Exploring Correspondences Between Gibsonian and Telic Affordances for Object Grasping

Aniket Tomar
Colorado State University
Fort Collins, CO USA
aniket.tomar@colostate.edu

Nikhil Krishnaswamy
Colorado State University
Fort Collins, CO USA
nkrishna@colostate.edu

Abstract

Coined by Gibson, the notion of an *affordance* has become an important problem in AI, beyond its original formulation in psychology. Following both Gibson as well as Pustejovsky's notion of the *telic* affordance, we explore the abilities of certain machine learning models to learn how to group objects in terms of their affordances, by training models to classify objects into different "grasp classes," using both 3D meshes and human annotations. We train two classifiers, one to predict which objects are grasped similarly based on their geometries, and one to predict which objects are grasped similarly according to human annotations. We find that there is no clear correspondence between the embedding spaces of the two classifiers and hypothesize that this is due to the data capturing fundamentally different distributions of affordances with respect to objects, one representing Gibsonian affordances, and the other, telic affordances.

1 Introduction

J. J. Gibson introduced the concept of affordances in 1977 to describe the functional and ecological relationship between organisms and their environments (Gibson, 1977). To say an object "affords" an action is to say that the object facilitates the action being taken with it. *Gibsonian* affordances are those behaviors afforded due to the physical object structure, and can be directly perceived by animals. Pustejovsky (2013) introduced the notion of a *telic* affordance, or a behavior that is conventionalized due to an object's typical use or purpose.

Predicting affordances has proven to be a difficult task for artificial intelligence. Humans more often learn about affordances (e.g., "cups contain things," "spoons are used for stirring") by using objects or watching them being used rather than being told about or reading about them. Hence this information is often absent from or sparsely distributed in linguistic corpora. Tasks like human-object interaction (HOI) or video action recognition bear a more direct relation to affordances, but often datasets for these tasks conflate an affordance with any type of action that can be taken with an object, rather than a specific relation denoting *what the object offers the agent* a la Gibson.

The problem is also important for commonsense reasoning about actions. Word embedding-based approaches show low vector similarity between object words and action words, even when those actions are commonly associated with objects (e.g., "stir" and "spoon"). Encoded knowledge of habitats and affordances has been shown to be useful, even over small sample sizes, at determining similarities between objects based on their known behaviors, and at acquiring partial information about novel objects (Pustejovsky and Krishnaswamy, 2021). However, this encoded knowledge is usually handcrafted (e.g., in VoxML (Pustejovsky and Krishnaswamy, 2016)), and difficult to acquire at scale. Studies have shown that Large Language Models possess some commonsense world knowledge and can "guess" the affordances and properties of many objects, but they cannot reason about the relationship between these properties and affordances (Rogers et al., 2020). For example, BERT "knows" that people can walk into houses, and that houses are big, but it cannot infer that houses are bigger than people. It would then seem that if a house was smaller than a person BERT would still suggest that it can be walked into.

In this paper we explore the ability of machine learning methods to learn grasping affordance-based correspondences between objects based on their geometries, and based on human annotations of how objects are grasped for purpose or use. We then train classifiers on the two datasets and explore an affine transformation technique a la McNeely-White et al. (2020) to explore correspondences between the embedding spaces of the two classifiers.

We find that the two classifier embedding spaces are not easily mapped using an affine transformation, and hypothesize that this is because the two datasets are capturing fundamentally different information about the object affordances: Gibsonian affordances vs. telic affordances, and show that these fall into distinct distributions, making a mapping between the spaces difficult.

Section 2 presents related work, including the MeshCNN method that is core to this research (Section 2.1). Section 3 presents our dataset and preprocessing. Section 4 presents our methodology and results. Section 5 discusses some of the implications of this finding, and Section 6 concludes with future research directions.

2 Related Work

Learning to predict object affordances based on perceptible attributes is important for many use cases such as autonomous robot learning. There exist a number of popular human-object interaction (HOI) datasets for image and video recognition (e.g., Chao et al., 2015; Chao et al., 2018; Goyal et al., 2017), and methods that learn object affordances from visual features exist (Fang et al., 2018; Nagarajan et al., 2019; Xiao et al., 2019), but few attempts have been made to use 3D data like polygonal meshes or point clouds that can allow for direct access to information such as the structure of the object to ground other models. Polygonal meshes explicitly and efficiently capture both shape surface and topology in detail. High-quality 3D polygonal mesh data, unlike images, is difficult to acquire at scale. However, there have been significant recent advances in capturing 3D data or synthesizing views from a few images, largely based on Neural Radiance Fields (Yu et al., 2020; Jain et al., 2021; Lin et al., 2021; Ye et al., 2021), in stylizing a sample mesh based on a text prompt (Michel et al., 2021) and in reconstructing high quality polygonal mesh surfaces from acquired 3D point cloud data (Hanocka et al., 2020a; Metzer et al., 2021). This suggests that 3D data and polygonal meshes themselves are suitable for a number of AI tasks.

