Variable-Viewpoint Representations for Object Recognition

By

Tengyu Ma

Dissertation

Submitted to the Faculty of the

Graduate School of Vanderbilt University

in partial fulfillment of the requirements

for the degree of

DOCTOR OF PHILOSOPHY

in

Computer Science

December 12, 2020

Nashville, Tennessee

Approved:

Maithilee Kunda, Ph.D.

Richard Alan Peters, Ph.D.

Ipek Oguz, Ph.D.

Matthew Berger, Ph.D.

Hao Su, Ph.D.

To my wife, Xiaojia.

You are my heart, my world, my soulmate, and my love forever.

To my parents, Guiwen and Xiaoling.

You let me grow up in a family of joyfulness and happiness.

To my kitty, Nana.

You tell me how easy love, smile, and trust could be.

ACKNOWLEDGMENTS

Thank you, Dr. Maithilee Kunda.

You taught me how to do scientific thinking and great research. Much beyond this, you taught me how to face challenges and adversity. “Fail early, fail often” is what you often told me, and this has always become the source of power for me while I was looking for hope in the darkness. If what Steve Jobs said, “stay hungry, stay foolish” is why I’m always looking for the dream; what you said is why I can always insist on the way of looking for my dream.

Thank you, Dr. Xiaohan Wang.

You introduced me to this fantastic Lab and you have left so many things to me, the foundation and direction of science, the characters of calm and humble, and much more. You are just like the rain, nourishing the ground silently but deeply.

Thank you, AIVAS Lab members.

We are a young Lab, and this is why we are full of energies and potentials. I was so lucky I could live and study with all of you. Let’s ignite our passion and do things big.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Figure Page

Chapter 1

Introduction

1.1 Introduction

Object recognition is an essential task for computer vision in which substantial computational resources, large-scale datasets, and deep learning models all together have made remarkable achievements. However, although both deep learning models and humans can recognize objects, the training and testing representations are quite different. For humans, the equivalent of "large-scale training datasets" are accumulated while they are growing up.In particular, humans first get trained on a small number of objects with many viewpoints per object during their infancy, and these early years of visual training build the foundation to later learn new objects in almost a one-shot way [4].

Furthermore, because the world is 3D while the human visual system involves 2D image projections to a large extent, humans can receive different 2D visual stimuli by observing the same objects from various viewpoints along with the haptic feedback about the 3D structure of objects and observations about naive physics, to name a few such properties. Among all these interactive properties of objects in the real world, we will focus on the process of the 3D to 2D projections of object information. Projecting a single physical object from 3D to 2D have so many possibilities than just a single image of the canonical view of the object.For humans, the "training" dataset established through this projecting procedure has some intrinsic properties different from common large-scale datasets such as ImageNet [16] and Coco [17]. This will be noted as viewpoint-rich 2D representations in this dissertation, which may come from the change of perspectives, the rotation or translation of the object, or a mix of transformations.

In machine learning, researchers have designed many data augmentation techniques for 2D images, including geometry transformation (flipping/cropping/rotating), color adjust-

ment (brightness/hue/contrast), kernel filters, or modeling image properties using Generative Adversarial Networks (GANs) [18]. These are all post-data-collection augmentations such that the images have been fixed and we then tweak the images. Data augmentations in these approaches may not bring authentic viewpoint-rich representations, where "authentic" here means different representations of the same object that you could plausibly observe in the real world.

For example, cropping an image will lose resolution but bringing an object closer in the real world will not. Also, imagining the new perspective of an object by GAN will be a probabilistic guess but rotating an object to a new viewpoint in the real world will be the ground truth.

1.2 Research Problem

Among many differences between the training environment for humans and deep learning models, I mainly focus on the following three key properties in my research:

1. **Egocentric.** Humans mostly get visual inputs from the first-person perspective of this world, and especially for the learning of object recognition of infants during their 12-24 month life [4], many visual inputs are objects manipulated by hands from the egocentric view. However, many large-scale or multi-view datasets don't have the egocentric property, and many egocentric datasets may not be designed for object recognition or in a relatively small-scale.

2. **Instance-limited.** The instances that the infants can get in touch with are typically very common objects in their daily life. These limited instances become the largest portion of their learning resources for object recognition. In contrast, large-scale datasets as web-crawling images may not include instance-level information, beyond just a broad category label, though an increasing number of datasets do try to include this instance information.

3. **Viewpoint-rich.** Although infants may just have a few instances, they will observe these objects from extensive viewpoints. Such a rich experience of the same object lays the foundations for our robust object recognition ability in the early life of humans.

These properties serve as the basis for the research problem of this dissertation.

- **In order to understand the developmental view of object recognition, we should be able to quantify how key properties of human visual experiences, which are egocentric, instance-limited, and viewpoint-rich, will impact learning.**
- **In order to quantify how these properties impact learning, we need datasets and representations that can support systematic experiments over those properties.**

Thus, inspired by findings from developmental psychology and considering current practices in deep learning [19], I proposed a dataset called Toybox, benchmarked the object recognition performance of humans and deep learning models on the Toybox dataset, and interpreted what the latent neurons of neural networks might have learned given the spatiotemporal consistency of the Toybox dataset. These three works become the first three contributions of this dissertation:

1. an egocentric dataset of structural transformations, Toybox [20];
2. an object recognition benchmark to investigate differences between human and computer vision on the Toybox dataset, including how differences in input representations can affect object recognition performance.
3. novel methods for using Fast Fourier Transform (FFT) [21], combined with the spatiotemporal properties of the Toybox dataset, to analyze the frequency correlations between the input and the latent space to interpret the representations at the last-hidden layer in neural networks.

In addition, I designed and implemented a novel projecting framework to generate rich representations from a 3D object to the 2D space, and demonstrated how these rich representations could achieve strong performance on a 3D object recognition task. This work represents the last contribution of this dissertation:

4. the Variable-Viewpoint (V^2) representational framework [22] of systematically projecting a 3D object into 2D space, and the investigation of how the V^2 representations could be helpful for the 3D object recognition task.

Figure 1.1 shows where the four contributions of this dissertation fit together within a general procedure of object recognition. As marked by the letters A, B, C, and D, A is the structural-egocentric dataset we collected, the Toybox dataset; B is the object recognition performance benchmark for deep learning models and humans on Toybox; C is the temporal consistency representation interpretation using Toybox; and D is the projecting framework for variable-viewpoint (V^2) representations;

1.3 Motivation

The computer vision field has changed significantly because of the achievements of deep learning [23], convolutional neural networks [24, 25], and large-scale datasets [16, 17]. Object recognition as an important sub-field of computer vision has also been improved substantially, and many state-of-the-art network architectures keep pushing the ImageNet benchmark higher [26].

One interesting finding from research on human cognition and learning is that, in spite of the boosted performance of object recognition through training on large scale datasets, from the developmental research perspective, humans train their visual ability in a much different or even reversed way. For example, Smith *et al.* found that [4] infants learned their early object categories by highly repeated exposures to a few objects they played with every day, such as their favorite toys. Then, while they are growing up, they are able to

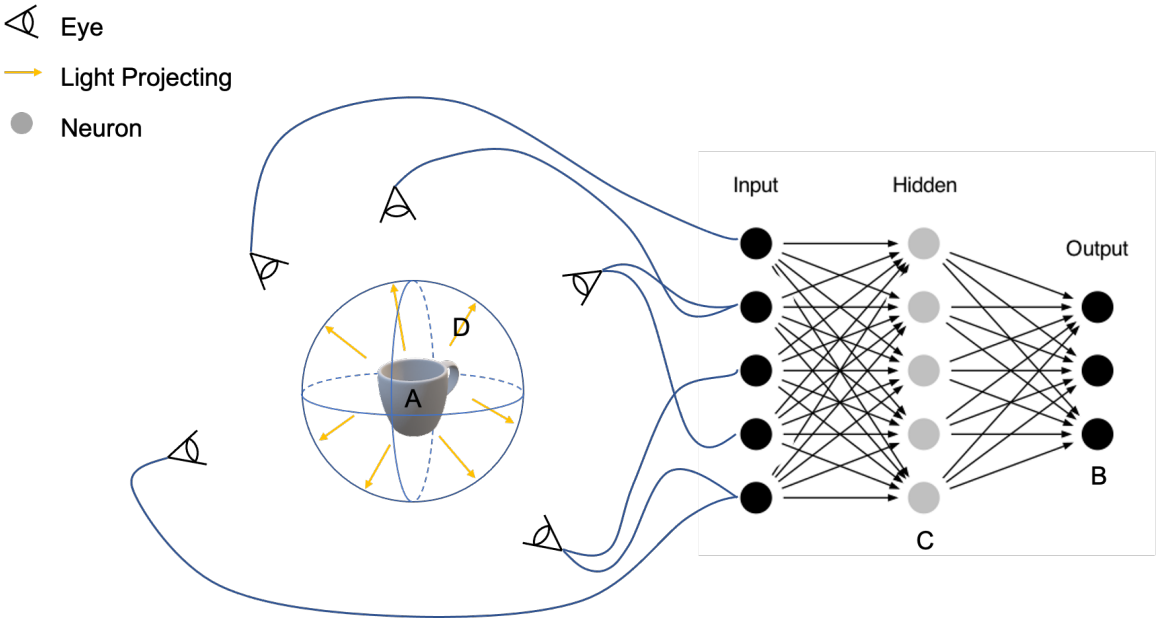


Figure 1.1: A conceptual diagram to demonstrate different stages in a general procedure of object recognition. This figure also provides a holistic view of this dissertation, in terms of how the contributions fit together: A) the structural-egocentric Toybox dataset, B) the object recognition benchmark for deep learning models and humans on the Toybox dataset, C) the temporal consistency representation interpretation using the Toybox dataset, and D) the Variable-Viewpoint (V^2) projecting framework and its usage for 3D object recognition.

recognize more and more new objects with just a few exposures of new objects. Similarly, in a diary study of her infant son, Mervis *et al.* [27] found that the initial concept of "duck" was likely only from a few ducks around their house, such a plush duck toy or a plastic duck rattle, that are extensively observed. Such a learning pattern for the human visual system is much different from the large-scale dataset training that most current computer vision models use. The early learning of object recognition for humans stems from a few objects with very dense viewpoint observations which then become the visual foundations to enable future learning of new objects in a one-shot or few-shot way.

One could imagine that babies have a very limited chance to have a large-scale dataset of real objects but instead have access to just some toys they may play around with every day. However, babies could benefit a lot from just a few toys to build their visual recognition system robustly. While time goes by, babies will be exposed to more objects, but the

interaction time per object will keep decreasing as well because of the larger moving region and the faster learning ability for the new objects, i.e., the learning patterns are changing while they are growing up. According to the paper [4], around two years of age, a child's visual system can learn a new object in an almost instant or one-shot way, i.e., by just taking a quick look, the new object can be well learned.

We want to better investigate the experience of how infants are learning their visual abilities by interacting with the world, and this was the motivation for collecting our dataset, Toybox, and for proposed the Variable-Viewpoint (V^2) representational framework. In particular, we observe that infants essentially produce their own data augmentations by thoroughly observing a few objects they have in hand. The Toybox dataset is trying to model this procedure by recording rich representations of the same object from various viewpoints, and the V^2 framework is trying to model this procedure by bringing data augmentation into the projecting stage from 3D objects to 2D images.

The data augmentation that happens through all of these 3D-to-2D conversions has infinite possibilities because of the continuity of the real world. Every viewpoint and every time you look at the same object, you may not get the exact same projection. Toybox and V^2 each model a subset of all these representation possibilities. And, from the perspective of scientific experimentation, both Toybox and V^2 are designed so that researchers can have systematic control over dataset and representation parameters. Furthermore, the representations from Toybox and V^2 are deep-learning friendly to help bridge experiments and findings between infants' learning experiences and deep learning models.

1.4 Approaches

To investigate object recognition training and testing in developmentally relevant ways, this dissertation is composed of four key components of knowledge representations for computational object recognition.

1.4.1 The toybox dataset

Toybox is a dataset with both multi-view and egocentric properties inspired by how infants manipulate and observe toys in their hands. We collected Toybox by recording egocentric videos while a participant is systematically manipulating an object by hand following well-defined instructions. Before Toybox, many multi-view datasets or egocentric datasets have been published, but few contain both properties, and many egocentric datasets were not designed for object recognition.

Toybox is a video dataset, which contains viewpoint-rich representations of 360 handheld object instances across 12 categories. A single object in Toybox will have 12 different transformations such as translations and rotations resulting in $\sim$6,600 images per object. As a part of the contribution of this dissertation, the Toybox dataset has been published online at https://aivaslab.github.io/toybox/, and some other research groups are using Toybox for their studies [28, 29].

1.4.2 The toybox benchmark

In the Toybox benchmark, we first did a full resolution (1920×1080) object recognition test of rotation videos for human performance by crowdsourcing recognition tasks on MTurk [30], and then we used a different, deep-learnin-friendly resolution (128×128) of rotation videos to compare performance between humans and current mainstream neural networks such as ResNet [31] and Inception-v3 [32], both with and without pretraining.

This work reveals where the differences are between human and computer vision models for performance on the Toybox dataset, and establishes new benchmark goals for what we should expect our computer vision models to eventually achieve on Toybox.

1.4.3 The toybox interpretation

Because of the spatiotemporal consistency in the Toybox dataset, we found that we could utilize this property to interpret representations learned by neural networks using conventional spatiotemporal analysis tools.

In particular, we showed that if the input neurons are activated in a continuous manner, such as using the continuous frames in Toybox rotation videos, the neurons in the latent space would be activated in a continuous manner accordingly. If we converted the continuous signals of the latent neurons into the frequency domain, we can observe how different neurons would have different frequency signatures to reflect how they may respond to the corresponding inputs differently.

Therefore, when we are showing inputs from different categories to the neural networks, we could identify which latent neurons having stronger responses than others for a specific category, and these latent neurons could be the main contributors to the recognition of this category. We further examined this category-level representation interpretation of latent neurons by pruning these identified neurons out to observe how the output neuron activations will change as a result.

1.4.4 The V^2 representation

When a 3D object is projected into the 2D space, there will be unlimited possibilities of projections, for example, from different perspectives, different distances, different resolutions, etc. The V^2 framework models several aspects of this projecting procedure in a systematic way. Supported by the V^2 framework, we then can leverage, for object recognition, the potential rich and varied representations generated through the projecting process.

The V^2 frame provides a continuum of viewpoint sampling around a single object in which we could control the distribution of these viewpoints and how the receptive field of each viewpoint looks like. By tuning the parameters in the V^2 framework, we could

generated a lot of different V^2 representations of just a single object. The V^2 framework is publicly available now at https://github.com/aivaslab/v2/ as a part of the contribution of this disseration.

One important contribution of V^2 is that it unifies two current approaches, the multi-view [6] representations and the spherical representations [10] to 3D object recognition. In terms of how many viewpoints to observe of an object and how many pixels allocated per viewpoint, the multi-view end will be using a few viewpoints but each view will have a higher resolution, and the spherical end will be using extensive viewpoints but each view will only have one pixel. Such a changing of the number of viewpoints and the pixels per view is under the scope of the V^2 representations, and we identified that in the V^2 space these two representations are just two extremes where a broad region of intermediate representations exist between them to be explored.

We conducted a series of three computational experiments using the V^2 framework, in which we evaluated the expressive power of the V^2 representations on the ModelNet40 [33] dataset for the 3D object recognition task. In particular, we tested various parameter setups for the V^2 representations on different deep learning architectures including MVCNN [6] designed for multi-view representations, S2CNN [10] designed for spherical representations, and classic networks such as ResNet [31]. We also proposed Multi-Representation Convolutional Neural Networks (MRCNN) to fuse several layers of V^2 representations together which showed strong performance to the 3D object recognition task on ModelNet40 with no traditional data augmentation and no hyperparameter.

1.5 Organization

Chapter 1, is the introduction of this dissertation including introduction, research problem, motivation, approaches, and organization.

Chapter 2 contains related work about computer vision datasets in terms of different viewpoints, object representations, object recognition architectures, data augmentation

methods, and neural network interpretation methods.

Chapter 3 introduces the Toybox dataset, e.g., how Toybox was designed, collected, and annotated, and also presents some exploratory experiments and analysis on Toybox.

Chapter 4 explains the object recognition benchmark design for human and computer vision on Toybox, and then presents and analyzes the benchmark results.

Chapter 5 illustrates how we could use the FFT algorithm to analyze the temporal consistency propagating through neural networks to interpret the representations of latent neurons, and further evaluates this kind of interpretation through ablation studies, i.e., by examining neuron pruning effects.

Chapter 6 demonstrates how the variable-viewpoint V^2 representational framework is constructed, and evaluates how different deep learning architectures perform on various V^2 representations, including a new Multi-Representation Convolutional Neural Network (MRCNN) architecture.

Chapter 7 is the conclusion of this dissertation, including what has been achieved, how this dissertation may benefit our research community, and what might be promising in the future based on findings and results.

Chapter 2

Related Work

2.1 Introduction

In the introduction chapter 1, we have mentioned four main contributions of this dissertation, the Toybox dataset, the Toybox benchmark, Toybox-based neural network interpretation, and the V^2 representational framework. Thus, this chapter will go through related work on computer vision datasets, neural network interpretation methods, and image data augmentation, and then discuss how our work could add additional value to these existing research areas of the computer vision community.

2.2 Datasets Overview

Numerous computer vision datasets have been published for supporting all kinds of computer vision research. If we categorize these datasets by which data formats they are in, we could have RGB images, RBG-D images, videos, 3D meshes, point clouds, etc. Furthermore, inside each data format, we can consider variations in the research purpose of the dataset, for example, object-centered RGB datasets such as ImageNet [34] for object classification, or fine-grained objects-in-context RGB datasets such as COCO [17] for object detection and segmentation.

Now, if we focus on how vision datasets are built using different projecting methods from 3D space to 2D space, we can find another way to categorize them in terms of the viewpoint distributions they provide, i.e., from which viewpoint dataset images or videos got captured. For 3D datasets in the formats of 3D meshes or point clouds, even though the dataset itself can support arbitrary view setups, there are many available choices for building 2D representations from the 3D model to perform visual tasks such as object recognition, and these choices can affect eventual task performance just as much as the

choice of a particular system architecture. Thus, in this section, we will review several existing datasets organized by how the viewpoints of the objects are distributed within each, and then explain how our dataset, Toybox, has joint properties across the classes of egocentric-view datasets and multi-view datasets.

2.2.1 Web crawling viewpoint 2D datasets

Web crawling is one of the traditional methods to build large scale 2D datasets; for example, ImageNet [16] has reigned as the benchmark dataset for many computer vision tasks in recent decades. These datasets crawled online have some particular viewpoint distributions, because most of the images are taken from some common photographers' viewpoints such that the viewpoint variations mostly lie within a limited zone of azimuth and altitude [2] as shown in Figure 2.1.

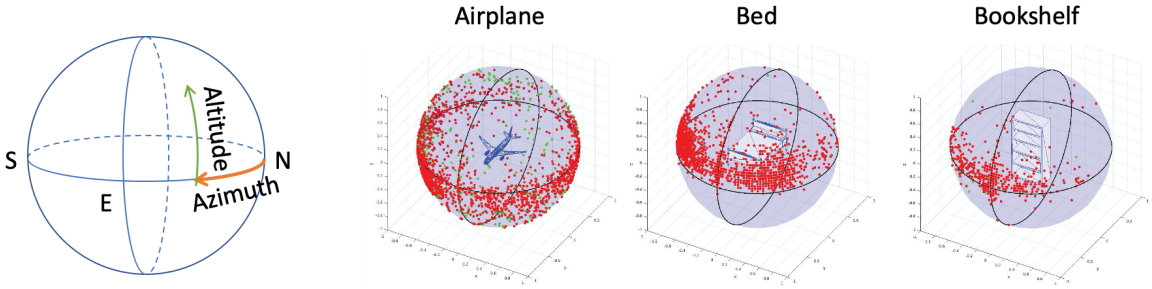


Figure 2.1: The leftmost sphere is a diagram of the direction of azimuth and altitude change, and the three spheres on the right are the viewpoint distribution for the categories Airplane, Bed, and Bookshelf in the ImageNet (The ImageNet viewpoint figures are from the paper ObjectNet3D [2] by Xiang *et al.* Notice that how the viewpoints are biased on the direction azimuth with a limited range of altitude.

Typically, these web crawling 2D datasets such as ImageNet by themselves do not have the viewpoint information, so viewpoint estimation needs to be done post hoc. One classic estimating procedure is to align 3D objects to the corresponding 2D images such that the projecting plane of the 3D objects and the images look similar to each other.

ObjectNet3D [2] is the result of one effort to provide viewpoint information for ImageNet images. To align the images by 3D objects, they collected related 3D models from

the Trimble 3D Warehouse [35] and ShapeNet [36]. Figure 2.1 shows some viewpoint distribution samples from their work.

Pascal3D [37] is another dataset trying to align 3D objects to 2D images, where the 2D images are mainly from the Pascal VOC 2012 challenges [38] plus some additional images from ImageNet, and the 3D objects are from the Google 3D Warehouse [39].

Similarly, IKEA [40] collected 2D IKEA furniture images from Flickr [41] and then collected 3D IKEA furniture models from the Google 3D Warehouse [39] to make the 3D-2D alignment. Pix3D [42] extended the IKEA dataset by crawling more IKEA 2D images online with the original 3D IKEA models for alignment, and they also built an additional 2D-3D alignment dataset by taking images of the object from multiple viewpoints locally and then using Structure Sensor [43] to scan the object to construct 3D shape from the exact same object.

2.2.2 Viewpoint-controlled 2D dataset

In addition to aligning 3D models to get post hoc viewpoint estimation for 2D images, another approach is to record viewpoint information along with the original picture-taking activity, either through recording instructions or viewpoint-controlled apparatuses.

ObjectNet [44] is an example of a viewpoint-instructed dataset that was build by collecting data on the crowdsourcing platform Amazon Mechanical Turk (MTurk) [30]. Using MTurk, the workers followed specific instructions to position the required objects and then take pictures from required views. By implementing such a bias control data collection procedure, though ObjectNet has 113 overlapped categories with ImageNet, the viewpoint distribution is very different, that is, the viewpoints are uniformly distributed all around the object, even including the bottom view.

Some other datasets have similar properties but did not use crowd-sourcing; the researchers will follow instructions themselves to collect images. For instance, 3D Object [45] captured 10 different everyday object categories from 8 different viewpoints, and

EPFL-GIMS08 [46] captured real cars during an auto show from 20 different viewpoints.

The images taken via viewpoint instructions can bring increased viewpoint diversity into the dataset but may inject some human "noise" as well, since holding cameras in hands cannot always be precise. Thus, another approach, using viewpoint-controlled apparatuses, can help researchers to reduce human variance and thus provide more accurate viewpoint control. Though the setups of the apparatuses might be different, the core components are typically a turntable to control the azimuth degree and a camera arm to control the altitude degree. Varying in total number of categories, total number of objects per category, and total number of viewpoints per object, these datasets include COIL-100 [47], SOIL-47 [48], NORB [49], ALOI [50], 3D Objects on Turntable [51], Large-Scale RGB-D [52], Big-BIRD [53], and iLab-20M [54]. More details about these datasets can be found in Table 3.1 in Chapter 3

2.2.3 Egocentric viewpoint 2D dataset

Besides the datasets with web-crawling viewpoint distributions and viewpoint-controlled viewpoint distributions, the egocentric view perspective is another important property that reflects how humans are sampling visual input from the world. In the area of egocentric datasets, most of them are recorded for specific activities in specific environments, e.g., making pizza in the kitchen, shopping in the grocery store, or walking from the metro station to the office. Because these egocentric datasets are generally geared towards research in life-logging areas, e.g., often to record actions rather than observe objects, the viewpoint distributions of objects may not be well managed.

Therefore, most of the egocentric datasets contain variations in scenes and motions but are not object oriented. Some egocentric datasets were collected for indoor or outdoor activities, for example:

- EDUB [55] for shopping, working, etc.;
- GTEA GAZE [56] for making coffee, tea, etc., in the kitchen;

- GTEA GAZE+ [56] for making pizza, pasta, etc., in the kitchen;
- First Person Social [57] for traveling at Disney World such as walking, sitting, etc.
- CMU Multi-Modal Activity [58] for scrambling eggs, making pizza, etc., in the kitchen;
- Huji EgoSeg [59] for sitting, standing, walking, in different scenes;
- UT Ego [60] for daily activities such as watching TV, grabbing a dish, or fetching the milk, etc.;
- VINST [61] for working from a metro station to an office;
- ADL [62] for indoor activities such as combing hair, make up, drinking water, etc.;
- LENA [63] for watching videos, walking straight/up/down, drinking, etc.;
- COGNITO [64] for industry scenarios such as taking nail/screw, picking screwdriver, putting down hammer, etc.;
- Bristol [65] for opening a door, making coffee, operating a gym machine, etc.

Some egocentric datasets were collected for human-related detection, for example:

- EGO-HPE [66] for face orientation;
- EGO-GROUP [66] for group composition;
- JPL First Person [67] for human interaction;
- Interactive Museum [68] for the gesture recognition in the museum;
- NUS First Person [69] for human-human or human-object interactions;
- CMU EDSH [70] for hand detection.

Some egocentric datasets were collected for scenes, for example:

- All I Have Seen [71] for kitchen, dinning room, tennis field, etc.;
- Michigan Milan Indoor [72] for indoor scene understanding such as floor and ceiling.

As mentioned earlier, few datasets among these egocentric datasets serve the purpose of object recognition. To the best of our knowledge, there are three datasets that are both egocentric and object-oriented: GTEA [73], Intel Egocentric [74], and CORe50 [75].

2.2.4 Viewpoint-controlled egocentric 2D dataset - Toybox

As we described in the previous sections, viewpoint-controlled datasets may not have egocentric properties, whereas the egocentric datasets may lack viewpoint control and very few of them were designed for object-oriented visual tasks.

However, inspired by how infants are developing visual abilities from this world by controlling the viewpoints of the toys in their hands to make egocentric observations, we thought that a dataset that has both viewpoint-control and egocentric properties will help us to investigate differences between computer vision models and humans for the object recognition task, especially from a developmental perspective. Therefore, we constructed a new dataset in this dissertation, Toybox [20], a dataset of egocentric visual object transformations, recorded by wearable glasses with systematic control of viewpoint distribution.

Among existing object-oriented egocentric datasets, GTEA has 16 objects with 1 object per category, Intel Egocentric has 42 objects with 1 object per category, and CORe50 has 50 objects with 5 objects per category. In comparison, Toybox has 360 objects with 30 objects per category, which significantly increases the scale of objects represented in this type of dataset. Serving as a stepping-stone dataset, Toybox possesses both egocentric and viewpoint control properties through systematic viewpoint rotations and translations around three main orthogonal axes in the 3D space. These transformations provide considerable egocentric viewpoint variance.

2.2.5 3D datasets

The 3D datasets described in this section are datasets stored in explicit 3D formats such as object meshes or point clouds. These datasets have a distinct advantage in viewpoint-related research because we can have perfect control over viewpoints by rendering any 2D representations from the original 3D objects. 3D datasets have been collected for many different fields, such as research on the human body [76, 77], scene understanding [78, 79,

80, 81, 82] and object recognition [33, 36].

Particularly for the 3D object recognition domain, ModelNet [33] and ShapeNet [36] are two common benchmark datasets in which objects are represented as meshes stored in .OFF or .OBJ. formats. SHREC17 [83] is a competition version of ShapeNet which is also commonly in use. Toys-200 [84] is a 3D toy dataset whose objects are similar to Toybox with the difference that they are synthesized and stored as mesh files.

2.3 Representations and Architectures

Representations and architectures in object recognition are typically associated with each other, that is, some representations are particularly designed for some architectures and vice versa.

Representations of 2D images are generally formulated as vector or raster images. In contrast, representations of 3D objects have a relatively large family. Range images (depth images) [85, 86] were used to store 3D information in early 3D research, and even nowadays, the depth channel still provides important information to describe the 3D object's shape. Polygon soup [87, 88], sweep-CSGF [89], and spline-based representations [90] are typically designed for other 3D tasks besides object recognition.

In the deep learning era, many architectures have been specially designed for certain 3D representations, for instance, Graph CNN on Mesh [91] for Polygon Mesh [92]; PointNet [93] for Point Cloud [94]; and Octree-based O-CNN [95] for Octree/Quadtree-based representation [96], etc.

Multi-view (multiple viewpoint images containing the whole object to represent the 3D object) and voxel (similar to pixel but stacking in the 3D space) have also been explored a lot recently using deep learning models. They have similar input representations as 2D pixel-based representations, so that many existing deep learning components such as 2D or 3D convolutional layers can be directly applied on them. For example, researchers have designed MVCNN [6], GVCNN [9], RotationNet [7], and OrthographicNet [97] for multi-

view representations. Orientation VoxelNet [98], BV-CNN [99], VoxNet [100], VRN Ensemble [101], and 3DShapeNets [33] have been designed for voxel-based representations. Some comparisons have been done to compare voxel and multi-view representations [102] regarding the 3D object recognition performance based on CNN architectures.

Spherical signals [103, 104] have been studied since the late 20th century and in year 2002, Vranic and Saupe [105] demonstrated the usage of the spherical signals as 3D shape descriptors [105] for 3D vision tasks. Because of the rotation-invariant property of spherical representations and the increasing usage of omnidirectional cameras in drones and autonomous cars, spherical deep learning models have been investigated more in recent years such as S2CNN [10], SPCNN [15], SPNet [12], and UGSCNN [11]. Sphere sampling [106] algorithms affect how spherical representations are composed, and these algorithms include Driscoll-Healy grid [103], McEwen-Wiaux [107], Geodesic [108], or Goldberg methods. The main difference among these algorithms is the viewpoint distribution on the sphere.

The Variable-Viewpoint (V^2) representations proposed in this dissertation are intended as a general framework to cover both the multi-view and spherical representations mentioned above. In the V^2 representing space, multi-view and spherical are just two extremes varying in the number of views and pixels per view. Our V^2 framework can help to build a platform for interesting comparisons among different representation-architecture combinations.

Also, we found a broad region of new representations in the "middle" of the V^2 representing space, and by utilizing these "middle" representations, we achieved strong performance on the 3D object recognition task on the ModelNet40 dataset.

2.4 Data Augmentations of 2D Images

Image data augmentation is a common technique to help with the performance of object recognition models. Typically the current methods to augment 2D images are post hoc, which means we don't have the original object information anymore but just a single picture

of it to be augmented.

Many approaches have been developed for post hoc image data augmentation, including some classic transformations such as affine transformations (rotation, cropping, reflection, flipping), elastic transformations (brightness, contrast, color distortion), and the application of kernel filters (Gaussian kernel), etc.

Some deep learning based data data augmentations have been proposed in recent years, especially taking advantage of the adversarial examples [109], neural style transfer [110, 111], and generative adversarial networks (GAN) [18]. The image augmentation by adversarial examples is to counter the adversarial attacks to the deep learning models, such that by training on these adversarial examples, the models could be more robust while facing malicious inputs. The image augmentation by neural style transfer could change the background of an image, adjust the color style, or merging images together. Researchers have been using the neural style transfer to augment their datasets [112, 113] to transition stimulated environments to the real-world. GAN can be used for different purposes of image data augmentation, for example, Bzowles *et al.* [114] shows that GANs reveal "additional" information in their CT dataset and Xinyue *et al.* [115] shows the usage of CycleGAN [116] to augment the face emotions for emotion classification.

We noticed that besides these post hoc augmentations, image data could be augmented significantly through projecting from 3D space to 2D space, i.e., viewpoint-varying augmentations. Such a viewpoint augmentation is a core part of training the human visual system, as found in studies in developmental psychology such as for face recognition [117] or object recognition [118].

While GAN based methods can generate how the other viewpoints of an object might look, another approach is to add this kind of augmentation as part of the original data collection itself. In other words, we can also proactively model this projecting procedure if we have the original object "in hand", i.e., either having the physical object or having the 3D object mesh stored so that we could observe it from any given angle.

Therefore, the dataset Toybox and the projecting framework V^2 proposed in this dissertation leverage the viewpoint-variance of 3D objects to investigate different kinds of viewpoint augmentation in the training of object recognition models, in addition to the post hoc augmentation methods typically used in current practice.

2.5 Neural Network Interpretation

The interpretability of deep learning has been capturing much attention recently in many domains such as speech recognition, natural language processing, and computer vision, as a part of the focus on Explainable Artificial Intelligence (XAI) [119]. In this section, we will mainly discuss deep neural network interpretation in the computer vision domain.

Activation Maximization (AM) is one of the common techniques to interpret hidden neurons. The core idea behind AM is to construct the input so that it can maximize the activation of a designated neuron [120], and such an activation maximized input image will help us reveal which kinds of visual input will be more likely to cause that neuron to fire.

