DEGREE THESIS

Grasp Prediction with Convolutional Neural Networks

AUTHOR: Francesc Lluís Salvadó

ADVISORS: Deniz Erdogmus
           Fernando Quivira
           Xavier Giró

UNIVERSITAT POLITÈCNICA DE CATALUNYA
Escola Tècnica Superior d’Enginyeria de Telecomunicació de Barcelona

and

NORTHEASTERN UNIVERSITY
Department of Electrical and Computer Engineering

BOSTON, AUGUST 2017
Abstract

This thesis explores the application of a deep learning computer vision approach for grasp classification in order to improve hand prosthesis control. First, a criteria based on prehensile human hand is adopted and object images from different datasets are labeled to a type of grasp. Then, two different convolutional neural networks architectures using weights from pretrained models are designed in order to deal with single or fusion streams of information. Next, architectures are trained with color and encoded depth images. Finally, offline and online results for both architectures and type of data are reported. Results show that it is possible to classify objects into grasps without recognizing them or having any knowledge about their dimensions.
Acknowledgements

I would like to thank Deniz Erdogmus, Dana Brooks, Fernando Quivira and Xavier Giró for their guidance, help, sharing their passion for research and the opportunity to join the B-SPIRAL group to realize my project.

I would also like to thank the other UPC students doing his bachelor thesis in Boston and my lab partners Biel, Jaume, and Paula to make me feel at home.

Finally, and most important, I would like to thank immensely my family for the support I've always received. Specially to my mother: "Per creure en mi més que jo mateix, pel teu exemple, i per ensenyar-me a ser feliç. Per molt lluny que estiguis et tinc ben present; aquesta tesis és per tu."
### Revision history and approval record

<table>
<thead>
<tr>
<th>Revision</th>
<th>Date</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>15/07/2017</td>
<td>Document creation</td>
</tr>
<tr>
<td>1</td>
<td>31/07/2017</td>
<td>Document revision</td>
</tr>
<tr>
<td>2</td>
<td>02/08/2017</td>
<td>Document approbation</td>
</tr>
</tbody>
</table>

### DOCUMENT DISTRIBUTION LIST

<table>
<thead>
<tr>
<th>Name</th>
<th>e-mail</th>
</tr>
</thead>
<tbody>
<tr>
<td>Francesc Lluís Salvadó</td>
<td><a href="mailto:francesc.luis@gmail.com">francesc.luis@gmail.com</a></td>
</tr>
<tr>
<td>Deniz Erdogmus</td>
<td><a href="mailto:erdogmus@ece.neu.edu">erdogmus@ece.neu.edu</a></td>
</tr>
<tr>
<td>Dana Brooks</td>
<td><a href="mailto:brooks@ece.neu.edu">brooks@ece.neu.edu</a></td>
</tr>
<tr>
<td>Fernando Quivira</td>
<td><a href="mailto:quivira@ece.neu.edu">quivira@ece.neu.edu</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Written by:</th>
<th>Reviewed and approved by:</th>
<th>Reviewed and approved by:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Date</td>
<td>Date</td>
<td>Date</td>
</tr>
<tr>
<td>15/07/2017</td>
<td>02/08/2017</td>
<td>02/08/2017</td>
</tr>
<tr>
<td>Name</td>
<td>Name</td>
<td>Name</td>
</tr>
<tr>
<td>Francesc Lluís Salvadó</td>
<td>Deniz Erdogmus</td>
<td>Fernando Quivira</td>
</tr>
<tr>
<td>Position</td>
<td>Position</td>
<td>Position</td>
</tr>
<tr>
<td>Project Author</td>
<td>Project Supervisor</td>
<td>Project Co-Supervisor</td>
</tr>
</tbody>
</table>
1 Introduction

1.1 Problem Statement

1.2 Previous Work

1.3 Objectives

1.4 Requirements and specifications

1.4.1 Framework used

1.4.2 GPU power

1.5 Work Plan

1.5.1 Work Packages

1.5.2 Gantt diagram

1.5.3 Deviations and Incidents

1.6 Thesis Outline

2 Background Theory

2.1 Neural networks

2.2 Convolutional Neural Network

3 Methodology

3.1 Introduction

3.2 Probabilistic Framework

3.3 Datasets

3.3.1 RGB-D Object Dataset

3.3.2 Amsterdam Library of Object Images

3.3.3 Multi-sensor 3D Object Dataset

3.3.4 BigBIRD

3.4 Architectures
List of Figures

1.1 Gantt diagram .................................................. 12

2.1 Example of a neural network. Each circular node represents an artificial neuron and an arrow represents a connection from the output of one neuron to the input of another. .................................................. 14

2.2 Artificial neuron model ........................................... 15

2.3 LeNet5 architecture ................................................ 16

3.1 Jet colormap ....................................................... 23

3.2 Depth preprocessing example: a) original depth image, b) preprocessed image .................................................. 23

3.3 First seven feature maps for several example images after the third convolutional block of VGG16: a) RGB lateral pinch, b) Depth lateral pinch, c) RGB medium wrap, d) Depth medium wrap, e) RGB thumb two finger, f) Depth thumb two finger, g) RGB tripod, d) Depth tripod .................................................. 24

4.1 Single Stream Seen Objects Confusion Matrix: a) RGB images, b) Depth images .................................................. 27

4.2 Single Stream Uneen Objects Confusion Matrix: a) RGB images, b) Depth images .................................................. 28

4.3 Examples of True and False predictions done by single stream architecture on unseen RGB objects .................................................. 28

4.4 Fusion Confusion Matrix: a) Seen Objects, b) Unseen Objects .................................................. 29
List of Tables

3.1 Grasp Type - Object Shape characteristics ........................................ 19
4.1 Single Stream Seen Objects Results .................................................. 27
4.2 Single Stream Unseen Objects Results ............................................. 28
4.3 Fusion Results .............................................................................. 29
5.1 Budget of the project .................................................................... 30
Chapter 1

Introduction

1.1 Problem Statement

We are not enough aware about the awesome complex pieces of machinery which our body contains, like it could be a hand. Only the ones without his limb, know the range of abilities they were provided: from texting a message to paint a picture.