CNNs have been very successful at many computer vision tasks, because the inductive biases in a CNN are well-suited to images and the significant amount of image data available allows for learning invariant representations. The success of CNNs in 2D perception suggests that they can also be exploited in learning 3D representations of objects from images. However, this has proven to be a challenging task. The limited availability of 3D data makes it challenging to learn 3D representations using a CNN, and the additional computation required to accommodate the additional dimension often renders the process infeasible. Directly using the convolution operation on 3D data is also challenging because of the absence of an implicit neighborhood and uniformity as in images and non-Euclidean geometry of 3D data.

2.1 MeshCNN

MeshCNN (Hanocka et al., 2019) is an adaptation of convolutional neural networks for the analysis of 3D triangular meshes. MeshCNN uses specialized convolution and pooling operators analogous to the convolution and pooling operators of conventional CNNs, thereby importing the benefits of these well understood models to 3D meshes. These operators are designed such that they can directly operate on mesh edges (akin to how conventional CNNs can operate on pixels) in a task-aware fashion, unlike previous work on making the convolution operation intrinsic to the mesh (Masci et al., 2015; Henaff et al., 2015; Boscaini et al., 2016; Sinha et al., 2016; Maron et al., 2017), or using a convolution operation on point cloud-based representations (Qi et al., 2017; Guerrero et al., 2018; Li et al., 2018), or work that involved first transforming 3D data into a regular representation on which a convolution operator could be applied (Su et al., 2015; Kalogerakis et al., 2017; Ruizhongtai Qi et al., 2016; Wu et al., 2015; Brock et al., 2016; Tchapmi et al., 2017).

A convolution operator can be applied on unambiguously ordered input features in the neighborhood so that the learned features are invariant. The conventional image convolution operator can be directly applied to images because images are represented as a regular grid with inherent neighborhood, features and ordering, which is not true for which is not true for irregular and non-uniform 3D triangular meshes. To design the convolution operator for meshes in MeshCNN, the authors define a neighborhood for each edge that the operator operates on as edges contained in the faces incident on that edge. The vertices are ordered counter-clockwise. This ordering is ambiguous, which the authors address by defining input features of an edge as a 5-dimensional feature—the dihedral angle, two inner angles and two edge-length ratios between the edge and the perpendiculars for each face from the edge. Further, the authors aggregate the four incident edges that make a ring around the edge being operated on into two pairs of edges which have an ambiguity and generate new features by applying simple symmetric functions like summation on each pair. Thus, the neighborhood, features and ordering are defined in a way that a convolution operator can be applied to the edges and can learn invariant features. The pooling operation is defined as the collapse of incident edges to a point on the edge being operated on. The edges are put in a priority queue and edges with features having the smallest norm are pooled first making the pooling operation task-aware. Using these operators, MeshCNN has demonstrated good performance on a number of different learning tasks including segmentation and classification (Hanocka et al., 2020b; Wiersma et al., 2020; Vosylius et al., 2020), including using fewer parameters and compute time than comparable methods. Subsequently, we explore the ability of MeshCNN on an affordance-based classification task.

3 Dataset

This section describes our data collection process. Section 3.1 describes how we collected ground-truth data from human annotations selecting for telic affordances. Section 3.2 describes our collection and preprocessing of object geometries.

We first began by selecting a test set of common household objects graspable by a single hand. These objects, all found in a kitchen and therefore reminiscent of common problems in this domain (e.g., Damen et al. (2018)), include: *bottle*, *mug*, *knife*, *bowl*, *plate*, *wine glass*, *pen*, *apple*, *jar*, *spoon*, *fork*, *glass*, *teapot*, *banana*, *pan*.

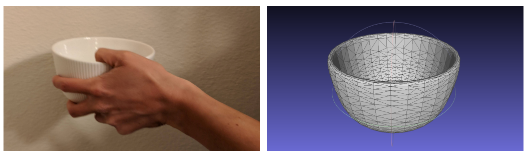


Figure 1: Bowl being grasped and bowl geometry.

Figure 1[L] shows a bowl being grasped in a typical fashion. [R] shows a geometry of a bowl object.

3.1 Human Annotation

We created a survey to elicit the canonical grasp pose for each object. The survey was posed as a multiple-choice questionnaire: "*Consider how your hand is posed while grasping each object for typical use. Then, for each object, select all other objects which are grasped using a similar hand pose.*" This phrasing, particularly the phrase "for typical use," was chosen to elicit the *telic* affordance for each of these objects. For each object, annotators could select which of the 15 objects satisfied the question, allowing for multiple objects to be selected as being grasped similarly to the one in question.

We had 28 annotators take the survey in total, resulting in 28 15-dimensional k-hot vectors. Each object was assumed to be grasped like itself, allowing us to keep indices constant across all objects. We used standard statistical techniques for identifying outliers, such as z-score filtering (with a z-score of 5) and normalization (Rousseeuw and Hubert, 2011). Because a single outlier can make the standard deviation large, it is common to use the median of all absolute deviations from the median (MAD) as a more robust measure of the scale (Leys et al., 2013). We computed the Kraemer kappa reliability score (Kraemer, 1980), to account for more than 2 annotators and the variable number of choices each annotation may have made, resulting in $\kappa \approx .32$, indicating "fair agreement" according to Landis and Koch (1977).

3.2 Meshes and Preprocessing

For each object, we collected 40 3D meshes (e.g., see Figure 1) from public repositories[1].

