3D Object Recognition: An Experimental Review

Bernard Renardi
Faculty of Science and Engineering
Rijksuniversiteit Groningen
Groningen, The Netherlands
b.jo@student.rug.nl

Duygu Bayram
Faculty of Arts
Rijksuniversiteit Groningen
Groningen, The Netherlands
d.bayram.1@student.rug.nl

***Abstract*—Object Recognition is an important, and in most cases, an unavoidable task when it comes to designing Robots. There are many components that go into building a robust 3D Object Recognition System, with many approaches to choose from. In this project we employ an Instance Based Learning (IBL) approach and run experiments to optimize one hand-crafted and one deep learning model. The hand-crafted model compares the performances of two descriptors (GOOD, ESF), and four similarity functions (Motyka, Cosine, Intersection, Euclidean), along with experimenting with different K values for the K-NN algorithm for the classification. In the first phase of the experiment, we find that the ESF descriptor with the Motyka similarity function and the k = 1 value achieves the best and the fastest result. In the second phase, we compare this best performing model with deep learning approaches, OrthographicNetV2, or MobileNetV2 [2] and VGG16 [9]. We find that the MobileNetV2 deep learning model achieves the best performance. For the report, we focus on the Motyka and the Intersection functions, and the MobileNetV2 architecture as they showed more interesting results.**

***Index Terms*—3d object recognition, instance based learning, descriptors, similarity functions**

I. Introduction

The field of 3D object classification and recognition has been growing vastly in the last few years with the popularization of 3D sensors, the increased availability of 3D object databases, and the immense size of processed data. The methods developed for this purpose are applied in many domains such as robotics, which focuses on assisting robot movement based on the visual perception of the environment so that it can perform object manipulation to reach a goal. The goal is often that the robot must reliably recognize the object to a certain level of accuracy in less than the expected amount of time to process.

Robots cannot percept their surroundings as a human can, they rely on a representation method of vision called the point cloud [1]. The point cloud is a kind of data generated by the reflection of the lidar that can store 3D information. The point cloud can make up for the shortcomings of the traditional RGB camera that only provides 2D information, which cannot be used to effectively perform 3D object detection. Therefore, many methods currently exploit point clouds for 3D object detection. In order to identify the most preferable method for our datasets, we performed an experimental review.

This review is organized as follows: Section 2 reviews the existing works that were previously done based on available related literature. Section 3 presents all the steps performed and details of each step to generate the result of the experiment as well as the parameters being used. Section 4 shows an analysis of all the selected works, choosing the outperforming method for each experiment of the 3D object recognition, and merges all the gathered information in a general overview. Lastly, the conclusion is given in Section 5.

II. Related Work

Some of the notable methods from previous works include local feature scale-invariant feature (SIFT), Signature of Histogram of Orientation (SHOT), and Global Orthographic Object Descriptor (GOOD) that is built to be robust, descriptive and efficient to compute and use. The GOOD descriptor is implemented through the following steps: (i) using an object point cloud, (ii) applying the principal component analysis (PCA) method and determining the object three principal axes, and (iii) applying a disambiguation method to define the three principal axes directions and calculating a local reference frame (LRF) [2].With the LRF calculated, the next step is to concatenate the object orthographic projections in the three orthogonal planes.

Another review looks into 3D point cloud descriptors to investigate the existing approaches for extracting 3D point cloud descriptors, and to perform a thorough evaluation of the performance of several state-of-the-art 3D point cloud descriptors. The selected cloud descriptors are widely used in practice in terms of descriptiveness, robustness and efficiency detection [3]. Based on the review, we can pinpoint several approaches and their traits. Viewpoint Feature Histogram (VFH) outperforms Spin Image (SI) in terms of its fast processing time and its robustness to large surface noise. The Ensemble of Shape Function (ESF) outperforms the Shape Distribution on Voxel Surfaces (SDVS), VFH, Clustered Viewpoint Feature Histogram (CVFH), and Global Signature of Histogram of Orientation (GSHOT). GOOD outperforms VFH and ESF because of its robustness, efficiency, and real-time application suitability.

In open-ended domains, some methods related to object improvements are also being proposed to achieve a robust and online learning system capable of informing us of the procedure of the learning that takes place while the robot is running. Deep Convolutional Neural Networks (DCNNs), or often so-called Deep Learning, have achieved state-of-the-art results. The strength of DCNNs is their ability to learn richer representations than conventional hand-crafted representations. Tuning deep neural architectures to strike an optimal balance between accuracy and performance has been an area of active research for the last several years. Considerable work carried out by numerous teams has lead to dramatic improvements over early designs such as AlexNet [10], VGGNet [9], GoogLeNet [11], and ResNet [12] [4].

