Open-Ended 3D Object Recognition: Investigating the Effect of Deep Features, Color Spaces, and Similarity Measures

Bachelor's Project Thesis

Andreea-Mihaela Țoca, s3454177, a.toca@student.rug.nl,
Supervisor: Dr. Hamidreza Kasaei, hamidreza.kasaei@rug.nl
Department of Artificial Intelligence

Abstract: Object recognition is a challenging task in unpredictable environments, where service robots need to react fast, and cooperate with human-users. Lack of training data and limited use of object information, such as only shape information, reduces the robustness of object recognition algorithms. This study investigates the influences of shape information, color spaces, and similarity measures in open-ended 3D object recognition. Towards this end, three experimental setups, color-only, shape-only, and color-shape, were evaluated in both offline and online setups. Following the OrtohraphicNet construction, the experiments were conducted using two deep learning networks, Mobilenet-v2, and VGG16 in combination with four parameters: orthographic image resolution, similarity distance functions, k-values (in a KNN algorithm), and three color spaces. In the online evaluation, extensive experiments showed that the k-nearest neighbor algorithm had a neglectable influence over the system's performance. It was also observed that the color information improves the performance over shape information alone, color-only RGB obtaining the best result. The online evaluation showed a decrease in accuracy compared with the offline results, predicting the lack of robustness of classical train-test experiments in open-ended domains. The results' hierarchy from offline followed in the online evaluation as well.

1 Introduction

For the past decades, significant advancements have been made in the field of computer vision. State-of-the-art algorithms are getting closer to mimic human behavior, automating the biological vision system. Computer vision deals with the following tasks: scene reconstruction, object recognition, image segmentation, 3D pose estimation, motion tracking, object classification, and more. This paper will solely focus on object recognition. One application of such an algorithm can be evaluated in service robots.

Compared with industrial robots that need to make repetitive motions in standardized environments, service robots must make free motions in various conditions [1]. They should interact with humans in natural ways, go to specified places, behave in a human-like manner, perform manipulation tasks, and recognize the surrounded environment [2]. Since the service robots need to operate in unpredictable circumstances, it is not enough to implement a system that successfully completes one task (e.g., recognizing a limited amount of objects). Considering the advancements in computer vision, we can still see goal-oriented approaches that perform well only on specific tasks. The lack of robustness is on the basis of insufficient data and the limitations of neural network algorithms.

It does not matter how big the training dataset is, it is nearly impossible that the dataset represents the complexity of the real world [3]. A. Yuille and C. Liu [3] discusses the limitations of current Deep Nets (deep neural networks), arguing that for many real-life scenarios, especially when the human interaction is involved, to capture the world's complexity, the training has to be conducted on an exponentially large dataset. Therefore, this will imply more computational power, time, and memory on behalf of the system. The goal is to remove these bottlenecks while achieving a robust working system. There is no solution for all; nevertheless, improvements have been made in all directions. *Open-*

ended object category learning is an advocated solution for minimizing the memory overflow in considerably large datasets.

In this paper, open-ended learning refers to continuous learning in dynamic environments. It gives the machines the ability to learn new object categories that have not been seen before while integrating further information in the preexisted knowledge. This approach increases the system's robustness by helping the robot quickly face unknown situations, which brings it closer to human behavior. The robot does not know in advance what will be the unknown object categories or when they will occur in the environment or what information will be available to support the learning. This approach also facilitated the training process. The system architect does not have to integrate all object instances and does not have to worry that a new object category means restarting the training process. Apart from programming, the human plays a teacher role for the robot, supporting the learning process with constant feedback.

Aside from continuous learning, other types of learning bring their advantages in computer vision. For example, transfer learning can transfer the knowledge applicable to one task to solve different tasks. The main difference between continuous learning and transfer learning is that the latter only focuses on the current goals, not being concerned about keeping the ability to solve previous tasks [4]. One application of this type of learning is easily seen in neural networks. Once a network is trained on a task, to demonstrate the system's generalization power, the network weights are used for future tasks in different settings. Another type of learning is online learning, which incorporated continuous learning. Online learning is suited for situations where the action should be done immediately, so the updates are done on a single data point basis [4].

Open-ended learning in service robots represents a vital tool for future developments that will allow the robot to adapt to new circumstances and gain more experiences. We can expect the robots to learn to recognize object categories and grasp affordances seamlessly to facilitate the human-robot interaction naturally. Hence, a long term perspective with emphasis on open-ended domains is preferred for service robots because it helps the robots to react in a live environment as humans do.

Besides the learning mechanism, it is essential to look at what features make an object unique and easy to categorize. Humans have an incredible ability to remember thousands of objects. In doing this, they create a visual map of the target object, retaining it as a whole and the unique features that characterize it. This flexible representation allows for recognizing familiar objects under various conditions (e.g., texture, color, lighting, shape) [5].

The main research question of this study is:

Which combination of color spaces, deep shape feature, similarity measures, and deep learning architectures performs the best in an open-ended 3D object recognition task?

Towards this goal, this paper aims to analyze two essential features of objects in object recognition tasks: shape and color. For this, the first step is to create a global object reference frame from the given object using Global Orthographic Object Descriptor (GOOD) developed by Kasaei et al. [6].

In the second step, using the orthographic projection method to create the object's views, we use the projections as input for a convolutional neural network and extract view-wise object representations. The result is then used for both learning and recognition processes. In this project, we mainly use an instance-based learning approach and a K-nearest neighbor recognition to evaluate the object representations. Since the object is represented as a feature vector, we use different distance functions to determine the similarity between the target object and the other classes.

The performance is assessed through the accuracy metric calculated by averaging the number of instances correctly classified.

2 Related work

The object recognition task can be divided into four steps:

0. Object detection
1. Object representation
2. Object evaluation in the recognition module
3. Category classification

The dataset used in this study have all the objects already segmented from the scene, so the first step *Object detection* is not discussed. Henceforth, no related work is provided for this topic. Category classification is an essential step in object recognition tasks; however, in this study, the problem is tackled using only two approaches similarity measures and k-nearest neighbor algorithm. Both are discussed in the next section. No other related works about this topic are discussed.