These meshes were all converted to the Wavefront 3D Object file format (`.obj`). Following this, we standardized the varying number of faces in each mesh to approximately 8,000 by subtriangulation to create additional faces or edge fusion to remove edges as necessary. Following this process, each mesh had approximately 15,000 edges.

Following this we used the open-source MeshLab tool to clean up each mesh by removing islands (faces unconnected to other faces), zero faces (3 edges bounding a topological one-dimensional hole), and non-manifold meshes (non-continuous meshes or meshes violating properties of Euclidean space at close resolution, such as by having crossing edges). We then visually inspected each mesh to make sure that this process did not cause the mesh to be deformed beyond recognition of its original identity: that is, a cleaned mesh of a bowl still needed to visually resemble a bowl to a human observer in order to make valid comparisons to MeshCNN's capabilities.

Finally, we validated each mesh with MeshCNN itself. MeshCNN has been demonstrated with known benchmark datasets, e.g., the SHREC

[1] `cgtrader.com`, `turbosquid.com`, `free3D.com`

dataset (Avola et al., 2018), but when training and testing on new meshes, there is a chance that the MeshCNN pooling operation (see Section 2.1) in one of the early convolutional layers will cause the resulting feature map equivalent to be non-manifold when it enters later layers. Therefore we trained MeshCNN on each mesh for 10 epochs on a dummy classification task with only 1 class, but the initial mesh and pooling resolution set to the same values to be used in the actual classification task. We then discarded any mesh that threw an error due to the aforementioned property of the pooling operator.

4 Methodology

This section describes our methodology. Section 4.1 describes the derivation of grasp classes from the ground truth annotations. Section 4.2 describes the classifier trained over the human annotations, and Section 4.3 describes the training of the MeshCNN model. Section 4.4 describes the results of the 2 classifiers. Section 4.5 describes the linear mapping procedure we used to make the embedding spaces directly comparable. We focus on neural network techniques in order to produce embeddings that can be made directly comparable using linear transformations, as it is not yet known whether linear mapping techniques between embeddings are applicable to embeddings derived from other methods like Random Forests.

4.1 Deriving Grasp Classes

In order to assess MeshCNN's ability to classify objects according to their graspability, we needed to organize our ground truth object annotations into classes according to their grasp poses.

We first summed the 28 vectors for each object into single vectors for that object. The resulting matrix can then be treated as a co-occurrence matrix. For each object (i.e., each row), we computed the Positive Pointwise Mutual Information (PPMI) with each other object (i.e., each column), according to $PPMI(a,b) = max\left(ln\left(\frac{P(a,b)}{P(a)P(b)}\right), 0\right)$, and then used Euclidean distance as a similarity measure to find the similarity between each object.

We then ordered object pairs based on the computed similarity scores and assigned the objects in the pairs starting with the most similar to the same class. For pairs where one object was already assigned to a class but the second object was not, we put the second in the same class. When both objects in a pair had not yet been assigned to a class,

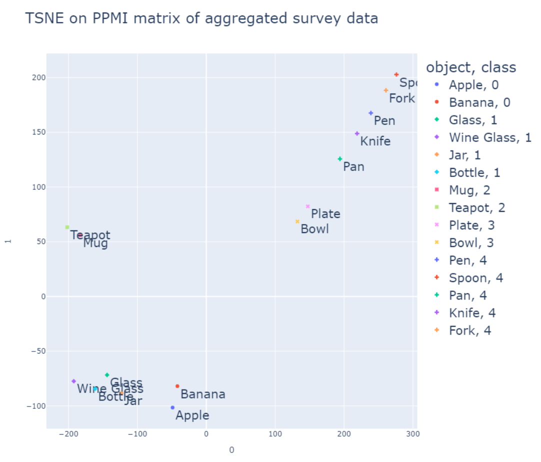


Figure 2: TSNE plot of object PPMI vectors.

we assigned both to a new class, and repeated until all the objects were assigned to a grasp class. To check the validity of this assignment we plotted a TSNE plot for the aggregated object PPMI vectors (Figure 2).

Table 1 shows the resulting grasp classes, which can be denoted by a description of the hand pose. Following terminology used in occupational therapy (Kamakura et al., 1980), which has since been adopted by the robotics community (Feix et al., 2015) Class 0 is a *spherical* grasp class, holding a fruit as if for consumption. Class 1 is the similar *cylindrical* grasp, a canonical pose when holding a glass for drinking. Class 2 contains the only objects in the dataset with "ear" handles joined to the object at both ends, which both use the *hook* grasp. Class 3 contains two objects typically held from the side or bottom (e.g., cf. Figures 1 and 3), which use the *palmar pinch*. Class 4 is a *tripod grasp*, and contains objects where the hand is held as if eating with a spoon or writing with a pen.

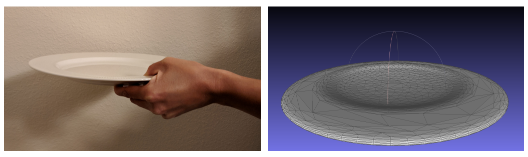


Figure 3: Plate being grasped and plate geometry. Bowl and plate form class 3.