According to recent statistics, in EEUU alone, there are more or less half million of upper-limb amputees and by 2050 the number of cases will be doubled \[1\]. Furthermore, upper extremity amputations are most frequently indicated by severe traumatic injuries which mainly affect people within working ages. Apart the expensiveness of life-time care for this group, Jang et al. \[2\] reported that many of the amputees had difficulties in complex tasks and either changed jobs or became unemployed. For that reason, prosthetic hands can provide users a better quality of life for a long period. While the challenges are great, remarkable progress has been made over the past few decades with the desire to make the prosthesis as close as possible to the real thing.

There are different kinds of prosthetic limbs depending on their functioning: active prostheses, which include body-powered and externally powered, and passive prostheses, which include cosmetic and functional.

Body-powered prostheses are controlled with the power of an individual’s residual limb, pectoral girdle, and upper-body muscles. The energy required by the user is one of the main drawbacks of this kind of prostheses. For this reason, externally powered prosthesis use motors to move their components and batteries to power them. Further, they are divided into myoelectric (controlled by electromyography signals) and electric (controlled by external buttons).

Within the passive prostheses, a cosmetic prosthetic limb, known as cosmesis, is designed to simulate the appearance of the missing body part. Advanced plastics and pigments uniquely matched to the patient’s own skin tone enable a cosmesis to take on an awesome life-like appearance, sometimes indistinguishable from the original. The functional ones, have the purposes of facilitate the execution of different activities.

Noticing that different kinds of prosthesis are combined to achieve a better user satisfaction, several researches about the relationship between users’ satisfaction and prosthesis characteristics have been carried out. The user satisfaction topics that most concern the following work are related with grasping, activities of daily living desired by users, and the worries of myoelectric prosthesis users.

Kyber and Hill \[3\] said that subjects want the prosthesis to adapt the grasping configuration to the object shape. Biddiss et al. \[4\] noted that myoelectric users pay more attention to function and comfort. For example, they require improvements in movement and grip functions as the ability to move fingers independently. Cordella et al. \[5\] specifies in the discussion section that performing activities of daily living like handling buttons or tying shoelaces is desired by prosthetic users. Therefore, a prosthetic system is required to perform basic grasping actions,
perform stable grasp, and be accurate when handling objects. Finally, they conclude that one of the main engineering challenges in the development of prosthetic devices is to improve prosthesis control that notably affects functionality.

1.2 Previous Work

It can be observed that visual input for object grasping has been widely used in the robotics or biomedicine community.

In robotics area, images captured by camera are extremely important because they represent one of the main sources of information when a robot is operating in uncontrolled environments. Usually, the task is divided to make it easier. As Zaharescu report in [6], the lines of research that have been individually attempted are: algorithms that deal with the object directly (thus producing grasp points), methods used to separate and track the object from the background, outline the issues encountered when dealing with multiple coordinate systems, and find robot path planning methods.

In the same field, others have studied the human hand-object relationship to achieve better hand design and robot action planning [7]. They show that humans use the similar grasp types for certain types of objects. Moreover, they report that the shape of the object has a large influence on the applied grasp.

In biomedicine subject, focusing on myoelectric hand prostheses, the main research line is based on myoelectric pattern recognition [8]. In latest years, computer vision techniques have been introduced to achieve better accuracy when decoding user’s movements. Dosen et al [29, 30] and Markovic et al [9] shown that appropriate grasp types can be determined using shape features.

Ghazal et al [10] used a deep learning-based artificial vision system which classifies objects with regards to the grasp pattern without explicitly identifying them or measuring their dimensions. They use a two-layer CNN architecture reporting that it generalizes well with novel objects. They classify four different grasp types achieving an accuracy of 85% of the seen and 75% for novel objects.

DeGol et al [11] showed that automatic grasp selection can be achieved by placing a camera in the palm of a prosthetic hand. Their approach is to train a VGG-VeryDeep-16 architecture with DeepGrasping and ImageNet dataset achieving a 93.2% of accuracy.

1.3 Objectives

This work wants to improve prosthesis control that affects functionality presenting a novel grasp recommendation system based on computer vision techniques which merges depth and RGB information captured by cameras.

It aims to prove that a user-driven recommendation system is possible. That means to allow different behavior of grasp recommendations depending on the user role and, furthermore, it wish
to be easy trainable for other user needs and to provide grasp scalability if requested.

Aware of the recent advances in the deep learning field, this research uses Convolutional Neural Networks because of his excellent performance in some tasks such as object classification or segmentation. In this case, the purpose is to build models with them relating object and grasp.

The hypothesis of this work states that it is possible to classify objects into grasps without recognizing them or having any knowledge about their dimensions. In addition, it is believed that it can be achieved using pretrained networks allowing to train the whole system only using a small set of RGB and depth images.

