Learning rich touch representations through cross-modal self-supervision
Supplementary material

Martina Zambelli
Deepmind, London, UK
zambellim@google.com

Yusuf Aytar
Deepmind, London, UK
yusufaytar@google.com

Francesco Visin
Deepmind, London, UK
visin@google.com

Yuxiang Zhou
Deepmind, London, UK
yuxiangzhou@google.com

Raia Hadsell
Deepmind, London, UK
raia@google.com

1 Considerations on CoRL2020 and the Pandemic

During the few months before the CoRL2020 deadline, we did not have a reach to our real robots, due to the lockdown that followed the pandemic outbreak. This implied that we could not carry on our experiments (in particular the data collection) on the real robotic platform. We were still able to collect that data in simulation, using Mujoco, a simulated Shadow hand, and a set of diverse objects, both synthetic (generated by us) and externally available realistic objects (i.e. the YCB objects).

Despite the limitations on the data collection, the main contributions of our paper still hold: the proposed approach is mainly related to perception, and it is agnostic of the specific data source that is used for training. All the networks and architecture used for self-supervision can be easily adapted, if needed, to real world data. Our approach can be applied to data coming from a real robot setup, provided that tactile data (of some form) can be collected, to learn touch features.

2 Dataset generation

Props Props are generated in Mujoco as entities of four different shapes: cube, sphere, cylinder, ellipsoid. Each shape is generated by setting either a height and/or a radius, sampling from $\{0.030, 0.035, 0.040, 0.045, 0.050, 0.055\}$cm.

To generate deformable versions of these props we construct composite entities starting from simple shapes: each soft prop is composed of several elements or sub-body, called capsules. Each capsule is built by defining its dimensions (length and radius), and a soft prop is then characterized by the number of capsules that will form the prop itself: the more capsules, the denser the prop; the bigger the capsules, the bigger the prop.

As explained in the paper, we generated two sets of data including disjoint sets of props. We chose random disjoint subsets and fixed the selections for repeatability. The two sets are composed as follows, where each number correspond to an object's class: $\text{Set1} = [29, 4, 26, 30, 32, 37, 34, 40, 7, 10, 11, 31, 33, 27, 47, 2, 46, 18, 15, 28, 22, 16, 41, 20]$, $\text{Set2} = [42, 8, 13, 25, 5, 17, 35, 14, 38, 1, 12, 43, 24, 6, 23, 36, 21, 19, 9, 39, 45, 3, 0, 44]$. Each object's class is computed from a triplet of codes identifying the shape, the size and the rigidity (binary) of each prop. There are on average 1700 instances for each class.

YCB objects YCB objects are imported as entities in Mujoco from the open source dataset. Each object is imported with its standard size and it is proportionate to the Shadow hand model. We chose a set of ten objects, following [1]: cracker box, sugar box, mustard bottle, potted meat can, banana, pitcher base, bleach cleanser, mug, power drill, scissors.

In order to simplify the data acquisition, each object appears in a reachable position for the hand. In this way, the MPO agent can more easily learn a policy that maximizes the contact points between the fingertips and the different objects, without spending too much time to find the object in the scene first. We explicitly chose to keep the setup as simple as possible, to keep the focus of this work on the touch representations learned, rather than on the policy used to explore the space.

4th Conference on Robot Learning (CoRL 2020), Cambridge MA, USA.

We vary the rotation of each object around its vertical axis, and the tilt of the object from the vertical axis. Rotations are sampled from the set of $\{0, \pm 30, \pm 60, \pm 90, \pm 120, \pm 150\}$ degree angles. Tilt angles are sampled from the set $\{0, \pm 15, \pm 30\}$ degrees.

Two disjoint subsets are generated also for YCB objects to perform the evaluation on unseen objects. The two sets are as follows: Set1 = [banana, bleach, cracker box, driller, mug], Set2 = [mustard bottle, pitcher, potted meat, scissors, sugar box]. There are on average 10000 instances of each objects, presented with different orientations.

3 Tactile sensors

In simulation, each fingertip has a spatial touch sensor attached, with a spatial resolution of 4×4 and three channels: one for normal force and two for tangential forces (in Newtons). Thus, each fingertip has 16 channels that can collide with other objects, and that can sense 3D forces during such contacts. We simplify this by taking the absolute value and then summing across the spatial dimensions, to obtain a single 3D force vector for each fingertip.

On our real Shadow Hand, fingertips are equipped with BioTac® sensors [2, 3], which provide a more complex array of tactile signals. Despite the difference with the simulated sensors, readings from the pressure channel of each tactile sensor can be acquired and then normalized to match the range of the simulated tactile sensors. In this way, it is possible to directly compare results in simulation and on the real robot, without having to change the parameters of the task or learning algorithm. On the other hand, these adjustments are not necessary, since our proposed learning methods are totally agnostic and independent of the input dimensions and characteristics. Hence, new representations can be easily trained from real world robot data.