4.2 Human Annotation Classifier

We trained a classifier to classify every vector representing an object from every human annotation into one of the 5 different grasp classes. Because the grasp classes had different number of objects, the classes also had different number of training vectors, making the dataset imbalanced. To cor-

Grasp Class	Objects
0	Apple, Banana
1	Bottle, Wine Glass, Glass, Jar
2	Mug, Teapot
3	Bowl, Plate
4	Spoon, Fork, Knife, Pen, Pan

Table 1: The 5 classes derived from the assignment scheme.

Hyperparameter	Value
`input size`	15
`hidden layer size`	200
`# classes`	5
`# epochs`	60
`learning rate`	0.2
`batch size`	46*5
`optimizer`	Adam

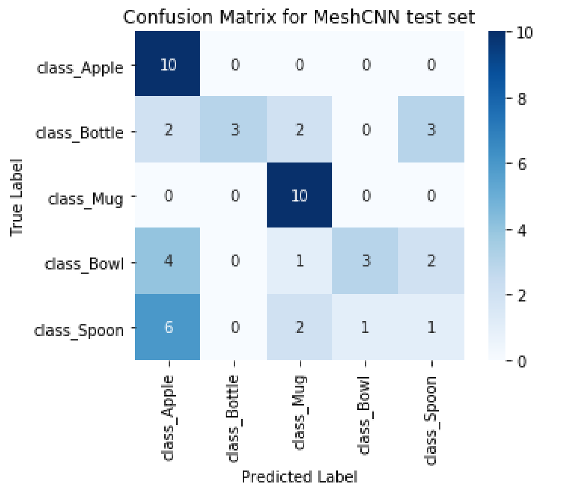
Table 2: Hyperparameters for Human Annotation classifier.

rect this imbalance, we randomly discarded excess training vectors from each class to achieve a sample balanced across classes, resulting in 56 training vectors per class (280 training object vectors total). We then divided the data into 82% training and 18% test splits, corresponding to 46 samples per class in the training set (230 samples total) and 10 samples per class in the test set (50 samples total). We built an MLP classifier in PyTorch, using grid search to tune hyperparameters, arriving at the hyperparameter set shown in Table 2.

4.3 MeshCNN Classifier

To classify the meshes according to grasp class, we trained an instance of MeshCNN using empirically-derived hyperparameters (shown in Table 3). `flip edges` refers to a data augmentation technique used by MeshCNN where a percentage of edges in the mesh are selected randomly and flipped[2]. `slide verts` refers to a similar data augmentation technique achieved by sliding vertices along the mesh surface. With the meshes in our dataset, this was a cause of the problems with non-manifold intermediate representations that we encountered during preprocessing (see Section 3.2), and so we did not use this hyperparameter.

[2]What is meant by "edge flipping" is well beyond the scope of this paper but a detailed treatment is given by Cheng and Jin (2015).



	class_Apple	class_Bottle	class_Mug	class_Bowl	class_Spoon
class_Apple	10	0	0	0	0
class_Bottle	2	3	2	0	3
class_Mug	0	0	10	0	0
class_Bowl	4	0	1	3	2
class_Spoon	6	0	2	1	1

Figure 4: Confusion matrix of the MeshCNN classifier test output.

4.4 Classifier Results

Using the given architectures and hyperparameter combinations above, the human annotation classifier achieved a test accuracy of 72%. This result was validated using a Random Forest classifier as well. The MeshCNN classifier, despite the extensive preprocessing, achieved only a test accuracy of 54%. Figure 4 shows the confusion matrix for the MeshCNN classifier. Here there are only two classes that achieve high classification accuracy: Class 0 (the spherical grasp class) and Class 2 (the hook grasp class). It is observed that topologically the two objects in each class have an obvious correspondence: both apples and bananas are topological spheroids while both teapots and mugs are topological toroids.

4.5 Linear Mapping Between Embedding Spaces

The poor performance of MeshCNN on this task necessarily raised questions about what the network was learning in this case. MeshCNN has advertised effectiveness on related tasks, such as classifying whether a vase has a handle (Hanocka et al., 2019) using similar data sizes, which made the relative difficulty of the grasp class task curious. In addition, the comparatively better but still middling performance of the classifier over the human annotations, combined with the relatively low agreement between annotators suggested that different humans use different heuristics when assessing the telic qualities of objects.

Previous research (McNeely-White et al., 2020) into the properties of embedding spaces has demon-

Hyperparameter	Value
`pool res`	15000, 15000, 15000, 15000
`# conv filters`	32, 64, 128, 128
`# neurons in FC layer`	200
`normalization`	group
`# resnet blocks`	1
`flip edges`	0.2
`slide vertices`	0
`# augmentations`	20
`# epochs with initial LR`	100
`# epochs with LR decay`	50
`# input edges`	15600
`batch size`	1
`optimizer`	Adam

Table 3: Hyperparameters for MeshCNN.

strated that, in closed-set tasks where a fixed set of final-layer labels is shared between two networks A and B, some level of interchangeability is in fact expected, up to a matrix $M_{A \to B} \in \mathbb{R}^{d_A} \times \mathbb{R}^{d_B}$ that minimizes the distance between paired points in $\mathbb{R}^{d_A} \times \mathbb{R}^{d_B}$ feature space that correspond to the same label. In a geometric sense, this is equivalent to asking, given two objects A and B, are they likely to be the "same" when deformed under, at most, a warping or affine transformation? Here, however, the objects are not meshes a la a MeshCNN object type classification task, but points in high dimensional vector space. If the grasp classes we derived can in fact be represented as roughly equivalent subspaces in both the human-annotation MLP embedding space and the MeshCNN embedding space, then the poor test performance of MeshCNN could simply be attributed to overfitting to the training data, so we set out to evaluate if this was in fact the case.