1.4 Requirements and specifications

This research has been developed in the Biomedical Imaging and Signal Processing Laboratory at Northeastern University. The whole project consists in designing and building a prosthetic hand. I was asked to design a grasp recommendation system using context information. No previous work was done by any lab partner.

The resultant prosthetic hand will be controlled mainly using three types of signals: The EEG and the EMG, which are provided by the user physiological, and the context signals, which are not. The whole controller software is thought to be composed by modules. Using this approach, the system is capable to weigh the decisions given by each module.

Context signals are provided by the environment. In this case, the context information corresponds to signals captured by cameras which can be located on the wrist and/or mounted at the head of the user.

The requirement is to provide a system which produce accurate grasp recommendations to the whole system each short period.

1.4.1 Framework used

The main programming language chosen to develop this work has been Python [12]. Keras [13], a high-level neural networks API written in Python and capable of running on top of either TensorFlow, CNTK or Theano, has served as a framework to design the different network architectures. Also, Matlab [14] and ImageMagick [15] have been slightly used for data analysis and image preprocessing respectively.

1.4.2 GPU power

The GPU power supply has been provided by the Image Processing Group (GPI) from UPC. In addition to Keras, CUDA libraries [16] for parallel computing and programming model developed by NVIDIA have been required to utilize the GPUs (Graphics Processing Unit). NVidia GeForce GTX TITAN X [17] GPUs have been used to train and evaluate the implementation.
1.5 Work Plan

The work plan is split into several work packages detailed in this section. The modifications and deviations occurred are described in Section 1.5.3.

1.5.1 Work Packages

• WP 1: Project documentation
  – Project proposal
  – State-of-the-art research
  – Project approval

• WP 2: Introduction to Deep Learning
  – Learning Python
  – Deep Learning and CNN
  – Learning Keras

• WP 3: Dataset Analysis
  – Research RGBD Datasets
  – Prepare datasets
  – Find a probabilistic framework
  – Label datasets
  – Learning to deal with large datasets

• WP 4: Experimentation
  – Find depth preprocessing
  – Learn about different architectures
  – Test different designs
  – Test for seen objects
  – Software improvements
  – Test for unseen objects

• WP 5: Documentation delivery
  – Documentation delivery
  – Project presentation

1.5.2 Gantt diagram

The Gantt diagram of Figure 1.1 illustrates the work breakdown structure of the project.
1.5.3 Deviations and Incidents

The only incidence the project suffered was related to the GPU power supply by the hosting university. Despite the availability of GPUs, there was a problem using the GpuArray Backend. It was solved using the old Theano GPU backend available at computation service from the Image Processing Group at UPC. This way, while the project was developed in Boston, the actual computation was run in Barcelona.

1.6 Thesis Outline

Starting with the introduction, in the Chapter 1 the motivation of the thesis is described, along with a selection of literature of the state of the art in the research field. Then, the objectives are presented and a brief overview of the software used is given. Finally, is provided how the project has been developed in terms of organization. Chapter 2 provides a basic background of deep learning and some technical concepts behind convolutional neural networks required to understand why they work and the reason they can be useful in the current approach.

The core of the practical part of this thesis is comprised between the third and fourth chapters. First in chapter 3 it is reported the probabilistic framework adopted, an overview of the datasets
used, and the information related with the training of the different network architectures designed.
In chapter 4, the metrics evaluated are explained and the results obtained for each experiment
are presented and analyzed.

To finalize the report, chapter 5 notes the main costs of the project and chapter 6 states the
contributions done by this research and some conclusions which lead to future work.
Chapter 2

Background Theory

Following, it is presented a briefly explanation of some technical concepts behind Neural Networks and Convolutional Neural Networks useful to understand why they work and why they can be useful in the current approach.

The reader is referred to chapter 5 of *Pattern recognition and machine learning* [18] by Bishop and *CS231n: Convolutional Neural Networks for Visual Recognition* course [19] by Stanford University for a more detailed explanation.

2.1 Neural networks

Neural Networks were first proposed by Warren McCullough and Walter Pitts in 1944 with a paper entitled "A Logical Calculus of Ideas Immanent in Nervous Activity" [20]. It was an attempt to find a mathematical representations of information processing in biological systems. Since then, and after a lot of research, neural networks have become the technique utilized by the best-performing artificial-intelligence systems using an approach named deep learning.

A neural network can have from a few dozen to millions artificial neurons arranged in a series of layers classified as input, hidden, and output. The input layer’s function is to receive information from the outside and process them. The hidden layers main job is to transform inputs into something that output layer can use. Finally, output layer contains signals that represent how network responds to the information it has learned.

![Figure 2.1: Example of a neural network. Each circular node represents an artificial neuron and an arrow represents a connection from the output of one neuron to the input of another.](image)

The artificial neuron, which is the basis of a neural network, is inspired on a biological neuron. It contains input variables \(x_i\) that represents external stimulus or outputs from other neurons. After adding a bias term \(b\), the mentioned inputs are multiplied by weights \(w_i\) which are understood to perform as synapses. Then, \((x_iw_i)\) are passed through an aggregation function
and a nonlinear activation function \( h() \) which is presumed to be the cellular body. Finally, it yield an output variable \( (y) \) assumed as the axon.

![Artificial neuron model](image)

**Figure 2.2: Artificial neuron model**

The mathematical expression of the model takes the form

\[
y = h\left(\sum_{i=0}^{n} w_i x_i\right)
\]  

(2.1)

Note that in equation (2.1) the bias parameter \( (b) \) is absorbed into the set of weight parameters by defining an additional input variable \( x_0 \) whose value is clamped at \( x_0 = 1 \).