4 Network and training implementation details

We train the self-supervised cross-modal representation for $300\,000$ iterations with a batch size of 16 where each sample has 200 time steps. We finetuned the model for few-shot classification for $50\,000$ steps. We perform gradient-descent on GPU with an Adam optimizer using a fixed learning rate of 0.001 for CM-GEN and 0.0001 for all the other networks. The learning rate for few-shot finetuning is fixed to 0.0001 for all feature types.

Vision encoding: $64 \times 64 \times 3$ frames are processed through a convolutional encoder with residual connections; we use 4 residual blocks with $[8, 16, 32, 64]$ channels and ReLU activations. Each residual block has 3 conv2D layers with 3×3 filters and 1×1 stride, followed by 3×3 pooling layers with 2×2 stride. The resulting features are flattened and downsized to 128 dimensions through a linear layer.

Touch and proprioception encoding: to process the concatenation of proprioception and touch we use a four-layers MLP with $[256, 256, 256, 128]$ channels respectively, and ReLUs nonlinearities. The MLP embeddings are processed by an LSTM with size 128, and skip-connections.

$\mathbf{L^3}$-net: On top of the encoded multimodal pair 2-layer MLP with 256 channels (hidden layer size) and a linear layer for binary classification output are used. ReLU activations are applied in the MLP.

CMDC: Similar to L^3-net, 2-layer MLP with 256 channels with ReLU activations are utilized here. A final linear layer computes the logits for the multi-class classification. The predicted distance intervals are $\{[0], [1], [2-3], [4-5], [6-200]\}$.

TCN and CMC: n-pairs implementation doesn't require any extra layers.

CM-CPC and CM-CPC-H: For each time step a linear layer with embedding size of 128 is used to predict future (or history). CM-CPC is trained to predict 20 steps ahead. CM-CPC-H is trained to predict 20 steps into the future, the current frame, and 20 steps back into the history.

CM-GEN: The decoder network has 5 conv2D layers with $[256, 128, 64, 32, 16]$ channels and 3×3 filters. A 2×2 bilinear resize operator followed by leaky ReLU is applied after each convolutional layer. Finally a single conv2D layer with 3×3 filter is utilized to generate the $64 \times 64 \times 3$ image.

Few-shot classifier: This classifier is only utilized in few-shot finetuning stage. It is a 3-layers MLP, with size $[256, 256, c]$, where c is the number of classes for the relevant classification task.

5 Supplementary experimental results

We report here other visualizations of our experimental results. The following tables show the proposed methodology performance on few-shot classification of unseen objects. Table 1 and 2 show results of few-shot object classification on unseen objects that belong to the same dataset. Table 3 shows the results of few-shot classification across the two datasets. Learned features are trained on one of the sets, and evaluated on few-shot classification on the other set, using 1, 3, 5, 10, 25, 50 or 100 examples for each object's class.

Fig. 1 reports all few-shot classification experiments on pose estimation on the YCB objects dataset. Table 4 reports all accuracy scores for all objects and all methods for the 10-shots experiment.

Table 1: Evaluation on unseen objects - Props classification, from Set2 to Set1, and from Set2 to Set1. The mean across these results is also reported. Darker colors indicate better scores.

	Props: set1 → set2								Props: set2 → set1							
	1	3	5	10	25	50	100	mean	1	3	5	10	25	50	100	mean
CM-CPC	30.0%	41.7%	58.7%	71.7%	81.7%	87.8%	91.6%	66.1%	24.8%	42.6%	49.1%	67.9%	81.8%	86.8%	93.1%	63.7%
CM-CPC-H	29.5%	46.0%	60.3%	73.1%	83.1%	86.9%	92.6%	67.4%	22.1%	41.1%	60.2%	71.7%	83.9%	88.8%	92.6%	65.8%
CM-GEN	34.5%	49.8%	61.6%	72.1%	78.4%	85.6%	88.0%	67.1%	33.9%	51.5%	65.4%	70.4%	80.4%	85.0%	90.5%	68.1%
CMC	28.9%	44.8%	54.1%	67.3%	84.3%	86.9%	92.6%	65.6%	24.8%	41.7%	55.9%	69.6%	83.5%	90.2%	91.5%	65.3%
CMDC	26.9%	46.9%	57.0%	71.8%	80.5%	90.7%	92.8%	66.7%	28.6%	41.4%	55.5%	70.8%	81.3%	89.3%	92.4%	65.6%
L3-NET	27.2%	38.9%	52.4%	70.7%	84.7%	87.4%	91.8%	64.7%	26.1%	40.9%	52.1%	68.5%	80.8%	88.1%	93.8%	64.3%
TCN	26.7%	43.8%	56.9%	70.3%	82.6%	88.1%	92.5%	65.8%	28.6%	44.8%	56.9%	69.0%	85.8%	89.1%	91.6%	66.5%
scratch	29.2%	41.1%	57.3%	71.2%	82.3%	90.7%	90.2%	66.0%	21.6%	40.4%	56.8%	66.8%	79.4%	85.0%	89.1%	62.7%

Table 2: Evaluation on unseen objects - YCB objects classification, from Set1 to Set2, and from Set2 to Set1. The mean across these results is also reported. Darker colors indicate better scores.

