3DQ-Nets: Visual Concepts Emerge in Pose Equivariant 3D Quantized Neural Scene Representations

Mihir Prabhudesai[*], Shamit Lal[*], Hsiao-Yu Fish Tung,
Adam W. Harley, Shubhankar Potdar, Katerina Fragkiadaki
{mprabhud,shamitl,htung,aharley,smpotdar,katef}@cs.cmu.edu
Carnegie Mellon University

1. Introduction

Concept learning lies at the very heart of intelligence, providing organizing principles with which to comprehend the world (6). Most computer vision models learn concept classifiers or detectors using labelled examples of object boxes, poses and categories. Self-supervised or unsupervised approaches mostly focus on (pre)training CNNs on auxiliary pretext tasks to lessen the need for human labels for a downstream recognition task. The visual "concepts" learnt from pretext tasks are implicit and represented as distributed neural CNN activations (7). We see the following limitations with representing concepts (solely) as neural activations across multiple layers of a deep network: i) visual memory and computation are not separated (3), which means that computation increases exponentially with the number of visual concepts learnt, ii) concepts are not stored somewhere and cannot be referred to or retrieved on demand, iii) the number of concepts cannot increase automatically based on new visual experiences, rather it is determined by the processing architecture; this misaligns with the idea that animals are capable of spontaneous concept instantiation in novel scenes (2), iv) concepts cannot be mentally manipulated by imagining variations, transformations or mental simulations (8), v) concepts do not have any spatial dimension and are hard to use for spatial reasoning.

In light of the above, we explore automated visual concept learning in a 3D visual feature space inferred from 2.5D (RGB-D) input images using differentiable fully convolutional inverse graphics networks.

Our model, which we call 3DQ-Nets, detects objects in the inferred 3D scene feature representations, and quantizes the object 3D feature maps into a set of scale and pose canonical 3D concepts, as shown in Figure 1. 3DQ-Nets operate as a learnable inverse graphics engine: they reverse camera projection and infer a set of 3D prototypes alongside their scales and 3D poses, that when neurally rendered, match the input 2.5D images. Just like the environment in a graphics engine, our 3D visual feature space is free of cross-object interferences or occlusions. As our 3D visual concepts learned do not account for partial object visibility it makes them maximally compressible: a small set of prototypes can explain object images taken from different camera views and various spatial arrangements.

We do not know of any other system that pursues 3D object perception from the type of supervision considered in this work. Our contributions over previous approaches are summarized as follows: (i) A model that learns to detect objects in 3D without 3D annotations. We outperform baselines that learn 3D detection by triangulating 2D bounding boxes. (ii) A method to summarize object instances into prototypes in a pose-equivariant way.

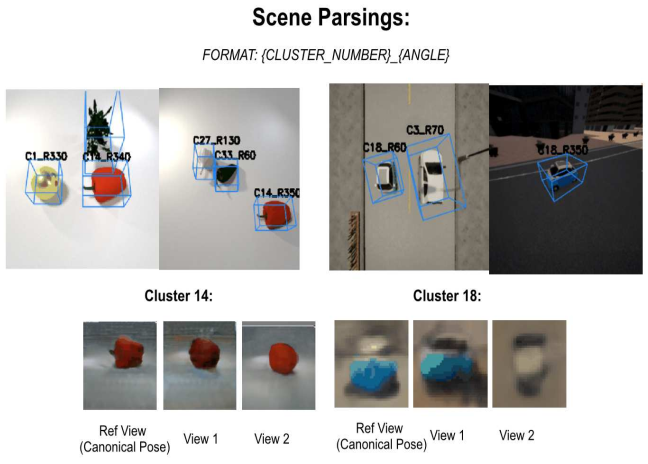

Figure 1: **3D scene parsing using 3DQ-Nets.** We show that given a RGB-D scene our model can parse the object instances in the scene as a list of prototypes (C) and their rotation(R) wrt to their canonical pose. Our real world scene parsing results can be found here : **[website link]**

