

Computer Vision enhanced by Deep Learning for Fashion Products Measurement

Bruno Miguel Rocha Teixeira

Master's Dissertation

Supervisor from FEUP: Prof. Ana Camanho

U. PORTO

FEUP FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO

Master in Engineering and Industrial Management

2020-06-29

Abstract

In a logistics business, the size of an item to be shipped directly impacts the transportation costs. HUUB is responsible for managing the supply chain of fashion brands - its partners, which are charged per each transportation done by an outsourced carrier. As the charged value is determined prior to the actual shipping, the success of the relationship between HUUB and its partner brands heavily relies on an accurate estimate of these costs. These expenses depend on the size of the transported items. This thesis aims to improve the measurement procedure for the items to be transported, which is currently a manual process. The development of a model capable of predicting the size of an item by returning its width, depth and height resorting to the use of a picture taken to the item is the main goal of this thesis.

In order to process an image, Computer Vision techniques are adopted. Based on a literature review, Computer Vision problems may be addressed through three different methodologies: (i) a traditional Computer Vision algorithmic approach; (ii) a Deep Learning method based on Convolutional Neural Networks; (iii) hybrid configurations, where Computer Vision algorithms empower Deep Learning models. Hence, this thesis presents three alternative models, developed using the three alternative methodologies and compares the performance of the different approaches. The first is a Computer Vision algorithm built from scratch, whereas the second one consists on a Convolutional Neural Network and the third is a combination of both.

Among the three, the Convolutional Neural Network model was selected to be implemented as it achieves the best results. The model provides the mean absolute errors of 0.79cm, 0.75cm and 0.46cm for the width, depth and height respectively. Moreover, the selected model also provides extremely fast estimations by performing over 1.000 predictions in one minute. Finally, the impact of the project on HUUB procedures for the estimation of items dimensions is very significant, representing a 65% reduction of the time spent in this task.

Resumo

No contexto de uma empresa logística, o tamanho de um item a ser expedido tem impacto direto no que são os seus custos de transporte. A HUUB é responsável por realizar a gestão da *supply chain* de marcas de moda - os seus parceiros de negócio - que são cobradas por cada transporte realizado por uma transportadora subcontratada. Dado que o montante a ser cobrado é determinado anteriormente ao transporte efetivo, a relação da HUUB com os seus parceiros depende seriamente de um alto nível de precisão no processo de estimativa destes custos. De modo a obter estes custos, o tamanho dos itens transportados tem de ser naturalmente considerado. Esta dissertação tem como objetivo a melhoria do processo de medição, que atualmente é realizado de forma manual, recorrendo ao desenvolvimento de um modelo capacitado para providenciar o tamanho de um produto através do retorno do seu comprimento, largura e altura necessitando apenas de um *input*: uma fotografia.

De modo a que a imagem possa ser processada, são adotadas técnicas de Visão Computacional. Tendo como base a revisão da literatura, problemas de Visão Computacional podem ser abordados através de três metodologias: (i) uma tradicional abordagem baseada em algoritmia; (ii) desenvolvimento de um modelo *Deep Learning*, no qual Redes Neurais Convolucionais podem ser usadas; (iii) configurações híbridas, onde os tradicionais algoritmos são utilizados para fortalecimento dos modelos *Deep Learning*. Tendo esta informação como base, os três tipos de modelos são desenvolvidos e, por fim, comparados. O primeiro modelo desenvolvido consiste num algoritmo feito de raiz, enquanto o segundo baseia-se numa Rede Neuronal Convolutiva e o terceiro é uma combinação de ambos.

De todos os modelos desenvolvidos, o modelo baseado na Rede Neuronal Convolutiva é selecionado para ser implementado, uma vez que atinge os melhores resultados, providenciando erros médios absolutos de 0.79cm, 0.75cm e 0.46cm para o largura, comprimento e altura. Adicionalmente, o modelo selecionado é capaz de fornecer mais de 1.000 estimativas em um minuto, sendo assim extremamente rápido no processo de medição. Finalmente, o impacto do projeto no processo de medição em termos operacionais é melhorado consideravelmente, levando a uma redução de 65% da sua duração.

Acknowledgments

I wish to express my most deepest gratitude to my thesis advisor, Professor Ana Camanho, for her invaluable patience and support through the dissertation. Her staggering advice and endless availability were crucial for the project success. For this, I will always be grateful.

To HUUB, my new home, and to the Artificial Intelligence team that has wonderfully treated me from the first day and unconditionally provided me all the conditions to the dissertation accomplishment.

To Carolina and Daniel, for embracing this new adventure alongside me.

To Cristina Fernandes, my mentor inside the company, I would like to pay my special regards. Her extraordinary guidance made the thesis to be possible by consistently steering me in the right direction. No words can explain the importance of her input but the most sincere thank you.

To FEUP, for providing me with the best years of my life. All the countless stories and experiences lived in the last five years are completely irreplaceable.

To Miguel and all my beloved friends FEUP has brought me, but specially him. They are responsible for wonderful and beautiful lived memories and will be undoubtedly responsible for many more. But now without being broke.

Finally, to my mother, Lúcia and to my father, Rui, for guiding and supporting me in the journey of my life. They are the most hard-working people I have ever met and an example of humility. The person I am today is the result of growing up by having both of them as role models and I can only aspire to be like them in the future. I will be indebted to my parents for the rest of my life.

To all of you, thank you.

“The real question is not whether machines think but whether men do. The mystery which surrounds a thinking machine already surrounds a thinking man.”

Burrhus Frederic Skinner

Contents

1	Introduction	1
1.1	Company Outline	1
1.2	Problem Addressed	2
1.3	Project Objectives	3
1.4	Methodology	3
1.5	Thesis Outline	4
2	Literature Review	5
2.1	Previous Computer Vision models applied to the measurement of items	5
2.2	Computer Vision Fundamentals	6
2.2.1	Understanding what an image is	6
2.2.2	Traditional Computer Vision and Deep Learning	6
2.2.3	Image Segmentation	7
2.2.4	Image Sharpening	7
2.3	Deep Learning Oriented for Computer Vision	8
2.3.1	Overview of DL techniques applicable to CV	8
2.3.2	Convolutional Neural Networks	8
2.3.3	The Learning Process	11
2.3.4	Optimisation Methodologies	13
2.3.5	Other Features of Convolutional Neural Networks	16
2.3.6	Building a CNN	17
2.3.7	Prevent overfitting through training monitoring	18
2.4	Connecting the Literature and the Project	18
3	Problem Framework and Description	19
3.1	Operation AS-IS	19
3.1.1	Teams Involved	19
3.1.2	The processes inside the AS-IS situation	20
3.1.3	Main Problems of the AS-IS situation	22
3.2	Operation TO-BE	22
3.2.1	What changes and what to expect	23
3.2.2	Item photographing	23
3.2.3	Next steps	24
4	Models Outline and Results	25
4.1	Dataset Outline	25
4.2	Model I - CV Algorithm	26
4.2.1	Axis and grid detection	26
4.2.2	Drawing lines for axis and grid	30

4.2.3	Scale Calibration	32
4.2.4	Items' dimensions estimation	33
4.3	Model II - DL model using a CNN	34
4.3.1	The Architecture	34
4.3.2	CNN training	35
4.4	Model III - Hybrid Configuration	37
4.5	Comparison of Models' performance in the estimation	38
5	Discussion of Results	39
5.1	The reason behind the hybrid unsuccess	39
5.2	Further errors analysis	40
5.2.1	Model I	40
5.2.2	Model II	43
5.3	Models global comparison and Verdict	48
5.4	Impact in the time spent on the measurement process	48
6	Final Considerations and Future Work	49
6.1	The innovative side of the adopted methodology	50
6.2	Key Outcomes	50
6.2.1	Models Summary	50
6.2.2	Impact at HUUB	51
6.3	Further Improvement Opportunities	52
A	One Point Lesson for Photographing	55

Acronyms and Symbols

CV	Computer Vision
ML	Machine Learning
DL	Deep Learning
ANN	Artificial Neural Network
PCA	Principal Component Analysis
NN	Neural Networks
CNN	Convolutional Neural Networks
CL	Convolutional Layer
FCL	Fully Connected Layer
RGB	Red Green Blue
HSL	Hue Saturation Lightness
SVM	Support Vector Machine
SGD	Stochastic Gradient Descent
BGD	Batch Gradient Descent
MGD	Mini-batch Gradient Descent
ReLU	Rectified Linear Unit
LReLU	Leaky Rectified Linear Function
ELU	Exponential Linear Unit
RBM	Restricted Boltzmann Machines
MSE	Mean Squared Error
CLT	Central Limit Theorem

List of Figures

2.1	Convolution between a matrix P and a smaller one K , originating S . Under the context of CNN, P may be a picture, K is named <i>kernel</i> and S is called <i>activation map</i>	9
2.2	The behaviour of two Convolutional Layers of 32 and 64 kernels after a coloured image enters the network.	10
2.3	Simplistic visualisation of all the convolutions between a kernel K and the input matrix P , originating the activation map S	16
2.4	Illustration of the max pooling operation.	17
2.5	Schema showing the commonly used CNN architecture patterns.	17
2.6	Left: Validation and Training loss for a converging model. Right: Validation and Training Loss for an overfitting model.	18
3.1	Current map of processes.	19
3.2	Current map of processes performed for an item measurement.	20
3.3	Map of processes after the project implementation. The processes with a green outline are directly impacted.	23
3.4	With this project, the manual item's width, depth and height measuring is replaced by a picture.	23
3.5	Example of an item's picture correctly taken according to photographing guidelines.	23
4.1	Distribution of the items' dimensions in the dataset.	25
4.2	Schema of the first step of the model I.	26
4.3	Pictures are distinguished according to the presence or absence of shadows.	27
4.4	Architecture of the CNN responsible for distinguishing between pictures with and without shadows.	27
4.5	Training evolution of the CNN responsible for distinguishing between pictures with and without shadows.	28
4.6	The grid pixels usually have larger standard deviations against the remaining pixels whose maximum RGB component is the green one. The dashed vertical line "image average standard deviation" represents the average standard deviation of the whole image. The picture used for the graph generation is the type II picture from Figure 4.3 and this graph has a similar shape for the remaining pictures.	29
4.7	Grid mask obtained.	30
4.8	The algorithm "walks" inside a mask in 3x3 squares looking for lines identification.	30
4.9	The image on the left shows the walking path performed by the algorithm before the line fitting turns each path into straight lines, which is seen in the image on the right.	31

4.10	The intersection points are represented by the yellow dots. Their spatial positioning is crucial to build a conversion from pixels in the image to cm in real life.	33
4.11	Example of a quadratic regression matching the distance on the image in pixels to real distance in cm for the x-axis of the picture present in Figure 4.10.	33
4.12	The red and green lines from the image on the left allow the box building.	34
4.13	CNN architecture used for Model II.	34
4.14	The impact of the mini-batch size in the model generalisation.	36
4.15	Model II training. No signs of overfitting, the model early converges to a low generalisation error and keeps fluctuating around it for the remaining epochs.	36
4.16	Model III configuration. Firstly uses the Model I to perform image segmentation and then the Model II to accomplish the estimate.	37
4.17	Image segmented through the highlight of the axis and grid and the input of the CNN.	37
4.18	Absolute errors distribution for the 3 models.	38
5.1	Difference between the outputs coming from the second CL for models II and III.	39
5.2	Exhibition of the problems leading to the largest errors when estimating with Model I	41
5.3	Errors distribution per root cause and dimension for the considered outliers.	41
5.4	Individual histograms for each of the three dimensions' errors.	43
5.5	Height errors distribution in pictures with and without shadows.	44
5.6	Box plot showing the influence of shadows per dimensions groups	45
5.7	Picture with a dense dark region around the box origin.	47
5.8	The item photographing reduces the measurement procedure by 65%.	48
A.1	One Point Lesson for Photographing (1/2).	55
A.2	One Point Lesson for Photographing (2/2).	56

List of Tables

4.1	Two of the colour restrictions imposed to the pictures of each type.	28
4.2	Mean absolute errors in cm per model and dimension (lower is better).	38
5.1	Mean absolute error (cm) caused per each problem raised in the analysed outliers	42
5.2	Results obtained by applying the Levene test and the two-sample t-test to compare the impact of shadows on each dimension.	44
5.3	Results obtained by applying the Levene test and the two-sample t-test to compare the impact of shadows on each height's group.	45
5.4	Pairwise comparison of the heights groups regarding the errors in pictures with shadows.	46
5.5	Pairwise comparison of the heights groups regarding the errors in pictures without shadows.	46
5.6	Items distribution per cm inside each height group.	47

Chapter 1

Introduction

Within a logistics operation, the efficiency of supply chain management is critical. A core objective involves the reduction of logistic costs. Among the main drivers of these costs are transportation costs. This chapter describes the project carried out in a logistics company - HUUB, a logistics platform for fashion brands. The aim of the project is to enhance the estimation of transportation costs for the items managed by the company.

HUUB is a start-up with a business model consisting in a partnership with fashion brands where HUUB has full responsibility for their supply chain management. Inside this association, HUUB's functions go from warehousing to transportation which, in turn, is performed with the collaboration of carriers. Importantly, a partner fashion brand is charged per each transportation with the cost defined prior to the actual shipping. Hence, the accuracy of this cost estimate is of crucial importance for HUUB in order to maintain its competitiveness and a healthy relationship with the partner brands.

This dissertation compares three models developed under Computer Vision (CV) and Deep Learning (DL) techniques, each intending to provide robust estimates of dimensions for the shipped items, which directly impact the transportation costs predictions and, thereby, HUUB's overall performance.

1.1 Company Outline

HUUB's mission states the commitment to grow its brands' business by putting itself "at the center of a dynamic ecosystem: customers, end users, suppliers, alliance partners, the real end-to-end". Founded in 2015, HUUB is a startup focused on accomplishing its mission by operating with fashion brands, being fully responsible for managing their supply chain. This way, HUUB aims to provide the partners a competitive edge, helping them to grow in a sustainable way.

By joining HUUB, partner fashion brands no longer need to spend a large fraction of their time devoted to managing their supply chains. Instead, these companies are able to concentrate their efforts on production, marketing and sales, relying on HUUB to handle the path from production to the final client, either through wholesale (B2B) or e-Commerce (B2C). Moreover, by managing

the transportation flow of its fashion partners all together, HUUB is able to build scale economies and therefore negotiate better transportation prices than brands would be able to get alone.

HUUB's services include both distribution and warehousing. HUUB is currently operating with three different warehouses, two in Portugal and one in the Netherlands. The distribution is performed resorting to partnerships with carriers.

HUUB also developed a web-based platform named SPOKE, aiming to connect all its stakeholders, allowing the partner brands to visualise the status of their whole supply chain in a collaborative, integrated and unified experience. SPOKE's service portfolio is significant, allowing transparency and control of the supply chain in real time through the possibility of tracking and analysing orders, which can be uploaded and edited on the platform. SPOKE also provides the possibility of inventory management by checking in real time if a given product is physically available in the warehouse. Additionally, the platform offers other features such as integration with Shopify, Ucommerce or carriers' tracking systems.

