ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΜΗΧΑΝΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΔΠΜΣ ΣΥΣΤΗΜΑΤΑ ΑΥΤΟΜΑΤΙΣΜΟΥ

Διπλωματική εργασία

Semantic Segmentation for ADL in Assistive Robotics and Comparison of RGB/RGB-D Input and Single/Multiple Viewpoints

Οικονόμου Αικατερίνη-Μαρία

Επιβλέποντες καθηγητές : Κωνσταντίνος Ι. Κυριακόπουλος (NTUA)
Axel Gerd Peter Graeser (University of Bremen)

ΑΘΗΝΑ 2021

Περίληψη

Η υπολογιστική όραση είναι ένας ευρύτατα μελετημένος κλάδος της τεχνητής νοημοσύνης ο οποίος με τη αξιοποίηση αλγορίθμων στοχεύει στην ερμηνεία και την κατανόηση του οπτικού κόσμου. Συνδυάζοντας την ψηφιακή επεξεργασία εικόνας, την αναγνώριση μοτίβου (pattern) και την μηχανική μάθηση επιδιώκει μέσα από ακατέργαστα δεδομένα να εξάγει χρήσιμες πληροφορίες. Η μηχανική όραση στοχεύει στην ανάπτυξη συστήματος ώστε τα ρομπότ να κατανοήσουν τις εικόνες με τρόπο παρόμοιο με αυτόν των ανθρωπων.

Σε αυτή τη διπλωματική θα επικεντρωθούμε στην σημασιολογική τμηματοποίηση, τη διαδικασία διαχωρισμού (labelling) κάθε περιοχής –ή pixel– μιας ψηφιακής εικόνας σε συγκεκριμένες περιοχές ή αντικείμενα. Κάνουμε χρήση βαθιών νευρωνικών δικτύων (Deep Neural Networks), τα οποία τα τελευταία χρόνια έχουν αποδειχθεί ότι υπερτερούν των προηγούμενων τεχνικών μηχανικής μάθησης σε διάφορους τομείς, με την υπολογιστική όραση να είναι μια απο τις πιο εμφανής περιπτώσεις. Τα βαθιά νευρωνικά δίκτυα μπορούν να εκπαιδευτούν να αναγνωρίζουν και να ομαδοποιούν αντικείμενα σε μια εικόνα και πιο συγκεκριμένα τα συνελικτικά νευρωνικά δίκτυα (Convolutional Neural Networks (CNN)), μια κατηγορία των νευρωνικών δικτύων που χρησιμοποιείται συνήθως για την εξαγωγή χαρακτηριστικών στην περίπτωση των εικόνων, έχει αποδειχθεί ότι μπορούν να εξάγουν σημαντικά οπτικά χαρακτηριστικά που μοιάζουν με αυτά που δημιουργούνται στον οπτικό φλοιό του ανθρώπου.

Συγκεκριμένα αναπτύσσουμε μια πρωτότυπη συνελικτική αρχιτεκτονική νευρωνικών δικτύων για σημασιολογική τμηματοποίηση στα πλαίσια υποβοήθησης της ανθρώπινης δραστηριότητας στην καθημερινή ζωή, με έμφαση στην σημασιολογική τμηματοποίηση εικόνων σε ένα σενάριο πρωινού. Χρησιμοποιούμε εικόνες RGB και RGB-D (εικόνες με 4 κανάλια: RGB και βάθος) ως σήματα εισόδου και εξετάζουμε τα οφέλη της προσθήκη της πληροφορίας του βάθος κατά την εκπαίδευση του δικτύου. Σε επόμενο βήμα, εξετάζουμε την επίδοση του νευρωνικού δικτύου μας οταν εκπαιδευτεί σε δεδομένα από διαφορετικές οπτικές γωνίες. Χρησιμοποιούμε δύο κάμερες που στοχεύουν το τραπέζι από διαφορετικές οπτικές γωνίες. Η πρώτη κάμερα τοποθετείται σε headset (οπτική του χρήστη) και η δεύτερη στο τραπέζι.

Επιπλέον για την παρούσα έρευνα δημιουργήσαμε ένα νέο σύνολο δεδομένων χρησιμοποιώντας δύο Intel RealSense κάμερες στην δομή που ήδη αναφέρθηκε. Τα πρώτα δύο συνελικτικά νευρωνικά δίκτυα (CNN) εκπαιδεύτηκαν με το σύνολο δεδομένων που συλλέξαμε από το headset (ένα με RGB και το επόμενο με RGB-D). Το τρίτο νευρωνικό δίκτυο εκπαιδεύτηκε με το σύνολο δεδομένων που συλλέχθηκε από τη δεύτερη κάμερα και το τελευταίο με το σύνολο δεδομένων και από τις δύο οπτικές.

Ο προσδιορισμός της σημασίας χρήσης εικόνων από διαφορετικές οπτικές γωνίες και της πληροφορίας του βάθους στη σημασιολογική τμηματοποίηση, θα βοηθήσει στην ανάπτυξη ισχυρών, γρήγορων και ακριβέστερων αλγορίθμων για την αναγνώριση αντικειμένων σε εφαρμογές υποβοήθησης της ανθρώπινης δραστηριότητας στην καθημερινή ζωή (ADL). Αυτό είναι ένα σημαντικό βήμα για την επίτευξη του τελικού στόχου της δημιουργίας ρομπότ (π.χ. ρομποτικούς βραχίονες, αναπηρικές καρέκλες και εξωσκελετούς) που μπορούν να βοηθήσουν τα άτομα με σοβαρές κινητικές βλάβες να απολαύσουν το πρωινό τους.

Abstract

Computer vision, the field of artificial intelligence that develops computational algorithms to interpret and understand the visual world, has been studied in many perspectives. It expands from raw data recordings into techniques and ideas combining digital image processing, pattern recognition, machine learning and computer graphics. As its extensive usage has attracted many scholars to integrate with many disciplines and fields, in assistive robotics, computer vision aims to develop systems to help robots understand images in a similar way humans do.

In this thesis, we focus on semantic segmentation, the process of labelling each area –or pixel– of a digital image according to a representation class. We make use of deep learning methods, which, over the last years, have been shown to outperform previous state-of-the-art machine learning techniques in several fields, with computer vision being one of the most prominent cases. Deep learning models can be trained to identify and classify objects in an image, and, in particular, Convolutional Neural Networks (CNN), a class of deep neural networks most commonly used in visual imagery, has been shown to be able to extract important visual features that resemble the ones generated in the visual cortex of humans.

In particular, we develop a novel convolutional neural network architecture for semantic segmentation in the framework of Activity of Daily Life (ADL), with focus on semantic segmentation of images in a breakfast scenario. We use RGB and RGB-D images as input signals (images with 4 channels: RGB colors and Depth) and quantify the benefits of the additional information of depth in object classification. As a next step, we evaluate the performance of our deep neural network when trained with data from multiple viewpoints. We make use of two cameras pointing on the table with a different field of view. The first camera is placed on a headset (user's point of view) and the second on the table.

Furthermore, for this project we create a new dataset using two Intel RealSense camera sensors, in the apparatus mentioned above. This dataset is used to train three different convolutional neural networks for semantic segmentation. The first two CNNs are trained with the dataset collected from the headset (one with RGB and one with RGB-D information). The third CNN is trained with the dataset collected from the second camera, and the last one with the dataset from both camera views.

Quantifying the importance of multiple viewpoints and depth information in semantic segmentation will help in developing robust, fast, and more precise perception algorithms for object recognition in ADL applications. This is an important step towards the final goal of creating assistive robot agents (e.g. robotic arms, wheelchairs, and exoskeletons) that can help people with severe motor impairments enjoy their first meal of the day.

Acknowledgment

I would like to express my gratitude toward both of my supervisors, Axel Gerd Peter Graeser from University of Bremen and Konstantinos J. Kyriakopoulos from National Technical University of Athens for giving me the opportunity to work on this thesis. Especially I would like to thank Felix Goldau who has provided me with much technical guidance and support during my entire time of study in Bremen. Finally I thank my family for supporting me throughout this project. Mostly I am thankful that I was able to conduct this thesis despite the difficulties caused by the COVID-19 pandemic.

Contents

1 Introduction **3**

2 Technical Background **6**

2.1 Computer Vision and Object Recognition 6

2.1.1 Image Classification and Object Detection 6

2.1.2 Semantic and Instance Segmentation 7

2.1.3 Semantic Segmentation for Assistive Robotics in Activities of Daily Living 7

2.2 Machine Learning 8

2.2.1 A Historical Retrospective of Machine Learning. 8

2.3 Deep Convolutional Neural Networks 9

2.3.1 Hyper-parameters 10

2.3.2 Training Algorithms and Evaluation 13

3 Methodology **15**

3.1 Deep Learning for Semantic Segmentation 15

3.2 Simulated Dataset 17

3.3 Real Dataset 18

4 Experimental Evaluation **20**

4.1 RGB Data 20

4.1.1 Simulated data 21

4.1.2 Real data (headset view) 24

4.1.3 Real data (table view) 26

4.2 RGB-D Data 28

4.2.1 Simulated data 29

4.2.2 Real data (headset view) 32

4.3 Multiple view images 34

5 Conclusion **37**

Chapter 1

Introduction

Image segmentation is an essential component in many visual understanding systems. It involves partitioning images (or video frames) into multiple segments or objects [1]. Segmentation plays a central role in a broad range of applications [2], including medical image analysis (e.g., tumor boundary extraction and measurement of tissue volumes), autonomous vehicles (e.g., navigable surface and pedestrian detection), video surveillance, and augmented reality to count a few. Numerous image segmentation algorithms have been developed in the literature, from the earliest methods, such as thresholding [3], histogram-based bundling, regiongrowing [4], and k-means clustering [5], to more advanced algorithms such as conditional and Markov random fields [6], and sparsity based [7, 8] methods. Over the past few years, however, deep learning (DL) models have yielded a new generation of image segmentation models with remarkable performance improvements —often achieving the highest accuracy rates on popular benchmarks— resulting in a paradigm shift in the field [9]. For example, Figure presents image segmentation outputs of a popular deep learning model, DeepLabv3 [10].

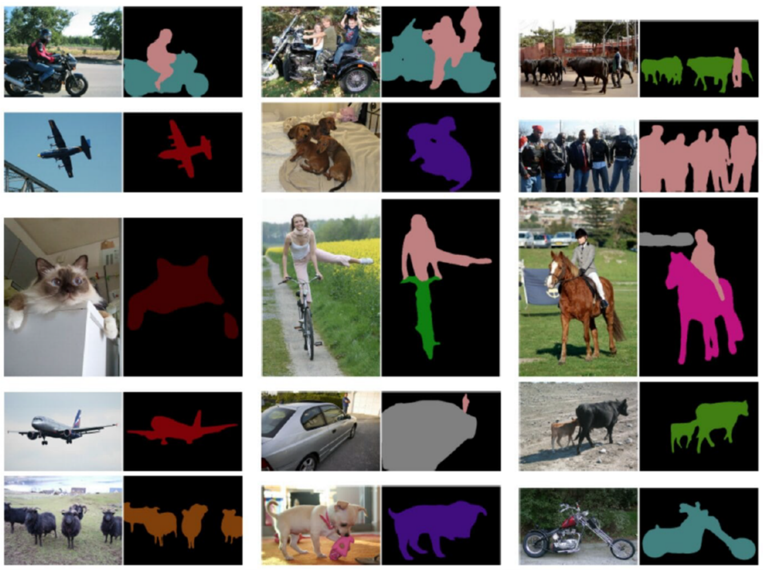


Figure 1.1: Segmentation results of DeepLabV3 [10] on sample images.

Image segmentation can be formulated as a classification problem of pixels with semantic labels (semantic segmentation) or partitioning of individual objects (instance segmentation). Semantic segmentation performs pixel-level labelling with a set of object categories (e.g., human, car, tree, sky) for all image pixels, thus it is generally a harder undertaking than image classification, which predicts a single label for the entire image. Instance segmentation extends semantic segmentation scope further by detecting and delineating each object of interest in the image (e.g., partitioning of individual persons).

In this thesis, we will focus on semantic segmentation, the process of labelling each area –or pixel– of a digital image according to a representation class. We will make use of deep learning methods, which, over the last years, have been shown to outperform previous state-of-the-art machine learning techniques in several fields, with computer vision being one of the most prominent cases. Deep learning models can be trained to identify and classify objects in an image, and, in particular, Convolutional Neural Networks (CNN), first proposed by Lecun in [11]. CNNs are a class of deep neural networks most commonly used in visual imagery, and has been shown to be able to extract important visual features that resemble the ones generated in the visual cortex of humans [12, 13].

In this regard, we will develop a novel convolutional neural network architecture, based on the MobileNetV2 framework [14], for semantic segmentation in the framework of assistive robotics for Activities of Daily Living (ADL) [15, 16]. In particular, we will focus on semantic segmentation of images in a breakfast scenario. We will use RGB-D images as input signals (images with 4 channels: RGB colors and Depth) and quantify the benefits of the additional information of depth in object classification. We will then evaluate the performance of our deep neural network when trained with data from multiple viewpoints. We will make use of two cameras pointing on the table with a different field of view. The first camera will be placed on a headset (user's point of view) and the second on the table.

Moreover, for this project we will create a new dataset for assistive robotics applications, inspired by [16, 17], using two Intel RealSense camera sensors, in the apparatus mentioned above. This dataset will be used to train four different convolutional neural networks for semantic segmentation. The first two CNNs will be trained with the dataset collected from the headset (one with RGB and one with RGB-D information). A third CNN will be trained with the dataset collected from the second camera and the last one with the dataset from both cameras.