Similar to our temporal consistency method in Chapter 5, the AM interpretation algorithm also tries to establish connections between the input and hidden space, where the main difference is that AM is trying to find a "snapshot" that can maximize the activation of a hidden neuron, whereas our method is trying to maximize the activation-changing of a hidden neuron by changing the inputs. Additionally, our temporal consistency method could take advantage of the input space continuity to reveal the features stored in hidden neurons along with the temporal axis that may not be captured in AM.

Another common technique to interpret hidden neurons is to visualize CNNs through some backpropagation strategies such as Deconvolutional Networks [121, 122], Guided Backpropagation [123], Class Activation Mapping (CAM) [124], or Gradient-weighted Class Activation Mapping (Grad-CAM) [125], to reveal which regions in the input space are more relevant for the deeper layers. Each approach will have some particular core

components to pass the information backward from the feature space to the input space, such as the deconvolutional layer or the guided backpropagation layer, which can reverse the convolutional operator while preserving the information from the feature space. These backpropagation approaches were mainly designed for the interpretation of an entire layer of neurons, and our temporal consistency was designed for the interpretation of a single neuron.

Occlusion sensitivity [121] and network dissection [126, 127] are both forms of system ablation, to interpret neural networks by trimming some parts of the neurons out, either in the input layer or hidden layers, in order to observe the activation changing in the hidden or output layers. Our temporal consistency method utilized a similar idea to further evaluate neuron interpretation by pruning the identified neurons. The hidden neurons whose activations are changing significantly while the inputs are changing are pruned, and we monitor how the activations of output neurons will change as a result.

Chapter 3

The Toybox Dataset of Egocentric Visual Object Transformations

3.1 The Research Question

How can we design a dataset to support computational object recognition research that contains the properties that we know are important in human development, i.e., egocentric, instance-limited, and viewpoint-rich?

This chapter will go through the methodology about how we collect, annotate, and organize the dataset; the overview of the Toybox dataset and how it compares to ImageNet; some exploratory experiments to show the properties of the Toybox dataset; and finally, the summary of the Toybox dataset.

3.2 Abstract

In object recognition research, many commonly used datasets (e.g., ImageNet and similar) contain relatively sparse distributions of object instances and views, e.g., one might see a thousand different pictures of a thousand different giraffes, mostly taken from a few conventionally photographed angles. These distributional properties constrain the types of computational experiments that can be conducted with such datasets and do not reflect naturalistic patterns of embodied visual experience.

As a contribution to the small (but growing) number of multi-view and egocentric datasets that have been created to bridge this gap, we introduced a new video dataset called Toybox that contains egocentric (i.e., first-person perspective) videos of common household objects and toys being manually manipulated to undergo structured transformations, such as rotation, translation, and zooming. In Toybox, the naturalistic embodied patterns of human visual experience can be captured by our egocentric recordings, and the diversity of viewpoint distribution of the objects can be captured through structured manual trans-

formations.

In this chapter, we will discuss the motivation for the Toybox dataset, illustrate how the Toybox videos were collected and annotated, describe how the structure of Toybox is different from ImageNet, and show some exploratory experiments of how the viewpoint variance in Toybox will affect object recognition performance.

3.3 Introduction

Many recent breakthroughs in computer vision, such as for the problem of visual object recognition, have been driven by the creation and use of large-scale labeled image datasets collected from the Internet, with ImageNet being a canonical example [34]. In a sense, these datasets have *coevolved* with the algorithms that learn so successfully from them—researchers continue to develop algorithms that work better and better on the types of data distributions found on the Internet, and also continue to collect ever larger and more densely labeled datasets using similar online data collection methods.

While the scientific advances and practical applications generated by these efforts have undoubtedly transformed the landscape of AI and machine learning, there are still many fundamental open research questions about how (and how well) intelligent agents can learn to recognize objects under very different types of training regimens. In addition, there are many problems going beyond recognition that may require richer visual experiences with objects than are typically gathered online, for instance, problems of visual commonsense reasoning or mental simulation. Two areas that will especially benefit from studying such questions are:

1. Research in AI + cognitive science to study the development of human object recognition. Research in developmental psychology has produced many interesting findings about the number and types of object instances children experience while learning category concepts. In a fascinating diary study of her infant son, Mervis [27] observed that his initial concept of "duck" was likely based on seeing live ducks at a nearby park, his

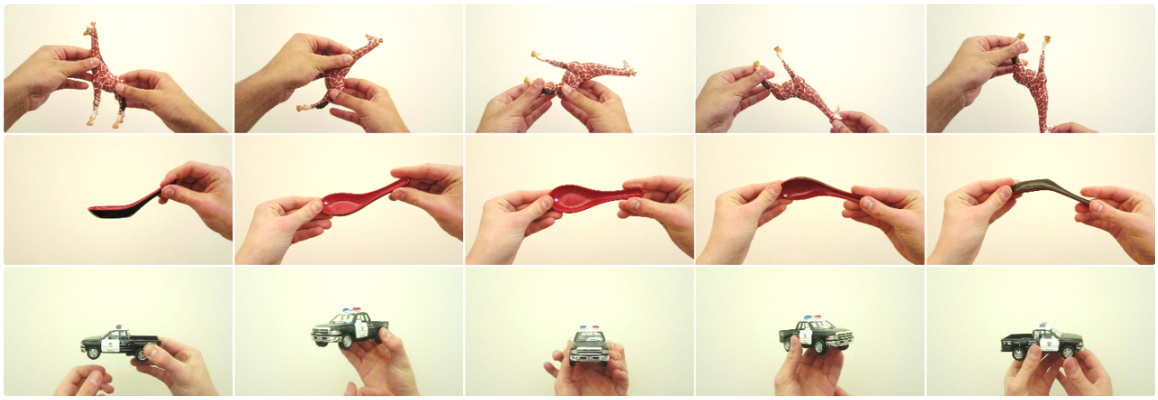


Figure 3.1: Toybox examples. (Color adjusted for PDF view.) The 1*st* row is a ryplus rotation sample from the animal super-category; The 2*nd* row is a rxplus rotation sample from the household super-category; The 3*rd* row is a rzplus rotation sample from the vehicle super-category. The details of different rotation terminologies can be found at Section 3.5.5.

plush duck toy, a plastic duck rattle, a rubber duck, and a handful of other odd, duck-themed household objects, toys, and picture books, and was gradually generalized and pruned over time.

Recent wearable-camera-based infant studies have found similar uneven distributions of experience across object instances, categories, and viewpoints, and these distributions also change with an infant's age [128, 129]. It is likely not the case that infants learn *despite* these irregularities, but rather that infant learning *leverages* these distributional properties, e.g., through bootstrapping, curriculum learning, etc. In other words, the infant's "learning algorithm" has coevolved alongside the neural, sensorimotor, cognitive, and sociocultural factors that combine to create an infant's visual world. Thus, AI research that explores interactions between training distributions and learning algorithms is poised to play a critical role in the cognitive science of visual learning, including for understanding the effects of technology (e.g., advent of print media, television, and now Internet) on child development.

2. Research in AI + robotics to advance the learning capabilities of embodied agents. Robots or other physically embodied agents (e.g., stationary cameras) may have access to online information but will likely also need to be able to learn from their immedi-

ate environment. For example, a household robot may need to learn about specific objects in a person's kitchen through a series of naturalistic visuomotor interactions that generate complex, unevenly distributed, and heavily occluded object views. What kinds of learning algorithms would be capable of learning from such data? One-shot learning, active recognition, and various forms of active learning are all relevant to this question, and will benefit from the creation of richer visual datasets.

The Toybox dataset. Here, we present a new dataset called Toybox (see example images in Figure 3.1) designed to facilitate computational experiments on visual object recognition and related vision problems, especially in the context of studying aspects of embodied (e.g., human or robot) visual object experience, including: (1) *continuously sampled views* of objects undergoing several different types of transformations, including rotation, translation, and zooming; (2) *an egocentric perspective* (i.e., handheld, first-person views), which means that objects are held in naturalistic grips and thus are always partially occluded; (3) *a range of everyday categories*, including household objects, animals, and vehicles; (4) *a diversity of object instances*, with 30 distinct physical objects representing each category.

For the kinds of studies that Toybox is designed to support, we will present one experiment in this chapter to studying the effects of instance and viewpoint diversity on recognition performance. In chapter 4 the Toybox benchmark, we will present more detailed experiments to compare different viewpoints, different zooming, different network architectures, etc., between deep object recognition models and human object recognition performance.

Then in chapter 5, we will use Toybox as an example dataset to investigate how hidden layer neurons in a trained neural network respond to continuous variations in object pose and to show how such a temporal continuity can be utilized to interpret hidden neuron representations.

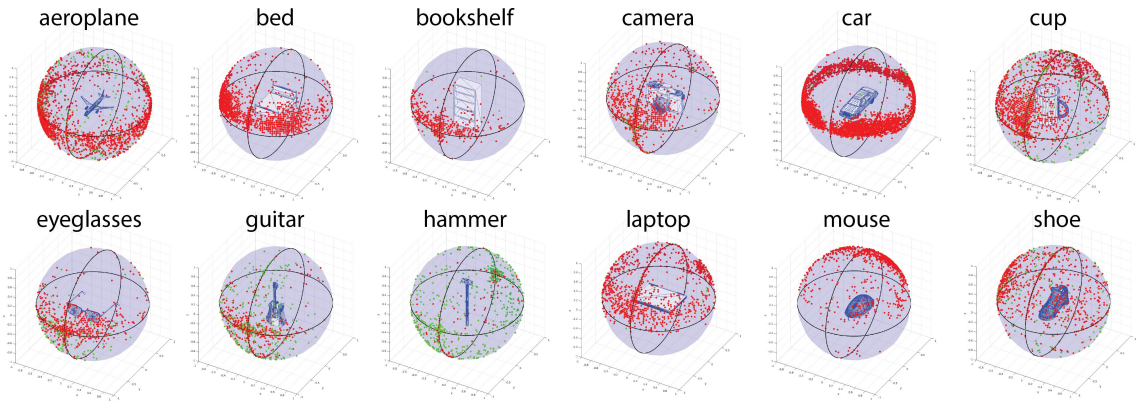


Figure 3.2: Viewpoint distributions in ImageNet. Viewpoint information is annotated by aligning 3D models. Figure is from the ObjectNet3D dataset [3].

3.4 Multi-View Object Recognition Datasets

As described above, ImageNet and many other widely used, "Google Image Search"-type vision datasets contain only one image per real-world object [34]. In addition, object viewpoints are constrained by the fact that most online images are created by adult humans using handheld camera devices [130].

Providing an interesting demonstration of this viewpoint bias, the ObjectNet3D dataset contains images from ImageNet annotated with the 3D pose of pictured objects [3]. As shown in Figure 3.2, different categories show very different viewpoint distributions, based on how people (adults) tend to encounter certain objects in everyday life.

Taking a complementary approach, several vision datasets have been created to provide multiple views of the same physical object, as reviewed in Table 3.1. These datasets are typically of two types: (1) *discrete but structured object viewpoints* are collected with the help of a turntable, e.g., the NORB, RGB-D, and iLab-20M datasets [131, 132, 133]; or (2) *continuous but unstructured* objects viewpoints are collected using human handheld object and/or camera manipulations, e.g., the Intel Egocentric and CORe50 datasets [134, 135].

We designed the Toybox dataset to capture the advantages of both approaches. Toybox contains egocentric-perspective videos of the camera-wearer holding and manipulat-

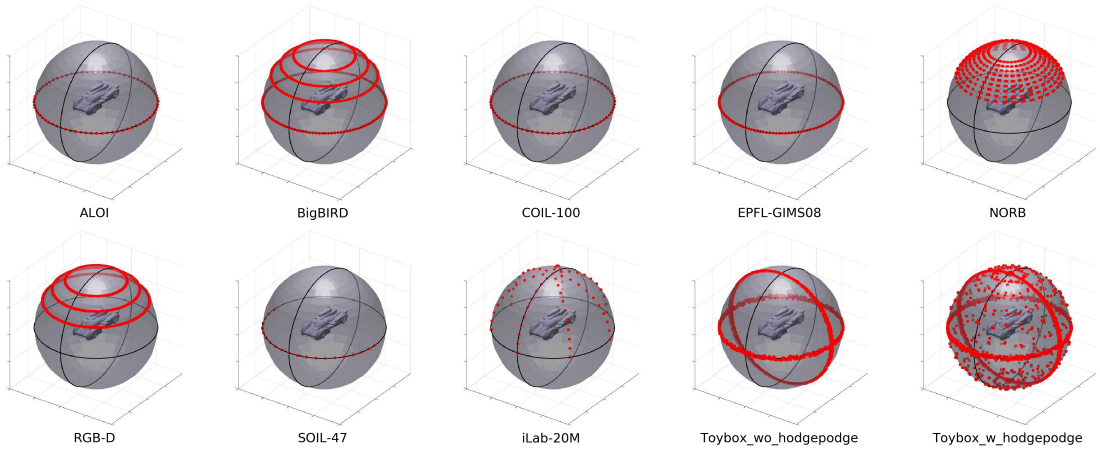


Figure 3.3: Comparison of viewpoint distributions across several multi-view datasets. We include two versions of the Toybox datasets, with or without the hodgepodge viewpoints, i.e., those viewpoints are distributed while the camera-wearer was freely manipulating the object. (For the hodgepodge version, Toybox off-axis viewpoints are estimations only and will vary from object to object.)

ing various objects in structured ways, e.g. completing two full revolutions of an object along a specified axis of rotation at a (roughly) constant speed, among other types of transformations. We also include a period of unstructured manipulation to capture a random assortment of off-axis views.

Figure 3.3 shows the viewpoint distributions provided by several existing multi-view datasets, as well as by the Toybox dataset. As can be seen in this figure, most of the existing multi-view datasets use turntables, and thus no bottom-facing views of objects are available. Toybox aims to provide a more complete set of object views. We do not know how valuable such views might (or might not) be for object recognition, but at least having the data available will enable experiments to study viewpoint distributions and visual learning in more detail.

Inspired by how ObjectNet3D [3] analyzed the viewpoint distributions of ImageNet as shown in Figure 3.2, we created a comparison of viewpoint distributions as shown in the Figure 3.3, which clearly demonstrates how the systematically structured viewpoints of Toybox dataset complements existing datasets for the purpose of studying the effect of

viewpoint distributions on object recognition.

Many common object recognition datasets (e.g., ImageNet, Microsoft COCO, etc.) contain only one image per real-world object[34, 17]. While these datasets have driven much exciting research in computer vision in recent years, they are, by their construction, limited in their applicability for supporting experiments to understand the viewpoint-varied object recognition of deep CNNs.

Several existing datasets are already beginning to fill this gap, as listed in Table 3.1. Each of these datasets contains more than one viewpoint, i.e., naturally captured image, per object. We do *not* include synthetic image manipulations in this table, such as artificially skewing or scaling original images to create new ones. We also do not include datasets that rely on fully synthesized objects/images.

While most of these datasets have provided background variations such as lighting, location, etc., many of these datasets have captured only a discrete collection of viewpoints, using, for example, a turntable turned by every 3° or 45 ° [131, 133, 132]. iLab-20M [133] is the only dataset that has more images per object than Toybox by counting all the other variants but the number of viewpoints is just 88 per object. Of the datasets that provide dense images of objects, the majority of them captured arbitrary handling of the objects [135, 134].

Also, while at least one other dataset has captured manually performed, continuously varying rotations (e.g., iCubWorld-Transformations), these rotations are labeled only by broad type of rotation (e.g., in-plane or in-depth), and thus for a given image frame, specific pose information is not immediately available (i.e., would need be annotated).

Toybox contains images captured continuously at 30fps spanning full object rotations along all three rotational axes, as well as horizontal, vertical, and front-to-back (i.e., zooming) object translations. Manual rotations and other transformations in Toybox were timed to follow fixed patterns so that estimates of object pose can be calculated for the majority of Toybox images.

Table 3.1: Computer vision datasets that contain multiple real (i.e., not synthesized) images of the same physical object from different viewpoints.

Dataset	Reference	Categories	Objs/Cat	Viewpoints/Obj	Other Variants	Imgs/Obj	Total Imgs
COIL-100	[136]	100	∼1	72	n/a	72	7,200
SOIL-47	[48]	47	∼1	21	lighting	42	1,974
NORB[1]	[131]	5	10	648	lighting	3,888	194,400
ALOI	[137]	1000	∼1	72	lighting	111	110,250
3D Objects on Turntable	[138]	100	∼1	144	lighting	432	43,200
3D Object	[139]	8	10	24	zooming	72	∼7,000
Intel Egocentric[4,5]	[134]	42	1	various[6]	background, manual activity	1,600	70,000
EPFL-GIMS08	[46]	1	20	∼120	n/a	∼120	2299
RGB-D[2]	[132]	51	3-14	750	camera resolution	750	250,000
BigBIRD[2]	[140]	100	∼1	600	n/a	600	60,000
iCubWorld-Trfms.[3,4]	[141]	20	10	150-200	lighting, background, zooming	∼3,600	∼720,000
iLab-20M	[133]	15	25-160	88	lighting, background, focus	>18,480	21,798,480
CORe50[2,4,5]	[135]	10	5	various[6]	indoor/outdoor, slight handheld movement	∼300	164,866
eVDS	[142]	35	37-97	various[6]	n/a	>144	∼420,000
Toybox[4,5]	**[this paper]**	**12**	**30**	**∼4,200**	**translating, zooming, manual activity**	**∼6,600**	**∼2,300,000**

[1] *Stereo pairs not included in counts.* [2] *RGB-D video.* [3] *Updated counts from dataset website.*
[4] *Handheld objects.* [5] *Egocentric video.* [6] *Unstructured viewpoint distributions.*

As a final note on datasets, the datasets in Table 3.1 can also be divided by whether objects are on a table or other fixed surface (e.g., COIL, NORB, etc.) versus handheld (e.g., Intel, iCubWorld-Transformations, CORe50). Because objects in Toybox are handheld, images do contain some occlusions, which are unstructured; people collecting Toybox data were instructed to hold objects naturally while performing each object manipulation. Thus, Toybox may also be interesting as a testbed for studying the manual affordances of objects.

3.5 Toybox Dataset Collection, Organization and Annotation

Figure 3.4 provides an overview of the Toybox dataset. This section provides details about the design of the dataset and our recording methods.

3.5.1 Selection of categories and objects.

Toybox contains 12 categories, roughly grouped into three super-categories: *household items* (cup, mug, spoon, ball), *animals* (duck, cat, horse, giraffe), and *vehicles* (car, truck, airplane, helicopter).

To maximize the usefulness of Toybox for comparisons with studies of human learning,

Figure 3.4: Toybox overview. Toybox contains 12 categories with 30 individual physical objects per category. There are 12 video clips per object. Each clip contains a defined transformation of the object: two full revolutions for rotation clips and three back-and-forth shifts for translation clips. A final "hodgepodge" clip contains unstructured object motion, mostly rotations. Please see Supplementary Video 1 for representative clips.

all 12 of these categories are among the most common early-learned nouns for typically developing children in the U.S. [5]. Categories were also selected to provide shape variety in each super-category (e.g., spoon vs. ball, duck vs. cat, etc.) as well as shape similarity (e.g., cup vs. mug, car vs. truck, etc).

Each category contains 30 individual physical objects. For both animals and vehicles, we cannot include real objects, and so objects are either realistic, scaled-down models or "cartoony" toy objects. Objects were purchased mostly in local stores, with some acquired online. Individual objects were selected to provide a variety of shapes, colors, sizes, etc., and can be considered a representative sampling of typical objects available in the U.S.

3.5.2 Recording devices

Videos were recorded using Pivothead Original Series wearable cameras, which are worn like sunglasses and have the camera located just above the bridge of the wearer's

nose. Camera settings included: video resolution set to *1920* x*1080*; frame rate set to *30 fps*; quality set to *SFine*; focus set to *auto*; and exposure set to *auto*.

3.5.3 Canonical views

For each category, we defined a canonical view of the object, roughly centered in front of the camera-wearer's eyes. Cups and balls start in an upright position with no need to adjust the rotation due to the shape symmetry. Mugs and spoons start in an upright position with the handle to the right. Animals and vehicles start in an upright position facing towards the left.

3.5.4 Video clips

For each object, a set of 12 videos was recorded, as shown in Figure 3.4. Each clip is $\sim$20 seconds long, with the exception of absent/present video clips, which are $\sim$2 seconds long. For rotations, each clip contains two full revolutions of the object; for translations, each video contains three back-and-forth translations starting from the minus end of each axis. Rotations and translations were controlled to have an approximately constant velocity over the 20-second duration of the video. To do this, we developed a set of audio "temporal instruction templates" that camera-wearers would listen to while creating each video. Thus, the pose of the object in every frame of a given video clip can be estimated according to its time.

3.5.5 Rotation terminology

We defined six different rotations in our dataset as shown in Figure 3.5. We follow the right-hand rule to define the plus direction around the x, y, and z axes. So the rxplus rotation is to rotate the object towards you and the rxminus rotation will is to rotate the object away from you. The ryplus rotation is to rotate the object clockwise parallel to the

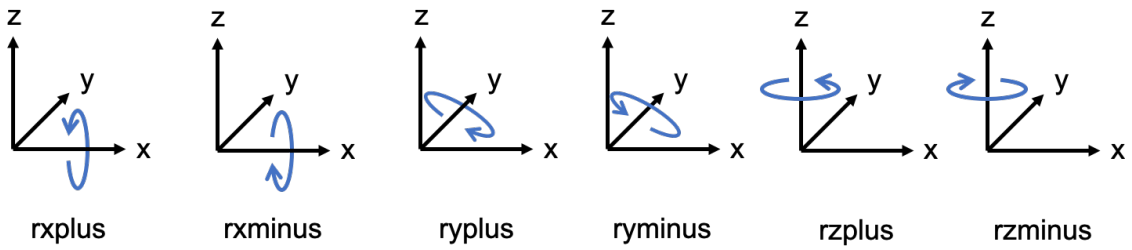


Figure 3.5: The six rotation directions collected in the Toybox dataset.

wall and the rymius rotation is to rotate the object counter-clockwise parallel to the wall. The rzplus rotation is to rotate the object counter-clockwise parallel to the ground and the rzminus rotation is to rotate the object clockwise parallel to the ground.

3.5.6 Recording procedures

Objects were semi-randomly assigned to individual camera-wearers (members of our research lab) such that no individual was over-represented in any category or object size class, to reduce any biases related to specific personal attributes or individual hand gestures. All videos were collected in an indoor setting against an off-white wall. Recordings were made across various times of day and lighting conditions, and so there is variation in lighting across different objects (as can be seen in Figure 3.4).

3.5.7 Organization of objects

After we finished all the recordings, we sorted our objects into well labeled boxes and padded them with foam materials for future reference. Figure 3.6 and 3.7 show the overview of the organization of the Toybox dataset objects. We have also built a reference book based on these blue tags over the object for the quick indexing of the object in the future.

Figure 3.6: An overview of all Toybox objects well sorted in padded boxes. In general, each box will contain all of the objects in the same category. The blue tag on the top of each object is the object number we assigned to uniquely identify each object. Certain balls has been deflated to save space but were inflated while recording.

3.5.8 Bounding box annotation

Figure 3.18 shows an overview of all 360 objects (12 categories $\times$ 30 objects per category) in the Toybox dataset. These objects were the objects we published online for bounding box annotation, i.e., drawing one bounding box for the object held in hands in each image.

As an object-centered dataset, to trace where the object is from the egocentric view is important to explore object recognition models. For example, the changing of viewpoints may change the size and aspect ratio a lot even for the same object.

Figure 3.7: The left figure shows the large objects in Toybox dataset that cannot fit in a single box. The right figure is an overview of all objects sorted in labeled boxes.

Thus, we crowdsourced bounding box annotations through Amazon Mechanical Turk (MTurk) [30] for the 6 rotations (rxplus, rxminus, ryplus, ryminus, rzplus, and rzminus) and the hodgepodge videos at 1 FPS, which is approximately 30 degrees changing per frame in the rotation videos.

We instructed the workers to only draw one bounding box per image for the object held in hand. Since most of the objects will be partially occluded by hands, we also instructed that if you couldn't see the boundaries of the object, please guess where the boundaries would be and include those invisible parts of the object inside the bounding box as well.

Since the video recording was at 30 FPS and 1920x1080 resolution, we downsampled the frame rate to 1 and kept the original resolution for bounding boxes. We first tried 3 assignments per Human Intelligence Task (HIT), i.e., the MTurk terminology to indicate how many workers will work on one task. We found that, after further specifying worker requirements of task acceptance rate $> 99\%$ and total completed HITs > 1000, even for 1 assignment per HIT, i.e. 1 bounding box per image, the quality of the bounding boxes was high enough for our research purpose. Thereafter we only collected 1 bounding box per

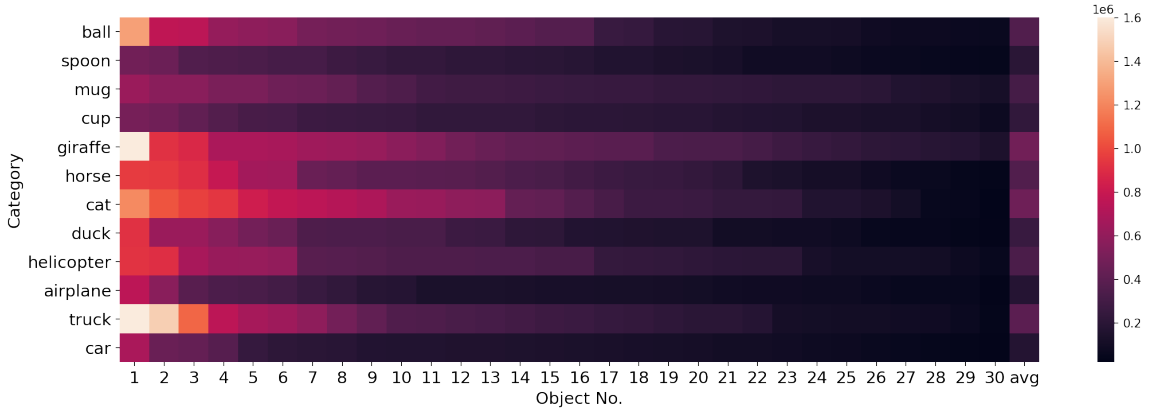


Figure 3.8: The bounding box size, i.e., $boxwidth \times boxheight$, calculated from the 1st frame of the rzplus rotation video for the objects in Toybox. The largest possible size is $1920 \times 1080 = 2.0736 * 1e6$. Each row is a different category and each column is the object number, except the last column which shows the average bounding box size for that category. Categories belonging to the same super-category are grouped together, i.e., the first 4 rows are the household objects, the second 4 rows are the animals, and the last 4 rows are the vehicles.

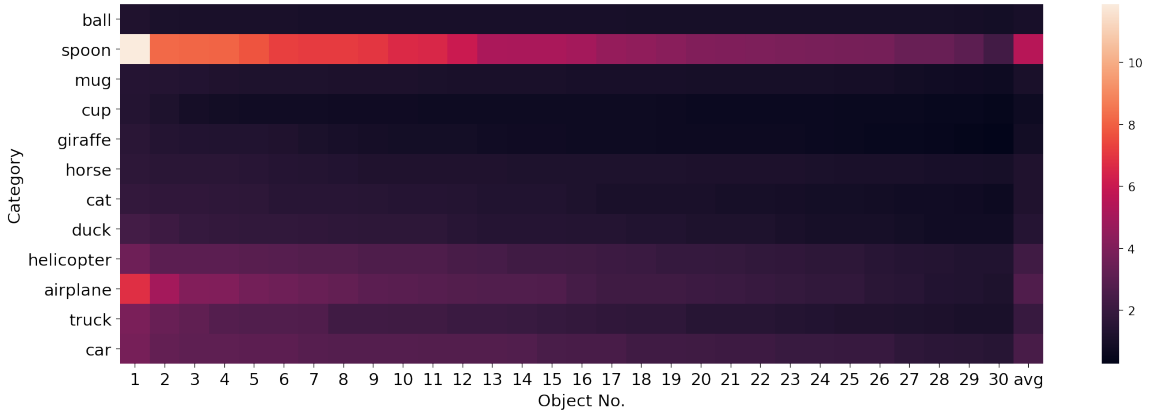


Figure 3.9: The bounding box aspect ratio, i.e., $boxwidth/boxheight$, calculated from the 1st frame of the rzplus rotation video for all of the objects in Toybox. Each row is a different category and each column is the object number, except the last column to show the average bounding box aspect ratio for that category.

image. After the completion of the crowdsourcing annotation, we manually went through each bounding box to fix occasionally missing values or misplaced boxes. Eventually, we got 48868 valid bounding boxes for rotation videos and 8317 valid bounding boxes for hodgepodge videos.

Figure 3.8 and Figure 3.9 shows the bounding box size and aspect ratio in Toybox from

the first frame of the rzplus rotation. Since these objects were all held in hand and recording using an egocentric, head-worn camera, the bounding box size and the aspect ratio of each object will reflect approximately the real size and ratio projected onto the receptive field of humans' eyes.

As a result, we can tell one object is "bigger" than another object, which is harder to determine in datasets like ImageNet [34] or Coco [17]. Also notice how the object size could vary a lot for the same category, i.e., an object may occupy almost the entire receptive field (80% of the entire view) such as giraffe 29 or truck 7 (whiter boxes), or just a small field (10% of the entire view) such as airplane 1 or spoon 23 (darker boxes).

Compared to the size variance, the aspect ratio is more fixed within the same category; for example, most of the spoons have a larger aspect ratio because of their thin and long shape. Such a similar aspect ratio property even exists at the super-category level; animals tend to have an aspect ratio around 2 (see the similar colors for giraffe, horse, cat, and duck in the middle of Figure 3.9), and vehicles tend to have an aspect ratio around 5 (see the similar colors for helicopter, airplane, truck, and car at the bottom in Figure 3.9). If you take a look at the last column for an average of all objects in the same category, you will notice how the super-category groups similar aspect ratio categories together.

Figure 3.10 shows how the aspect ratio will change along with viewpoint changes during rzplus rotation. Since in Figure 3.9, we showed that the aspect ratio remains similar inside each category, to plot this graph, we chose the first object in each category as the representative to show the aspect ratio changing trend. Categories in the same super-category tend to oscillate in the same range, with spoon as an exception. If we think about the symmetry of these objects, for perfect rotations without "human noise," the line graph should be a perfect sine or cosine curve, but if we take a look at how humans rotated these objects in Figure 3.10, we will notice how humans will constantly make some perturbation to the object that augments the visual experience of the same object.

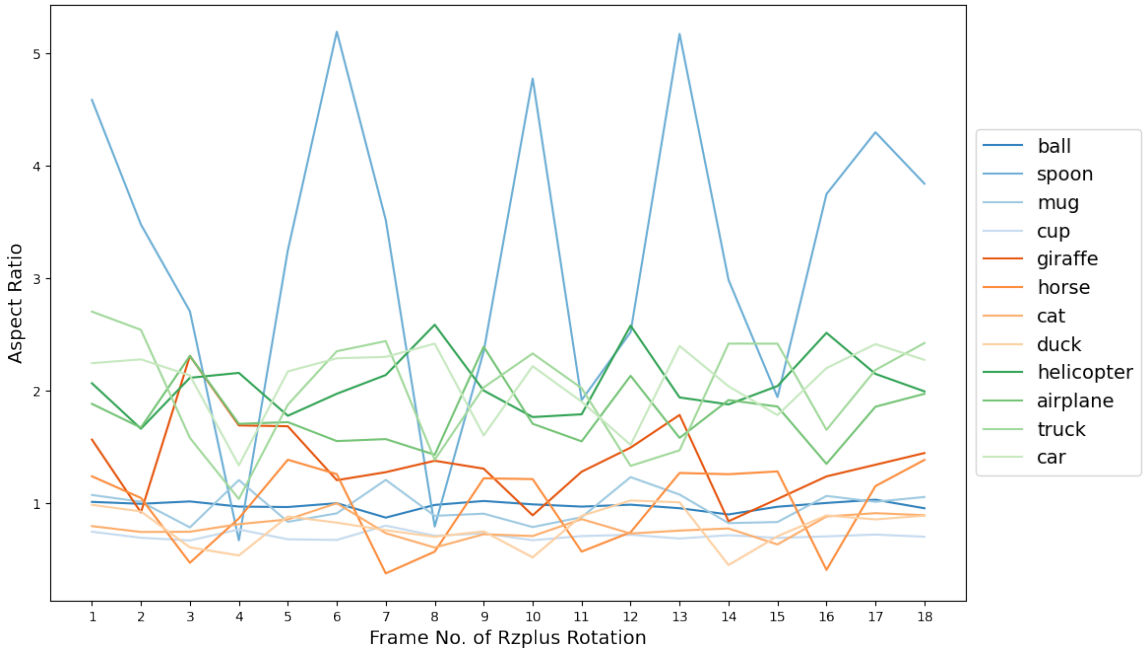


Figure 3.10: The bounding box aspect ratio changes along with the viewpoint changes during rzplus rotation for the first object in each category. The y-axis is the aspect ratio and the x-axis is the frame number in rzplus rotation (approximately $360/18 = 20$ degrees per frame.) Colors are grouped by super-category: blue for households, red for animals, and green for vehicles.)