Importantly, HUUB charges partner fashion brands in a twofold way. Firstly, HUUB establishes with the partner a fixed price per item in order to assure the logistics (for instance, warehousing costs), usually defined at the start of each season or when the fashion brand becomes a partner. Secondly, per each shipping, the brand incurs on a fee, which is intended to be close to the carrier cost, but with a margin for HUUB. Despite having to pay the HUUB' margin, fashion brands gain an advantage due to the bargaining power of HUUB with carriers, coming from scale economies.

HUUB is currently in a growth phase, having fulfilled over 20.000 orders and managing more than 500.000 items, which sum up to €15.000.000 in revenue from handled products. The company is not constrained by borders, acting in over 80 countries spread over all continents, connecting with more than 50 partner brands at the moment.

1.2 Problem Addressed

The shipping cost to be imposed to a partner brand is estimated taking into account the weight and volume of the items to be shipped, as the carrier performs its pricing taking into consideration these two physical features. Consequently, the need for having these features in the database is clear. However, at the time of writing, less than 3% of the items transacted by HUUB had the corresponding physical features catalogued.

In order to overcome the lack of measured items, HUUB had already developed a predictive algorithm responsible for predicting the items' dimensions and weight of an item taking into account features such as the type of product (if it is a t-shirt, a skirt...), the gender, age group (baby, kids, adults...) or size. This algorithm needs, however, real measurements data for training which, at the moment, are not available at a large scale for the algorithm to perform at a satisfactory level.

Real measurements are currently obtained after the item is picked, through a manual procedure in HUUB's warehouses, making use of a measuring tape and a weighing scale, which is a very time consuming task. Moreover, with a hand-operated procedure, the human error is unavoidable

in measurements reading and in the process of data imputation to the database. These can significantly jeopardise the models and predictions which rely on them. Hence, the need for tackling this particular problem and accelerating the measuring process is an issue with critical importance for HUUB.

1.3 Project Objectives

The dissertation focus is the improvement of the measurement process through the implementation of a model that is able to estimate the dimensions of an item having as input a single captured image. The model development is performed resorting to CV and DL techniques. With the deployment of the work developed within this project, a specific and a general picture objectives can be stated.

Firstly, considering the specific short-term objective, the target is to eliminate the need of measuring the width, height and depth of an item with a measuring tape. Instead, the task is just to capture an image of the item. Consequently, this new approach intends to mitigate the human error and to speed up the measuring process.

Secondly, as a more general objective, a faster measurement procedure means a larger rate of training data received by other predictive models, already in production at HUUB (referred in Section 1.2), leading to the enhancement of their performance. This results on a better transportation cost estimation and, consequently, a stronger relationship with the partner brands.

Furthermore, and less related to the task of the developed model itself, the project had in view meeting HUUB's desire of exploring and introducing CV know-how allied to DL in the company.

1.4 Methodology

Prior to the model development itself, the project's first stage is related to the data gathering process. In this situation, the data is a single picture of an item and the corresponding width, depth and height measurements. Therefore, the focus is on developing a set of guidelines capturing pictures.

Regarding the model building and consequent problem solving, a careful literature review allows to categorise CV models in three groups: (i) traditional CV algorithms where the use of DL tools is reduced or nonexistent; (ii) pure DL models where CV algorithms play a minimal role; (iii) hybrid models, where CV algorithms and DL models complement each other. From this point, the project's path is crystal clear: the different approaches should be tested in order to choose the best one to achieve HUUB's objective.

First, using the data already gathered, a traditional CV algorithm is developed. Secondly, a DL model is developed in an attempt to overcome the results of the traditional CV model. Finally, a hybrid model is assembled.

In the end, a comparison between the three models is performed and a verdict is rendered. The comparison is established through the mean absolute error, a continuous metric, for each of the

three measured dimensions. Other performance indicators, such as the time needed for reaching an estimate, are also explored.

1.5 Thesis Outline

The dissertation is structured in six chapters. The next chapter is the Literature Review, where the theoretical background for the work done is presented. The technical concepts applied to the models developed are described, based on the analysis of papers available in the literature. This chapter has a first section dedicated to reviewing related past work, a second one concerning fundamental CV concepts and finishes with a section focused on DL.

The third chapter elaborates deeper on the problem description. It is divided in two main sections. First, the AS-IS condition is presented through a map of processes where the items' dimensions estimation problem has an impact on. Each of these processes is explained and the HUUB's teams responsible for each of these tasks is introduced. In the second part, the TO-BE situation describes where the gains with the project implementation.

Afterwards, the fourth chapter starts by briefly describing the used dataset. Subsequently, the focus is on a thorough explanation of the work done regarding the models development. Each model is individually delineated, starting by a CV algorithm programmed from scratch, going through the DL model and ending up with the model with the hybrid configuration. The chapter ends with the presentation of the models' results in the estimation task assessed resorting to the mean absolute error.

The fifth chapter, starts by carefully analysing the errors from each model. The study of errors has the fundamental objective of identifying the main reasons behind less accurate predictions. Then, pros and cons with respect to each model are discussed, culminating with the choice of the model to be implemented. Afterwards, the impact of the project on the measurement process is quantified.

Finally, the last chapter of the thesis provides an overview of the work done and a reflection regarding the outcomes of the project and the contributions of the dissertation to the scientific field of CV. The dissertation closes with the statement of improvement opportunities identified for future work.

Chapter 2

Literature Review

The current chapter starts by presenting the existing models related to the project developed in the dissertation. Afterwards, the focus is on explaining the theory behind the models developed, including CV foundations and DL models oriented to CV.

2.1 Previous Computer Vision models applied to the measurement of items

The CV methods used for items' measurement often consist of the development of either specialised algorithms or DL models intended to solve regression problems.

Karakose et al. (2019) proposed an interesting algorithmic-based work to provide a real-time object detection and its 2D size measurement. The method consisted on simultaneously capturing two objects: (i) a reference object, whose dimensions are known; (ii) the object to be measured. The measurement procedure started by detecting both objects through the Canny edge detector, which is an algorithm used to find an item's edges. Once the detection is performed, the object is measured through its size in relation to the reference. Also, White et al. (2006) applied an algorithmic approach to CV aiming to automatically measure the length of fish. The length was obtained after the algorithm detected the fish on the picture through colour thresholding techniques. As the picture was captured always on the same station with a fixed camera, the length estimation was straightforward.

Concerning DL approaches to CV, in fact, the amount of work directed to regression problems is considerably inferior to other problems such as classification or detection. Nevertheless, a commonly addressed problem is the age estimation as in Liu et al. (2015), where a Convolutional Neural Network (CNN) is applied to predict the age of a person on a given image. However, regarding DL models intending to actually get items measurements from pictures, the most similar work was done by Shimobaba et al. (2018) in the context of digital holography. The work consists on the depth prediction of a given object resorting to a single hologram. For the prediction, the authors proposed a DL approach over the gathered images by applying a CNN based regression.

2.2 Computer Vision Fundamentals

CV can be defined as the scientific field of extracting information from images. It can be seen as a combination of image processing and pattern recognition where the expected output is the comprehension of an imputed image. CV pursues the development of an intelligent machine that has its own eyes (Wiley and Lucas, 2018).

2.2.1 Understanding what an image is

An image is a large set of single units named pixels. Therefore, these pixels are the raw building blocks of an image (Rosebrock, 2017). They are responsible for carrying information representing a colour. The pixel colour is usually represented by a colour model such as RGB (red, green, blue) or HSL (hue, saturation, lightness), among others.

Regarding the RGB colour code, the pixel carries a list composed by three values: one for each of the red, green and blue dimensions. Importantly, each of these values is contained in the range [0-255] where 0 represents the absence of the given colour and 255 is full presence. Another key thing to remember is the fact the RGB colour code is additive. This is to say, the more each colour is added, the brighter and whiter the pixel is going to be from (0, 0, 0) representing black to (255, 255, 255) representing white (Rosebrock, 2017).

2.2.2 Traditional Computer Vision and Deep Learning

Since the release of the AlexNet, a CNN for Large Scale Visual Recognition (Krizhevsky et al., 2012), CV experienced a turning point with the introduction of DL to the field. Problems which were assumed to be unsolvable through traditional CV algorithms are now being solved with Artificial Neural Networks (ANN). However, traditional CV cannot be considered obsolete as DL does not solve all the problems and has its own drawbacks (O'Mahony et al., 2020).

DL advantages are clear, as greater accuracy is obtained in tasks as semantic segmentation, image classification/regression or object detection. Also, DL offers much higher levels of flexibility, since one of the major drawbacks of CV traditional approaches is that the algorithms tend to be very specific and oriented to a particular problem. However, in occasions where simpler techniques may be applied (for instance, colour thresholding) DL may be considered an overkill, since traditional CV might easily solve the issue. In fact, DL usually requires large datasets, whereas traditional CV does not. Moreover, a DL model requires using enormous amounts of parameters which means expending large amounts of computing resources. All things considered, DL provides greater accuracy and versatility, being however sometimes considered as a black box method, whereas classical CV is more transparent and optimised for performance and power efficiency (O'Mahony et al., 2020).

The two techniques should not be necessarily considered as substitutes. Instead, traditional CV and DL may work together on hybrid models for better performance. DL offers better classification metrics, but at a significant computational cost. CV techniques are able to limit the problem

for DL models by, for instance, highlighting certain features on images (background subtraction and segmentation) that ease the identification of patterns, augmenting data or simple image modifications such as sharpening. The key idea is to generate superior models through combining the best of both worlds (O'Mahony et al., 2020).

2.2.3 Image Segmentation

Image segmentation consists on highlighting important parts of an image based on certain criteria. This way, the model may focus only on the relevant areas of an image and alleviate the computational weight. There are four main groups of image segmentation algorithms: region-based segmentation, edge detection segmentation, segmentation based on clustering and, finally, segmentation based on weakly-supervised learning in CNN (Song and Yan, 2017).

Regarding region-based segmentation, it is focused on partitioning an image into multiple regions, where all pixels on it are similar with respect to some features, such as colour or intensity (Myilsamy and Muthukrishnan, 2011).

Edge detection segmentation may be described as the process of classifying and placing sharp discontinuities over an image, which are found through abrupt alterations in pixel concentration (Myilsamy and Muthukrishnan, 2011). The most traditional procedures on edge detection involve convolving over an image through an operator (convolution operation is detailed in Section 2.3.2.1). This kind of operators are built aiming for the detection of large pixel concentration variations, whereas returning zero on steady regions (Myilsamy and Muthukrishnan, 2011).

Another group of segmentation algorithms group is composed by clustering based methods. Intuitively, these algorithms perform the pixels segmentation using clustering techniques from which K-means is the most used. The benefits concerning K-means are that the algorithm is fast, simple and highly efficient. However, the determination of the correct value for K (number of groups) is usually hard to accomplish. The functioning concept of clustering based methods is to assign each pixel to a group (Song and Yan, 2017).

Finally, the group of segmentation algorithms is composed by CNNs. For this type of segmentation, the network is built and trained to classify each pixel of an image into a given class (Song and Yan, 2017). The CNN functioning is explained in more detail in Section 2.3.1.

2.2.4 Image Sharpening

Image sharpening is an operation applied to images with the objective of sharpening details and accentuate edges. Similarly to the edge detection segmentation explained in Section 2.2.3, the sharpening is applied making use of a convolution between an operator and a collection of pixels belonging to the image (Szeliski, 2011). The number of convolutions performed in this process is identical to the number of pixels present in the image. A commonly used operator in image sharpening is represented by the matrix in expression (2.1).

$$Sharp = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix} \quad (2.1)$$

The convolution between this matrix and a 3x3 square from the image may have two outcomes: (i) if the pixels values belonging to the 3x3 square are similar, the result from the convolution is identical to the original pixel value; (ii) if there are considerable differences inside the considered square, then the result is distinct and the found variation is accentuated.

2.3 Deep Learning Oriented for Computer Vision

DL may be seen as a branch of Machine Learning (ML) that emerged to overcome the limited ability of conventional ML techniques to process natural data in their raw form. Under the context of ML, and therefore DL as well, supervised learning is the most common form of learning, i.e., the learning is done knowing the ground truth value of the dependent variable. Both in classification and regression problems, the functioning essence is the same: the machine is imputed with a given image of a dataset and produces some output. This output may be, on a classification problem, a vector with a probability for each class or, for a regression problem, a continuous value. In both cases, the output is compared with the ground-truth label/value to determine the prediction error. The model will then iterate in order to reduce this error after which it proceeds to calculate again a new output, and so on (Lecun et al., 2015).

2.3.1 Overview of DL techniques applicable to CV

Sinha et al. (2018) presented an outline of the DL algorithms most frequently applied in the area of CV. These algorithms are commonly branched into CNNs, Restricted Boltzmann Machines (RBM) and Autoencoders. In fact, RBMs and Autoencoders are notable variants of ANNs, which perform reconstruction instead of classification or regression on an unsupervised learning environment. Reconstruction consists on identifying hidden features over an image (also named as *latent representation*) through an iterative learning process, and using them to reconstruct the imputed image. On the other hand, CNNs are more indicated to classification or regression problems. Therefore, as this dissertation mainly focuses on a regression problem, CNNs come as the most indicated methodology.

2.3.2 Convolutional Neural Networks

CNNs or *ConvNets* are ANNs conceived to process data coming in the form of multiple arrays, as the case of a colourised image composed of three 2D arrays, where each contains the respective colour intensity (Lecun et al., 2015). A big motivation for the development of CNNs is the fact that a regular ANN does not scale well to images. The number of parameters needed to work with images making use of a regular ANN easily scales to a computationally unmanageable amount.

Moreover, with large number of parameters comes a tendency to an undesired overfitting (Rosebrock, 2017; Goodfellow et al., 2016).

2.3.2.1 The Convolution Operation and Kernels

The "Convolutional" designation in CNNs comes from the fact the network executes a mathematical operation named *Convolution*, which is a specialised kind of linear operation. In terms of DL, a convolution is an element-wise multiplication of two matrices followed by a sum (Rosebrock, 2017; Goodfellow et al., 2016). Consider the (m,n) sized matrix P and a smaller matrix K . In mathematical terms, the convolution between the two matrices is given by the following expression.

$$S(i, j) = K * P(i, j) = \sum_m \sum_n K(i+m, j+n)P(m, n) \quad (2.2)$$

In order to visualise the expression (2.2), note Figure 2.1, inspired by a visual example present in Goodfellow et al. (2016). In this figure, the smaller matrix K slides over P , performing the convolution operation over each square 2×2 .

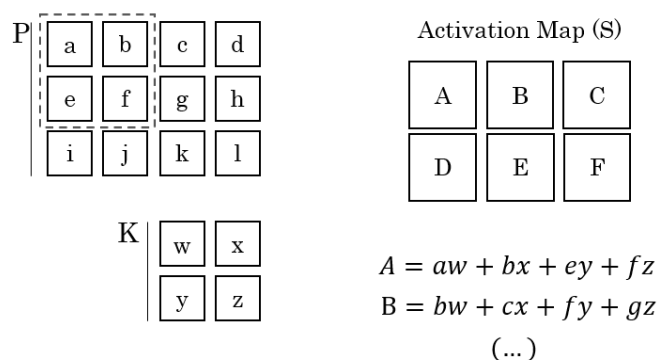


Figure 2.1: Convolution between a matrix P and a smaller one K , originating S . Under the context of CNN, P may be a picture, K is named *kernel* and S is called *activation map*.