The hypothesis here was that if the MeshCNN embeddings can be transformed into the MLP embedding space such that the R^2 coefficient of determination is high enough for the training pairs, then poor performance can be attributed to overfitting to the training data. If not, then the more likely explanation is that the two representation spaces are underlyingly different due to fundamental differences in what the training data itself represents.

We retrieved all 200-dimensional embeddings for each input (training and test) from each of the two classifiers. Because we now wanted to evaluate equivalency between the *trained* embedding spaces, we use all data, including the embeddings representing the training inputs to the respective classifiers, to compute the mapping, in order to minimize the distance between points that each classifier "knows" belong to the correct class. Because there were 350 training inputs to the MeshCNN classifier and only 280 inputs to the human-annotation classifier, we discarded 70 randomly selected extra inputs until we had 1-to-1 paired embeddings representing 56 pairs each per grasp class. We divided these into the same 82:18 train/test split used in the MLP classification, resulting in 46 train and 10 test embedding vectors per grasp class.

To compute our linear mapping, we used an MLP regressor with no hidden layer (i.e., a multivariate linear model) as an affine mapping from one embedding space to another. The MeshCNN embeddings were the inputs and the MLP embeddings, as the classifier with higher accuracy and therefore presumably better-defined subspaces, serve as the outputs. This process maps the individual MeshCNN representations as closely as possible to their MLP-space equivalents. Since all embeddings come from a network whose final layer is a 5-node softmax activation representing the 5 grasp classes, this attempts to align the embedding spaces in which the grasp classes are represented by the respective models as closely as possible.

4.5.1 Linear Mapping Results

The regressor process was largely unsuccessful in aligning the two embedding spaces. $R^2 = 0.06$ on the training set and when the test embeddings were premultiplied by the computed mapping matrix, $R^2 = 0.02$. Even after mapping, the two embedding spaces remained almost entirely orthogonal. indicating that the two classifiers had learned fundamentally different representations, and that the poor test performance of MeshCNN was most likely not due to overfitting on the training data (more on this in Section 5).

Figure 5 shows a 3D TSNE plot of the vectors extracted from the two embedding spaces *after* the MeshCNN embeddings were mapped into the MLP embedding space.

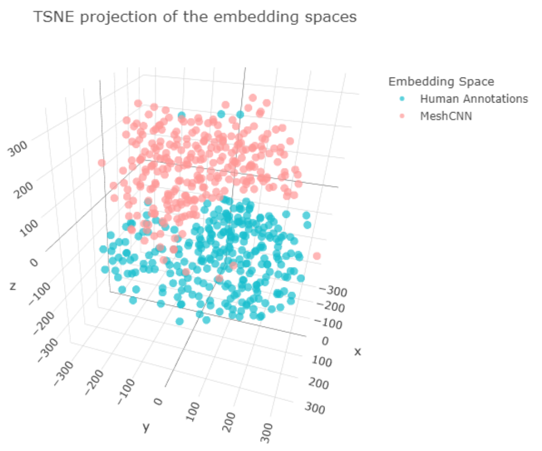


Figure 5: 3D TSNE plot of all embedding vectors, colored by original embedding space.

5 Discussion

Recall that the ground-truth grasp classes were derived from human annotations of objects that were designed to specifically elicit judgments on the *use or purpose* of the object, i.e., the telic affordance (Section 3.1). 3D meshes alone, however, capture none of this information. There may be some geometric correspondences that correlate with typical purpose-denoting grasps, such as handles, but the geometry itself, being a representation of *structure*, is naturally Gibsonian.

In Figure 5, we see two almost completely separable regions. If the MeshCNN classifer had actually overfitted to its training data, we would expect to see far more of the MeshCNN embeddings—the pink points—map closely to a subset of the human annotation embeddings—the blue points—because that is the same data that the output labels were derived from. Instead, the MeshCNN embeddings were mapped into the MLP embedding space (as seen by the fact that the entire convex hull of the MLP embeddings, including outliers at the top of the plot, encompasses nearly all the MeshCNN embeddings), but remain neatly separated from the bulk of the MLP embeddings, including the paired outputs that the input embeddings were trained against.

This suggests that MeshCNN learned certain representations of Gibsonian affordances but, being trained against telic affordance labels, did not have the information available in the structure of the mesh itself to learn appropriate Gibsonian-telic correlations. Meanwhile, the MLP trained over human annotations of telic affordances learned different information. We surmise, therefore, that Gibsonian and telic affordances represent related but fundamentally different ways of interpreting the same sets of objects.

This can be further confirmed by examining the nearest neighbors for specific objects within each embedding subspace.

