}$$

where sim refers to the cosine similarity, and τ is the temperature parameter (in practice $\tau = 0.1$). Then L_{temporal} and L_{hand} losses are computed using s^{oh}, s^{hh}, s^{oo} as follows:

$$L_{\text{temporal}} = -\sum_i \left(\log \frac{\exp(s^{oo'}_{i,i})}{\sum_{j:j\neq i} \exp(s^{oo}_{i,j}) + \sum_k \exp(s^{oo'}_{i,k})} \right) - \sum_i \left(\log \frac{\exp(s^{oo'}_{i,i})}{\sum_{j:j\neq i} \exp(s^{o'o'}_{i,j}) + \sum_k \exp(s^{oo'}_{k,i})} \right) \quad (7)$$

$$L_{\text{hand}} = -\sum_i \left(\log \frac{\exp(s^{oh}_{i,i})}{\sum_{j:j\neq i} \exp(s^{oo}_{i,j}) + \sum_k \exp(s^{oh}_{i,k})} \right) - \sum_i \left(\log \frac{\exp(s^{oh}_{i,i})}{\sum_{j:j\neq i} \exp(s^{hh}_{i,j}) + \sum_k \exp(s^{oh}_{k,i})} \right). \quad (8)$$

S6.2. EPIC-STATES Dataset and State Classification Task

S6.2.1 Data Annotation

EPIC-STATES is collected on top of the *ground-truth* object-of-interaction tracks and corresponding object category labels from Damen *et al*. [10]. We filter out 13 object categories of interest: drawer, knife, spoon, cupboard, fridge, onion, fork, egg, potato, bottle, microwave/oven, carrot, and jar. We chose a maximum of 5 frames from each track. These object crops are then annotated for states individually for each object category.

We used a commercial service to obtain annotations for our dataset. Each image was annotated once and then reviewed by a high-quality annotator (determined using the accuracy on the task). We also included an AMBIGUOUS class and reject images with the AMBIGUOUS label, resulting in 14,346 annotated images.

Annotation Instruction. We gave the annotators the following instructions:

> Given an image, choose all applicable categories/states from the ones available. If the image is considerably noisy or the object of specified category cannot be identified, annotate the image as ambiguous. When multiple objects are visible, annotate the most dominant object of the specified category. Note that images are captured in-the-wild and small motion blur, therefore, should not be considered as noise.

For each object category, we specify the set of states to consider, and any other object category specific instructions. Below we club the instructions for multiple objects, but note that images from different object categories were annotated *separately*.

1. **Microwave/Oven, Cupboard, Drawer, and Fridge**: The applicable states are OPEN, CLOSE, and AMBIGUOUS. From OPEN/CLOSE only one state would be applicable, *i.e*. a drawer can not be both, OPEN and CLOSE at the same time.

2. **Jar and Bottle**: The applicable states are INHAND, OUTOFHAND, OPEN, CLOSE, and AMBIGUOUS. From INHAND/OUTOFHAND, and OPEN/CLOSE only one state would be applicable, *i.e*. a bottle can not be both, INHAND and OUTOFHAND; or both, OPEN and CLOSE.

3. **Onion and Potato**: The applicable states are INHAND, OUTOFHAND, RAW, COOKED, WHOLE, CUT, PEELED, UNPEELED, and AMBIGUOUS. From INHAND/OUTOFHAND, RAW / COOKED, PEELED / UNPEELED, and WHOLE / CUT only one state would be applicable. For green onions, we asked the annotators to not label the PEELED/UNPEELED attribute.

4. **Carrot**: The applicable states are INHAND, OUTOFHAND, RAW, COOKED, WHOLE, CUT, and AMBIGUOUS. From INHAND/OUTOFHAND, RAW/COOKED, and WHOLE/CUT only one state would be applicable.

5. **Spoon, Fork, and Knife**: The applicable states are INHAND, OUTOFHAND, and AMBIGUOUS. From INHAND/OUTOFHAND, only one state would be applicable.

6. **Egg**: The applicable states are INHAND, OUTOFHAND, RAW, COOKED, and AMBIGUOUS. From INHAND/OUTOFHAND, RAW/COOKED, only one state would be applicable.

S6.2.2 Dataset Statistics

Splits. We split the dataset into train, val and test splits based on participants. See participant assignment to the different splits in Table S3. Participants were assigned between val and test by minimizing the difference in joint object state (fridge-open, onion-cut, *etc*.) distribution between the sets. This ensures a good split of both objects and states.

Novel Object Categories. We list object categories that were used for training and testing for the novel object category experiment in Table S4.

Table S3. EPIC-STATES participants in each data split.

Split	Participants
Train	P01, P03, P06, P08, P13, P17, P21, P25, P26, P29
Validation	P04, P05, P07, P14, P22, P23, P27
Test	P02, P10, P12, P15, P16, P19, P20, P24, P28, P30, P31

Table S4. Novel Category Experiment. For the novel category experiment, we limited the training to objects in the first row, and evaluated on categories in the second row.

Train Objects	fridge, knife, drawer, potato, carrot, jar, egg
Novel Objects	spoon, cupboard, onion, fork, microwave / oven, bottle

Object and State Distributions. Table S6 and Table S5 shows the distribution of states and objects in EPIC-STATES, respectively. We also show the joint distribution of objects and states in Table S7 across the entire dataset. As noted, many different states are applicable to the same object.

Table S5. Objects in EPIC-STATES. For each objects in EPIC-STATES, we list the applicable states and how many instances we have for that object in each split.

Object	Applicable States	Train	Val	Test	Total
fridge	OPEN, CLOSE	779	448	732	1959
spoon	INHAND, OUTOFHAND	751	482	717	1950
knife	INHAND, OUTOFHAND	749	551	729	2029
cupboard	OPEN, CLOSE	683	449	429	1561
drawer	OPEN, CLOSE	681	666	446	1793
onion	INHAND, OUTOFHAND, RAW, COOKED, WHOLE, CUT, PEELED, UNPEELED	487	337	474	1298
fork	INHAND, OUTOFHAND	353	206	259	818
microwave/oven	OPEN, CLOSE	306	118	179	603
bottle	OPEN, CLOSE, INHAND, OUTOFHAND	294	179	257	730
potato	INHAND, OUTOFHAND, RAW, COOKED, WHOLE, CUT, PEELED, UNPEELED	164	192	182	538
carrot	INHAND, OUTOFHAND, RAW, WHOLE, CUT	97	114	196	407
jar	OPEN, CLOSE, INHAND, OUTOFHAND	54	99	89	242
egg	INHAND, OUTOFHAND,RAW, COOKED	32	199	187	418

Table S6. States in EPIC-STATES. For each state in EPIC-STATES, we list the object categories it is applicable to, and how many instances we have for that state in each split.

State	Applicable Objects	Train	Val	Test	Total
INHAND	bottle, carrot, egg, fork, jar, knife, onion, potato, spoon	1861	1236	1699	4796
OPEN	bottle, jar, cupboard, drawer, fridge, microwave / oven	2099	1440	1459	4998
OUTOFHAND	bottle, carrot, egg, fork, jar, knife, onion, potato, spoon	1280	1112	1227	3619
RAW	carrot, egg, onion, potato	623	561	783	1967
CLOSE	bottle, jar, cupboard, drawer, fridge, microwave / oven	686	496	633	1815
CUT	carrot, onion, potato	477	459	499	1435
PEELED	onion, potato	450	423	465	1338
WHOLE	carrot, onion, potato	270	178	351	799
COOKED	egg, onion, potato	152	273	245	670
UNPEELED	onion, potato	197	104	186	487

S6.2.3 State Classification Task

We construct binary state classification tasks by considering all *non-ambiguous* crops from object categories that afford the particular state label, as noted in Table S6. We throw out images that were AMBIGUOUS overall for all categories, or were AMBIGUOUS for the specific state category under consideration.

Table S7. Joint object and state distribution for EPIC-STATES. Note that multiple states are applicable to objects.