[*]Equal contribution

2. Object-Quantized 3D Mapping Networks (3DQ-Nets)

The architecture of our model is depicted in Figure 2. Given a set of posed RGB-D images of a static scene our model constructs a 3D scene feature representation by neural lifting and registering features extracted from each frame. 3D scene representations are oriented and projected to sampled viewpoints to predict the corresponding target views, as a form of self-supervision for the weights of the encoder and decoder. Additionally, our model detects objects in the inferred 3D scene representation and uses them to improve the 3D visual feature representation by iteratively inferring 3D part correspondences across objects detected in different scenes, and using metric learning to supervise the feature representation to reinforce the inferred correspondences (Section 2.1). Finally our model matches these 3D object feature tensors against a set of 3D prototypes by searching over 3D rotations (Section 2.2). Our model optimizes over weights of the encoder, decoder, 3D detector module and prototypes. We detail each module in their respective section and present the learning of the model in Section 3.

2.1. 3D feature learning using view prediction and part correspondence mining (3DMining)

Geometry-aware Inverse Graphics Networks (GIGNs) introduced in (11; 4) learn to "lift" RGB-D images of static scenes to 3D scene feature maps differentiably while optimizing end-to-end for a downstream task. We will denote the 3D scene feature map inferred from the tth input image as $\mathbf{M}_t \in \mathbb{R}^{w \times h \times d \times c}$ where t, w, h, d, c denote the view number, width, height, depth and number of channels, respectively.

Upon training, GIGNs map RGB-D image sequences to complete 3D feature maps of the scene they depict, i.e., the model learns to *imagine* the missing or occluded information; we denote this 2D-to-3D mapping as $\mathbf{M}_t = E_{\text{GIGN}}(I_1..I_t), \mathbf{M}_t \in \mathbb{R}^{w \times h \times d \times c}$. For notation simplicity, we will drop the subscript from $\mathbf{M}$. For further details on GIGNs, please see (11; 4).

We train the encoder and decoder of our architecture, depicted in Figure 2, by predicting views, following work of (4). The scene feature map $\mathbf{M}$ is oriented to a sampled query viewpoint v_{qr} and decoded to an RGB image and a depth map, and compared with the ground truth RGB (I_{qr}) and depth map (d_{qr}) respectively: $\mathcal{L}^{pred} = \|D^{RGB}_{\text{GIGN}}(\mathbf{M}, v_{qr}) - I_{qr}\|_1 + \|D^{depth}_{\text{GIGN}}(\mathbf{M}, v_{qr}) - d_{qr}\|_1$. Our model also iterates between inferring part based correspondence between objects in different scenes and updating its weights to emulate the inferred correspondences. We empirically found that we consistently obtained better 3D feature representations by additionally considering learning from cross-scene part-based correspondences along with view prediction objective. We adopt the correspondence mining method of ArtMiner (9) to operate in 3D as opposed to 2D.

2.2. Pose-equivariant 3D object quantization

We define a set of 3D prototypes, $\mathbf{e}_k \in \mathbb{R}^{w_p \times h_p \times d_p \times c}, k = 1 \cdots K$ that compress objects in the scene in a view and scale equivariant manner: similar object instances that vary in scale and pose are mapped to the same prototype. We match detected object 3D feature tensors to prototypes using a rotation sensitive feature matching. Specifically, we exhaustively search across rotations R in a parallel manner, considering increments of $10°$ across the vertical axis:
(i) $(z^o_{id}, z^o_R) = \arg\min_{k,\text{R}} \|\mathbf{e}_k - \text{Rotate}(\mathbf{M}^o, \text{R})\|, \forall o \in \{1, ...|\mathcal{O}|\}$.