Sandler et al. (2018) [5] conduct an experiment on low-end devices to investigate the performance of some prominent architectures on certain object detection algorithms. The task specifically aims to detect masked individuals in images while leveraging cloud-based services to provide facial verification for the detected individuals. The authors perform a comparative analysis on a sequential Bi-layered CNN, VGG-16 CNN and MobileNetV2, and find that the MobileNetV2 outperforms other models with an accuracy of 99.2% [6].

III. Methodology

For the following experiments, we utilize the Robot Operating System (ROS) as the software development kit. We also use Ubuntu 18.04 for our operating system and ROS Melodic as our ROS version. Having all these requirement fulfilled, we can finally setup our virtual machine (VM) on the computer.

A. Installation and Setup

All required steps have been explained systematically on the provided classroom Github[1]. There are several parts for this step including:

- Clone and Compile ROS packages
- Add the folder as workspace
- Set up all dependency of packages
- Compile all packages

B. Offline 3D Object Recognition

The first task is to optimize offline 3D object recognition systems that take an object view as the input and produce the category label as the output. We intend to use an instance-based learning (IBL) approach to form new categories. From a general perspective, IBL approaches can be viewed as trying out combinations of 1-3 parameters on the Restaurant RGB-D Object Dataset to observe the performance of the learning process. The dataset consists of 10 different object categories that can be seen in Fig. 2 in Appendix, using the `pcl_viewer` function.

1) Object Representation: There are four different approaches available for the object descriptors: GOOD, ESF, VFH, GRSD. Based on the literature, we are able to minimize the combination of the output by choosing GOOD and ESF descriptors for this experiment. For the GOOD descriptor, we run the experiment for two different numbers of histogram bins where $\mathtt{n} \in [15, 25]$

[1]https://github.com/SeyedHamidreza/cognitive_robotics_ws

2) Similarity Measure: In terms of similarity measurement, we are provided with 14 different distance functions: Euclidean, Manhattan, χ^2, Pearson, Neyman, Canberra, KL divergence, symmetric KL divergence, Motyka, Cosine, Dice, Intersection, Bhattacharyya, Gower, and Sorensen.

Since the distance functions have unique characteristics for each type, based on the literature, we select four different distance functions,where two of them are closely related (Cosine and Euclidean), and two are non-related (Motyka and Intersection). We present the non-related ones to draw a contrast.

3) Classification Rule: To classify the object, we implement the K-Nearest Neighbour (K-NN) algorithm with different K values. For the setup, we want to see the effect of various values of K from the set K $\in$ [1, 3, 5]

At the end of the first experiment, we can expect to have 3 x 4 x 3 = 36 results from the combinations of the previous parameters. We experiment with the GOOD descriptor and the ESF descriptor separately to produce two outputs of 24 and 12 results respectively. By separating the folder output, we ensure that the results file will not overwrite each other.

`hand_crafted_results_GOOD`$\rightarrow$ GOOD folder
`hand_crafted_results_ESF`$\rightarrow$ ESF folder

C. Open-Ended Scenario

In this phase, we evaluate our findings from the previous experiment in open-ended learning systems with the goal of simultaneous recognition of a larger dataset without predefined categories. To this end, we deploy `simulated_teacher` by repeatedly picking unseen object views from the currently known categories as the representation of human knowledge, and present them to the agent for testing. Subsequently, with such simulation, we make it possible to conduct systematic, consistent and reproducible experiments for different approaches automatically.

1) Protocol Threshold: A protocol threshold defines how good the agent should learn categories, by setting a minimum accuracy score for the recognition. The default of the parameter is set to τ = 0.67, which means the recognition accuracy is at 67% or twice better than the error (error rate = 33%). We run the experiment and obtain several results with three different configurations of $\tau \in$ [0:7; 0:8; 0:9] while also changing the `random_sequence_generator:=false`.

2) Hand-Crafted Object Representation: We set the descriptor and the distance function parameters inside the launch file `simulated_user_hand_crafted_descriptor` according to the best results from the previous experiment.

3) Deep Learning Based Object Representation: Some options are introduced by the provided scipt for deep CNN, also called the base network. We implement MobileNetV2 and VGG16, and report on MobileNetV2 based on its better performance in the reviewed literature as well as our own results.