	OPEN	CLOSE	INHAND	OUTOFHAND	RAW	COOKED	WHOLE	CUT	PEELED	UNPEELED
bottle	286	358	521	205	✗	✗	✗	✗	✗	✗
carrot	✗	✗	223	184	399	✗	272	132	✗	✗
egg	✗	✗	118	300	210	204	✗	✗	✗	✗
fork	✗	✗	525	293	✗	✗	✗	✗	✗	✗
jar	136	103	154	78	✗	✗	✗	✗	✗	✗
knife	✗	✗	1331	699	✗	✗	✗	✗	✗	✗
onion	✗	✗	411	887	1000	296	290	1005	992	303
potato	✗	✗	236	300	358	170	237	294	346	184
spoon	✗	✗	1277	673	✗	✗	✗	✗	✗	✗
cupboard	1262	310	✗	✗	✗	✗	✗	✗	✗	✗
drawer	1479	315	✗	✗	✗	✗	✗	✗	✗	✗
fridge	1505	456	✗	✗	✗	✗	✗	✗	✗	✗
microwave/oven	330	273	✗	✗	✗	✗	✗	✗	✗	✗

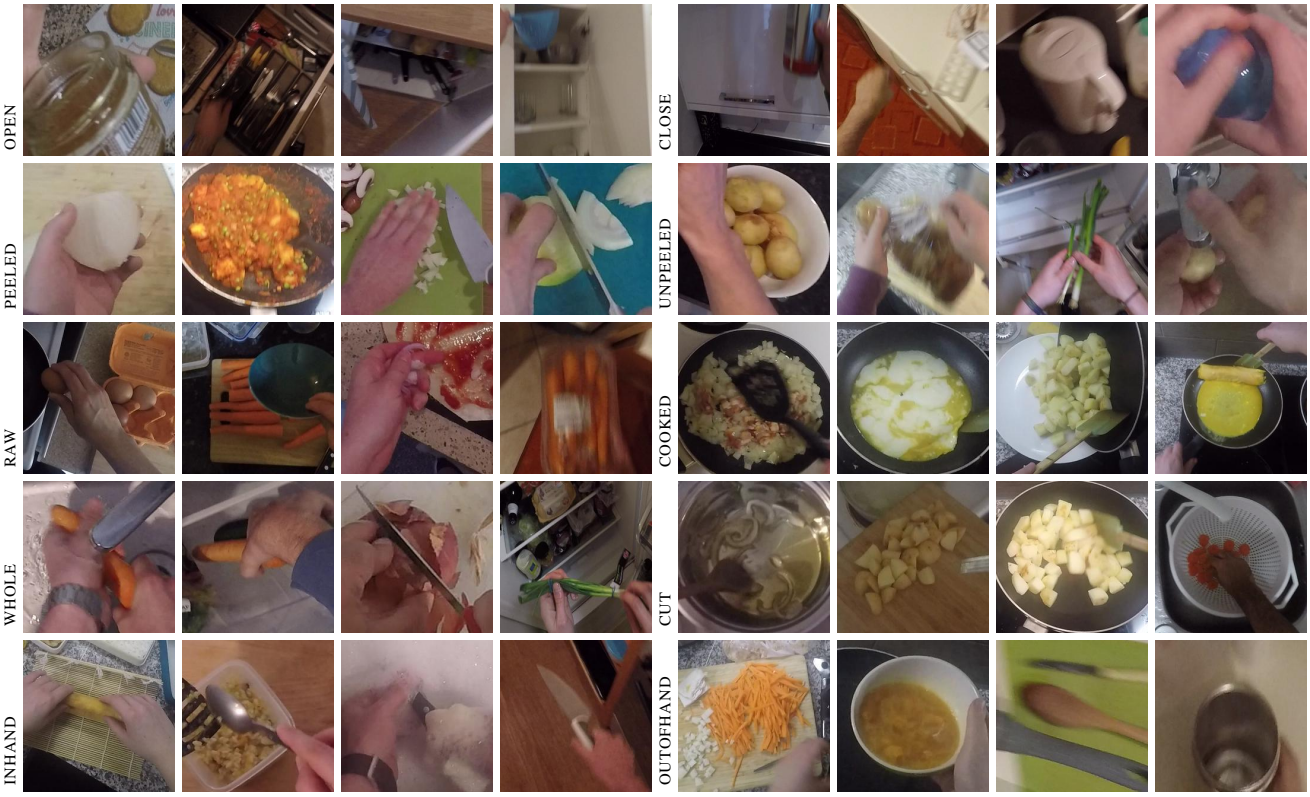

Figure S9. Image samples corresponding to the different states in the EPIC-STATES dataset.

S6.3. Detailed Results and Ablations

S6.3.1 State-wise Performance

State-wise performance of considered methods for state classification is shown in Table S8. In particular, we see that ResNet-18 without any feature learning on our tracks already performs well on the OPEN, RAW, COOKED, PEELED, and CUT categories for most settings. Nonetheless, our methods improve performance across the board. TSC+OHC improves upon TSC on all but one category in the challenging setting of novel categories with limited data.

Table S8. State-wise performance of models on EPIC-STATES test set in the different settings: novel objects with 12.5% training data, novel objects with 100% training data, all objects with 12.5% training data, and all objects with 100% training data.

Novel Objects [12.5%]	**OPEN**	**CLOSE**	**INHAND**	**OUTOFHAND**	**RAW**	**COOKED**	**WHOLE**	**CUT**	**PEELED**	**UNPEELED**	**Mean**
ImageNet Pre-trained	71.9 ±0.0	53.5 ±0.0	76.5 ±0.0	62.8 ±0.0	97.6 ±0.0	87.1 ±0.0	46.6 ±0.0	82.1 ±0.0	78.1 ±0.0	45.9 ±0.0	70.2 ±0.0
TCN [53]	67.4 ±4.6	50.9 ±5.2	63.9 ±2.5	46.7 ±2.4	75.8 ±1.3	32.9 ±1.6	36.6 ±2.6	76.6 ±0.8	72.7 ±1.7	37.3 ±0.9	56.1 ±1.9
SimCLR [6]	77.4 ±2.8	61.7 ±2.6	72.0 ±1.1	53.6 ±1.4	96.0 ±1.0	84.6 ±2.0	55.2 ±5.2	82.3 ±3.3	81.7 ±1.0	54.2 ±3.0	71.9 ±0.2
SimCLR+TCN	73.7 ±2.3	57.2 ±2.8	68.9 ±1.1	51.8 ±2.5	90.5 ±0.7	65.3 ±1.3	42.2 ±2.3	76.0 ±1.9	72.9 ±3.7	38.5 ±3.7	63.7 ±0.3
Semantic supervision											
via EPIC action classification	76.0 ±2.1	57.7 ±1.5	76.4 ±0.4	64.0 ±1.4	95.3 ±2.5	73.6 ±9.0	56.3 ±4.7	81.6 ±2.0	79.4 ±3.3	48.6 ±4.2	70.9 ±2.0
via MIT States dataset [24]	75.0 ±0.9	56.2 ±4.3	73.7 ±0.8	60.4 ±2.6	94.5 ±2.7	77.5 ±9.1	51.4 ±1.3	87.1 ±1.6	78.1 ±2.8	46.8 ±1.5	70.1 ±1.5
Ours [TSC]	79.5 ±1.1	**63.5 ±3.0**	74.8 ±1.9	54.5 ±2.5	96.8 ±1.3	81.7 ±7.7	64.2 ±3.3	87.0 ±1.9	83.9 ±1.6	58.8 ±3.2	74.5 ±1.0
Ours [TSC+OHC]	**81.1 ±0.5**	62.5 ±1.1	**82.9 ±0.9**	**67.5 ±1.6**	**98.3 ±0.6**	**88.1 ±4.6**	**67.1 ±2.6**	**90.4 ±1.4**	**89.1 ±1.8**	**70.4 ±2.1**	**79.8 ±0.6**