Quantifying the importance of the information of depth in semantic segmentation in object recognition, will help in developing robust, fast, and more precise perception algorithms for object recognition in ADL applications [18]. This is an important step towards the long-term goal of creating assistive robot agents (e.g. robotic arms, wheelchairs, and exoskeletons) [16, 19, 20] that can help people with severe motor impairments enjoy their first meal of the day.

Finally, the contribution of this work can be summarised in the following:

- We introduce a deep convolutional neural network to be used for semantic segmentation in assistive robotics for activities of daily living
- We create a high-quality real-life labeled dataset of images from a breakfast table scenario multiple views are provided with potentially partial visibility
- We evaluate the performance of the proposed learning architecture for semantic segmentation in these new real-life data, and compare it to simulation results
- We quantify the effect of multiple views and depth in the performance of the proposed algorithms.

Chapter 2

Technical Background

2.1 Computer Vision and Object Recognition

Computer vision has been expanded into the vast area of field ranging from recording raw data into the extraction of image pattern and information interpretation [21]. It has a combination of concepts, techniques, and ideas from digital image processing, pattern recognition, artificial intelligence and computer graphics [22]. Most of the tasks in computer vision are related to the process of obtaining information on events or descriptions, from input scenes (digital images) and feature extraction. The methods used to solve problems in computer vision depend on the application domain and the nature of the data being analysed.

In the context of assistive robotics, the goal of computer vision algorithms is to develop systems that can help robots understand images as human vision system does. There is a variety of computer vision techniques that are widely used today and choosing the right one for each purpose may sometimes be challenging. These include:

- Image Classification
- Object Detection
- Instance Segmentation
- Semantic Segmentation

2.1.1 Image Classification and Object Detection

Image classification refers to classifying an image as a whole into different categories. While many applications are based in classification algorithms such as these [23], the context of an image cannot be directly interpreted [24].

Object detection, also known as bounding-box detection, involves outputting bounding boxes and labels for each object found inside an image. The difference between object detection and image classification method is that the first approach classifies and localise many objects instead of a single one. Object detection is used in vast domains like scene understanding, self-driving, quality control of parts in manufacturing etc [25].

2.1.2 Semantic and Instance Segmentation

Image segmentation is an essential component in many visual understanding systems. It involves partitioning images (or video frames) into multiple segments or objects [1]. Segmentation plays a central role in a broad range of applications [2], including medical image analysis (e.g., tumor boundary extraction and measurement of tissue volumes), autonomous vehicles (e.g., navigable surface and pedestrian detection), video surveillance, and augmented reality to count a few.

Image segmentation can be formulated as a classification problem of pixels with semantic labels (semantic segmentation) or partitioning of individual objects (instance segmentation). Semantic segmentation performs pixel-level labelling with a set of object categories (e.g., human, car, tree, sky) for all image pixels, thus it is generally a harder undertaking than image classification, which predicts a single label for the entire image. Instance segmentation extends semantic segmentation scope further by detecting and delineating each object of interest in the image (e.g., partitioning of individual persons).

Numerous image segmentation algorithms have been developed in the literature, from the earliest methods, such as thresholding [3], histogram-based bundling, regiongrowing [4], and k-means clustering [5], to more advanced algorithms such as conditional and Markov random fields [6], and sparsity based [7, 8] methods. Over the past few years, however, deep learning (DL) models have yielded a new generation of image segmentation models [26] with remarkable performance improvements –often achieving the highest accuracy rates on popular benchmarks– resulting in a paradigm shift in the field [9, 13, 18].

Sensor Inputs

Semantic segmentation approaches have been traditionally focused on two- dimensional images [27–31]. The sensor information typically involves three colour channels (RGB) –which constitute a complete basis in the colour space– where each channel consists of a 2D array of pixel values, usually in the range of $(0, 255)$. Additional depth information can be added in the form of an extra channel to form the so called 2.5D data (RGB-D), which, however, are still in the minority compared to the RGB datasets available today [32–36].

2.1.3 Semantic Segmentation for Assistive Robotics in Activities of Daily Living

In the recent years, the adoption of robotic devices in rehabilitation and assistive field is widely increased for a number of reasons; they can assure a high repeatability of movements and intensity of treatment, an active role of the patients and an enhancement of their level of independence and social participation. Robot motion planning plays an important role in the interaction between user and assistive technology, especially for the execution of Activities of Daily Living (ADLs) [16, 17, 37], which are fundamentals for a social and professional reintegration. Since shared robotic schemes heavily rely on image sensory for decision making, object recognition and semantic segmentation stand at the core of most assistive robotics problems.

In order to deal with semantic segmentation in assistive robotics, however, one needs to employ advanced notions and algorithms from the field of machine learning, which is the topic of the next section.

2.2 Machine Learning

Machine Learning, the study of computer algorithms that improve automatically through experience, has been established as an integral part of the research focus in a wide range of fields, including computer science, mathematics and neuroscience. As a sub-field of Artificial Intelligence it is associated with computational algorithms that try to recognise patterns, approximate functions, learn data representations and automatically make decisions. Due to the interdisciplinary nature of these problems, Machine Learning has adopted mathematical models and methods from functional analysis, optimisation and systems theory, and has taken inspirations from biology and neuroscience, the scientific study of the nervous system of humans towards the understanding of the biological basis of learning, memory, behavior, perception, and consciousness.

2.2.1 A Historical Retrospective of Machine Learning.

What came to be known as "Machine Learning" starts in the 1950's and 1960's with statistical methods for probabilistic inference. At the same time, digital signal processing methods and system identification and filtering are introduced for linear systems [38, 39], as well as a mathematical model for a biologically inspired artificial neural network with the Hebbian and perceptron learning rules [40, 41]. In the 1970's, limitations of the perceptrons and neural networks, such as the famous XOR problem, trigger a pause phase for neural networks called the "AI winter".

In the 1980's the rediscovery of the backpropagation algorithm [42] causes a resurgence in machine learning research, and the concepts of reinforcement learning and convolutional neural networks are first introduced [43, 44].

In the 1990's, the work on Machine Learning shifts from a knowledge-driven approach to a data-driven approach. Support vector machines (SVMs) [45] and recurrent neural networks (RNNs), such as Long Short-Term Memory (LSTM) networks [46], become popular, and the fields of computational complexity via neural networks and super-Turing computation start. At the same time, a hierarchical vision system, is suggested as a working model of biological vision [47], based on a pre-configured hierarchy of modules, such as Gabor edge detectors, which are well represented by wavelet transforms [48], a mathematical theory that gives new insights in signal decomposition.

In the 2000's, kernel methods [49, 50] and unsupervised methods for learning sparse features and representations become widespread [51–53], and, the existence of an intrinsic universal learning architecture is strongly supported by a groundbreaking neuroscientific study [54] which shows that not only can the rewired auditory cortex develop the specific Gabor features characteristic of visual cortex, but it can also become functionally visual [55, 56]. Along those lines, in 2006, a greedy algorithm is proposed to train Deep Belief Networks (DBNs) [57] in what is considered to be the breakthrough that initiated the third wave of neural networks and introduces the term Deep Learning.

In the 2010's, Deep Learning becomes feasible and is scaled up on fast parallel hardware (GPUs). Deep Convolutional Networks [43], Variational Autoencoders [58] and Generative Adversarial Networks (GANs) [59] are employed to learn hierarchical sparse representations that are structurally and functionally similar to representations in analogous regions of biological cortex, improving the state-of-the-art in speech and visual object recognition, and, as will be discussed, semantic segmentation as well.

2.3 Deep Convolutional Neural Networks

Deep learning is a rich family of methods, encompassing neural networks, hierarchical probabilistic models, and a variety of unsupervised and supervised feature learning algorithms. The recent surge of interest in deep learning methods is due to the fact that they have been shown to outperform previous state-of-the-art techniques in several tasks, including semantic segmentation.

In particular, Convolutional Neural Networks (CNN), first proposed by Lecun in [11], are a class of deep neural networks most commonly used in visual imagery, and has been shown to be able to extract important visual features that resemble the ones generated in the visual cortex of humans [12, 13].

Convolutional Neural Networks (CNNs) were inspired by the visual system's structure, and in particular by the models of it proposed in [60]. A CNN has a hierarchical architecture. It comprises three main types of neural layers, namely,

- convolutional layers
- pooling layers, and
- fully connected layers.

Each type of layer plays a different role. Figure 2.1 shows a CNN architecture for an object detection task, and Figure 2.2 shows a CNN architecture for an image segmentation task.

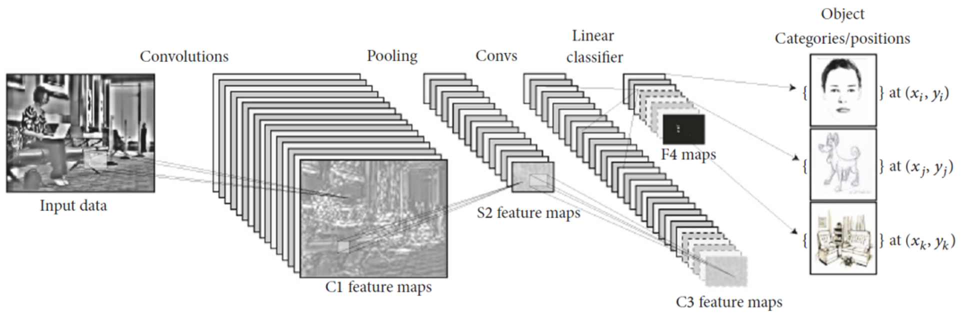


Figure 2.1: Example architecture of a CNN for a computer vision task (object detection) [13].

Starting from the input signal x, each subsequent layer x_j is computed as

$$x_j = \rho W_j x_{j-1} \tag{2.1}$$

Here W_j is a convolutional operator and ρ is a rectifier function, usually the ReLU function $max(x, 0)$, or the sigmoid function $s(x) = \frac{1}{1+e^{-x}}$. Because W_j represents a convolution, it is easier to think of it as a stack of convolutional filters, such that the layers become filter maps and each layer can be written as a sum of convolutions of the previous layer, i.e.

$$x_j(u, k_j) = \rho\left(\sum_k (x_{j-1}(\cdot, k) * W_{j,k_j}(\cdot, k))(u)\right) \tag{2.2}$$

where $*$ represents the discrete convolution operator:

$$(f * g)(x) = \sum_{u=-\infty}^{\infty} f(u)g(x-u) \tag{2.3}$$

Typically a downsampling step is implemented after the application of the non-linear rectifying function ρ, and the downsampling is done by max-pooling.

We note that convolution operators that take place inside the neural network are represented by W_j, which, for inputs of one dimension takes the form of a matrix, but for higher dimensions, including the two-dimensional images, takes the form of a general tensor.

From a designer's viewpoint, the effect of the various hyper-parameters of the network needs to be considered. A brief overview is presented next.

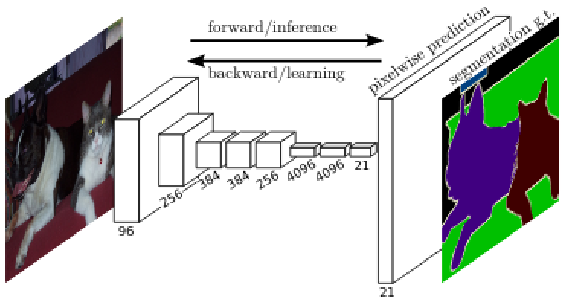


Figure 2.2: Fully Convolutional Network figure by Long et a [9].

2.3.1 Hyper-parameters

In this section, we discuss the effect of various parameters of the convolutional neural network from a practitional standpoint.

Input layers

The input layer brings the information into the system for further processing. For images as inputs the input layer typically consists of three channels (RGB) of arrays of pixel values. Depth, as well as different viewpoints may be used as additional channels.

Convolutional layers

Convolutional layers are usually the first layers after the input layer as they has the advantage that it scales nicely with the size of the input images. Convolutional layers are filters of certain dimension that is being applied to the whole image in the entire depth and compute the output of neurons that are connected to local regions in the input. They compute a dot product between their convolution filter and a small region they are connected to in the input image matrix. A convolutional filter or kernel decodes the informations in pixels. Each matrix element in the convolution filter is the weights that are being trained. These weights will impact the extracted convolved features. An example of an image with one channel can be seen in figure 2.3 where we apply a filter to extract vertical edges. By convolving the defined filter over the image and calculating the results we get the activation map that give us the information of where the vertical edges are.

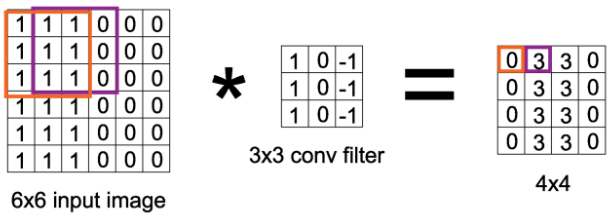


Figure 2.3: An example of a 2D convolution operation.

Convolutional layers are defined using several parameters tree of which will be explained in depth in the following:

- Depth
- Stride
- Zero Padding

The depth parameter corresponds to the number of filters we want to apply in an image. Different filters extract different information for the image region such as edges or colors. The term stride indicates the step that the filter uses when it interacts with input image. For example a stride with value 1 indicated that the filter interacts with the image by passing one pixel per time. Convolutional layers create a very precise output (feature map) where a change in the input image can generate a fully different feature map. Increasing the stride parameter is a solution to this problem. Bigger stride values jumps more pixel and the result is smaller outputs. By changing the stride of the convolution filter along the input matrix we achieve down-sampling. As a result a lower resolution matrix of the input matrix is created that still contains the important structural elements, without the fine detail that may not be as useful to the task.