3.5.9 UMAP visualization of Toybox

UMAP [143] is an algorithm to reduce higher dimension data into a lower dimension space. It achieves this by first constructing a high-dimensional graph of all data points by the nearest neighborhood (the distance metric can be defined differently) and then building a low-dimensional graph that could be as similar as possible by optimizing the loss between the high- and low- dimensional graphs. The author claimed that by stipulating each dot connect to at least one nearest neighbor, the global structure can be preserved so that the distance in the low-dimensional space are not just meaningful for points insides the same cluster but for points in different clusters.

In the UMAP paper, the author used the COIL-20 [144] dataset as an example to show the dataset structure topology. COIL-20 captured 20 different objects from different viewpoints on the turntable, and we found that the UMAP visualization of it is interesting such

that many "loops" formed for each object because in a full rotation of the same object, the first view is smoothly transitioning to the last view and the first view and the last view are very similar.

Toybox as a viewpoint-rich dataset that has the hierarchical structure, we expected the structure of the Toybox dataset would be very different from the ImageNet dataset, and hence we utilized the UMAP as tool to inspect the dataset structure of Toybox.

The implementation of UMAP is the official UMAP repo [145].

The Toybox images used here are all square cropped images for the objects in all 6 rotation videos. We collected the bounding boxes for the original resolution (1920×1080), so we post-processed the bounding boxes into squares and then resized images to 128×128.

We ran UMAP on the raw input space and the feature space extracted by a trained Inception-v3 [32] network.

To train the network to extract features from the Toybox dataset, we followed the following process. In Toybox, we have 30 objects per category, so we split these 30 objects into 27 objects for training and 3 objects for testing. We use all the cropped images from the rz rotation videos so that each object will have 2 rotations (rzplus/rzminus) and 18 images per rotation $=$ 36 images per object. The training dataset then will then contain 12 categories $\times$ 27 objects per category $\times$ 36 images per object $=$ 11664 images in total and the testing dataset will contain 12 categories $\times$ 3 objects per category $\times$ 36 images per object $=$ 1296 images in total. We trained the network with 100 epochs, batch size 64, and learning rate 0.01 with 0.5 times learning rate decaying every 10 epoch. Finally, the network achieved 0.859568 accuracy on the testing dataset.

The dimension reduction for raw the input space was calculated from the flattened images, i.e., from $128 \times 128 * 3$ to a vector of length 49152. The feature space has a vector of length 2048.

For UMAP visualizations in Figures 3.11 and 3.12, the left sub-figure is the visualiza-

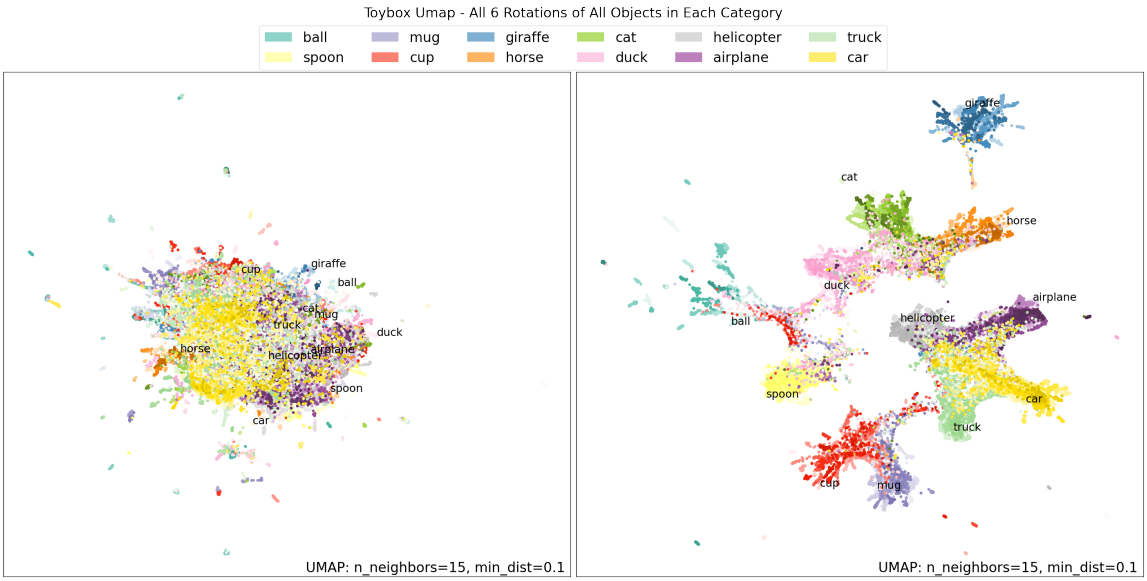


Figure 3.11: The UMAP visualization for all objects in all 6 rotations. Color series encodes the category and the darkness of colors in the same series encodes the object. The little text on each cluster indicates the category of that cluster as well.

tion for the input space, and the right sub-figure is the visualization for the feature space.

Figure 3.11 shows the hierarchy of our Toybox dataset from super-category, category, to object level. The feature space shows how super-categories are clustered together, i.e., animals are clustered at the top right in the feature space, households clustered at the bottom left, and vehicles clustered at the bottom right.

Figure 3.12 shows the UMAP visualization for all 6 rotations of a single object in each category. Notice that the viewpoints from different rotations in the input space form cycles, since when an object gets rotated in a full revolution, the first viewpoint will be very similar to the last viewpoint, and viewpoint changes are incremental in between. Thus, such a start view — different view — end view — start view cycle will be reflected as a cycle in the embedding space.

In the feature space, to some extent, this viewpoint information seems to disappear in the feature extracting process, as many cycles have shrunk to crowded clusters. The viewpoint information disappearing through the feature extraction of the network is expected since the network we trained is just for an object classification task with no need for re-

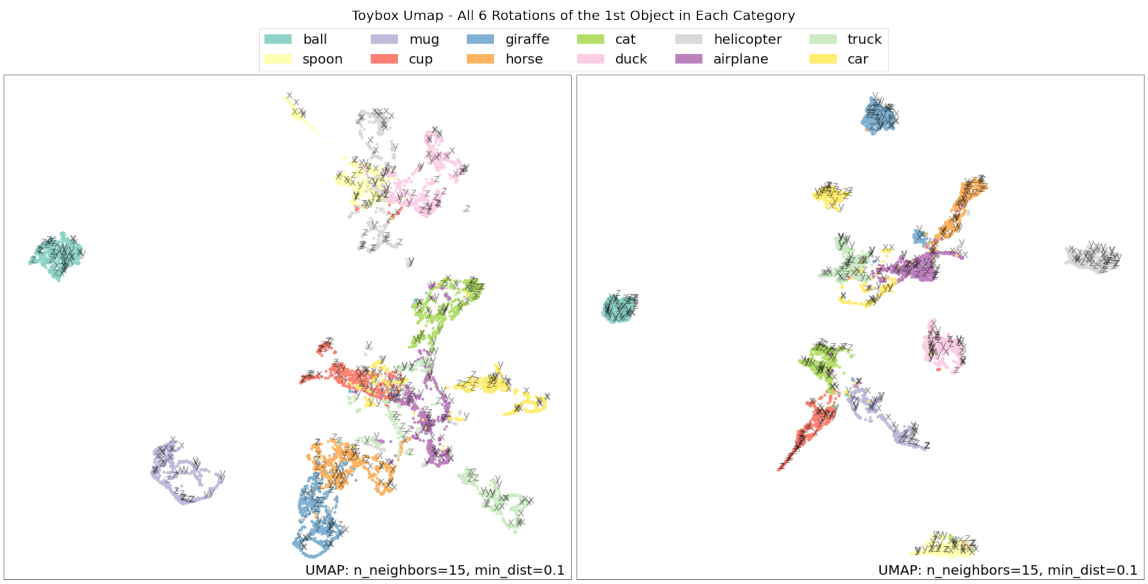


Figure 3.12: The UMAP visualization for all 6 rotations of the 1st object in each category. The little text on each dot is which rotation the dot belongs to. The color encodes the category. Notice different viewpoints in the input space will form cycles, whereas those cycles will get reduced in the feature space. This shows how viewpoint information is lost or reduces through the feature extracting process in neural networks.

taining viewpoint-specific information. In fact, the network is essentially trained to be viewpoint invariant. If we do not inject pose information somewhere in the loss function or the network architecture, the classic convolutional neural networks will not keep this information.

3.6 Comparison with ImageNet

The design principle for Toybox and ImageNet is quite different. Toybox is intended to capture developmentally-relevant aspects of embodied human visual experience, whereas ImageNet uses web-crawled photos, essentially covering a "Google-Image-search" flavor of visual experience, as shown in Figure 3.13.

Figure 3.14 shows the UMAP visualization comparison between the Toybox and ImageNet. The hierarchical structure designed in Toybox, i.e., the super-category, category, object, etc., makes the island clustered in such a hierarchical way, whereas the ImageNet,

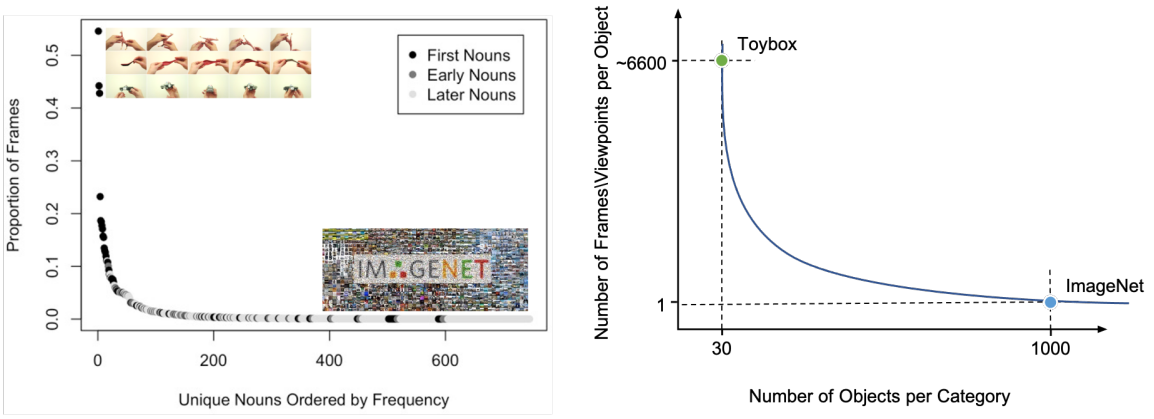


Figure 3.13: The left figure is updated from the paper [4] showing where the Toybox and ImageNet would be located in a space of possible object/image distributions, in comparison to typical infant visual experience. In this figure, the y-axis is the proportion of frames analyzed from the head-mounted camera videos on infants' heads. The x-axis is the unique nouns, such that "nouns" is a developmental psychology preferred terminology for "categories" in the machine learning field. Thus, the y-axis is essentially meaning that how long the same category will be exposed to the infants and the x-axis is for different categories. Since Toybox have collected the common early-learned categories for typically developing children in the U.S. [5] and record these categories in a long time exposure ($\sim$ 6600 images per category), the Toybox dataset will be on the top-left corner in this space. On the contrary, ImageNet collected more than 1000 categories in which many of them are out of the early-learned categories, and each category will have $\sim$1000 images, the ImageNet will be on the bottom right corner in this space. The right figures compares Toybox and ImageNet in the object level. The y-axis is still the number of frames as the left figure, but the x-axis now is the number of objects per category. Toybox is on the top-left corner where just a few instances are available but with extensive viewpoints per instance to better capture the instance-limited and viewpoint-rich features we think that is important for human vision development as discussed in Chapter 1. ImageNet is on the bottom-right corner where tons of instances are available but with very few viewpoints that lacks such a property.

composed of web-crawling images, does not have such a hierarchical structure.

3.7 Exploratory Experiments Using Toybox

For initial, proof-of-concept object recognition experiments with Toybox, we use the transfer learning methodology appearing in many recent studies, e.g., [146, 141], which involves retraining the last layer of a pretrained, deep convolutional neural network.

We used the ImageNet ILSVRC 2012 pretrained Inception v3 network[147] as a fixed

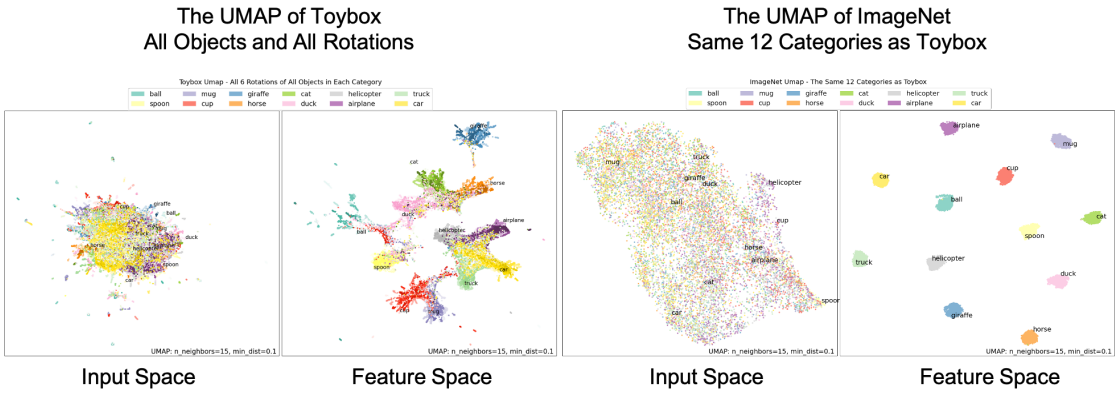


Figure 3.14: The UMAP visualization comparison between Toybox and ImageNet. The left figure is for Toybox and the right figure is for ImageNet. The hierarchical structure designed in Toybox is clearly visible, whereas no such structure is observed for ImageNet.

feature extractor, and then retrained the last layer as specified for each experiment. More than half of the Toybox categories do appear in the original 1,000 categories used for pretraining in ImageNet—except for helicopter, giraffe, horse, and duck. Notice that although the ILSVRC 2012 dataset does not contain all categories in the Toybox dataset, the entire ImageNet dataset does so we could build a test ImageNet dataset which has the same categories as Toybox.

We ran two experiments related to viewpoint variance. The first experiment is to investigate the training effects by different number of objects and viewpoints. The second experiment is to investigate the object recognition accuracies tested along with different viewpoints. The Toybox images used in these proof-of-concept experiments are the images sampled from the original videos without any bounding box cropping.

3.7.1 Experiment 1: Using Toybox to study the effects of instance diversity and view diversity on recognition

In Experiment 1, we retrained the last layer of the ILSVRC 2012 pretrained Inception v3 network using images from Toybox, and then tested recognition performance using images from the same categories from ImageNet. Test performance is measured as the

top-1 error rate.

Note that the choice of using ImageNet images (instead of hold-out Toybox images) for testing was deliberate. We aimed to explore how well training on a small number of handheld, often toy objects would be able to generalize to the very different objects/views in ImageNet (e.g., training on toy cats to recognize real cats). (Certainly other testing approaches would also be interesting and will be discussed in Chapter 4 on the Toybox benchmark.) We constructed this ImageNet test set to contain 100 images/category across the 12 Toybox categories.

Object variance. We first looked at the effect of *object variance* on recognition performance by varying the number of individual physical objects per category in the training dataset, while keeping the total number of training images per category fixed at 1100 and uniformly drawn from the various video clips contained in the Toybox dataset.

For example, with one object per category, each of the 12 categories is represented by 1100 images of a single object from that category. With two objects per category, each category is represented by 1100 images uniformly drawn across two objects (550 images per object on average).

Results from this experiment are shown in Figure 3.15A. A training set with images of only a single Toybox object per category (i.e., 1100 images per object) yields an average error rate of 60.63%, which while not excellent, is well below the random-guessing baseline error rate of 91.7%. Adding a second object (i.e., about 550 images per each of two objects) further reduces error to 51.98%. Adding more objects per category (with total training images per category fixed at 1100) continues to improve performance significantly, with our final experiment using 30 objects per category yielding an average error rate of 21.43%.

We also characterized the performance improvement by computing best-fit lines using both linear and exponential models. As shown in Figure 3.15A, the exponential curve yields a better fit. Therefore, at least from the perspective of this model fitting, it appears that increasing object diversity will reduce the error rate in an exponential manner, with

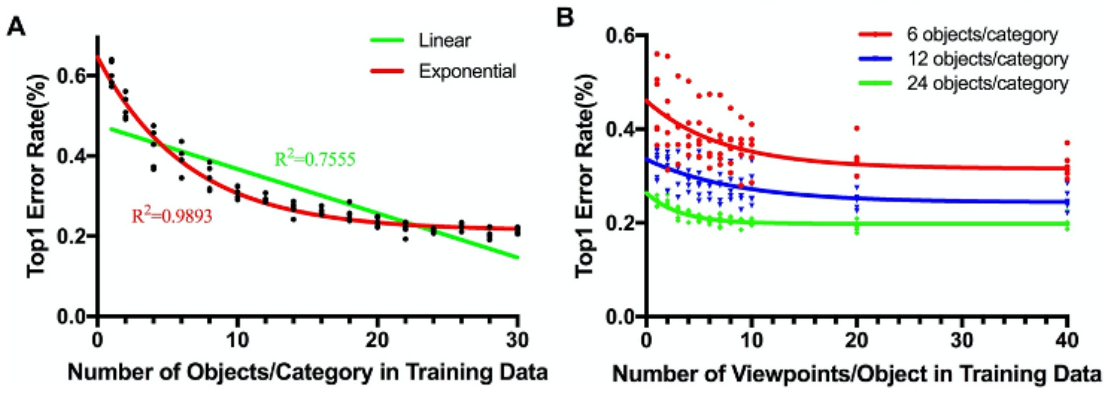


Figure 3.15: Effects of object variance and viewpoint variance on recognition performance, measured as top-1 error rate on an ImageNet-sourced test set. Each trial was run 5-6 times, shown as individual data points. **A**. Recognition as a function of instance diversity, i.e., number of objects per category in the Toybox training set, with the total size of the training set held fixed. **B**. Recognition as a function of view diversity, i.e., number of images per object in the Toybox training set, ranging from 2 to 40 images per object.

much greater improvements in performance for the first few added objects, and smaller increases thereafter (especially after about 20 individual objects).

Viewpoint variance. We also looked at *viewpoint variance*, by varying the number of images per object included in the Toybox training set. By sampling these images uniformly across all Toybox video clips, the number of images per object can be used as a proxy for views per object. We conducted this experiment under three conditions, with the total number of objects per category fixed at 6, 12, and 24, respectively.

For example, for 12 objects per category condition, we varied the total number of images per object from 2 to 100, drawn uniformly across all 12 objects. Specifically, if we pick 2 images per object, the training dataset would have $2 \times 12 = 24$ images per category, and similarly, if we pick 100 images per object, the training dataset would have $100 \times 12 = 1200$ images per category.

Figure 3.15B shows results from this experiment. (Although we experimented with numbers of images per object up to 100, we noticed a near constant error rate once this number exceeded 40, and so the graph in the figure is truncated at $x = 40$.) Results across the three conditions show similar trends, and so we focus our discussion here on the 12

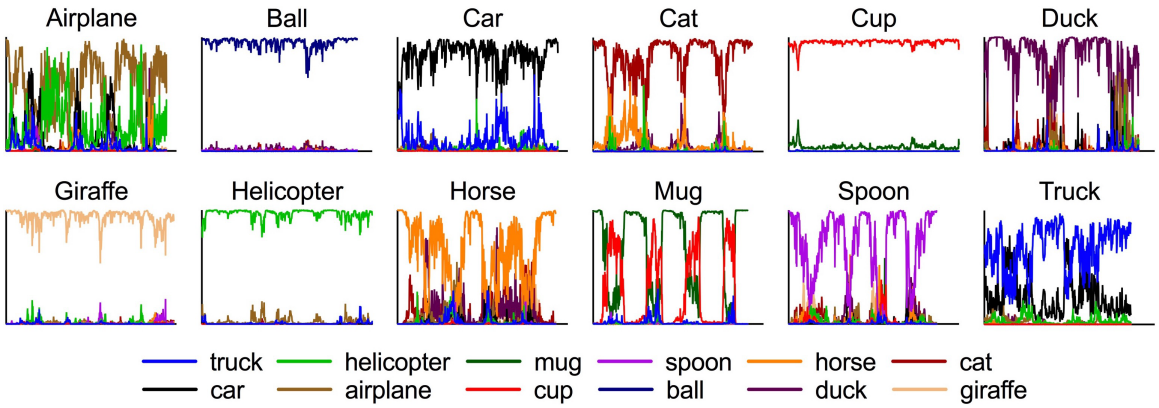


Figure 3.16: There are 12 sub-figures, in which each figure is the activation changing of all 12 output neurons as encoded by colors while the network is watching rzplus rotating objects in the category shown in the sub-title. The y-axis is the output layer activation, and the x-axis is the rotating degree of the object. We noticed many interesting viewpoint-varied recognition patterns such as mug vs cup and cat vs horse.

objects/category condition (blue data points and curve).

With a single image per object, the average top1 error rate is 33.0%. This error rate is subsequently reduced to 27.5% if we have 10 images per object, and is further reduced to 25.6% and 24.8% for 20 and 40 images per object, respectively. As with object diversity, the effects of view diversity appear to show an exponential trend, with only modest improvements after about 5-10 views per object.

3.7.2 Experiment 2: Using Toybox to study object recognition for continuous viewpoints

The neural network setup is similar to the previous experiment, i.e., the Inception v3 network pretrained on ImageNet and then transfer-learned on 12 categories in Toybox. However, to test the effect of the changing of continuous viewpoints of the same object for object recognition, we can't use ImageNet anymore because ImageNet does not have the continuous viewpoint information as the Toybox has. Instead, the test set is the leave-out set of Toybox, which contains 3 objects that have never been seen in the training dataset. Then the network will watch a rzplus rotating object to give the classification prediction for different viewpoints.

Figure 3.17: A mug example of rzplus rotation. Notice how the appearance will get confusing from mug to cup if we didn't capture enough viewpoints from the same object. This confusion has also been reflected in the neural networks as how the output neurons, mug and cup, are oscillating relative to each other while the network is watching a rotating mug (Figure 3.16).

Figure 3.16 shows the average value of output neuron activations over 3 test objects while the network is watching them rotating along with the z-axis. Different colors represent different output neurons.

The recognition of a rotating ball, cup, giraffe, and helicopter is viewpoint-robust along the z-axis viewpoints; however, some categories have clear confusion patterns with other categories. For example, a rotating airplane will make the neural network get confused between an airplane and a helicopter; a rotating cat will make the neural network get confused between a cat and a horse; and a rotating mug will make the neural network get confused between a mug and a cup. In fact, these kinds of confused rotating objects might make human vision confused as well. Figure 3.17 is a good example of how a rotating mug might be recognized as a cup from a certain range of viewpoints.

These results indicate an important conclusion that different categories might have very similar appearances as a function of viewpoint; for example, the bottom view of a car and a truck or the back view of a mug and a cup. Therefore, this type of error is neither a bias error, e.g., the model is too simple to fit the data, nor a variance error, e.g., the model is too sensitive to the trivial details of the dataset, but rather intrinsic noise existing in the dataset.

Surprisingly, in the human performance benchmark we conducted in the Chapter 4, human workers on MTurk [30] can still achieve high recognition accuracy on rare viewpoints.

3.8 Discussion

In this chapter, we presented the new Toybox dataset of egocentric visual object transformations from design, to collection, to annotation. We also provided results from two sample experiments showing how this dataset can be used to study visual learning, including (1) effects of instance diversity and view diversity on recognition performance, and (2) using Toybox to study object recognition for continuous viewpoints.

In addition to the experiments presented here and in Chapter 4, we expect that the Toybox dataset will be valuable for studying new types of representations and learning algorithms that lend themselves to continuous image sequence inputs (though not the research direction in this dissertation).

For example, in human vision, object motion is critical for segmentation, and also likely plays a role in many other aspects of object detection and recognition. Most of the large-scale computer vision datasets are composed of discrete images that are randomly selected, while humans learn most vision tasks through continuous input. Toybox allows us to start addressing questions such as whether getting continuous visual input is beneficial for recognition performance, and how and why this might be the case. This continuous viewpoint property of Toybox will also be relevant to study different learning regimens such as sequential learning [148] and incremental learning [84].

In addition, the ability to represent object motion and self-motion is essential for both humans and artificial intelligent agents. However, since most of the deep CNNs are built to train on randomly selected, static images, these CNNs cannot detect object motion at least from early convolutional layers. In contrast, humans have neurons both in the retina and visual cortex that prefer certain motion directions [149, 150]. How motion features affect recognition performance, and how object motion might contribute to the learning phase as well (for instance, by providing a real-time version of data augmentation), are currently open research questions in the study of visual learning.

With its structured object transformations and wide selection of categories and object

instances, we believe the Toybox dataset will help drive continued research advances on these and many other important questions in AI and cognitive science.

3.9 Conclusion

In this chapter, I present the design principles, collection, annotation, and properties of the Toybox dataset. I show how Toybox has the egocentric, instance-limited,and viewpoint-rich properties that characterize human visual learning. Finally, I explain how, because Toybox has these properties, it fills important gaps in currently available object recognition datasets to support research on how variations along these dimensions may affect object recognition performance.

Figure 3.18: The overall view of the objects in the Toybox dataset. Images sampled from the first frame in the rzplus rotation videos. Each column is a different category and each row is a different object in that category.

Chapter 4

Egocentric Variable-Viewpoint Object Recognition Benchmark

4.1 The Research Question

How well do different deep learning models perform on the Toybox dataset both with and without pretraining? What are the differences in performance between deep learning models and humans on the Toybox dataset, in terms of overall accuracy and as a function of categories, objects, and viewpoints?

To answer these questions, this chapter will go through the Toybox dataset setup for benchmarking the object recognition performance between deep learning models and humans; the Toybox object recognition benchmark among non-pretrained deep learning models, pretrained deep learning models, and humans; the investigation of where the differences are between deep learning models and humans in different levels of the hierarchical structure of the Toybox dataset; and the summary of the Toybox benchmark.

4.2 Introduction

While deep learning has pushed object recognition to higher and higher levels of performance, the training schema used in conventional deep learning models is actually quite different from the developmental trajectory for humans. Infants in the first two years of life are often in a visual environment where only a few objects are available to them, but these objects are extensively observed from various viewpoints [151, 118, 4]. In contrast, deep learning models generally rely on large-scale datasets with only one image (and thus only one viewpoint) per physical object and a large diversity of objects, though the distribution of viewpoints per object is also often limited.

Additionally, besides the training environment difference, how humans and deep learning models will handle object- and viewpoint- level variance at test time might also be

different. The investigation of these differences may provide insights about how we could design models that can learn from more naturalistic inputs and also that might perform more like humans at test time.

Egocentric, instance-limited, viewpoint-rich are the visual properties that characterize our everyday visual experience (including both "training" and "testing") while we are growing up, and so benchmarking model performance in this environment will drive us to build models that might learn, and perform, in more developmentally relevant ways. Supported by the Toybox dataset that contains these properties, this chapter will provide a new object recognition benchmark for deep learning models and humans, and then analyze where the differences are between them in different levels in the Toybox hierarchical structure.

4.3 Methodology

To maximize the viewpoint variance in data for our benchmark experiments, we mainly focused on the rotation videos in the Toybox dataset, i.e., the rotating directions of rxplus, rxminus, ryplus, ryminus, rzplus, and rzminus as shown in Figure 3.5 in Chapter 3.

We used Amazon Mechanical Turk (MTurk) [30] as the crowed sourcing platform to collect human performance data. When we first published object recognition tasks online, the results we retrieved seemed overly high, i.e., even for some "nonsense" images to us, e.g., some cropped images with only a small patch of the object in the image, these images could still be recognized successfully. We came up with some potential explanations such as the workers might work together so they can share information for what they have annotated, or the workers themselves haven't been blocked properly for the same objects online that are more clear to see. Finally, we realized that the urls referring to these images hadn't been encrypted, so that the categories of these images could be directly found through the urls if the workers know how to extract them. Therefore, we carefully maintained a worker ID list to keep tracking who has done what, and we applied hashed functions to the image urls to hide all information that might point to the answers to avoid potential target leakage.

After resolving these issues, we designed two experiments to build the benchmark.

The purpose of the first experiment is to calibrate the benchmark across objects and viewpoints to figure out the hard objects and viewpoints in Toybox. Thus, we used full resolution images (1080p) and rxplus/rzplus rotations only, since rxminus/rzminus are just reversed viewpoint sequences and ryplus/ryminus are in-plane rotations.

According to the Toybox recording instructions, we could estimate which viewpoint the current frame is. Based on such a viewpoint estimation, we took front, back, top, and down views for the rxplus videos and front, back, left, and right views for the rzplus videos. Figure 4.1 shows the locations of these 8 viewpoints where the front and back views from rxplus and rzplus videos are overlapped to each other; however, they may not be exactly overlapped because of the perturbation introduced by each human camera-wearer's manual object manipulations.

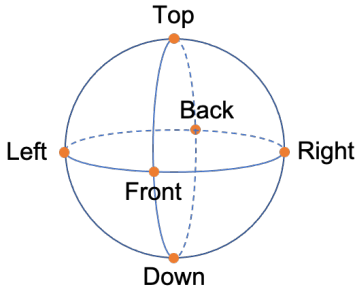


Figure 4.1: The 8 viewpoints used in our full resolution human performance benchmark experiment. This figure only shows 6 viewpoints since the front and back view are roughly overlapped to each other for the rzplus and rxplus videos.

We expected some viewpoints such as the bottom view of car and truck or the side view of mug and cup may bring object recognition confusions as shown in Figure 4.2. In particular, some objects may look very similar to each other in some particular perspective even though they are from different categories.

The second experiment is to compare the object recognition performance between deep learning models and humans, and therefore the images used here were cropped in square images with resolution 128×128. The bounding boxes are in 1FPS, approximately 30

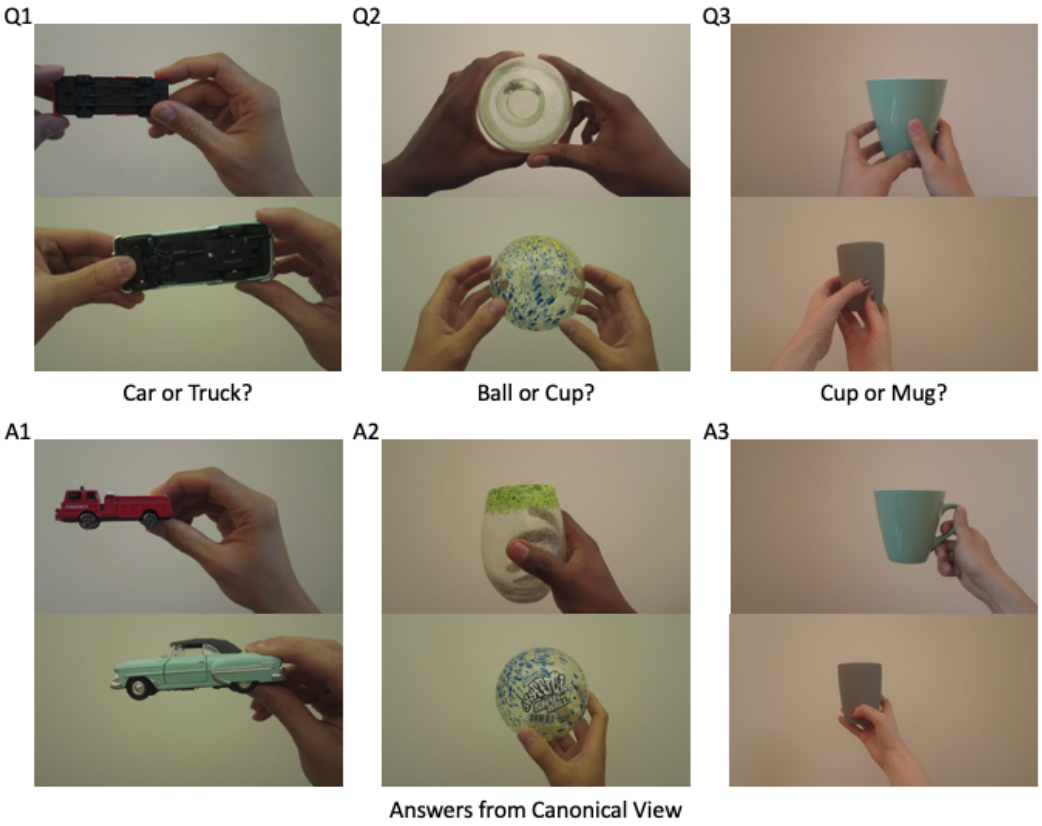


Figure 4.2: Objects from different categories could look similar to each other from particular viewpoints. Q1, Q2, and Q3 provide examples of the confusing viewpoints of the objects from different categories, and A1, A2, and A3 show how these objects could be easily recognized from other viewpoints.

degree rotation per frame, and are tight boxes. Hence, we implemented a small snippet to convert them into squares while keeping the object in the center.