2.1 Research on object representation

The way the object is represented is essential. The approach can contain too much unnecessary information that increases the noise or too little information that does not benefit the architecture. In real-life applications, an easy way to extract the relevant information from a scene is crucial since a sophisticated approach is time-consuming and abolishes the robot's role. Han et al. [7] mentions three classes of 3D point cloud object descriptors: local-based descriptor, global-based descriptor, hybrid-based descriptor. Local descriptors encode distinct patches of the object in a piece-wise style. The authors conduct a comparison among multiple descriptors concluding that the two global descriptors Ensemble of Shape Functions (ESF) [8] and Viewpoint Feature Histogram (VFH) [9] are suitable for real-time object recognition, and provide a right balance between accuracy and running efficiency. A local descriptor, SHOTColor [10], also shows satisfactory results, but the higher computational time required can pose a disadvantage.

Finding the best compromise solution between computation time and complexity, the Global Orthographic Object Representation (GOOD) [6][11] is both efficient and descriptive. GOOD is a lightweight object descriptor that creates the object representation from a 3D point cloud. It performs a principal component analysis on the point cloud object and extracts the depth information. In a similar work, Kasaei et al. [12] investigates the importance of shape feature, color constancy, color spaces for open-ended 3D category learning, adjusting the GOOD descriptor by incorporating the color information. They conclude that the color information, combined with the shape information, shows a significant improvement over the shape feature alone. They also investigates three color spaces, RGB, YUV, and HSV, observing that the HSV color space outperforms the other two. This work represents a starting point and the inspiration for this study. The mechanisms of the GOOD descriptor is presented in the next section.

2.2 Research on features

The object representation should be descriptive enough, to be able to recognize the same object from multiple perspectives. This leads to the question *what is the relevant information that should be encoded?* We have already seen in Kasaei et al. [12] that the color information with the shape information improves the results in object recognition tasks. Other studies have shown that image classification has better results when only one feature, such as color, is used for classification [13]. S. Gowda and C. Yuan [13] proposes a model, ColorNet, that investigates this approach and shows the state of the art results can be obtained with different color spaces than RGB and smaller densenets. Even though the results are satisfactory, the approach is not as close as possible to the human visual system, since it does not use all the information of the object on the scene. Such models tend to reinforce biases towards a single feature and reduce the model's robustness in different environments. A transfer learning approach is also excluded since the performance drastically drops in different setups. The bias problem is also discussed by Geirhos et al. [14], where they observe a texture bias in the CNNs trained on ImageNet dataset [15]. When the CNN is presented with a cat shape but elephant texture, the model wrongly classifies the image as an elephant and not as a cat as humans do. Their solution to improve the performance to overcome unseen image distortions is to adapt the dataset used by applying AdaIN style transfer [16].

Cognitive science shows that the most important cue for human object recognition is shape [17], and color, lighting, texture, orientation, and size are used as secondary cues. Hence, the global shape is the most used feature in object recognition applications, which draw many researchers' attention and motivated them to test the boundaries of this hypothesis [18] [19].

2.3 Research on convolutional neural networks

After the object representation frame is computed based on the calculation of a set of features, the obtained representation is sent to the object recognition module. The recognition module consists of a deep CNN. CNNs have been progressively used for 3D object recognition. However, the fixed number of categories and the increased size of the training data represent limitations that make integrating a CNN in open-ended domain difficult. A way to solve this problem could be to transfer the features from a pre-trained CNN and then fine-tuning to the target dataset [20]. This solution increases the network's robustness, but it still not enough to operate in an online environment. H. Kasaei [21] proposes, OrtographicNet, an online classifier that, in combination with a pre-trained CNN, can handle open-ended object category learning and recognition. His approach starts by constructing the unique reference frame of the object extracting three orthographic projections. Each projection is fed into CNN to obtain the view-wise features that are merged into a global feature of the object using an element-wise pooling function. Concerning the CNNs, he analyzes the performance of MobileNet-v2 [22], VGG16-fc1 [23], and ResNet50 [24]. In terms of CNN performance, MobileNet-v2 and ResNet50 perform similarly. VGG16-fc1 does not master all the categories learned in 4 out of 10 experiments. This research extends the OrtographicNet by analyzing the effect of the color information on 3D object recognition.

3 Methods

In the following section, each of three modules that complete the object recognition task is discussed in the context of this study:

1. Object Representation
2. Object Recognition
3. Object Classification

3.1 Object representation

3.1.1 Object descriptor

This module consists of calculating a set of features that give the mathematical notation for the recognition module. The point cloud representation of an object consists of a set of points $p_i : i = \{1, ..., n\}$. Each point has 3D coordinates [x,y,z] and RGB information. The approach proposed by Kasaei et al. [6] starts by constructing a global object reference frame (RF) for the object. Towards this goal, they perform a principal component analysis on the point cloud of the object and find the eigenvectors of the object [v_1, v_2, v_3]. By looking at the eigenvectors' direction, we can conclude the X, Y, and Z axes. Afterward, the views of the object are generated using the orthographic projection method. They only look at the front, top, and right-side views since the other three, rear, bottom, and right-side are their mirrors'. The projections are divided into $n \times n$ bins, which are used to compute a normalized distribution matrix by counting how many points fall within each bin. By stringing the matrix's rows together, we obtain a histogram that is fed into the deep learning architecture. Each projection is feed individually.

The projection with the maximum information is considered to have the highest entropy. For studying the effect of color spaces on object recognition, only the view with the highest entropy is chosen. Figure 3.1 shows the three projections of a vase. It can be observed that the front-view contains the

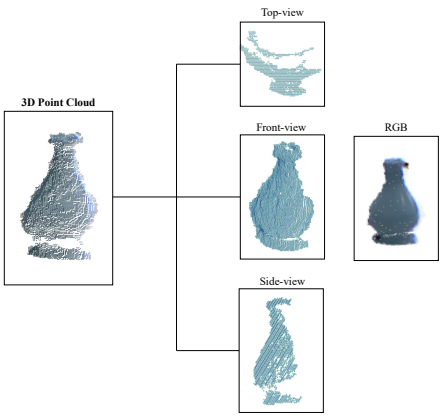

Figure 3.1: The three orthographic projections of a sample point cloud of a vase. The view with the highest entropy (front-view) is presented in the RGB color space.

most information, which means that it is presented in the color space. As an example, we can see the RGB image of the front-view.

3.1.2 Color spaces

In many neural network applications for image classification, the models do not perform any color transformation to the image and instead only use the RGB image directly for the classification. The datasets' nature mostly influences this bias, since most of them consist of color images represented in RGB format. Color spaces are an essential tool for image classification, providing the medium to describe colors as numbers. We convert the color information into three different color spaces: RGB, YUV, HSV.

RGB is the most popular color space consisting of three channels, R (red), G (green), B (blue). Each channel has a range of values from 0 to 255, encoded in 8-bit each.