The zero padding parameter is a way to “resize” the output image by symmetrically adding zeros to an image tensor in order to adjust the matrix. With this technique we can control the size of the output so it will have the same height and width with the input.

Among the different convolutional neural network approaches, in this work we adopt the key ideas from the MobileNetV2 network [14] which based on depthwise separable convolutions. The basic idea of separable convolutions is that they split the convolution into two separable layers. The first layer is the depthwise convolution and the second the pointwise. With depthwise convolution, a single convolutional layers is being applied thought all the input channels and collects spatial features from each one of them and with pointwise we can compute linear combinations of the input channels and extract new features.

Activation layers

Activation layers provide a non linearity to the model and are an important part of the architecture. The activation functions ρ are often rectifying functions, usually the ReLU function $max(x, 0)$, or the sigmoid function $s(x) = \frac{1}{1+e^{-x}}$.

For multiclass classification problems, however, the softmax operator is widely used, as it translates to conditional probabilities, i.e. probabilities for a data point to belong to a specific class. The softmax operator is defined as follows:

$$\rho(x)_i = \frac{e^{x_i}}{\sum_j^d e^{x_j}}, \quad x = [x_1, \ldots, x_d]^T \tag{2.4}$$

Softmax is also the only activation function that can be used with the categorical crossentropy loss function.

Pooling layers

Pooling layers are usually added after a convolutional layers or a ReLU layer. They reduce the parameters to learn and as a result the computation of the network. Like convolutional layers, pooling operations are also like filters and they are applied along to each feature map. The difference between the convolutional layers and pooling layers is that the last summarise features lying within the region covered by the filter instead of precisely positioned as the first do. This makes the model more robust to variations in the position of the features in the input image. They usually reduce the dimensions of the feature map by half and there are two common operations :

- Max Pooling
- Average Pooling

Max Pooling filters performs a downsampling in the feature map and they divide it in smaller parts where then they compute the maximum value in these part.

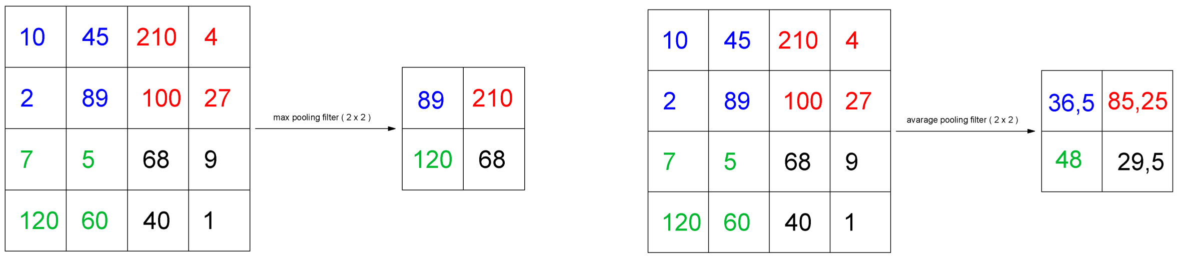


Figure 2.4: Max-Pooling (left) and AVerage-pooling (right) filters.

Average pooling compute the average values present in the region of feature map covered by the filter.

Fully connected layers

The structure of the fully connected layers is represented in figure 2.5 where each colored circle is a neuron and the lines in between show the connections.

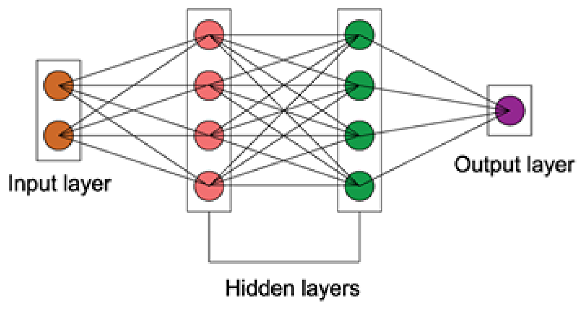


Figure 2.5: Fully connected layers.

Fully connected layers takes the output from the previous layers and turns them into a single vector. If we assume that we want to extract features from a color image (RGB) that has a size 25*25*3 (width*height*channels) then we have an image of 1,875 pixels where each pixel is an input (X) in the input layer. If we assume that the next layer has a 4 neurons then the system has 1875 * 4 = 7500 parameters to calculate. More complex images and more dense networks requires more computational cost. For that reason in complex networks fully connected layers are usually used only as last layers in a dense network.

2.3.2 Training Algorithms and Evaluation

Deep neural networks are trained with the Backpropagation algorithm [42], which is a stochastic gradient descent algorithm [61] that tries to minimize the expected error:

$$E(X, w) = \frac{1}{n} \sum_{n} d(y_d - y) \tag{2.5}$$

with respect to the weights w of the neurons. Here y_d is the desired output of the network and y the actual output. The dissimilarity metric d, which is called the loss function, is usually chosen to be the Euclidean distance, but can be some other measure, such as the categorical crossentropy, which is the one that we are going to be using in this work.

Now, stochastic gradient descent takes the form:

$$w_{t+1} = w_t - a_t \frac{\partial E(X, w_t)}{\partial w} \tag{2.6}$$

and the partial derivatives propagate through the network as a result of the chain rule, which gives the training algorithm the name backpropagation.

An important parameter for the convergence of the stochastic gradient descent algorithms is the choice of the stepsizes a_t, which are called the learning rates. In particular, adaptive learning rates have been studied for accelerated learning, with the Adam method [62] being one of the most widely used. We will adopt this method as well.

Finally, in order to evaluate the performance of the network, we are going to be using the classification loss and the precision measure (true positives over all positives).

Chapter 3

Methodology

In this chapter we showcase our methodology, including the datasets used (simulated and real) and the proposed neural network learning architecture. We first introduce a novel deep convolutional neural network to be used for semantic segmentation in assistive robotics for activities of daily living, and test its performance on a dataset of simulated RBG-D images. In order to assess our methodology in real-life scenarios, we create a high-quality real-life labelled dataset of images from a breakfast table scenario, that are generated from two different camera views.

3.1 Deep Learning for Semantic Segmentation

For our Breakfast scenario approach we need a lightweight neural network in order to be easily used with mobile robots. Among the numerous deep learning architectures that exist in the literature, we build upon the MobileNet V2 [14] architecture for various reasons. MobileNetV2 is an optimised neural network for mobile devices, offers high accuracy values and requires low computational power. The main idea of MobileNetV2 is to use *depthwise separable convolutions* with residuals to build lighter deep neural networks. In convolutional layers, as explained in Chapter 2, the convolution filter is applied to each of the channels of the input image. MobileNetV2 uses convolution filter only in the first layer and the next layers consists a 3x3 depthwise and 1x1 pointwise convolutions after one another. The depthwise convolution does the convolution on each channel separately and is used to filter the input channels. Pointwise convolution is similar to regular convolution and is responsible for extracting new features from depthwise convolutions.

According to Sandeler et al. [14] depthwise separable convolutions is comparable in performance to convolutional filters while, at the same time, reduce significant the computational cost. In MobileNetV2, there are two types of blocks. One block is a residual block with stride of 1 and a second block with stride of 2 for downsampling. Both blocks include three layers mentioned above (see Figure 3.1). Finally, for the activation function, the authors use a variant of ReLU, called Relu6 ($\rho(x) = \min(\max(0, x), 6)$) in an attempt to enhance the performance of the training rule when faced with low-precision computations.

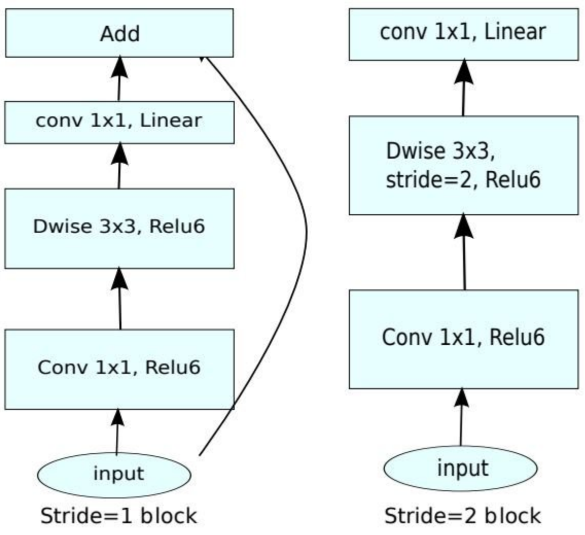


Figure 3.1: MobileNetV2 blocks as proposed in [14].

In our implementation we use MobileNetV2 as a backbone with transferring knowledge from ImageNet dataset [63]. The idea of transfer learning is to re-use model weights from pre-trained models in order to minimise training time on large datasets. In Figure 3.2 we can observe the architecture of our model with the simulated RGB inputs. Our input dataset has input shape (240,424,3). After the backbone architecture the output shape is (8,14,320). Before we start the upsampling we add block17 and block18. Block 17 is consisted of a DepthwiseConv2D layer with stride 2 followed by a ReLU layer and a Convolutional 2D with ReLU activation function. Block 18 is a zero Padding 2D layer needed in order to achieve the desired dimensionality for upsampling layers. Each of the upsampling blocks from block 19 to 22 are consisted of 2 layers, an upsampling layer and a concatenate layer. There are two methods for merging information between layers. Adding and concatenating. Usually the first is used when we want to to interpret one of the inputs as a residual correction to the other input. Concatenating on the other hand is more sufficient when the two inputs aren't very closely related. The last layer of our structure is a convolutional 2D array with softmax activation function. As mentioned in chapter 2, the softmax function returns the conditional probability for a data point to belong to a specific class. Lastly, we use the Adam optimiser as a training rule, and categorical crossentropy as a loss function.

In order to prevent overfitting, we randomly split our datasets to training and testing sets. We manually set a 10% of data used for testing. Training with more data can help the algorithms perform more accurate predictions.

The same blocks, as described above, were used for both the simulated and real dataset 3.3 with different input and output layers. MobileNetV2 architecture requires exactly 3 inputs channels, and, as a result training the network with RGB-D datasets was not achievable. For this reason, we designed a new model based in MobinetNetV2 blocks in order to be trained with RGB-D inputs. Our new model accepts 4 channels as inputs and the rest architecture is the same as has already described. For RGB-D input we did not use transfer learning.

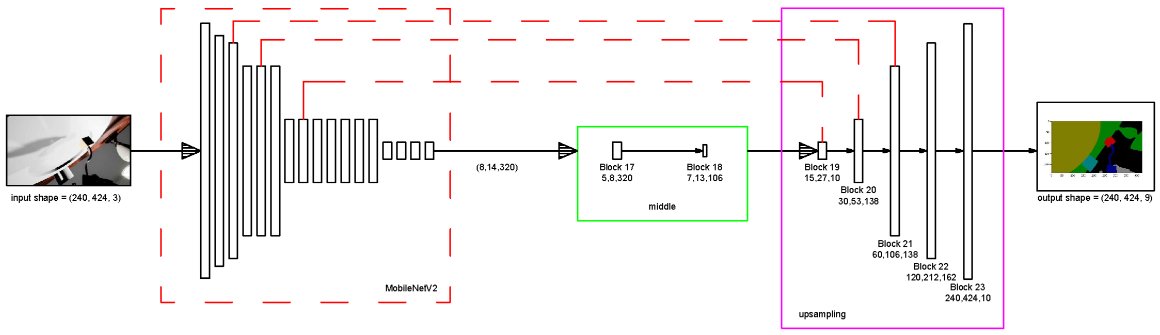


Figure 3.2: Block Diagram of the proposed architecture for simulated Dataset.

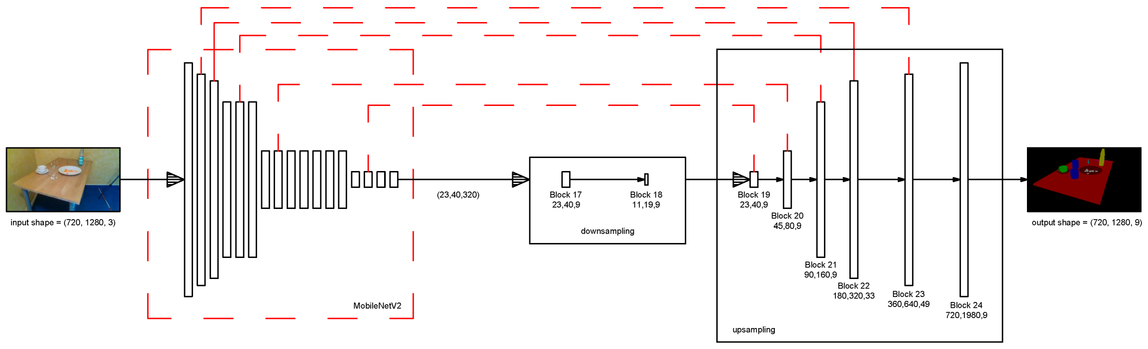


Figure 3.3: Block Diagram of the proposed architecture for real Dataset.

3.2 Simulated Dataset

The simulated data consist of RGB-D, segmented images that were produced by the Unreal Engine. The dataset size is 18611 images (424 height x 240 width) and the scene presents a breakfast scenario with 10 classes of objects. The objects presented are "Standard", "Room", "Furniture", "Plate", "Fork", "Bottle", "Cup", "Robot", "Head", "Food". The camera model used by the Unreal Engine in this scenario, emulates an Intel RealSense camera, which is the model that we are going to use later to record the real-life images for our real dataset. The camera is mounted on the robot's end-effector. This approach is adopted to examine how the network's efficiency in recognising objects is affected by different camera angles 3.5, and under what conditions it fails to accurately classify them.