Also different camera-wearers originally recorded the videos at different lengths, among which the shortest one has length 18 seconds. Thus, this shortest video will have 18 cropped images. For longer videos, we uniformly sampled 18 timestamps and chose the cropped images that are the closest to these timestamps. We now also include the rzminus/rxminus rotations to collect more data points.

4.4 Experiments and Results

4.4.1 Experiment 1. Human performance of the full resolution Toybox dataset

In this experiment, the images for testing contain 12 categories $\times$ 30 objects per category $\times$ 8 viewpoints per object $=$ 2880 images.

To ensure the same worker will not see the same object from a different viewpoint (if the worker can, the recognition may not be solely based on the current image but some features she or he remembered from other viewpoints), we didn't publish all 2880 images on MTurk all together. Instead, we divided 8 viewpoints into 8 batches, published them in sequence, and blocked all workers who have participated in previous batches from taking part on the next one. Also, we did 3 rounds of 8 batches to cover potential recognition bias or occasional mis-selections.

The questionnaire we published to workers is a multiple choice question with 13 choices for 1 answer. The left panel shows the full resolution image and the choices located at the right bar including 12 Toybox categories, ball, spoon, mug, cup, giraffe, horse, cat, duck, helicopter, airplane, truck, and car, and a "none of the above" option.

The results contain the object recognition information in different levels, i.e., which category, object, and viewpoint the image is, and in all 3 rounds, what the recognized categories are from the crowdsourcing workers.

Category Level Analysis. First, we will analyze the category level results.

Three rounds of object recognition from humans result in 2880 $\times$ 3 $=$ 8640 recognition data points, and the maximum value to get everything correct for a category is 8640 \$12=720.

Figure 4.3 shows the confusion matrix of the sum of all data points, in which we sorted the true label and human label by Toybox super-categories, i.e., the first four rows are household categories, the middle four rows are animal categories, and the bottom four rows are vehicle categories with an additional row of none-of-the-above at the end.

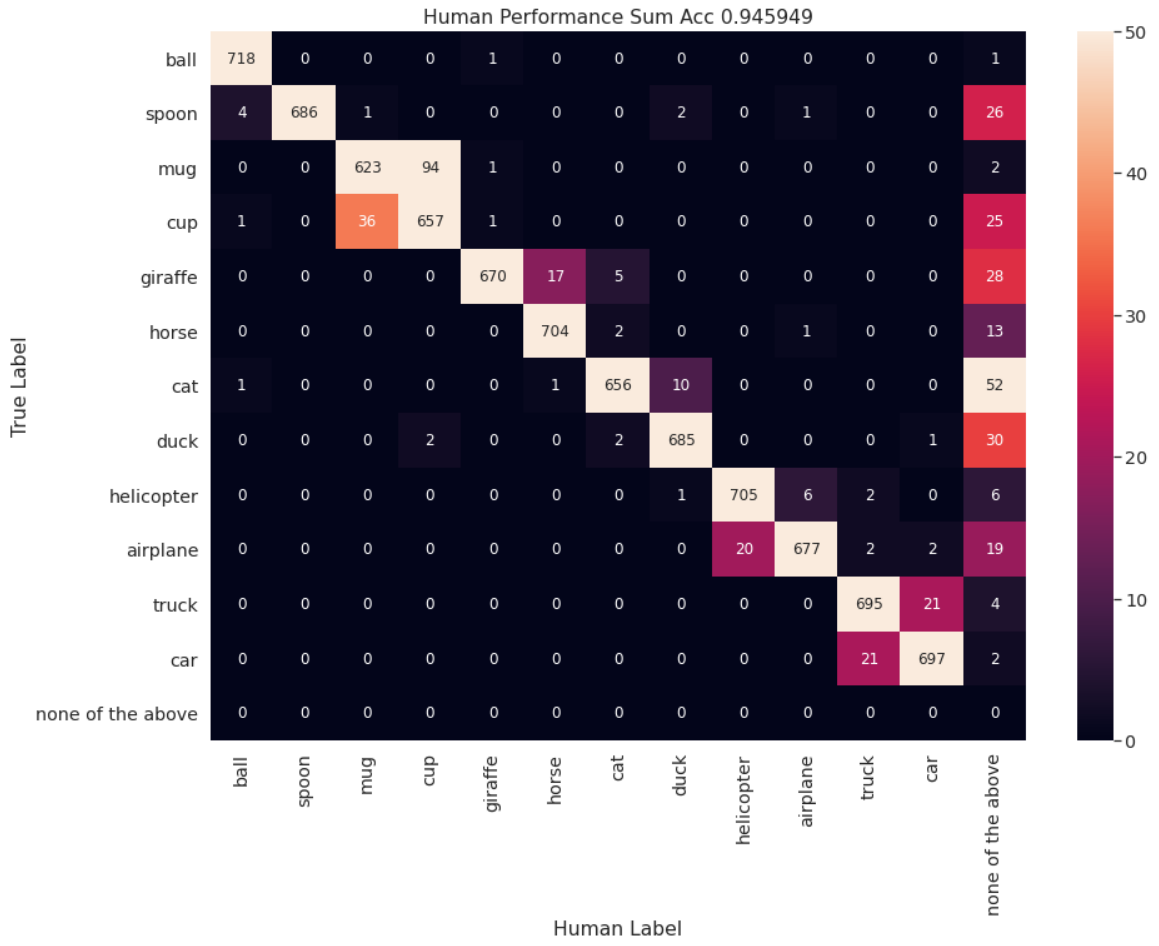


Figure 4.3: The confusion matrix of the sum of all human object recognition data points. The y-axis is the true label and the x-axis is the human label. Toybox by itself doesn't have the "none of the above" category but we allow workers to make this prediction if they choose. Color in a cell encodes how many correct recognitions happened in the associated true label and human label.

The overall accuracy is 0.945949 and we found some additional interesting patterns. First, categories in the same super-category are easier to get confused to each other, such as mug vs cup, giraffe vs horse, cat vs duck, helicopter vs airplane, and truck vs car, as most of the confusions are distributed near the main diagonal.

To investigate why these confusions happened for humans, we will get into the object level and viewpoint level analysis for the full resolution Toybox benchmark.

Object Level Analysis. Results from the object level analysis are shown in Figure 4.4. We sorted each row such that the objects that got fewer correct recognitions will be put in

the beginning, and the last column is the average value across all 30 objects in that category.

Different categories show different patterns; for example, for the car category, there is one object that got a very limited number of correct recognitions, but the rest got recognized very well. This is a surprising pattern to us, since if we think about the bottom view example for car and truck in Figure 4.2, we expected this viewpoint confusion would happen for most of the car objects, but this did not turn out to be the case.

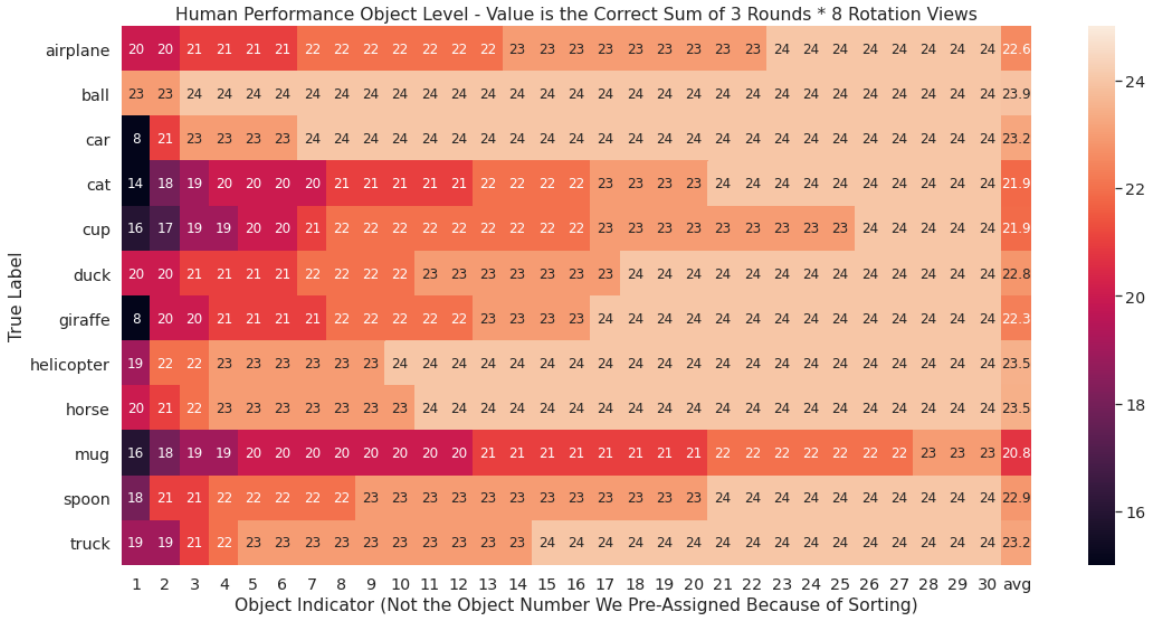


Figure 4.4: The object level analysis of the sum of all human object recognition data points. The y-axis is the true label, and the x-axis is the object indicator. Note the object indicator is not the object number we pre-assigned in the Toybox dataset, because we have sorted the values in each row for better visualization. The value in each cell is the sum of the number of correct recognitions from all 3 rounds and 8 views. Color encodes how many correct recognitions were obtained for that particular object.

However, viewpoint confusion is not the main reason for the misclassification of the car category. Therefore, we then examined our viewpoint estimation procedures for cars in Toybox for those frames we expected to be the bottom view, i.e., how we chose the Toybox images to use in our benchmark experiments. As shown in Figure 4.5, for the 30 objects we have in the Toybox dataset, 11 of them from our estimation are the true bottom-facing views marked by the white txt at the bottom right. However, humans seem still recognize

Figure 4.5: A sanity check for the viewpoint estimation in Toybox, used to select specific images for use in our benchmark experiments. The true bottom views are marked by the white text at bottom right.

most of these 11 bottom-facing views successfully.

One potential explanation could be that even though the bottom views of car and truck are very similar to each other, the number of wheels, the chassis' length, etc., can still provide enough information for humans to make correct recognition decisions.

After the viewpoint sanity check, we confirmed that some object-level recognition difficulties exist in the Toybox dataset, as Figure 4.6 shows the objects that got most misclassified. The car that only got 8 correct recognitions is very "cartoony" with eyes, tails, and tiger skin. Other difficult objects in Toybox include a short neck giraffe, a duck with hair, etc. These difficult objects are, we believe, one main reason for human misclassifications.

Viewpoint Level Analysis. In the viewpoint level analysis, since we have identified the difficult objects above, we will remove them to focus on the error patterns more on the viewpoint level.

Figure 4.7 shows all the images that no one recognized correctly (i.e. 0 out of 3 correct recognitions per image), including the bottom view of toy animals because it is just a flattened base, the side view of mugs because the handle has been hidden behind the body,

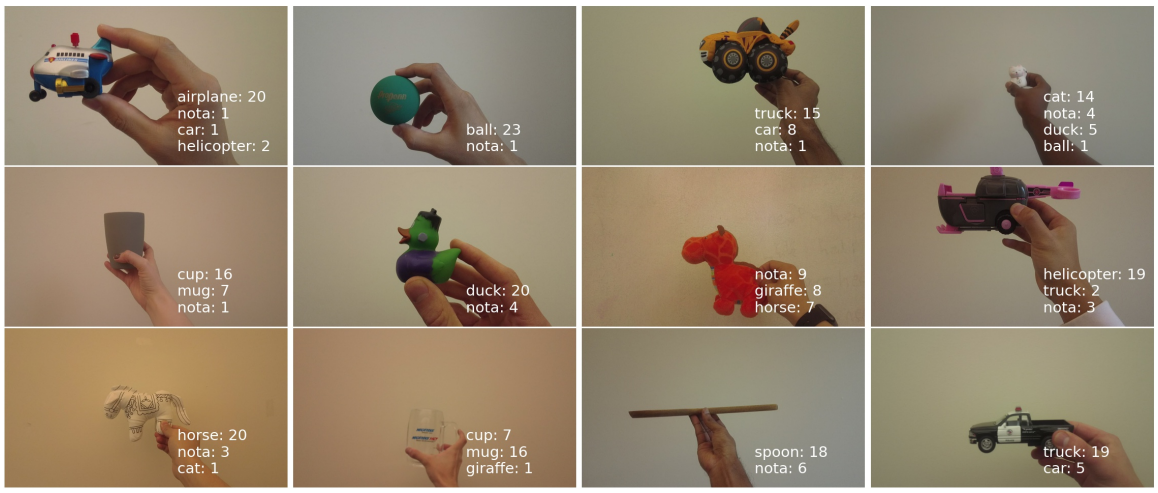


Figure 4.6: The objects that got most misclassified for human recognition. The white text at the bottom right annotates the recognition results (nota meaning none of the above.)

Figure 4.7: The viewpoints that no one recognized correctly, out of three crowdsourcing workers. The white text at the bottom right annotates the true label and the majority vote of human labels (ties broken at random.)

and the side view of spoons because the spoon becomes so tiny.

This ends the full resolution object recognition benchmark for the Toybox dataset. We identified some very interesting category-, object-, and viewpoint- level error patterns, and overall humans achieved 0.945949 accuracy.

4.4.2 Experiment 2. Deep learning models vs humans benchmark

In this experiment, we used the rxplus/rxminus and rzplus/rzminus rotation videos in the Toybox dataset, and we also included 4 different zoom-in ratios as in Figure 4.8.

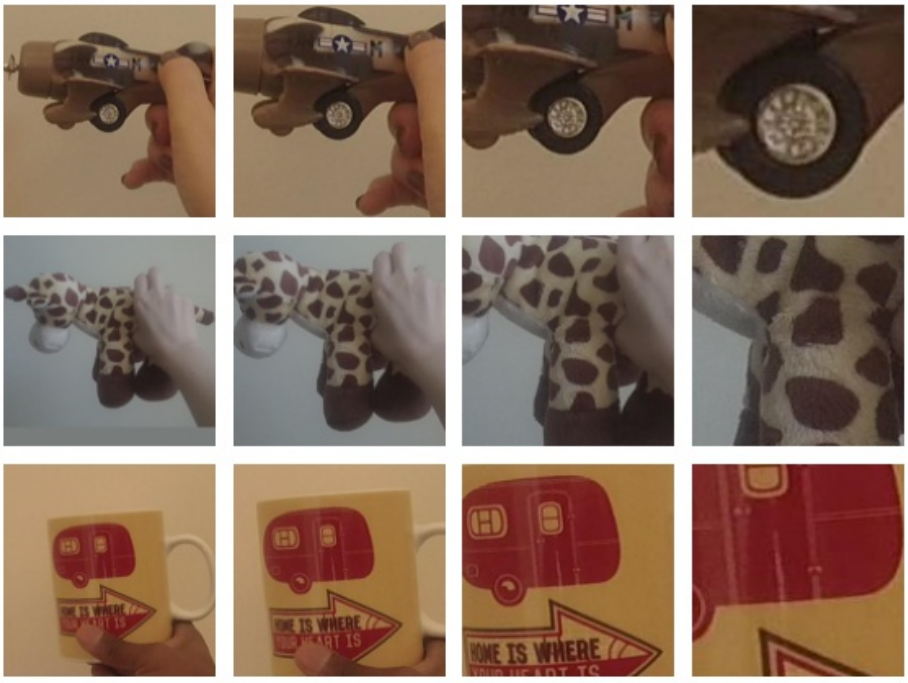


Figure 4.8: 4 different zoom-in ratios in the deep learning models vs humans benchmark of the Toybox dataset. Each row is a different object and each column is a different ratio from 100, 75, 50, to 25.

We published different zoom-in ratios sequentially again to isolate workers between ratios, so that each batch of an individual ratio will have 12 categories $\times$ 3 testing objects per category $\times$ 4 rotations per object $\times$ 18 viewpoints per rotation $=$ 2592 total images.

Toybox has 30 objects per category in total, so we carefully chose 3 objects per category as the testing dataset to cover object diversity. The other 27 objects are the training dataset to train deep learning models.

The networks in our experiments are classic architectures that were on the leader board for the ImageNet [16] benchmark: ResNet-50 [152], ResNeXT-50 [153], Inception-v3 [147], DenseNet-121 [154], and VGG-13 [155].

The hyper-parameters are all the same: batch size 64, epoch 100, learning rate $5e^{-5}$ with 0.5 times decay every 10 epochs, and Adam [156] optimizer with betas $= (0.9, 0.999)$ and eps $= 1e^{-}8$. Normalization was applied to input images but no other data augmentation was performed. Since some networks require a larger input, we resized the images

from 128×128 to 299×299 using bicubic interpolation. Also we included both the non-pretrained and ImageNet-pretrained version for each network.

Therefore, the deep learning models vs humans benchmark finally consists of 2 rotations (rxplus/rxminus and rzplus/rxminus separately) $\times$ 4 ratios $=$ 8 subsets of experiments. We use rxplus/rxminus rotation to illustrate the general findings and then give a summary for both at the end.

Model level analysis. Figure 4.9 shows the accuracy comparison for non-pretrained neural networks, pretrained neural networks, and humans.

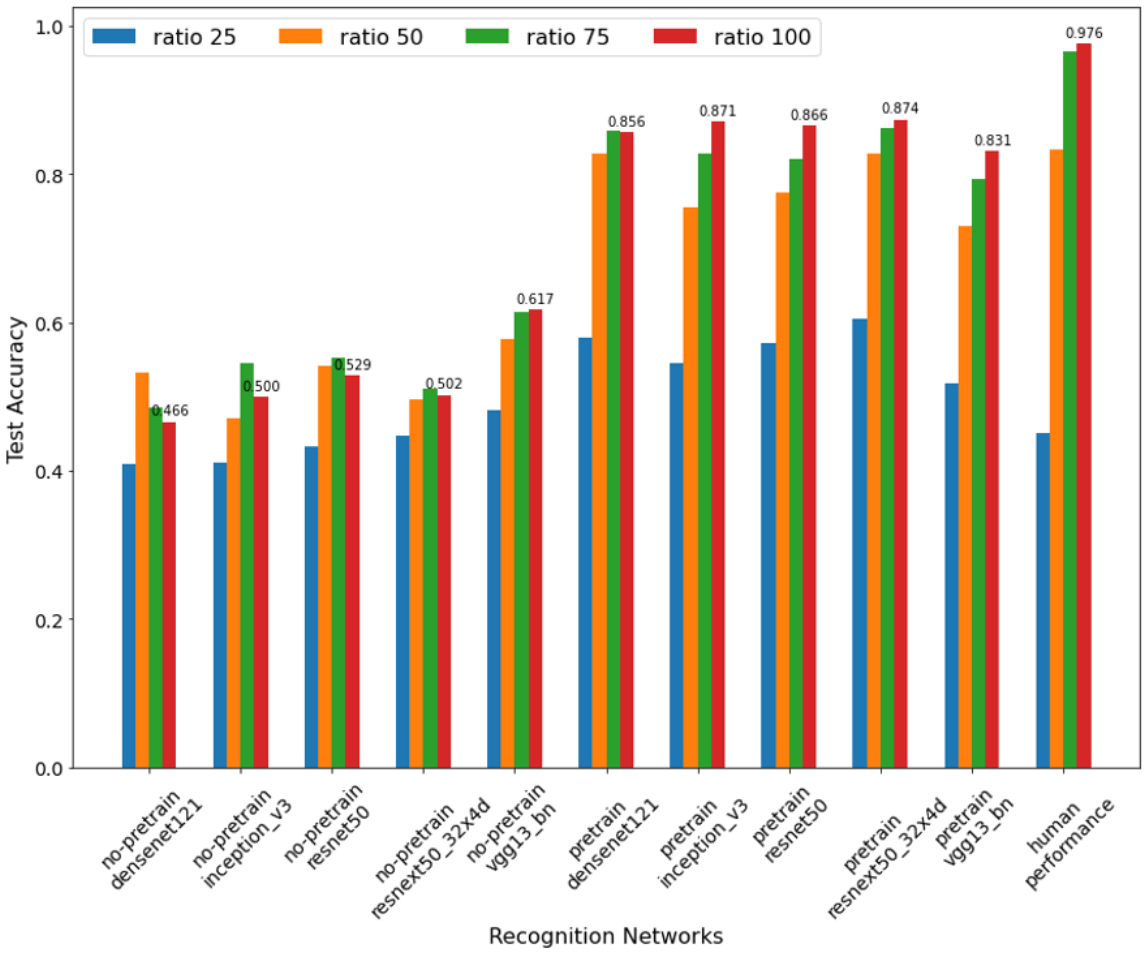


Figure 4.9: The deep learning models vs humans benchmark on the rx rotation images in the Toybox dataset. The y-axis is accuracy. The x-axis is the recognition model where the first 5 are non-pretrained models, the middle 5 are pretrained models, and the last one is humans. Color encodes the zoom-ratios as shown in the legend.

Some key observations from this result.

Training from scratch on Toybox is a challenging task for these classic deep learning models, suggesting that the way how infants may learn from this world, i.e. with egocentric, instance-limited, and viewpoint-rich inputs, is not a trivial learning environment for deep learning models, i.e., Toybox, as a dataset having developmental properties, is **a challenging dataset for current deep learning models.**

For ratio 100, pretraining on ImageNet helped a lot for the object recognition task in Toybox. Some explanations could be that 1) the kernel quality pre-built from the large-scale datasets are much higher than training from scratch as shown in Figure 4.10 and 2) the large-scale setup of ImageNet ($\sim$1000 categories and $\sim$1000 objects per category) brings more variances in the category level and object level that could be more helpful to train deep models than the variance mainly in the viewpoint level.

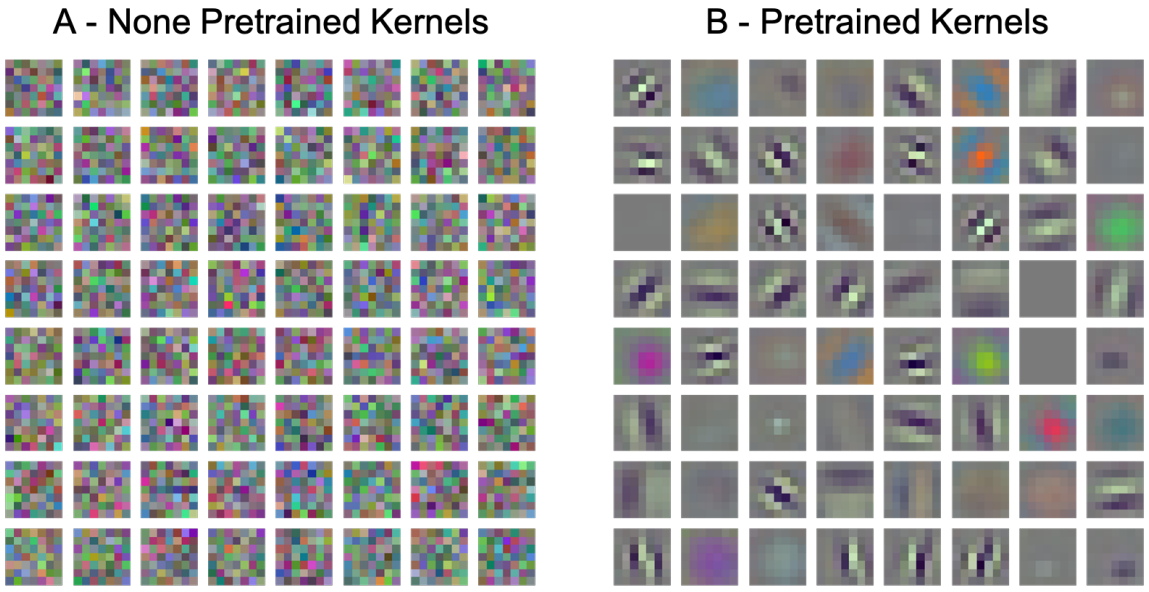


Figure 4.10: The visualization of the first layer convolutional kernels. Figure A shows the kernels from the non-pretrained ResNeXt50 and Figure B shows the kernels from the pretrained ResNext50.

However, even though the ImageNet-pretrained models improve the performance a lot comparing to the non-pretrained models, the deep learning models performance is still at least 10 percent below the human performance, and importantly, pretraining is not the

developmental trajectory we are looking for. Specifically, the developmental trajectory should be similar to how infants are learning object recognition. They first get trained on dataset like Toybox from densely sampled egocentric viewpoints of a few objects [118, 4], and then learn new objects in a nearly one-shot learning fashion [157, 158, 159], rather than first get trained on large-scale datasets with high category and object variance and then get trained and tested on a few objects with extensive viewpoints.

If we compare the performance between deep learning models and humans for zoom-in ratio 25 where the texture features are the main visual clues and zoom-in ratio 100 where the shape features are the main visual clues, we could observe an important performance difference between them. On the texture side, all the pretrained models and even some of the non-pretrained models outperform humans, for example, the best pretrained model ResNeXt50 achieved accuracy 0.604 and humans just achieved accuracy 0.451. But once moving to the shape side, the recognition accuracy of humans increases dramatically from 0.451 to 0.976 (0.525 improvement) but the accuracy of the pretrained ResNeXt50 just increases from 0.604 to 0.874 (0.274 improvement), so that the performance of humans is biased on the shape side but the performance of deep learning models is biased on the texture side.

Some literature has investigated this scenario as the shape bias for humans [160, 161, 162] and the texture bias for deep learning models [163]. Geirhos [163] *et al.* found that in the test stage for a full-ImageNet-pretrained and 16-class-ImageNet [164] retrained neural network such as AlexNet [165], when the textures and the shapes are giving conflicting clues for the object recognition, e.g., a cat shape with an elephant texture, the neural networks will prefer to make decision based on the texture information, e.g., recognize the image as an elephant in this case.

Category level analysis. To get into the category level analysis, we chose the best pretrained models, the ResNeXt50 and its non-pretrained version for comparison, and we focus on the zoom-in ratio of 100. At this zoom-in ratio, i.e., when the shape information

is clear to be observed by humans, humans performance dominates the best deep learning models by around 10 percent in accuracy.

Figure 4.11 shows the confusion matrix comparison among the no-pretraining ResNeXt50, the pretrained ResNeXt50, and humans in the category level. Again, all of them show a pattern of confusions for categories mainly inside the same super-categories.

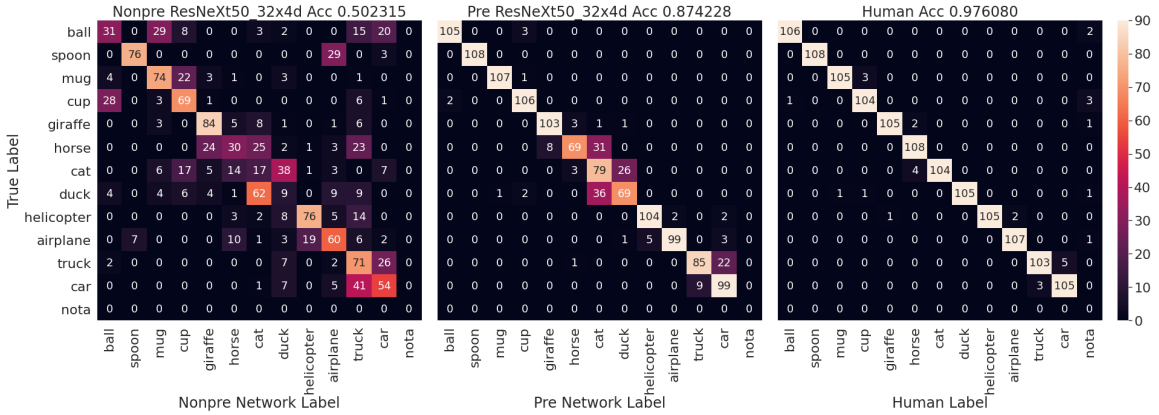


Figure 4.11: The confusion matrix comparison for object recognition at zoom-in ratio 100 among the non-pretrained ResNeXt50, pretrained ResNeXt50, and humans.

Whether the deep learning models and humans make incorrect recognitions for similar reasons will be investigated in the object level analysis below.

Object Level Analysis. Figure 4.12 shows the object level analysis for the same three models again. The non-pretrained ResNeXt gets confused for most of the animals, whereas the pretrained ResNeXt has difficulties on just a few difficult animals, as shown in Figure 4.13. The reasons for these patterns may be that these objects were too large to be captured properly inside the camera field of view, or because some of the objects are quite "cartoony."

We noticed an interesting difference between pretrained ResNeXt50 and humans: the difficult objects seem to be the main reasons for the incorrect recognitions for the pretrained models, whereas humans are much more robust to the difficult objects but tend to make more errors at the viewpoint level.

Viewpoint Level Analysis. To better reveal the viewpoint level patterns, we have also

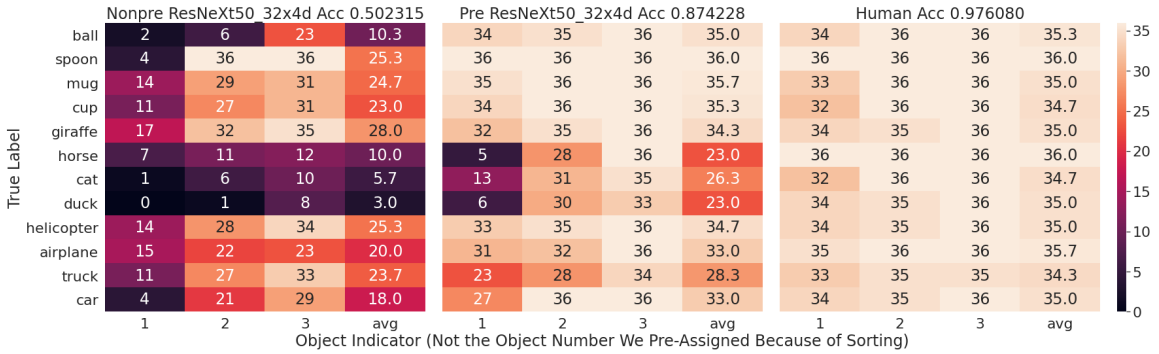


Figure 4.12: The object level analysis for the object recognition at zoom-in ratio 100 among the non-pretrained ResNeXt50, pretrained ResNeXt50, and humans.

Figure 4.13: The animal objects on which the pretrained ResNeXt made a lot of incorrect recognitions.

first removed those difficult objects for deep learning models. Since humans are robust no matter which objects they recognized, we keep all three testing objects for humans in the viewpoint level analysis.

The viewpoints of each image are now manually aligned for 9 settled viewpoints from the two full rotations of the front — top — back — bottom — front viewpoints, and since we have both the rxplus and rxminus rotations, we finally got $9 \times 2 = 18$ viewpoint aligned images per object.

Since the viewpoint is more accurate now, we only have 2 testing objects per category and 2 rotations per object for the viewpoint analysis for the deep learning models, that is, 4 data points per category per viewpoint. Humans have 3 testing objects per category, so we have 6 data points per category per viewpoint as shown in Figure 4.14.

The interesting pattern shown in this figure is that, in the viewpoint level, deep learning

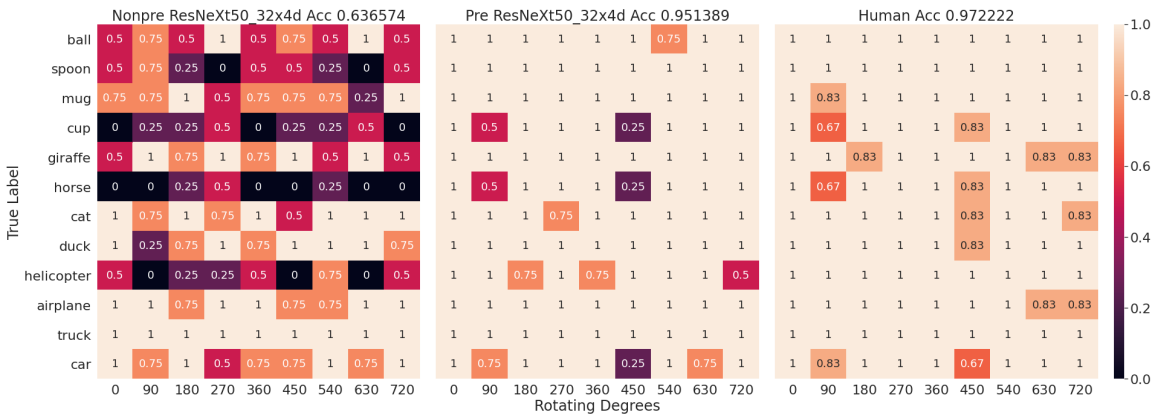


Figure 4.14: Viewpoint analysis for the non-pretrined ResNeXt50, pretrained ResNeXt50 and humans. The y-axis is the true label and the x-axis is the rotating degree. The value in each cell is the test accuracy. Color encodes the accuracy value.

models tend to keep making errors on the same difficult viewpoints, whereas incorrect recognitions that humans made are spread more across different viewpoints.