When talking about a CNN, considering the very first layer of a network, P is a picture and, if colourised, has depth equal to 3 (for red, green and blue colours). On the other hand, the smaller matrix K is called *kernel* (or *filter*). A kernel has arbitrary 2D dimensions (2×2 in the Figure 2.1 example) but the depth fully depends on the depth of the larger matrix. Hence, one can infer that the kernel depth must be 3, if the image is colourised. A kernel assumes primordial importance within a CNN as it contains the changeable parameters which allow the actual learning. In Figure 2.1, the kernel has 4 learnable parameters and, for the RGB image case, this number becomes 12 (Goodfellow et al., 2016; Rosebrock, 2017).

The result of a convolution operation between the two matrices results in a single number, which is then placed in a new matrix S . Within a CNN, S is called *activation map*. Each element of an activation map may result of a single convolution (as the example in Figure 2.1) or may result from the a sum of several. The latest occurs when the depth of P and K is larger than 1. For instance, considering the situation of P , which has depth 3, to originate an element of the

activation map, 3 convolutions are needed. These 3 convolutions are performed at the same exact 2D positions but each at different depth levels. The sum of these three convolutions provides a value for the activation map (Goodfellow et al., 2016).

2.3.2.2 Convolutional Layers

Convolutional layers (CL) are the most important layers in a CNN. Sinha et al. (2018) describes CLs as layers where a CNN employs several kernels to convolve a full image generating different activation maps. An important aspect of CLs is that, on the first CL, the kernel depth coincides with the number of channels in the imputed image (for instance, if the image is coloured then 3 channels are accounted). On deeper CLs, the depth is given by the number of kernels applied on the previous layer (Goodfellow et al., 2016; Rosebrock, 2017). In Figure 2.2, note there is an input layer followed by two CLs. The number of kernels in a CL is arbitrary but is fully responsible for the depth of the next CL kernels. For instance, CL1 has 32 kernels which will lead to 32 activation maps from the imputed image. The input of CL2 is now the set of 32 activation maps coming from CL1 and, thereby, each kernel of CL2 must have depth 32.

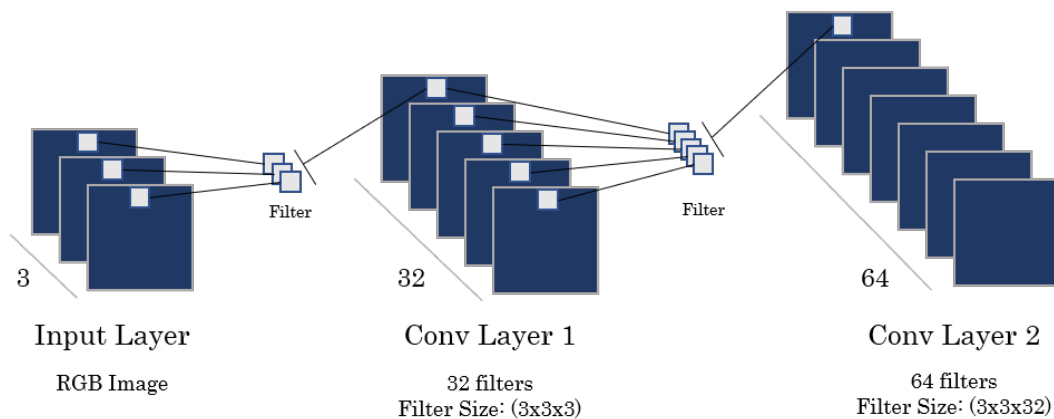


Figure 2.2: The behaviour of two Convolutional Layers of 32 and 64 kernels after a coloured image enters the network.

An interesting exercise is to evaluate the amount of parameters of a network. Considering what could be a partial CNN represented in 2.2, the number of parameters present in CL1 is given by $32 \times (3 \times 3 \times 3 + 1)$ and in CL2 by $64 \times (3 \times 3 \times 32 + 1)$, which sums to 19392 weights. Note the value 1 summed to each kernel, which represents the bias parameter.

A much relevant issue in CLs is the output size control of the activation maps against the input size. Considering the small example in Figure 2.1, it is possible to notice that although the input size is a matrix 4x3, the resulting activation map is 3x2, representing information losses. Regarding this issue, Karpathy (2019) highlights two crucial hyperparameters. The first is *stride* which is the step size when performing the kernel sliding over the image. In the Figure 2.1, stride equals one as the kernel is moved one pixel at a time. Intuitively, if the stride is two, then the movement is done on a two-pixels sized step, and so on. One can convince himself that the larger

the stride, the smaller the size of the activation map. The second hyperparameter to be referred is the size of the *zero-padding* which is simply pad the input volume with zeros around the border (in the Figure 2.1, no padding is used). Zero-padding is useful when it is intended to keep the input volume since, when performing the convolution of a kernel, the width and height are naturally reduced. Karpathy (2019) provided a very useful equation that allows to control the output size taking into account not only the input size but also the kernel size, padding and stride:

$$Size_{out} = \frac{Size_{in} - F + 2P}{S} + 1 \quad (2.3)$$

, where the input has to take a square shape, F is the 2D size of the kernel, P is the amount of padding used and S is the stride. Importantly, if the result of expression (2.3) is not an integer, then the setting of hyperparameters is considered as not being valid (Karpathy, 2019).

2.3.3 The Learning Process

The learning process can be considered to have two stages. The first one is the forward pass. At this stage, a model is shown one or several images of a dataset and outputs a vector of scores, using a *scoring function*. The output is then put against the ground-truth label/value and a *loss function* is generated. At this point, the second stage starts. It is named *backpropagation* and consists on minimising the loss function through altering the network parameters. The way these values are changed is dictated by an optimiser (Lecun et al., 2015; Rosebrock, 2017).

2.3.3.1 Scoring Function

The scoring function maps the data into the output. In a classification problem, the scoring function assigns an image to a given class label. Conversely, on a regression problem, the scoring function output is a continuous value. Concisely, the scoring function results from passing an image through the network and making the computations inherent to it (Rosebrock, 2017).

2.3.3.2 Activation Functions

An activation function is a function that is coupled with each neuron on a network. It is responsible to decide whether a neuron is fired or not, taking into consideration if the neuron's input is pertinent for the model's prediction (Rosebrock, 2017). The Rectified Linear Unit (ReLU) is the most popular non-linear activation function (Lecun et al., 2015) and it is given by expression (2.4).

$$f(t) = \max(0, t) \quad , \quad t = \sum_{i=0}^n w_i x_i \quad (2.4)$$

The positive part of the function is the identity, allowing ReLU to alleviate the vanishing gradient effect, which happens when the gradient used in the optimisation becomes so small the learning cannot go further (subject thoroughly explained in Section 2.3.4. This way, the corresponding neuron is highly efficient computationally wise (Lecun et al., 2015). In fact, as shown

by Glorot et al. (2010), ReLU typically learns a lot quicker in networks with a high number of layers. By contrast, neurons coupled with a ReLU are at a sensible position during the optimisation since the gradient results on the value 0 whenever the neuron does not activate. This could lead to cases where a neuron never activates given that a gradient-based optimisation algorithm does not adjust the weights of a unit that never fires initially. Consequently, the learning is expected to be slower when training ReLU networks with constant 0 gradients (Maas et al., 2013).

Aspiring to solve the ReLU main drawback, Maas et al. (2013) used an altered version of ReLU named Leaky Retified Linear Function (LReLU). The Leaky ReLU is characterised for allowing a neuron to have a small, non-zero gradient when it is not active. Similarly, other variants of ReLU were created such as the Parametric ReLUs (PReLU) or Exponential Linear Unit (ELU) (Clevert et al., 2016).

2.3.3.3 Loss Function

After the scoring function is applied, the loss function focuses on measuring how consistent the predicted scores are regarding the ground truth label/value y_i present in the training data (Karpathy, 2019). Naturally, the goal is to minimise this loss function. Tang (2013) essentially explores the two commonly used classifiers in classification tasks. One of those is Multiclass Support Vector Machine (SVM) and the other is Softmax. Between these two, according to Tang (2013), Softmax is the standard method for this type of tasks.

Softmax can be described as a generalisation of the binary form of Logistic Regression to multiple classes. The output is a probability for each class, which translates on a much easier and intuitive interpretation (Karpathy, 2019). Softmax starts its process through the usage of the resulting value from the scoring function. This method uses the cross-entropy loss which is given by the expression (2.5), where k represents the ground truth label for the image x_i and s_k is the corresponding scoring value.

$$L_i = -\log(e^{s_k} / \sum_j e^{s_j}) \quad (2.5)$$

The functioning principle of the cross entropy loss function is the minimisation of the negative log likelihood of the correct class (Rosebrock, 2017). The Softmax function is, in fact, $e^{s_k} / \sum_j e^{s_j}$ and contains a probabilistic interpretation:

$$P(k | x_i; W) = \frac{e^{s_k}}{\sum_j e^{s_j}} \quad (2.6)$$

Expression (2.6) may be seen as the normalised probability set to the correct class k corresponding to the image x_i calculated resorting to the network parameters, represented in the matrix W . Considering only the scores for each label, these are interpreted by the Softmax classifier as the unnormalised log probabilities. In order to eliminate the log scale, the exponentiation occurs. However, probabilities are still unnormalised and, because of that, Softmax performs the division so that these sum to one. In the loss calculation (expression 2.5), the negative natural logarithm is

calculated for the ground-truth label and this is the x_i associated loss. The whole dataset loss may be found through the average of all losses (Rosebrock, 2017; Karpathy, 2019).

Regarding regression problems, Softmax is not suitable as the output is not intended to be a class but a continuous number. Under these circumstances, Shimobaba et al. (2018) developed a DL model responsible for measuring the depth of objects where the loss function is calculated using the mean squared error (MSE) between the output of the network (d_o) and the ground truth depth value of the object (d_t). The MSE loss function can be defined as follows:

$$e = \frac{1}{N} \sum_{i=0}^{N-1} (d_o^{(i)} - d_t^{(i)})^2 \quad (2.7)$$

, where i designates the i -th entry in the dataset of size N . The intention with the error squaring is to prevent the cancelling effect between positive and negative values and, simultaneously, penalise large error on a heavier manner. Moreover, MSE has the important feature of being differentiable which allows the application of gradient based optimisation (see Section 2.3.4).

2.3.4 Optimisation Methodologies

When the forward pass is ended and the loss value found, then the backwards pass starts, aiming to alter the parameters in such a way that this loss function is reduced. In this section, the objective is to present several optimisers which can be distinguished in two main groups: fixed or changeable learning rate. The learning rate, η , is a hyperparameter of great importance as it determines how much the parameters vary on their update (Rosebrock, 2017). For that reason, it requires a careful tuning when using any type of fixed learning rate optimisation (Karpathy, 2019).

2.3.4.1 Gradient Based Optimisation

The impact of a given weight w_i in the loss function L is provided by a partial derivative. The *gradient* of L is a vector containing each partial derivative with respect to each of the variables, written as $\nabla L(W)$, where W constitutes the set composed by the model's parameters. Thereby, the gradient informs about the direction containing the steepest rate of increase. Then, it is possible to efficiently decrease L by assuming the opposite direction and set new parameters (Ruder, 2016); (Goodfellow et al., 2016). When considering $L(W)$ as a convex function, the main idea is to take steps on the steepest direction into the global minimum. However, for an ANN, most often, $L(W)$ as a non-convex function with a bumpy shape where the global minimum is rarely reached. Instead, the gradient descent finds a local minimum, which translates to a reasonable loss function value (Goodfellow et al., 2016; Karpathy, 2019; Rosebrock, 2017).

Ruder (2016) distinguishes three types of gradient descent variants, differing on how much training data is used on the gradient of the objective function computation. These are *Batch Gradient Descent* (BGD), *Stochastic Gradient Descent* (SGD) and *Mini-batch Gradient Descent* (MGD). The amount of data used constitutes, in fact, a trade-off between the accuracy of a parameter update and the time taken to perform this update.

On the one hand, BGD uses the whole dataset in order to calculate a single gradient and update a parameter. This means high levels of accuracy on the parameter estimation but, naturally, the process is very slow and completely unusable when dealing with datasets with millions of entries. On the other hand, SGD executes a parameter update for each training example. The frequent parameters updating translates on a much faster convergence than BGD but the high variance verified on each parameter update leads to heavy fluctuations of the loss function. On the bright side, the fluctuation allows to jump over the loss function and possibly to find a better local minima. By contrast, the fluctuation also means a more complicated convergence to the exact minimum (Ruder, 2016; Goodfellow et al., 2016).

The MGD is a mid term between BGD and SGD. Ruder (2016) identifies MGD as the best of the three methods and for that reason is way more used in DL problems.

2.3.4.2 Mini-batch Gradient Descent

MGD differs from SGD and BGD through computing the gradient over batches of data. Here, the parameters update is executed for every mini-batch of n training data entries (Ruder, 2016).

$$w_{t+1} = w_t - \eta \cdot \frac{\partial L(x_{i:i+n}, y_{i:i+n})}{\partial w_t} \quad (2.8)$$

This way, the BGD problems of time consuming computations are overtaken. Furthermore, the SGD struggles related to high fluctuations on the objective function are also attenuated. This method is so used many authors refer to MGD as SGD where it is assumed mini-batches are used (Karpathy, 2019; Lecun et al., 2015). From now on, also on this dissertation whenever talking about SGD it is assumed the mini-batch usage.

Regarding the size of the mini-batch, it is a hyperparameter to be tuned but, usually, powers of 2 are used since many vectorised operation implementations are faster when the inputs are power of 2 sized (Goodfellow et al., 2016; Karpathy, 2019). Nevertheless, there is evidence that larger batches lead to a deterioration of the model quality as it loses the ability to generalise, although the reason is still not known (Keskar et al., 2016).

2.3.4.3 Adam

Adam (Adaptive Moment estimation) is an adaptive learning rate method, meaning η is adjusted individually to the parameters, performing larger updates for parameters which are altered infrequently and smaller updates to frequently changing parameters. There are other adaptive learning rate method, such as Adagrad (Duchi et al., 2012) or RMSprop (unpublished but presented by Geoff Hinton in the slides of his Coursera Class), having Adam evolved from these. Introduced by Kingma and Ba (2014), the Adam optimiser, in addition to storing a moving average of past squared gradients, also maintains a moving average of past gradients, similarly to momentum

(Ruder, 2016; Rosebrock, 2017).

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) \cdot \frac{\partial L}{\partial w_t} \quad , \quad v_t = \beta_2 v_{t-1} + (1 - \beta_2) \cdot \frac{\partial L^2}{\partial w_t} \quad (2.9)$$

where m_t and v_t are estimations of the first and second moments (mean and variance) of the gradients. The creators of Adam initialise both as vectors of 0's and note these are biased towards zero, mainly across the initial steps. Therefore, m_t and v_t are usually bias-corrected through the expressions in 2.10 (Ruder, 2016).