Novel Objects [100%]	**OPEN**	**CLOSE**	**INHAND**	**OUTOFHAND**	**RAW**	**COOKED**	**WHOLE**	**CUT**	**PEELED**	**UNPEELED**	**Mean**
ImageNet Pre-trained	74.8 ±0.0	57.6 ±0.0	78.5 ±0.0	70.3 ±0.0	98.6 ±0.0	92.0 ±0.0	62.1 ±0.0	91.4 ±0.0	74.5 ±0.0	45.4 ±0.0	74.5 ±0.0
TCN [53]	70.2 ±3.9	52.6 ±3.3	71.0 ±1.6	53.5 ±1.8	87.0 ±3.9	49.9 ±10.6	50.6 ±3.5	84.6 ±1.9	73.6 ±3.5	45.6 ±4.6	63.9 ±1.1
SimCLR [6]	77.9 ±2.2	62.2 ±1.9	77.4 ±1.3	63.1 ±1.1	97.4 ±1.3	94.0 ±3.4	65.4 ±4.3	90.3 ±0.5	82.0 ±0.5	61.6 ±3.9	77.1 ±1.0
SimCLR+TCN	76.1 ±2.8	61.0 ±2.9	75.2 ±0.7	60.3 ±2.0	92.7 ±0.7	77.6 ±5.9	52.5 ±4.0	82.9 ±2.8	69.6 ±3.0	35.7 ±5.2	68.4 ±1.6
Semantic supervision											
via EPIC action classification	80.0 ±2.4	59.8 ±3.1	82.4 ±1.4	73.6 ±1.0	97.1 ±0.3	83.8 ±8.2	59.6 ±4.5	87.7 ±2.1	84.6 ±2.4	62.0 ±7.9	77.0 ±0.9
via MIT States dataset [24]	77.2 ±1.7	58.3 ±4.4	78.8 ±2.1	67.9 ±0.9	95.6 ±2.4	85.6 ±6.8	48.6 ±2.2	89.2 ±0.9	82.8 ±4.5	54.9 ±10.9	73.9 ±0.7
Ours [TSC]	80.2 ±1.2	**63.8 ±3.7**	80.5 ±1.6	63.5 ±2.5	98.8 ±0.4	94.1 ±1.7	**73.1 ±1.6**	**93.1 ±0.7**	**87.9 ±0.2**	**67.2 ±2.4**	80.2 ±0.4
Ours [TSC+OHC]	**81.2 ±0.4**	63.2 ±1.7	**87.9 ±0.7**	**77.3 ±1.6**	**99.2 ±0.4**	**94.7 ±3.0**	69.7 ±3.3	92.9 ±0.8	87.7 ±1.6	64.5 ±5.0	**81.8 ±0.4**

All Objects [12.5%]	**OPEN**	**CLOSE**	**INHAND**	**OUTOFHAND**	**RAW**	**COOKED**	**WHOLE**	**CUT**	**PEELED**	**UNPEELED**	**Mean**
ImageNet Pre-trained	90.1 ±0.0	62.3 ±0.0	83.1 ±0.0	73.4 ±0.0	97.4 ±0.0	83.8 ±0.0	68.7 ±0.0	85.1 ±0.0	87.2 ±0.0	51.2 ±0.0	78.2 ±0.0
TCN [53]	83.0 ±1.7	48.6 ±2.2	73.9 ±2.1	55.9 ±2.2	85.0 ±0.8	40.4 ±1.4	53.0 ±2.2	72.7 ±1.3	78.6 ±0.3	33.6 ±1.7	62.5 ±0.8
SimCLR [6]	91.0 ±0.8	65.3 ±1.5	79.5 ±0.9	64.6 ±2.2	97.1 ±0.6	82.0 ±3.7	70.0 ±1.8	82.1 ±1.6	85.8 ±2.3	57.2 ±0.5	77.4 ±1.0
SimCLR+TCN	87.9 ±1.2	59.7 ±1.6	76.8 ±0.4	60.7 ±2.2	95.4 ±0.7	76.1 ±4.9	63.3 ±1.8	80.3 ±0.6	82.3 ±3.5	46.3 ±3.5	72.9 ±1.3
Semantic supervision											
via EPIC action classification	88.5 ±1.1	55.0 ±1.6	83.1 ±0.9	72.2 ±0.2	95.9 ±0.7	68.8 ±3.7	60.0 ±1.8	73.4 ±2.2	79.7 ±3.3	44.5 ±1.5	72.1 ±0.8
via MIT States dataset [24]	89.9 ±0.1	58.5 ±1.6	81.1 ±1.2	70.0 ±3.1	96.3 ±1.2	78.9 ±3.3	66.5 ±2.0	86.3 ±1.3	86.6 ±1.3	50.4 ±3.9	76.4 ±0.6
Ours [TSC]	**91.9 ±0.5**	**65.6 ±1.0**	83.6 ±0.6	69.8 ±2.3	**98.2 ±0.0**	86.7 ±2.6	**74.3 ±2.7**	88.3 ±1.5	**90.3 ±0.1**	**65.2 ±0.6**	81.4 ±1.0
Ours [TSC+OHC]	**91.9 ±0.6**	63.5 ±1.6	**88.7 ±0.4**	**77.9 ±0.2**	**98.2 ±0.1**	**88.5 ±1.1**	73.5 ±0.9	**89.5 ±0.7**	89.8 ±0.6	64.8 ±2.2	**82.6 ±0.2**

All Objects [100%]	**OPEN**	**CLOSE**	**INHAND**	**OUTOFHAND**	**RAW**	**COOKED**	**WHOLE**	**CUT**	**PEELED**	**UNPEELED**	**Mean**
ImageNet Pre-trained	92.9 ±0.0	68.3 ±0.0	85.3 ±0.0	78.2 ±0.0	98.3 ±0.0	88.5 ±0.0	73.0 ±0.0	89.4 ±0.0	91.4 ±0.0	65.1 ±0.0	83.1 ±0.0
TCN [53]	87.0 ±1.0	56.7 ±0.6	80.6 ±1.0	67.2 ±0.8	94.2 ±0.9	68.5 ±2.3	65.8 ±2.3	82.7 ±1.5	85.1 ±1.5	46.4 ±4.7	73.4 ±1.4
SimCLR [6]	92.1 ±0.8	67.6 ±2.3	82.9 ±0.5	71.3 ±1.2	97.8 ±0.6	86.0 ±3.1	72.8 ±1.6	86.9 ±0.8	88.9 ±2.2	64.1 ±3.0	81.0 ±0.9
SimCLR+TCN	90.5 ±1.2	65.5 ±1.3	81.3 ±0.3	69.1 ±1.5	96.9 ±0.9	78.3 ±6.3	68.9 ±3.1	84.2 ±1.8	84.9 ±2.6	54.2 ±2.5	77.4 ±1.2
Semantic supervision											
via EPIC action classification	91.1 ±0.7	63.2 ±1.2	86.6 ±0.7	79.2 ±1.3	97.0 ±0.3	73.4 ±4.4	65.2 ±1.4	80.9 ±1.3	87.1 ±1.7	56.0 ±9.9	77.9 ±1.3
via MIT States dataset [24]	92.0 ±0.1	66.9 ±1.7	83.3 ±0.5	73.9 ±1.3	96.7 ±1.8	82.5 ±4.6	75.6 ±2.3	89.3 ±1.3	90.3 ±1.7	64.4 ±7.8	81.5 ±1.3
Ours [TSC]	**93.0 ±0.3**	**69.6 ±2.1**	86.9 ±0.6	75.8 ±2.1	**98.6 ±0.1**	89.0 ±2.5	**77.5 ±2.6**	89.8 ±1.5	92.1 ±0.6	**69.3 ±4.2**	84.2 ±1.0
Ours [TSC+OHC]	92.4 ±0.8	66.1 ±2.5	**90.5 ±0.5**	**83.1 ±1.4**	98.5 ±0.1	**91.1 ±0.8**	76.3 ±2.9	**91.2 ±1.1**	**92.5 ±1.1**	66.7 ±4.1	**84.8 ±0.4**

S6.3.2 TSC+OHC Ablations

We analyse the individual contribution of hand motion (h_i^m) and hand appearance (h_i^a) towards the performance of TSC+OHC. Table S9 shows that both components, by themselves, improve upon just TSC. Motion information gives larger boosts than appearance information; and both together lead to the best performance in the challenging setting of novel categories with limited data.