Scenario	Predictor	Bins	Distance	K	Instance Acc	Avg Acc	Total Time	Hardest Object	Object Accuracy	TP	FP	FN
1	ESF	-	cosine	1	95.77%	95.06%	4.859	Fork, Spoon	81.82%	294	13	13
2	ESF	-	cosine	3	95.44%	94.35%	4.768	Fork, Spoon	72.73%	293	14	14
3	ESF	-	cosine	5	94.79%	92.41%	4.755	Fork, Spoon	54.55%	291	16	16
4	ESF	-	motyka	1	96.74%	95.97%	4.153	Fork, Spoon	81.82%	297	10	10
5	ESF	-	motyka	3	95.77%	94.66%	4.336	Fork, Spoon	72.73%	294	13	13
6	ESF	-	motyka	5	95.11%	92.71%	4.48	Fork, Spoon	54.55%	292	15	15
7	ESF	-	euclidean	1	95.77%	95.06%	5.069	Fork, Spoon	81.82%	294	13	13
8	ESF	-	euclidean	3	95.44%	94.35%	5.002	Fork, Spoon	72.73%	293	14	14
9	ESF	-	euclidean	5	94.79%	92.41%	5.094	Fork, Spoon	54.55%	291	16	16
10	ESF	-	intersection	1	0.00%	0.00%	4.646	All	0.00%	0	307	307
11	ESF	-	intersection	3	0.00%	0.00%	4.517	All	0.00%	0	307	307
12	ESF	-	intersection	5	0.00%	0.00%	4.315	All	0.00%	0	307	307
13	GOOD	15	cosine	1	95.77%	95.06%	4.551	Fork, Spoon	81.82%	294	13	13
14	GOOD	15	cosine	3	95.44%	94.35%	4.438	Fork, Spoon	72.73%	293	14	14
15	GOOD	15	cosine	5	94.79%	92.41%	4.389	Fork, Spoon	54.55%	291	16	16
16	GOOD	15	motyka	1	96.74%	95.97%	4.488	Fork, Spoon	81.82%	297	10	10
17	GOOD	15	motyka	3	95.77%	94.66%	4.439	Fork, Spoon	72.73%	294	13	13
18	GOOD	15	motyka	5	95.11%	92.71%	4.298	Fork, Spoon	54.55%	292	15	15
19	GOOD	15	euclidean	1	95.77%	95.06%	5.039	Fork, Spoon	81.82%	294	13	13
20	GOOD	15	euclidean	3	95.44%	94.35%	5.052	Fork, Spoon	72.73%	293	14	14
21	GOOD	15	euclidean	5	94.79%	92.41%	4.498	Fork, Spoon	54.55%	291	16	16
22	GOOD	15	intersection	1	0.00%	0.00%	4.18	All	0.00%	0	307	307
23	GOOD	15	intersection	3	0.00%	0.00%	4.086	All	0.00%	0	307	307
24	GOOD	15	intersection	5	0.00%	0.00%	4.276	All	0.00%	0	307	307
25	GOOD	25	cosine	1	95.44%	94.16%	5.026	Fork, Spoon	72.73%	293	14	14
26	GOOD	25	cosine	3	93.81%	90.85%	5.122	Fork, Spoon	45.45%	288	19	19
27	GOOD	25	cosine	5	95.44%	92.90%	4.977	Fork, Spoon	54.55%	293	14	14
28	GOOD	25	motyka	1	96.42%	95.07%	5.027	Fork, Spoon	72.73%	296	14	14
29	GOOD	25	motyka	3	94.79%	91.76%	4.722	Fork, Spoon	45.45%	291	16	16
30	GOOD	25	motyka	5	95.77%	93.20%	4.813	Fork, Spoon	54.55%	294	13	13
31	GOOD	25	euclidean	1	95.44%	94.16%	6.104	Fork, Spoon	72.73%	293	14	14
32	GOOD	25	euclidean	3	93.81%	90.85%	6.024	Fork, Spoon	45.45%	288	19	19
33	GOOD	25	euclidean	5	95.44%	92.90%	5.793	Fork, Spoon	54.55%	293	14	14
34	GOOD	25	intersection	1	0.00%	0.00%	4.922	All	0.00%	0	307	307
35	GOOD	25	intersection	3	0.00%	0.00%	4.839	All	0.00%	0	307	307
36	GOOD	25	intersection	5	0.00%	0.00%	5.155	All	0.00%	0	307	307

TABLE I: Summary of Offline 3D Object Recognition

IV. Results and Discussion

After obtaining the results from both experiments, we conduct analysis on the two separate parts: (1) Offline 3D Object Recognition and (2) Open-Ended Scenario.