The second color space that is discussed in this study is YUV. YUV is mainly used for television transmission. It has one channel for luminance (Y, which refers to brightness) and two chrominance channels. The V channel takes values from 0 to 255, while the U and V channels range from -128 to +127 (they determine the color itself).

Another color space that is analyzed in this paper is HSV. HSV stands for hue, saturation, and value. It encodes the color (hue) as a combination of saturation and brightness (value). The S and V channels have a range of values from 0 to 255, while the H channel extends to 350. This method of representing colors is closer to how the human eyes see and perceive the colors.

3.2 Recognition module

Once the representation of the object is computed, the histograms are fed into the CNN. In this study, the performance of two states of the art architectures, MobileNet-v2 [22] and VGG16 [23], is evaluated. The configurations are kept as suggested in the original research, and both architectures are pre-trained on the ImageNet dataset [15]. Besides being two excellent vision model architectures, they also showed remarkable results in the research by H. Kasaei [21]. The pre-trained weights and the pre-trained feature layers are used in OrtographicNet to perform instance-based learning. Due to this transfer of weights and feature layers to perform a new type of task, we can consider this as an instance of transfer learning.

Following the OrtographicNet structure (Figure 3.2), the three feature vectors corresponding to the three views are combined using a pooling function. As a result of the pooling function, the feature vector is both scale and rotation invariant. Henceforth, we have chosen two widely used pooling functions, *Max Pooling* and *Average Pooling*. One value in the final feature vector is calculated following the formula 3.1 for *Max pooling*, and formula 3.2 for *Average pooling* respectively, where the i represents the position in the vector. In both equations, the notation stands for:

- f_i: corresponding feature vector for the *front-view* on one
- t_i: corresponding feature vector for the *top-view*
- s_i: corresponding feature vector for the *side-view*

$$F_i = \max(f_i, t_i, s_i) \tag{3.1}$$

$$F_i = avg(f_i, t_i, s_i) \tag{3.2}$$

where F_i is the final feature vector after applying the pooling function.

3.3 Object classification

In this module, the target object is finally recognized by comparing its feature representation against all known objects' descriptions in the perceptual memory. The output of the object recognition module influences the learning as well as recognition directly. For the first part of the study, we look at the influence of the similarity measures on accuracy performance. Since the given object representation represents a feature vector, the dissimilarity between the resulted feature vector and the learned instances can be calculated using different distance functions. In this case, following the approach used by Kasaei et al. [12], and the insight survey provided by S. Cha [25], we looked at the

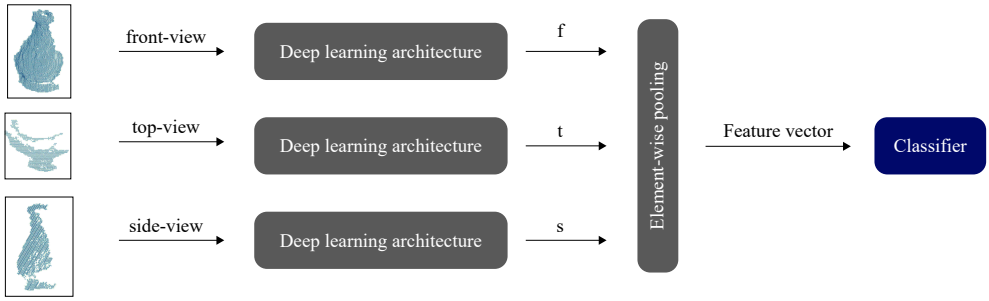

Figure 3.2: OrtographicNet pipeline where three orthographic representations are individually fed into the deep learning architecture. The final global representation of a given object is calculated by performing element-wise pooling on the three feature vectors representative for each view. The final representation is used in the classification module, the final stage in the object categorization task.

following distance functions: *KLDivergance, SymmetricKL, Motyka, Divergence, Euclidean, Manhattan, Intersection, Cosine, Dice, Bhattacharyya, Sorensen, Canberra, Pearson, Neyman, Gower*, and *ChiSquared*. For the mathematical equations, we refer the reader to S. Cha [25].

The final step in the classification process is to determine the category of the object. We can look at the shortest distance and conclude the category or use an additional classification algorithm to possibly reduce noise sensitivity. The proposed algorithm is the k-nearest neighbor (KNN). It assumes that similar objects exist in close proximity to each other. The distances from the given object to all the other instances are already calculated using the distance functions, so the remaining part is to judge the k closest categories. This means that for a k-value of 3, we look at the first three closest instances. By majority voting, the best category is decided.

3.4 Experiments

Two types of experiments are performed to evaluate the proposed methods: *offline evaluation*, and *online evaluation*.

3.4.1 Offline evaluation

Offline evaluation consists of two stages. In both stages, the evaluation is carried using the Restaurant RGB-D Object Dataset, representing household objects, Figure 3.3. It consists of 305 instances distributed over ten classes. Even though it is a small dataset, it provides a significant intra-class variation suitable for performing exhaustive sets of experiments.

The evaluation is conducted on a 10-fold cross-validation approach because it offers less biased performance. It reduces the variance by averaging over ten different partitions. This advantage is desired in small datasets. The cross-validation algorithm implies that the dataset is divided into ten equal subsets. In one iteration, one subset (fold) is kept for the test phase, and all the remaining data is used for training. We perform ten iterations, one for each fold. The value 10 for the cross-validation is considered a standard throughout the literature.

In the first stage, for each deep learning architecture, MobileNet-v2 and VGG16, we determined the top 5 best configurations of *orthographic image resolution*, *pooling function*, and *distance function*. Hence, we perform 288 experiments for each CNN, where we took into consideration 9 orthographic image resolutions increasing by 25 from 100×100 to 300×300, two pooling function, *max pooling* and *average pooling* (see Section 3.2) , and 16 distance functions discussed in Section 3.3. The best config-

Figure 3.3: The 10 classes of Restaurant RGB-D Object Dataset developed by Kasaei et al. [26]

urations provide a good balance among recognition performance, and computation time.

Taking the five best configurations from the previous stage, we continue testing on the KNN algorithm in the second stage. We take into account five k values: *1, 3, 5, 7, 9*. We only look at odd values smaller than the number of categories in order to avoid the chance of a draw. To investigate the effect of color spaces, we perform 7 experiments for each configuration: *color RGB + shape*, *color HSV + shape*, *color YUV + shape*, *color RGB*, *color HSV*, *color YUV*, and *shape*. *Color RGB* means that we only look at the orthographic image representations with the highest entropy that integrates the color information. The same applies to other color spaces. In addition, the combination *color + shape*, means that the evaluation is conducted on 4 representations. The three orthographic image representations for shape and the new representation for color.