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t} \quad , \quad \hat{v}_t = \frac{v_t}{1 - \beta_2^t} \quad (2.10)$$

, where Kingma and Ba suggest 0.9 for β_1 and 0.999 for β_2 . After this adjustment, the parameters update is given by the expression 2.11:

$$w_{t+1} = w_t - \frac{\eta}{\sqrt{\hat{v}_t} + \epsilon} \cdot \hat{m}_t \quad (2.11)$$

2.3.4.4 Decide on which optimiser to use

Interestingly, even though adaptive learning rates optimisers are available, many DL state-of-the-art models use SGD optimisation, which was introduced 60 years ago. According to Rosebrock (2017), this may be due to the fact that researchers are more familiar with SGD and prefer to perform better on the tuning of the hyperparameter η and have a slower converging model than the opposite. Yet, regardless of the chosen optimiser, the gradient is usually calculated resorting to mini-batches of the data instead of using all training examples.

Ruder (2016) states that if the amount of input data is not vast, adaptive learning rates methods are likely to provide better results. Furthermore, there is no need to tune the learning rate as the best results are usually able to be obtained with the default value. Among these advanced methods, Adam may be the best choice (Ruder, 2016).

2.3.4.5 The role of the chain rule

Among the optimisation methods previously presented, irrespective of it being an adaptive learning rate method or not, they all have the partial derivative in common, which explains the loss function value variation with regards to a given parameter. In fact, in practical terms, the back-propagation is not more than a practical application of the chain rule in order to determine the derivatives where the impact of a given parameter on the final loss is explained. The backpropagation equation is applied repeatedly to propagate gradients through all units, starting from the output until the input layers (Rosebrock, 2017). In order to understand the chain rule functioning in a CNN, see Figure 2.3 where a kernel K (composed by the changeable parameters) performs a set of convolutions with an input matrix P in f and, naturally, the output is the activation map S . The goal is to understand how the parameter values impact the loss function L .



Figure 2.3: Simplistic visualisation of all the convolutions between a kernel K and the input matrix P , originating the activation map S

The impact of K on L is therefore given by $\frac{\partial L}{\partial K}$. However, K is not a single value but a matrix composed by several changeable values. Therefore, in order to find the impact of each of these values, then the chain rule may be applied as seen in expression (2.12).

$$\frac{\partial L}{\partial K_i} = \sum_{j=1} \frac{\partial L}{\partial S_j} \cdot \frac{\partial S_j}{\partial K_i} \quad (2.12)$$

where S_j represents each value of S and K_i is each element belonging to the kernel.

2.3.5 Other Features of Convolutional Neural Networks

2.3.5.1 Fully Connected Layers

A fully connected layer (FCL) is the most usual type of layer in traditional ANN, where neurons between two adjacent layers are fully pairwise connected. However, no neuron shares connections within a single layer (Karpathy, 2019). Inside most CNN architectures, these layers are usually placed after all the convolutional processes are ended, meaning it is not common at all to apply a CL, a FCL and then again a CL. In a CNN, FCLs play a very important role as they take the output from and turn it into the desired CNN output (Rosebrock, 2017).

2.3.5.2 Pooling Layers

Pooling layers are used to alter a CL after the activation function. A pooling function replaces the output of a convolutional layer plus activation function with a given summary statistic of the nearby outputs. The intention with pooling is to turn the representation invariant to small translations of the input, turning the model into a more robust one. In fact, invariance to local translation is useful when it is more important to know whether some feature is present than to know its exact location (Goodfellow et al., 2016). Through pooling, the amount of parameters is naturally reduced, alleviating the computation heaviness. Moreover, it also helps to reduce overfitting (Rosebrock, 2017). The max pooling, where the maximum output within a rectangular neighbourhood is obtained, is the most common operation and visualised in Figure 2.4.

Other common pooling functions are the average pooling, the L^2 norm of a rectangular neighbourhood or a weighted average based on the distance from the central pixel (Goodfellow et al., 2016). Currently, some works disregard the pooling operation. In fact, Springenberg et al. (2015) developed a simplistic approach to CNNs, proposing the reduction of the size only using larger values for stride.

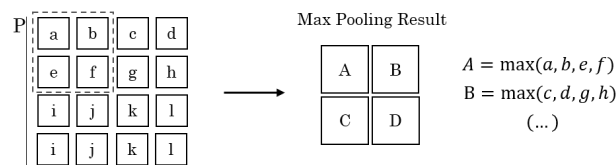


Figure 2.4: Illustration of the max pooling operation.

2.3.5.3 Dropout

Dropout consists in a procedure aiming to prevent overfitting, which was introduced by Srivastava et al. (2014). In practice, the key idea is to randomly drop units (and the corresponding connections) from the ANN during training with some probability p . The connection is only disregarded during one epoch, this means that after a forward and backward pass the layers are re-connected (Srivastava et al., 2014). Overfitting is reduced as the network architecture is being randomly changed during training time, allowing an efficient combination of many distinct ANN architectures Srivastava et al. (2014).

2.3.6 Building a CNN

Finally, the understanding of how all the presented components interact with each other on a network is fundamental. In fact Karpathy (2019) presents an overview on CNNs architectures. The most usual shape of a ConvNet architecture is composed by blocks of a convolutional layer followed up by a non-linear activation. In between each of these blocks, there is generally a pooling layer. The last layer is commonly fully-connected, holding the output. Therefore, the common CNN architecture has the structure presented in Figure 2.5, proposed by Karpathy (2019).

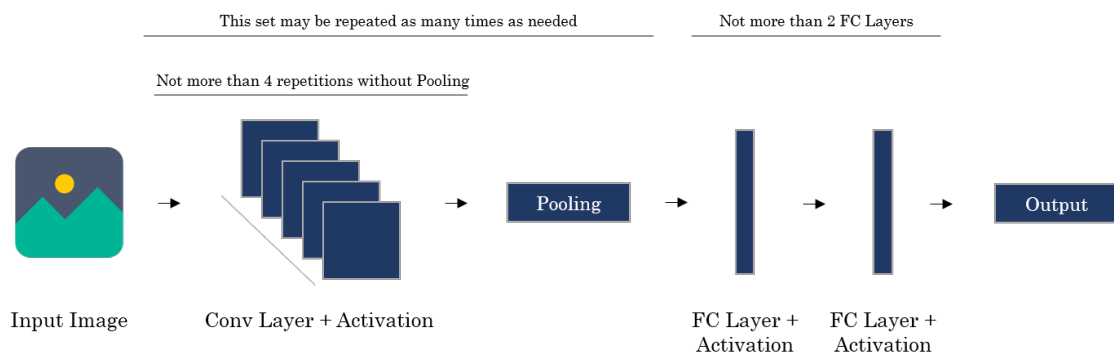


Figure 2.5: Schema showing the commonly used CNN architecture patterns.

Some layer sizing patterns may be found in CNNs. For instance, the input layer size has to be usually divisible by 2 by a considerable amount of times. This number goes from 32, as in Kolesnikov et al. (2019), to 128 (Shimobaba et al., 2018) or 224, present in the Alexnet (Krizhevsky et al., 2012), or an even larger number as 512.

Regarding CLs, the kernels are generally small (3x3 or 5x5) and used with stride equal to 1. However, the presence of larger ones (like 7x7 or 11x11) is not uncommon on the very first

convolutional layer of the CNN (as in Krizhevsky et al. (2012)). The execution of stride 1 complements the down-sampling performed by pooling layers, with the CL therefore only transforming the input-volume depth-wise. Concerning pooling layers, again, max pooling is the most common procedure, usually acting on 2x2 receptive fields (Karpathy, 2019).

2.3.7 Prevent overfitting through training monitoring

The objective of resorting to an ANN, either a CNN or not, is to obtain a model which can perform well when evaluating unseen data. In other words, the model should be as general as possible. The generalisation error may be estimated by the average error obtained on a *validation set*, which is an independent group of data not belonging to the data used in training - *training set*. The validation error is usually named *validation loss* and its control during training is very useful in the process of verifying if the model is overfitting (Prechelt, 2012; Rosebrock, 2017). A possible behaviour from the validation loss is to first show a decreasing trend converging to a minimum value, but then increase after that, as the model starts to overfit data as seen in Figure 2.6 (Prechelt, 2012).

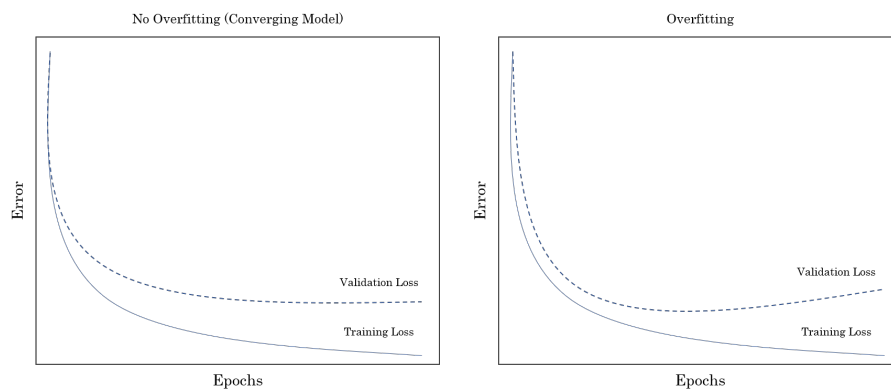


Figure 2.6: Left: Validation and Training loss for a converging model. Right: Validation and Training Loss for an overfitting model.

On the other hand, if the validation loss initially decreases to a given error value and develops on a nearly constant and stabilised manner through time (left image in Figure 2.6), this means the model converged and no signs of overfitting are evident.

2.4 Connecting the Literature and the Project

From this point, the intention during the project is to take advantage from the topics explored in this chapter. In fact, from Section 2.2, the idea that a CV problem may be approached either through a traditional basis, a DL procedure or a hybrid configuration serves as basis for the work developed. Then, the results of the models are compared to choose the most appropriate model to be adopted at HUUB. Yet, prior to the models development and comparison, the next chapter presents a detailed explanation of the problem addressed in this thesis and its relevance for the company.

Chapter 3

Problem Framework and Description

This chapter presents the problematic to be tackled. Section 3.1 focuses on the identification of how the full sequence of processes is currently executed at HUUB and describes the entities involved in each process. Section 3.2 describes the aims of the project where the TO-BE scenario is presented in terms of the process improvement. In this part, the focus is on the modified processes with respect to the AS-IS situation by describing the new methodological approaches and their differences in relation to the current situation. Finally, the chapter ends by stating what HUUB can expect from the new procedure.

3.1 Operation AS-IS

Figure 3.1 illustrates the map of processes associated with the measurement of items. In summary, there are two main active teams involved in the whole process: the Operations team and the AI team. There is a specific output for each team, corresponding to the package with the set of items selected in an order and the transportation cost estimation. The full operation is explained in more detail in the next sections.

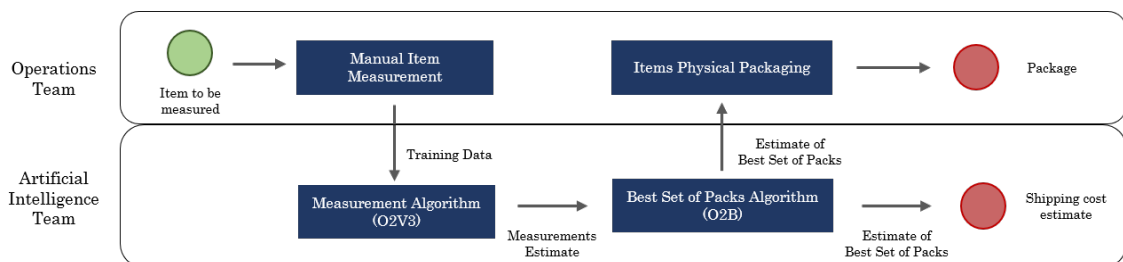


Figure 3.1: Current map of processes.

3.1.1 Teams Involved

3.1.1.1 Operations Team

The Operations team is in charge of the warehouse related tasks. The team is responsible for very wide range of jobs as, for instance, receiving items from brands suppliers, doing the picking

for orders or executing the physical packaging. In the current state-of-affairs, the team is also responsible for weighing and measuring the depth, width and height of a sample of items on a regular basis, to feed the AI team models.

3.1.1.2 Artificial Intelligence Team

The AI team plays the role of gathering, processing and thoroughly analysing the data generated inside HUUB, through analytical procedures. In fact, the problems explored by the AI team cover the three areas of business analytics: predictive, descriptive and prescriptive analysis. This team work's revolves around the goals of automatising and optimising several process within the company, resorting to the development of algorithms and machine learning models.

3.1.2 The processes inside the AS-IS situation

3.1.2.1 Item to be measured

Fashion items are the input of the whole operation. HUUB's database comprises information about numerous features of each item. Such features include the fashion brand it belongs to, the supplier who manufactured it, the type of product (e.g. Clothing, Footwear, Homewear), the product family (e.g. Top, Bottom, Shoes), the product subfamily (e.g. T-shirt, Trousers, Flat sandals), model size (e.g. S, M, L), age group (e.g. Baby, Kid, Adult), gender (e.g. Female, Male, Unisex), the type of fabric of the product, and dozens of other specifications. Each product model is identified by a unique number, the *product model id*. The items features are generally informed by the brand and provide great details about the product models. Some features which are not known, however, include the weight, height, depth, width and volume of an item. Since these features in particular are required to any estimate of the size an item takes in a pack, the need to acquire them led to the process of manual measurements and weighing of items to be implemented within Operations.

3.1.2.2 Manual Item Measurement

Currently, the Operations team is responsible for gathering the four measurements of an item through the procedure represented in Figure 3.2, where the mean time per task is also explicit.

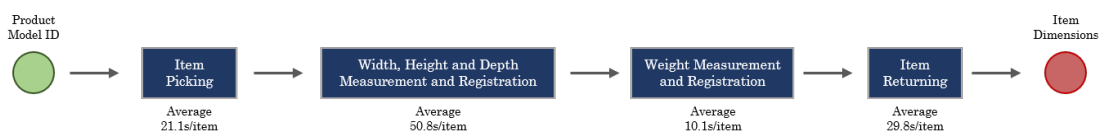


Figure 3.2: Current map of processes performed for an item measurement.

The team has a list of items to be measured, identified by the product model ID and its location in the warehouse. The operator starts by picking all the items in the list that are stored in a given warehouse aisle. Afterwards, these are brought to a station where each item goes through a process where its width, height and depth are obtained resorting to a tape measure. After each

measurement, the value observed is registered in the database. The item is then weighted using a weighing scale, and the value is registered. Finally, the whole batch is then returned to the corresponding warehouse location.

3.1.2.3 O2V3 - a Measurement Prediction Model

HUUB processes thousands of items and it is not possible to know the measurements and weights of each of them. Hence, an AI model - called O2V3 - was developed, and is already in production, to predict weights and measurements of items. It is a decision-tree-based ML ensemble model built with a XGBoost implementation which takes as input some item features present in the database (product subfamily, model size, gender and age group) and, basing on that, predicts its dimensions. This algorithm needs to be trained with real data, therefore making use of the measurements provided by the manual item measurement stage.

3.1.2.4 Order to Box (O2B) - a Packs Prediction Model

The O2B is live at HUUB to estimate the best set of packs for a given set of items. This is used to estimate packs both for deliveries and for products transportation in between warehouses. O2B returns packs predictions based on the items measurements and weight, which are themselves predicted by O2V3. O2B is a ML model which uses heuristics and meta-heuristics to perform an optimisation of packs, while taking into account all the packaging restrictions and preferences from brands (e.g. Specific boxes/flyers for each brand are to be used for B2C and others for B2B). It selects its best solution by minimising the total number of packs, as well as their total volume. This model is on the basis of the cost estimation for any transportation carried out by HUUB, and thus for the transportation price which is charged to brands. For this reason, it is ultimately important to have accurate items measurements and weights feeding this model.