A. *Offline 3D Object Recognition*

The first section revealed 36 combinations of different results as mentioned previously. The summary of the results can be viewed in Table I.

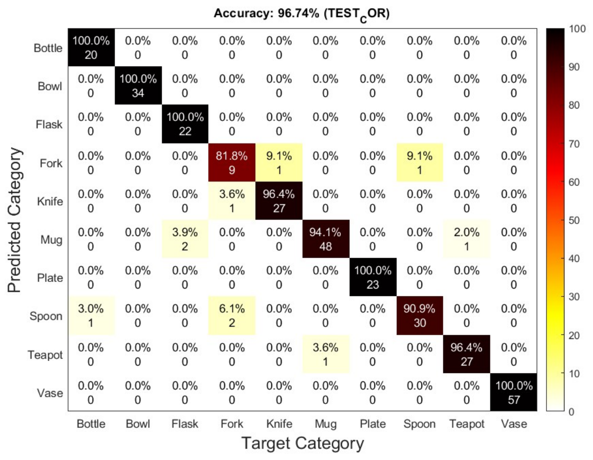


Fig. 1: Confusion Matrix for Scenario 4

Using the bash file, we can finish the running process in 2.88 minutes while the total average of all scenarios is 4.796 seconds. Overall, we can quantify that the best scenario is Scenario 4, which is the combination of ESF as the descriptor, with k = 1, and Motyka as the distance function. This gives an instance accuracy of 96.74% and an average accuracy of 95.97% with *Fork* as the hardest category to identify. It competes with the GOOD descriptor in Scenario 16 with `n` = 15 with almost the same result regarding the accuracy, lowest accuracy object, and its sensitivity level. However, there is a slight difference when it comes to the run time. While Scenario 4 takes 4.153 seconds to finish, Scenario 16 takes 4.488 seconds. As such, Scenario 4 outperforms Scenario 16 in this case. We can see the confusion matrix of our best scenario on Fig. 1

In order to understand why, first of all, we can analyze the difference between the ESF and the GOOD descriptors based on the structure of the algorithms and the way they are executed inside ROS. The ESF descriptor is an ensemble of ten 64-bin-sized histograms of shape functions describing characteristic properties of the point cloud, which means it has fixed 640 bins in total [7](Aldoma 2012). GOOD divides object up in a grid of `n × n` bins, for black and white objects, and for the RGB model, `m` is a constant that is set to the shape descriptor as (`m × n x n`). If we assign `n` = 15, we have 3 x 15 x 15 = 675 bins in total. This is why in terms of the run time, it takes slightly less time to run the ESF descriptor compared to the GOOD descriptor.

Moreover, it is important to address Motyka and its connnection with the changing of the k value.We know that using Motyka with any different descriptor would produce the highest result. By observing the summary results on Fig. 3 in Appendix for Scenario 4, we can see there are two objects in the *Mug* category being mislabeled as the *Flask* category by our detector. If we take a look at the object on Fig 4 in Appendix, we can see why the detector fails to identify the object as the key features are visually difficult to identify even for our eyesight compared to the Mug_Object19 on Fig. 4(a). However, as shown by Scenario 16, we can get different results by increasing the k value so the detector can identify the object better, even though the instance accuracy of the scenario decreases. This explains why the accuracy is inversely proportional to the k number in most cases, due to the overfitting in the K-NN classifier.

Finally, to explain the contrast between the Motyka result and the Intersection result, we want to look at the equation of this Intersection family of similarity functions. As taken from [8], given d as the number of bins in the histogram, let H(X) and H(Y) be the representation histogram of two different objects being compared, while P and Q be the normalized histogram for H(X) and H(Y) respectively. Then we have (1) for Intersection similarity as

$$S_{IS} = \sum_{i=1}^{d} min(P_i, Qi) \tag{1}$$

and we also have (2) for Motyka similarity as

$$S_{Mot} = \frac{\sum_{i=1}^{d} min(P_i, Qi)}{\sum_{i=1}^{d} (P_i + Q_i)} \tag{2}$$

Given by these equations, it can be observed that the Motyka performs better on handling data in lower magnitudes. For instance, if we have P={1, 2, 3} and Q={2, 4, 6}, then by definition we have

$$S_{IS} = min(1,2) + min(2,4) + min(3,6) = 1 + 2 + 3 = 6$$

$$S_{Mot} = \frac{min(1,2)}{1+2} + \frac{min(2,4)}{2+4} + \frac{min(3,6)}{3+6} = \frac{1}{3} + \frac{2}{6} + \frac{3}{9} = 1$$