On important observation is that we take the view with the highest entropy for color-only experiments, so only one representation is fed into the deep learning architecture. Hence, the pooling function is not necessary, and for those experiments, the pooling function is omitted.

The evaluation metric for these stages is a standardized object recognition accuracy calculated as follows:

$$\textbf{accuracy} = \frac{\#\textit{correctly predicted categories}}{\textit{total target categories}} \quad (3.3)$$

Due to system constraints, it was beneficial for this study to first asses the effect of distance functions and then the effect of KNN algorithm together with color spaces. Otherwise, it would have implied to perform 10,080 experiments for each CNN, in order to exhaust all possible combinations of *orthographic image resolution* × *pooling function* × *distance function* × *k-value for KNN* × *color space configurations*.

3.4.2 Online evaluation

The dataset used for this evaluation is the Washington RGB-D dataset. It consists of 250,000 views from 300 household objects, organized into 51 categories. (Figure 3.4)

Online learning is not a task easy to achieve using an offline evaluation that follows the classical Train-Test methodology. Nevertheless, online learning can be quickly evaluated using an open-ended protocol proposed by Kasaei et al. [26]. The teaching protocol simulates the interaction of the robot with a real environment. It integrates both the learning and recognition phases. It also has a Train and Test methodology, but it is a Train-then-Test scheme compared with the offline evaluation. Here, the protocol determines which instances are used for the Train phase and which are used for the Test phase by evaluating the system's performance. This scheme can be followed by a human user or a simulated user. In this study, we adapt the simulated user strategy. The simulated user interacts with the system using one of the three actions [28]:

Figure 3.4: Example objects from the Washington RGB-D Object Dataset developed by Lai et al. [27]

- **Teach**: used for introducing a new object category
- **Ask**: used for asking the system what the category of a given object view is
- **Correct**: used for providing the corrective feedback in case of misclassification

The training process starts with the simulated user presenting three randomly selected object views of a category to the system. The system creates a model of that class using the three examples. Afterward, the user presents a never-seen-before object view to the system and checks whether it had successfully learned the category. When it makes a misclassification, the simulated user provides feedback with the correct category. Hence, the system adjusts its category model using the mistaken instance. When the recognition performance is higher than a specific threshold, τ, the simulated user introduces a new category. The threshold in this

study was set to 67%, following the setup form the literature [21].

This process goes until the system manages to learn all categories, moment when the training process stops due to lack of data. This means that the system can learn more categories, but it is no longer possible to continue the protocol. The process can also halt when the system, cannot meet the protocol threshold τ. The latter is evaluated after a fixed number of iterations. Following the literature [21], we choose the maximum number of iterations to be 100.

Since the order of categories presented is random, we conducted ten experiments for each configuration and analyzed the averaged result. In this experiments, we investigate the influence of color spaces and test whether the offline results perform similarly in an open-ended learning domain.

In evaluating the open-ended domain's performance does not suffice to only look at the standardized accuracy. In this case, M. Oliveira et al. [29] introduces three new metrics:

- **NLC**: The number of learned categories (indicator for how much the system is capable of learning)
- **AIC**: Average number of stored instances/category (indicator for the necessary resources for learning)
- **QCI**: The number of question/correction exercises (indicator of how long it took to learn)
- **Accuracy**: Average accuracy (an indication of how well the system learns)

4 Results and discussion

This section presents the results obtained for the *Offline* and *Online* evaluation and also discusses what they mean in the context of this project.

4.1 Offline evaluation

The results corresponding to the first stage of the offline evaluation are presented in Table 4.1, and Table 4.2. We can observe that the best results are obtained by MobileNet-v2 with 0.9455 accuracies. In these cases, we cannot observe a conclusive improvement over the number of bins since the highest accuracy was obtained with various configurations. On the other hand, VGG16 shows better results with a lower orthographic image resolution. With a higher resolution of the orthographic images, the representation provides more details about the object, but it also increases computation time and noise sensitivity. We can observe that the max pooling function showed good performances in more configurations than the average pooling function. However, we can not formulate a concrete conclusion about the effect of the pooling functions, and weather max pooling is outperforming average pooling.

Looking at the distance functions, we can observe that distance functions with the same number of bins (i.e., 125x125 - Sorensen and 125x125 - Motyka) have the same average class accuracy. Taking each CNN aside, and having another look at all the results, it was visible that some distance functions, when tested on the same number of bins and the same pooling function yield the same accuracy every time. For MobileNet-v2, this insight is supported by Sorensen and Motyka, while for VGG16, we have the effect followed by Cosine, Gower, Euclidean, and Manhattan. For simplicity, we decid to only look at the configuration with the best computational time in such cases. Between Sorensen and Motyka for MobileNet-v2, Motyka took 332 seconds to complete a full experimnet, while Sorensen took 333 seconds. Regarding VGG16, the time difference

Table 4.1: Top 5 best accuracy results against four system parameters for the MobileNet-v2 architecture. The results chosen or the next stage of the offline evaluation are highhanded by blue

MobileNet-v2			
# of bins	**Pooling function**	**Distance function**	**Accuracy**
250	Avg	Canberra	0.9544
100	Max	SymmetricKL	0.9544
125	Max	Sorensen	0.9544
125	Max	Motyka	0.9544
200	Max	Sorensen	0.9544

Table 4.2: Top 5 best accuracy results against four system parameters for the VGG16 architecture. The results chosen or the next stage of the offline evaluation are highhanded by blue

VGG16			
# of bins	**Pooling function**	**Distance function**	**Accuracy**
125	Max	Cosien	0.9349
125	Max	Gower	0.9349
125	Max	Euclidean	0.9349
125	Max	Manhattan	0.9349
300	Max	Canberra	0.916

between the four configurations is little, but following the same reasoning, the best time was Manhattan with 232 seconds. One important observation is that the two CNNs cannot be compared in terms of computation time due to system constraints. The results used for the second stage in offline evaluation are highlighted with blue in Table 4.1, and Table 4.2.

Proceeding to the second stage of the offline experiments, we look at the influence of the k-nearest neighbor algorithm on seven types of experiments, which investigates the influence of color spaces on the object recognition task.