To make a neural network useful for a specific task, it has to be trained which means that weights from layers are transformed until network’s outputs are close enough to desired outputs. Initially, all the weights are set to random values. Then, the input layer is fed with training data and it passes through all layers. After being multiplied and added in complex ways it arrives transformed at the output layer and the difference between the target outputs and the actual outputs is calculated. During the learning process, this difference is back-propagated to the previous layer and the weights are normally adjusted using the Delta Rule. It finishes when the initial layer is reached.

\[
\Delta w_{ij} = \alpha (t_j - y_j) h'(z_j) x_i
\]  

(2.2)

Equation (2.2) shows the delta rule for weight \( w_{ji} \) of a neuron \( j \) with activation function \( h(x) \). Where \( \alpha \) is the learning rate, \( h(x) \) is the neuron’s activation function, \( t_j \) is the target output, \( z_j = \sum x_i w_{ji} \) is the weighted sum of the neuron’s inputs, \( y_j = h(z_j) \) the actual output, and \( x_i \) is the \( i^{th} \) input.

Finally, although a substantial computational resources have to be invested and, furthermore, the function to minimize when training is not a convex function of the model parameters, the reason why in this of work has been used neural networks is because the resulting model is significantly more compact and faster to evaluate with new data than the others existing.
Convolutional Neural Network

Convolutional Neural Networks (CNN) have been extensively applied to image data due to the invariance properties into his neural network structure. Furthermore, this approach takes into account that the nearby pixels in an image are more correlated than distant pixels which allows to extract local features of image subregions. At the top of the structure, this information can be merged in order to provide information about the image as whole.

Convolutional Neural Networks want to emulate vision processing in living organisms. Hubel and Wiesel in [21] studied the neural basis of visual perception concluding that monkeys visual cortices contain neurons that individually respond to small regions of the visual field. In addition, eye’s receptive fields location and size varies depending on the cortex region but being overlapped and similar in neighboring cells.

One of the first Convolutional Neural Networks was LeNet5. It was designed by Yann LeCun [22] and served to several banks to recognize hand-written numbers on checks digitized in 32x32 pixel images.

![LeNet5 architecture](image)

In recent years, CNN’s have became powerful and popular algorithms because of his effectiveness in image recognition, classification, etc. and the rising of computing power and data available.

CNN’s and ordinary Neural Networks are really similar taking into account they are built with neurons which have trainable weights and biases. Moreover, they use the same training mechanism. However, CNN’s are thought to have images as inputs allowing them to reduce the parameters of the network and be more efficient to implement.

As other Neural Networks, CNN consists of an input and an output layer, as well as multiple hidden layers. The main types of layers to build a CNN architecture are:

- Input layer: take the raw pixel values of the image.
- Convolutional layer: Hidden layer which compute the output of neurons that are connected to local regions in the input.
- Pooling Layer: Hidden layer that perform a downsampling operation along the spatial dimensions.
- Non linearity layer: Hidden layer which apply an elementwise activation function.
- Fully-Connected Layer: Output layer compute the class scores.
Finally, by combining these layers in a proper way, the resultant CNN is capable to transform the input image into class scores.
Chapter 3

Methodology

3.1 Introduction

In order to study the impact and tradeoffs of creating a grasp recommendation system using context information, the following steps are addressed. First, it is reviewed a probabilistic framework which relates object shape with grasp types. Then, several datasets are chosen and images are labeled using the rules learned from the probabilistic framework. Finally, various network architectures with different type of data are tested. With this procedure, it is hoped to achieve relevant accuracy in order to help the whole performance of the prosthetic hand together with the EEG and EMG information.

3.2 Probabilistic Framework

The review of the research done by Ian et al. [23] [24] and Feix et al. [25] have serve as the support to create the probabilistic framework. This is an important part, if not the most, of the success of the final result. It represents the relationship we want the network to learn between object shape and type of grasp. In addition, knowing the existence of infinite grasp types, it allows to choose correctly a set of canonical hand poses, which can be used to handle a wide range of objects.

They studied the prehensile human hand use during daily work activities of four subjects: two housekeepers and two machinists. It is concluded that given the fact that many grasps can be used for effectively the same purpose, there are different smaller subset of grasp types than can be used to grasp the most objects. In addition, it is noted that humans use similar grasp types for certain types of objects and thus, enables to predict a large percentage of grasps based on the object shape data alone.

In table 3.1 there is a summary of the top seven grasps used by subjects and his specific characteristics on object shape and his use when picking up an object.

It is important to mention that the top ten grasps are used the 81 percent of time for all subjects. For the housekeepers, the first five grasp already provide 80 percent of the grasping duration while for the machinists 10 are required to reach a similar level.