Let us see when P={0.1, 0.2, 0.3} and Q={0.2, 0.4, 0.6}

$$S_{IS} = min(0.1,0.2) + min(0.2,0.4) + min(0.3,0.6)$$
$$= 0.1 + 0.2 + 0.3 = 0.6$$

$$S_{Mot} = \frac{min(0.1,0.2)}{0.1+0.2} + \frac{min(0.2,0.4)}{0.2+0.4} + \frac{min(0.3,0.6)}{0.3+0.6}$$
$$= \frac{0.1}{0.3} + \frac{0.2}{0.6} + \frac{0.3}{0.9} = 1$$

Therefore, we can draw the conclusion that the Intersection function is more susceptible to small changes and noise, while Motyka can still maintain the similarity level unbiased with less fluctuation.

B. Open-Ended Scenario

In the second phase of our experiments, we implement our best model as determined in the previous part for object recognition, into the hand-crafted descriptor simulation file, and compare the result with deep learning methods. For the setup, we define ESF as the descriptor, k=1, and Motyka as the distance function for the hand-crafted simulation. For the deep learning simulation, we choose VGG16 and MobileNetV2 and chose to report on the MobileNetV2 as it has shown the best performance. We can see a summary of the results from Fig 5 in Appendix as it compares to the hand-crafted model, and in Fig 6 in Appendix in comparison to the VGG16.

Generally, with the `protocol_threshold` = 0.67 fixed, MobileNetV2 clearly performs better than the hand-crafted simulation. In Table II, we are able to see, through the comparison between Scenario 1 and Scenario 5, that Scenario 5 needs less iterations to converge. It also has a lower average number of instances being stored of only about 6-7 objects per category, saving more space in the perceptual memory of the system. Moreover, for the global classification accuracy and average protocol accuracy, it produces 5.5% and 3.6% better results respectively. The reason behind this is because by feeding the CNN with the similarity data through the Motyka distance function, we can extract features more efficiently than a regular object descriptor would be able to. The CNN also offers a light weight of processing load.

To evaluate the effects of varying `protocol_threshold` $-(\tau)$ values, we change the parameter in the launch file three times where $\tau \in$ {0.7, 0.8, 0.9}. As explained previously, τ determines the level of accuracy allowed for the protocol. If we increase the τ value, it will limit the result only when the accuracy of the protocol is bigger than or equal to τ. Consequently, we have fewer categories learned because of that as we can see from Scenarios 1 and 4. On scenario 4, there are only 24 categories capable of achieving more than 90% protocol accuracy, there is also a noticeable increase in iterations. Therefore, we can see here that the iteration number is negatively correlated to τ.

Scenario	Type	Iterations	Categories	Instances	GS	ACS	Threshold
1	Hand-crafted	1333	51	7.804	0.8162	0.8436	0.67
2	Hand-crafted	1353	51	8.125	0.786	0.8039	0.7
3	Hand-crafted	1374	51	8.314	0.7997	0.8095	0.8
4	Hand-crafted	1979	24	8.765	0.8744	0.9334	0.9
5	Deep Learning	1325	51	6.353	0.9709	0.8797	0.67

TABLE II: Open-Ended Scenario Summary

V. Conclusion

3D Object Recognition is a crucial task in robotics with many varied approaches. We experimented with some hand-crafted methods (GOOD and ESF) and some deep learning methods (MobileNetV2 and VGG16), and reported the important portion of our findings. For the hand-crafted measures we found that the ESF descriptor with the Motyka similarity function and the k value set to 1 performs best. We also find that MobileNetV2 performs better than the hand-crafted model and VGG16 in the second phase. We find that the threshold has a decreasing affect on the iteration number and the number of categories learned.

Authors' Contributions

Bernard Renardi: ESF, MobileNetV2
Duygu Bayram: GOOD, VGG16
Authors contributed equally in other parts.

References

[1] Sun, F. (2022). Cognitive systems and information processing.

[2] Kasaei, S. H. (2020). OrthographicNet: A Deep Transfer Learning Approach for 3-D Object Recognition in Open-Ended Domains. IEEE/ASME Transactions on Mechatronics, 26(6), 2910-2921.

[3] Hana, X. F., Jin, J. S., Xie, J., Wang, M. J., & Jiang, W. (2018). A comprehensive review of 3D point cloud descriptors. arXiv preprint arXiv:1802.02297, 2.

[4] Alzubaidi, L., Zhang, J., Humaidi, A. J., Al-Dujaili, A., Duan, Y., Al-Shamma, O., ... & Farhan, L. (2021). Review of deep learning: Concepts, CNN architectures, challenges, applications, future directions. Journal of big Data, 8(1), 1-74.