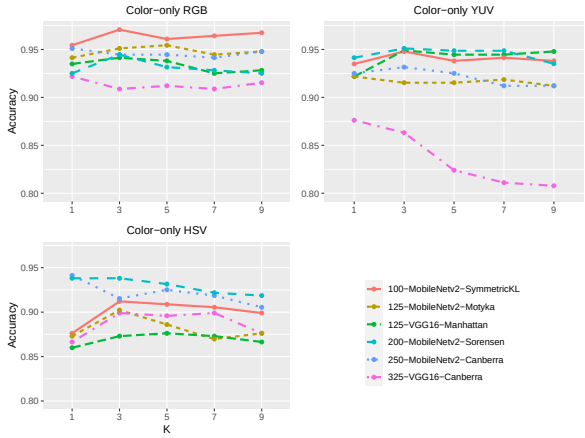

Figure 4.1: Summary of offline evaluation: the graphs show the accuracy versus k-values of KNN for color-only experiments in all color spaces. Each plot presents the performance of the six best system's configurations

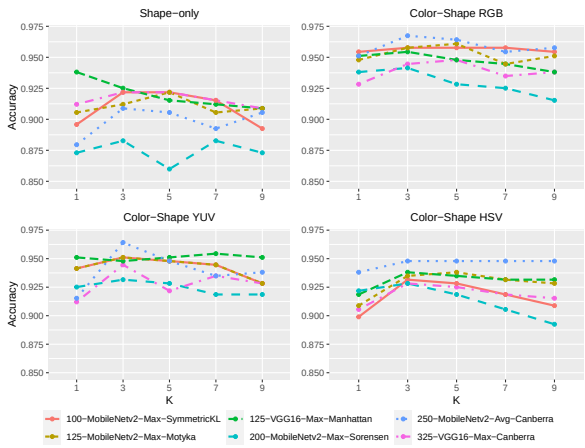

Figure 4.2: Summary of offline evaluation: the graphs show the accuracy versus k-values of KNN for all shape experiments. Each plot presents the performance of the six best system's configurations

Looking at the k-nearest neighbor algorithm's influence, we cannot conclude a definite improvement in the correctly classified classes over the five values of k. We can observe that as the k-value increases, the accuracy decreases, Figure 4.1 and Figure 4.2. In most of the combinations, at a k-value of 5, the trend is downwards, and at a k-value of 9, the lowest accuracy is observed (e.g., color-shape experiments). The high values of k indicate an unstable performance, as many instances influence the final decision in the classification module. Similar objects, e.g., red apple with red tomato or spoon with a fork, are misclassified easily. While in datasets with many classes, it could show better performances, in smaller datasets, it is not favorable to work with high values of k. What is left is to investigate weather from k = 1 to k = 3 is a significant improvement. As we can see, in some combinations, it is a slight improvement (e.g. color-shape HSV in Figure 4.2. At first glance, the boxplots do not show a significant difference between the accuracies of k = 1 and k = 3, Figure 4.3.

To answer the question, *Is there a significant difference in different k-values on accuracy?*, we perform a statistical test. Since the normality assumption for the data is not meet, we perform a nonparametric statistical test, *Kruskal-Wallis test*. The results of the test show a p-value of $0.107 > 0.05$. This means that there cannot be detected any significant differences in the accuracy between the k-values. Taking this result into consideration, for the online experiments, we do not be using the k-

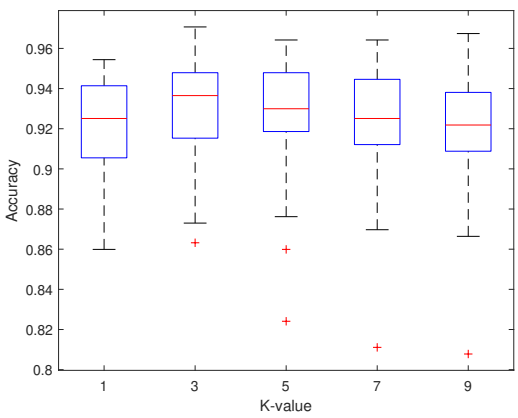

Figure 4.3: The boxplots represent the distribution of obtained accuracies for each value of k

nearest neighbor algorithm and only assess the closest class using only the distance functions.

Looking at the performance of color spaces on system performance, we can observe that all six configurations of the *number of bins* × *deep learning architecture* × *pooling function* × *distance function* performed better in the RGB color space, Figure 4.1 top-left, and Figure 4.2 top-right. 100-MobileNetv2-SymmetricKL obtains the best result with 0.9707 accuracy. For the YUV color space, the best result is obtained by 200-MobileNetv2-Sorensen with 0.9511 accuracy. For the HSV color space, the best result is obtained by 250-MobileNetv2-Canberra with 0.9414 accuracy. For the color-shape experiments, in all three color spaces, the combination 250-MobileNetv2-Avg-Canberra obtains the best results, Figure 4.2. The accuracy in the RGB color space is 0.9674, in YUV color space is 0.9642, and for the HSV color space is 0.9479.

The combination of color and shape yields better results than shape and color alone. While for color-only experiments, we can observe that a few spikes are exceeding 0.95 (Figure 4.1), we also have many results bellow 0.9 (color-only YUV and color-only HSV). In contrast, the color-shape experiments have only one value lower than 0.9, in the HSV color space for a k-value of 9 and combination 200-MobileNetv2-Sorensen.

Out of all three color spaces, the RGB color space results in a better performance for both color-only experiments and color-shape experiments. The shape-only experiments show the worst results. However, the lowest result is reported with the combination 325-VGG16-Max-Canberra, in the YUV color space, color-only experiment, where all five results are bellow 0.9 (Figure 4.1 top-left). This can be influenced by the noise sensitivity of a sizeable orthographic image resolution. The combination 125-VGG16-Max-Manhattan determines the best result for the shape-only experiments with an accuracy of 0.9381.