Figure 4.15 shows examples of the misclassified viewpoints of the non-pretrained ResNeXt50, pretrained ResNeXt50, and humans.

We found the shape vs texture bias again in this viewpoint level analysis, as the pretrained ResNeXt50 recognized a horse with a giraffe-skin saddle as a giraffe, whereas humans recognized a short neck giraffe as a horse.

4.5 Conclusion

In this Chapter, I designed, implemented, and analyzed a benchmark for human performance on the full-resolution Toybox dataset, and also performed a comparison in recognition performance among non-pretrained networks, pretrained networks, and humans, organized by the hierarchical structure of the Toybox dataset. Rz rotations have similar results as rx rotations so we will summarize the findings together in this section.

The benchmark of the object recognition task on the Toybox dataset between classic deep learning models and humans were conducted in different levels, i.e., category, object, and viewpoint. By analyzing the performance differences in these different levels, I found

Figure 4.15: The incorrect recognition examples mainly because of the viewpoint confusions for the non-pretrained ResNeXt50, pretrained ResNeXt50, and humans.

that training from scratch on the Toybox dataset for deep learning models is a non-trivial task, such that these non-pretrained models can just achieve accuracies around 50%-60%. Also difficult objects seem to affect the deep learning models a lot, whereas humans seem to have more confusion for the difficult viewpoints.

This benchmark reveals an important research question on how we could improve deep learning models in a developmental approach, with no pretraining on large-scale dataset and no data augmentation, but just the Toybox dataset itself.

Chapter 5

Utilizing Temporal Consistency to Interpret Neural Networks

5.1 The Research Question

The Toybox dataset has an interesting property of spatiotemporal continuity, i.e., at any timestamp the current frame is just slightly different from the previous frame so that if we plot the activation value changing of each pixel, the curve will form a continuous signal in the temporal domain.

The primary research questions addressed in this chapter are: 1) how can we use this property of spatiotemporal continuity in a dataset to quantify relationships in activation between input neurons and hidden neurons in a neural network? We refer to the existence of these relationships as *temporal consistency* within a network. And 2) how can we utilize this temporal consistency to enhance the interpretability of hidden neurons?

5.2 Introduction

The interpretability of deep neural networks is an important research question from many perspectives such as whether we could trust a black-box model or not, what the reason is a model might make a particular decision, and how to make the decision process transparent for practical domains such as credit review or health diagnosis. In the computer vision domain, we also want interpretability in order to understand what is being learned, and how visual information is extracted and propagated through different layers of neural networks, including how information is gained, lost, or modified along the way.

In addition to the many neural network interpretation methods mentioned in the related work Chapter 2, such as Activation Maximization [120] and Deconvolutional Networks [121, 122], we developed a new approach that relies on examining how the latent neurons inside regular feedforward neural networks change while the networks are getting

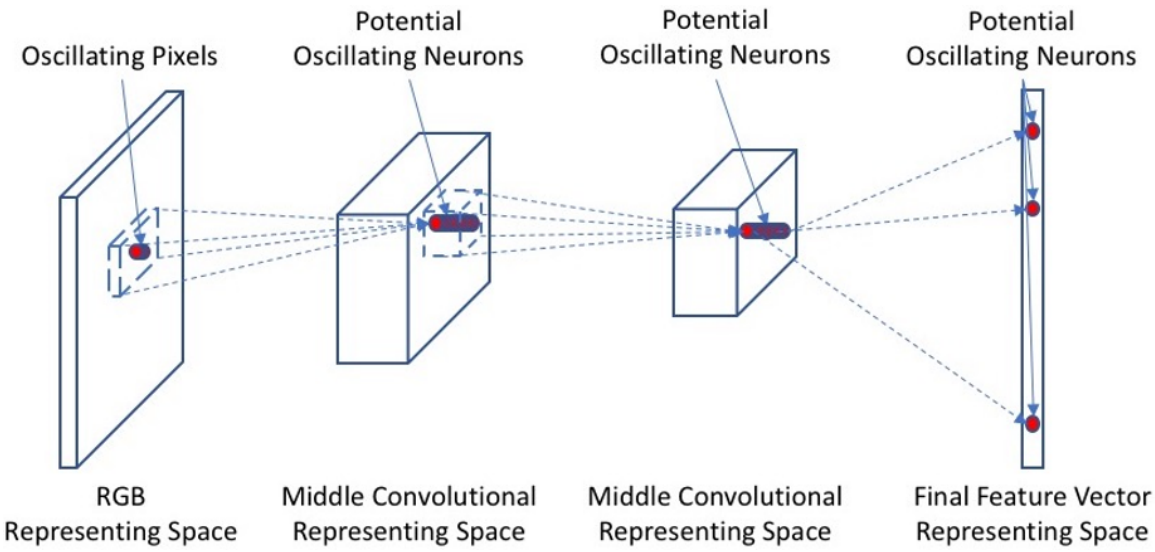


Figure 5.1: Diagram showing how the oscillation of neurons in the input space can be propagated through neural networks to the latent neurons. We refer to this property as temporal consistency: at any timestamp for a static input, the entire network will be activated according to the current input, and so inputs with spatiotemporal structure should also lead to hidden layer activations with corresponding spatiotemporal structure.

continuous inputs. If we can identify relationships between the latent neurons and input neurons in terms of how their activations are changing, we may get some idea about what these latent neurons represent.

This is where the Toybox dataset's spatiotemporal continuity property plays a role, such that we can let a trained neural network watch a continuous video from the Toybox dataset and meanwhile monitor how hidden neurons change in this situation.

Figure 5.1 shows how the input neurons and latent neurons could potentially exhibit temporal consistency. At any static timestep, the latent neurons will be activated according to the activations of the current input neurons, and thus the continuous pattern produced by these input neurons might be propagated to the latent space, such that we might observe some temporal correlations between neurons from different layers.

5.3 Methodology

Our temporal-consistency-based analysis may be applicable to many types of neural networks; however, in the scope of this dissertation, we will mainly focus on convolutional neural networks (CNNs).

To establish spatiotemporal continuity in the input space, we may either manually add continuous perturbations to static datasets or directly use datasets that have intrinsic continuity. Since continuity is an intrinsic property of humans' visual experience and under the scope of this dissertation to bring the naturalistic visual stimuli to the realm of deep learning models, we leverage the Toybox dataset as a tool for conducting our temporal consistency analyses.

Our method is designed as a post-hoc explanatory approach such that the object recognition models have already been trained, and we want to interpret what the models have learned. In comparison, there is another camp to interpret models by making the models self-explanatory ante-hoc. Existing work on post-hoc and ante-hoc interpretability of machine learning models can be found in recent surveys [166, 167]

In our approach, we use continuous inputs such as the rotating Toybox videos to investigate the temporal consistency between the latent neurons and inputs.

Our key insight is this: **the temporal patterns of both the input neurons and latent neurons can be converted to the frequency domain**, which enables analysis via amplitude and frequency comparisons.

Figure 5.2 demonstrates how a neuron in any representing space could be converted to the frequency domain through a fast Fourier transform (FFT). The left figure is a stack of images based on the temporal sequence. The figure in the middle shows how the activation of a single neuron would fluctuate in the temporal domain. The figure on the right shows how the temporal signal could be converted to the frequency domain for further analyses of of amplitude and frequency.

Neurons in different locations within a single layer may present different fluctuating

patterns. For example, if a network is viewing a video of a rotating car, some neurons corresponding to background regions may stay silenced, some neurons in the car's main body may keep firing, and some neurons in the car's front may fluctuate periodically depending on when the front is in view.

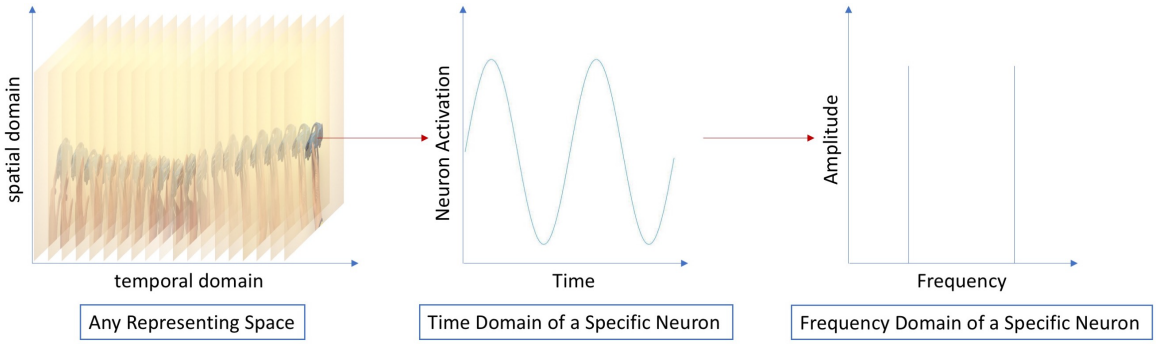


Figure 5.2: The left figure shows how a neuron in a representing space would fluctuate along with the continuity of the inputs. The middle figure shows an example plot for the neuron's activation as a function of time. The right figure shows the frequency pattern of this example neuron.

If some latent neurons have higher amplitudes than others in particular frequencies while watching a specific rotating input, these latent neurons might be more sensitive to the recognition of the current category. In order to further examine such a category-level interpretation, we conducted pruning experiments to evaluate the magnitude of contributions produced by these "highly responsible" neurons towards recognition performance.

In particular, the latent neurons that are identified as the main contributors according to the above procedure are sequentially pruned out based on their oscillating amplitudes. We then test the pruned neural network on the entire testing set to observe how the activations of the output neurons change on objects from different categories. If these neurons are indeed the main contributors for the category of the rotating object we used in the temporal consistency step, the output neuron for the corresponding category should be suppressed significantly.

5.4 Experiments and Results

To examine our temporal consistency methodology for post-hoc neural network interpretation, we retrained an ImageNet-pretrained Inception-v3 [32] on the Toybox dataset. Based on different retraining schemes, the experiments were conducted in two conditions, one with retraining on the full resolution Toybox dataset, such that all the off-white wall backgrounds and hand poses are included, as described in in Section 5.4.1, and the other with retraining on the square cropped Toybox dataset such that the objects are well centered and the occupancy of backgrounds is very limited, as described in Section 5.4.2. The retraining details can be found in each section for these two experiments.

Figure 5.3 shows the layers for which we examine temporal consistency: the input layer, the last hidden layer (which we refer to as the *bottleneck layer* in the rest of this chapter), and the output layer. We expect that our methods could be expanded to investigate other layers of a neural network as well.

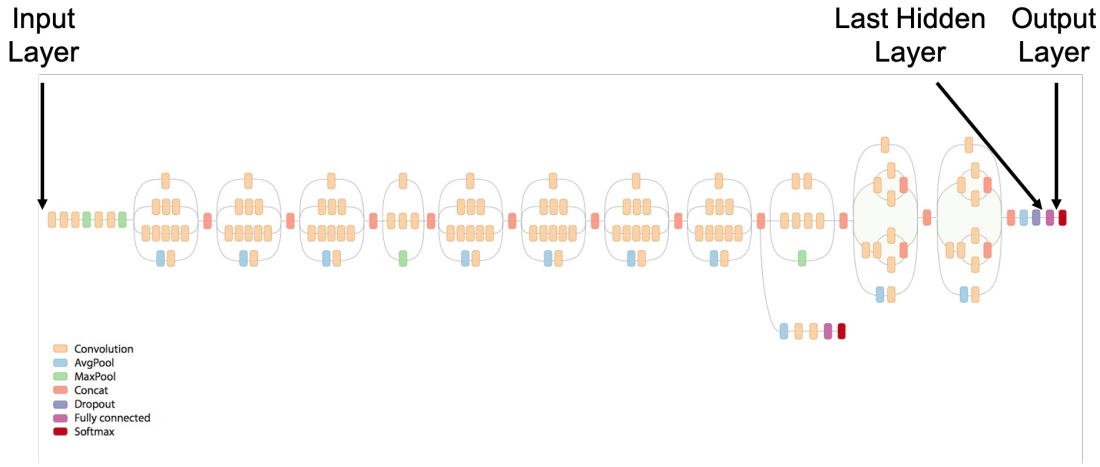


Figure 5.3: The layers of the Inception-v3 network that we focus on in our temporal consistency interpretation experiments.

5.4.1 Utilizing temporal consistency on full resolution Toybox and examining the pruning effect on ImageNet

In this experiment, the final output layer was retrained on full resolution Toybox data with 1100 images per category randomly selected across all Toybox objects and transformations.

For the pruning examination, we collected 100 images per category from ImageNet that are in the same 12 categories as Toybox to test the generalizability of the representations we identified for the bottleneck neurons; that is, we trained Inception-v3 on Toybox but tested on ImageNet.

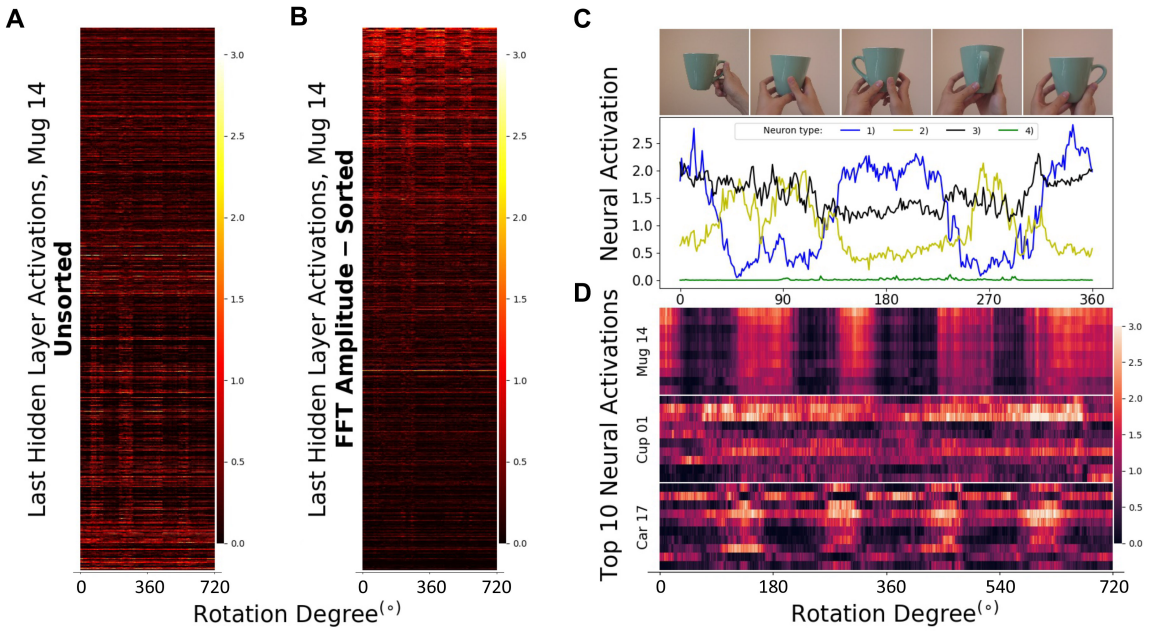


Figure 5.4: A novel method to identify neurons that correlate with a rotating object using our Toybox video dataset. **A**. Temporal raster plot of neural activations in the final hidden layer of the pretrained Inception v3 network while "watching" a rotating mug. Each row shows an individual neuron, and the x-axis depicts time/rotation. **B**. The same neurons sorted based on their FFT amplitude from high (top) to low (bottom). Note the stripe pattern on the top half of the plot showing strong periodicity. **C**. Four different types of neurons identified using this method. **D**. Comparison of FFT analysis of mug, cup, and car, showing top 10 neurons after FFT amplitude sorting.

Quantifying neuron temporal activation profiles. We began by studying the "temporal" activation profiles of neurons in the last hidden layer of the Inception network, while

the network is receiving a sequence of Toybox input images depicting a mug rotating along the z(+) axis for two full cycles. Figure 5.4 shows visualizations of these activations.

In particular, Figure 5.4A depicts the activations over time of all 2048 neurons in the final hidden layer of the network. Each row shows the activations of an individual neuron (unsorted in this subfigure), and the x-axis indicates time, which also approximates the rotation degree of the mug. (This visualization method is adapted from the temporal raster plots used in neural physiology research.) The various neuron "firing patterns" are clearly heterogeneous: some neurons are constantly firing throughout the two rotation cycles, some remain silenced, and some fluctuate as the mug rotates.

To differentiate these neuron types, we applied a Fast Fourier Transformation (FFT) to the activation of each neuron over the two rotation cycles. To capture general viewpoint trends, we focused our FFT analysis on a frequency of 4 (i.e., four cycles within the 20-second long Toybox video that contains two complete rotations). We then sorted the 2048 neurons shown in Figure 5.4A based on their FFT amplitude at the frequency of 4—larger amplitudes indicate more robust oscillations. Figure 5.4B shows a visualization of the same neurons but sorted (top-to-bottom) by their FFT amplitudes.

Since FFT analysis also returns the phase information (positive phase correlates with handle presence, negative phase correlates with handle absent in this case), we were able to identify four different types of neurons based on their activation profiles. Examples of these four types are shown in Figure 5.4C and also in Supplementary Video 2: (1) neurons that fire when the mug handle is present (blue line); (2) neurons that fire when the mug handle is *behind* or *in front of* the mug body (yellow line); (3) neurons that fire throughout the video clip (black line); and (4) neurons that do not fire at all (green line, these neurons presumably do not contribute to the representation of the mug).

We also tested this FFT analysis method on objects from other categories using our Toybox videos. As shown in Figure 5.4D, the ability to identify robust oscillating neurons mainly depends on the degree of asymmetry of the object along the z-axis. For instance,

we were able to identify neurons with more robust oscillation for a mug and a car than for a cup (which is symmetric along the z-axis).

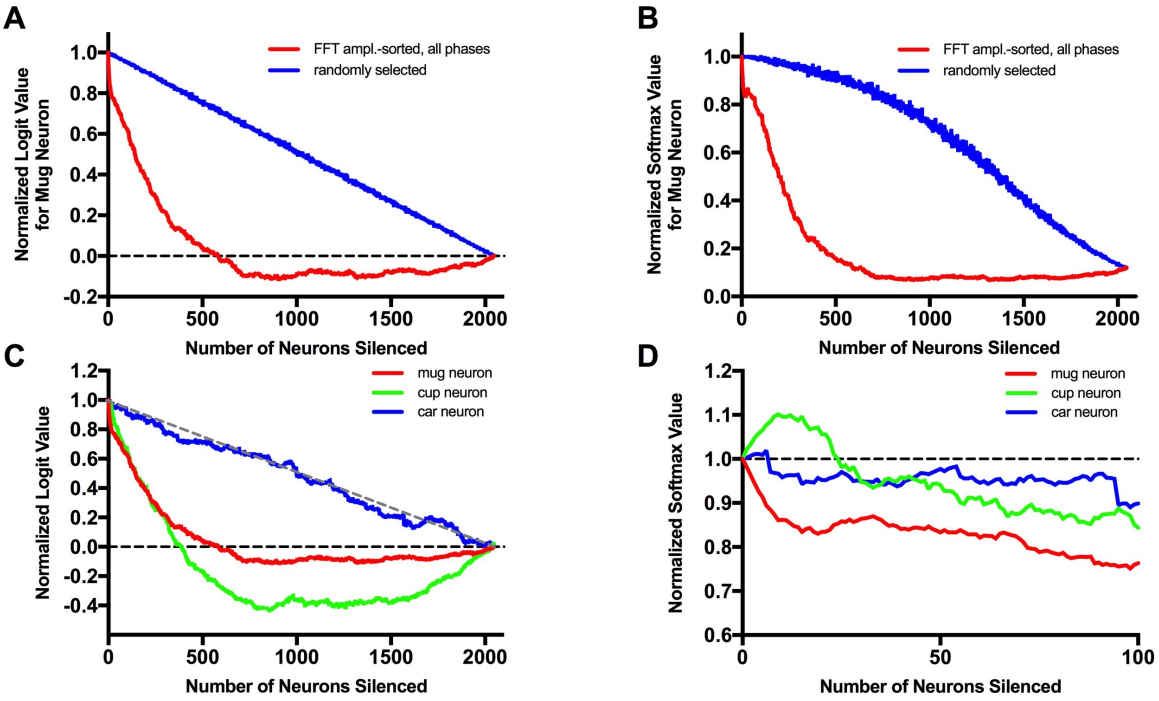


Figure 5.5: Effects of silencing hidden layer neurons on output layer activations, averaged over 100 images per category from ImageNet-sourced testing set. **A**. Silencing top N neurons based on FFT amplitude sorting leads to a much steeper reduction in normalized logit value of the mug output neuron. The blue line shows the reduction rate of silencing N randomly selected neurons as a control. **B**. Similar to A, showing softmax values instead of logit values. **C**. Silencing *mug-preferred neurons* (MPNs) has a similar effect on the logit value of the cup output neuron but has no effect on that of the car output neuron. **D**. Zoomed-in plot of softmax values, showing that silencing the top $\sim$20 neurons decreases mug prediction confidence while increasing cup prediction confidence, consistent with the fact that the majority of these neurons correlate with the presence of the mug handle.

Effects of neuron silencing. To investigate the implicit representations of the various types of neurons identified above, we performed a neuron silencing/lesion experiment by selectively "zeroing out" the activations of certain neurons in the last hidden layer, and then observing effects on recognition performance. Figure 5.5 shows results from these experiments, where the logit and softmax values for particular output neurons are shown as averages over 100 images from various categories in the ImageNet-based testing set.

First, for testing images from the mug category, we silenced N neurons (N varying from

0 to 2048) in the last hidden layer and examined the changes of both logit and softmax values of the mug neuron in the output layer. As shown in Figure 5.5A, silencing 0 of these hidden layer neurons has no effect, while silencing all 2048 neurons reduces the normalized logit value of the mug neuron to 0. Randomly silencing a subset of N neurons leads to a linear reduction of the logit value with respect to N. However, if we silence neurons based on the FFT amplitude sorting as shown in Figure 5.4B (i.e., first silencing the neuron with the highest FFT amplitude, then the top two, then top three, and so on), we observed a much steeper drop in mug logit value at the beginning. After $\sim$700 neurons, silencing has no more reduction effect on the mug logit value. Similar effects can be seen with the softmax value of the mug neuron (Figure 5.5B). In a sense, by selecting neurons with highest FFT amplitude, we can identify what we call *mug-preferred neurons* (MPNs).

To examine the specificity of these MPNs, we tested the silencing effect on cup and car output neurons. Silencing the top MPNs has a significant impact on the logit value of the cup neuron (Figure 5.5C). This is not surprising given that a mug and a cup share many common features. However, the zoomed in softmax plot shows that silencing the top $\sim$20 MPNs slightly increases the softmax value of the cup output neuron, which is consistent with the fact that most of these neurons fire when the handle is present (Figure 5.5D and Supplementary Video 2). In other words, these neurons might be contributing to the difference between a mug and a cup.

In contrast, silencing the top MPNs has almost no effect on the car output neuron (Figure 5.5C). In fact, the effect of silencing MPNs is almost identical to that of silencing random neurons (dotted grey line in 5.5C, see also 5.5A blue line).

These experiments showed that the MPNs contribute significantly to mug identity and much less to the identity of other categories like car. A small portion of the MPNs may be coding the handle feature to differentiate a mug from a cup. Interestingly, although silencing one or a few neurons that are most prominent does decrease the input value to a specific output neuron, there is a significant amount of recognition value that remains.

This result confirms that object features do not seem to be represented by a single or few neurons, but rather by an ensemble of neurons.

5.4.2 Utilizing temporal consistency and examining the pruning effect on square cropped Toybox

The inputs are now the square cropped images of Toybox in the rxplus/rxminus, rzplus/rzminus videos (ryplus/ryminus is an in-plane rotation in which the viewpoint doesn't change very much). The definition of these rotations can be found in the rotation terminology Section 3.5.5 in the Toybox dataset Chapter 3). The cropped images have the resolution $299 \times 299 \times 3$. We will again use the abbreviations of rx for rxplus/rxminus rotations and rz for rzplus/rzminus rotations in the Toybox dataset in the rest of this chapter.

For this experiment, we retrained ImageNet-pretrained Inception-v3 networks, one using the rx rotations only and the other one using the rz rotations only, so these two networks should have different viewpoint sensitivity. For each network, the training dataset contains 12 categories $\times$ 27 objects per category $\times$ 2 rotations per object $\times$ 18 viewpoints per rotation $=$ 11664 images in total. To test the bottleneck pruning effect, we will use the remaining 3 objects with the same data setup.

The rx retrained network achieved 0.871142 accuracy on the testing dataset and the rz retrained network achieved 0.859568 accuracy on the testing dataset, so we believed the quality of the latent representations is good enough to support our experiment.

We first use the rz network as an example to illustrate the interpretation process, and then we show the interpretation results in a more holistic view for both neural networks.

The Temporal Consistency Between Input and Latent Neurons. We chose the amplitude at frequency of 4 as the main index to sort latent neurons while they are watching rotation videos. A frequency of 4 means that some features appeared 4 times in the entire video. Recall that Toybox has 2 revolutions per video, so this frequency points to those features repeated every 180 degrees. Because of the symmetry of many of the objects in

Toybox, if a feature only exist at one side of the object, it will have frequency of 4, for example, the tail of a horse, the head of a car, or the handle of a mug. For example, mug handles are most in view in "profile," and then handle appears once on either side of a mug during a single rotation.

To identify which input neurons have higher amplitudes at a frequency of 4, we converted the RGB images into gray images by averaging three channels together, so a unique pixel position will only have one value. The input size is $299 \times 299 \times 1$, and so this finally becomes 89401 neurons compared to the 2048 neurons in the bottleneck layer.

Figure 5.6 shows the temporal consistency analysis of an rzplus rotating mug. The handle of the mug and the hands are highlighted in the heatmap. Two observations arise from this figure. First, the hand pose may contribute to the category prediction as well as the object itself. Second, the top 10 latent neurons at a frequency of 4 do not have frequency of 4 as the maximum, even though they already had higher amplitudes than other neurons. This suggests that these top 10 latent neurons may be responsible for features in other frequencies in the input space at the same time.

Pruning Effects.

Figure 5.7 shows the pruning effects while we are pruning bottleneck neurons according to the amplitude descending sequence of all the mug objects.

Mathematically, we showed all 27 rzplus rotating mug objects in the training dataset to the neural network, so that we had a matrix of $27 \times 18 \times 2048$ to record the bottleneck neuron activations of 27 objects, 18 viewpoints, and 2048 neurons. Then, we averaged these activations over the object axis so we would have a matrix of 18×2048 to indicate the average activation in the mug category. Next, we converted each neuron to the frequency domain which still gave us an 18×2048 matrix; however, each row now represents the amplitude at that row's frequency now.

We then picked the 4th row for frequency of 4 which gave us a 1×2048 matrix, and now each element in this matrix represented the amplitude at frequency of 4 for the neuron

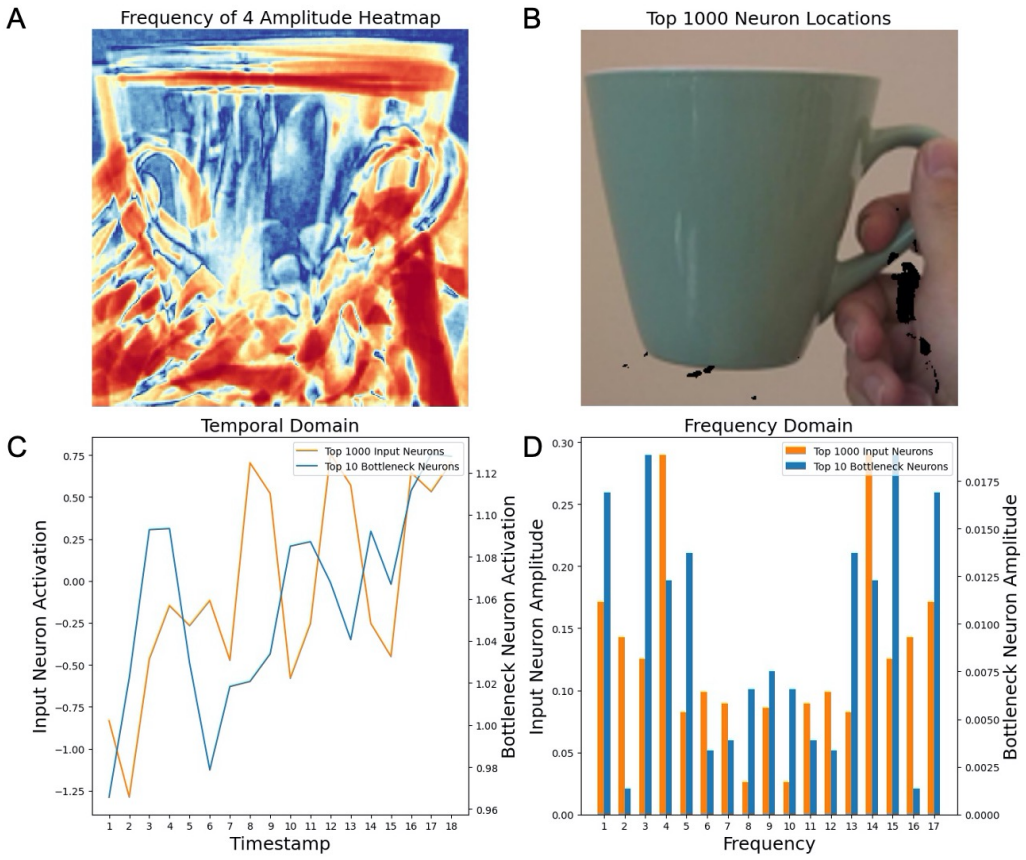


Figure 5.6: A figure to demonstrate the temporal consistency analysis. Figure A shows the amplitude heatmap of frequency of 4 for a rotating mug. Regarding frequency of 4, Figure B shows the location of the top 1000 neurons; Figure C shows the average activation of the top 1000 input neurons and top 10 bottleneck neurons in the temporal domain; and Figure D shows the same top neurons in the frequency domain.

in its column. Finally, we sorted these 2048 neurons according to the amplitude value in this 1×2048 matrix.

The pruning then happens while the neural network is recognizing objects in the testing dataset. After pruning bottleneck neurons in this mug sequence, we found that the effect was mainly applied to the mug output neuron. If you scan the 12 sub-figures in Figure 5.7, while we are testing other categories, the neuron activation will just decrease linearly, similar to just pruning neurons in a random sequence as in Figure 5.8, but for the mug category, it decreases exponentially.

This indicates that the neurons we identified using our FFT analysis indeed contribute

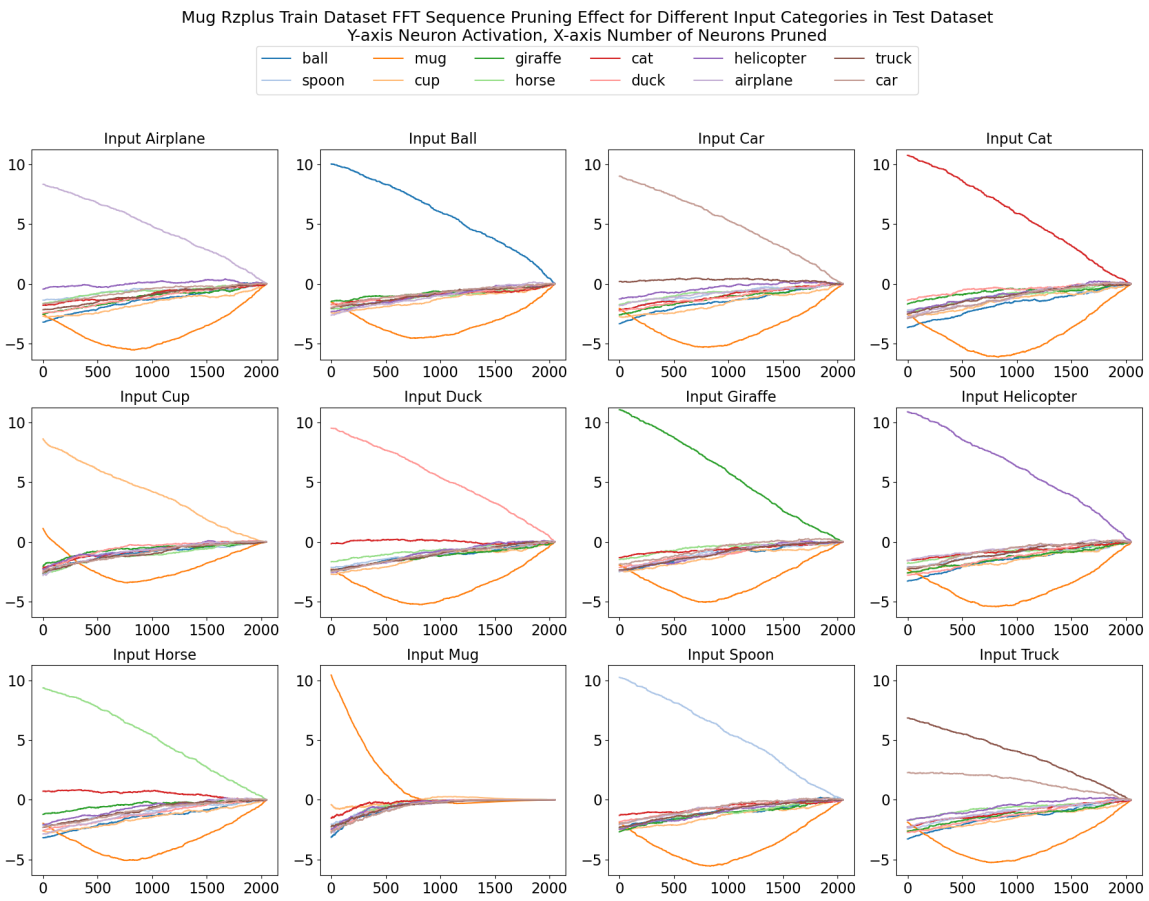


Figure 5.7: The effect of pruning bottleneck neurons according to the amplitude descending sequence while the neural network is watching all the rzplus rotating objects in the mug category. The pruning sequence is the same across sub-figures but different sub-figures get inputs from different categories. The y-axis is the output neuron activation and the x-axis is the number of neurons pruned.

significantly to the recognition of mugs in comparison to other categories.