This information allow to convincingly decide the best set of grasps that network should learn. Remembering that the principal concern of using context information is to have prior knowledge about which is the best grasp to handle an object before grasping it, only the grasps that are frequently used to pick up objects have been considered. These are: medium wrap, lateral pinch, tripod, and thumb two finger.
<table>
<thead>
<tr>
<th>Grasp Type</th>
<th>Object Characteristics</th>
<th>Pick Up</th>
</tr>
</thead>
<tbody>
<tr>
<td>medium wrap</td>
<td>Larger cylindrical objects</td>
<td>Yes</td>
</tr>
<tr>
<td>precision disk</td>
<td>Objects in wiping actions and some small handles</td>
<td>Rarely</td>
</tr>
<tr>
<td>lateral pinch</td>
<td>Smaller knobs, flat objects such as keys and cords</td>
<td>Yes</td>
</tr>
<tr>
<td>tripod</td>
<td>Spherical parts and small knobs</td>
<td>Yes</td>
</tr>
<tr>
<td>lateral tripod</td>
<td>Small parts and tools with a small cylindrical shape</td>
<td>Rarely</td>
</tr>
<tr>
<td>power sphere</td>
<td>To be used with many other soft, compliant objects</td>
<td>Quite</td>
</tr>
<tr>
<td>thumb-2 finger</td>
<td>Small parts and tools, especially cylindrical</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Table 3.1: Grasp Type - Object Shape characteristics
3.3 Datasets

Four datasets have been used during the development of the grasp classification system: RGB-D Object Dataset [26], Amsterdam Library of Object Images [27], BigBIRD [28], and Multi-sensor 3D Object Dataset [29].

The reason why different datasets are chosen is because it is wanted to have the same number of images for each type of grasp. With this approach, the resultant dataset for training and testing is not biased which is preferred during the evaluation process. In addition, many of the objects contained in the chosen datasets correspond to medium wrap label while there are a few corresponding to lateral pinch. Finally, trying to avoid the presence of a bias in each image data collection, the objects have been cropped with the will of getting the image that a camera in the wrist would capture during the user’s object grasping intention.

3.3.1 RGB-D Object Dataset

The RGB-D Object Dataset [26] is a large dataset of 300 common household objects. The objects are organized into 51 categories arranged using WordNet hypernym-hyponym relationships (similar to ImageNet). This dataset was recorded using a Kinect style 3D camera that records synchronized and aligned 640x480 RGB and depth images at 30 Hz. Each object was placed on a turntable and video sequences were captured for one whole rotation.

Unlike many existing datasets, such as Caltech 101 and ImageNet, objects in this dataset are organized into both categories and instances. In these datasets, the class dog contains images from many different dogs and there is no way to tell whether two images contain the same dog, while in the RGB-D Object Dataset the category soda can is divided into physically unique instances like Pepsi Can and Mountain Dew Can.

3.3.2 Amsterdam Library of Object Images

ALOI DB [27] is a color image collection of one-thousand small objects recorded for scientific purposes. In order to capture the sensory variation in object recordings, viewing angle is systematically varied. Regarding the object viewpoint, the frontal camera was used to record 72 aspects of the objects by rotating the object in the plane at 5 degrees.

3.3.3 Multi-sensor 3D Object Dataset

Multi-sensor 3D Object Dataset [29] is a dataset for 3D object recognition using the new high-resolution Kinect V2 sensor and some other traditional low cost devices like PrimeSense Carmine containing 30 objects. Each object is captured by the camera 64 times in a full 360° turn of the platform. For each capture, the direct processing of the Kinect V2 data streams provides us three different information sources: a 1920x1080 RGB color image, a 512x424 depth map and a 512x424 infrared image.
3.3.4 BigBIRD

BigBIRD [28], is a dataset composed of, for each object, 600 3D point clouds and 600 high-resolution (12 MP) images spanning all views. The 600 images are obtained by taking shots from 5 polar angles and 120 azimuthal angles, the latter equally spaced by 3.

3.4 Architectures

Two different architectures are proposed for automatic grasp selection. Both use the well-known VGG16 [30] as basis.

3.4.1 VGG16

Convolutional neural networks have recently been shown to be remarkably successful for recognition on RGB images. In this case, VGG16, named after the Visual Geometry Group from Oxford who developed it, has won the ILSVR (ImageNet) competition in 2014 and to this day is still considered to be an excellent vision model.

The original input to the ConvNet is a 224x224 RGB image but in this case it has been resized to 150x150. Then, the image is passed through a stack of convolutional layers, where a filter with a very small receptive field of 3x3 is used. This is considered to be the smallest size to capture the notion of left/right, up/down, and center. The convolution stride is fixed to 1 pixel; the spatial padding of convolutional layer input is such that the spatial resolution is preserved after convolution. Spatial pooling is carried out by five max-pooling layers, which follow some of the convolutional layers. Max-pooling is performed over a 2x2 pixel window, with stride 2. Finally, the stack of convolutional layers is followed by three fully-connected (FC) layers.

3.4.2 Single stream architecture for grasp classification

The single stream architecture is thought to deal with one type of data at a time, like it can be RGB or depth. In this case, the VGG16 last fully-connected layers are modified to perform the four grasp type classification. The design of single stream architecture can be found at appendix A.

3.4.3 Fusion architecture for grasp classification

The fusion architecture is thought to deal with two types of data at a time, RGB and depth. In this case, for each data stream, the first three convolutional blocks of the VGG16 architecture are maintained. Then, the outputs are merged and it serves as input for two more convolutional blocks. Finally, the stack of convolutional layers is followed by three fully-connected layers. The design of fusion architecture can be found at appendix B.
3.5 Training samples

A total of 2000 images, 500 for each class, corresponding to 120 different objects from The RGB-D Object Dataset, have served as training data for the Single Stream Architecture. For the Fusion Architecture, a total of 4000 images, 1000 per class, corresponding to 120 different objects from The RGB-D Object Dataset are used. During labeling, the criteria adopted has been to associate object images with the type of grasp that follows the characteristics shown in table [3.1]. It is important to mention that objects which don’t have the characteristics shown in the table are not taken into consideration. In order to avoid choosing similar images, samples corresponding to the same category have been randomly selected.