An illustration of the dataset is provided in Figure 3.5, which shows the breakfast scenario approach where the robot arm faces the scene, and in Figure 3.4, which shows the camera view when the robot arm grabs the food from the table.

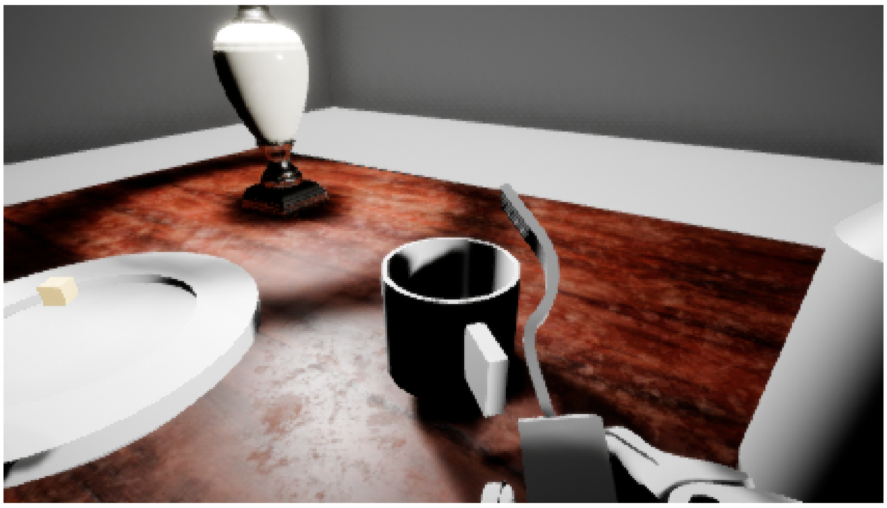


Figure 3.4: RGB Image from simulated with normal scene view.

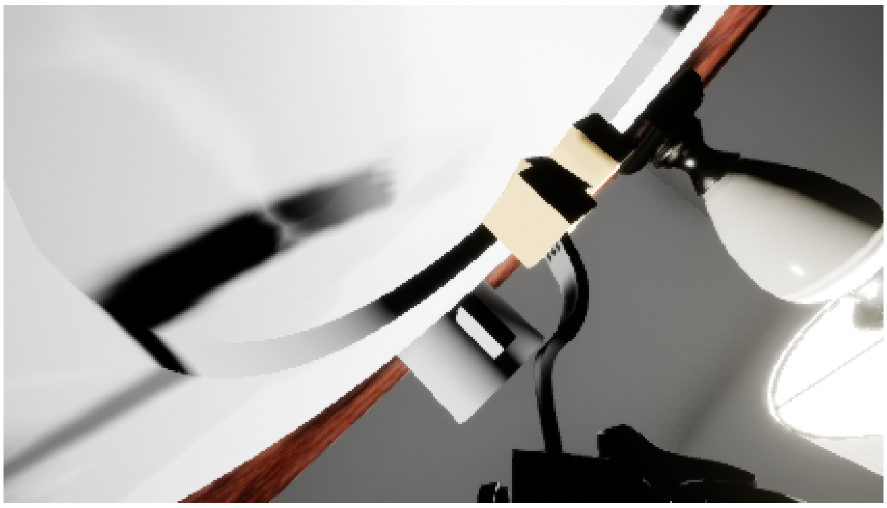


Figure 3.5: RGB Image from simulated data (observation from an angle).

3.3 Real Dataset

In order to assess our methodology in real-life scenarios, we create a high-quality real-life labeled dataset of images from a breakfast table scenario, that are generated from two different camera views. RGB-Depth images were taken using two Intel Realsense cameras pointing on the table, one from the long side and the other from the short side as shown in figure 3.6. For this dataset 448 RGB-D images were taken in total, 224 (1280*720 pixels) images from each side of the table. There are 9 distinct objects:

(background, table, cup, bottle, glass, fork, knife, food, plate)

The scene was in a room with yellow colored wall with partly artificial light, partly sunlight. Lighting conditions poses a great challenge to the accuracy of object detection. The intensity of the light affects both the color and the texture characteristics of objects. Moreover angle lighting can cause shadows which has also and import effect in network's accuracy. In our collected dataset due to the nature of our research, we couldn't achieve to eliminate shadows from both table views. As we can see in 3.6 from the short side of the table the shadows are not visible while we can observe them by the view from long side of the table. We will use these contours to asses the network's accuracy under real-life light artifacts.

Compared to the existing datasets, we have collected our dataset with:

- transparent glass
- overlapped objects
- objects what are not visible from both sides
- objects what are in different angles.

All the above assumptions could be seen in a Breakfast scenario and it's interesting to investigate network performance in these unusual cases. Some of the above assumptions can be observed in figure 3.6. Finally, the labelling process was done using the labelme tool [64]. An example of created labels is presented in figure 3.7.

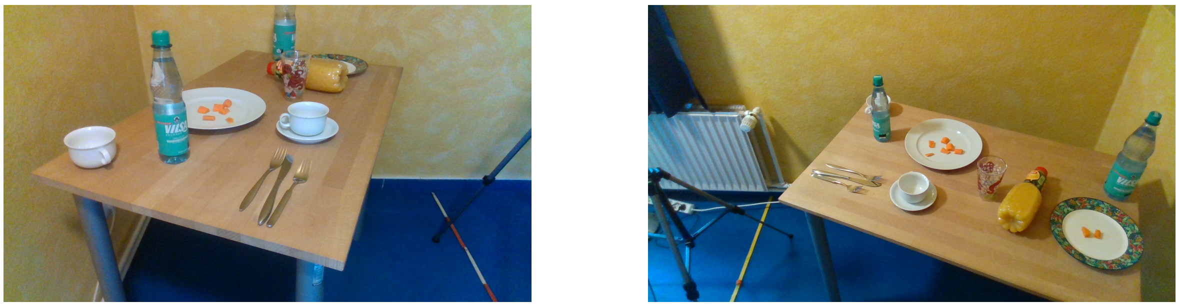


Figure 3.6: Table side view.

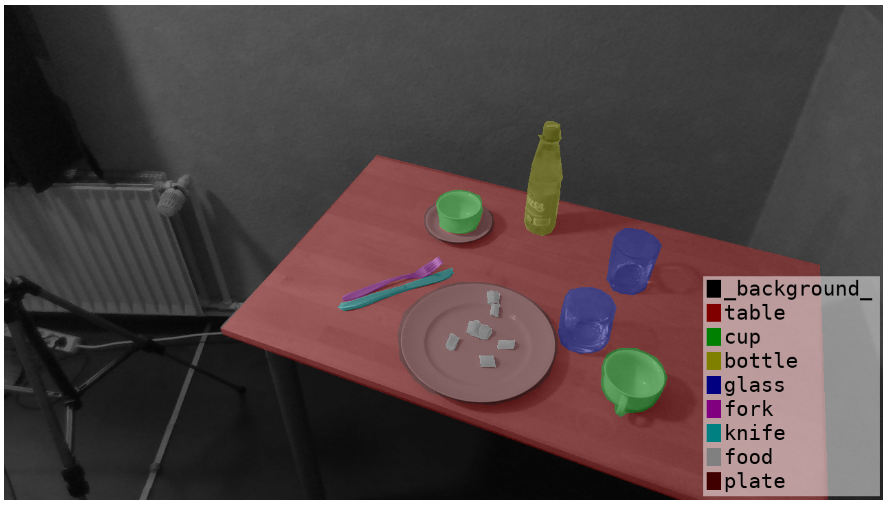


Figure 3.7: Labeled scene.

Chapter 4

Experimental Evaluation

In this section we evaluate the performance of the proposed neural network architecture in the simulated and real-life data, for RGB and RGB-D images. We also test our learning algorithm in multi-view input images.

4.1 RGB Data

As shown in Table 4.1, the proposed learning architecture achieves state of the art accuracy when trained and tested on the RGB images, regardless of whether these are generated by simulation or captured in real life. The latter is very important and is telling of the robustness of this architecture in sensor artifacts, especially in those coming from strange light conditions or bad camera angles. We would like to emphasize that this is a very encouraging result, since the end goal of the proposed architecture is to be used in real-life assistive robotics scenarios.

Input Signals	Dataset length	Epochs	s/step	Duration	loss	precision
RGB simulated	1024	250	$2s/step$	$500min$	0.043	0.988
RGB real headset view	224	150	$2s/step$	$300min$	0.080	0.978
RGB real table view	224	150	$2s/step$	$300min$	0.157	0.980

Table 4.1: The loss and precision of the model across different epochs and inputs.

In the following figures, we present an example of the actual input and the produced segmentation of (a) an RGB simulated image, (b) a real RGB image from the headset view, and (c) a real RGB image from the table view.

4.1.1 Simulated data

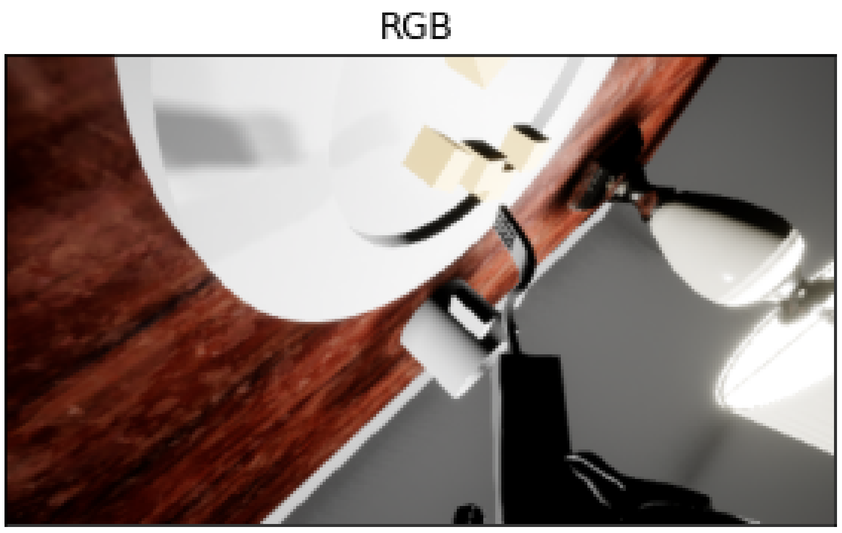


Figure 4.1: An example of an input simulated image and the model's segmentation.

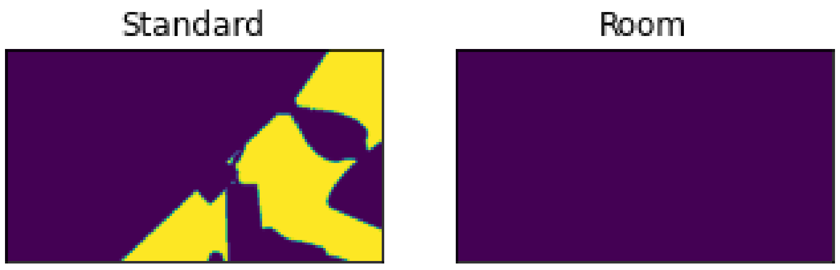


Figure 4.2: True segments values for standard and room.

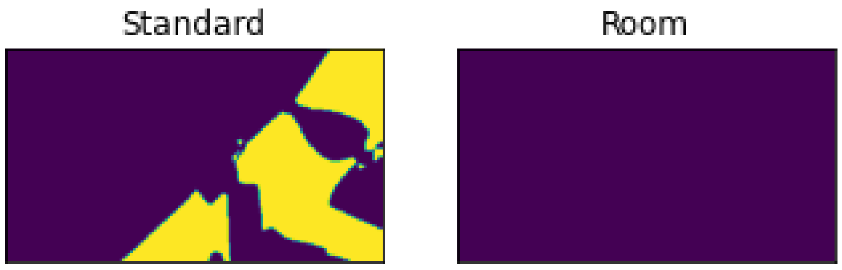


Figure 4.3: Network results for standard and room.

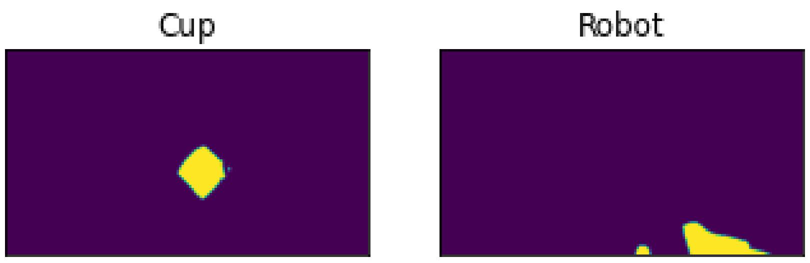


Figure 4.4: True segments values for cup and robot.

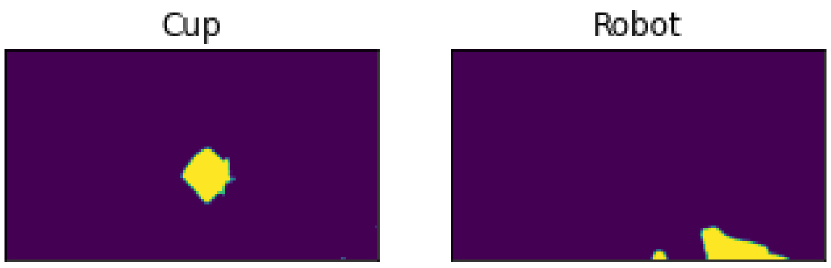


Figure 4.5: Network segments values for cup and robot.