3.1.2.5 Transportation Cost Estimate

The transportation cost estimate is a vital point in HUUB's business model. As explained in Chapter 1, a partner brand is charged per each transportation with a cost determined as the sales order is placed (and thereby before the transportation actually happens). The point here is that the estimated value has to be as accurate as possible: on the one hand, if this price is too low then HUUB will incur on a loss, on the other hand, if too high, then the partner brand may argue that it would be more beneficial for them to directly work with the carrier and loose confidence in HUUB's reliability.

For each package provided by the O2B solution, the transportation cost is estimated. The transportation cost for a shipping volume, which is a package with items inside, is charged by carriers to whichever is the maximum: its weight or its volumetric weight.

If the package is a rigid cardboard box, its volume is fixed and known beforehand. In contrast, if items are packed in a flyer, the volume of the flyer is variable, according to the items inside. Regarding the weight of a shipping volume, this is the sum of the weight of all items, plus the

weight of all packs. In the end, the transportation cost for an order is the aggregate of the cost estimated for each of its packs.

3.1.2.6 Items Physical Packaging

Back to the Operations Team, this process consists in the actual packaging inside the warehouse. This stage receives the input from the O2B model, indicating the theoretic best set of packs to use. From this process comes out the other output of the operation which is the package itself.

3.1.3 Main Problems of the AS-IS situation

The current problems are related with the items' dimensions predictions in the O2V3. The model lacks the necessary amount of high quality training data to have a good performance which is critical for the subsequent stages, leading an erroneous transportation cost estimate. The O2V3 problems regarding the real measurements come from two points:

- The manual measurement of items is a slow-moving task, leading to training data scarcity. At the time of writing, less than 2.000 items have their dimensions catalogued in the database which is critically low as HUUB is currently transacting over 70.000 (due to the ongoing company's growth, this number is even expected to increase);
- Human error is present. Often, real measurements come with evident human error (which tends to be by excess) due to wrong interpretations of the measuring tape or mistakes during the data registering.

Due to these problems, O2V3 wrongly predicts the item dimensions and, consequently, O2B will most certainly indicate an unnecessary quantity of packs to be used which, in turn, mean that transportation costs end up being overpriced. Therefore, Operations cannot fully rely on the set of packs provided from the O2B as they usually can notice that fewer packs are needed in reality. Hence, the team executes a more efficient wrapping, but not necessarily optimal.

Finally, in the AS-IS situation, there is another problem not yet referred. After the packaging, when items are put in flyers (instead of rigid boxes) the volume varies from flyer to flyer. Currently, as the O2V3 still estimates items' dimensions with a considerable error, HUUB struggles to estimate how much it has to pay to carriers for each shipping. Often, the real price of transportation involving flyers is only known when the carrier invoice is received, which means less control over budget.

3.2 Operation TO-BE

The TO-BE situation proposed, after the implementation of this project, directly concerns, first and foremost, the specific task of manual item's width, depth and height measurement. Furthermore, the model also intends to tackle the problem of not being able to estimate the price to be paid to carriers in case flyer are used. The TO-BE situation is proposed in Figure 3.3.

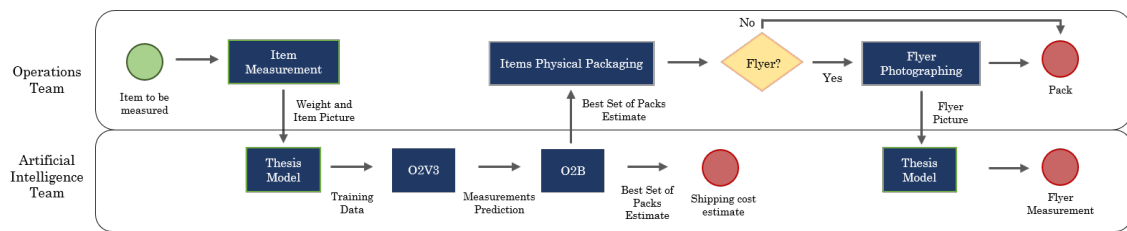


Figure 3.3: Map of processes after the project implementation. The processes with a green outline are directly impacted.

3.2.1 What changes and what to expect

The process of an item's width, depth and height measurement is no longer executed through human hand. Instead, a picture of the item is captured and passed to the model developed in this thesis, which provides O2V3 with training data. The photographing operation includes capturing the picture itself and immediately save the picture with the item's ID, which is done resorting to a simple copy and paste operation. The item's measurement full procedure proposed is present in Figure 3.4.



Figure 3.4: With this project, the manual item's width, depth and height measuring is replaced by a picture.

By joining the integration of the item's photographing process with the measuring model developed in this work, HUUB can expect two outcomes: (i) the measurement procedure to be much faster and therefore the O2V3 to receive training data at a larger rate; (ii) the human error to be eliminated from the measuring process. Moreover, the company can also look forward to have reliable estimates of the flyers volume before shipping.

3.2.2 Item photographing

Regarding the picture capture, a special setting as well as guidelines were produced in order to ensure optimal model performance. Figure 3.5 represents an example of a picture correctly taken and accepted as data for all the developed CV models.

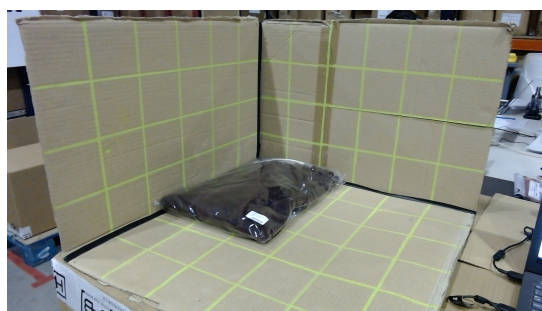


Figure 3.5: Example of an item's picture correctly taken according to photographing guidelines.

3.2.2.1 The cardboard box

The visible cardboard box in Figure 3.5 is used as a frame for the item. In there, the black stripes are considered as the axis of the box. These are of major importance as they represent the direction of each dimension to be estimated. The width estimate is measured in the most left axis, whereas the depth estimate is defined by the most right axis. Naturally, the height is read in the vertical one. The intersection of the three stripes provides the box origin. From this point, the width, depth and height axis are referred as "x-axis", "y-axis" and "z-axis", respectively.

Furthermore, the box also has a green grid with 10cm wide squares. The grid acts as a reference scale, which enables the actual measurement of items' dimensions and also allows the correction of perspective deformations. In fact, in Figure 3.5, one can notice that 10cm in a square close to the box origin seem much smaller than 10cm in the cardboard box border, even though they are the same 10cm in real life. The presence of the grid allows to figure out how this deformation varies within an image.

3.2.2.2 Main guidelines

Inside the warehouse, there is no specific location for the picture to be captured and no tripod is imposed for the camera. The box could be taken anywhere as long as the light allows clear visibility of both the grid and the item. The idea is to be as flexible and easy as possible concerning the photographing process in order to disrupt Operations the least possible.

There are, however, some rules regarding the picture:

- The camera has to be diagonally oriented in relation to the box origin. This positioning allows to catch both vertical plans of the box in a similar proportion.
- The item should be as close as possible to the box corner (box origin).
- The three axes must be captured in their full length.
- Some common sense rules are important, such as the picture should not be blurred.

All these rules are explicit in an One-Point Lesson (OPL) which was formulated and passed to the Operations team. The OPL can be consulted in the Appendix A.

3.2.3 Next steps

This chapter intended to explain where the project aims to impact and the changes in the measurement procedure, namely substituting the manual tape measurement of items by the capture of an image with a camera. The picture is thereafter processed by the model developed in this project and the item's dimensions automatically inserted in HUUB's database. The next chapter describes the three measuring models developed, as well as their functioning and principles.

Chapter 4

Models Outline and Results

This chapter describes the framework adopted concerning the models' development. Firstly, the dataset is succinctly presented. Then, the models developed are presented, with a careful description of their main features. The first model is a CV algorithm built from scratch, followed by a model based on DL techniques and, finally, a model with a hybrid design. The chapter ends with the direct comparison of the models with respect to the prediction task.

4.1 Dataset Outline

At the time of writing, the dataset is comprised by a total of 1224 pictures, each of a distinct item. The amount of pictures was previously larger than 1224. However, following the computer science known motto "garbage in, garbage out", these were subject to a quality control with respect to: (i) the picture accordance with the photographing guidelines exposed in Appendix A; (ii) the nonexistence of evident human error in the real measurements.

Regarding the items captured, the distribution of their real dimensions is visible in Figure 4.1.

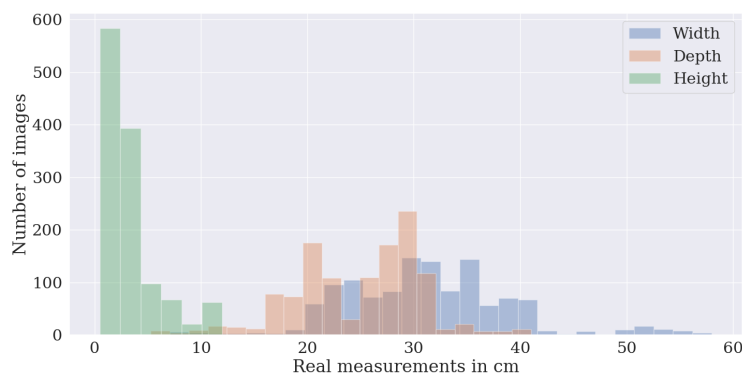


Figure 4.1: Distribution of the items' dimensions in the dataset.

Clearly, the height takes lower values than the width and depth. The height values are usually smaller than 10cm. On the other hand, the range of values of the depth and width is more alike.

Nevertheless, the width values tend to be larger than depth, which was on purpose. In fact, the dataset was built in a way such that the larger side of the base of an item was the width. The dataset was in first place intended to train the O2V3, a model which benefits from this pattern to distinguish the depth and width. However, this is not relevant for the present work, as the width is always read in the axis on the left, regardless of being the largest side or not.

4.2 Model I - CV Algorithm

Model I provides an algorithmic approach to the problem. It consists of four main steps which culminate with an estimate for the depth, width and height. However, this is not a pure traditional CV algorithm as it uses one simple CNN in order to calibrate the model in regards to the lighting conditions of the picture, enhancing the model performance. Importantly, the model requires the visualisation of a minimum part of all three axis. The threshold was found to be close 10cm and, therefore, the model cannot predict items' dimensions beyond 50cm, 50cm and 30cm for the depth, width and height, respectively.

4.2.1 Axis and grid detection

The first step is to detect the black stripes and the green grid of the box. This is done by generating a *mask* for each, which are maps indicating the presence of the grid or axis. As the three axis and the grid are mainly composed by straight lines, an appropriated implemented practice is to apply image sharpening as it enhances these features. Figure 4.2 represents an inputted image that is then sharpened which, in turn, generates two masks.

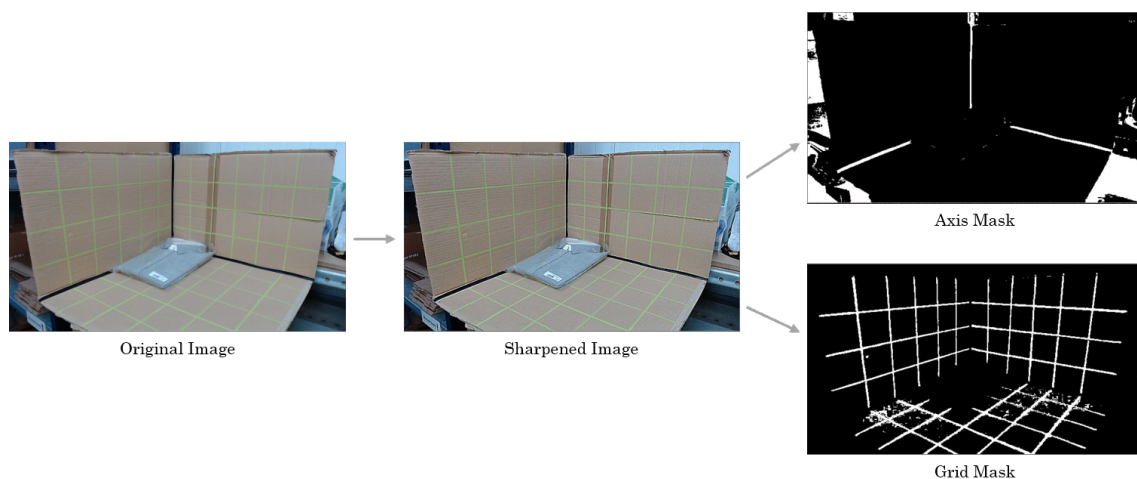


Figure 4.2: Schema of the first step of the model I.

However, the process of finding the axis mask is distinct from the grid mask. This is due to how the corresponding colours vary from image to image regarding the lighting conditions. In fact, black colour varies much less than the green, which can get too dark and too bright within the same picture.

4.2.1.1 Axis mask generation

The concept to find the axis mask is based on the formulation of colour selection rules which are simply restrictions for the RGB values of a given pixel. For instance, a colour selection rule for an axis detection is $\text{Red} \leq 50$, $\text{Blue} \leq 50$ and $\text{Green} \leq 50$.

However, the impact of different lighting conditions is felt in the axis RGB colour codes due to the presence of shadows. A careful observation of the pictures allows to split them in two types, as seen in Figure 4.3.

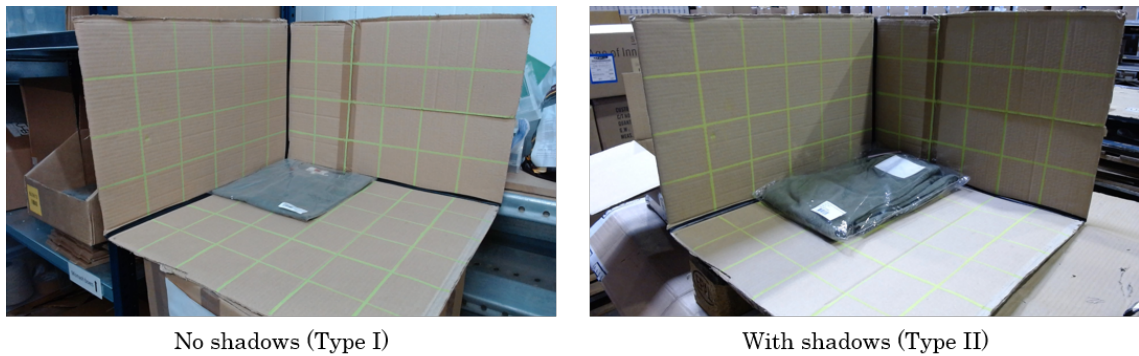


Figure 4.3: Pictures are distinguished according to the presence or absence of shadows.

The pictures separation in two types allows to adapt the colour selection rules to each group of images. In order to make the algorithm distinguish pictures, a very simple CNN model is implemented. This procedure The CNN has the architecture presented in Figure 4.4.