3.6 Preprocessing

3.6.1 RGB images

Some basic preprocessing is applied to RGB images. First, image normalization is applied forcing pixel values to lie between 0 and 1. Finally, images are resized to 150x150 pixels in order to satisfy network requirements.

3.6.2 Depth images

The preprocessing technique applied to depth images regards the one proposed by Eitel et al. in [31] defined as ‘colorizing’ depth. With this approach, depth information is encoded in a way that is compatible with the RGB encoding. Thus, current CNN models developed using RGB images can be trained with depth images. This preprocessing has been preferred among others, like the HHA encoding proposed by Gupta et al. [32], because of his non-complexity, which also means fastness, and his good results.

The first thing is to convert depth image pixel values into 0 to 255 range which creates another image \( f \). Then, \( p \) is defined as the normalized histogram of \( f \) with a bin for each possible intensity.

\[
p_n = \frac{\text{number of pixels with intensity } n}{\text{total number of pixels}} \quad n = 0, 1, ..., 255
\]

After, pixel intensities, \( k \), of \( f \) are transformed by the function:

\[
T(k) = \text{floor}(255 \sum_{n=0}^{k} p_n)
\]

where \( \text{floor()} \) rounds down to the nearest integer.

Next, a jet colormap is applied to \( f \) transformed image converting it to three channel. Near points are colorized with blue while far points with red.
Finally, images are normalized and resized to 150x150 pixels to satisfy network requirements.

![Figure 3.2: Depth preprocessing example: a) original depth image, b) preprocessed image](image)

3.7 Training method

Before entering to comment the training method for each architecture, an important point is that the final value of all hyper-parameters (Batch size, Epoch, Optimizer, ...) has been selected in order to get the best performance of the network. To evaluate the performance, it has been considered the loss function applied to a validation set containing 20% of the total number of training samples.

### 3.7.1 Training method for single stream architecture

Since it is used very little training examples, just a few hundred pictures from each class, the convolutional part of the VGG16 architecture is instantiated with pre-trained weights on the ImageNet [33] dataset. With this approach, it is wanted that the learned features will be relevant to our classification problem, and likely to generalize well although there are not any grasp types classes on ImageNet. Then this model is run on the training and validation data once, recording the output. Then it it is trained the small fully-connected model on top of the stored features. With this method, the classifier is trained in a computational efficiency way.

The small fully-connected model is trained minimizing the multiclass cross-entropy of the training data.

\[
\min_{W,\theta} - \sum_{i=1}^{N} y_i \log(\sigma(Wg(x^i; \theta)))
\]  

(3.2)

Where \(\theta\) are the parameters of the network, \(g(x^i; \theta)\) is the output from the last layer for image \(x^i\), \(W\) are the weights applied in the softmax layer, \(\sigma\) represents the softmax function, \(y_i\) is the \(x^i\) image label in one-hot encoding, and \(N\) represents the total number of training images.
3.7.2 Training method for fusion architecture

As with Single Stream Architecture, the first three convolutional blocks from VGG16 architecture are instantiated with pre-trained weights on the ImageNet dataset. This weights will be keep intact during the training of the whole network. The remaining convolutional blocks and the fully connected layers are trained minimizing the multiclass cross-entropy of the training data analogous to Equation 3.2

$$\min_{W,θ} - \sum_{i=1}^{N} y_i \log(\sigma(W g(x^{rgb}, x^d, θ)))$$ (3.3)

Where $\theta$ are the parameters of the network, $g(x^{rgb}, x^d; θ)$ is the output from the last layer for images $x^{rgb}$ and $x^d$, $W$ are the weights applied in the softmax layer, $\sigma$ represents the softmax function, $y_i$ is the $x^{rgb}$ and $x^d$ images label in one-hot encoding, and $N$ represents the total number of training images.
Chapter 4

Results

This chapter explains the experiments that have been carried out and the metrics used to evaluate the performance. It also explains how the test data has been gathered and presents the results for each experiment.

4.1 Experimental setup

The following experiments have been performed:

- Experiment 1. Train the single stream architecture with RGB images corresponding to seen objects. Obtain the predictions done by the model for seen objects and unseen objects.
- Experiment 2. Train the single stream architecture with depth images corresponding to seen objects. Obtain the predictions done by the model for seen objects and unseen objects.
- Experiment 3. Train the fusion architecture with RGB and depth images corresponding to seen objects. Obtain the predictions done by the model for seen objects and unseen objects.

4.2 Metrics

The evaluation of the proposed architectures has been based on the metrics reported by Sokolova and Lapalme in [34]. Specifically, has been used the ones referring to multi-class classification tasks in which the input is to be classified into one, and only one, of \( l \) non-overlapping classes.

For an individual class \( C_i \), the assessment is defined by: \( t_{pi} \), true positive; \( f_{ni} \), false negative; \( t_{ni} \), true negative; \( f_{pi} \), false positive; \( \text{Accuracy}_i \); \( \text{Precision}_i \); \( \text{Recall}_i \) all associated to \( C_i \).