The Holistic View of the Bottleneck Neurons.

Figure 5.9 shows the holistic view of the amplitude sequence in each category for all bottleneck neurons. We noticed that the main recognition contributions seem from a small group of neurons, as those neurons on the left end are constantly oscillating across categories.

Therefore, to verify the contribution identified by our temporal consistency method, we tried to only use those top-n neurons to make predictions, and the results in Figure 5.10

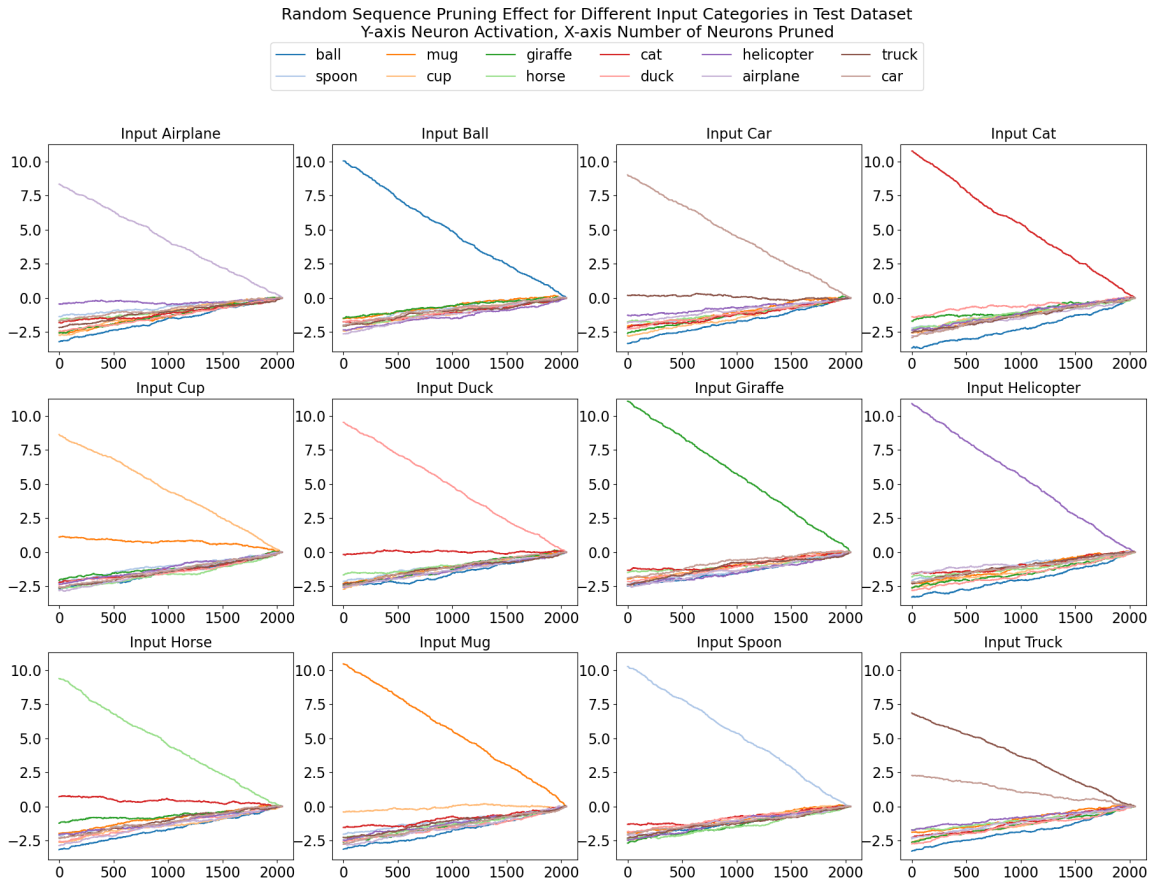


Figure 5.8: The effect of pruning bottleneck neurons according to some random sequence. Each sub-figure is for inputs from different categories. The y-axis is the output neuron activation and the x-axis is the number of neurons pruned.

further supported our interpretations.

For the rz rotation experiment, only using the top 16 neurons in each category with 190 neurons in total (the total is not $12 \times 16 = 192$ because that one neuron may contribute to multiple categories as illustrated above), we achieved 0.8735 accuracy which is even higher than the usage of all neurons. This small group of neurons dominated the recognition activation under the current model and dataset environment. Similarly, for the rx rotation experiment, only using the top 65 neurons in each category with total 646 neurons can also achieve 0.8657 accuracy.

5.5 Conclusion

We showed how temporal consistency can be quantified in neural networks by leveraging the spatiotemporal properties in the Toybox dataset, and how these observed relationships can be used to help interpret category-level representations in the latent (hidden layer neuron) space.

The neurons identified by our methods are a small group of neurons that dominate object recognition for the category they represent for, i.e., the main contributors for the object recognition of the category that they are primarily responsible for in the temporal consistency analysis.

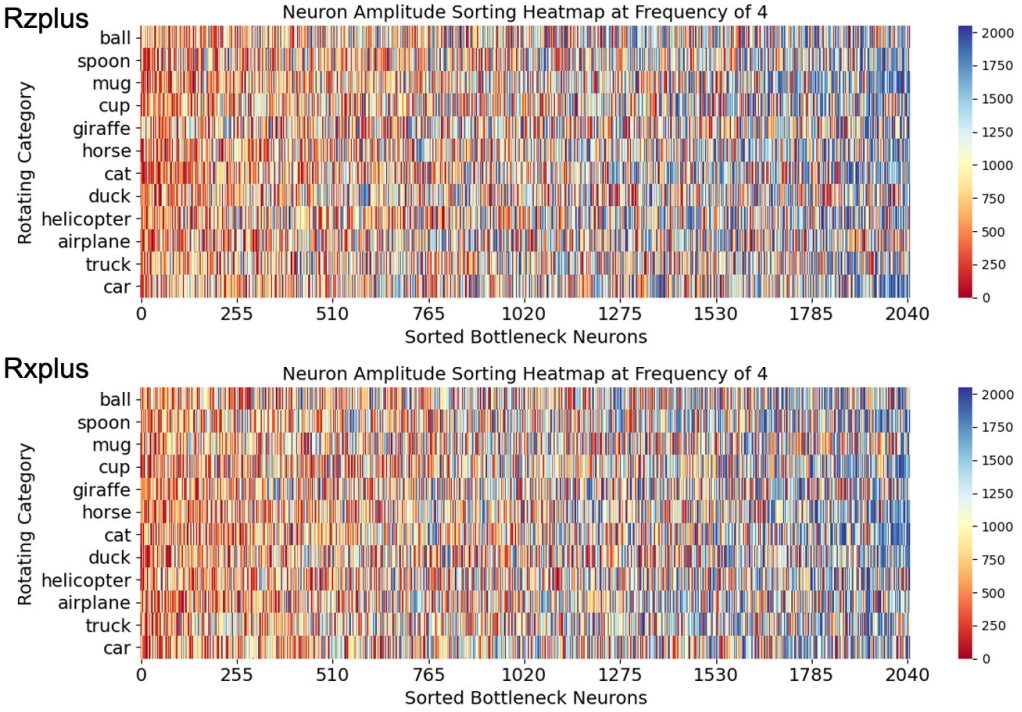


Figure 5.9: The top figure is for rzplus rotation amplitude sequence and the bottom figure is for the rxplus rotating amplitude sequence. The y-axis is the category and the x-axis is the bottleneck neuron sorted by the average amplitude across all categories by the amplitude at frequency of 4 for all rotating objects in each category in the bottleneck layer. The color encodes the sequence.

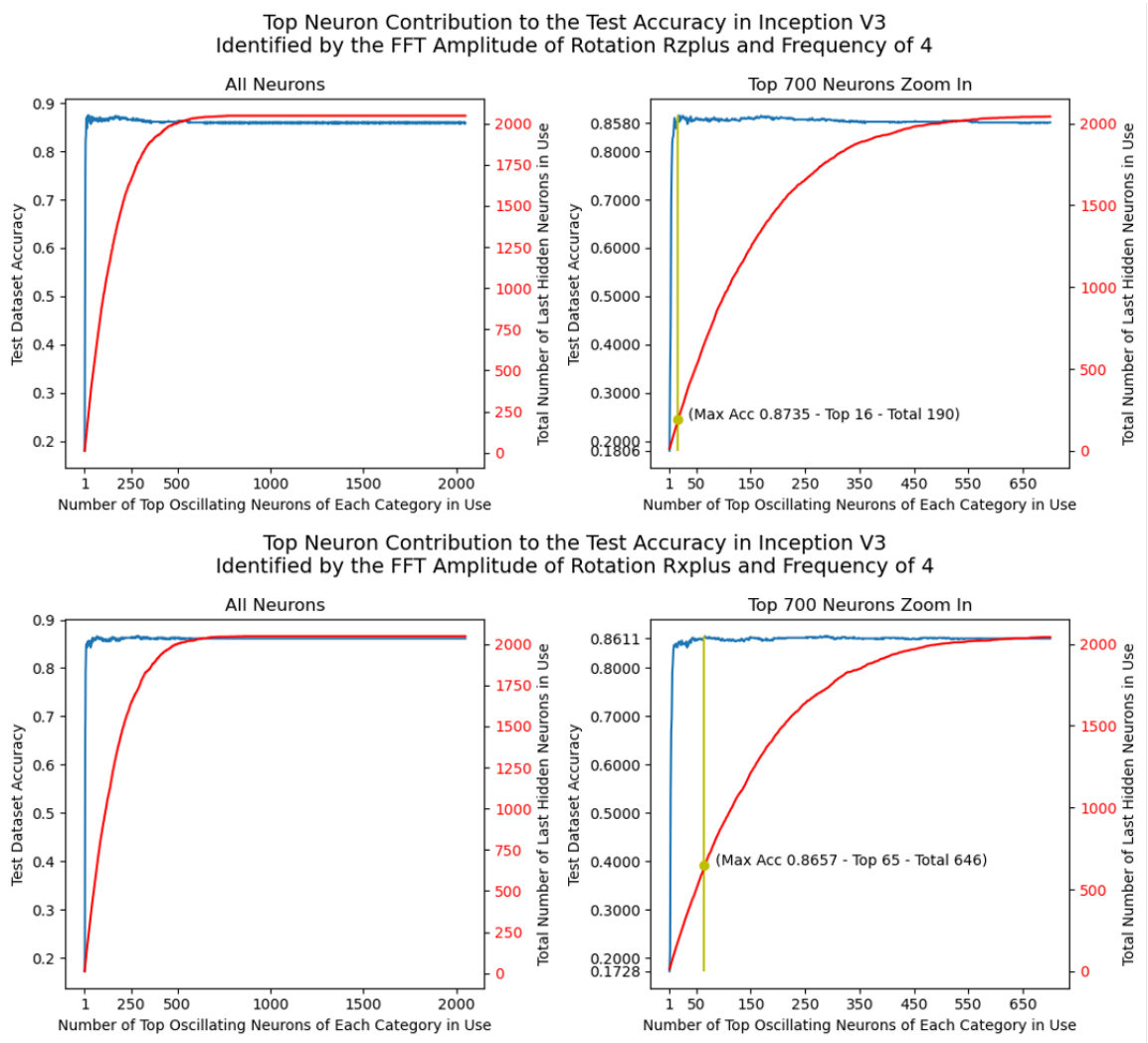


Figure 5.10: The top figure is for rzplus rotation amplitude sequence and the bottom figure is for the rxplus rotating amplitude sequence. In each amplitude sequence, the left and right figures are from the same results where the right figure is a top 700 neurons zoom in. The left y-axis is the testing dataset accuracy, the right y-axis is the total number of neurons in use, and the x-axis is the number of top neurons in each category we used for prediction. The blue curve is the accuracy curve and the red curve is the total number of neurons curve. The yellow bar indicates where the network achieved the highest accuracy.

Chapter 6

Variable-Viewpoint Representations for 3D Object Recognition

6.1 The Research Question

In the introduction chapter we have discussed the importance of the viewpoint-rich representations generated from the projecting process, and thus, this chapter will investigate the viewpoint-rich representations from three perspectives:

1. How can we design a unified framework that covers a continuous range of representations having different distributions of viewpoints?

2. How well do deep learning models designed for one type of representation perform across the range of representations in our framework?

3. To what extent does fusing information across the range of representations in our framework improve object recognition performance?

To address these questions, we will first demonstrate the design and implementation of the Variable-Viewpoint (V^2) Representations, a viewpoint-rich projecting framework, and show how we could utilize these representations for the 3D object recognition task.

6.2 Abstract

For the problem of 3D object recognition, researchers using deep learning methods have developed several very different input representations, including "multi-view" image snapshots taken from discrete viewpoints around an object, as well as "spherical" representations consisting of a dense map of ray-traced samples of the object from all directions. These representations offer trade-offs in terms of what object information is captured and to what degree of detail it is captured, but it is not clear how to measure these information

trade-offs since the two types of representations are so different. We demonstrate that both types of representations in fact exist at two extremes of a common representational continuum, essentially choosing to prioritize either the number of views of an object or the pixels (i.e., field of view) allotted per view. We identify interesting intermediate representations that lie in between these two extremes, and we show, through systematic empirical experiments, how accuracy varies along this continuum as a function of input information as well as the particular deep learning architecture that is used. Finally, we propose a multi-representation network that utilizes several of these representations together and achieves strong performance without hyperparameter tuning.

6.3 Introduction

Imagine you are visiting a museum to photograph a historical artifact. Because of the limited disk size of your camera, you can only store a total of 1920×1080 pixels. Every view of this artifact is appealing, and so you face a difficult choice: will you take a single high resolution picture ($1 \times 1080p$) of the front-view only, several medium resolution pictures ($6 \times 480p$) of a few more views, or an omnidirectional picture with as many views as possible? Which approach would help you remember the artifact best?

This question exemplifies a key point of differentiation among current, high-performing approaches to 3D object recognition. Multi-view representations [6, 7] essentially capture a discrete collection of whole-object views from various viewpoints. Spherical representations [10, 15, 11] capture a continuous sampling of single-point views from all around the object. (We focus here on approaches to 3D object recognition using inputs that are projections of 3D information onto 2D image arrays. Other approaches use inputs represented explicitly in 3D, such as voxels [100, 33] and point-clouds [168, 93], but these fall outside the scope of our discussions in this paper.)

Which representation is better, multi-view or spherical? Both yield impressive results with various deep learning architectures, but it is difficult to make generalizable, apples-

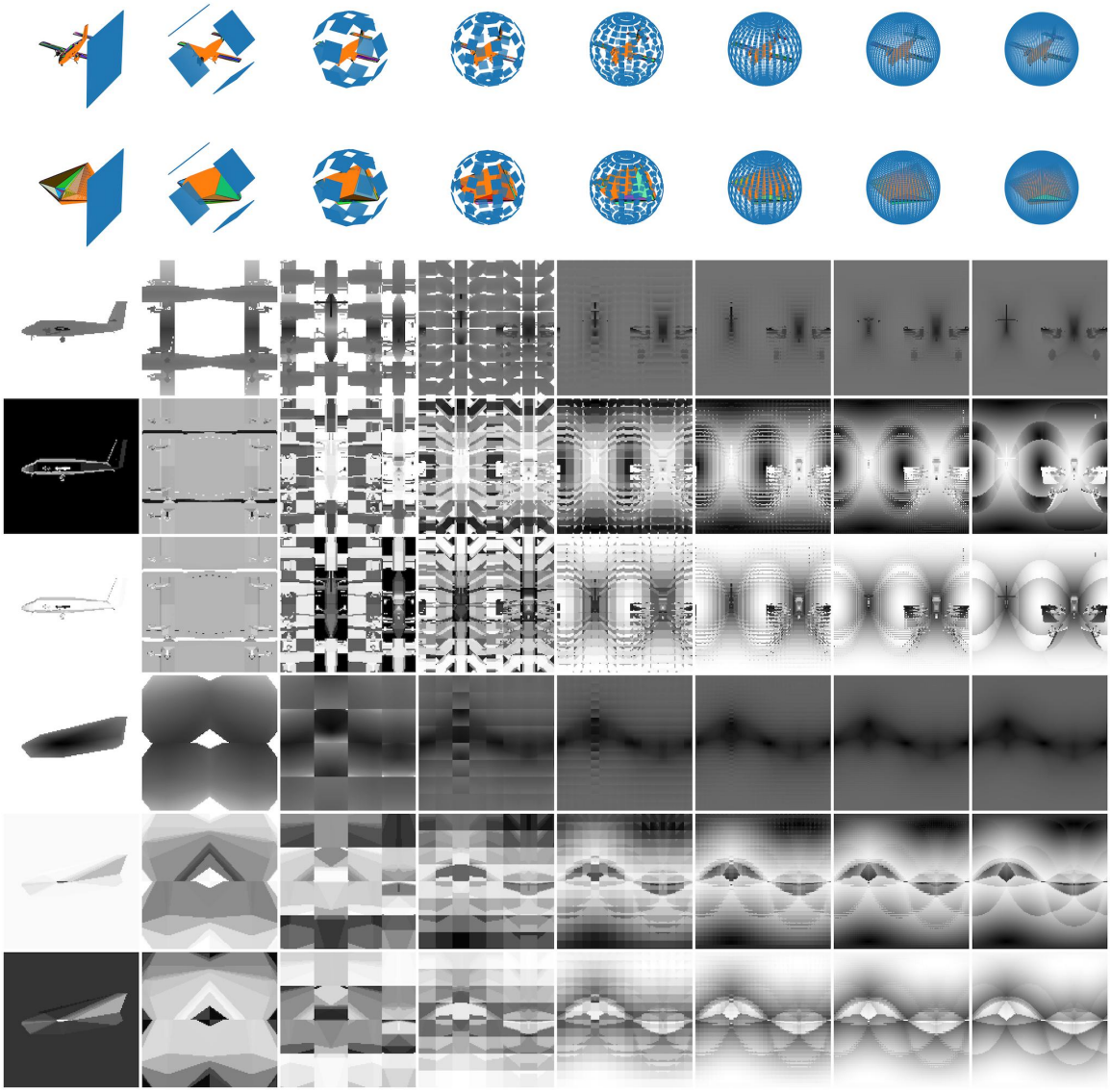


Figure 6.1: The continuum of our Variable Viewpoint (V^2) representations from the Multi-View extreme (left) to the Spherical extreme (right). The first two rows show the sampling points for the mesh and its convex hull. The rest of the rows show the Depth, Sine, and Cosine channels (details in Section 6.4.1) of the mesh and its convex hull. All representations are from the same 3D object: an airplane in ModelNet40.

to-apples judgments between them when they draw from such different types of input information. And furthermore, are these two our only options for projecting a 3D object into 2D images?

- In this paper, we demonstrate that **multi-view and spherical representations are**

essentially two extreme cases of a unified representational continuum, which we call the Variable Viewpoint (V^2) representational framework, illustrated in Figure 6.1 and described in detail in Section 6.4.1.

- We show, through systematic empirical experiments, how 3D object recognition accuracy varies along this V^2 continuum as a function of the input information available to the learner, and we find that, even with architectures specifically developed for either multi-view or spherical representations, **peak performance occurs in middle regions of our V^2 representational continuum**.
- We present the Multi-Representation Convolutional Neural Network (MRCNN) that combines multiple representations across the V^2 continuum, and we show that **by fusing the representations in the V^2 continuum together, we can achieve strong performance** without hyperparameter tuning.

6.4 Variable Viewpoint Representations

We define *variable viewpoint (V^2)* representations as a collection of samples of 2D projections from around a 3D object, specified by following parameters:

1. The **number** of views (e.g., akin to the number of cameras positioned around an object)
2. The **position** of views (where each camera is located)
3. The **size** of each view (field of view of each camera)
4. The **density** of each view (resolution of each camera)

The number and the position determine the view distribution, and the size and the density determine the view field and resolution. If the view distribution is sparse and the resolution is high, V^2 is equivalent to multi-view representations. If the distribution is dense and the resolution is only one pixel, V^2 is equivalent to spherical representations.

V^2 representations are generated by first centering and normalizing a given 3D object inside a sphere. Then, each sampling point on the sphere becomes the center point of

a view plane that is tangent to the sphere. On each view plane, sampling points form a evenly spaced grid surrounding the center point. All points on a plane will shoot rays towards the object that are perpendicular to the plane, finally reaching the object (or in some cases missing it entirely, becoming a "background" pixel in the resulting projection). Figure 6.2 shows intuitive examples of different generating configurations of V^2. Parameters are explained in Section 6.4.1

The information obtained from each ray when it intersects the object populates the channels of the 2D projection. For example, this information could consist of the traveling distance, the RGB value on the surface, and/or the incident angle. We denote the Number

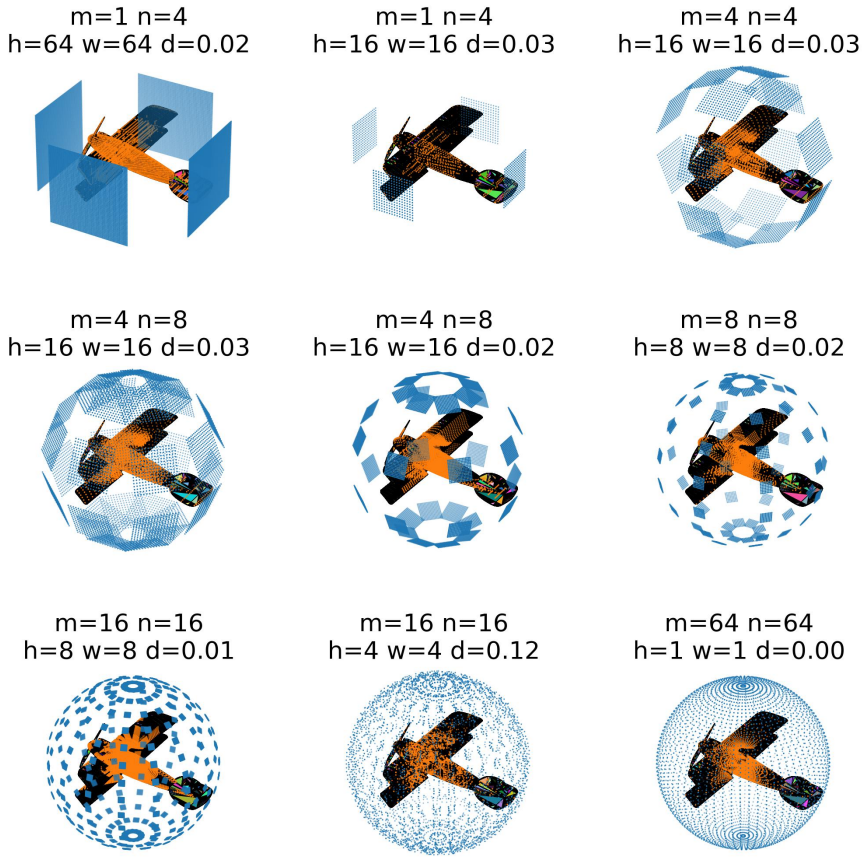


Figure 6.2: Samples of V^2-generating configurations. Blue dots are the sampling points shooting rays to the object. Orange dots are the intersected points with the object.

of Channels as *NC*.

6.4.1 V^2 specification for this paper

There would be many ways to define numerical parameters to capture the V^2 framework. In this paper, we define five specific parameters:

1. m = number of rows in view distribution (e.g. latitude lines)
2. n = number of columns in view distribution (e.g. longitude lines)
3. w = width of each view, in pixels
4. h = height of each view, in pixels
5. d = density of pixels in a view, i.e., distance between sampling rays in a view place

For the studies in this paper, each ray captures 6 channels of information (i.e., $NC = 6$: the depth (traveling distance) from the origin to the object mesh, the sine and cosine of the incident angle between the ray and the surface of the mesh, and the same three channels for the convex hull of the object. Of course, the V^2 framework could be used with any set of channels that capture information from a 3D object.

In our studies, we use different subsets of these 6 channels, which we refer to using D for the depth channel of the mesh alone, DSC for the depth, sine, and cosine channels of the mesh, and DSCDSC for the depth, sine, and cosine channels of both the mesh and convex hull.

6.4.2 Number of Views vs Pixels per View

On the principle that more information is better, we might expect the best V^2 representations to be those that use lots of cameras, positioned all around an object, where each camera has a super-wide field of view and a super-high resolution. But, which of these parameters is the *most* helpful to maximize?

Of many tradeoffs that could be considered, we focus on the tradeoff between the Number of Views (NV) versus the Pixels per View (PV). As in our museum example, suppose

we control for a constant amount of input information by holding fixed the total number of pixels available. We could either spend those pixels on a small number of views, with many pixels per view (i.e., multi-view approach), or on a large number of views, with one pixel per view (i.e., spherical approach), or something in the middle.

Figure 6.3 shows how V^2 representations fall along a straight line in loglog scale in the

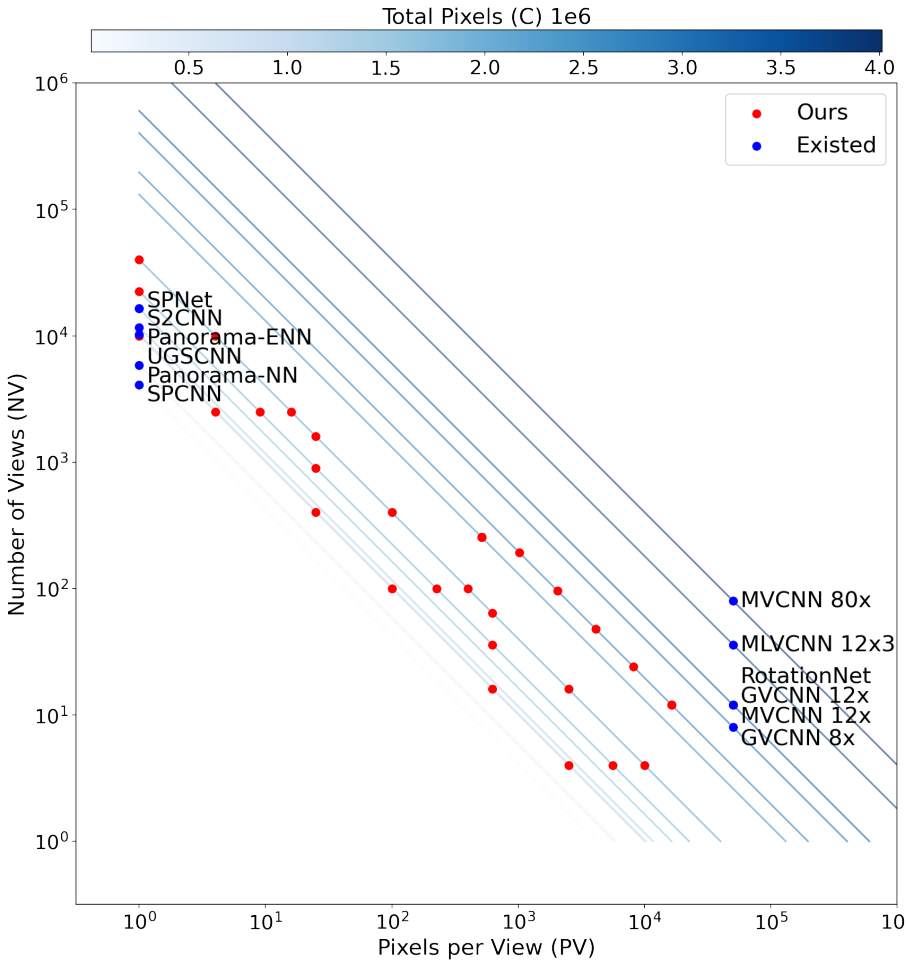


Figure 6.3: By fixing the total number of pixels C per channel, the number of views NV (y-axis) and the pixels per view PV (x-axis) will form a straight line in loglog scale. Different lines represent different C. Blue dots indicate where multi-view [6, 7, 8, 9] and spherical representations [10, 11, 12, 13, 14, 15] from existing research are located in this space. Red dots indicate where the new V^2 representation experiments presented in this paper are located in this space (including experiments that fuse several of these representations). In other words, we expect that the V^2 framework and our initial experiments serves to open up a new region of representations for 3D object recognition that is ripe for exploration.

space defined by number of views versus pixels per view. The dots on each straight line have the same number of total pixels per channel; the closer to the top-right corner of the figure the line is, the higher number of total pixels that representations lying upon that line use.

In this figure, the blue dots show where existing work on multi-view (bottom right) and spherical representations (top left) would be located, and the red dots show the new V^2 representation experiments presented in this paper, including experiments that fuse representations at several points along one of these lines.

6.4.3 V^2 representations in our experiments

Next, we describe in detail how we generate V^2 representations. Figures 6.4 (for experiment 1) and 6.6 (for experiments 2 and 3) show concrete examples of the range of V^2 representations from multi-view to spherical, while keeping the total number of pixels constant.

Data Normalization. Objects are normalized into a sphere with the longest dimension of the object defined to be slightly smaller than the diameter of the sphere. For our first experiment, we used a sphere of diameter 1 with objects having a longest side of length 0.8. For our second and third experiments, we used a sphere of diameter 2 with objects having longest sides of length 0.99 * 2 = 1.98.

View Center. View centers are points sampled on the surface of the sphere. Sphere sampling is an open research problem [106], and our V^2 implementation supports several sampling methods. For this paper, we adopt a straightforward approach for generating regular grid points on the sphere along lines of longitude and latitude.

View Grid. Each view plane is a tangent plane contacting the sphere at each view center. Plane sampling [169] is another research area in its own right; for our experiments, we slightly refined the approach of centric systematic [170] sampling to generate the view grid.

Specifically, every point on the grid is symmetrically surrounding the center point, constructing a matrix of dimension w by h and shooting a ray parallel to its plane normal in which the position of the center point is $(\lfloor w/2 \rfloor, \lfloor h/2 \rfloor)$. The closest point interval d determines the density of each view plane, and we set $d = 1/x$ to guarantee each ray will at least have a segment in the normalized sphere.

6.5 Experiments and Results

Table 6.1 shows how previous research can fit into our V^2 framework and how our experiments examine novel combinations of V^2 parameters. We designed and carried out three experiments to investigate the utility of the V^2 representation space.

In experiment 1, we applied a vanilla ResNet18 architecture to several V^2 representations and found that, while images appear distorted, several "middle" V^2 representations not previously explored in the literature can support surprisingly decent performance, especially given that the architecture was specially designed for these representations.

In experiment 2, we tested how well deep learning models designed for one type of representation would perform across the range of V^2 representations. We adopted ResNet18 [31] as a baseline architecture, MVCNN [6] as the multi-view architecture, and S2CNN [10] as the spherical architecture.

In experiment 3, we fused several different representations along the V^2 continuum using what we call a multi-representation CNN (MRCNN) (Section 6.5.3).

The dataset to evaluate our V^2 representations in all three experiments is ModelNet40 [33], with an 80-20 split for training and testing as in previous research [33, 6].

6.5.1 Experiment 1. V^2 representations using vanilla ResNet18

In experiment 1, the containing sphere has a diameter of 1, and so the ray traveling distance ranges from $[0.0, 1.0]$. The longest side of each object is normalized to 0.8.