[5] Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L. C. (2018). Mobilenetv2: Inverted residuals and linear bottlenecks. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 4510-4520).

[6] Lad, A. M., Mishra, A., & Rajagopalan, A. (2021, July). Comparative Analysis of Convolutional Neural Network Architectures for Real Time COVID-19 Facial Mask Detection. In Journal of Physics: Conference Series (Vol. 1969, No. 1, p. 012037). IOP Publishing.

[7] Aldoma, A., Marton, Z. C., Tombari, F., Wohlkinger, W., Potthast, C., Zeisl, B., ... & Vincze, M. (2012). Tutorial: Point cloud library: Three-dimensional object recognition and 6 dof pose estimation. IEEE Robotics Automation Magazine, 19(3), 80-91.

[8] Cha, S. H. (2007). Comprehensive survey on distance/similarity measures between probability density functions. City, 1(2), 1.

[9] Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.

[10] Krizhevsky, A., Sutskever, I., Hinton, G. E. (2017). Imagenet classification with deep convolutional neural networks. Communications of the ACM, 60(6), 84-90.

[11] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., ... Rabinovich, A. (2015). Going deeper with convolutions. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 1-9).

[12] He, K., Zhang, X., Ren, S., Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778).

Appendix

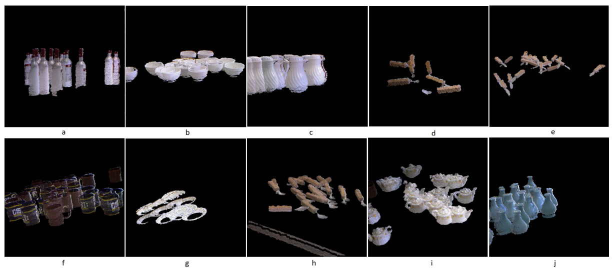
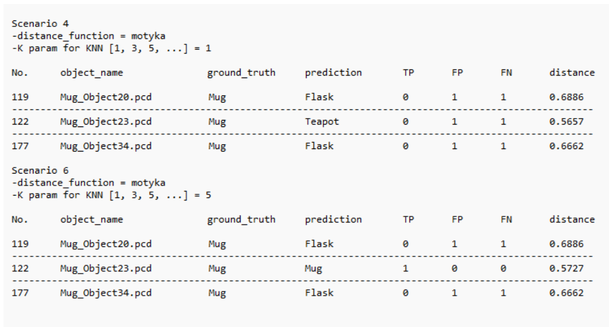
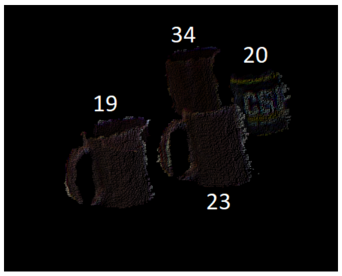


Fig. 2: Samples for each category in the Restaurant RGB Dataset. The categories are as follows: a) Bottle, b) Bowl, c) Flask, d) Fork, e) Knife, f) Mug, g) Plate, h) Spoon, i) Teapot, and j) Vase. The images are point cloud images using pcl_viewer.

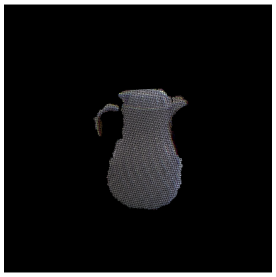


```
Scenario 4
-distance_function = motyka
-K param for KNN [1, 3, 5, ...] = 1

No.     object_name             ground_truth    prediction      TP      FP      FN      distance

119     Mug_Object20.pcd        Mug             Flask           0       1       1       0.6886
------------------------------------------------------------------------------------------------
122     Mug_Object23.pcd        Mug             Teapot          0       1       1       0.5657
------------------------------------------------------------------------------------------------
177     Mug_Object34.pcd        Mug             Flask           0       1       1       0.6662

Scenario 6
-distance_function = motyka
-K param for KNN [1, 3, 5, ...] = 5

No.     object_name             ground_truth    prediction      TP      FP      FN      distance

119     Mug_Object20.pcd        Mug             Flask           0       1       1       0.6886
------------------------------------------------------------------------------------------------
122     Mug_Object23.pcd        Mug             Mug             1       0       0       0.5727
------------------------------------------------------------------------------------------------
177     Mug_Object34.pcd        Mug             Flask           0       1       1       0.6662
```

Fig. 3: Sample summary of Scenario 4 (above) and 6 (below) with different k number