Table S9. We perform ablations on the components of TSC+OHC on the validation set. Both hand motion and appearance contribute to the performance over TSC, with motion being more important.

	Novel Objects		**All Objects**	
Linear classifier training data	12.5%	100%	12.5%	100%
TSC	72.3 ±1.3	77.8 ±0.4	78.3 ±0.3	81.2 ±0.3
TSC+OHC (appearance)	73.8 ±0.8	77.3 ±0.5	78.4 ±0.5	**82.5 ±0.6**
TSC+OHC (motion)	74.6 ±1.6	**78.7 ±0.6**	**78.8 ±0.4**	82.4 ±0.2
TSC+OHC (motion + appearance)	**75.1 ±0.4**	78.2 ±0.2	78.6 ±0.2	81.7 ±0.3

S6.3.3 Track Ablations

Mining object-level tracks from in-the-wild videos presents two challenges: a) how to select a *useful* patch to track, and b) how to successfully track it in the given ego-centric video.

Egocentric videos showcase objects that are undergoing *non-trivial* transformations (deformations, state changes, occlusion by hands). Furthermore, use of hand context could aid with tracking in egocentric videos that have large amounts of egomotion. We test the extent to which these advantages of working with egocentric videos contributes to performance. We generate several sets of tracks that ablate the two aforementioned factors. Visualizations for these tracks are shown by Figure S10, and the quantitative results are presented in Table S10.

Source for Starting Patches. We experiment with the following sources for the starting patch.

1. **Object-agnostic Starting Patch.** Here, we consider an arbitrary starting patch source, either a random crop or center crop in a frame. Random crops vary in scale and location, while the center crop is always a 256×256 crop from the 456×256 image.

2. **Starting Patch on Background Object.** We detect background objects (*i.e*. not overlapping with objects of interaction as detected by the model from [55]) using Mask RCNN [22] with a ResNet101-FPN backbone trained on MS-COCO 2017 instance segmentation dataset. We only detections for categories commonly found in kitchens and remove classes like car, train, *etc*. We only consider the 10 highest scoring detections, and sample a detection that doesn't overlap with the object-of-interaction as the starting patch.

3. **Starting Patch on Object-of-Interaction (Ours).** We use the object-of-interaction detections from Shan *et al*. [55]. As noted in the main paper, we use leave one out predictions from [55]: we split the train set into 5 parts by participants, retrain [55] on 4, use predictions on the 5^{th} (*i.e*. unseen participants); and repeat this 5 times over.

4. **Ground Truth Objects-of-Interaction (Ceiling).** For reference, we also report performance on using ground truth objects of interaction as annotated in the EPIC-KITCHENS dataset. We use these with ground truth tracking (see below).

Tracking Algorithm. We experiment with the following tracking algorithms.

1. **No Tracking.** Here, we don't do any tracking and copy over the box from the previous frame, to the same location in the current frame.

2. **Off-the-Shelf Tracker.** We use SiamRPN++ tracker from [35] to track the object from one frame to the next. Given a starting crop, the tracker produces bounding boxes for crop in consecutive frames. In practice, we only consider a tracker-produced bounding box to be valid if it has a score above 0.1. We allow for up to two frames of either missing or invalid detections, or a max of 256 frames tracked, before sub-sampling and saving the track.

3. **Hand-context (Ours).** To construct our tracks, we focus on objects-of-interaction detected by [55] along with information about what hands do they correspond to. We do this jointly with the object-of-interaction starting patches described above. In more detail, we utilize hand-object detections for both, finding the starting patch and tracking it. Specifically, we start with a frame and find all interacted objects with a score above 0.2 and start tracking them independently. At the next frame, we receive another set of valid objects bounding boxes and try to match them with the previous frame's detections by posing the problem as a linear sum assignment in a bipartite graph where the cost is the intersection over

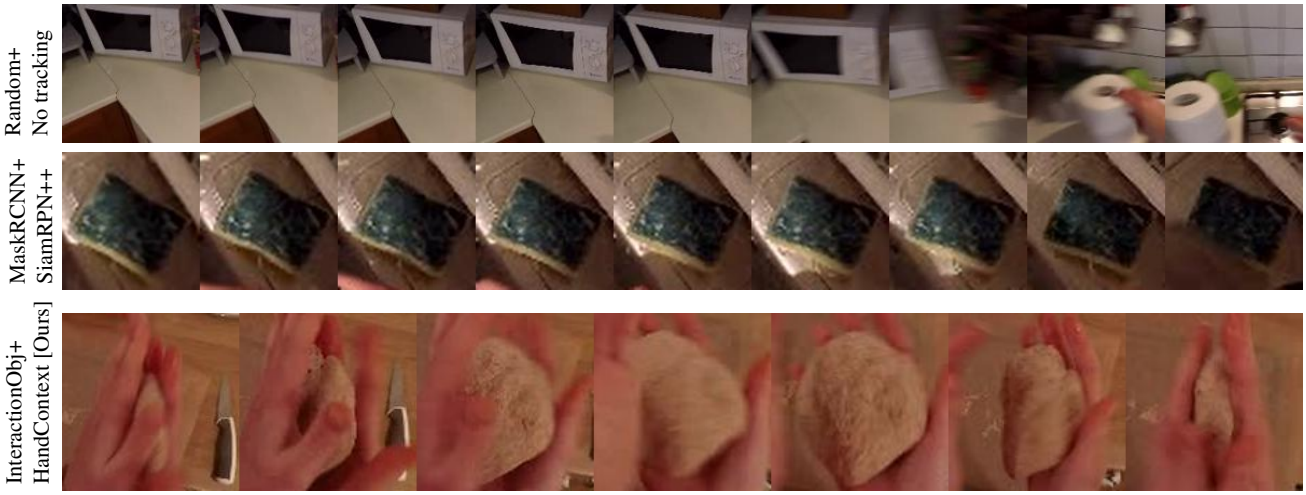

Figure S10. Sample track from the random crop with no tracking (Random), the background object crop with MaskRCNN + tracking with SiamRPN++ (MaskRCNNN + SiamRPN++), and our tracks that use objects-of-interaction and track using hand context. We see that ego-motion in egocentric videos leads to large drift by the end without any tracking. We see that the background object tracks fail to capture meaningful appearance changes. For our tracks, we see the object in a variety of poses with distinct appearances.

union (IoU) over the two bounding boxes (provided that IoU > 0.4). Boxes still not matched with previous boxes start their own track. Tracks also have an 8 frame buffer with no matches before they are closed. We subsample the tracks to 10fps. We cap the track length at 25.6s, and split longer tracks. We get a total of 61.3K tracks.

4. **Ground Truth (Ceiling).** Here we use the ground truth object-of-interaction tracks provided in the EPIC-KITCHENS dataset (used in conjunction with ground truth object-of-interaction above). We use the bounding box annotations for the object-of-interaction from Damen *et al*. [10] on EPIC-KITCHENS dataset. Since these annotations are provided at 0.5 fps, we interpolate the bounding box for the intermediate frames to get dense tracks. This gives us 16,474 tracks with an average length of 66 frames.

Results. Table S10 shows the performance of TSC on the various tracks. As noted in the main paper, use of object-of-interaction tracks offers two advantages: they stabilize for the large egomotion in egocentric videos, and focus on aspects of the scene that are undergoing interesting (non-viewpoint) transformations. No stabilization performs poorly. Stabilization using off-the-shelf tracker SiamRPN++ [35] also works well. However, tracking with hand context enables use of Object-Hand Consistency which aids performance. Ground truth tracks annotated in EPIC-KITCHENS dataset lead to better learning, indicating that better detection and tracking of objects-of-interaction can improve performance further.