	YCB: set1 → set2								YCB: set2 → set1							
	1	3	5	10	25	50	100	mean	1	3	5	10	25	50	100	mean
CM-CPC	21.7%	32.5%	36.5%	45.1%	55.3%	64.3%	71.6%	46.7%	26.5%	32.1%	35.1%	38.7%	54.7%	62.6%	69.5%	45.6%
CM-CPC-H	29.0%	37.1%	36.5%	46.7%	58.5%	67.0%	71.2%	49.4%	29.8%	32.3%	37.6%	45.7%	60.7%	63.6%	73.1%	49.0%
CM-GEN	32.0%	46.2%	49.9%	52.7%	58.5%	62.3%	66.3%	52.6%	30.6%	41.2%	49.6%	52.9%	56.7%	62.2%	67.2%	51.5%
CMC	27.8%	30.6%	38.0%	41.3%	59.0%	68.8%	75.5%	48.7%	30.8%	28.3%	37.3%	41.5%	55.9%	70.2%	78.4%	48.9%
CMDC	29.2%	31.4%	39.1%	40.5%	60.4%	69.1%	77.5%	49.6%	29.3%	29.5%	39.5%	41.9%	59.1%	67.3%	74.8%	48.8%
L3-NET	27.5%	29.6%	33.1%	42.7%	57.5%	69.0%	75.8%	47.9%	29.8%	36.7%	36.0%	43.5%	57.5%	67.0%	76.6%	49.6%
TCN	30.1%	33.5%	34.0%	43.0%	57.2%	69.4%	73.6%	48.7%	31.7%	30.4%	37.2%	37.7%	56.2%	67.2%	76.5%	48.1%
scratch	26.2%	28.9%	33.6%	40.3%	53.1%	51.3%	59.9%	41.9%	24.5%	33.6%	32.2%	41.0%	50.6%	62.5%	61.8%	43.7%

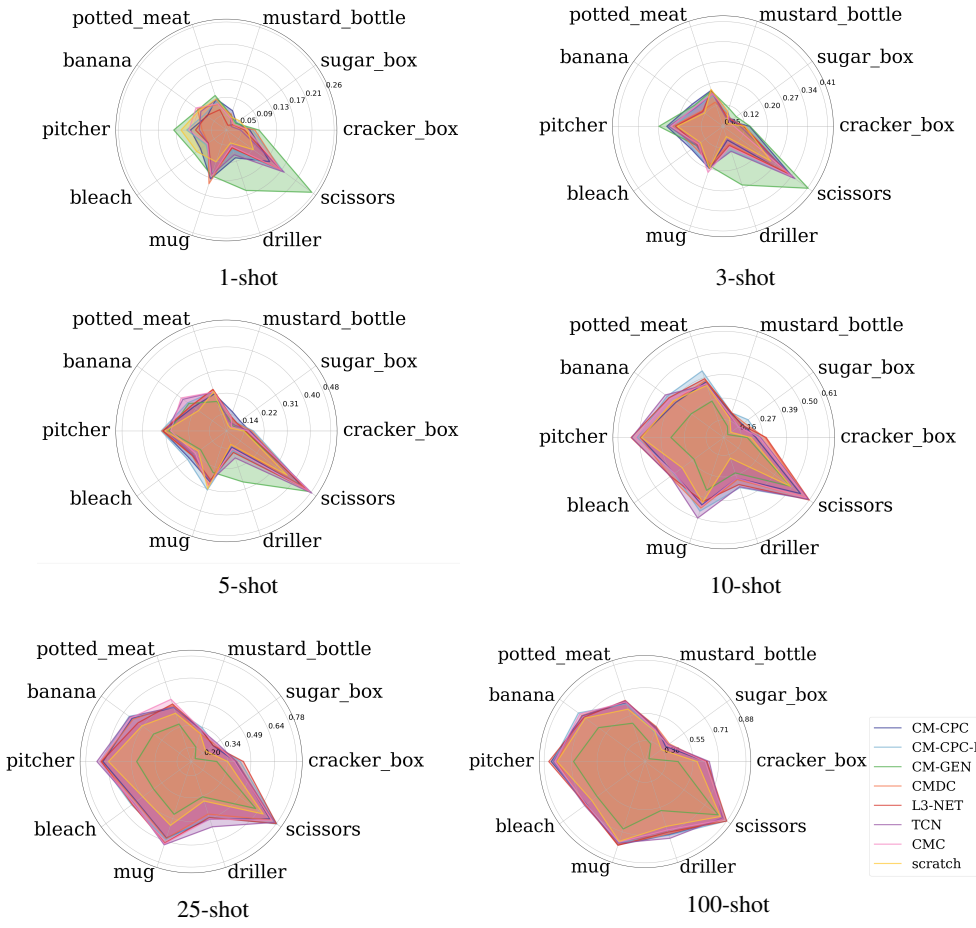
Table 3: Evaluation on unseen objects across datasets, from Props to YCB objects, and from YCB objects to Props. he mean across these results is also reported. Darker colors indicate better scores.