Having assigned objects to oriented prototypes, we update our prototypes to minimize their Euclidean distance to the assigned oriented and scaled object tensors:

$$\text{(ii) } \mathcal{L}^{3DQ}(e) = \sum_{k=0}^{K} \sum_{\{o|z^o_{id}=k\}} \|\mathbf{e}_k - \text{Rotate}(\mathbf{M}^o, z^o_R)\|_2$$

Equations i and ii can be seen as expectation maximization steps for iterating between assignment of object instances to prototypes and prototype updates.

3. Learning - optimization

We optimize our model with a combination of end-to-end backpropagation and EM iterations. We then iterate over the following steps: (i) 3D object detection. This generates a set of 3D object proposals. (ii) Prototype updating assigns detected object instances to prototypes and updates the prototypes through end-to-end backpropagation of the clustering loss to $\mathbf{e}$ as mentioned in Section 2.2. (iii) Re-Labelling of 3D proposals using 3D center-surround saliency and matching to prototypes. We keep the 3D object proposals that have a good matching score against the prototypes. We further filter out the 3D proposals whose 3D center-surround feature match score is below a threshold. We then update the 3D object detector module to emulate such labels through standard gradient based supervised learning.

4. Experiments

Our experiments aim to answer the following questions: **(i)** How do 3DQ-Nets compare against their 2D equivalent, 2DQ-Nets? **(ii)** How do 3D features learnt by our model compare to 2D features supervised by Imagenet classification in task of Few Shot Learning? **(iii)** How much does visual compression help 3D object detection?

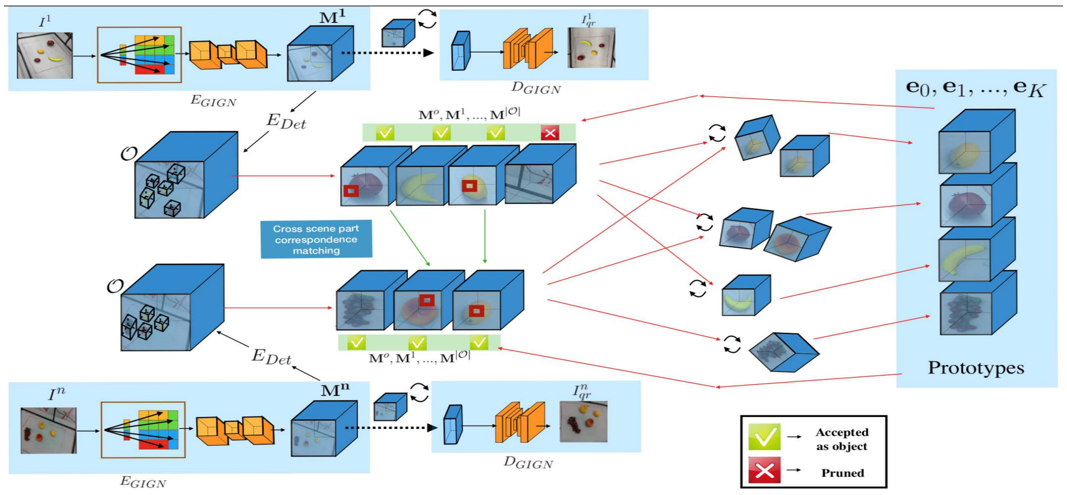
I^1
$\mathbf{M^1}$
I^1_{qr}
E_{GIGN}
D_{GIGN}
E_{Det}
$\mathcal{O}$
$\mathbf{M}^o, \mathbf{M}^1, ..., \mathbf{M}^{|\mathcal{O}|}$
$\mathbf{e}_0, \mathbf{e}_1, ..., \mathbf{e}_K$
Cross scene part correspondence matching
Prototypes
I^n
$\mathbf{M^n}$
I^n_{qr}
Accepted as object
Pruned

Figure 2: Architecture for **object-quantized 3D mapping networks (3DQ-Nets)**.