Table 4.3: Summary of final results of offline evaluation. Each of the seven configurations obtained the highest score in the corresponding experiment

Experiment	Configuration			
	# bins	Architecture	Pooling function	Distance function
Color-only RGB	100	MobileNet-v2	-	SymmetricKL
Color-only YUV	200	MobileNet-v2	-	Sorensen
Color-only HSV	250	MobileNet-v2	-	Canberra
Color-Shape RGB	250	MobileNet-v2	Avearge	Canberra
Color-Shape YUV	250	MobileNet-v2	Avearge	Canberra
Color-Shape HSV	250	MobileNet-v2	Avearge	Canberra
Shape-only	125	VGG16	Max	Manhattan

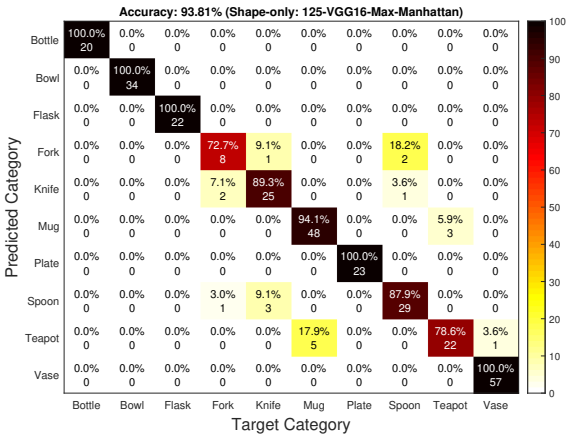

Figure 4.4: Confusion matrix of the best configuration in the shape-only experiments

Taking the best results for each type of experiment presented in Table 4.3, we look at their confusion matrix to evaluate the misclassified classes and discuss the possible reasons for that (all confusion matrices are presented in Appendix A). Figure 4.4 represents the confusion matrix for the best result of shape-only experiments. Since the results for this category were the lowest, with this confusion matrix, we can discuss more conflicts among the categories. We can observe that the misclassifications mainly occur among classes that look alike. In Figure 4.5, we can observe the similarities between the spoon, fork, and knife classes. It is easy to observe that in all combinations, the classes previously mentioned are continuously mixed up. The system is even more sensitive in the color-shape combinations.

Even though the shape information should distinguish the object at a detailed level, since the color is very similar, the results show the opposite effect. In the shape-only experiments, we can also observe that the mug is misclassified as a teapot

Figure 4.5: One sample from class *spoon*(left), *fork*(center), *knife*(right) [26]. The image aims to underline the similarities among the three categories

and vice versa (Figure 4.4). This underlines the shape sensitivity for objects that have a similar silhouette.

4.2 Online evaluation

All results represent the average of 10 experiments (together with standard deviation) for each of the color-only, color-shape, and shape-only configurations. The overview of the performances is presented in Table 4.4. At first glance, we can observe that the RGB color space performed the best in both color-only and color-shape experiments, achieving an accuracy of 0.84 in color-only and 0.78 in color-shape (Table 4.4, blue rows). Looking at the table, it can also be observed that all color spaces show improvement to the shape information. Shape-only experiments perform the worst in all four metrics. While all six color configurations manage to learn all 51 categories and stop prematurely due to lack of data, the shape-only configuration succeed in learning all 51 categories in only 2 out of 10 experiments (Table 4.4, second column). Regarding the YUV and HSV color spaces, the latter perform better in color-shape experiments, while the former perform very similarly in both types of experiments. The similarities can be observed in Figure 4.6, where the averaged accuracy results are represented in boxplots. This type of plot is representative of analyzing the significant differences among the results by showing the minimum, first quartile, median, third quartile, and maximum performances. While color-only RGB and shape-only have very distinctive results, the other combinations have the distribution around the same results.

Overall, we can observe that the hierarchy from the offline results is followed in the online evaluation. It is worth noticing that the system performance dropped in the online experiments. While in offline conditions, all configurations successfully score over 90% accuracy, none of the conditions manage to reach such high performance in the current experiments. This observation shows a lack of robustness on the side of offline implementations and standard Train-Test methodology for open-ended domains.

Looking further through results, the number of question/correction exercises is similar for all configurations expect color-only HSV, which scores the highest QCI (Table 4.4 fourth column, third row).

Looking at the number of stored instances/category, it is noticeable that the smaller

Table 4.4: Summary of online evaluation of all approaches. The results represent the average of 10 tests. The best result for each type of experiment color-only, color-shape, and shape-only is highlighted with blue

Experiment	NLC	AIC	QCI	Accuracy
Color-only RGB	51	7.11±0.22	1329.2±3.42	0.84±0.008
Color-only YUV	51	9.09±0.42	1374.4±39.44	0.77±0.01
Color-only HSV	51	10.77±0.84	1516.4±77.21	0.74±0.01
Color-Shape RGB	51	8.7±0.6	1357.2±36.45	0.78±0.01
Color-Shape YUV	51	9.1±0.62	1367.4±25.83	0.77±0.01
Color-Shape HSV	51	9.24±0.46	1389.4±30.44	0.76±0.01
Shape-only	37.2±11.4	14.194±42.02	1345.2±635.87	0.67±0.01

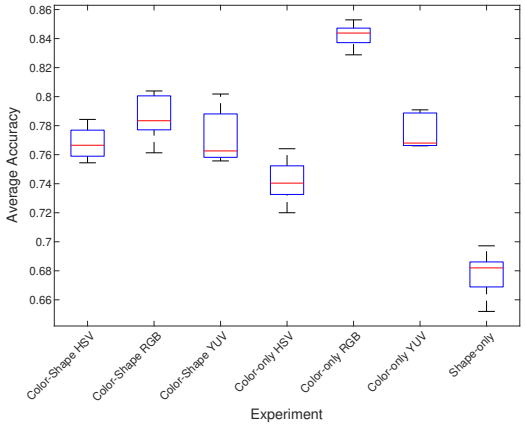

Figure 4.6: Average accuracy plotted as boxplot distribution, for each of the seven experiments. The accuracy is a measurement of how well the system learns

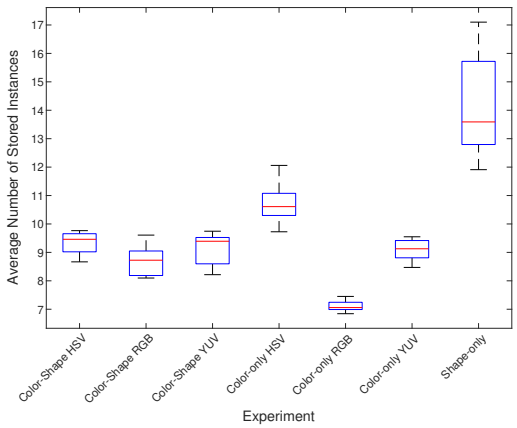

Figure 4.7: Average number of instances necessary per category(AIC) plotted as boxplot distribution, for each of the seven experiments. The number of instances is a measurement of the amount of necessary resources for learning