Table S10. Comparison on validation set for various tracking approaches used for learning state sensitive features with Temporal SimCLR. Tracks obtained with the hand-object-interaction detector perform the best and come close to hand annotated tracks [10] in performance. Italicized rows correspond to our proposal in this paper.

Starting Patch Source	**Tracking Algorithm**	**TSC Val mAP**
Center crop	None	72.9
Random crop	None	77.1
Object-of-interaction [55]	None	79.2
Random Background Crop	SiamRPN++ [35]	77.6
Random Background Object	SiamRPN++ [35]	80.6
Object-of-interaction [55]	SiamRPN++ [35]	79.7
Object-of-interaction [55]	*Hand context*	*81.2*
GT Object-of-interaction	Ground Truth	83.5
Object-of-interaction [55]	*Hand context*	*81.7 (TSC+OHC)*

S7. Object Affordance Prediction

S7.1. EPIC-ROI Dataset and Task

Task Definition. The ROI (Region of Interaction) task is to predict regions where human hands *frequently* touch in everyday interaction. Specifically, image regions that afford any of the most frequent actions: TAKE, OPEN, CLOSE, PRESS, DRY, TURN, PEEL are considered as positive.

Data Sampling. We randomly sampled 500 images from 9 participants: P01, P08, P11, P02, P32, P18, P04, P09, P03. From these participants, we only use videos present in the test set of EPIC-KITCHENS dataset (2018 version). We annotated frames at 1920×1080 resolution. The images may or may not contain participant hands (if they were present, they were annotated and ignored during evaluation) . We manually filter out images to minimize motion blur, out of distribution frames (for example, completely dark frames at the starting of some videos or rare views such as picking a spoon that fell onto the floor). We made sure to minimize redundancy among frames by selecting the most diverse 7 – 15 frames from each participant.

Annotation Procedure. To determine where participants frequently interact in the scene, we manually watched the videos from these participants and created a list of objects that underwent interaction objects and also identified the interacted regions. Then, for every considered action, we annotated applicable regions of interaction using polygons for larger objects (such as bottles, jars *etc*.), and lines for thin regions (wires, rims and object edges). The lines for the rims and edges of objects were converted to regions by dilating them by 25 pixels to convert them to strips. Annotation for 10 images from 1 participant took 120 minutes on average. Lastly, we aggregate the annotations across all actions to generate the EPIC-ROI ground truth segmentation mask.

To enable detailed analysis, every annotation is also assigned one of the four labels: COCO objects, Non-COCO objects, COCO parts, and Non-COCO parts. To determine the set of COCO objects, we first select only the relevant classes removing categories like cat, dog, bird etc. This leaves us with the following categories: BACKPACK, UMBRELLA, HANDBAG, TIE, SUITCASE, SPORTS BALL, BASEBALL BAT, BASEBALL GLOVE, TENNIS RACKET, BOTTLE, WINE GLASS, CUP, FORK, KNIFE, SPOON, BOWL, BANANA, APPLE, SANDWICH, ORANGE, BROCCOLI, CARROT, HOT DOG, PIZZA, DONUT, CAKE, CHAIR, MOUSE, REMOTE, KEYBOARD, CELL PHONE, TOASTER, BOOK, VASE, SCISSORS, HAIR DRIER, TOOTHBRUSH, MICROWAVE, OVEN, SINK, and REFRIGERATOR. We further observed that removing MICROWAVE, OVEN, SINK, and REFRIGERATOR from the relevant categories improves the performance of Mask RCNN on the validation split (see Table S12). Thus, we don't include objects from these 4 categories into COCO objects. Figure S11 shows some annotated images from our validation split where different categories (out of above four) are assigned different colors.

Dataset Splits. We split the collected dataset into validation and testing sets based on the participants. P03, P04, and P09 are in the validation set with a total of 32 frames. P01, P02, P08, P11, P18, and P32 are in the test set with a total of 71 images.

Table S11. Val and test sets for EPIC-ROI dataset.

Split	Participants	# Frames
Validation	P03, P04, P09	32
Test	P01, P02, P08, P11, P18, P32	71

S7.2. Grasps Afforded by Objects (GAO) Task

Task Definition. The task is to predict the hand-grasps afforded by objects present in the scene where each object can afford multiple grasps. The task also requires reasoning about the occlusion between objects which can leave some of the hand-grasps inapplicable.

Datasets. We utilize YCB-Affordance dataset [9] that builds upon YCB-Videos [62] (sample frames shown in Figure S12) to set up GAO Task. The dataset annotates each object with the afforded hand grasps (see Figure S12 (right)).

For our methods and the baseline alike, we assume that objects have already been localized (we use the object masks provided with the dataset). This side-steps the detection problem and allows us to focus on the task of predicting afforded grasps. For the baselines, predictions are made on a crop around the object of interest. For our method, dense predictions from our model are aggregated over the segmentation mask to obtain the final classification (more below).

Splits. We divide the YCB-Affordance dataset into three parts for training, validation and testing. We make sure that the training, validation and test sets do not overlap in the objects. Further, there is no overlap in the videos from training, validation and test sets. This results in a training set consisting of 77 videos, validation set with 6 videos and testing set with 9 videos. We further only use 15 objects (out of 21) from the training set to train the supervised baseline. For validation and testing, we use the remaining 6 objects to compute the metrics. Since the scene is static, and the camera motion is slow, we sub-sample 60 frames (10 from each video) from the validation videos to create the validation set, and 180 frames (20 from each video) to create the testing set. We use all the frames (110K) from training videos to create the training set for supervised ceiling. The splits ensure that we test generalization to novel objects.

Figure S11. Annotated images from EPIC-ROI dataset. We show some sample images from the dataset annotated for evaluating region of interaction predictions. Each annotated region is attributed with one of the four labels: COCO objects (red), Non-COCO objects (green), COCO parts (blue), and Non-COCO parts (magenta).

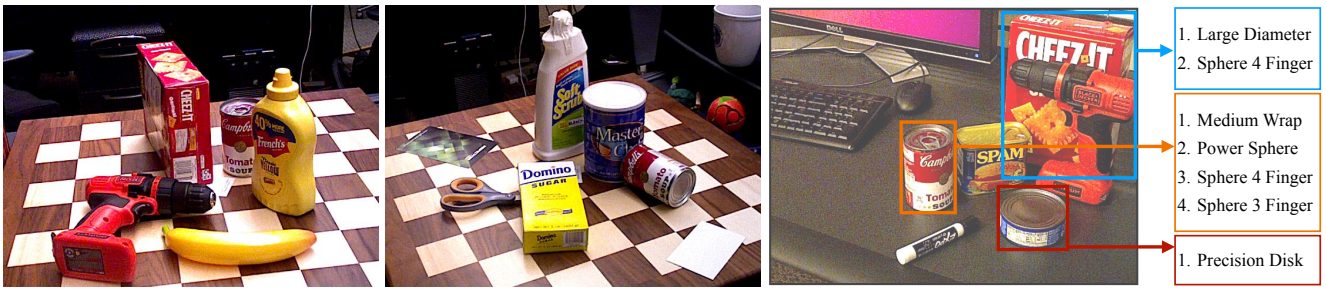

Figure S12. Sample frames from YCB-Videos [62]. We use annotations from Corona *et al*. [9] on the YCB-Videos [62] to setup the GAO task. The task is to predict the hand-grasps afforded by the objects present in the scene (see right figure). This requires reasoning about object shape and occlusion patterns that can render some grasps inapplicable.