4.1. Datasets

We show our model's performance on various datasets. For Our **CLEVR veggie** dataset, we build upon the CLEVR Blender simulator (5) and add 17 vegetable object models bought from Turbosquid. Each scene is recorded by 28 RGB-D cameras. Our **CARLA** dataset uses the 26 vehicle classes available in the CARLA simulator of (1) with 17 RGB-D cameras placed within a view sphere. **BigBIRD** (**10**) contains multiview shots for 125 different objects rotating on a table. We assign the objects to 41 different object category labels, combining similar objects into a single class. Our **Real world desk scenes dataset** training setup consists of 8 Microsoft Kinect Azure sensors surrounding the table to capture multiview RGB-D data. During test time, we move a single Kinect sensor around the scene.

4.2. Visual feature compressibility

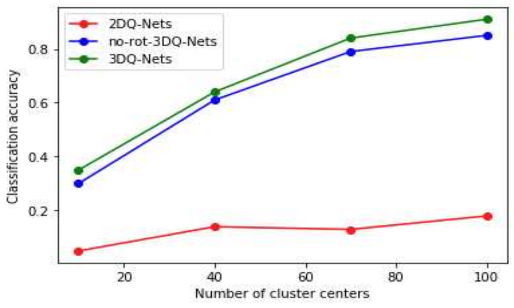

Figure 3: **Classification accuracy** with varying length of prototype dictionary for the CARLA dataset.

In this section, we measure classification accuracy across varying length of the prototype dictionary, where accuracy is measured by assigning category labels to prototypes and letting such labels propagate through instance-to-prototype assignments. Higher accuracy for the same number of prototypes indicate increased compressibility of visual information. We compare our model that quantizes 3D object instances into pose-equivariant 3D prototypes (3DQ-Nets) against the following formulations: (i) 2DQ-Nets, a 2D CNN model that optimizes an autoencoding objective, and quantizes detected 2D boxes into a discrete set of 2D prototypes using 2D rotation search, and (ii) no-rot-3DQ-Nets, a model similar to ours that assigns instances to 3D prototypes without rotation search, rather, minimizing direct Frobenius norm of the tensor-difference between a 3D instance and a 3D prototype.

We show these comparisons in Table 1 and in Figure 3. We see from Figure 3 that compressing visual features in the 3D feature space yields significantly higher classification accuracy than applying such compression in the 2D feature space, which is expected, since objects appear very different when projected in 2D from different viewpoints. As a result, 2D models usually require multiple prototypes to handle such appearance variation even of a single object instance. We further show that adding rotation search in 3D

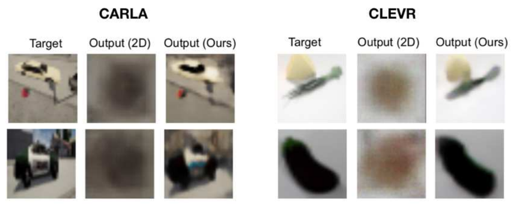

Figure 4: Scene reconstruction of target RGB (column 1) using the learnt prototypes from the 2DQ-Net (column 2) and 3DQ-Net models (column 3).

enforces objects with similar appearance but with different poses to be clustered together. In Figure 4 we also show the reconstruction of a scene after replacing the object instance in the scene with the learnt prototype. We see that the 2D prototype ends up learning the mean representation of objects in different poses, which appears as a circular blur.

Datasets.	2DQ-Nets	no-rot-3DQ-Nets	3DQ-Nets
CLEVR	0.23	0.73	**0.77**
BigBIRD	0.28	0.81	**0.83**

Table 1: **Classification accuracy** with dictionary size of 50 prototypes on CLEVR and BigBIRD datasets.

4.3. 3D object detection supervised by compression

In this section, we show how the mAP of our 3D detector improves over time when supervised by visual compression and 3D center-surround. We consider two initialization schemes for our 3D detector: i) we train our detector with a set of gt 3D boxes (3D-pretrain), ii) we train our detector by triangulating 2D object proposals from our 2D objectness detector (2Dtriang-pretrain). We show results in Table 2. From the results, we see our detector can improve its detection by a large margin after finetuning its weights by learning on the positive examples suggested by the learned object prototypes and negatives examples from the center surround check.

