CIGMO: Categorical invariant representations in a deep generative framework (Supplementary material)

Haruo Hosoya[1]

[1]Brain Labs., ATR International, Kyoto, Japan

1 DATASET DETAILS

ShapeNet We used an object image dataset derived from a core subset of (public-domain) ShapeNet database of 3D object models (Chang et al., 2015) used in SHREC2016 challenge[1]. The subset contained 55 object classes and did not include material data. We selected 10 out of the pre-defined classes: car, chair, table, airplane, lamp, boat, box, display, truck, and vase. Our criterion of selection was those with a large number of object identities but avoid including visually similar classes, e.g., chair, sofa, and bench. We rendered each object in 30 views consisting of 15 azimuths (equally dividing $360°$) and 2 elevations ($0°$ and $22.5°$ downward) in a single lighting condition; the images were gray-scale and had size 64×64 pixels. All the rendering used Blender software[2]. We divided the data into training and test following the split given in the original database.

MVC Cloth We used a subset of MVC Cloth image dataset (Liu et al., 2016)[3], which contains a number of photos of cloths worn by fashion models; the same cloth type is shown in multiple viewing angles. The dataset provides no class label, but provides 264 binary attribute labels that are related to cloth kinds (Dresses, Denim, TShirts, etc.), cloth styles (Short, Sleeveless, LongSleeves, etc.), cloth materials (Cotton, Nylon, Black, White, etc.), and prices (hundred1U, fiftyU, etc.). We rescaled the images to 64×64 pixels and split the data into training and test sets (each of size $\sim$112K and $\sim$28K) so that the cloth types do not overlap between these.

2 PLATFORM DETAILS

We used 3 computers with the following specifications: (1) 56 core CPUs (256G memory) with 4 V100 GPUs (16G memory each), (2) 56 core CPUs (256G memory) with 4 V100 GPUs (32G memory each), and (3) 96 core CPUs (256G memory) with 4 A100 GPUs (40G memory each). All code is implemented with Python (3.7.4) / Pytorch (1.7.1).

3 ARCHITECTURE DETAILS

In a CIGMO model, the categorizer deep net u consisted of three convolutional layers each with 32, 64, and 128 filters (kernel 5×5; stride 2; padding 2), followed by two fully connected layers each with 500 intermediate units and C output units. These layers were each intervened with Batch Normalization and ReLU nonlinearity, except that the last layer ended with Softmax. The shape and view encoder deep nets had a similar architecture, except that the last layer was linear for encoding the mean (g and h_c) or ended with Softplus for encoding the variance (r and s_c). The decoder deep nets f_c had two fully connected layers (103 input units and 500 intermediate units) followed by three transposed convolutional layers each with 128, 64, and 32 filters (kernel 6×6; stride 2; padding 2). These layers were again intervened with Batch Normalization and ReLU nonlinearity, but the last layer was Sigmoid. To save the memory space, the shape encoders shared the first four layers for all categories and for mean and variance. The decoders shared all but the first layer for all categories.

4 ADDITIONAL RESULTS FOR SHAPENET

In Section 3.2 and Section 3.3, we have raised several design alternatives in the model construction. The first choice regards how to combine instance-specific categorical probability distributions and has three options: averaging (default), normalized product, and logit averaging. The second choice regards whether views are dependent on category or not (default). Table 1 summarizes performance results in invariant clustering and one-shot identification tasks, changing the

[1]`https://shapenet.cs.stanford.edu/shrec16/`

[2]`https://www.blender.org`

[3]`https://github.com/MVC-Datasets/MVC`

Accepted for the 38th Conference on Uncertainty in Artificial Intelligence (UAI 2022).

options from the default, for ShapeNet. Overall, the default design tended to give slightly better performance than the other options (though mostly statistically insignificant in invariant clustering). In particular, we could not see any advantage of using category-dependent view representations despite potentially different meanings of views. In addition, Tables 2 and 3 compare the design options in terms of degree of shape-view disentanglement and swapping errors, respectively. The results again show that the default design tended to give slightly better performance than the other options (though often statistically insigificant). As an additional remark, we found the product option numerically rather unstable.