S7.3. Model Details

S7.3.1 Affordances via Context Prediction (ACP) Details

Architecture. We use the ResNet-50 backbone as the encoder. The region of interaction branch uses a decoder that consists of 4 deconvolution layers with 4×4 kernels and stride length of 2 and a padding of 1. Lastly, we have a 5×5 average pooling layer with padding 2 and stride 1 which outputs the final ROI-prediction. The grasp prediction branch, uses one

fully-connected layer followed by 33 binary classifiers on top of the output from the encoder.

Training Splits. We use participants P05, P06, P07, P10, P12, P13, P14, P15, P16, P17, P19, P20, P21, P22, P23, P24, P25, P26, P27, P28, P29, P30, P31 for training the model. Note that these are disjoint from the participants used for validation and testing in EPIC-ROI, as listed in Table S11.

Data Sampling. For training our ACP model, we extract patches from 456×256 EPIC-KITCHENS frames (2018 version). We use the hand and object-of-interaction detections to generate the ground-truth segmentation mask (by pasting the detections). We only use object-of-interaction detections that have a confidence score ≥ 0.8, and that are smaller than 150 pixels in width and height. Next, we sample positives around the detected hands and detected objects-of-interaction. For hands, we randomly select a square patch 1 to 1.3 times the size of the detected hand box, centered at the hand (positive) or randomly located elsewhere in the image (negative). For sampling around the objects, we only consider object-of-interaction detections that have width and height greater than 20 pixels, and we sample a square patch inside the object-of-interaction box (positive). The width of the sampled patch is randomly varied between 0.5 to 0.75 times the size of the detected object-of-interaction box. We train on all participants except the ones in the EPIC-ROI validation and testing sets. For training the grasp prediction branch, we only use the positive patches sample centered at the hand.

Patches are resized to 128×128 for training. Note that the bottom center 64 region is masked out before feeding into the networks both at train and test times.

Loss function. Our loss contains two terms L_{seg} and L_{grasp}, the former training the region of interaction branch and the latter training the grasp prediction branch. Both losses encourage the network to focus on the surrounding context to make prediction about the hidden hand.

L_{seg} computes the binary cross-entropy between the predicted segmentation mask and the ground truth segmentation mask (as derived by pasting detection boxes). We weight the positive pixels by a factor of 4.

L_{grasp} is trained on predictions from a classifier trained on the GUN-71 dataset [51]. We only consider scores for the 33 hand grasps (that are annotated in YCB-Affordance dataset) from the GUN-71 classifier. We use the highest scoring class as the positive class. We create a set of negatives which consists of the least scoring $K = 15$ classes. This generates both the positive and negative data for training the grasp prediction head.

L_{seg} is trained on all positive and negative patches described above. L_{grasp} is only trained on positive patches (*i.e.* those that are around the hands).

We train both the segmentation head and the grasp prediction branch jointly by combing the two loss functions as,

$$L = L_{seg} + 0.5 \cdot L_{grasp}. \tag{9}$$

Training Details. We use a batch size of 64 and Adam optimizer with a learning rate of 10^{-4}. During training, we also perform horizontal flips, motion blur and color jitter augmentations on the input image. The masked context region is resized to 128×128 before being input to the model. We train for a total of 400 epochs (each epoch consisting of 256 iterations on randomly sampled batches with batch size of 64) and then validate checkpoints at epoch 300, 350, and 400 to select the best model for evaluation. Training took 6 hours on a single modern GPU (RTX 2080 Ti or equivalent).

Supervision for Grasp Prediction Branch. Here, we provide more details about the GUN-71 classifier that is used to generate the necessary supervision for training ACP.

This GUN-71 classifier is trained on hands cropped from the GUN-71 dataset from Rogez *et al*. [51]. We use a hand detector [55] to crop out hands from the GUN-71 dataset resulting in 8403 crops for training (Subjects 1, 2, 3, 4, 5, and 6) and 1655 crops for validation (Subject 7). The classifier uses a ResNet-18 backbone with two fully connected layers (512 and 128 units), followed by a 71-way classification layer. This model is trained using hand grasp annotations in the GUN-71 dataset with a cross-entropy loss.

In addition to this grasp classification layer, we also have another head consisting of one linear layer (128-dimensional output) that is trained using L_{temporal} on the EPIC-KITCHENS dataset to *adapt* the GUN-71 classifier to work well on EPIC-KITCHENS dataset. This L_{temporal} uses the hand tracks obtained using detector from [55] on the EPIC-KITCHENS dataset, as used by the other parts of our paper.

This network is trained jointly by minimizing $L_c + L_{\text{temporal}}$ using Adam optimizer with a learning rate of 10^{-4} and 0.05 weight decay. We use a batch size 128. We perform random crops, horizontal flips, and color jitter as augmentations. We train for a maximum of 60 epochs where we early stop based on the validation performance. For L_{temporal}, we use a window of length 10 to sample positive hand crops. We only train on tracks with a minimum length of 15 frames. We save the model after every 3 epochs, and select the snapshot based on the validation performance on the GAO task. Typically, training for 24-30 epochs resulted in the best performance where each epoch consisted of training for 64 iterations.

ACP (no L_{temporal}). This model is trained similarly to ACP but only for minimizing L_c cross entropy loss for classification.

Inference. At test time, we uniformly sample square context regions of size 60, 100, and 160 from 1920×1080 images. We sample 4000 regions for each size, resulting in a total of 12000 regions. We resize the sampled patches to 128×128 and mask out their bottom center 64×64 region, before feeding them into our learned models to obtain the 12000 64×64 predictions.

These predictions are spatially aggregated, individually for both the afforded hand grasps and regions of interaction, by resizing and pasting at the corresponding locations. For GAO task, we only use the spatially aggregated grasp predictions, and for the ROI-prediction task, we only use the spatially aggregated ROI prediction. For evaluation on EPIC-ROI, we also smooth our predictions using a Gaussian kernel (with standard deviation of 25) to suppress high frequencies. To generate affordances (for example in Figure 8), we simply multiply the two spatial predictions.

Inference on YCB-Affordance. We do inference over 800 160×160 patches to obtain pixel-wise grasp predictions for each of the 33-hand grasps. We then compute the average score within each object mask and use that to compute 33 scores, one each for each grasp type.

S7.3.2 Region of Interaction (RoI) Baselines

Mask RCNN. We use an FPN-based (Feature Pyramid Network) Mask RCNN model trained on MSCOCO with a ResNet-101 backbone for implementing this baseline. For inference, we predict 1000 detections per image with an NMS threshold of 0.7. To get the RoI prediction, we multiply the class-score with the soft instance segmentation mask, and paste it at the corresponding detection locations.

Mask RCNN [relevant]. Before pasting the predicted segmentations, we filter out detections corresponding to the relevant categories. Specifically, we consider the following object categories that we selected so as to maximize the AP on the validation set (see ablation in Table S12): BACKPACK, UMBRELLA, HANDBAG, TIE, SUITCASE, SPORTS BALL, BASEBALL BAT, BASEBALL GLOVE, TENNIS RACKET, BOTTLE, WINE GLASS, CUP, FORK, KNIFE, SPOON, BOWL, BANANA, APPLE, SANDWICH, ORANGE, BROCCOLI, CARROT, HOT DOG, PIZZA, DONUT, CAKE, CHAIR, MOUSE, REMOTE, KEYBOARD, CELL PHONE, TOASTER, BOOK, VASE, SCISSORS, HAIR DRIER, and TOOTHBRUSH.

Table S12. Selecting Relevant COCO Categories to Maximize Mask RCNN Performance. We observe that using predictions for microwave, oven, sink, refrigerator or all four, reduces the performance of Mask RCNN on validation set. This is because the region-of-interaction task requires localizing the regions of interaction on these objects and not segmenting them out as a whole. Consequently, we remove these 4 object classes from the relevant categories. We compare to this stronger Mask RCNN baseline.