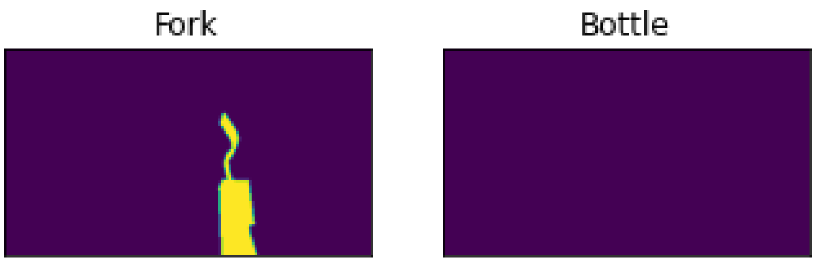


Figure 4.6: True segments values for fork and bottle.

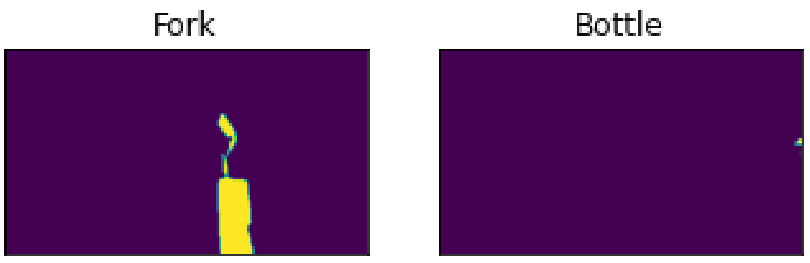


Figure 4.7: Network segments values for fork and bottle.

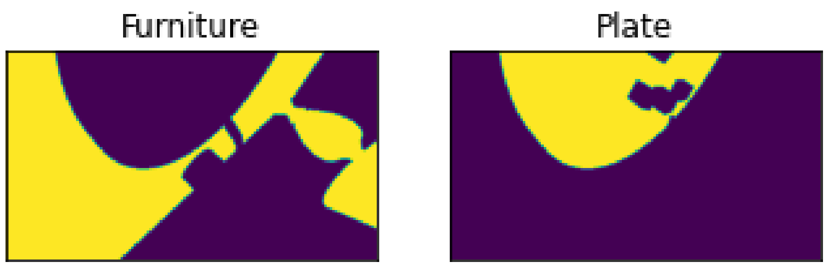


Figure 4.8: True segments values for furniture and plate.

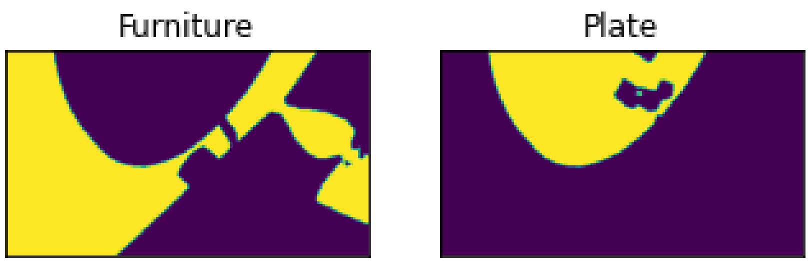


Figure 4.9: Network segments values for furniture and plate.

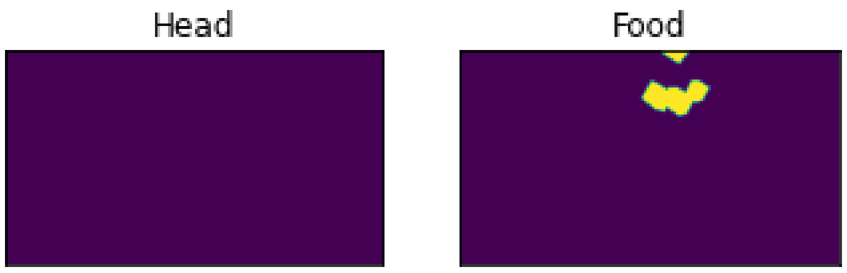


Figure 4.10: True segments values for head and food.

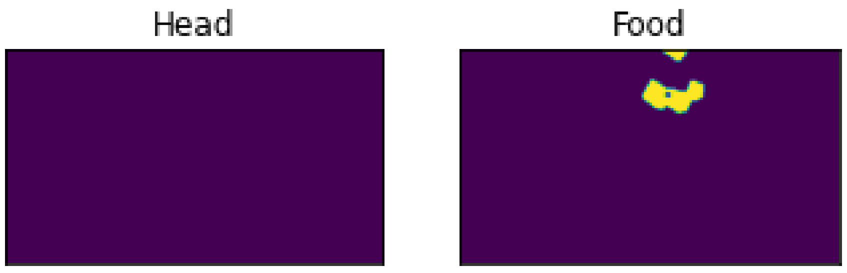


Figure 4.11: Network segments values for head and food.

4.1.2 Real data (headset view)

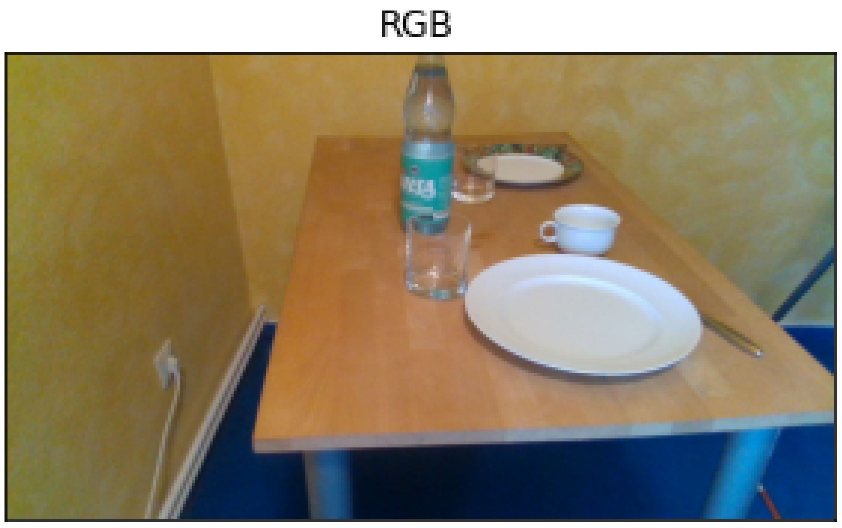


Figure 4.12: An example of an input image (headset view) and the model's segmentation.

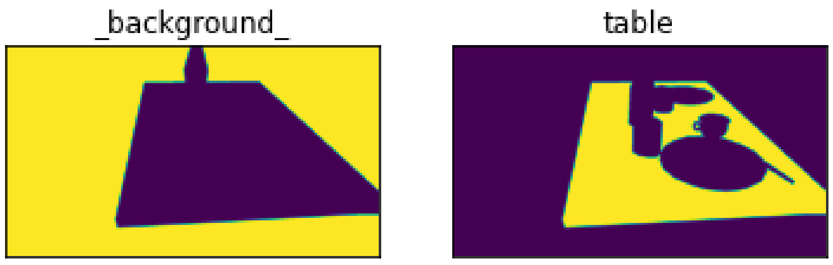


Figure 4.13: True segments value for background and table.

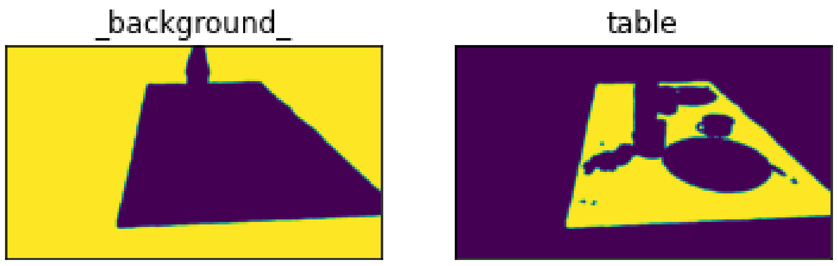


Figure 4.14: Network values for background and table.

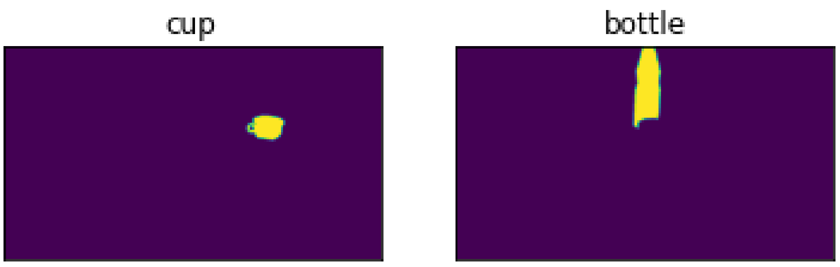


Figure 4.15: True segments for cup and bottle.

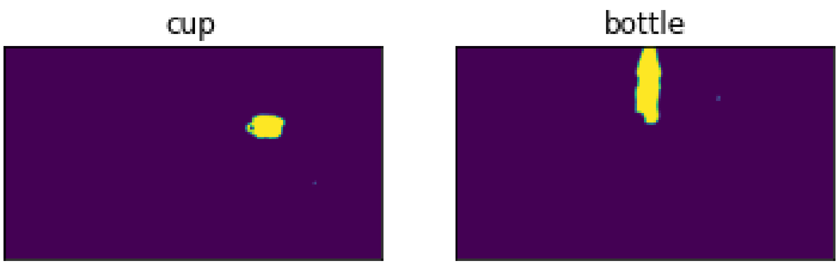


Figure 4.16: Network segments for cup and bottle.

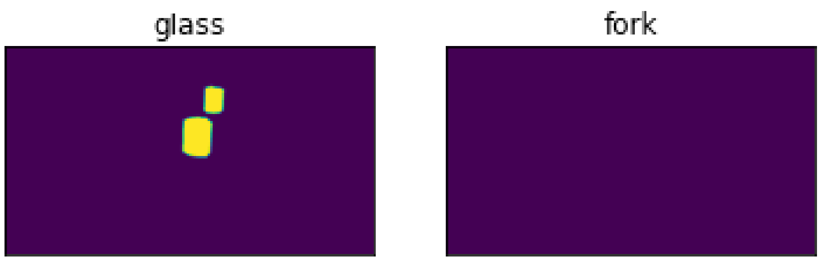


Figure 4.17: True segments for glass and fork.

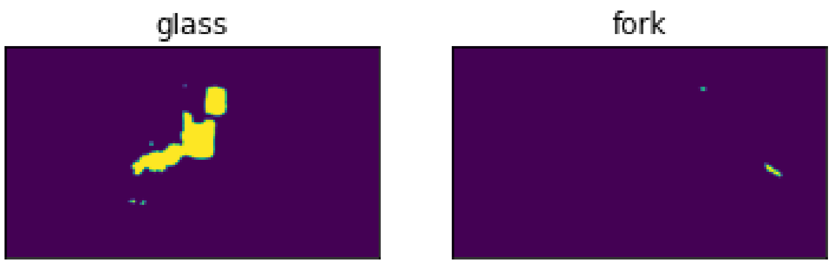


Figure 4.18: Network segments for glass and fork.

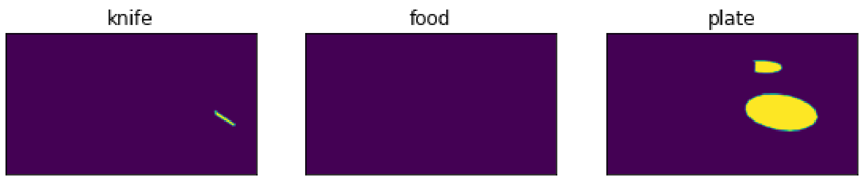


Figure 4.19: True segments for knife food and plate.

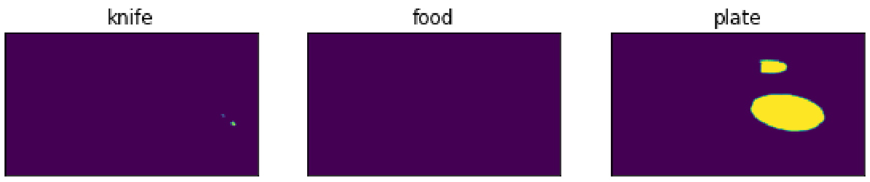


Figure 4.20: Network segments for knife food and plate.

4.1.3 Real data (table view)

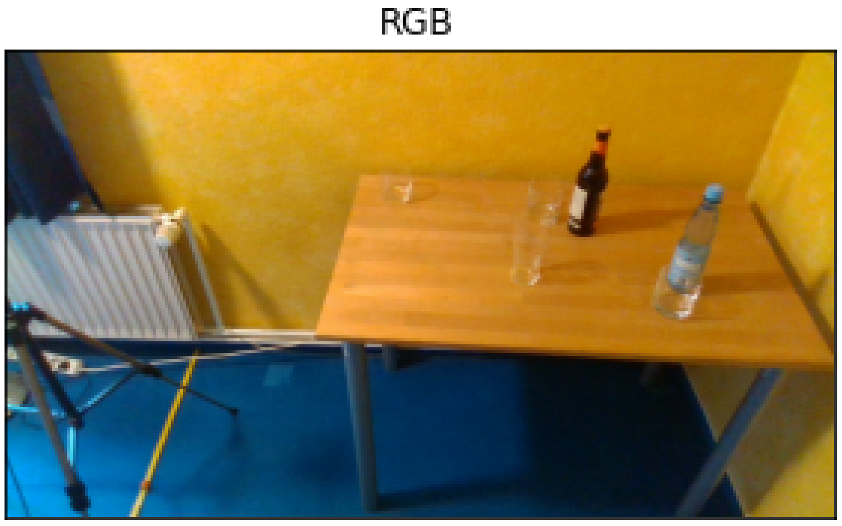


Figure 4.21: An example of an input image (table view) and the model's segmentation.