Figure 6 shows the 55 nearest neighbors[3] of a representative *bowl* object (a member of grasp class 3) in each of the respective embedding spaces. In the bottom cluster, marked with circles, showing the data from the human annotation MLP embedding space, we see that the vast majority of nearest neighbor objects are either other bowls, or plates (the other member of grasp class 3). There are a few other neighbors belonging to other grasp classes, which we attribute to the disagreement the human annotators themselves showed, but these are overwhelmingly outnumbered by neighbors that belong the correct cluster.

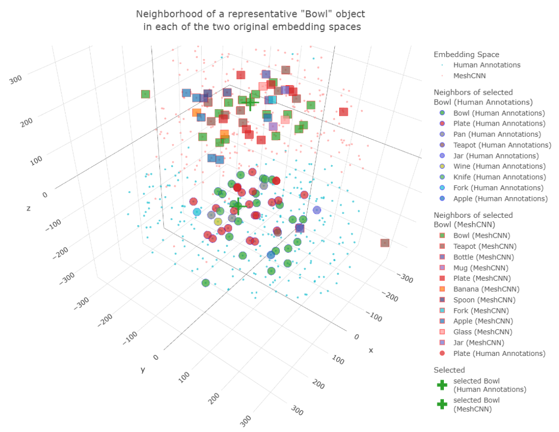


Figure 6: Nearest neighbors of representative *bowl* object across both embedding types.

Meanwhile in the top cluster, marked by squares, the nearest neighbors of a representative *bowl* object are much more diverse, roughly equally divided between more bowls and plates but also bottle, jars, and even teapots.

Nevertheless, when grasped for a typical use or purpose as the human annotators were asked (e.g., drinking from a bottle vs. filling a bowl), the actual

[3]Because there are 56 objects in a class when training and testing sets are put together.

hand pose is markedly different (e.g., see Figure 7).

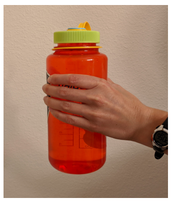


Figure 7: Bottle being grasped.

Figure 8 shows nearest neighbors of a *mug* object, which is a member of class 2, one of the classes on which MeshCNN actually performed well. Most neighbors here, among both types of embeddings, are mugs and teapots (the other class 2 member). In fact, among the nearest neighbors of the MeshCNN embeddings are *MLP embeddings* of teapots and mugs, indicating that MeshCNN did learn a representation of the handle geometry correlated with that type of grasp.

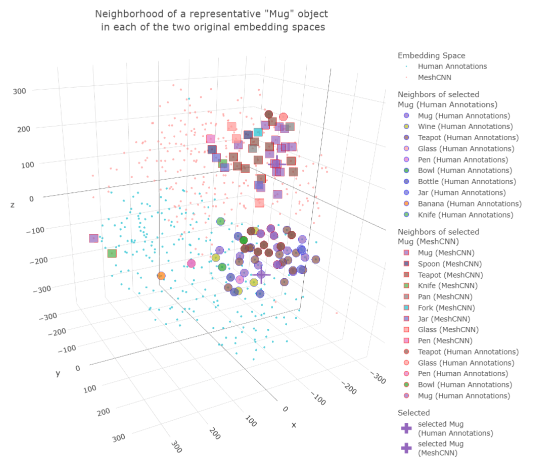


Figure 8: Nearest neighbors of representative *mug* object across both embedding types.

Grasping is typically a Gibsonian affordance, based on object structure, as would be encoded in a geometric mesh representation, but grasping for a particular use or purpose implies a *telic* affordance. Encoding information this way, as our human annotations did, appears to result in a markedly different representation from the geometric representation learned by MeshCNN. Even with the same output labels, models trained on this differing data do not appear to be learning equivalent representations that can be correlated to relationships between Gibsonian and telic affordances.

5.1 Implications for Action Recognition

An affordance is not just any action taken with an object (i.e., not every human-object interaction exploits the object's affordances). An affordance is a distinct action possibility that an object allows an agent to take, that is more particular to that object than would arise by chance. In particular, for *human*-object interaction, the human manipulators, i.e., the hands, play a critical role in determining how an object is used. Therefore simple object detection is not enough, and false positives on human-object interaction detection tasks are often the result of detecting the presence of an object in an image when it is *not* being held for typical use or purpose, or in a telic-enabling fashion (e.g., see the discussion of false positives in Gkioxari et al. (2018).) The ability to recognize and detect Gibsonian vs. telic affordances and, critically, the difference between the two, will be an important point in future successes in activity recognition and evaluation of human-object interaction tasks.

6 Conclusion and Future Work

In this paper we hope to have demonstrated that an embodied task like affordance classification is still difficult for even specialized models like MeshCNN that can operate directly over 3D data. We have shown that the problem becomes more difficult if affordances are characterized by a telic/Gibsonian distinction and that even a single afforded behavior such as grasping, when thought about in telic terms, carries quite different information from the same affordance viewed from a purely Gibsonian perspective. A broader implication is that Gibsonian and telic affordances may carry fundamentally different information about an object. Shape, the correlate for Gibsonian affordances encoded in geometries, underspecifies use, or telic affordances,

One specific angle for future work is examining the possibility of getting telic affordance information out of a mesh, which would make both Gibsonian and telic affordances encodable using meshes. When comparing our human annotations to the 3D mesh, one piece of information explicitly singled out by the human annotations was the pose of the hand. This information was nowhere to be found in the 3D meshes. Think about a cup on the table. It can potentially afford anything such as drinking, pushing, etc. The final action cannot be predicted, e.g., by an HOI classifier, unless coupled with a grasp pose. Methods like ContactOpt (Grady et al., 2021) and HandsFormer (Hampali et al., 2021) offer the possibility of fitting a 3D mesh of a hand to an object in either image or mesh format. Retrieving either the mesh or the joints of the fitted hand may potentially provide mesh- or mesh-like 3D information that could be provided to a method like MeshCNN to increase performance on tasks like affordance classification.