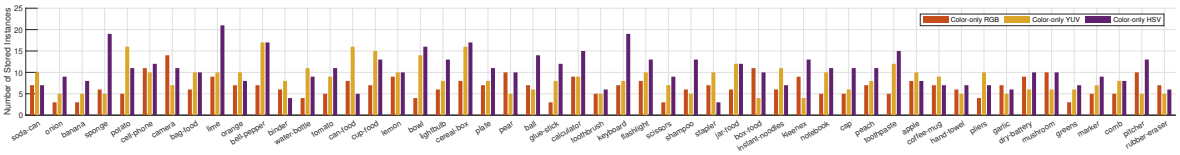

Figure 4.8: The graph shows the number of instances stored in the models of all the categories in three system configurations: Color-only RGB, Color-only YUV, and Color-only HSV. The categories in the end seem to be presented fewer times, which means that they have also been testes less. In all three configurations, the system managed to successfully learn all 51 categories

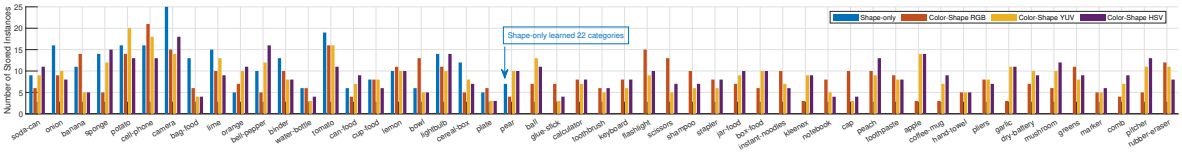

Figure 4.9: The graph shows the number of instances stored in the models of all the categories in four system configurations: Shape-only, Color-Shape RGB, Color-Shape YUV, and Color-Shape HSV. The categories in the end seem to be presented fewer times, which means that they have also been testes less. Out of all experiments, the shape-only model succeeded in learning only 22 categories. All the other three models successfully learned all 51 categories

the AIC, the higher the accuracy. In Figure 4.7, we look at the averaged number of instances needed for learning in all six configurations. Color-only RGB needs an average of 7.11 instances/category while the shape-only uses twice as much (Table 4.4, the third column). The remaining five configurations need, on average, 9 instances/categories for learning.

Figure 4.8 and Figure 4.9 represent the average number of needed instances on category. Since the order of the categories is randomly chosen, the classifier's performance is negatively influenced by similar objects presented one after another. For example, a red apple presented after a red ball can mislead the system in deciding that both are from the same category. In contrast, a red ball showed after a yellow banana is unlikely to be misclassified as part of the same class. In Figure 4.9, we cannot observe a definite pattern that reflects the sensitivity to one shape. One notable observation is that only 22 categories were commonly presented among the 10 experiments in the shape-only configuration. Hence, even though the average number of learned categories is 37, as reported in Table 4.4, only 22 categories were able to be presented here.

Since the configuration is not the same for all seven experiments, we cannot conclude an evident influence of color-spaces, and only the influence of color-spaces at an discrete level. The comparison is conducted between the offline evaluation and the online evaluation but not among themselves. However, it can be noticed that the configuration 250-MobileNetv2-Avg-Canberra showed good results in four out of seven online configurations. A definite improvement can be observed for the HSV color spaces, and that the shape information is an improvement to the recognition module. It is interesting to observe that the performance of color space alone does not match the configuration with the shape information. For example, color-only HSV needed 21 instances for the "lime" category, while color-shape HSV needed only 9, Figure 4.9.

5 Conclusion

5.1 Research question

This study aimed to answer the following research question: *Which combination of similarity measures, color spaces, shape information, and deep learning architectures performs the best in an open-ended 3D object recognition task?*

For deep learning architectures, we analyzed the performance of two state-of-the-art CNNs, MobileNet-v2, and VGG16. Overall, MobileNet-v2 showed better results in both offline and online evaluations, confirming the results of H. Kasaei [21]. Following the result from the offline exper-

iments, the average pooling function showed the highest accuracies for color-shape experiments.

Concerning the similarity measure, we looked at the impact of two approaches, distance functions, and k-nearest neighbor algorithm. Conducting the second stage of offline experiments, the k-nearest neighbor algorithm did not significantly improve the system's performance over the distance functions alone. However, it is worth investigating the effect of this classification algorithm on more extensive datasets with more than 10 classes. From the 16 distance functions evaluated in the offline scenario, the final results showed that SymmetricKL, Canberra, and Sorensen showed the best performance for the MobileNet-v2. At the same time, Manhattan was suited better for VGG16 architecture.

The three color spaces that we evaluated in this study are RGB, YUV, HSV. RGB color space outperformed the other two in both offline and online evaluations. The YUV color space followed in second place, and in the last place was HSV color space. YUV and HSV color spaces, in combination with the shape information, showed better results than color or shape information alone.

However, the RGB color space obtained a higher accuracy in the color-only configuration. The training data could influence the bias towards the RGB color space. MobileNet-v2 was pre-trained on the ImageNet dataset, which could have influenced the recognition module to have an easier time classifying RGB images. Conclusively, the experiments using only the global shape information (shape-only) revealed that only the object's figure is not enough to master an object classification task, especially in the open-ended domain.

5.2 Further work

When deciding upon the object representation, the chosen method can change the system's performance decidedly. Naturally, to mimic the human vision system, we tend to aim to integrate or extract as many objects features as possible. (e.g., shape, color, texture, etc.). As we discussed in section 2, global shape information is considered to embody enough characteristics of the target object. Hence, other features are ignored as a compromise for a faster approach [30]. However, we showed that the shape information alone performed the worst among the seven configurations of color-only, color-shape, and shape-only. This study showed that color spaces are already an improvement to methodological shape-only experiments. Further research should investigate the importance of texture information.

One approach is to apply a texture filter over the original image, such as Gabor Filters, that are widely used for texture analysis, and extract the texture information that follows to be bind or integrated into the object representation before being sent to the recognition module. When using deep learning architectures, they are trained to extract such features automatically. If not trained on texture-rich datasets, the CNNs does not look for texture specifically. Hence, for investigating the texture influence, a richer dataset is needed.

In Section 4.1, we mentioned that some distance functions when tested on the same number of bins, and the same pooling function yielded the same accuracy every time. Since the distance function's formula is different, it is worth discovering what parameter of a combination of parameters concluded this effect. It is also interesting to investigate whether such parameters cancel the effect and minimize distance function performance in precise evaluations.