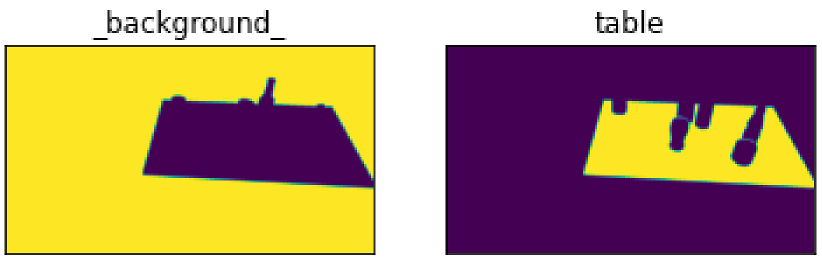


Figure 4.22: True segments value for background and table.

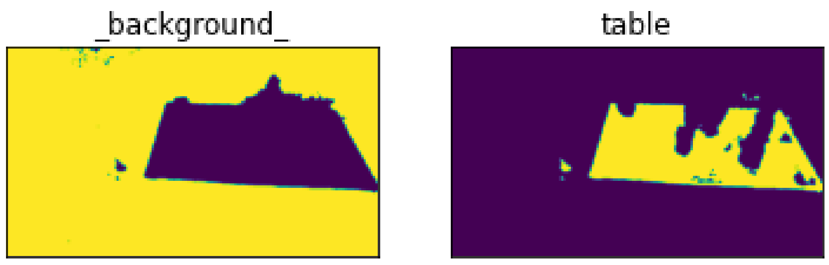


Figure 4.23: Network values for background and table.

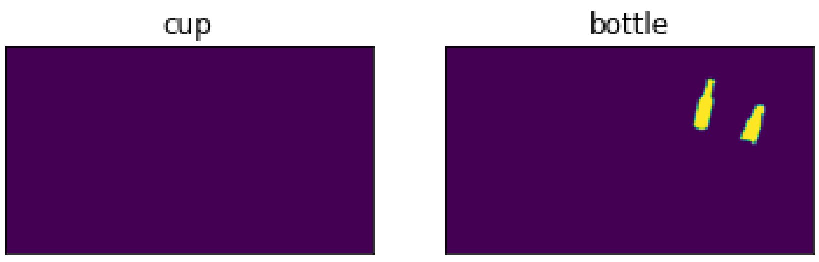


Figure 4.24: True segments for cup and bottle.

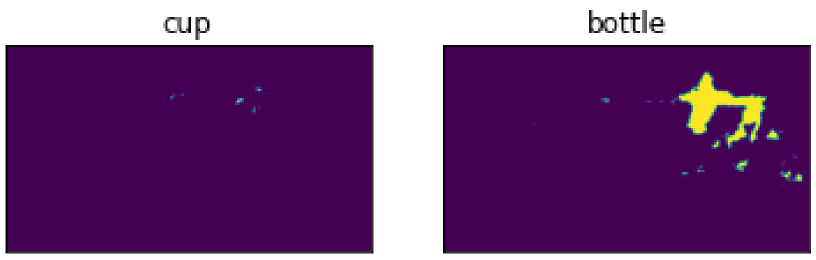


Figure 4.25: Network segments for cup and bottle.

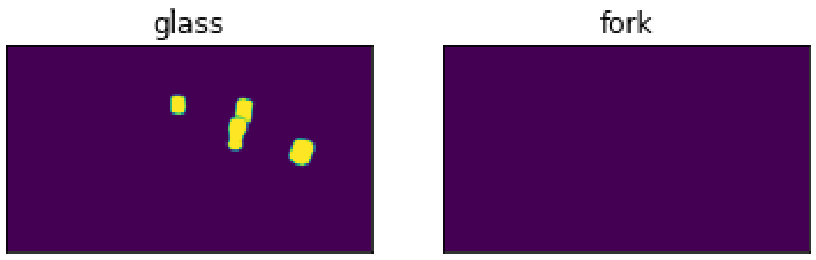


Figure 4.26: True segments for glass and fork.

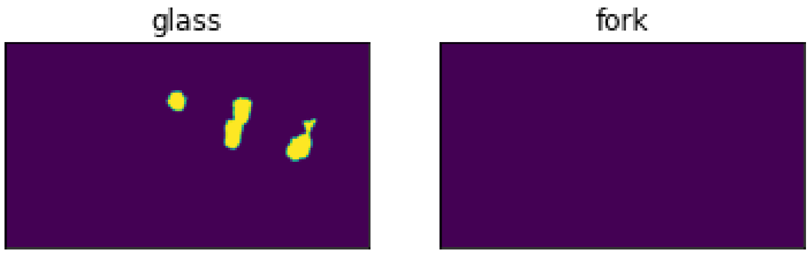


Figure 4.27: Network segments for glass and fork.

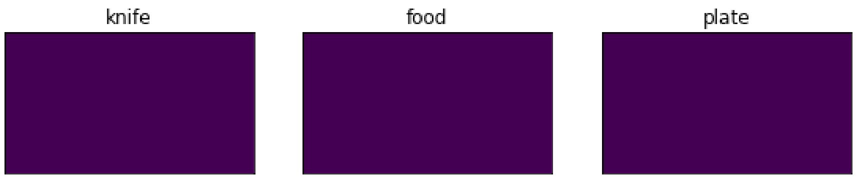


Figure 4.28: True segments for knife food and plate.

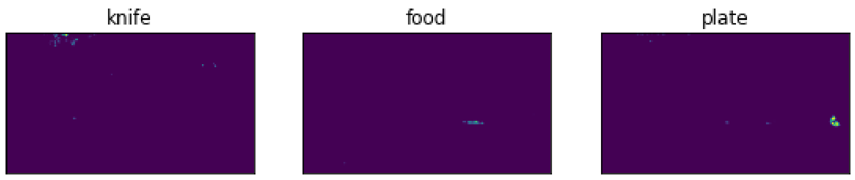


Figure 4.29: Network segments for knife food and plate.

4.2 RGB-D Data

In this section we test our model using the RGB-Depth dataset that we collected. As shown in Table 4.2, the proposed learning architecture achieves state of the art accuracy when trained and tested on the RGB-D images, regardless of whether these are generated by simulation or captured in real life. The latter is very important and is telling of the robustness of this architecture in sensor artifacts, especially in those coming from strange light conditions or bad camera angles. We would like to emphasize that this is a very encouraging result, since the end goal of the proposed architecture is to be used in real-life assistive robotics scenarios.

Compared to the case of RGB input only, we conclude that the information of depth adds no statistically significant benefits to the segmentation performance of our model. This may be a characteristic of the datasets at hand, or a feature of the model itself. Further experimentation needs to be conducted in order to deeply understand this phenomenon. However, based on these results, an RGB-D camera is not required for accurate image segmentation in our breakfast scenario.

In the following figures, we present an example of the actual input and the produced segmentation of (a) an RGB-D simulated image, and (b) a real RGB-D image from the headset view.

Input Signals	Dataset length	Epochs	s/step	Duration	loss	precision
RGB-D simulated	18611	50	$2s/step$	$100min$	0.049	0.980
RGB-D simulated	1024	1000	$2s/step$	$2000min$	0.145	0.972
RGB-D real headset	224	1000	$2s/step$	$2000min$	0.229	0.958
RGB-D real headset+table	448	1000	$2s/step$	$2000min$	0.154	0.966

Table 4.2: The loss and precision of the model across different epochs and inputs.

4.2.1 Simulated data

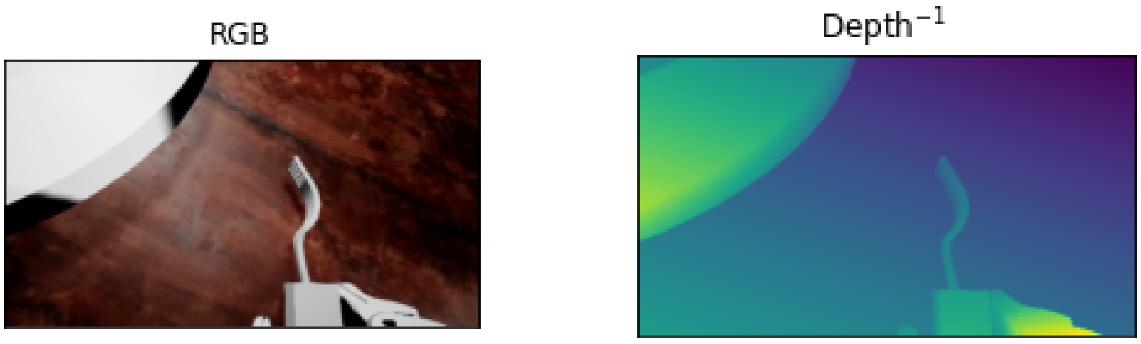


Figure 4.30: Input RGB and Depth image.

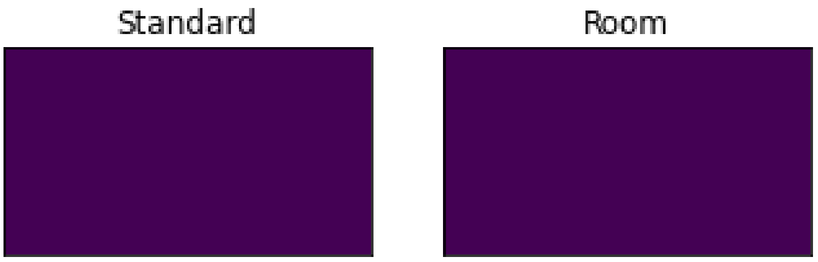


Figure 4.31: True segments for standard and room.

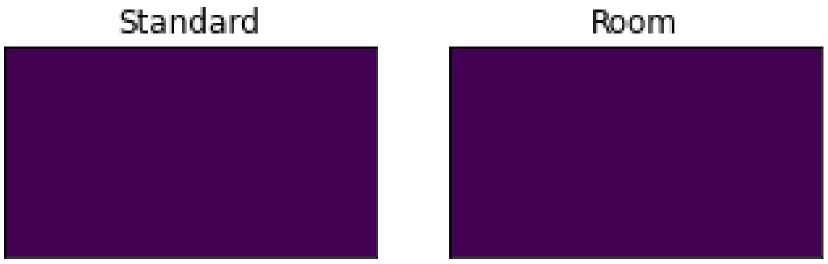


Figure 4.32: Network segments for standard and room.

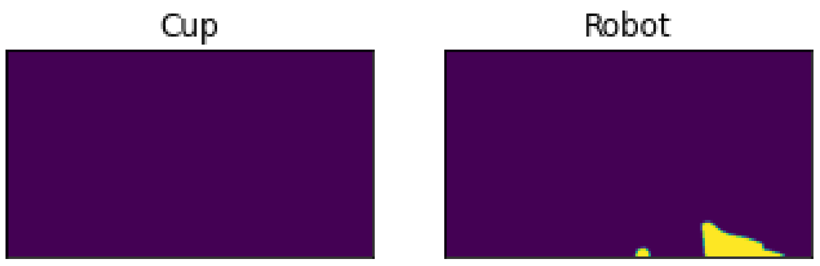


Figure 4.33: True segments for cup and robot.

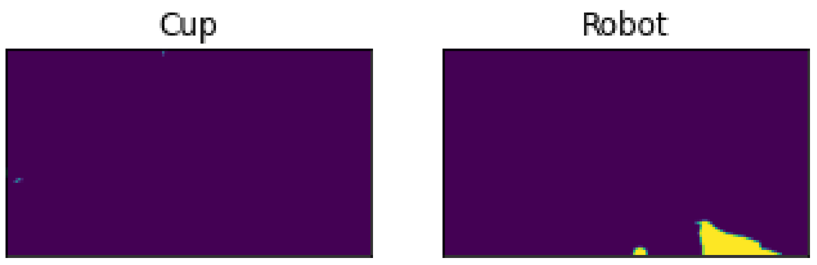


Figure 4.34: Network segments for cup and robot.

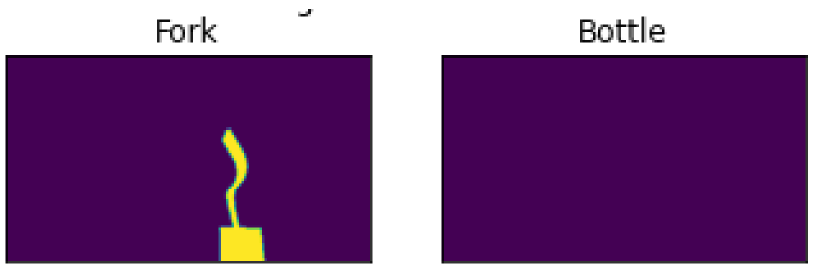


Figure 4.35: True segments for fork and bottle.

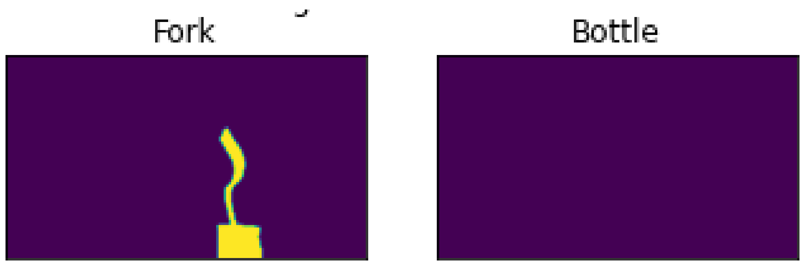


Figure 4.36: Network segments for fork and bottle.