References

Danilo Avola, Marco Bernardi, Luigi Cinque, Gian Luca Foresti, and Cristiano Massaroni. 2018. Exploiting recurrent neural networks and leap motion controller for the recognition of sign language and semaphoric hand gestures. *IEEE Transactions on Multimedia*, 21(1):234–245.

Davide Boscaini, Jonathan Masci, Emanuele Rodolà, and Michael Bronstein. 2016. Learning shape correspondence with anisotropic convolutional neural networks.

Andrew Brock, T. Lim, James Ritchie, and Nick Weston. 2016. Generative and discriminative voxel modeling with convolutional neural networks.

Yu-Wei Chao, Yunfan Liu, Xieyang Liu, Huayi Zeng, and Jia Deng. 2018. Learning to detect human-object interactions. In *2018 ieee winter conference on applications of computer vision (wacv)*, pages 381–389. IEEE.

Yu-Wei Chao, Zhan Wang, Yugeng He, Jiaxuan Wang, and Jia Deng. 2015. Hico: A benchmark for recognizing human-object interactions in images. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1017–1025.

Siu-Wing Cheng and Jiongxin Jin. 2015. Edge flips in surface meshes. *Discrete & Computational Geometry*, 54(1):110–151.

Dima Damen, Hazel Doughty, Giovanni Maria Farinella, Sanja Fidler, Antonino Furnari, Evangelos Kazakos, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, et al. 2018. Scaling egocentric vision: The epic-kitchens dataset. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 720–736.

Kuan Fang, Te-Lin Wu, Daniel Yang, Silvio Savarese, and Joseph J. Lim. 2018. Demo2vec: Reasoning object affordances from online videos. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2139–2147.

Thomas Feix, Javier Romero, Heinz-Bodo Schmiedmayer, Aaron M Dollar, and Danica Kragic. 2015. The grasp taxonomy of human grasp types. *IEEE Transactions on human-machine systems*, 46(1):66–77.

James J Gibson. 1977. The theory of affordances. *Hilldale, USA*, 1(2):67–82.

Georgia Gkioxari, Ross Girshick, Piotr Dollár, and Kaiming He. 2018. Detecting and recognizing human-object interactions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8359–8367.

Raghav Goyal, Samira Ebrahimi Kahou, Vincent Michalski, Joanna Materzynska, Susanne Westphal, Heuna Kim, Valentin Haenel, Ingo Fruend, Peter Yianilos, Moritz Mueller-Freitag, et al. 2017. The" something something" video database for learning and evaluating visual common sense. In *Proceedings of the IEEE international conference on computer vision*, pages 5842–5850.

Patrick Grady, Chengcheng Tang, Christopher D Twigg, Minh Vo, Samarth Brahmbhatt, and Charles C Kemp. 2021. Contactopt: Optimizing contact to improve grasps. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1471–1481.

Paul Guerrero, Yanir Kleiman, Maks Ovsjanikov, and Niloy Jyoti Mitra. 2018. Pcpnet learning local shape properties from raw point clouds. *Computer Graphics Forum*, 37.

Shreyas Hampali, Sayan Deb Sarkar, Mahdi Rad, and Vincent Lepetit. 2021. Handsformer: Keypoint transformer for monocular 3d pose estimation ofhands and object in interaction. *arXiv preprint arXiv:2104.14639*.

Rana Hanocka, Amir Hertz, Noa Fish, Raja Giryes, Shachar Fleishman, and Daniel Cohen-Or. 2019. Meshcnn: a network with an edge. *ACM Transactions on Graphics (TOG)*, 38(4):1–12.

Rana Hanocka, Gal Metzer, Raja Giryes, and Daniel Cohen-Or. 2020a. Point2mesh. *ACM Transactions on Graphics (TOG)*, 39:126:1 – 126:12.

Rana Hanocka, Gal Metzer, Raja Giryes, and Daniel Cohen-Or. 2020b. Point2mesh: A self-prior for deformable meshes. *ACM Trans. Graph.*, 39:126.

Mikael Henaff, Joan Bruna, and Yann Lecun. 2015. Deep convolutional networks on graph-structured data.

Ajay Jain, Matthew Tancik, and Pieter Abbeel. 2021. Putting nerf on a diet: Semantically consistent few-shot view synthesis. pages 5865–5874.

Evangelos Kalogerakis, Melinos Averkiou, Subhransu Maji, and Siddhartha Chaudhuri. 2017. 3d shape segmentation with projective convolutional networks. pages 6630–6639.