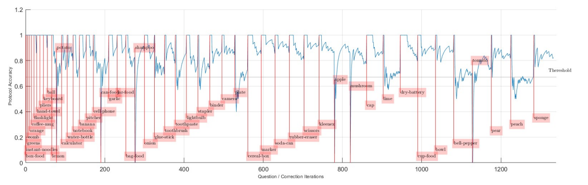


(a) Mug with object number

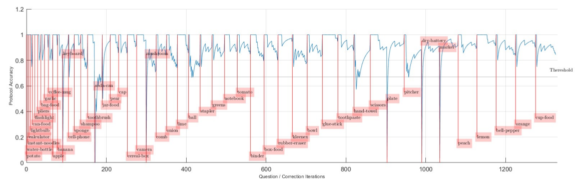


(b) Sample flask image

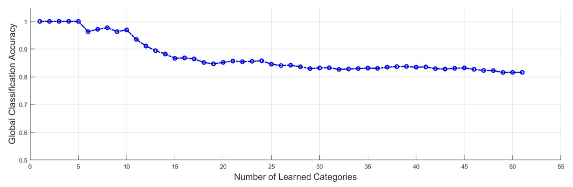
Fig. 4: Sample of failing object from Scenario 4 and 6



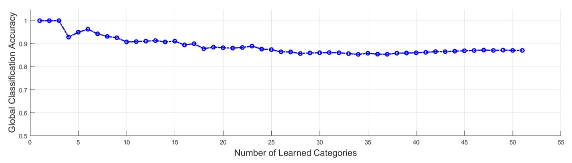
(a) Accuracy results of the hand-crafted model.



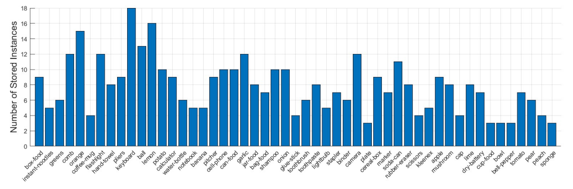
(b) Accuracy results of MobileNetV2.



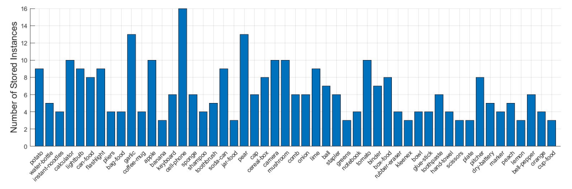
(c) Global F1 results of the hand-crafted model.



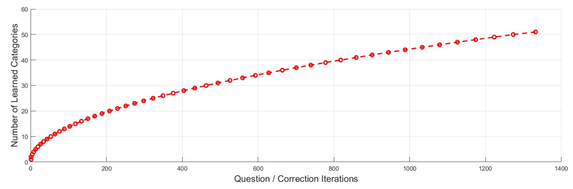
(d) Global F1 results of MobileNetV2.



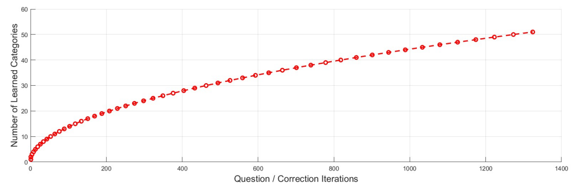
(e) Number of stored instances in the hand-crafted model.



(f) Number of stored instances in MobileNetV2.

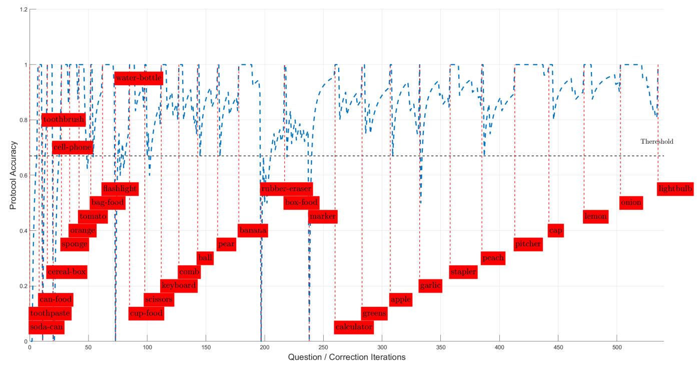


(g) Number of learned categories in the hand-crafted model.