Datasets	3D pre-train	3DQ-Nets (*final*)	2D triang-pretrain	3DQ-Nets (*final*)
CARLA	0.41	**0.59**	0.32	**0.41**
CLEVR	0.42	**0.61**	0.37	**0.52**

Table 2: **Initial and final 3D detection meanAP** at IoU=0.5 using detected 2D proposal triangulation versus ground-truth 3D bounding boxes in a training set.

4.4. Few-shot learning

In this section, we evaluate the ability of 3DQ-Nets to learn from a small set of annotations. We use gt boxes in our training process inorder to isolate the contribution of the feature quality. For our model, we use the annotated instances from the training set to assign labels to the prototypes by voting. In Table 3, we compare 3DQ-Nets against two 2D baseline models pretrained on the ImageNet classification task: (i) Finetuning ResNet-18 image classifier with our train set and using the top layer to assign labels to 2D object crops from the test set (ResNetClass), (ii) using the top average pool layer activations of ResNet-18 image classifier and retrieving the nearest labelled train set instance for each test set instance (ResNetRet).

We show the results in Table 3. Our model outperforms both baselines that either use or learn from ResNet features which are trained on huge number of image labels.

Datasets.	ResNetRet	ResNetClass	3DQ-Nets
CARLA	0.27	0.58	**0.71**
CLEVR	**0.8**	0.72	0.75
BigBIRD	0.40	0.67	**0.82**

Table 3: **Object labelling accuracy** under a few-shot learning setting for the proposed 3DQ-Nets and two baselines, ResNetRet and ResNetClass

References

[1] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun. CARLA: An open urban driving simulator. In *CORL*, pages 1–16, 2017. 3

[2] B. T. F, K. Talia, and A. G. A. A review of visual memory capacity: Beyond individual items and toward structured representations. 11:4–4, 2011. 1

[3] A. Graves, G. Wayne, and I. Danihelka. Neural turing machines. *CoRR*, abs/1410.5401, 2014. 1

[4] A. W. Harley, F. Li, S. K. Lakshmikanth, X. Zhou, H.-Y. F. Tung, and K. Fragkiadaki. Embodied view-contrastive 3d feature learning. *arXiv*, 2019. 2

[5] J. Johnson, B. Hariharan, L. van der Maaten, L. Fei-Fei, C. Lawrence Zitnick, and R. Girshick. CLEVR: A diagnostic dataset for compositional language and elementary visual reasoning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2901–2910, 2017. 3

[6] D. Kumaran, J. Summerfield, D. Hassabis, and E. Maguire. Tracking the emergence of conceptual knowledge during human decision making. *Neuron*, 63:889–901, 09 2009. 1

[7] Q. V. Le, M. Ranzato, R. Monga, M. Devin, K. Chen, G. S. Corrado, J. Dean, and A. Y. Ng. Building high-level features using large scale unsupervised learning. In *Proceedings of the 29th International Coference on International Conference on Machine Learning*, ICML'12, page 507–514, Madison, WI, USA, 2012. Omnipress. 1

[8] M. Palmiero, R. Di Matteo, and M. Belardinelli. The representation of conceptual knowledge: Visual, auditory, and olfactory imagery compared with semantic processing. *Cognitive processing*, 15:143–157, 04 2014. 1

[9] X. Shen, A. A. Efros, and M. Aubry. Discovering visual patterns in art collections with spatially-consistent feature learning. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 2

[10] A. Singh, J. Sha, K. S. Narayan, T. Achim, and P. Abbeel. Bigbird: A large-scale 3d database of object instances. *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 509–516, 2014. 3

[11] H.-Y. F. Tung, R. Cheng, and K. Fragkiadaki. Learning spatial common sense with geometry-aware recurrent networks. In *CVPR*, 2019. 2