	Overall AP	
Slack at segment boundaries	0%	1%
Mask RCNN [relevant]	**65.7**	72.1
Mask RCNN [relevant] w/ oven	50.0	58.1
Mask RCNN [relevant] w/ microwave	61.7	**73.1**
Mask RCNN [relevant] w/ sink	54.4	60.7
Mask RCNN [relevant] w/ refrigerator	52.6	57.7
Mask RCNN [relevant] w/ oven, microwave, sink, refrigerator	47.3	53.3

Interaction Hotspots. We use the pre-trained model (with a dilated ResNet-50 backbone) provided by Nagarajan *et al*. [42] to predict interaction hotspots on EPIC-ROI. Specifically, we uniformly sample 800 patches of size 400×400, resize to 224×224 to feed into their model, get 28×28 predictions from their model, upsample and paste these predictions at the corresponding location. We selected the (400×400) patch size based on the validation set performance. We did not observe any improvement in performance on increasing the number of patches sampled. Note that the model from [42] is a action-specific model. We convert their predictions into per-pixel interaction probability by taking the max score across actions at each pixel.

DeepGaze2. We use the predictions from DeepGaze2 [30] model to compute the AP on the RoI-prediction task.

SalGAN. We use the predictions from SalGAN [46] model to compute the AP on the RoI-prediction task.

Mask RCNN + X. We combine predictions, P_{X} from models (DeepGaze2, Ours) with predictions $P_{\text{Mask RCNN}}$ from Mask RCNN [relevant] to obtain combined predictions which are denoted as Mask RCNN + DeepGaze2 and Mask RCNN+ACP in the main paper. This is done by a pixel-wise combination with scalar weights:

$$P_X^{\text{comb}} = w \cdot P_{\text{Mask RCNN}} + (1 - w) \cdot P_{\text{X}} \tag{10}$$

We set w to $2/3$ when combining with ACP, and to $1/2$ when combining with DeepGaze2. This scalar weight was obtained through validation on the validation set. We additionally found it useful to smooth the output from our model (Gaussian filtering with standard deviation of 25 pixels, image size was 1920×1080). Such blurring wasn't useful for predictions from DeepGaze2.

S7.3.3 Grasps Afforded by Objects (GAO) Baselines

Chance. As chance performance, we report the fraction of positive data for each grasp in the dataset. This corresponds to a flat precision recall plot.

Supervised Ceiling. To train the supervised ceiling, we use the training split with 15 objects and 77 videos. We use a ResNet-50 backbone with a classifier head containing one fully-connected layer, followed by 33 binary classifiers. We train this network by sampling square patches centered at the object bounding boxes and use a binary cross entropy loss for training. We also use color jitter, horizontal flips and random crops on the sampled patches as data augmentation during training. We validate on the validation split on the held-out 6 objects.

S7.4. Detailed Results, Ablations, and Visualizations

S7.4.1 ACP Ablations

We study the effect of the different choices regarding supervision, data preparation and network input, and network architecture, made in the design of ACP. We conduct these experiments on the validation sets and report performance on the ROI task and the GAO task (where applicable). For the ROI task, we report the performance in isolation, and upon combination with Mask RCNN. Results are presented in Table S13.

Data preparation and network input. Our full model masks out the hand before feeding in patches to the network for training, and uses an asymmetrical context window around the masked region. Furthermore, we only make predictions for objects and hands when they are in contact with the hand. We ablate these choices, and find that all three of these choices contribute to the performance of the full ACP model.

Supervision and data sampling. Our full model uses the regions for both the hand and the object as target and for sampling data during training. We see a large drop in performance on the ROI task when not using the object regions for data sampling or as target (denoted as 'no object'). Not using the hand regions for data sampling or as targets (denoted as 'no hand') leads to a small drop in performance for the ROI task but additionally renders it impossible to train for the GAO task. The role of hands is further emphasized when we switch to using hand segmentation masks rather than box masks (as used in all other experiments). Richer understanding of the hands leads to improved performance on the ROI task.

Network architecture. Our ACP model as used in the main paper takes in a $2s \times 2s$ input and produces a $s \times s$ output. We also experimented with a symmetric architecture ($2s \times 2s$ input and output). This can lead to better spatial alignment and ease learning. We report metrics with two such architectures, (i) where we put the loss on the bottom center patch, and (ii) where we put the loss in the entire output window. We observe slight improvements in performance from these architectural modifications.

Table S13. Variations of ACP. Average precision for Region-of-Interaction prediction and mean average precision for GAO task, each on the respective validation sets. For ROI prediction task, we report results using raw ACP predictions as well as when combined with Mask RCNN. We train each ablation three times and report the mean and the standard deviation ($\mu \pm \sigma$).

	ROI (Overall AP)		**ROI (Overall AP) [+Mask RCNN]**		**GAO (mAP)**
Methods	0% Slack	1% Slack	0% Slack	1% Slack	
ACP (full model)	61.4 ±0.3	73.3 ±0.5	70.9 ±0.1	79.5 ±0.2	42.2 ±2.6
Data preparation and network input					
ACP (no hand hiding)	60.8 ±0.3	72.4 ±0.4	70.3 ±0.1	78.8 ±0.1	42.0 ±5.2
ACP (no contact filtering)	59.9 ±0.7	71.4 ±0.8	70.5 ±0.2	79.0 ±0.3	43.0 ±1.2
ACP (symmetric context)	60.2 ±0.2	72.4 ±0.2	70.0 ±0.2	78.5 ±0.1	39.1 ±1.0
Supervision and data sampling					
ACP (no object)	53.6 ±1.0	65.1 ±1.3	69.5 ±0.2	77.3 ±0.3	41.4 ±5.9
ACP (no hand)	60.8 ±0.8	72.8 ±0.7	70.7 ±0.3	79.3 ±0.2	N/A
ACP (hand segmentation masks as opposed to box-masks)	62.1 ±0.5	74.0 ±0.4	71.1 ±0.4	79.7 ±0.4	42.5 ±2.7
Network architecture					
ACP ($2s \times 2s$ output, loss everywhere)	61.5 ±0.4	73.7 ±0.5	70.6 ±0.2	79.2 ±0.1	40.7 ±2.6
ACP ($2s \times 2s$ output, loss on bottom center)	61.7 ±0.3	73.4 ±0.1	71.1 ±0.2	79.7 ±0.2	41.6 ±5.0

S7.4.2 GAO Category-wise Performance

The test set only contains 7 (out of 33) grasps, we report the mean average precision over these 7 categories. The class-wise performance for ACP is shown in Table S14. We also report the chance performance along with a supervised ceiling.

Table S14. Class-wise performance on the GAO test set. We report average precision for each of the 7 hand grasp type contained in the test set. For ACP (no L_{temporal}) and ACP, we conducted the experiment three times and report the mean performance. We also report the standard deviation over three runs for ACP and ACP (no L_{temporal}).

Grasp Type	**Chance**	**ACP** (no L_{temporal}) *[Ours]*	**ACP** *[Ours]*	**Supervised Ceiling**
Large Diameter	55.6	56.3 $\pm$5.7	45.2 $\pm$3.6	80.2
Medium Wrap	27.8	26.6 $\pm$5.8	20.4 $\pm$1.5	67.2
Power Sphere	27.8	23.6 $\pm$0.7	36.0 $\pm$3.6	68.4
Precision Disk	22.2	15.1 $\pm$0.8	14.9 $\pm$1.5	94.7
Parallel Extension	11.1	11.0 $\pm$0.4	28.1 $\pm$5.6	14.8
Sphere 4 Finger	50.0	68.8 $\pm$6.9	64.6 $\pm$1.9	56.6
Sphere 3 Finger	16.7	39.3 $\pm$6.2	57.3 $\pm$0.2	15.4
Mean	30.2	34.3 $\pm$0.8	38.1 $\pm$0.2	56.8

S7.4.3 Qualitative Results

1. We provide additional visualizations for ROI predictions made by ACP on the validation split of EPIC-ROI dataset (see Figure S13). We observe that our method can locate regions that afford interaction such as drawer handles, knobs and buttons which are not typically annotated in object segmentation datasets. We also see that predictions are localized to object regions that afford interaction *e.g.* edges of plates.