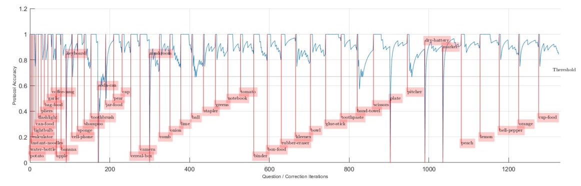


(h) Number of learned categories in MobileNetV2.

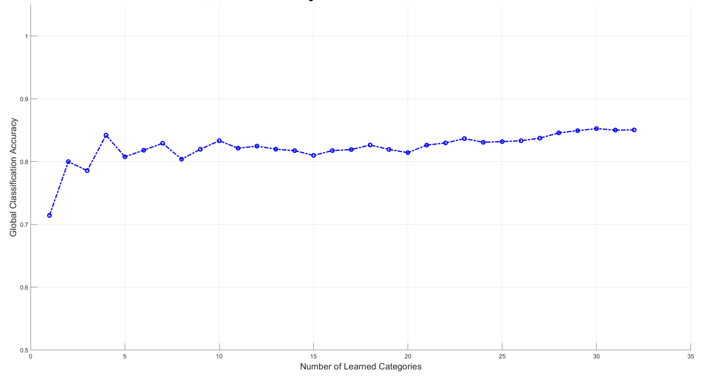
Fig. 5: Results of the hand-crafted model and the MobileNetV2 approach.



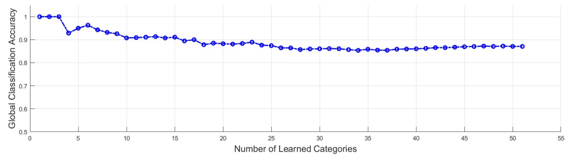
(a) Accuracy results of VGG16.



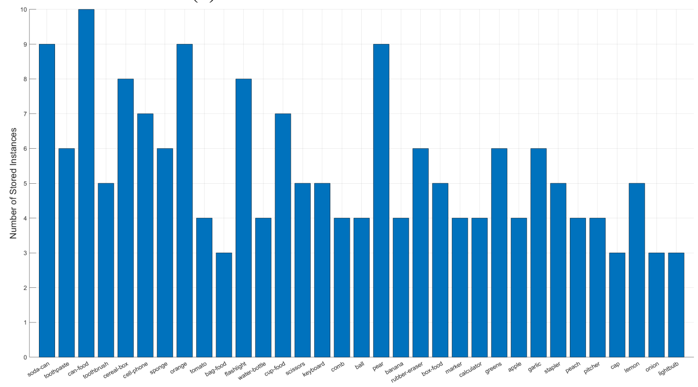
(b) Accuracy results of MobileNetV2.



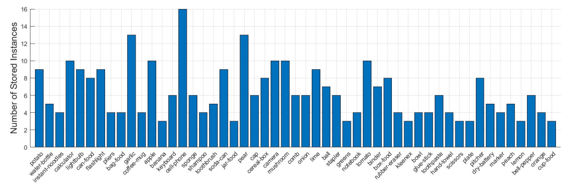
(c) Global F1 results of VGG16.



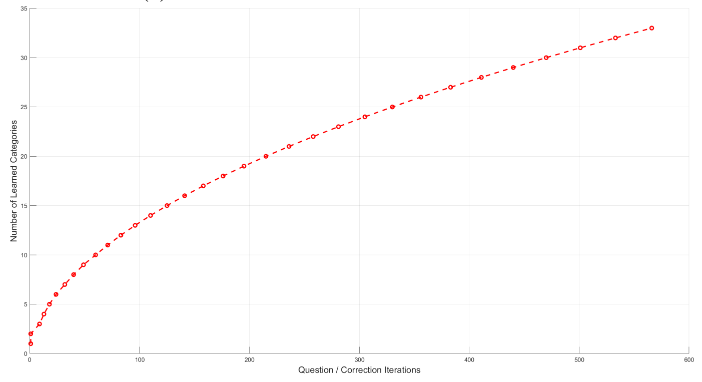
(d) Global F1 results of MobileNetV2.



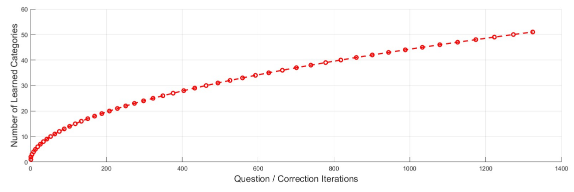
(e) Number of stored instances in VGG16.



(f) Number of stored instances in MobileNetV2.



(g) Number of learned categories in VGG16.



(h) Number of learned categories in MobileNetV2.

Fig. 6: Results of VGG16 and MobileNetV2.