As there is the same number of test samples for each class, all classes have been equally treated using macro-averaging. Final metrics are the following:

- **Error rate**, which focus on the average per-class classification error, takes the form

\[
\sum_{i=1}^{l} \frac{t_{pi} + f_{ni}}{t_{pi} + f_{ni} + f_{pi} + t_{ni}}
\]  

(4.1)

- **Precision**, which focus on an average per-class agreement of the data class labels with those of a classifiers, is given by

\[
\sum_{i=1}^{l} \frac{t_{pi}}{t_{pi} + f_{pi}}
\]  

(4.2)
• **Recall**, which focus on an average per-class effectiveness of a classifier to identify class labels, takes the form

\[
\sum_{i=1}^{l} \frac{tp_i}{tp_i + fn_i}
\]  

\[\text{(4.3)}\]

• **Fscore**, which focus on the relationship between data’s positive labels and those given by a classifier based on a per-class average.

\[
\frac{(\beta^2 + 1)\text{PrecisionRecall}}{\beta^2 \text{Precision} + \text{Recall}}
\]

\[\text{(4.4)}\]

### 4.3 Test Samples

Within test samples, a distinction has been made between two categories: seen objects and unseen objects.

Basically, the aim to have the seen objects category is to evaluate the network performance using different images from the same objects adopted during training. On the other hand, the unseen objects category serves to evaluate network’s performance using images from different objects which the network have never seen.

For more information about datasets used, see section 3.3. Finally, it is worth to mention that the criteria used during labeling test images it is the same explained in section 3.5.

#### 4.3.1 Test samples for single stream architecture

As RGB or depth test data for seen objects, it has been used images belonging to The RGB-D Object Dataset [26]. Specifically, it has been selected a set of 500 images for each type of grasp, making a total of 2000 images corresponding to 120 different objects. In order to avoid choosing similar images, samples corresponding to the same type of grasp have been randomly selected.

As RGB test data for unseen objects, it is used images from ALOI DB [27]. Specifically, it has been selected a set of 1152 images for each type of grasp, making a total of 4608 images and 64 different objects. It is important to mention that there are more RGB test images for unseen objects compared to other categories because it is wanted to have the same test volume of images as other researches in the same field.

As test depth data for unseen objects, it has been used images from BigBIRD [28], Multi-sensor 3D Object Dataset [29] and The RGB-D Object Dataset [26]. Specifically, it has been selected a set of 500 images for each type of grasp, making a total of 2000 images.

#### 4.3.2 Test samples for fusion architecture

As test data for seen objects, a total of 4000 images corresponding to 120 different objects from *The RGB-D Object Dataset* are used. For each type of grasp, there are 500 RGB images and its homologous depth image.

As test data for unseen objects, it has been used images from BigBIRD [28], Multi-sensor 3D Object Dataset [29] and The RGB-D Object Dataset [26]. Specifically, it has been selected a set of 500 RGB images and its homologous depth images for each type of grasp, making a total of 4000 images.
4.4 Single Stream Results

4.4.1 Seen Objects

Figure 4.1: Single Stream Seen Objects Confusion Matrix: a) RGB images, b) Depth images

As can be seen in Figure 4.1, Medium Wrap is always correctly classified no matter the type of data used. Regarding RGB data, Tripod is the class with worst classification score, specifically 89.2% correctly classified. Although 10.2% Tripod object images are incorrectly classified as Medium Wrap, it represents an acceptable error given the pose relationship between both grasps. Regarding depth data, Lateral Pinch is the class with worst classification score, specifically 74.6%. Should be noticed that 14.8% of Lateral Pinch object images are classified as Thumb Two Finger.

Overall metrics show successful behaviour of the classifier with seen objects for both type of data. However, there is more than twice per-class classification error rate with depth images than with RGB images. Furthermore, Recall, Precision and Fscore are four points higher using RGB images than depth images.

<table>
<thead>
<tr>
<th></th>
<th>Recall</th>
<th>Precision</th>
<th>Fscore</th>
<th>Error Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>RGB</td>
<td>0.96</td>
<td>0.97</td>
<td>0.97</td>
<td>0.015</td>
</tr>
<tr>
<td>Depth</td>
<td>0.92</td>
<td>0.93</td>
<td>0.93</td>
<td>0.037</td>
</tr>
</tbody>
</table>

Table 4.1: Single Stream Seen Objects Results

4.4.2 Unseen Objects

Classification accuracy for unseen RGB images is: Medium Wrap, 94.79%; Thumb two finger, 85.59%; Tripod, 78.3%; and Lateral Pinch, 56.5%. Classification accuracy for unseen depth images is: Tripod, 75.8%; Lateral Pinch, 67.8%; Thumb two Finger 67.4%; and Medium Wrap, 61.8%. It can be observed that for both type of data, there are classification difficulties between lateral pinch and thumb two finger classes. 39.4% RGB lateral pinch images are classified as thumb two finger and 10% of thumb two finger images are classified as lateral pinch. Similarly, with depth images, 32.2% lateral pinch images are classified as thumb two finger and 23.4% thumb two finger images are classified as lateral pinch. Also, it can be observed that the best top two classified grasp using specifically one type of data correspond to the worst two classified grasp using the other type of data.
Figure 4.2: Single Stream Unseen Objects Confusion Matrix: a) RGB images, b) Depth images

<table>
<thead>
<tr>
<th></th>
<th>Recall</th>
<th>Precision</th>
<th>Fscore</th>
<th>Error Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>RGB</strong></td>
<td>0.78</td>
<td>0.81</td>
<td>0.80</td>
<td>0.10</td>
</tr>
<tr>
<td><strong>Depth</strong></td>
<td>0.68</td>
<td>0.70</td>
<td>0.70</td>
<td>0.15</td>
</tr>
</tbody>
</table>

Table 4.2: Single Stream Unseen Objects Results

Finally, overall metrics show that error rate increased more or less by a factor of ten in relationship with seen objects. However, in this case there is less than twice per classification error rate comparing depth to RGB images.

Figure 4.3 shows one example image for each cell in Figure 4.2 a). It represents some error and correct predictions made by single stream architecture on unseen RGB objects.