References

Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. ShapeNet: An Information-Rich 3D Model Repository. 2015.

Kuan-Hsien Liu, Ting-Yen Chen, and Chu-Song Chen. MVC: A dataset for view-invariant clothing retrieval and attribute prediction. *6th ACM Int. Conf. Multimed. Retr.*, pages 313–316, 2016.

Table 1: Comparison of design options in CIGMO in terms of (1) how to combine category distributions: averaging (AV; default), product (PD), and logit-averaging (LA) and (2) whether views depend on the category or not (default). The comparisons are made with invariant clustering accuracy (left half; %) and one-shot identification accuracy (right half; %) for ShapeNet.

		invariant clustering (%)			one-shot identification (%)		
(1)	(2)	3 classes	5 classes	10 classes	3 classes	5 classes	10 classes
AV	N	**94.83** $\pm$ 6.06	**89.36** $\pm$ 4.53	68.53 $\pm$ 4.24	**27.33** $\pm$ 0.55	**24.51** $\pm$ 0.68	**21.79** $\pm$ 0.71
PD	N	90.44 $\pm$ 9.11	80.96 $\pm$ 9.65	68.30 $\pm$ 3.64	26.64 $\pm$ 0.57*	23.55 $\pm$ 0.56*	20.63 $\pm$ 0.80*
LA	N	88.16 $\pm$ 15.89	89.16 $\pm$ 3.08	**69.55** $\pm$ 3.26	26.52 $\pm$ 0.84*	23.57 $\pm$ 0.34*	20.58 $\pm$ 0.91*
AV	Y	92.32 $\pm$ 8.39	82.60 $\pm$ 6.53*	65.03 $\pm$ 6.62	26.54 $\pm$ 0.61*	24.16 $\pm$ 0.58	21.24 $\pm$ 1.00

Table 2: Comparison of different design options in terms of degree of shape-view disentanglement for ShapeNet, measured as neural network classification accuracy (%) for object identity from the shape (left half; higher is better) or view variable (right half; lower is better).

		shape $\rightarrow$ id			view $\rightarrow$ id		
		3 cats.	5 cats.	10 cats.	3 cats.	5 cats.	10 cats.
AV	N	**57.62** $\pm$ 0.93	**50.91** $\pm$ 0.97	**46.28** $\pm$ 0.87	**0.26** $\pm$ 0.04	0.65 $\pm$ 0.05	0.67 $\pm$ 0.07
PD	N	57.18 $\pm$ 1.30	49.13 $\pm$ 1.06*	44.17 $\pm$ 1.13*	0.27 $\pm$ 0.03	**0.63** $\pm$ 0.08	0.69 $\pm$ 0.04
LA	N	56.31 $\pm$ 1.90	49.34 $\pm$ 0.62*	43.35 $\pm$ 1.42*	0.29 $\pm$ 0.05	0.69 $\pm$ 0.10	0.73 $\pm$ 0.07
AV	Y	55.30 $\pm$ 0.78*	48.57 $\pm$ 0.82*	44.28 $\pm$ 1.21*	0.28 $\pm$ 0.05	0.63 $\pm$ 0.14	**0.63** $\pm$ 0.11

Table 3: Comparison of different design options in terms of swapping error for ShapeNet.

		3 cats.	5 cats.	10 cats.
AV	N	**0.220** $\pm$ 0.025	**0.300** $\pm$ 0.025	0.340 $\pm$ 0.035
PD	N	0.245 $\pm$ 0.031	0.333 $\pm$ 0.037	0.361 $\pm$ 0.025
LA	N	0.236 $\pm$ 0.031	0.301 $\pm$ 0.019	**0.340** $\pm$ 0.025
AV	Y	0.236 $\pm$ 0.047	0.318 $\pm$ 0.038	0.364 $\pm$ 0.060