Regarding the RGB color space, one view was enough, while for the other color spaces, color-shape showed better results. Since both deep learning architectures were pre-trained on the ImageNet dataset, it can underline a slight bias towards RGB color space. Hence, it is worth investigating if other CNNs pre-trained on ImageNet dataset show the same results or if CNNs pre-trained on other color spaces show different performances. In instance based learning approaches, the HSV outperformed the RGB color space. Therefore, when correctly used HSV color space from the training stage, the system's performance increases.

The final assessment of our results should be determined in real-time system demonstrations. In such experiments, we can immediately witness the system's actual flaws and understand what can be improved. Real-time system evaluations are vital in implementing the proposed system in service robots for a 3D object classification task. Consequently, these experiments should be performed as further research.

References

[1] H.-J. Kwak and G.-T. Park, "Study on the mobility of service robots," *International Journal of Engineering and Technology Innovation*, vol. 2, pp. 97–112, 2012.

[2] J. Miura, Y. Shirai, N. Shimada, M. T. Y. Makihara, and Y. Yano, "Development of a personal service robot with user-friendly interfaces," *Field and Service Robotics*, vol. 2, pp. 427–436, 2006.

[3] A. L. Yuille and C. Liu, "Deep nets: What have they ever done for vision?," *ArXiv*, vol. abs/1805.04025, 2018.

[4] T. Lesort, V. Lomonaco, D. M. Andrei Stoian, D. Filliat, and N. Díaz-Rodríguez, "Continual learning for robotics: Definition, framework, learning strategies, opportunities and challenges," *Information Fusion*, vol. 58, pp. 52–68, 2020.

[5] T. F. Brady, T. Konkle, G. A. Alvarez, and A. Oliva, "Visual long-term memory has a massive storage capacity for object details," *Proceedings of the National Academy of Sciences*, vol. 105, pp. 14325 – 14329, 2008.

[6] S. H. Kasaei, A. M. Tomé, L. S. Lopes, and M. Oliveira, "GOOD: A global orthographic object descriptor for 3D object recognition and manipulation," *Pattern Recognit. Lett.*, vol. 83, pp. 312–320, 2016.

[7] X.-F. Han, J. S. Jin, J. Xie, M.-J. Wang, and W. Jiang, "A comprehensive review of 3D point cloud descriptors," *ArXiv*, vol. abs/1802.02297, 2018.

[8] W. Wohlkinger and M. Vincze, "Ensemble of shape functions for 3D object classification," *2011 IEEE International Conference on Robotics and Biomimetics*, pp. 2987–2992, 2011.

[9] R. B. Rusu, G. Bradski, R. Thibaux, and J. Hsu, "Fast 3D recognition and pose using the viewpoint feature histogram," *International Conference on Intelligent Robots and Systems*, p. 2155–2162, 2014.

[10] F. Tombari, S. Salti, and L. D. Stefano, "A combined texture-shape descriptor for enhanced 3D feature matching," *IEEE International Conference on Image Processing*, pp. 809–812, 2011.

[11] S. H. Kasaei, L. S. Lopes, A. M. Tomé, and M. Oliveira, "An orthographic descriptor for 3D object learning and recognition," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4158–4163, IEEE, 2016.

[12] S. H. Kasaei, M. Ghorbani, J. Schilperoort, and W. van der Rest, "Investigating the importance of shape features, color constancy, color spaces and similarity measures in open-ended 3D object recognition," *ArXiv*, vol. abs/2002.03779, 2020.

[13] S. N. Gowda and C. Yuan, "ColorNet: Investigating the importance of color spaces for image classification," *ArXiv*, vol. abs/1902.00267, 2018.

[14] R. Geirhos, P. Rubisch, C. Michaelis, M. Bethge, F. A. Wichmann, and W. Brendel, "ImageNet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness," *ArXiv*, vol. abs/1811.12231, 2019.

[15] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255, 2009.

[16] X. Huang and S. J. Belongie, "Arbitrary style transfer in real-time with adaptive instance normalization," *ICCV*, pp. 1510–1519, 2017.

[17] B. Landau, L. B. Smith, , and S. S. Jones, "The importance of shape in early lexical learning," *Cognitive Development*, vol. 3, pp. 299–321, 1988.

[18] N. Baker, H. Lu, G. Erlikhman, and P. Kellman, "Local features and global shape information in object classification by deep convolutional neural networks," *Vision Research*, vol. 172, pp. 46–61, 2020.

[19] G. Heitz, G. Elidan, B. Packer, and D. Koller, "Shape-based object localization for descriptive classification," *International Journal of Computer Vision*, vol. 172, pp. 40–62, 2009.

[20] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?," in *NIPS*, 2014.

[21] S. H. Kasaei, "OrthographicNet: A deep learning approach for 3D object recognition in open-ended domains," *ArXiv*, vol. abs/1902.03057, 2019.

[22] M. Sandler, A. G. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4510–4520, 2018.

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

[24] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016.

[25] S. Cha, "Comprehensive survey on distance/similarity measures between probability density functions," *International Journal of Mathematical Models in Applied Sciences*, vol. 1, pp. 300–307, 2007.

[26] S. H. Kasaei, M. Oliveira, G. H. Lim, L. S. Lopes, and A. M. Tomé, "Interactive open-ended learning for 3D object recognition: An approach and experiments," *Journal of Intelligent & Robotic Systems*, vol. 80, no. 3-4, pp. 537–553, 2015.

[27] K. Lai, L. Bo, X. Ren, and D. Fox, "A large-scale hierarchical multi-view RGB-D object dataset," *2011 IEEE International Conference on Robotics and Automation*, pp. 1817–1824, 2011.

[28] S. H. Kasaei, L. S. Lopes, and A. M. Tomé, "Coping with context change in open-ended object recognition without explicit context information," in *International Conference on Intelligent Robots and Systems (IROS)*, pp. 1–7, 2018.

[29] M. Oliveira, L. S. Lopes, G. H. Lim, S. H. Kasaei, A. D. Sappa, and A. M. Tome, "Concurrent learning of visual codebooks and object categories in open-ended domains," *Intelligent Robots and Systems*, pp. 2488–2495, 2015.

[30] S. H. Kasaei, J. Sock, L. S. Lopes, A. M. Tomé, and T.-K. Kim, "Perceiving, learning, and recognizing 3D objects: An approach to cognitive service robots," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

A Appendix

These are the confusion matrices for each system configuration described in Table 4.3. The first column represents the color-only experiments while the second column represents the color-shape experiments. On the fourth row there is the shape-only experiment.