Noriko Kamakura, Michiko Matsuo, Harumi Ishii, Fumiko Mitsuboshi, and Yoriko Miura. 1980. Patterns of static prehension in normal hands. *The American journal of occupational therapy*, 34(7):437–445.

Helena Chmura Kraemer. 1980. Extension of the kappa coefficient. *Biometrics*, pages 207–216.

J Richard Landis and Gary G Koch. 1977. The measurement of observer agreement for categorical data. *biometrics*, pages 159–174.

Christophe Leys, Christophe Ley, Olivier Klein, Philippe Bernard, and Laurent Licata. 2013. Detecting outliers: Do not use standard deviation around the mean, use absolute deviation around the median. *Journal of experimental social psychology*, 49(4):764–766.

Yangyan Li, Rui Bu, Mingchao Sun, Wei Wu, Xinhan Di, and Baoquan Chen. 2018. Pointcnn: Convolution on x-transformed points. In *NeurIPS*.

Chen-Hsuan Lin, Wei-Chiu Ma, Antonio Torralba, and Simon Lucey. 2021. Barf: Bundle-adjusting neural radiance fields. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5721–5731.

Haggai Maron, Meirav Galun, Noam Aigerman, Miri Trope, Nadav Dym, Ersin Yumer, Vladimir Kim, and Yaron Lipman. 2017. Convolutional neural networks on surfaces via seamless toric covers. *ACM Transactions on Graphics*, 36:1–10.

Jonathan Masci, Davide Boscaini, Michael Bronstein, and Pierre Vandergheynst. 2015. Geodesic convolutional neural networks on riemannian manifolds.

David McNeely-White, Benjamin Sattelberg, Nathaniel Blanchard, and Ross Beveridge. 2020. Exploring the interchangeability of cnn embedding spaces. *arXiv preprint arXiv:2010.02323*.

Gal Metzer, Rana Hanocka, Raja Giryes, and Daniel Cohen-Or. 2021. Self-sampling for neural point cloud consolidation. *ACM Transactions on Graphics*, 40:1–14.

Oscar Jarillo Michel, Roi Bar-On, Richard Liu, Sagie Benaim, and Rana Hanocka. 2021. Text2mesh: Text-driven neural stylization for meshes. *ArXiv*, abs/2112.03221.

Tushar Nagarajan, Christoph Feichtenhofer, and Kristen Grauman. 2019. Grounded human-object interaction hotspots from video (extended abstract). *ArXiv*, abs/1906.01963.

James Pustejovsky. 2013. Dynamic event structure and habitat theory. In *Proceedings of the 6th International Conference on Generative Approaches to the Lexicon (GL2013)*, pages 1–10.

James Pustejovsky and Nikhil Krishnaswamy. 2016. Voxml: A visualization modeling language. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 4606–4613.

James Pustejovsky and Nikhil Krishnaswamy. 2021. Embodied human computer interaction. *KI-Künstliche Intelligenz*, 35(3):307–327.

C. Qi, L. Yi, Hao Su, and Leonidas J. Guibas. 2017. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *NIPS*.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in bertology: What we know about how bert works. *Transactions of the Association for Computational Linguistics*, 8:842–866.

Peter J Rousseeuw and Mia Hubert. 2011. Robust statistics for outlier detection. *Wiley interdisciplinary reviews: Data mining and knowledge discovery*, 1(1):73–79.

Charles Ruizhongtai Qi, Hao Su, Matthias NieBner, Angela Dai, Mengyuan Yan, and Leonidas Guibas. 2016. Volumetric and multi-view cnns for object classification on 3d data. pages 5648–5656.

Ayan Sinha, Jing Bai, and Karthik Ramani. 2016. Deep learning 3d shape surfaces using geometry images. volume 9910, pages 223–240.

Hang Su, Subhransu Maji, Evangelos Kalogerakis, and Erik Learned-Miller. 2015. Multi-view convolutional neural networks for 3d shape recognition.

Lyne Tchapmi, Chris Choy, Iro Armeni, JunYoung Gwak, and Silvio Savarese. 2017. Segcloud: Semantic segmentation of 3d point clouds. pages 537–547.

Vitalis Vosylius, Andy Wang, Cemlyn Waters, Alexey Zakharov, Francis Ward, Loïc Le Folgoc, John R. G. Cupitt, Antonios Makropoulos, Andreas Schuh, Daniel Rueckert, and Amir Alansary. 2020. Geometric deep learning for post-menstrual age prediction based on the neonatal white matter cortical surface. In *GRAIL@MICCAI*.

Ruben Wiersma, Elmar Eisemann, and Klaus Hildebrandt. 2020. Cnns on surfaces using rotation-equivariant features. *ACM Transactions on Graphics (TOG)*, 39:92:1 – 92:12.

Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 2015. 3d shapenets: A deep representation for volumetric shapes. pages 1912–1920.

Tete Xiao, Quanfu Fan, Dan Gutfreund, Mathew Monfort, Aude Oliva, and Bolei Zhou. 2019. Reasoning about human-object interactions through dual attention networks. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 3918–3927.

Yufei Ye, Shubham Tulsiani, and Abhinav Gupta. 2021. Shelf-supervised mesh prediction in the wild.

Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. 2020. pixelnerf: Neural radiance fields from one or few images.