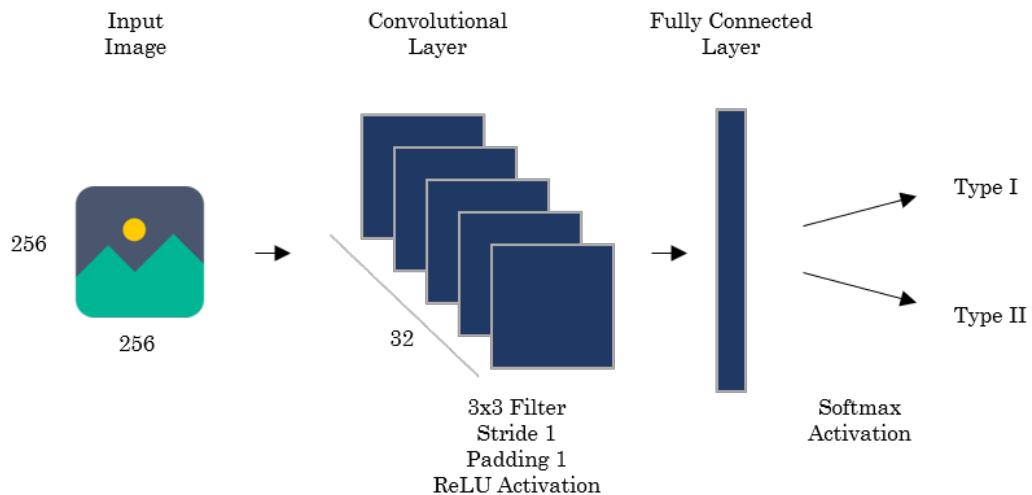


Figure 4.4: Architecture of the CNN responsible for distinguishing between pictures with and without shadows.

This simple model uses only one CL after the input image and is immediately followed by a FCL which, through a Softmax activation, outputs a probability for each type. Regarding the training itself, 987 images were used from which 505 were considered to contain shadows, whereas

482 were not. The optimiser used was the SGD with a learning rate equal to 0.001 and mini-batches of size 32. The training was performed during 15 epochs and a validation set of 25%. The training evolution is described in Figure 4.5.

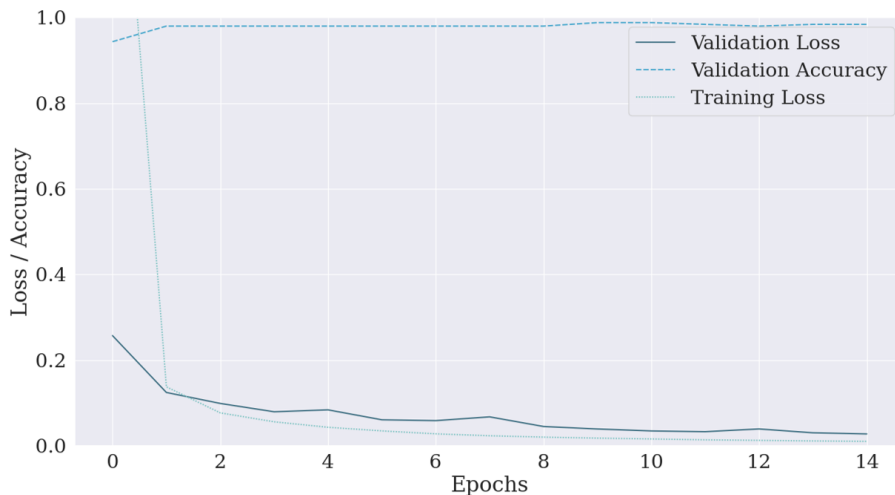


Figure 4.5: Training evolution of the CNN responsible for distinguishing between pictures with and without shadows.

Figure 4.5 shows the validation loss and training loss evolution with epochs. The validation loss decreases together with the training loss, as expected. No signs of overfitting are observed, as there is no increase of validation loss but a stabilisation instead. Regarding the performance, the model was tested in the same 25% pictures from the validation set (since the model does not use them for training), reaching an accuracy of 98.34%. The reason why the accuracy value is so high even though the dataset is small and the CNN is simple may be related with the high level of similarity of the pictures within groups. Whereas Type I pictures have the cardboard with an approximate constant colour, the Type II does not, which eases the distinction and leads to high accuracy levels.

After the identification of the picture type, the correct group of colour selection rules are applied to an image and the axis mask is obtained. The process of identifying the correct selection rules was empirical, meaning that a large number of collected pictures were analysed regarding the RGB of pixels belonging to the axis. In Table 4.1, two rules for each type are explicit. Any pixels complying with the restrictions then will belong to the mask.

Table 4.1: Two of the colour restrictions imposed to the pictures of each type.

Colour Selection Rules	
Type I images	Type II images
Red \leq 40 and Green \leq 40 and Blue \leq 40	Red \leq 25 and Green \leq 25 and Blue \leq 25
40 \leq Blue \leq 60 and 10 \leq Blue - Green \leq 40 and Green \leq Red + 30	25 \leq Blue \leq 55 and 3 \leq Blue - Green \leq 70 and 6 \leq Green - Red \leq 30

4.2.1.2 Grid mask generation

In Figure 4.3, in the type II picture, one can notice the grid is very dark around the vertical axis, while, in the same picture, the opposite happens in the box borders closer to the camera, where the grid is extremely bright. Due to these cases, it becomes unbearable to build a mask to every distinct types of grid colours with colour selection rules, as these rules would become too wide-ranging leading to the selection of much more than the grid. Another method is needed.

In order to identify patterns in the RGB colours of an image that allow to detect the grid regardless of the lighting conditions, an empirical scrutiny of pixels colours was done. The analysis inferences are: (i) pixels belonging to the cardboard box never contain large differences among its RGB components and, therefore, the standard deviation of its RGB components is small; (ii) in comparison, grid pixels have larger variations within their RGB components and typically the green component is usually greater than red or blue. Taking these insights into account, the grid pixels standard deviation is compared to the remaining image pixels of the image that contain green as the maximum RGB component in Figure 4.6.

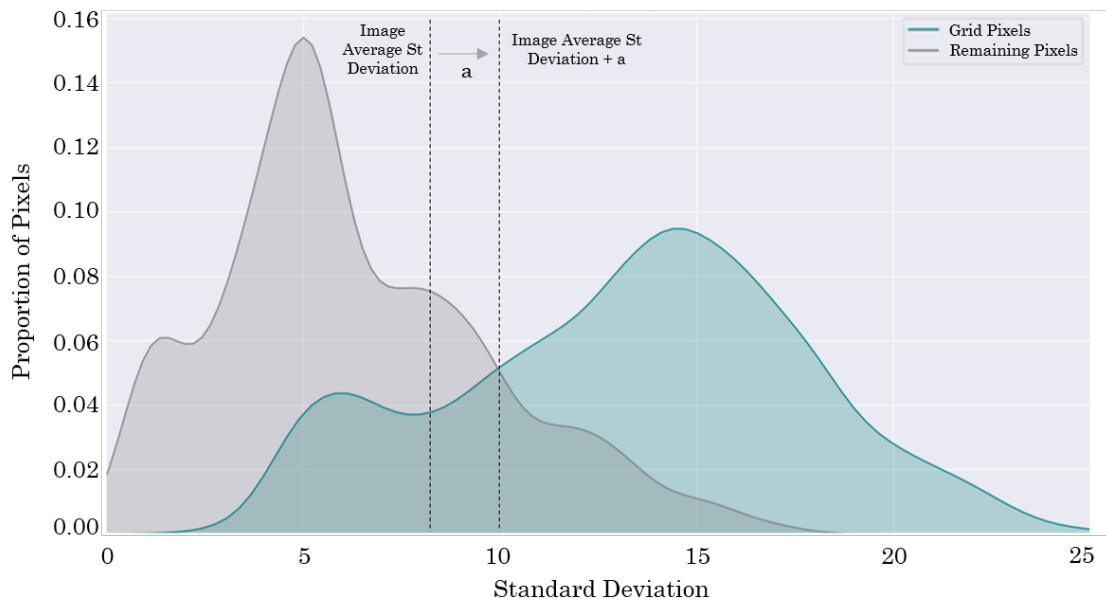


Figure 4.6: The grid pixels usually have larger standard deviations against the remaining pixels whose maximum RGB component is the green one. The dashed vertical line "image average standard deviation" represents the average standard deviation of the whole image. The picture used for the graph generation is the type II picture from Figure 4.3 and this graph has a similar shape for the remaining pictures.

Clearly, there is a mismatch between graphs. This can be very convenient to build a robust grid mask as the grid pixels have a standard deviation within a threshold delimited by the image average plus a constant a . The value of a was determined empirically and defined to -1 if the image is of Type I and 2 if Type II.

Therefore, the selection rule for the grid mask generation is: if a pixel has its standard deviation larger than the image average standard deviation plus a constant a and its maximum RGB

component is the green one, then it belongs to the grid. This methodology allowed to clearly identify the grid, as seen in Figure 4.7.

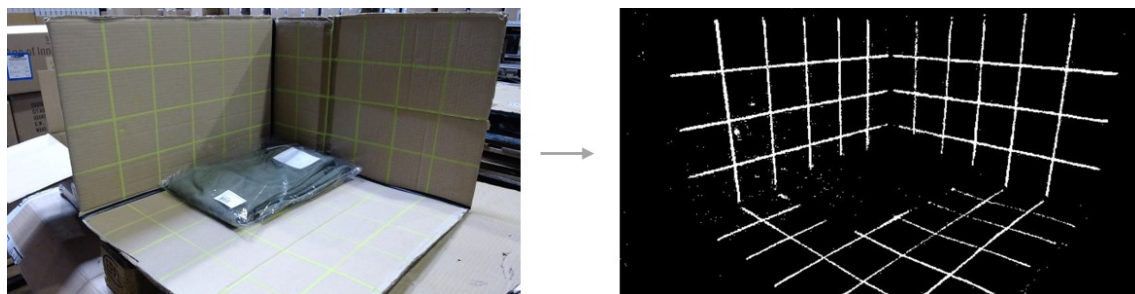


Figure 4.7: Grid mask obtained.

4.2.2 Drawing lines for axis and grid

Once both masks are obtained, the next step is to fit lines to all three axis and the grids. This procedure approximates each line to a simple line equation $y = mx + b$ within an image, which is a 2D Cartesian coordinate system composed by integer points only. Briefly, for this task, the algorithm "walks" on each line in the mask in 3x3 squares, collects points and then, resorting to linear regression techniques, approximates it to a line equation. This procedure is further detailed below.

4.2.2.1 Functioning Principle

In this explanation, consider an axis mask and goal of finding the x-axis, which is represented by the most left black stripe. The algorithm has specific locations in the image where it starts its search for each of the axis. For the x-axis, the algorithm starts looking for it on the image bottom left corner, as seen in Figure 4.8. For the y-axis, the starting location would be the bottom right corner and for the z-axis the top right region of the image.



Figure 4.8: The algorithm "walks" inside a mask in 3x3 squares looking for lines identification.

As illustrated in Figure 4.8, the walk begins on the gray square, which is evaluated regarding the presence of 1s. The square under evaluation is named as *active square* and its location is given

by the pivot, which is the top left pixel of the square. If only 0s are present, the active square moves three pixels upwards, to the position of the blue square. At this stage, there are three 1s inside the active square and therefore the algorithm enters the searching mode, meaning it will have to find out if this is an axis or polluting points from some dark object in the background.

When in searching mode, the algorithm no longer necessarily takes vertical steps but, instead, performs an evaluation of 3x3 squares in the active square neighbourhood - there is a total of 24 different 3x3 neighbour squares surrounding an active square if this is not located in the image border - and selects the one with more 1s. The movement is, however, restricted to some directions (explained in more detail in Section 4.2.2.2). The process is repeated as represented by the dashed squares in Figure 4.8, while every square is saved through the coordinates of the corresponding pivot. The walk finishes when the algorithm cannot execute any valid move to squares with at least one 1.

As soon as the walk finishes, if the euclidean norm between the first square and the last one is above a given threshold, i.e., if the length of the potential axis is longer than a threshold, it is considered to be an axis. In order to fit the axis to a straight line, the algorithm takes every squares' pivots coordinates and approximates it to a line equation through a linear regression. However, if the euclidean norm is below the minimum threshold, the algorithm goes back to the blue square and, again, changes the active square to the one above and the procedure is repeated until an axis is found or the image is ended. Due to the obligation of meeting a threshold, Model I cannot be used with items so large that cover the axis on their almost full extent. In practical terms, items with more than 50cm of depth/width or more than 30cm of height, cannot be measured.

When the algorithm is looking for grid lines, the procedure does not stop as soon a line is found but only when all grid lines are found. Figure 4.9 shows the algorithm walk each grid line and, afterwards, the line fitting which enables the drawing of straight lines.

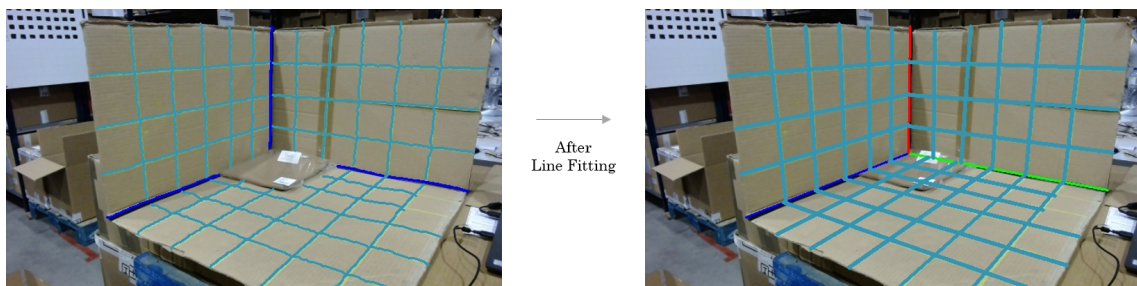


Figure 4.9: The image on the left shows the walking path performed by the algorithm before the line fitting turns each path into straight lines, which is seen in the image on the right.

4.2.2.2 Applied Restrictions

The directions in which the algorithm can walk have restrictions, which fully depend on the line the algorithm is looking for. Each line belongs to one of three groups of orientations and the algorithm looks for each group at a time. The first group is comprised by the width axis and parallel grid lines, where the walking direction imposes a movement to the right. The second group is the

depth axis and the parallel grid lines, where the walking direction requires a movement to the left. Finally, the third set is made up of vertical grid lines and the z-axis where, naturally, the walk has to occur downwards. Nevertheless, these restrictions are only applied during the first 5 steps, originating the *direction learning process*. At this stage, the algorithm selects the favourite movement (the most common) and, beyond 5 steps, the algorithm is only allowed to move in this favourite direction or in slight variations of it. These restrictions have the role of making the walk occur only in straight lines, which is fundamental for the well functioning of the algorithm for several factors. The obvious one is that, without restrictions, the algorithm could easily find a way of entering an infinite circular walk. Another one is that the algorithm would diverge from the lines due to its greediness. This is true specially in grid lines where, in intersection points, the algorithm could even change for an orthogonal line.