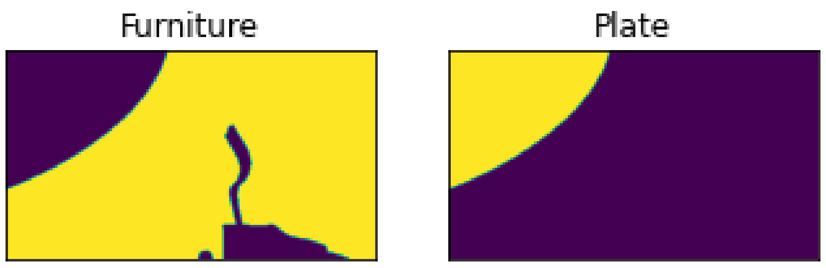


Figure 4.37: True segments for furniture and plate.

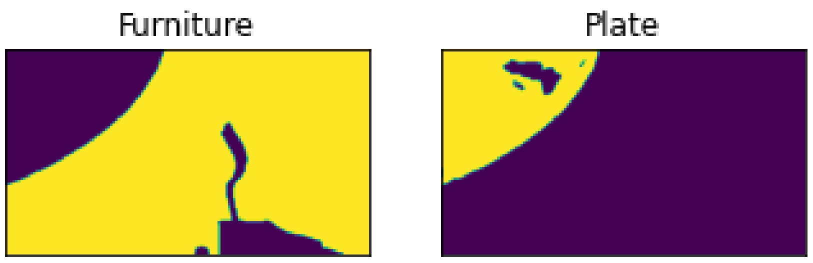


Figure 4.38: Network segments for furniture and plate.

Figure 4.39: True segments for head and food.

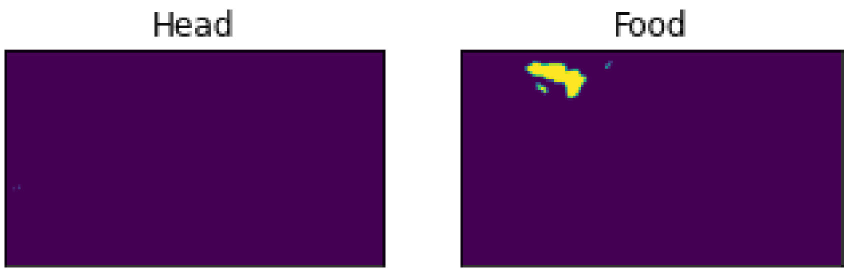


Figure 4.40: Network segments for head and food.

4.2.2 Real data (headset view)

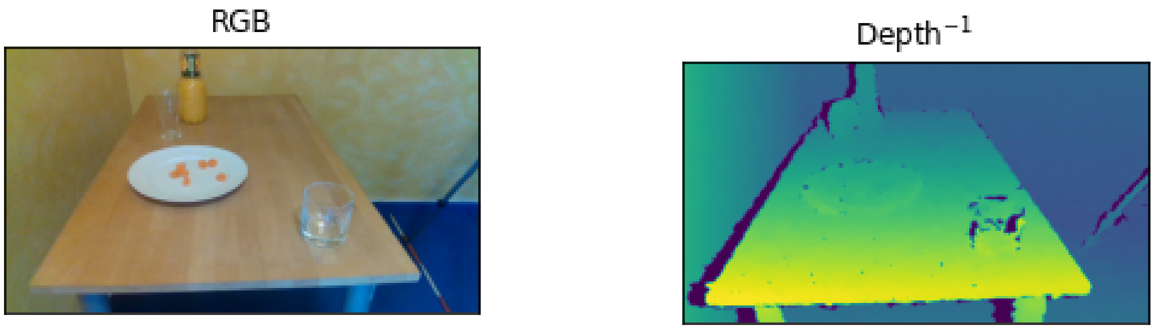


Figure 4.41: Input RGB and Depth image.

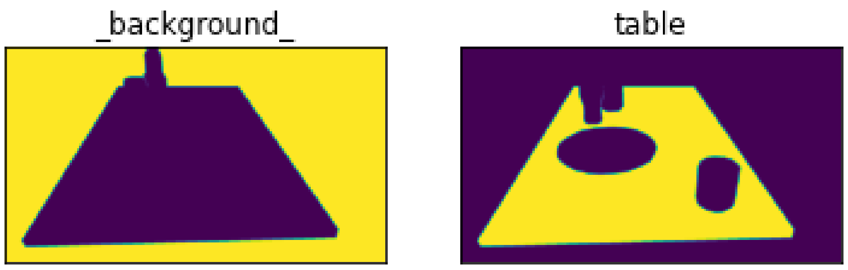


Figure 4.42: True segments for background and table.

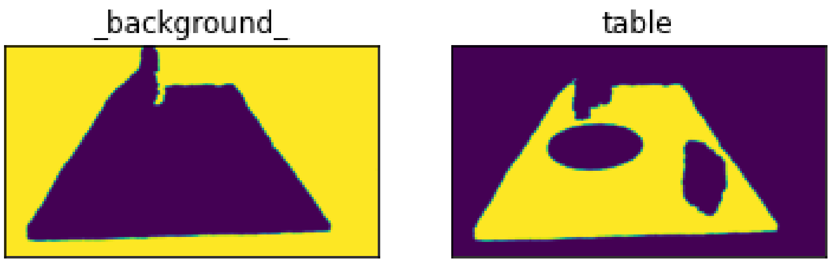


Figure 4.43: Network segments for background and table.

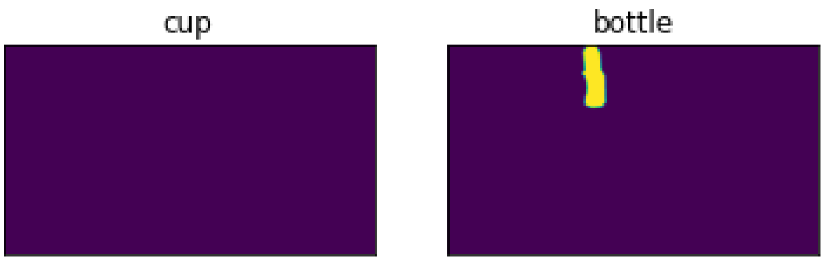


Figure 4.44: True segments for cup and bottle.

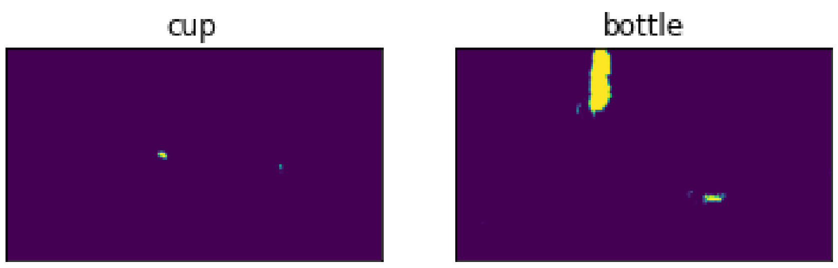


Figure 4.45: Network segments for cup and bottle.

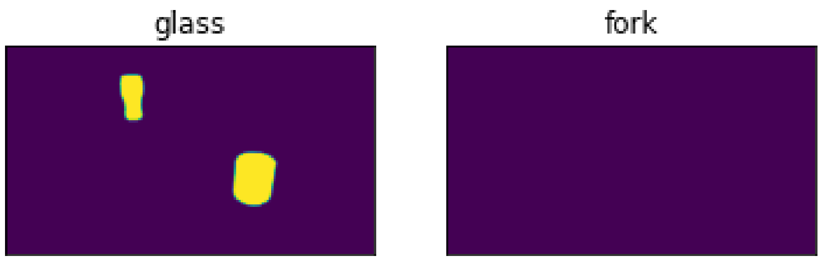


Figure 4.46: True segments for glass and fork.

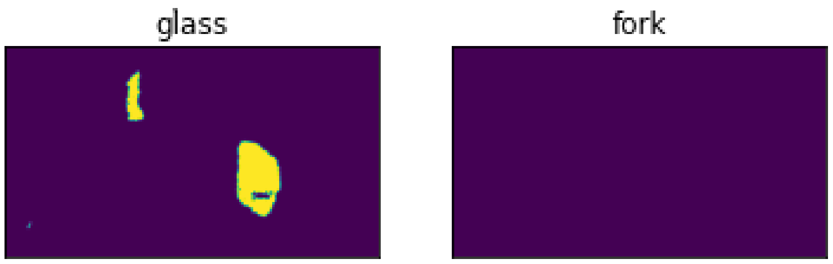


Figure 4.47: Network segments for glass and fork.

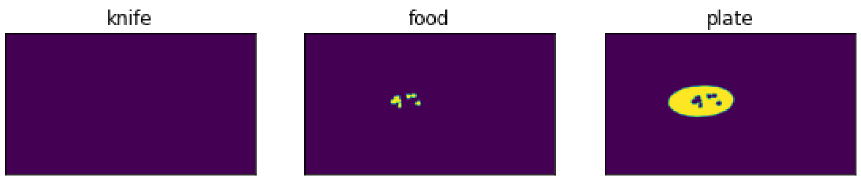


Figure 4.48: True segments for knife food and plate.

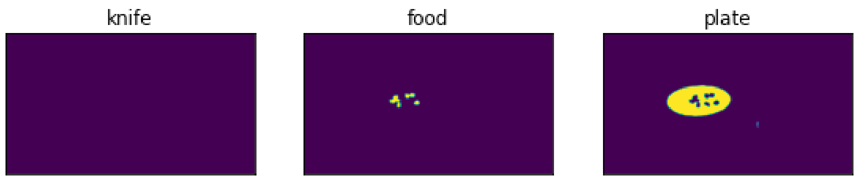


Figure 4.49: Network segments for knife food and plate.

4.3 Multiple view images

In this section we test our model using RGB input images from different viewpoints. As shown in Table 4.3, the proposed learning architecture achieves high precision and low loss in less training time comparatively to 4.2 and 4.1 .

Compared to the case of RGB-D inputs (from headset and table view), we conclude that using multiple view image dataset we achieved with less training data and also in significant less time, almost the same precision and loss.

We would like to point out this part as a very interesting result, since the end goal is to find a dataset for which our network performs with high precision, low loss in low training time.

Summarising we would like to re-mention that we achieved very promising results, since the end goal of the proposed architecture is to be used in real-life assistive robotics scenarios.

Input Signals	Dataset length	Epochs	Duration	s/step	loss	precision
RGB	224	100	$2s/step$	$200min$	0.444	0.909
RGB	224	150	$2s/step$	$300min$	0.155	0.978

Table 4.3: The loss and precision of the model.

In the following figures, we present an example of the actual input and the produced segmentation of a real RGB multi-view image.

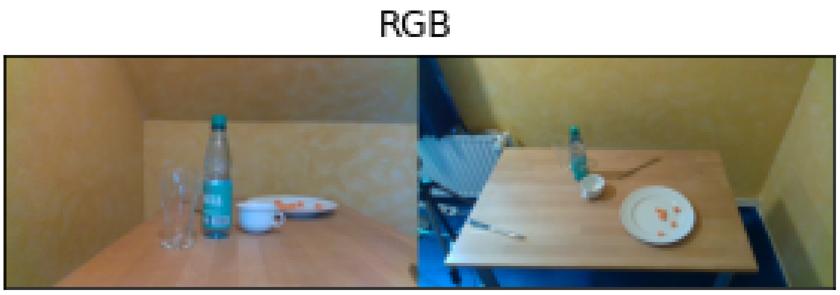


Figure 4.50: Input RGB data.

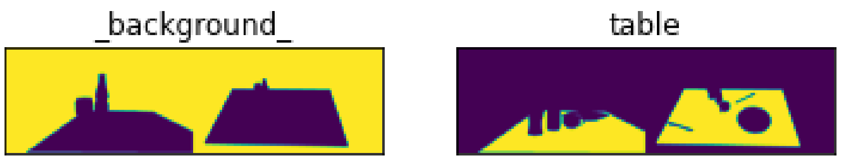
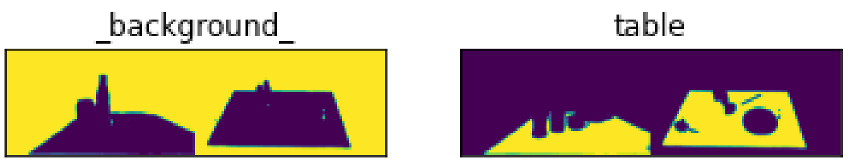


Figure 4.51: True values for background and table.



/

Figure 4.52: Network values for background and table.

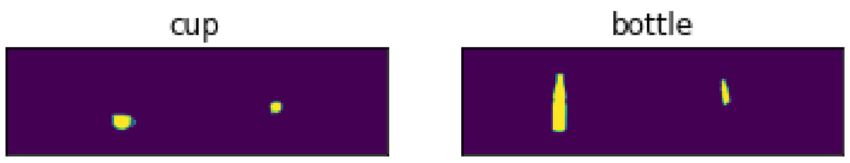


Figure 4.53: True values for cup and bottle.

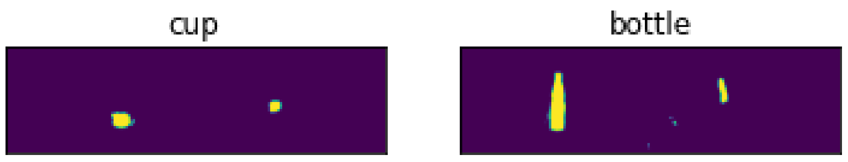


Figure 4.54: Network values for cup and bottle.