2. We also visualize predictions for afforded grasps on the EPIC-KITCHENS dataset. We convert predicted grasp-specific heatmaps into detections (by finding *scale-space blobs* in the heatmaps) and visualize the top scoring detections across the validation dataset in Figure S14.

3. In Figure S15, we show the top detections for each of the 7 grasps made by ACP on the validation set of GAO benchmark. The images are colored green if the corresponding grasp is applicable to the highlighted object or else colored in red. We observe that for many hand-grasp types, the top scoring objects actually afford the corresponding hand-grasp type.

Figure S13. Regions-of-Interaction (ROI) predictions on EPIC-ROI dataset. We show ROI predictions on 18 images from the validation dataset. We observe that our method can locate regions that afford interaction: drawer handles, knobs and buttons (not typically annotated in object segmentation datasets). We also see that predictions are localized to object regions that afford interaction *e.g.* edges of plates.

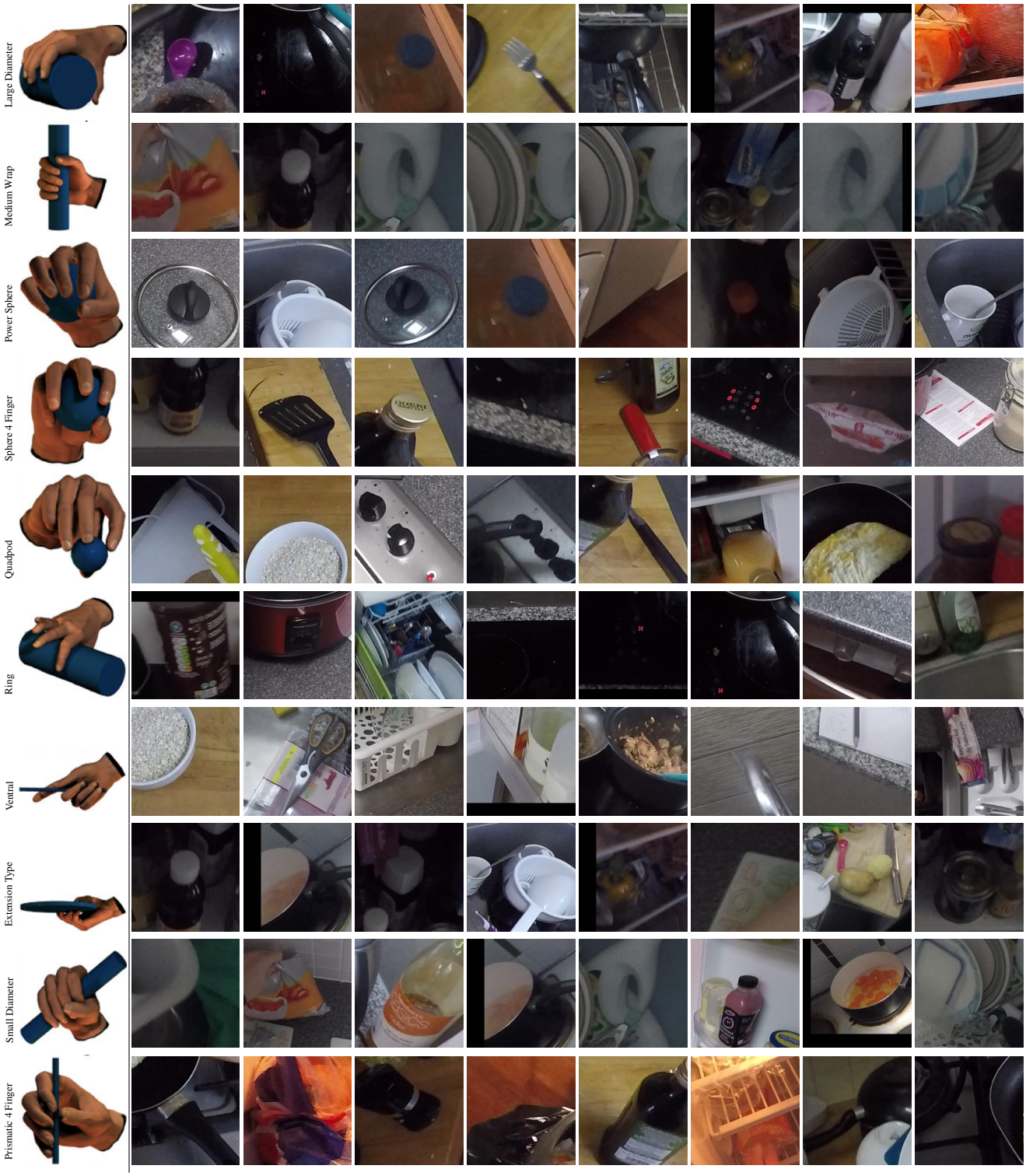

Figure S14. Visualizations for Afforded Grasps. Top detections for selected hand grasp types on the validation split of EPIC-ROI dataset. We convert predicted grasp-specific heatmaps into detections (by finding scale-space blobs in the heatmaps) and visualize the top scoring detections across the validation dataset. Many of these detections are plausible, *e.g.* lid handles for power sphere, and sphere 4 finger grasps; bottle caps and stove knobs for quadpod grasp.

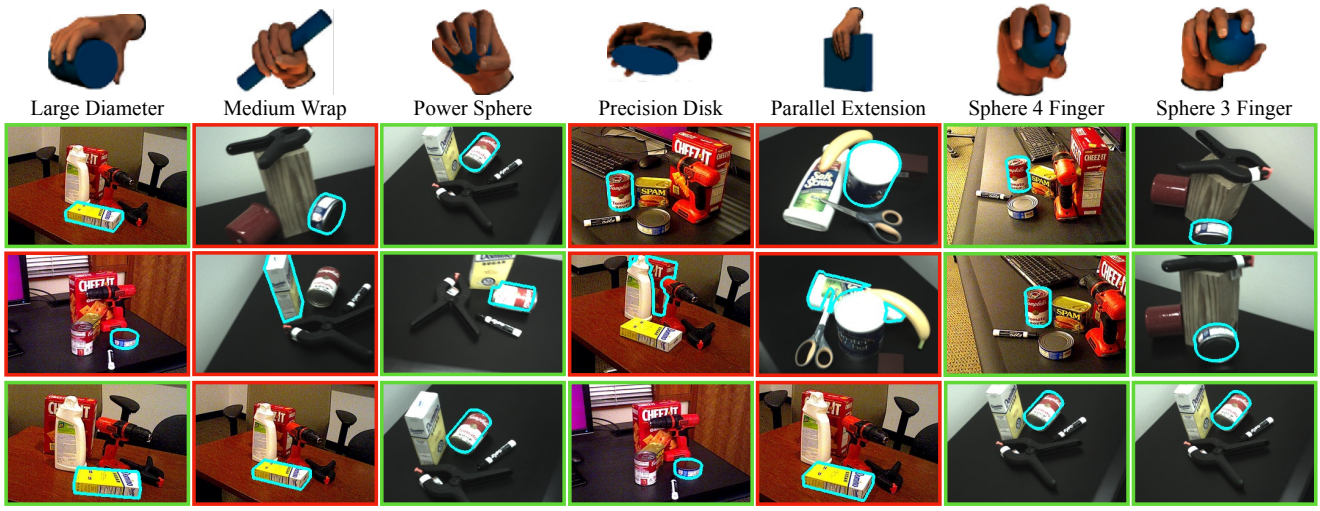

Figure S15. Visualizations of predictions for GAO task on YCB-Affordance dataset. Here we show the top predictions (object-wise) made by ACP for the 7 grasps contained in the validation set. For each of the 7 grasps, we visualize the grasp (reproduced from [15]) in the top row, and show the top three predictions (after removing images that were very similar). We highlight the object for which the prediction is being made for (in cyan). We color the image frame to indicate correctness of prediction based on ground truth from Corona *et al*. [9] (green indicates correct, red indicates incorrect).