	Variable-Viewpoint Representation									Network Architecture
	m	n	w	h	d	NV	PV	NC	C	
Su *et tl.* [1]	1	12	224	224	n/a	12	50176	3	602212	MVCNN
Kanezaki *et al.* [2]	1	12	224	224	n/a	12	50176	3	602212	RotationNet
Taco *et al.* [3]	16384		1	1	n/a	16384	1	6	98304	S2CNN
Chiyu *et al.* [4]	10242		1	1	n/a	10242	1	6	61452	UGSCNN
This Paper Experiment 1	1	4	50	50	1/50	4	2500	1	10000	ResNet18
	2	8	25	25		16	625			
	⋮	⋮	⋮	⋮		⋮	⋮			
	25	100	1	1		10000	1			
	1	4	75	75	1/75	4	5625	1	22500	
	3	12	25	25		36	625			
	⋮	⋮	⋮	⋮		⋮	⋮			
	75	300	1	1		22500	1			
	1	4	100	100	1/100	4	10000	1	40000	
	2	8	50	50		16	2500			
	⋮	⋮	⋮	⋮		⋮	⋮			
	100	400	1	1		40000	1			
This Paper Experiment 2 & 3	1	1	128	128	2/128	1	16384	3	49152	ResNet18 S2CNN MVCNN[5] MRCNN[6]
	2	2	64	64		4	4096			
	⋮	⋮	⋮	⋮		⋮	⋮			
	128	128	1	1		16384	1			
This Paper Experiment 3	1	1	128	128	2/128	1	16384	6	98304	MRCNN[6]
	2	2	64	64		4	4096			
	⋮	⋮	⋮	⋮		⋮	⋮			
	128	128	1	1		16384	1			

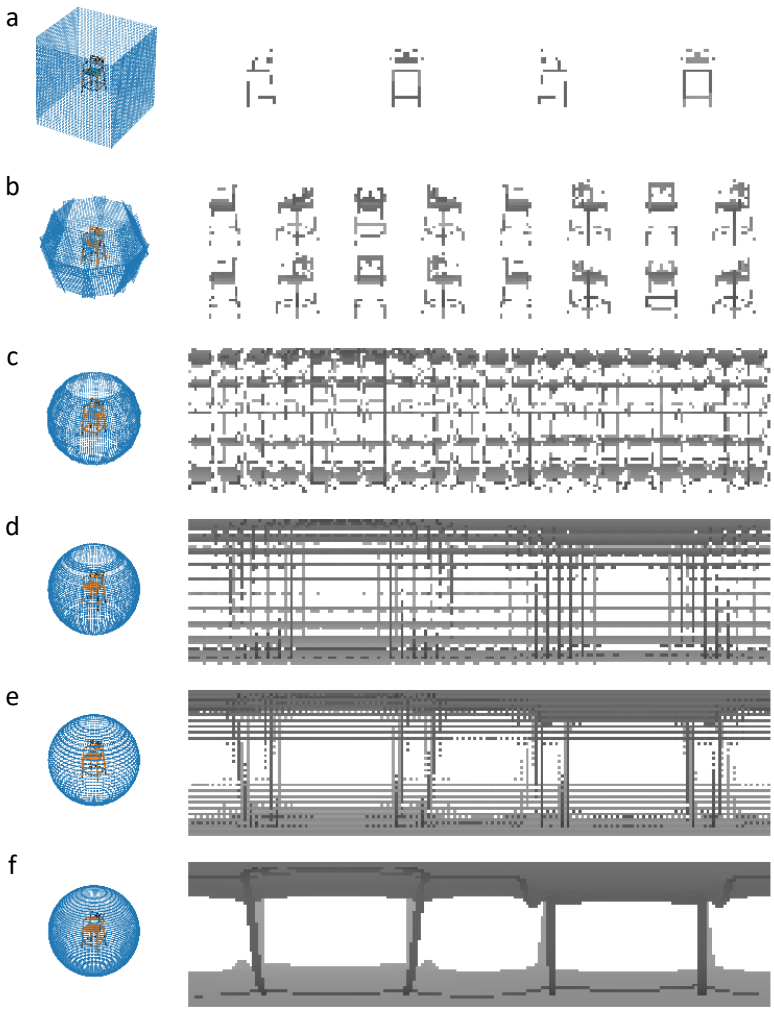
Table 6.1: Using the parameters described in Section 6.4.1, we characterize the multi-view and spherical representations from four previous research studies. [1] [2] Each view has RGB channels rendered by Phong [1] reflection model. [3] [4] Adopted different sphere sampling methods whose m and n are replaced by the number of sphere sampling points. [3] [4] Each view has DSCDSC channels. [5] We use ResNet18 as the backbone. [6] MRCNN is the Multi-Representation CNN proposed in this paper by using a fully-connected layer to fuse all representations together (details in Section 6.5.3).

The view centers on the sphere were sampled in a uniform distribution. Let θ, ϕ be the azimuthal and polar angle in polar coordinates; θ is evenly sampled m times from $[0, 2\pi)$, and ϕ is evenly sampled n times from $(0, 1)$, updated by $\phi = \arccos(1 - 2\phi)$ to avoid polar clustering. Then each pair of (θ, ϕ) is the coordinate of a view center.

In experiment 1, the V^2 representations have an aspect ratio of 4:1 (width: height) to illustrate the evolution from the multi-view of 4 to the spherical extreme. Starting from the multi-view end, the total number of pixels per channel, C, has been set to 10000 ($m = 1$ $n = 4$ $w = 25$ $h = 25$), 22500 ($m = 1$ $n = 4$ $w = 50$ $h = 50$), and 40000 ($m = 1$ $n = 4$ $w = 75$ $h = 75$).

Intermediate-level V^2 representations are configured by identically subdividing each

view into 4 quadrants, so ($m = 1$ $n = 4$ $w = 100$ $h = 100$) will be decomposed to ($m = 2$ $n = 8$ $w = 50$ $h = 50$). This process will keep constructing new representations until reaching the spherical end ($m = 100$ $n = 400$ $w = 1$ $h = 1$). Figure 6.4 shows examples



C = 10000	m	n	x	y	d
a	1	4	50	50	0.02
b	2	8	25	25	0.02
c	5	20	10	10	0.02
d	10	40	5	5	0.02
e	25	100	2	2	0.02
f	50	200	1	1	0.02

Figure 6.4: The V^2 representations used in experiment 1. From top to bottom the representations are moving from the multi-view end to the spherical end while keeping the total number of pixels constant. The left shows the sampling point distribution, the right shows the V^2 D channel, and the bottom table shows the five parameters for each V^2 setup.

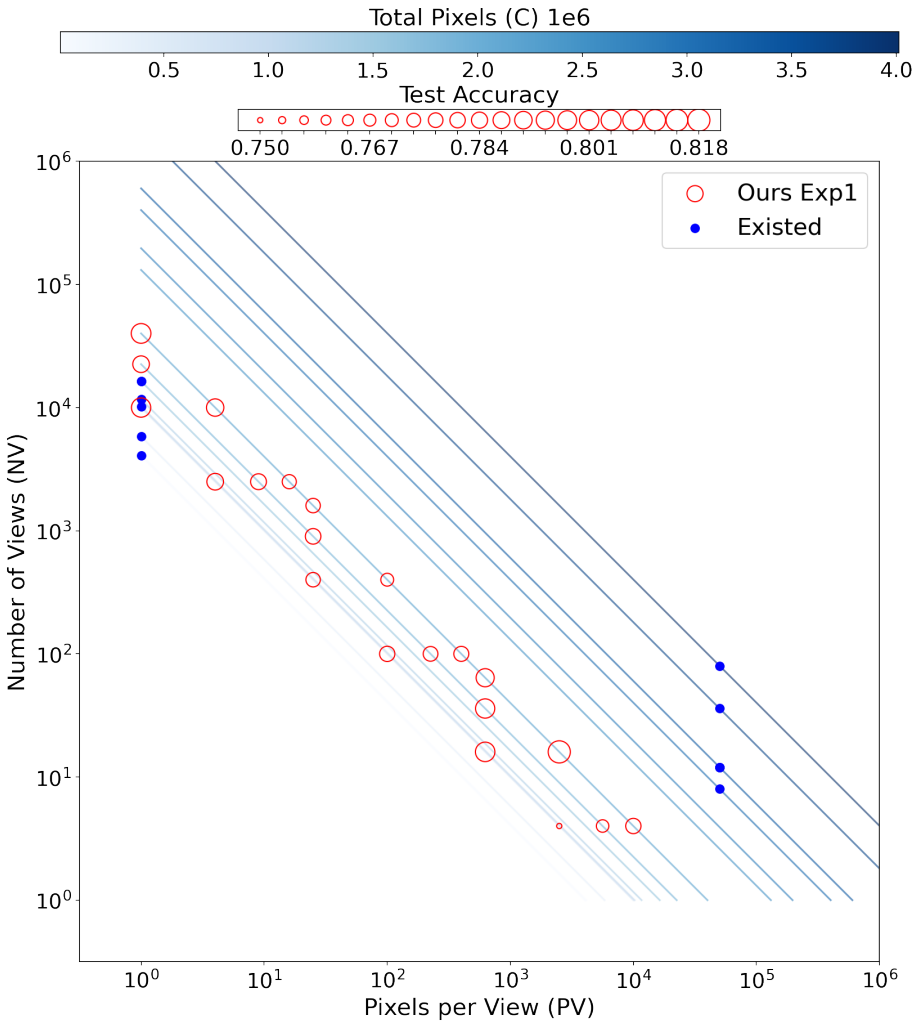


Figure 6.5: The ModelNet40 test accuracies for 3D object recognition in the *NV-PV* V^2 space of experiment 1. The y-axis, x-axis, blue dots, and the straight blue lines are the same as Figure 6.3. Red circles indicate our experiment 1 results using a vanilla ResNet18 with no data augmentation, no pretraining, and no hyperparameter-tuning. Circle size indicates accuracy for our experiments only.

when $C = 10000$.

Figure 6.5 shows ModelNet40 test accuracies for 3D object recognition varying across the V^2 space. Many representations in the V^2 space seem to be highly distorted, as shown in Figure 6.4, but every evaluated point in this space shows decent performance of around 0.750 to 0.818 by just using a vanilla ResNet18 with at most $w = 100$ $h = 100$ pixels, no data augmentation, no pretraining, and no hyperparameter tuning.

The peak accuracies are achieved in the middle of the V^2 space, close to the multi-view end but not quite all the way there, where the representations are similar to those in Figure 6.4b-c. These "middle" representations contain interesting properties that are not well captured by pure multi-view or spherical representations, i.e., depicting fine-scale edges of an object while balancing the overall structure and local patterns. In contrast, multi-view representations have a large field of view that can capture large edges, while spherical representations essentially make object edges implicit.

Existing research has focused on the top-left or the bottom-right areas in the V^2 space, e.g., leveraging more views or more pixels per view. Our research shows that besides pushing the extremes outwards, examining representations in the "middle" will also be an interesting research direction for 3D object recognition.

6.5.2 Experiment 2. V^2 representations on ResNet18, MVCNN and S2CNN

In experiments 2 and 3, the containing sphere now is a unit sphere, so the ray traveling distance ranges from $[0.0, 2.0]$, and the longest side of the object is normalized to 1.98. The view centers on the sphere were sampled using the SOFT [171] method, one of those adopted in S2CNN [10].

The V^2 representations in this experiment are all 128×128 square images with DSC channels resulting in $128 \times 128 \times 3 = 49152$ pixels in total as shown in Figure 6.6.

We rotated each object in 36 directions, with 12 views around each x-, y-, and z- axis to provide V^2 representations with different object positions. We started from a single view ($m = 1$ $n = 1$ $w = 128$ $h = 128$), and then we followed the same subdividing procedure to ($m = 2$ $n = 2$ $w = 64$ $h = 64$) as in experiment 1 until the spherical end at ($m = 128$ $n = 128$ $w = 1$ $h = 1$). This eventually gives us 36 rotations $\times$ 8 V^2 parameter setups $= 288$ representations per object as shown in Figure 6.7. We will use the z-axis rotation V^2 representations in our experiments 2 and 3, i.e., the last 12 columns in Figure 6.7, but this shows how the overall representing space can be enhanced by our V^2 framework.

For MVCNN and S2CNN, all hyper-parameters follow the default values reported in each paper including the learning rate, batch size, optimizer, etc. The vanilla ResNet18 follows the MVCNN hyper-parameter setup since it is also the backbone network for MVCNN. Both MVCNN and S2CNN may not be particularly designed for all the representations across the V^2 space, but because V^2 is a continuum of representations the properties these models are looking for would change smoothly so we believe these models can still perform reasonably on different representations.

The last 12 columns in Figure 6.7 and all 8 rows are the representations used in our experiments 2 and 3.

In experiment 2, we test how deep learning models designed for one type of represen-

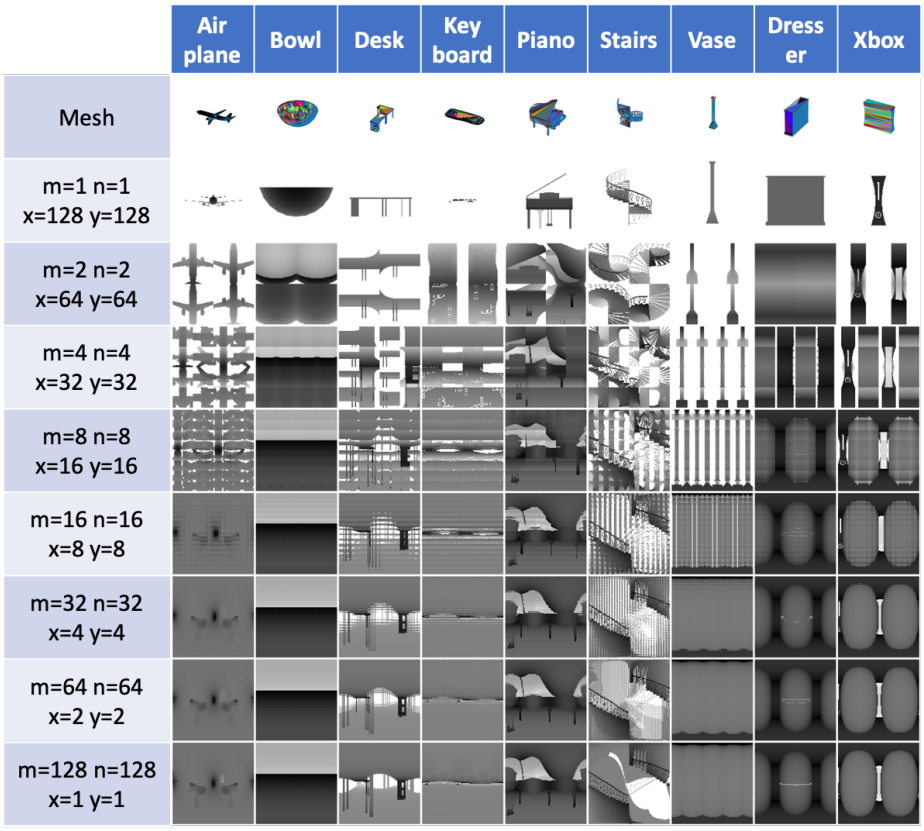


Figure 6.6: The V^2 representations used in experiments 2 and 3. Each column is a different object in ModelNet40. The first row is the mesh and the rest of the rows are V^2 representations generated from different parameters, as marked on the left.

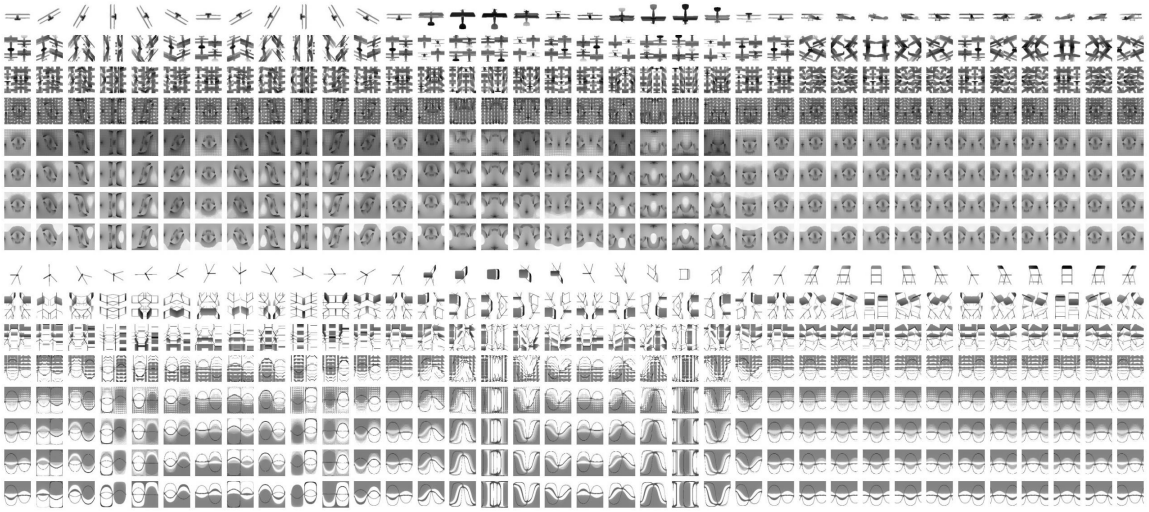


Figure 6.7: The large V^2 representing space for 288 representations per object where we will use the last 12 columns in our experiment 2 and 3. The top half is for an airplane in ModelNet40 and the bottom half is for a chair in ModelNet40. In each half, each column is a different rotation, i.e., the first 12 columns are for y-axis rotation, the middle 12 columns are for x-axis rotation, and the last 12 columns are for z-axis rotation. From top to bottom, each row is a different V^2 representation evolving from ($m = 1$ $n = 1$ $w = 128$ $h = 128$) to ($m = 128$ $n = 128$ $w = 1$ $h = 1$). Figure shows in D channel only.

tation perform across the V^2 continuum, so we treat each individual row (a different V^2 parameter setup) with all columns (different positions) as an individual experiment. Figure 6.8 shows the results for these 3 architectures tested on 8 different V^2 representations.

To better compare the pure expressive power of different representations in different architecture conditions, we eliminated most of the machine learning tricks for performance improvement including pretraining, rotation augmentation, and hyperparameter-tuning. We expect that this is why, in this comparison experiment, the performance levels of MVCNN and S2CNN are not as high as the reported numbers in their original papers.

The results of experiment 2 are shown in Figure 6.8, and several points are noteworthy about this chart.

First, the highest accuracies are achieved in the "middle" region of the V^2 space, ($m = 4$ $n = 4$ $w = 32$ $h = 32$) for MVCNN and ($m = 8$ $n = 8$ $w = 16$ $h = 16$) for ResNet18 and S2CNN. In other words, the home regions for MVCNN (multi-view input representations)

and S2CNN (spherical input representations) aren't actually the best performance regions for each respective architecture within the V^2 space.

Why might this be? We offer a few possible explanations. First, we have maximized the pixel occupancy rate for the multi-view end of our representing space by normalizing the longest side of each object to span 99% of the receptive field. However, there is still a lot of blank space in these images (as shown in the bottom left thumbnails in Figure 6.8), and so the level of possible information utilization may not be as high as for the other representations.

Second, not all architectures are created equal in their abilities to utilize information

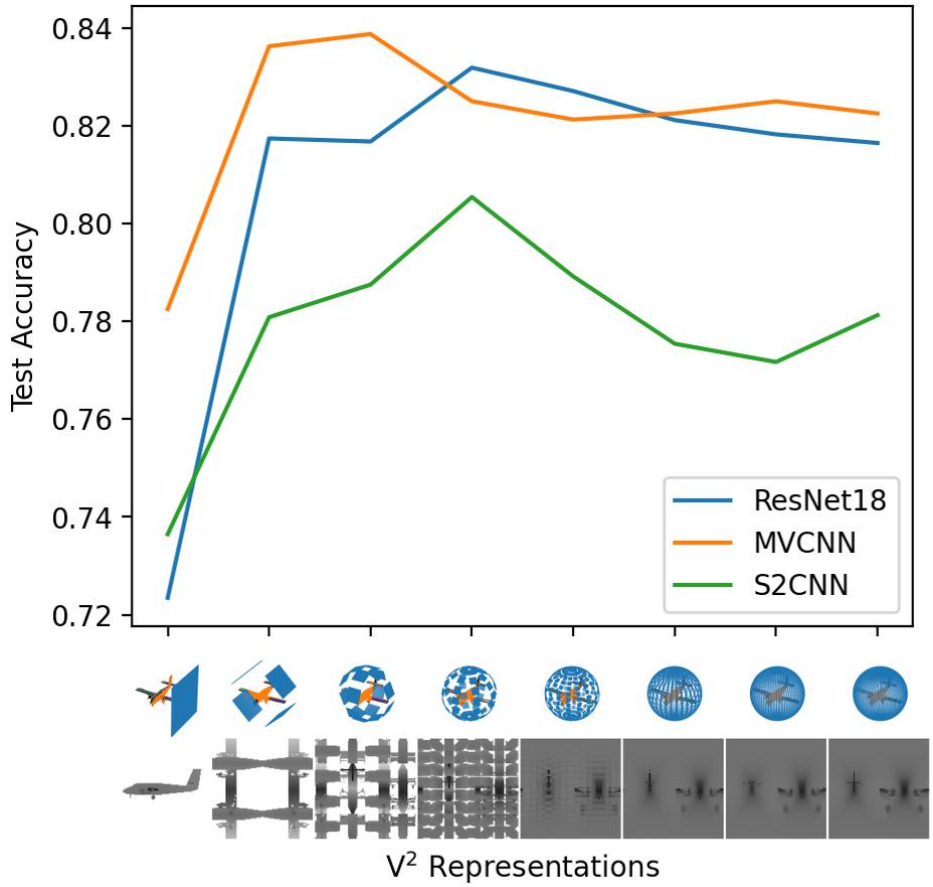


Figure 6.8: The ModelNet40 test accuracies for 3D object recognition across 8 V^2 representations. No matter which specific representation type the architecture was designed to use, the peak performance is located around the "middle" representations ($m = 2$ $n = 2$ $w = 64$ $h = 64$), ($m = 4$ $n = 4$ $w = 32$ $h = 32$), and ($m = 8$ $n = 8$ $w = 16$ $h = 16$).

from across the V^2 space. Other than the multi-view end on the left, and perhaps unsurprisingly, the plain ResNet18 architecture shows fairly consistent performance across many representations. More surprisingly, both MVCNN and S2CNN exhibit an interesting "S" shape in their performance. The "S" shape across all three different models indicate that among all kinds of V^2 representations, the "middle" representations closer to the multi-view end may contain more, or more useful, information in the same number of pixels than do the repreesntations on either "end" of the V^2 spectrum.

Third, in some of the "middle" representations, ResNet18 even outperforms the other architecture-representation combinations, even though it was not specifically designed for 3D object recognition at all. This again shows the expressive power of the "middle" V^2 representations.

In the Discussion (Section 6.6), we will provide our explanations of how these "middle" representations could provide more useful information, per pixel, than other regions of the representing space.

6.5.3 Experiment 3. The fusion of V^2 representations

V^2 provide a rich representing space, as shown in Figure 6.7. Thus to better utilize all kinds of representations, we propose the Multi-Representation Convolutional Neural Network (MRCNN) architecture, as shown in Figure 6.9. We adopted an ImageNet [16] pretrained ResNet18 as the backbone network. Besides pretraining, there was no hyperparameter-tuning done in this experiment.

Table 6.2 shows results for applying multi-view multi-V^2 representations on MRCNN networks. By fusing different V^2 representations together, many models achieved more than 90% accuracy.

By fusing the "middle" ($m = 2\ n = 2\ w = 64\ h = 64$) V^2 representations and 12 positions together, the MRCNN achieved the highest accuracy of 90.875%. For the other rows with the same backbone ResNet18 and rotation 12, i.e., the same architecture and C, we again

observed the "S" shape pattern, a lower start at $m = 1$ and then $m = 2$, $m = 4$, and $m = 8$ achieve the peak accuracies. The rest of the 90% accuracy group, among configurations that we tested, all resulted from fusing all 8 V^2 representations with the front position only, no matter which backbone the MRCNN used. This illustrates what we believe is untapped expressive power within these "middle" representations of the V^2 framework.

Revisiting the *NV-PV* V^2 space, we depict the originally reported accuracies of the representatives of the multi-view networks, the spherical networks, and our MRCNN networks in the same space, as shown in Figure 6.10. Note that the reported accuracies for other networks are typically highly hyperparameter-tuned, data augmented, and models ensembled to the ModelNet40 object recognition task.

A notable comparison is between the very middle group of 90+% accuracy models to the MVCNN 80x at the darkest straight line (the largest total number of pixels) that

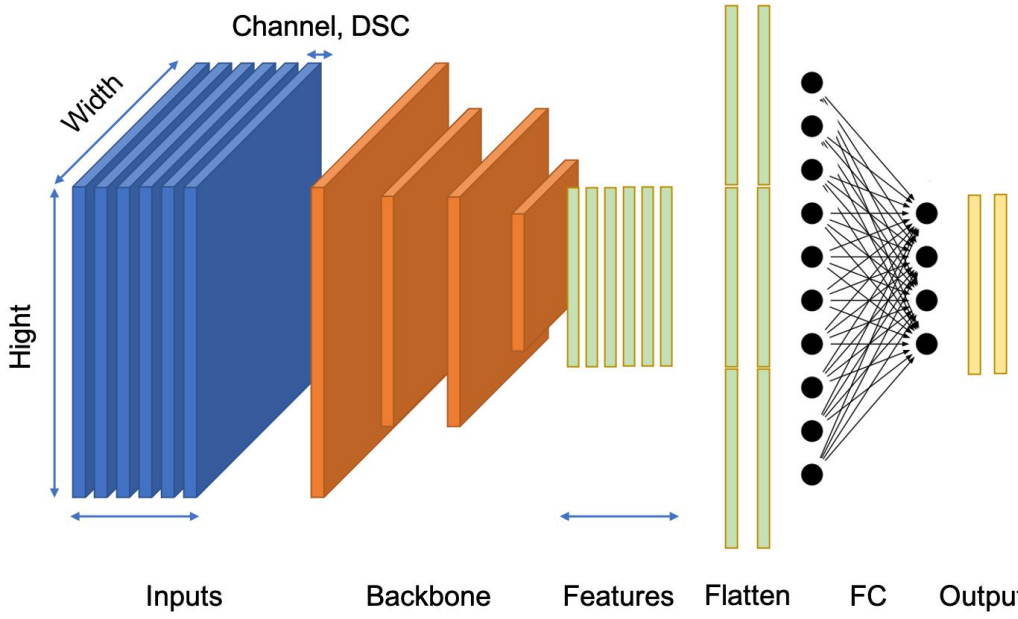


Figure 6.9: The Multi-Representation Convolutional Neural Network (MRCNN) architecture. This diagram shows an MRCNN example by using DSC channels, 3 representations per object, and predictions for 2 objects. The core is after getting the features of all 3 representations of the same object, the network will concatenate them together as the final shape descriptor.

Rank	Channel	Batch	Backbone	Rotation	V^2 Configuration	Accuracy	0.8 0.9
1	DSC	4	resnet18	12	[2]	0.90875	
2	DSC	4	inception_v3	1	[1, 2, ..., 64, 128]	0.90750	
3	DSC	2	resnext50_32x4d	1	[1, 2, ..., 64, 128]	0.90625	
4	DSC	4	resnet18	1	[1, 2, ..., 64, 128]	0.90375	
5	DSCDSC	2	resnet18	1	[1, 2, ..., 64, 128]	0.90375	
6	DSC	4	resnet50	1	[1, 2, ..., 64, 128]	0.90125	
7	DSC	4	resnet18	12	[8]	0.89500	
8	DSC	4	resnet18	12	[4]	0.89250	
9	DSC	4	resnet18	12	[1]	0.89000	
10	DSC	4	resnet18	12	[16]	0.88500	
11	DSCDSC	2	resnet18	12	[1]	0.88375	
12	DSCDSC	2	inception_v3	1	[1, 2, ..., 64, 128]	0.88125	
13	DSCDSC	2	inception_v3	12	[1]	0.86375	
14	DSC	1	resnext50_32x4d	1	[1, 2, ..., 64, 128]	0.83125	

Table 6.2: The accuracy table for fusing different V^2 representations through MRCNN. The batch column is the batch size of objects, so the actual batch size in memory will be multiplied by the number of representations per object. In the position column, 12 means "uses all 12 positions" (the last 12 columns in Figure 6.7) and 1 means "uses the front position only." The V^2 Configuration column uses the m value to indicate which V^2 representation is in use, e.g., 2 means the V^2 setup of ($m = 2$ $n = 2$ $w = 64$ $h = 64$).

achieves 90.1% accuracy. In the MVCNN paper, the author mentioned that they had applied data augmentation, low-rank Mahalanobis metric learning, hyperparameter-tuning, and 80x multi-views to achieve this 90.1% accuracy, whereas we just used 8 different V^2 representations, with no other adjustments or augmentations, to achieve several different 90+% model accuracy results.

The strong performance in the middle shows considerable potential to better utilize each pixel in that region of the V^2 space. The central field is much larger than the two extreme corners of the common multi-view and spherical representations, and waits to be explored further.

6.6 Discussion

Why might the "middle" representations perform better than other regions in the V^2 space no matter which architectures they interact with? Here are three perspectives:

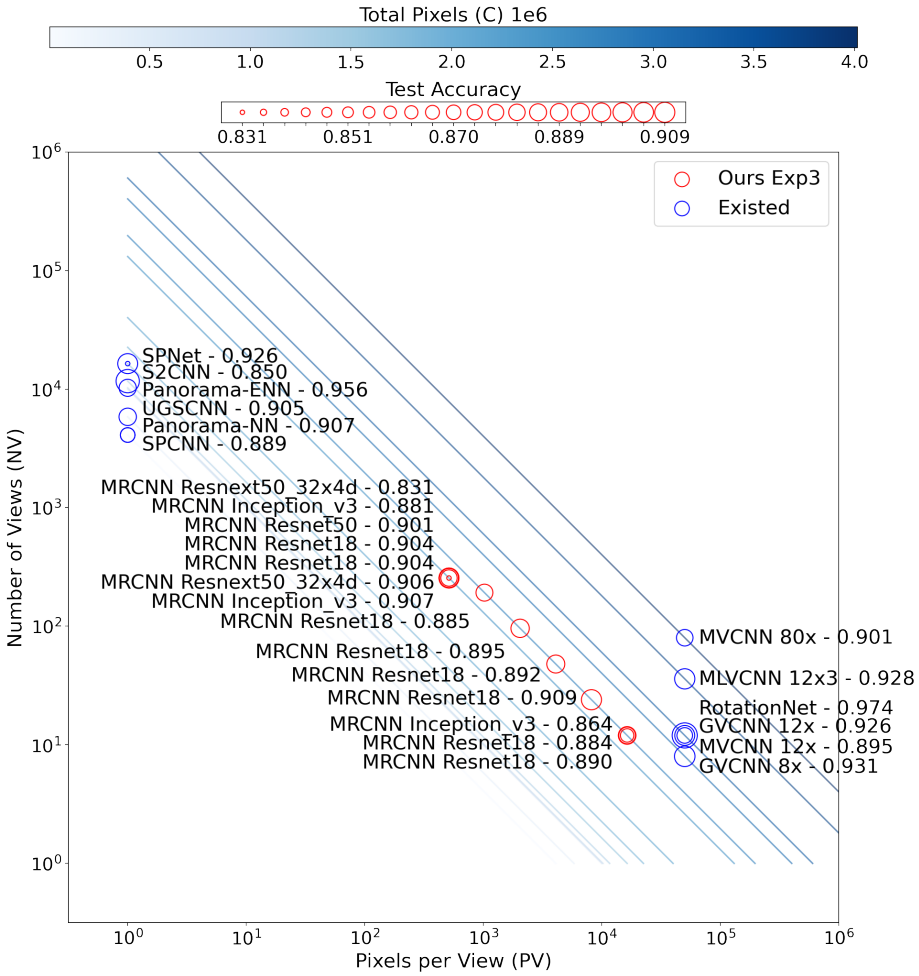


Figure 6.10: Revisiting the *NV-PV* V^2 space with accuracy comparisons between our results from experiment 3 and existing work. Red circles are ours and blue circles are existing work. Circle size indicates accuracy. The y-axis, x-axis, and the straight blue lines are the same as in Figure 6.3.

1. The "middle" representations are better for capturing small edges of the object. These small edges are easily overlooked if we just capture the whole body shape from the multi-view end. These edges also become implicit if captures from the spherical end.
2. The "middle" representations provide balance between focusing on the local patterns versus the global structure. The information from local and global elements is likely complementary to each other, and thus capturing both results in a better pixel utilization.

3. The "middle" representations balance both translation invariance and rotation invariance, and so provides robustness for these invariances in varying degrees.

6.7 Related Work

The family of 3D representations keeps on expanding. Some representations do not rely on pixel-based images, such as polygon soup [87, 88], sweep-CSG [89], and spline-based representation [90]. Some need extra data processing such as polygon mesh [92], point cloud [94], and Octree/Quadtree-based representations [96] in the application of Graph CNN on Mesh [91], PointNet [93], and Octree-based O-CNN [95]. Multi-view and voxel representations are commonly used for deep learning, as they can be directly applied to CNNs such as MVCNN [6] and VoxNet [100].

Spherical representations were investigated in 2002 [105], and because of their rotation-invariance properties, spherical deep learning models [10, 15, 11] have emerged over recent years. Some comparisons have been done between multi-view and voxel [102], and our work can cover the comparisons between multi-view and spherical.