Furthermore, another issue is the frequent existence of dark zones behind the cardboard box, as those visible in the axis mask present in Figure 4.2. When searching an axis, such areas may lead the algorithm to walk on them, instead of the actual black stripes. To overcome the problem, two more restrictions were implemented.

Firstly, the algorithm is set to only walk over the axis mask if it can grasp some grid point in the active square surroundings. Such constraint forces the walk to be done inside the cardboard box.

Secondly, the black stripes representing the axis are narrow, which means the active square surroundings can be analysed in order to know if the walking is happening inside a region with such features. Therefore, an upper bound was set for the amount of black points inside the local region and, if the threshold is exceeded, the walking stops. Moreover, this second constraint also avoids the walk to happen over items whose colour is similar to the axis' colour.

4.2.3 Scale Calibration

The scale calibration is a stage in which a relationship is found between the pixels on the axis on the image to cm in real life.

Consider the yellow points present in Figure 4.10 which identify the intersection points between grid lines and axis. To calculate the intersection points, the algorithm uses each line equation obtained from the previous step. Considering the yellow points are equally spaced by 10cm, the euclidean distance from the known box origin to any of these points (in pixels) provides conversion values for each dimension, allowing to build a discrete scale for each axis. For instance, in both depth and width axis, the algorithm knows how many pixels does it take to get from the origin to 10, 20 and so on until 60cm. The same for the height axis but only up to 40cm. Importantly, the distance from 0 to 10cm in pixels is much smaller than the distance from 50 to 60cm, due to perspective deformations and such distances vary from axis to axis because of the box orientation in relation to the camera.

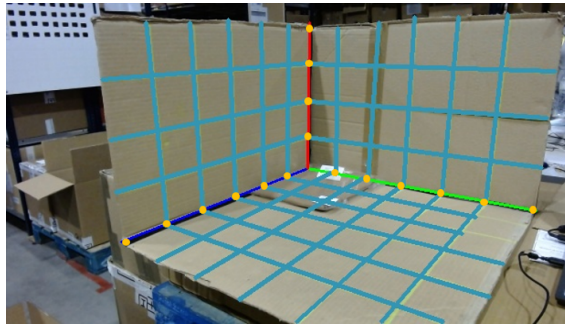


Figure 4.10: The intersection points are represented by the yellow dots. Their spatial positioning is crucial to build a conversion from pixels in the image to cm in real life.

To build a continuous scale, discrete points are fitted into a continuous line. Again, the regression techniques are used. Nonetheless, a linear regression was found not to be the most suitable method for this situation, given that the distance between points is not linear due to the perspective deformation. Thus, the fitting was performed through a quadratic regression, as seen in Figure 4.11 with a correlation coefficient almost equal to 1.

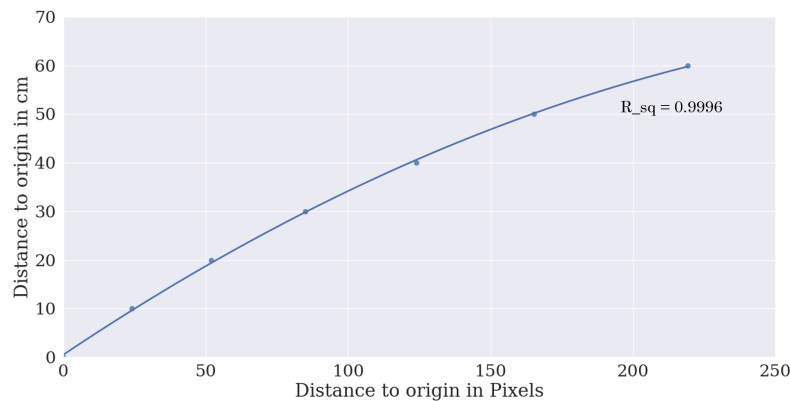


Figure 4.11: Example of a quadratic regression matching the distance on the image in pixels to real distance in cm for the x-axis of the picture present in Figure 4.10.

The fitting allows to find a relationship of the type $d_{cm} = a \cdot d_{px}^2 + b \cdot d_{px} + c$ (where a, b, c are constants and d corresponds to distance) for each axis. Therefore, the output from the current stage is comprised by three quadratic equations.

4.2.4 Items' dimensions estimation

Finally, this stage is responsible for outputting the item's depth, width and height. Considering Figure 4.12a, the red and the green lines are used to find the width and depth of the item, whose euclidean norms provide accurate estimates in pixels. The chosen line for the measurement is the one with the largest euclidean norm. The reason for this is to make sure that when an item has an irregular shape, i.e. not parallelepiped, the largest side width is the one considered (see example in Figure 4.12a). On the other hand, the height is directly measured through the red vertical line as empirical work have shown it is the best approach. As the distances to use are

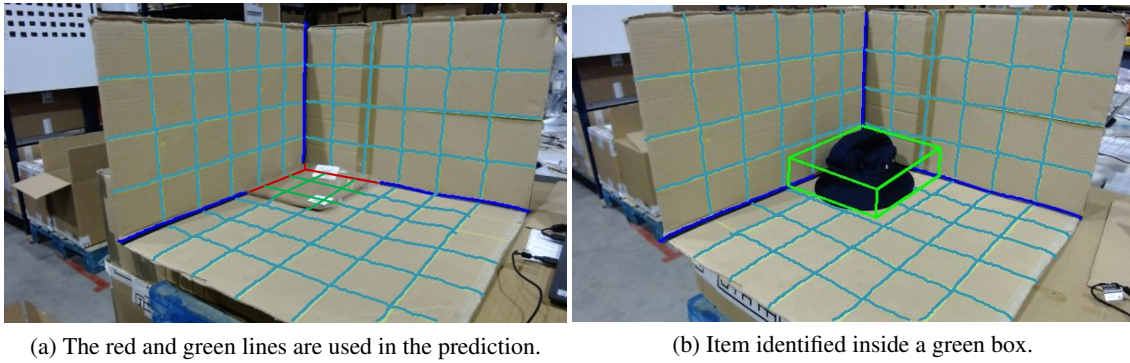


Figure 4.12: The red and green lines from the image on the left allow the box building.

chosen, the final step consists of entering the pixel values in the quadratic equations acquired in the previous stage, which allows obtaining the measurement in cm. Finally, the algorithm outputs, for visual inspection, an image with the item identified inside a green box of width, depth and height matching the item measurements predictions (see Figure 4.12b).

4.3 Model II - DL model using a CNN

The Model II solves the items' dimensions estimation problem resorting to a CNN. It is, thus, a DL regression problem. The model description starts by addressing the network architecture and, afterwards, the training operation.

4.3.1 The Architecture

Consider the CNN architecture of Figure 4.13, where the input is an image of size 128x128 pixels with three colour channels RGB. The network can be divided in two sections, the first is comprised by four CLs and the second involves two FCLs. There are several reasons underlying the specification of the network with this particular shape. First, it follows the suggested CNN architecture patterns in the literature described in Chapter 2. Second, the network configuration in Figure 4.13 is inspired in CNN architectures used in similar work, such as Shimobaba et al. (2018). Third, the computational needs for the network training are bearable and this architecture combines its simplicity with good results.

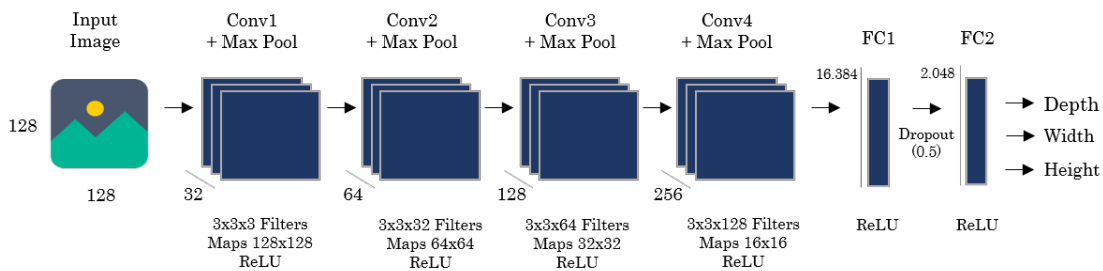


Figure 4.13: CNN architecture used for Model II.

Consider an image entering the network. Besides the resizing to 128x128, the image suffers no changes from the original size. Immediately, the input is subject to convolutional procedures in Conv1 where 32 different filters slide in the image originating 32 activation maps. The sliding process is similar in every CL: filters of size $3 \times 3 \times d$ (where d is the depth the corresponding CL input) slide with stride equal to 1 and padding also of value 1. These values for the stride and padding make the 2D size of the convolution output is the same as in the input. Following the convolution, resulting values are subjected to a ReLU activation. Afterwards, a max pooling operation happens in 2x2 regions within the obtained activation maps, reducing the size of the maps to half: in Conv1, the input is 128x128 and, after the max pooling, Conv2 receives an input of 64x64. This procedure is repeated for more three times with the difference that, in each CL, the number of resulting activation maps doubles and the corresponding width and height halves.

As soon as the max pooling occurs just after Conv4, there are 256 activation maps each with width and height of 8 (the max pooling reduces them from 16x16 to 8x8). The activation maps are now flattened into a single uni-dimensional vector in order to make use of FCLs. Between FCL1 and FCL2, the dropout with $p = 0.5$ is applied in order to attenuate overfitting. Finally, the network outputs three values, each representing an estimate for the depth, width and height.

4.3.2 CNN training

The training is carried out resorting to 75% of the whole dataset. Furthermore, as the CNN is intended to solve a regression problem, the loss function is naturally the mean squared error.

One of the most important choices regarding training is the optimiser to use. As seen in Chapter 2, when the dataset is not very large (as in this case, where less than 1300 images are used) then the Adam optimiser provides better results. Adam does not require parameter tuning due to its adaptive learning rate methodology to perform well with the starting learning rate default value of 0.001. To accomplish the gradient calculation, the mini-batch procedure is used.

Importantly, the idea is to train the final model for a total of 200 epochs and select the model with the smallest generalisation error. Hence, it becomes important for the model to converge as soon as possible to be in the convergence region for the largest possible amount of epochs, resulting on an increase probability of getting a better model. Furthermore, an intermediate level of fluctuation is important as it allows the model to eventually jump to better local minimums without deviating significantly from the convergence area.

The mini-batch size is a hyperparameter that must be tuned as shown in Figure 4.14. The graph shows the first 50 epochs of training with several mini-batches with power of 2 sizes. The represented validation loss is calculated in the remaining 25% of the dataset. Regarding the figure, one can notice larger batches, as 64 or 128, take a much greater number of epochs to converge, whereas a batch of 4 or 8 converges within 12 epochs. In fact, the mini-batch of 128 only converged after 35 epochs and the 64 batch stabilised around a higher loss value. Nevertheless, a batch of 4 means high levels of variation as visible through its bumpy graph. Surprisingly, the 8 mini-batch, a part from an early convergence, fluctuates much less than the 4 and has a similar behaviour to the 16 and 32 batches.

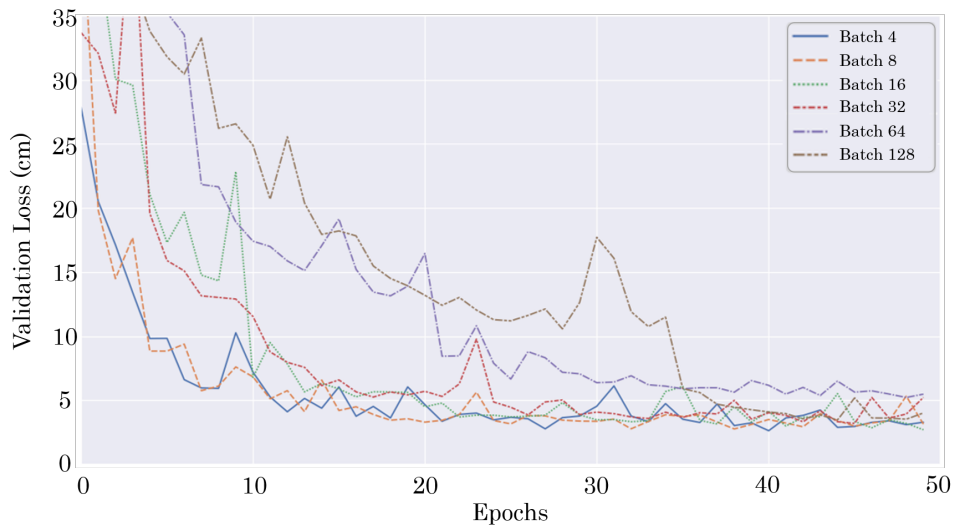


Figure 4.14: The impact of the mini-batch size in the model generalisation.

Taking into account the early convergence and the intermediate fluctuation, the chosen mini-batch size to use is 8.

Figure 4.15 shows the training and validation losses evolution during the training of the model with a mini-batch of 8 for 200 epochs.

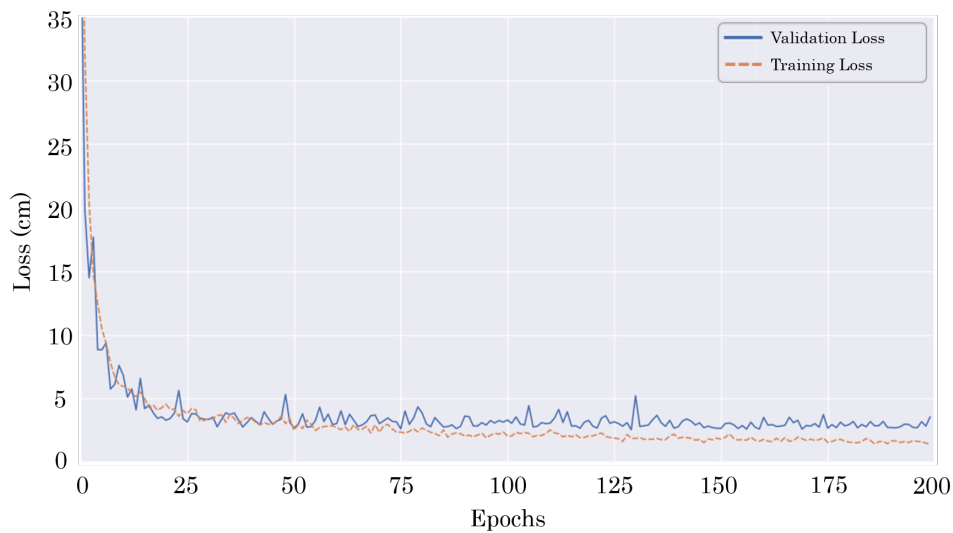


Figure 4.15: Model II training. No signs of overfitting, the model early converges to a low generalisation error and keeps fluctuating around it for the remaining epochs.

Since the generalisation error graphic converged and does not start increasing after 200 epochs, there are no signs of overfitting. The validation loss keeps fluctuating around approximately a mean squared error of 3cm and the chosen model is the one that represents the smallest loss. A more careful analysis of the model performance is present in the next chapter.

4.4 Model III - Hybrid Configuration

The third model is considered to be a hybrid approach as it includes components from the CV algorithm and the CNN model. Specifically, it uses the first stage of Model I to perform image segmentation through the highlight of important features in the image, such as the grid and the three axis. This can be considered as a region-based segmentation and the idea is to help the model focus on relevant areas of the picture. Theoretically, the model is going to easily identify the cardboard box position inside the picture and be able to understand the item dimensions through the length of the accentuated lines. After the segmentation, the image moves to the network presented in the Model II and an estimate for the depth, width and height is returned. Figure 4.16 illustrates the entire process.