	Props: all → YCB: all								YCB: all → Props: all							
	1	3	5	10	25	50	100	mean	1	3	5	10	25	50	100	mean
CM-CPC	12.9%	15.5%	19.5%	23.5%	31.6%	45.7%	57.3%	29.4%	09.3%	23.3%	31.2%	49.6%	70.2%	83.4%	85.4%	50.4%
CM-CPC-H	14.0%	16.1%	16.6%	26.8%	33.4%	44.7%	59.6%	30.2%	10.0%	25.3%	31.6%	56.2%	74.8%	85.5%	86.9%	52.9%
CM-GEN	13.3%	17.8%	18.5%	23.0%	24.9%	33.0%	38.0%	24.1%	06.5%	11.1%	17.8%	28.6%	51.1%	64.4%	77.8%	36.8%
CMC	15.1%	19.0%	20.5%	28.2%	35.4%	46.7%	59.0%	32.0%	09.2%	22.5%	30.2%	49.1%	69.2%	76.2%	84.2%	48.7%
CMDC	15.4%	16.8%	21.3%	29.0%	37.0%	47.6%	58.2%	32.2%	10.4%	23.8%	30.1%	49.1%	72.4%	78.1%	85.7%	49.9%
L3-NET	15.9%	17.4%	18.9%	25.9%	37.1%	47.6%	57.8%	31.5%	11.5%	22.2%	26.8%	46.5%	72.0%	82.6%	86.6%	49.7%
TCN	16.4%	19.7%	21.8%	26.9%	36.5%	46.2%	57.7%	32.2%	11.2%	24.8%	31.6%	50.5%	72.9%	83.2%	85.4%	51.4%
scratch	13.1%	15.2%	17.9%	25.0%	30.1%	36.0%	48.7%	26.6%	09.2%	23.9%	28.1%	48.6%	64.5%	79.2%	84.1%	48.2%

Table 4: Evaluation on 10-shots pose estimation of all ten YCB objects. Darker colors indicate better scores.

	banana	bleach	cracker_box	driller	mug	mustard_bottle	pitcher	potted_meat	scissors	sugar_box
CM-CPC	37.3%	38.2%	23.4%	28.7%	43.0%	19.7%	49.6%	36.6%	55.5%	18.6%
CM-CPC-H	42.7%	38.2%	25.5%	33.7%	46.3%	20.1%	51.2%	42.6%	60.3%	22.3%
CM-GEN	27.5%	25.7%	19.0%	25.9%	35.2%	13.1%	33.9%	26.4%	49.1%	09.5%
CMC	40.7%	38.0%	25.3%	28.7%	43.9%	18.5%	52.0%	35.7%	59.1%	16.4%
CMDC	38.4%	40.8%	28.1%	29.4%	44.6%	20.1%	49.5%	37.4%	58.7%	19.8%
L3-NET	41.4%	40.5%	28.6%	32.1%	41.2%	20.2%	54.3%	38.4%	61.0%	18.1%
TCN	43.8%	40.6%	25.8%	33.3%	50.0%	18.1%	54.0%	35.2%	60.4%	16.5%
scratch	38.7%	33.1%	20.8%	18.0%	41.7%	17.1%	48.9%	34.8%	49.2%	11.5%

Figure 1: Pose estimation accuracy of YCB objects for {1, 3, 5, 10, 25, 50, 100}-shot classification. Larger area is better.



References

[1] S. Hampali, M. Rad, M. Oberweger, and V. Lepetit. Honnotate: A method for 3d annotation of hand and object poses. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3196–3206, 2020.

[2] J. A. Fishel and G. E. Loeb. Sensing tactile microvibrations with the biotac—comparison with human sensitivity. In *Proceedings of the Fourth IEEE RAS & EMBS International Conference on Biomedical Robotics and Biomechatronics (BioRob)*, 2012.

[3] I. SynTouch. BioTac Technologies. https://www.syntouchinc.com/en/sensor-technology/.