Essentially, V^2 is a 2D mapping strategy that shows considerable potential applications for 3D object recognition and also considerable promise for deep learning approaches.

6.8 Conclusion

As we have shown, performance improvements from using V^2 representations are promising, both from using the "middle" representations alone or by fusing multiple V^2 representation types. Moreover, we observed these performance gains across fundamentally different models, including models that had not been specifically trained to accommodate these "middle" V^2 representations.

The V^2 framework will support many other interesting research questions beyond what we illustrated here. For example, we evaluated model performance after fixing C, (total number of pixels) but fixing NV (number of views) or PV (pixels per view) may also be

interesting. For example, if we fix PV, how would different settings of NV affect performance?

In this chapter, I design and implement the Variable-Viewpoint (V^2) representations to provide the continuum of representations from different viewpoint distributions.

I show that the Multi-View and Sphere are just two extremes in the V^2 space and evaluate how models designed for a single type of representation will perform object recognition in this space.

I propose the Multi-Representation Convolutional Neural Network (MRCNN) to fuse all representations together which show the strong performance for the "middle" representations in the V^2 space. This result indicates a direction to improve performance by moving to the "middle" to utilize input space more efficiently.

Chapter 7

Conclusion

7.1 Summary

In the deep learning era, large-scale datasets have become the soil for nurturing deep neural networks, and these deep models have made remarkable achievements on vision tasks such as object recognition.

However, for humans during infancy, the learning environment consists of extended, embodied experience with a few, intensively observed objects, which is quite different from the large-scale datasets commonly used in computer vision. Therefore, inspired by research from developmental psychology, this dissertation explores object recognition from the representational side in two main aspects, dataset collection and experiments on a new instance-limited and viewpoint-rich dataset — Toybox, and a new projecting framework to generate viewpoint-rich representations and the experiments on these viewpoint-rich representations — V^2.

7.2 Contributions

There are 4 main contributions in this dissertation:

First, I collected, annotated, and organized the Toybox dataset which contains egocentric, instance-limited, and viewpoint-rich videos of 12 transformations, 360 handheld objects across 12 categories. These properties can support novel research to bring developmentally relevant experiences to the machine learning field.

Second, I benchmarked the object recognition performance between deep learning models and humans on the Toybox dataset, including identifying where the gaps are in category-, object-, and viewpoint levels, which provides new goals for our computational models to achieve robust recognition performance on datasets like Toybox, both in terms

of training regimes and testing inputs.

Third, I demonstrated the temporal consistency propagating through the representing spaces of neural networks in the input layer and the bottleneck layer, and then utilized this consistency to enhance interpretations of hidden layer neurons.

Last, I designed and implemented the Variable-Viewpoint (V^2) representational framework and demonstrate 1) how representations designed for different architectures such as multi-view and spherical can be unified on the same space; 2) the broad region of the "middle" representations that may have potential to depict objects more efficiently; 3) the potential of the rich representations in V^2 to improve the performance of the object recognition task.

7.3 Discussion

For Artificial Intelligence (AI), some considerable progress has been made in recent years because of substantial computational resources, large-scale datasets, and deep learning models. Although we have made remarkable achievements under the philosophy of "more is better", there might be a risky loop in this research direction: that is, if we want a better performance on some AI task, we will want to make the models deeper and wider, we will then need to collect more data to train the model, we will then need more computational resources, and finally we want to improve the performance again by making the models deeper and wider...

Neural networks that have millions or even billions of parameters are common to see nowadays, for example, BERT [172] with 110 million parameters or the T-NLG [173] with 17 billion parameters. Those used-to-be the large-scale datasets may become small datasets again, such that mega-scale datasets will be the new "large-scale" in the future.

In the Computer Vision domain, many researchers have also pointed out the risks of this kind of performance-developing loop, and so a lot of ideas have been proposed such as few-shot, zero-shot learning, or the developmental approach of machine learning.

This last direction, which is inspired by developmental psychology, is the core motivation of this dissertation.

For the object recognition task, if infants could learn by observing a few instances with extensive viewpoints, it seems fair to expect that we could have computer vision models that behave similarly. Perhaps we do still need mega-scale datasets, but we could just let the models learn from the datasets in a developmental way, i.e., observing a few objects comprehensively and then learning new objects in a long-tail and one-shot way. This learning trajectory may mitigate the hunger for computational and mega-scale data resources that seems to be continuing on our current path.

7.4 The Future

I hope the Toybox dataset with its benchmark and interpretation experiments and the V^2 framework could support various kinds of research to bring the developmental approach closer to the computer vision domain. Any dissertation isn't just an end of a research journey but is a commencement of the research future.

BIBLIOGRAPHY

[1] Bui Tuong Phong. Illumination for computer generated pictures. *Communications of the ACM*, 18(6):311–317, 1975.

[2] Yu Xiang, Wonhui Kim, Wei Chen, Jingwei Ji, Christopher Choy, Hao Su, Roozbeh Mottaghi, Leonidas Guibas, and Silvio Savarese. Objectnet3d: A large scale database for 3d object recognition. In *European Conference Computer Vision (ECCV)*, 2016.

[3] Yu Xiang, Wonhui Kim, Wei Chen, Jingwei Ji, Christopher Choy, Hao Su, Roozbeh Mottaghi, Leonidas Guibas, and Silvio Savarese. Objectnet3d: A large scale database for 3d object recognition. In *European Conference Computer Vision (ECCV)*, 2016.

[4] Linda B Smith and Lauren K Slone. A developmental approach to machine learning? *Frontiers in psychology*, 8:2124, 2017.

[5] Larry Fenson, Elizabeth Bates, Philip S Dale, Virginia A Marchman, J Steven Reznick, and Donna J Thal. *MacArthur-Bates communicative development inventories*. Paul H. Brookes Publishing Company, 2007.

[6] Hang Su, Subhransu Maji, Evangelos Kalogerakis, et al. Multi-view convolutional neural networks for 3d shape recognition. In *Proceedings of the IEEE international conference on computer vision*, pages 945–953, 2015.

[7] Asako Kanezaki, Yasuyuki Matsushita, et al. Rotationnet: Joint object categorization and pose estimation using multiviews from unsupervised viewpoints. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5010–5019, 2018.

[8] Jianwen Jiang, Di Bao, Ziqiang Chen, et al. Mlvcnn: Multi-loop-view convolutional neural network for 3d shape retrieval. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 8513–8520, 2019.

[9] Yifan Feng, Zizhao Zhang, Xibin Zhao, Rongrong Ji, and Yue Gao. Gvcnn: Group-view convolutional neural networks for 3d shape recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 264–272, 2018.

[10] Taco S. Cohen, Mario Geiger, Jonas Köhler, and Max Welling. Spherical CNNs. In *International Conference on Learning Representations*, 2018.

[11] Chiyu Max Jiang, Jingwei Huang, Karthik Kashinath, Prabhat, Philip Marcus, and Matthias Niessner. Spherical CNNs on unstructured grids. In *International Conference on Learning Representations*, 2019.

[12] Mohsen Yavartanoo, Eu Young Kim, and Kyoung Mu Lee. Spnet: Deep 3d object classification and retrieval using stereographic projection. In *Asian Conference on Computer Vision*, pages 691–706. Springer, 2018.

[13] Konstantinos Sfikas, Theoharis Theoharis, et al. Exploiting the panorama representation for convolutional neural network classification and retrieval. *3DOR*, 6:7, 2017.

[14] Konstantinos Sfikas, Ioannis Pratikakis, and Theoharis Theoharis. Ensemble of panorama-based convolutional neural networks for 3d model classification and retrieval. *Computers & Graphics*, 71:208–218, 2018.

[15] Carlos Esteves, Christine Allen-Blanchette, Ameesh Makadia, et al. Learning so (3) equivariant representations with spherical cnns. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 52–68, 2018.

[16] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[17] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[18] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[19] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

[20] Xiaohan Wang, Tengyu Ma, James Ainooson, Seunghwan Cha, Xiaotian Wang, Azhar Molla, and Maithilee Kunda. The toybox dataset of egocentric visual object transformations. *arXiv preprint arXiv:1806.06034*, 2018.

[21] William T Cochran, James W Cooley, David L Favin, Howard D Helms, Reginald A Kaenel, William W Lang, George C Maling, David E Nelson, Charles M Rader, and Peter D Welch. What is the fast fourier transform? *Proceedings of the IEEE*, 55(10):1664–1674, 1967.

[22] Tengyu Ma, Joel Michelson, James Ainooson, Deepayan Sanyal, Xiaohan Wang, and Maithilee Kunda. Variable-viewpoint representations for 3d object recognition, 2020.

[23] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

[24] Yann LeCun, Yoshua Bengio, et al. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10):1995, 1995.

[25] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012.

[26] The benchmark of the image classification on imagenet. https://paperswithcode.com/sota/image-classification-on-imagenet.

[27] Carolyn Mervis. Child-basic object categories and early lexical development. In Ulric Neisser, editor, *Concepts and Conceptual Development: Ecological and Intellectual Factors in Categorization*, pages 201–233. Cambridge University Press, 1987.

[28] Emin Orhan, Vaibhav Gupta, and Brenden M Lake. Self-supervised learning through the eyes of a child. *Advances in Neural Information Processing Systems*, 33, 2020.

[29] Miguel Lagunes-Fortiz, Dima Damen, and Walterio Mayol-Cuevas. Learning discriminative embeddings for object recognition on-the-fly. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 2932–2938. IEEE, 2019.

[30] Cmu multi-modal activity database. https://www.mturk.com/. Accessed: 10-21-2020.

[31] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[32] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[33] Zhirong Wu, S. Song, A. Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and J. Xiao. 3d shapenets: A deep representation for volumetric shapes. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1912–1920, June 2015.

[34] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Fei-Fei Li. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009*, pages 248–255. IEEE, 2009.

[35] Google 3d warehouse. https://3dwarehouse.sketchup.com/. Accessed: 2016.

[36] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015.

[37] Yu Xiang, Roozbeh Mottaghi, and Silvio Savarese. Beyond pascal: A benchmark for 3d object detection in the wild. In *IEEE winter conference on applications of computer vision*, pages 75–82. IEEE, 2014.

[38] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010.

[39] Google 3d warehouse. http://sketchup.google.com/. Accessed: 2014.

[40] Joseph J Lim, Hamed Pirsiavash, and Antonio Torralba. Parsing ikea objects: Fine pose estimation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2992–2999, 2013.

[41] Flickr. https://www.flickr.com/. Accessed: 2013.

[42] Xingyuan Sun, Jiajun Wu, Xiuming Zhang, Zhoutong Zhang, Chengkai Zhang, Tianfan Xue, Joshua B Tenenbaum, and William T Freeman. Pix3d: Dataset and methods for single-image 3d shape modeling. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2974–2983, 2018.

[43] Structure sensor. https://structure.io/. Accessed: 2013.

[44] Andrei Barbu, David Mayo, Julian Alverio, William Luo, Christopher Wang, Dan Gutfreund, Josh Tenenbaum, and Boris Katz. Objectnet: A large-scale bias-controlled dataset for pushing the limits of object recognition models. In *Advances in Neural Information Processing Systems*, pages 9448–9458, 2019.

[45] Silvio Savarese and Li Fei-Fei. 3d generic object categorization, localization and pose estimation. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8. IEEE, 2007.

[46] M. Ozuysal, V. Lepetit, and P.Fua. Pose estimation for category specific multiview object localization. In *Conference on Computer Vision and Pattern Recognition*, Miami, FL, June 2009.

[47] Sameer A Nene, Shree K Nayar, Hiroshi Murase, et al. Columbia object image library (COIL-100). *Technical report CUCS-005-96*, 1996.

[48] Jan Burianek, Alireza Ahmadyfard, Josef Kittler, et al. Soil-47, the surrey object image library. *Centre for Vision, Speech and Signal processing, Univerisity of Surrey.[Online].*, 2000.

[49] Yann LeCun, Fu Jie Huang, and Leon Bottou. Learning methods for generic object recognition with invariance to pose and lighting. In *Computer Vision and Pattern Recognition, 2004.*, volume 2, pages II–104. IEEE, 2004.

[50] Jan-Mark Geusebroek et al. The amsterdam library of object images. *International Journal of Computer Vision*, 61(1):103–112, 2005.

[51] Pierre Moreels and Pietro Perona. Evaluation of features detectors and descriptors based on 3d objects. *International Journal of Computer Vision*, 73(3):263–284, 2007.

[52] Kevin Lai, Liefeng Bo, Xiaofeng Ren, and Dieter Fox. A large-scale hierarchical multi-view rgb-d object dataset. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 1817–1824. IEEE, 2011.

[53] Arjun Singh, James Sha, Karthik S Narayan, Tudor Achim, and Pieter Abbeel. Bigbird: A large-scale 3d database of object instances. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pages 509–516, 2014.

[54] Ali Borji, Saeed Izadi, and Laurent Itti. ilab-20m: A large-scale controlled object dataset to investigate deep learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2221–2230, 2016.

[55] Marc Bolanos, Mariella Dimiccoli, and Petia Radeva. Toward storytelling from visual lifelogging: An overview. *IEEE Transactions on Human-Machine Systems*, 47(1):77–90, 2016.

[56] Alireza Fathi, Yin Li, and James M Rehg. Learning to recognize daily actions using gaze. In *European Conference on Computer Vision*, pages 314–327. Springer, 2012.

[57] Alircza Fathi, Jessica K Hodgins, and James M Rehg. Social interactions: A first-

person perspective. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1226–1233. IEEE, 2012.

[58] Cmu multi-modal activity database. http://kitchen.cs.cmu.edu/. Accessed: 04-08-2020.

[59] Yair Poleg, Chetan Arora, and Shmuel Peleg. Temporal segmentation of egocentric videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2537–2544, 2014.

[60] Zheng Lu and Kristen Grauman. Story-driven summarization for egocentric video. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2714–2721, 2013.

[61] Omid Aghazadeh, Josephine Sullivan, and Stefan Carlsson. Novelty detection from an ego-centric perspective. In *CVPR 2011*, pages 3297–3304. IEEE, 2011.

[62] Hamed Pirsiavash and Deva Ramanan. Detecting activities of daily living in first-person camera views. In *2012 IEEE conference on computer vision and pattern recognition*, pages 2847–2854. IEEE, 2012.

[63] Sibo Song, Vijay Chandrasekhar, Ngai-Man Cheung, Sanath Narayan, Liyuan Li, and Joo-Hwee Lim. Activity recognition in egocentric life-logging videos. In *Asian Conference on Computer Vision*, pages 445–458. Springer, 2014.

[64] Ardhendu Behera, David C Hogg, and Anthony G Cohn. Egocentric activity monitoring and recovery. In *Asian Conference on Computer Vision*, pages 519–532. Springer, 2012.

[65] Dima Damen, Osian Haines, Teesid Leelasawassuk, Andrew Calway, and Walterio Mayol-Cuevas. Multi-user egocentric online system for unsupervised assistance

on object usage. In *European Conference on Computer Vision*, pages 481–492. Springer, 2014.

[66] Stefano Alletto, Giuseppe Serra, Simone Calderara, Francesco Solera, and Rita Cucchiara. From ego to nos-vision: Detecting social relationships in first-person views. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 580–585, 2014.

[67] Michael S Ryoo and Larry Matthies. First-person activity recognition: What are they doing to me? In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2730–2737, 2013.

[68] Lorenzo Baraldi, Francesco Paci, Giuseppe Serra, Luca Benini, and Rita Cucchiara. Gesture recognition in ego-centric videos using dense trajectories and hand segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 688–693, 2014.

[69] Sanath Narayan, Mohan S Kankanhalli, and Kalpathi R Ramakrishnan. Action and interaction recognition in first-person videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 512–518, 2014.

[70] Cheng Li and Kris M Kitani. Pixel-level hand detection in ego-centric videos. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3570–3577, 2013.

[71] Nebojsa Jojic, Alessandro Perina, and Vittorio Murino. Structural epitome: a way to summarize one's visual experience. In *Advances in neural information processing systems*, pages 1027–1035, 2010.

[72] Axel Furlan, Stephen D Miller, Domenico G Sorrenti, Fei-Fei Li, and Silvio Savarese. Free your camera: 3d indoor scene understanding from arbitrary camera motion. In *BMVC*, 2013.

[73] Alireza Fathi, Xiaofeng Ren, and James M Rehg. Learning to recognize objects in egocentric activities. In *CVPR 2011*, pages 3281–3288. IEEE, 2011.

[74] Xiaofeng Ren and Matthai Philipose. Egocentric recognition of handled objects: Benchmark and analysis. In *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pages 1–8. IEEE, 2009.

[75] Vincenzo Lomonaco and Davide Maltoni. Core50: a new dataset and benchmark for continuous object recognition. *arXiv preprint arXiv:1705.03550*, 2017.

[76] Federica Bogo, Javier Romero, Matthew Loper, and Michael J Black. Faust: Dataset and evaluation for 3d mesh registration. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3794–3801, 2014.

[77] Alexander M Bronstein, Michael M Bronstein, and Ron Kimmel. *Numerical geometry of non-rigid shapes*. Springer Science & Business Media, 2008.

[78] Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niessner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. Matterport3d: Learning from rgb-d data in indoor environments. *International Conference on 3D Vision (3DV)*, 2017.

[79] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 2017.

[80] Bryan C Russell and Antonio Torralba. Building a database of 3d scenes from user annotations. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2711–2718. IEEE, 2009.

[81] Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmen-

tation and support inference from rgbd images. In *European conference on computer vision*, pages 746–760. Springer, 2012.

[82] Shuran Song, Samuel P Lichtenberg, and Jianxiong Xiao. Sun rgb-d: A rgb-d scene understanding benchmark suite. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 567–576, 2015.

[83] Manolis Savva, Fisher Yu, Hao Su, Asako Kanezaki, Takahiko Furuya, Ryutarou Ohbuchi, Zhichao Zhou, Rui Yu, Song Bai, Xiang Bai, et al. Large-scale 3d shape retrieval from shapenet core55: Shrec'17 track. In *Proceedings of the Workshop on 3D Object Retrieval*, pages 39–50. Eurographics Association, 2017.

[84] Stefan Stojanov, Samarth Mishra, Ngoc Anh Thai, Nikhil Dhanda, Ahmad Humayun, Chen Yu, Linda B Smith, and James M Rehg. Incremental object learning from contiguous views. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8777–8786, 2019.

[85] Yang Chen and Gérard Medioni. Object modelling by registration of multiple range images. *Image and vision computing*, 10(3):145–155, 1992.

[86] Brian Curless. From range scans to 3d models. *ACM SIGGRAPH Computer Graphics*, 33(4):38–41, 1999.

[87] Peter Lindstrom. Out-of-core simplification of large polygonal models. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 259–262. ACM Press/Addison-Wesley Publishing Co., 2000.

[88] Thomas Colleu, Stéphane Pateux, Luce Morin, and Claude Labit. A polygon soup representation for multiview coding. *Journal of Visual Communication and Image Representation*, 21(5-6):561–576, 2010.

[89] KC Hui and ST Tan. Construction of a hybrid sweep-csg modeler—the sweep-csg representation. *Engineering with computers*, 8(2):101–119, 1992.

[90] Sheng-Chuan Wu, John F Abel, and Donald P Greenberg. An interactive computer graphics approach to surface representation. *Communications of the ACM*, 20(10):703–712, 1977.

[91] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in neural information processing systems*, pages 3844–3852, 2016.

[92] James D Foley, Andries Van Dam, et al. *Fundamentals of interactive computer graphics*, volume 2. Addison-Wesley Reading, MA, 1982.

[93] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 652–660, 2017.

[94] Hugues Hoppe, Tony DeRose, Tom Duchamp, John McDonald, and Werner Stuetzle. *Surface reconstruction from unorganized points*, volume 26. ACM, 1992.

[95] Peng-Shuai Wang, Yang Liu, Yu-Xiao Guo, Chun-Yu Sun, and Xin Tong. O-cnn: Octree-based convolutional neural networks for 3d shape analysis. *ACM Transactions on Graphics (TOG)*, 36(4):72, 2017.

[96] Chiun-Hong Chien and Jake K Aggarwal. Identification of 3d objects from multiple silhouettes using quadtrees/octrees. *Computer Vision, Graphics, and Image Processing*, 36(2-3):256–273, 1986.

[97] Hamidreza Kasaei. Orthographicnet: A deep learning approach for 3d object recognition in open-ended domains. *arXiv preprint arXiv:1902.03057*, 2019.

[98] Nima Sedaghat, Mohammadreza Zolfaghari, Ehsan Amiri, and Thomas Brox. Orientation-boosted voxel nets for 3d object recognition. *arXiv preprint arXiv:1604.03351*, 2016.

[99] Chao Ma, Wei An, Yinjie Lei, and Yulan Guo. Bv-cnns: Binary volumetric convolutional networks for 3d object recognition. In *BMVC*, volume 1, page 4, 2017.

[100] Daniel Maturana and Sebastian Scherer. Voxnet: A 3d convolutional neural network for real-time object recognition. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 922–928. IEEE, 2015.

[101] Andrew Brock, Theodore Lim, James M Ritchie, and Nick Weston. Generative and discriminative voxel modeling with convolutional neural networks. *arXiv preprint arXiv:1608.04236*, 2016.

[102] Charles R Qi, Hao Su, Matthias Nießner, Angela Dai, Mengyuan Yan, and Leonidas J Guibas. Volumetric and multi-view cnns for object classification on 3d data. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5648–5656, 2016.

[103] James R Driscoll and Dennis M Healy. Computing fourier transforms and convolutions on the 2-sphere. *Advances in applied mathematics*, 15(2):202–250, 1994.

[104] Ivan Dokmanić and Yue M Lu. Sampling sparse signals on the sphere: Algorithms and applications. *IEEE Transactions on Signal Processing*, 64(1):189–202, 2015.

[105] Dejan V Vranic and Dietmar Saupe. Description of 3d-shape using a complex function on the sphere. In *Proceedings. IEEE International Conference on Multimedia and Expo*, volume 1, pages 177–180. IEEE, 2002.

[106] Ivan Dokmanić and Yue M Lu. Sampling sparse signals on the sphere: Algorithms and applications. *IEEE Transactions on Signal Processing*, 64(1):189–202, 2015.

[107] Jason D McEwen and Yves Wiaux. A novel sampling theorem on the sphere. *IEEE Transactions on Signal Processing*, 59(12):5876–5887, 2011.

[108] A Ben Hamza and Hamid Krim. Geodesic object representation and recognition. In *International conference on discrete geometry for computer imagery*, pages 378–387. Springer, 2003.

[109] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[110] Yanghao Li, Naiyan Wang, Jiaying Liu, and Xiaodi Hou. Demystifying neural style transfer. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, IJCAI'17, page 2230–2236. AAAI Press, 2017.

[111] Yongcheng Jing, Yezhou Yang, Zunlei Feng, Jingwen Ye, Yizhou Yu, and Mingli Song. Neural style transfer: A review. *IEEE transactions on visualization and computer graphics*, 2019.

[112] Philip TG Jackson, Amir Atapour Abarghouei, Stephen Bonner, Toby P Breckon, and Boguslaw Obara. Style augmentation: data augmentation via style randomization. In *CVPR Workshops*, pages 83–92, 2019.

[113] Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 23–30. IEEE, 2017.

[114] Christopher Bowles, Liang Chen, Ricardo Guerrero, Paul Bentley, Roger Gunn, Alexander Hammers, David Alexander Dickie, Maria Valdés Hernández, Joanna Wardlaw, and Daniel Rueckert. Gan augmentation: Augmenting training data using generative adversarial networks. *arXiv preprint arXiv:1810.10863*, 2018.

[115] Xinyue Zhu, Yifan Liu, Jiahong Li, Tao Wan, and Zengchang Qin. Emotion classification with data augmentation using generative adversarial networks. In *Pacific-Asia conference on knowledge discovery and data mining*, pages 349–360. Springer, 2018.

[116] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017.

[117] Swapnaa Jayaraman, Caitlin M Fausey, and Linda B Smith. The faces in infant-perspective scenes change over the first year of life. *PloS one*, 10(5):e0123780, 2015.

[118] Elizabeth M Clerkin, Elizabeth Hart, James M Rehg, Chen Yu, and Linda B Smith. Real-world visual statistics and infants' first-learned object names. *Phil. Trans. R. Soc. B*, 372(1711):20160055, 2017.

[119] David Gunning. Explainable artificial intelligence (xai). *Defense Advanced Research Projects Agency (DARPA), nd Web*, 2(2), 2017.

[120] Dumitru Erhan, Yoshua Bengio, Aaron Courville, and Pascal Vincent. Visualizing higher-layer features of a deep network. *University of Montreal*, 1341(3):1, 2009.

[121] Matthew D Zeiler, Dilip Krishnan, Graham W Taylor, and Rob Fergus. Deconvolutional networks. In *2010 IEEE Computer Society Conference on computer vision and pattern recognition*, pages 2528–2535. IEEE, 2010.

[122] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer, 2014.

[123] Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller. Striving for simplicity: The all convolutional net. *arXiv preprint arXiv:1412.6806*, 2014.

[124] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2921–2929, 2016.

[125] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pages 618–626, 2017.

[126] David Bau, Bolei Zhou, Aditya Khosla, Aude Oliva, and Antonio Torralba. Network dissection: Quantifying interpretability of deep visual representations. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6541–6549, 2017.

[127] Bolei Zhou, David Bau, Aude Oliva, and Antonio Torralba. Interpreting deep visual representations via network dissection. *IEEE transactions on pattern analysis and machine intelligence*, 41(9):2131–2145, 2018.

[128] Karin H James, Susan S Jones, Linda B Smith, and Shelley N Swain. Young children's self-generated object views and object recognition. *Journal of Cognition and Development*, 15(3):393–401, 2014.

[129] Linda B Smith, Swapnaa Jayaraman, Elizabeth Clerkin, and Chen Yu. The developing infant creates a curriculum for statistical learning. *Trends in cognitive sciences*, 2018.

[130] Antonio Torralba and Alexei A Efros. Unbiased look at dataset bias. In *Computer*

Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on, pages 1521–1528. IEEE, 2011.

[131] Yann LeCun, Fu Jie Huang, and Leon Bottou. Learning methods for generic object recognition with invariance to pose and lighting. In *Computer Vision and Pattern Recognition, 2004.*, volume 2, pages II–104. IEEE, 2004.

[132] Kevin Lai, Liefeng Bo, Xiaofeng Ren, and Dieter Fox. A large-scale hierarchical multi-view rgb-d object dataset. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 1817–1824. IEEE, 2011.

[133] Ali Borji, Saeed Izadi, and Laurent Itti. ilab-20m: A large-scale controlled object dataset to investigate deep learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2221–2230, 2016.

[134] Xiaofeng Ren and Matthai Philipose. Egocentric recognition of handled objects: Benchmark and analysis. In *Computer Vision and Pattern Recognition Workshops, 2009. CVPR Workshops 2009. IEEE Computer Society Conference on*, pages 1–8. IEEE, 2009.

[135] Vincenzo Lomonaco and Davide Maltoni. Core50: a new dataset and benchmark for continuous object recognition. In Sergey Levine, Vincent Vanhoucke, and Ken Goldberg, editors, *Proceedings of the 1st Annual Conference on Robot Learning*, volume 78 of *Proceedings of Machine Learning Research*, pages 17–26. PMLR, 13–15 Nov 2017.

[136] Sameer A Nene, Shree K Nayar, Hiroshi Murase, et al. Columbia object image library (COIL-100). *Technical report CUCS-005-96*, 1996.

[137] Jan-Mark Geusebroek et al. The amsterdam library of object images. *International Journal of Computer Vision*, 61(1):103–112, 2005.

[138] Pierre Moreels and Pietro Perona. Evaluation of features detectors and descriptors based on 3d objects. *International Journal of Computer Vision*, 73(3):263–284, 2007.

[139] Silvio Savarese and Li Fei-Fei. 3d generic object categorization, localization and pose estimation. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8. IEEE, 2007.

[140] Arjun Singh, James Sha, Karthik S Narayan, Tudor Achim, and Pieter Abbeel. Bigbird: A large-scale 3d database of object instances. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pages 509–516, 2014.

[141] Giulia Pasquale, Carlo Ciliberto, Lorenzo Rosasco, and Lorenzo Natale. Object identification from few examples by improving the invariance of a deep convolutional neural network. In *Intelligent Robots and Systems (IROS)*, pages 4904–4911. IEEE, 2016.

[142] Eugenio Culurciello and Alfredo Canziani. e-Lab video data set. https://engineering.purdue.edu/elab/eVDS/, 2017.

[143] Leland McInnes, John Healy, Nathaniel Saul, and Lukas Grossberger. Umap: Uniform manifold approximation and projection. *The Journal of Open Source Software*, 3(29):861, 2018.

[144] Sameer A Nene, Shree K Nayar, Hiroshi Murase, et al. Columbia object image library (COIL-20). *Technical report CUCS-006-96*, 1996.

[145] Uniform manifold approximation and projection (umap). https://github.com/lmcinnes/umap.

[146] Sven Bambach, David J Crandall, Linda B Smith, and Chen Yu. Active viewing

in toddlers facilitates visual object learning: An egocentric vision approach. In *Proceedings of the 38th Annual Conference of the Cognitive Science Society*, 2016.

[147] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2818–2826, 2016.

[148] Alejandro Betancourt, Miriam M López, Carlo S Regazzoni, and Matthias Rauterberg. A sequential classifier for hand detection in the framework of egocentric vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 586–591, 2014.

[149] Kenichi Ohki, Sooyoung Chung, Yeang H Ch'ng, Prakash Kara, and R Clay Reid. Functional imaging with cellular resolution reveals precise micro-architecture in visual cortex. *Nature*, 433(7026):597, 2005.

[150] Shai Sabbah, John A Gemmer, et al. A retinal code for motion along the gravitational and body axes. *Nature*, 546(7659):492, 2017.

[151] Carolyn B Mervis, Cynthia A Mervis, Kathy E Johnson, and Jacquelyn Bertrand. Studying early lexical development: The value of the systematic diary method. *Advances in infancy research*, 1992.

[152] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[153] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1492–1500, 2017.

[154] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

[155] K. Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2015.

[156] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *International Conference on Learning Representations*, 12 2014.

[157] Eleanor Rosch, Carolyn B Mervis, Wayne D Gray, David M Johnson, and Penny Boyes-Braem. Basic objects in natural categories. *Cognitive psychology*, 8(3):382–439, 1976.

[158] Barbara Landau, Linda B Smith, and Susan S Jones. The importance of shape in early lexical learning. *Cognitive development*, 3(3):299–321, 1988.

[159] Larissa K Samuelson and Linda B Smith. They call it like they see it: Spontaneous naming and attention to shape. *Developmental Science*, 8(2):182–198, 2005.

[160] Lisa Gershkoff-Stowe and Linda B Smith. Shape and the first hundred nouns. *Child development*, 75(4):1098–1114, 2004.

[161] Larissa K Samuelson. Statistical regularities in vocabulary guide language acquisition in connectionist models and 15-20-month-olds. *Developmental psychology*, 38(6):1016, 2002.

[162] Meagan N Yee, Susan S Jones, and Linda B Smith. Changes in visual object recognition precede the shape bias in early noun learning. *Frontiers in psychology*, 3:533, 2012.

[163] Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A. Wichmann, and Wieland Brendel. Imagenet-trained CNNs are biased towards tex-

ture; increasing shape bias improves accuracy and robustness. In *International Conference on Learning Representations*, 2019.

[164] Robert Geirhos, Carlos RM Temme, Jonas Rauber, Heiko H Schütt, Matthias Bethge, and Felix A Wichmann. Generalisation in humans and deep neural networks. In *Advances in neural information processing systems*, pages 7538–7550, 2018.

[165] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[166] Ji Shouling, Li Jinfeng, and Li Bo Du Tianyu. Survey on techniques, applications and security of machine learning interpretability. *Journal of Computer Research and Development*, 56(10):2071, 2019.

[167] Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, Franco Turini, Fosca Giannotti, and Dino Pedreschi. A survey of methods for explaining black box models. *ACM computing surveys (CSUR)*, 51(5):1–42, 2018.

[168] Radu Bogdan Rusu and Steve Cousins. 3d is here: Point cloud library (pcl). In *2011 IEEE international conference on robotics and automation*, pages 1–4. IEEE, 2011.

[169] Ronaldo Iachan. Plane sampling. *Statistics & probability letters*, 3(3):151–159, 1985.

[170] A Milne. The centric systematic area-sample treated as a random sample. *Biometrics*, 15(2):270–297, 1959.

[171] Peter J Kostelec and Daniel N Rockmore. Soft: So (3) fourier transforms. *Department of Mathematics, Dartmouth College, Hanover, NH*, 3755, 2007.

[172] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[173] Turing-nlg: A 17-billion-parameter language model by microsoft. https://www.microsoft.com/en-us/research/blog/turing-nlg-a-17-billion-parameter-language-model-by-microsoft/.