Figure 4.3: Examples of True and False predictions done by single stream architecture on unseen RGB objects
4.5 Fusion Stream Results

Classification accuracy for fusion seen is: Medium Wrap, 100%; Tripod, 99.2%; Lateral Pinch, 98.6%; and Thumb two finger, 94.8%. Classification accuracy for fusion unseen is: Tripod, 94.4%; Medium Wrap, 86.8%; Thumb two finger, 83.0%; and Lateral Pinch, 62.4%.

Regarding seen objects, the only significant incorrect classification is that 4.8% of thumb two finger images are classified as lateral pinch.

As with previous cases in unseen objects, it is notable the classification confusion between thumb two finger and lateral pinch: 34.6% lateral pinch object images have been classified as thumb two finger and 14.2% thumb two finer object images have been classified as lateral pinch. Also, although Tripod class has the best classification results, it is worth mentioning that 4% of tripod object images have been classified as lateral pinch which represents an unacceptable error.

Overall metrics show that fusion architecture has better results than the single stream architecture for both seen and unseen.

<table>
<thead>
<tr>
<th></th>
<th>Recall</th>
<th>Precision</th>
<th>Fscore</th>
<th>Error Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Seen</td>
<td>0.98</td>
<td>0.98</td>
<td>0.98</td>
<td>0.009</td>
</tr>
<tr>
<td>Unseen</td>
<td>0.81</td>
<td>0.83</td>
<td>0.82</td>
<td>0.09</td>
</tr>
</tbody>
</table>

Table 4.3: Fusion Results


Chapter 5

Budget

This project has been developed using the resources provided by the B-SPIRAL group in Northeastern University and Image Processing Group of UPC. Thus, this cost will not be reflected in the budget.

The main costs of the project are due to the salaries of the researchers that worked on it. I consider my position as junior engineer while supervisors’ positions are considered as senior engineer.

Finally, all software used is open source and it does not have costs associated. The duration of the project is considered 25 weeks.

<table>
<thead>
<tr>
<th></th>
<th>Amount</th>
<th>Wage/hour</th>
<th>Dedication</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Junior engineer full-time</td>
<td>1</td>
<td>10.00 €/h</td>
<td>30 h/week</td>
<td>7500 €</td>
</tr>
<tr>
<td>Senior engineer</td>
<td>2</td>
<td>50.00 €/h</td>
<td>1 h/week</td>
<td>2500 €</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td><strong>Total</strong> 10000 €</td>
</tr>
</tbody>
</table>

Table 5.1: Budget of the project
Chapter 6

Conclusions

The main goal of the project was to design and implement an image based system for automatic grasp selection to improve prosthetic hands. In the introduction of the thesis as well as in the background theory section it was remarked the importance of researching into this field in order to improve user’s satisfaction and prosthetic functionality. Furthermore, it was declared the interest in using state-of-the-art techniques to solve it. This statement strongly suggested that deep learning artificial vision techniques could be used for grasp classification.

In order to meet the objectives, convolutional neural network architecture has been changed to recognize grasps instead of objects. Using this approach, there has been no need to train the network with large number of different objects; only with a set of objects that clearly contains the shape features for each type of grasp. Consequently, after the training process the network is able to classify novel objects with the same shape characteristics. Furthermore, transfer-learning and fine-tuning are assessed to exploit learned features from other classification tasks and to adjust the parameters of the model.

Two different architectures have been designed to handle single or fusion information streams. As it was expected, fusion architecture has better performance than the single one. However, single stream architecture results are good enough to be considered when only one type of data can be acquired.

Medium wrap achieves the highest classification accuracy using RGB images while tripod achieves the highest using depth images. That might be the reason why these two grasp types also achieve the highest accuracy using fusion configuration. It is suspected that tripod performed best in depth category because after the preprocessing, tripod labeled images show a defined and characteristic contour. In contrast, thumb two finger and lateral pinch generally achieve the lowest accuracy. The reason why is because both type of grasp are confused mutually. This is unsurprising because characteristics given in table 3.1 are quite similar for both type of grasps. In addition, this confusion is greater in unseen objects.

This work confirms that using deep learning-based computer vision techniques for grasp classification is useful for RGB information, as was reported by other studies. But, the work’s main novelty is to show that using depth information it is also possible to classify objects into grasps without recognizing them. Moreover, it shows that merging the two types of information, better results can be achieved. Furthermore, it’s worth mentioning that current work achieves state-of-the-art results using pretrained weights, which are kept frozen during training, and allow to use small sets of training data and generalize well with test data.

Knowing the randomness of EEG and EMG signals, this work demonstrates that incorporating more sources of information can help to reduce the lack of robustness and thus, improve user satisfaction and prosthetic functionality in terms of accuracy. Furthermore, grasp scalability can be accomplished by labeling new object images into new grasp types. Moreover, convolutional neural networks have good performance classifying numerous classes so it is expected to maintain great accuracies adding more grasps. Nevertheless, the crux of the problem will rely on which criteria is followed during the labeling process.
As future work, it might be interesting to adapt the system for cluttered scenes. Segmentation techniques could be used to identify an object and then introduce the output into the grasp classifier. Also, the best way to combine context signals with EEG and EMG altogether for grasp selection is left for future work. Finally, mobile app could be developed for prosthetic users allowing them to control the behaviour of the prosthetic hand for different tasks using the technology reported in this work.
Appendix A

Single stream architecture
Appendix B

Fusion architecture
Bibliography