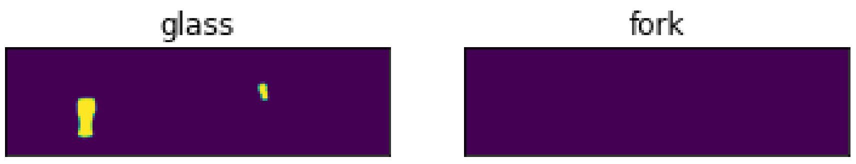


Figure 4.55: True values for glass and fork.

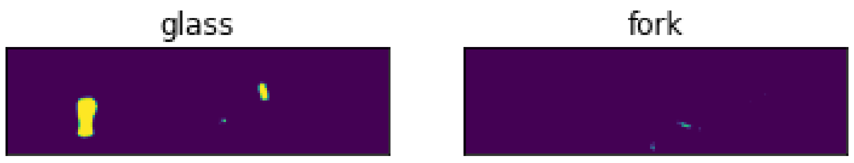


Figure 4.56: Network values for glass and fork.

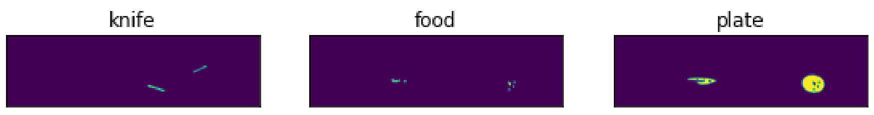


Figure 4.57: True values for knife food and plate.

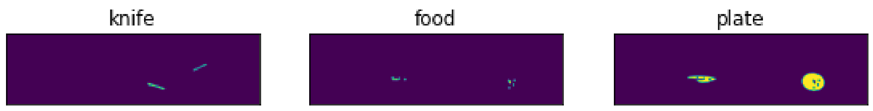


Figure 4.58: Network values for knife food and plate.

Chapter 5

Conclusion

We focused on semantic segmentation, the process of labelling each area –or pixel– of a digital image according to a representation class. We made use of deep learning methods, which, over the last years, have been shown to outperform previous state-of-the-art machine learning techniques in several fields, with computer vision being one of the most prominent cases. In particular, we developed a novel convolutional neural network architecture for semantic segmentation in the framework of Activity of Daily Life (ADL), with focus on semantic segmentation of images in a breakfast scenario. We used RGB and RGB-D images as input signals (images with 4 channels: RGB colors and Depth) and quantified the benefits of the additional information of depth in object classification. Next, we evaluated the performance of our deep neural network when trained with data from multiple viewpoints. We made use of two cameras pointing on the table with a different field of view. The first camera was placed on a headset (user's point of view) and the second on the table.

Furthermore, for this project we created a new dataset using two Intel RealSense camera sensors, in the apparatus mentioned above. This dataset was used to train three different convolutional neural networks for semantic segmentation. The first two CNNs were trained with the dataset collected from the headset (one with RGB and one with RGB-D information). The third CNN was trained with the dataset collected from the second camera, and the last one with the dataset from both camera views.

Quantifying the importance of multiple viewpoints and depth information in semantic segmentation will help in developing robust, fast, and more precise perception algorithms for object recognition in ADL applications. This is an important step towards the final goal of creating assistive robot agents (e.g. robotic arms, wheelchairs, and exoskeletons) that can help people with severe motor impairments enjoy their first meal of the day.

Bibliography

[1] R. Szeliski, *Computer vision: algorithms and applications.* Springer Science & Business Media, 2010.

[2] D. A. Forsyth and J. Ponce, *Computer vision: a modern approach.* Pearson„ 2012.

[3] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE transactions on systems, man, and cybernetics*, vol. 9, no. 1, pp. 62–66, 1979.

[4] R. Nock and F. Nielsen, "Statistical region merging," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 26, no. 11, pp. 1452–1458, 2004.

[5] N. Dhanachandra, K. Manglem, and Y. J. Chanu, "Image segmentation using k-means clustering algorithm and subtractive clustering algorithm," *Procedia Computer Science*, vol. 54, pp. 764–771, 2015.

[6] N. Plath, M. Toussaint, and S. Nakajima, "Multi-class image segmentation using conditional random fields and global classification," in *Proceedings of the 26th Annual International Conference on Machine Learning*, 2009, pp. 817–824.

[7] J.-L. Starck, M. Elad, and D. L. Donoho, "Image decomposition via the combination of sparse representations and a variational approach," *IEEE transactions on image processing*, vol. 14, no. 10, pp. 1570–1582, 2005.

[8] S. Minaee and Y. Wang, "An admm approach to masked signal decomposition using subspace representation," *IEEE Transactions on Image Processing*, vol. 28, no. 7, pp. 3192–3204, 2019.

[9] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[10] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," *arXiv preprint arXiv:1706.05587*, 2017.

[11] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[12] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[13] A. Voulodimos, N. Doulamis, A. Doulamis, and E. Protopapadakis, “Deep learning for computer vision: A brief review,” *Computational intelligence and neuroscience*, vol. 2018, 2018.

[14] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, “Mobilenetv2: Inverted residuals and linear bottlenecks,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 4510–4520.

[15] D. P. Miller, “Assistive robotics: an overview,” *Assistive Technology and Artificial Intelligence*, pp. 126–136, 1998.

[16] C. Martens, O. Prenzel, and A. Gräser, “The rehabilitation robots friend-i & ii: Daily life independency through semi-autonomous task-execution,” *Rehabilitation robotics*, vol. 1, pp. 137–162, 2007.

[17] I. Volosyak, O. Kouzmitcheva, D. Ristic, and A. Graser, “Improvement of visual perceptual capabilities by feedback structures for robotic system friend,” *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 35, no. 1, pp. 66–74, 2005.

[18] Z. Yin, C. Quanqi, and Z. Yujin, “Deep learning and its new progress in object and behavior recognition,” *Journal of image and graphics*, vol. 19, no. 2, pp. 175–184, 2014.

[19] F. Feng, R. H. Chan, X. Shi, Y. Zhang, and Q. She, “Challenges in task incremental learning for assistive robotics,” *IEEE Access*, vol. 8, pp. 3434–3441, 2019.

[20] S. H. Kasaei, M. Oliveira, G. H. Lim, L. S. Lopes, and A. M. Tomé, “Towards lifelong assistive robotics: A tight coupling between object perception and manipulation,” *Neurocomputing*, vol. 291, pp. 151–166, 2018.

[21] K. K. Patel, A. Kar, S. Jha, and M. Khan, “Machine vision system: a tool for quality inspection of food and agricultural products,” *Journal of food science and technology*, vol. 49, no. 2, pp. 123–141, 2012.

[22] O. Cosido, A. Iglesias, A. Galvez, R. Catuogno, M. Campi, L. Terán, and E. Sainz, “Hybridization of convergent photogrammetry, computer vision, and artificial intelligence for digital documentation of cultural heritage-a case study: the magdalena palace,” in *2014 International Conference on Cyberworlds*. IEEE, 2014, pp. 369–376.

[23] D. Shen, G. Wu, and H.-I. Suk, “Deep learning in medical image analysis,” *Annual review of biomedical engineering*, vol. 19, pp. 221–248, 2017.

[24] L. Deng, “The mnist database of handwritten digit images for machine learning research [best of the web],” *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 141–142, 2012.

[25] A. R. Pathak, M. Pandey, and S. Rautaray, “Application of deep learning for object detection,” *Procedia computer science*, vol. 132, pp. 1706–1717, 2018.

[26] A. Rosenfeld, “Picture processing by computer,” *ACM Comput. Surv.*, vol. 1, no. 3, p. 147–176, Sep. 1969. [Online]. Available: https://doi.org/10.1145/356551.356554

[27] M. Everingham, S. A. Eslami, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, “The pascal visual object classes challenge: A retrospective,” *International journal of computer vision*, vol. 111, no. 1, pp. 98–136, 2015.

[28] B. Hariharan, P. Arbeláez, L. Bourdev, S. Maji, and J. Malik, “Semantic contours from inverse detectors,” in *2011 International Conference on Computer Vision*. IEEE, 2011, pp. 991–998.

[29] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, “Microsoft coco: Common objects in context,” in *Computer Vision – ECCV 2014*, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds. Cham: Springer International Publishing, 2014, pp. 740–755.

[30] G. Ros, L. Sellart, J. Materzynska, D. Vazquez, and A. M. Lopez, “The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 3234–3243.

[31] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, “Vision meets robotics: The kitti dataset,” *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, 2013.

[32] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus, “Indoor segmentation and support inference from rgbd images,” in *European conference on computer vision*. Springer, 2012, pp. 746–760.

[33] J. Xiao, A. Owens, and A. Torralba, “Sun3d: A database of big spaces reconstructed using sfm and object labels,” in *Proceedings of the IEEE international conference on computer vision*, 2013, pp. 1625–1632.

[34] S. Song, S. P. Lichtenberg, and J. Xiao, “Sun rgb-d: A rgb-d scene understanding benchmark suite,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 567–576.

[35] K. Lai, L. Bo, X. Ren, and D. Fox, “A large-scale hierarchical multi-view rgb-d object dataset,” in *2011 IEEE international conference on robotics and automation*. IEEE, 2011, pp. 1817–1824.

[36] I. Armeni, S. Sax, A. R. Zamir, and S. Savarese, “Joint 2d-3d-semantic data for indoor scene understanding,” *arXiv preprint arXiv:1702.01105*, 2017.

[37] S. W. Brose, D. J. Weber, B. A. Salatin, G. G. Grindle, H. Wang, J. J. Vazquez, and R. A. Cooper, “The role of assistive robotics in the lives of persons with disability,” *American Journal of Physical Medicine & Rehabilitation*, vol. 89, no. 6, pp. 509–521, 2010.

[38] R. E. Kalman, “A new approach to linear filtering and prediction problems,” 1960.

[39] B. Ho and R. E. Kálmán, “Effective construction of linear state-variable models from input/output functions,” *at-Automatisierungstechnik*, vol. 14, no. 1-12, pp. 545–548, 1966.

[40] D. O. Hebb, *The organization of behavior: A neuropsychological theory*. Psychology Press, 2005.

[41] F. Rosenblatt, “Perceptron simulation experiments,” *Proceedings of the IRE*, vol. 48, no. 3, pp. 301–309, 1960.

[42] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, “Learning representations by back-propagating errors,” *nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[43] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[44] C. J. Watkins and P. Dayan, “Q-learning,” *Machine learning*, vol. 8, no. 3-4, pp. 279–292, 1992.

[45] M. A. Hearst, S. T. Dumais, E. Osuna, J. Platt, and B. Scholkopf, “Support vector machines,” *IEEE Intelligent Systems and their applications*, vol. 13, no. 4, pp. 18–28, 1998.

[46] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[47] M. Riesenhuber and T. Poggio, “Hierarchical models of object recognition in cortex,” *Nature neuroscience*, vol. 2, no. 11, pp. 1019–1025, 1999.

[48] S. G. Mallat, “A theory for multiresolution signal decomposition: the wavelet representation,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 11, no. 7, pp. 674–693, 1989.

[49] T. Poggio and S. Smale, “The mathematics of learning: Dealing with data,” *Notices of the AMS*, vol. 50, no. 5, pp. 537–544, 2003.

[50] T. Hofmann, B. Schölkopf, and A. J. Smola, “Kernel methods in machine learning,” *The annals of statistics*, pp. 1171–1220, 2008.

[51] B. A. Olshausen and D. J. Field, “Sparse coding with an overcomplete basis set: A strategy employed by v1?” *Vision research*, vol. 37, no. 23, pp. 3311–3325, 1997.

[52] A. Ng *et al.*, “Sparse autoencoder,” *CS294A Lecture notes*, vol. 72, no. 2011, pp. 1–19, 2011.

[53] M. Ranzato, C. Poultney, S. Chopra, and Y. L. Cun, “Efficient learning of sparse representations with an energy-based model,” in *Advances in neural information processing systems*, 2007, pp. 1137–1144.

[54] J. Sharma, A. Angelucci, and M. Sur, “Induction of visual orientation modules in auditory cortex,” *Nature*, vol. 404, no. 6780, pp. 841–847, 2000.

[55] L. Von Melchner, S. L. Pallas, and M. Sur, “Visual behaviour mediated by retinal projections directed to the auditory pathway,” *Nature*, vol. 404, no. 6780, pp. 871–876, 2000.

[56] L. Thaler, S. R. Arnott, and M. A. Goodale, “Neural correlates of natural human echolocation in early and late blind echolocation experts,” *PLoS one*, vol. 6, no. 5, p. e20162, 2011.

[57] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural computation*, vol. 18, no. 7, pp. 1527–1554, 2006.

[58] H. Lee, R. Grosse, R. Ranganath, and A. Y. Ng, "Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations," in *Proceedings of the 26th annual international conference on machine learning*, 2009, pp. 609–616.

[59] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014, pp. 2672–2680.

[60] D. H. Hubel and T. N. Wiesel, "Receptive fields, binocular interaction and functional architecture in the cat's visual cortex," *The Journal of physiology*, vol. 160, no. 1, pp. 106–154, 1962.

[61] L. Bottou, "Stochastic gradient descent tricks," in *Neural networks: Tricks of the trade*. Springer, 2012, pp. 421–436.

[62] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

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

[64] K. Wada, "labelme: Image Polygonal Annotation with Python," https://github.com/wkentaro/labelme, 2016.