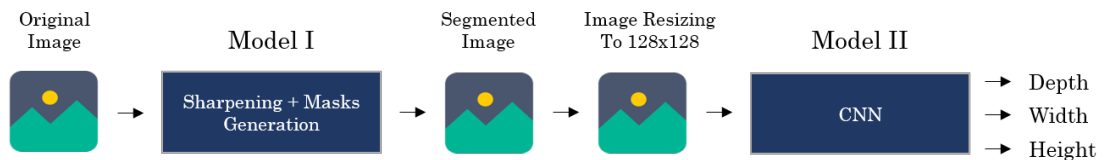


Figure 4.16: Model III configuration. Firstly uses the Model I to perform image segmentation and then the Model II to accomplish the estimate.

Resorting to the first stage of Model I, the image is sharpened and, afterwards, all the axis and grid are drawn on top of the original image as shows in Figure 4.17. The resulting picture is then resized to 128x128 before the CNN in order to have the correct input size.

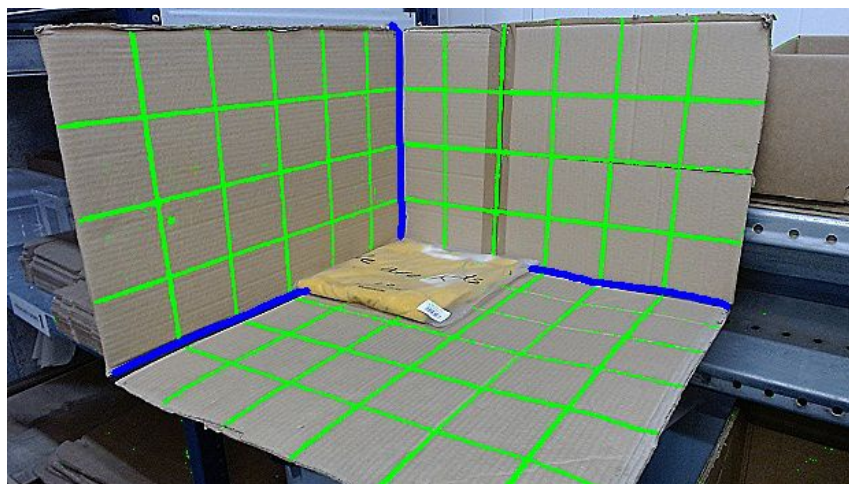


Figure 4.17: Image segmented through the highlight of the axis and grid and the input of the CNN.

Regarding the used CNN, the exact same architecture is used as in Model II. Concerning the training, also the Adam optimiser was chosen. Furthermore, the mini-batch size adopted is 8, similarly to the Model II.

4.5 Comparison of Models' performance in the estimation

The three models are compared regarding how well each performs in the estimation task. The performance of Model I is assessed by testing all the images whose items do not have depth or width over 50cm and height over 30cm, corresponding to 1134 pictures. On the other hand, Models II and III use all the 1224 pictures and their performance is measured resorting to the K-Fold cross validation methodology with 6 folds. The errors distribution is presented in Figure 4.18, and the corresponding averages are reported in Table 4.2.

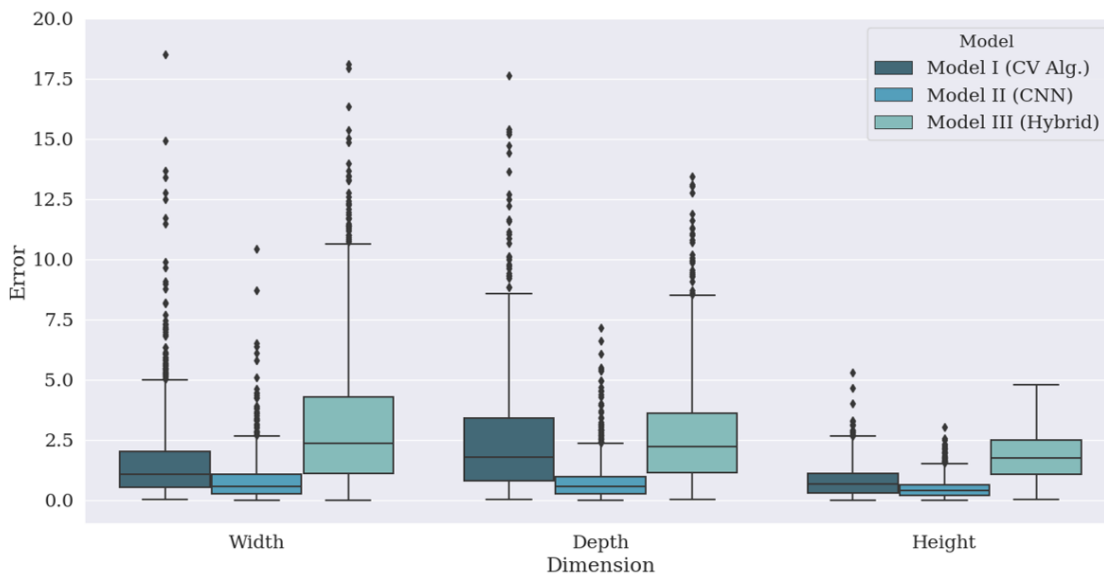


Figure 4.18: Absolute errors distribution for the 3 models.

Table 4.2: Mean absolute errors in cm per model and dimension (lower is better).

Dimension	Model I	Model II	Model III
Width	1.62	0.79	3.11
Depth	2.46	0.75	2.58
Height	0.80	0.46	1.78
Average	1.63	0.67	2.49

The Model II is significantly better than Model I and Model III with respect to the mean absolute error. In fact, Model II has as significantly lower average error, and smaller variability than what is observed in the other models. The CV algorithm arises as the runner-up model, outperforming the Hybrid model in all dimensions. These results are the starting point for the discussion in the next chapter, where the reason why the Hybrid model falls so much behind the CNN model is analysed. Moreover, an analysis over the largest errors present in Figure 4.18 is performed in order to acknowledge which are the type of items (or pictures) leading to such values.

Chapter 5

Discussion of Results

The most surprising result from the previous chapter is the poor performance from the hybrid model. This chapter starts by explaining why the chosen approach for the Model III did not result on a better outcome. Afterwards, Section 5.2 focuses in both Models I and II through an extensive analysis of their errors. The two best models are then globally compared and a choice is made over the model to be implemented. Finally, the chapter ends with the quantification of the project impact in the measurement process.

5.1 The reason behind the hybrid unsucces

The hybrid model provides the worst estimates among the three models. Although it uses the exact same architecture as the Model II, the error is significantly larger. At this stage, it would be important to understand why the CNN performs worse if the grid and axis are highlighted in the image. For this, it is fundamental to overcome the idea that any DL model must be seen as a black box. In fact, by analysing the activation maps resulting from convolutional processes, one can visualise which image features are being activated by the corresponding filter and, therefore, understand what the model is looking for. However, activation maps analysis can not be performed on very deep layers as the features identified at the later stages are not understandable for human eyes. Hence, the second CL output for models II and III are analysed in Figure 5.1, which conveys a visualisation of 8 activation maps out of the outputted 64 for each model.

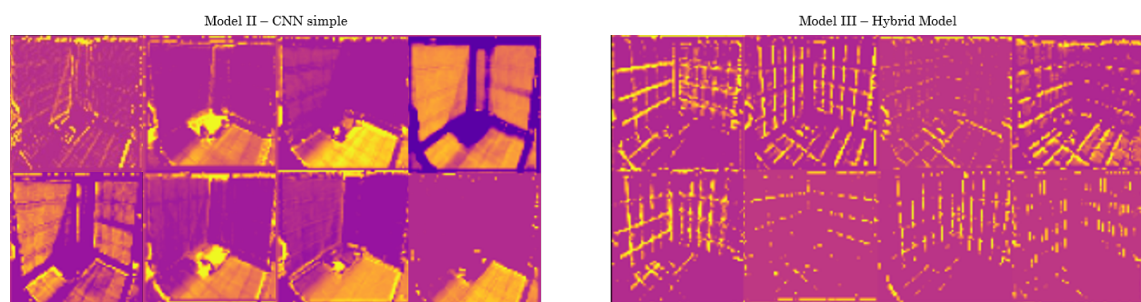


Figure 5.1: Difference between the outputs coming from the second CL for models II and III.

The difference between Models II and III is now clear. The highlighting of the grid and axis leads Model III to a substantial disregarding of the item silhouette and a natural focus in the accentuated lines, which does not translate in a good estimate. On the other hand, filters belonging to Model II clearly developed to identify the item within the image, which is certainly the reason behind the model's success.

In order to improve the performance of Model III, a possible solution would be to highlight the item instead of the three axis and the grid. Although one of the outputs from the Model I is a box identifying the item location, it would make no sense to use the CV algorithm to perform the item segmentation. This is because the box drawing is the exact last step of Model I and it is drawn resorting to the model estimates which would mean that, besides having to run the whole model only to get the box, the segmentation would be directly impacted by possible estimation errors and therefore jeopardise the Model III. Furthermore, possible approaches through colour selection rules are also not applicable as items are of the most diverse colours.

5.2 Further errors analysis

As seen in the box plot present in Figure 4.18, all the models have several outliers. An observation is considered an outlier if its value is outside of the range $[Q1 - 1.5 \times IQR; Q3 + 1.5 \times IQR]$, where Q1 and Q3 represent the first and third quartiles and IQR is the interquartile range of the corresponding error distribution. In this section, the focus is to explore the errors distribution and understand which type of pictures or items are leading to larger errors. The analysis is performed for the two best models, Models I and II.

5.2.1 Model I

Concerning the major absolute errors provided by the CV algorithm, a careful observation of the 54, 58 and 20 outliers from width, depth and height allowed to raise mainly four different issues:

- Axis walk to stop before reaching the item borders, as exemplified on the y-axis in Figure 5.2a. The reason is predominantly related to occasional areas in the black stripe where light is reflected and the region becomes very bright and, consequently, the belonging pixels take colours outside of the spectrum considered in the colour selection rules.
- Greedy axis walk goes over the items. This may happen either because the item is so shallow the axis keeps on being visible (see Figure 5.2b) or because, due to shadows, dark regions are created behind the item allowing the walk to go on, which is noticeable in Figure 5.2c.
- Walk stops due to the plastic bag involving the item, which provides greater error as the ground-truth measurement is done disregarding the wrapping. For instance, in Figure 5.2d, the y-axis walk ceased in the plastic of the item.
- Grid line walk stops before reaching the item leading to a wrong prediction, although the axis walk happens correctly (example in Figure 5.2e).

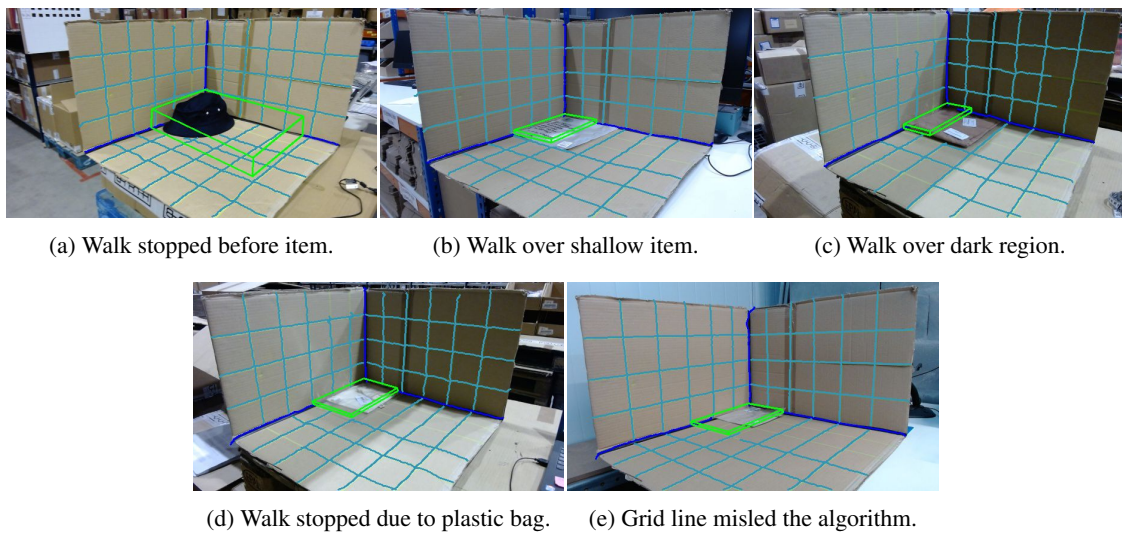


Figure 5.2: Exhibition of the problems leading to the largest errors when estimating with Model I

The outliers analysis also reveals the different items dimensions are affected by the described errors in distinct frequencies. For instance, the problems that affect the height prediction the most are the plastic bags stopping the walking algorithm or an early stop before the item. Figure 5.3 presents the distribution of error root cause from the analysed outliers for each of the three measurement predictions.

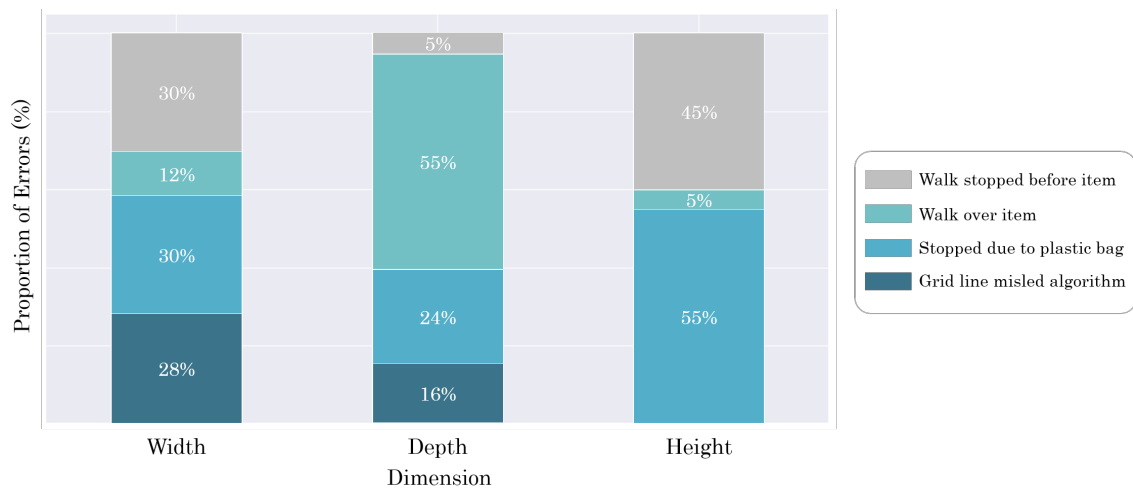


Figure 5.3: Errors distribution per root cause and dimension for the considered outliers.

More than half of the largest depth errors come from the algorithm to walk over the item. This is a valuable information to understand why the depth has a mean absolute error significantly larger than the width (more 0.84cm). As a matter of fact, walking over the item in the y-axis is a problem leading to an average error of 12.31cm, which is more than any other problem in any dimension, as seen in Table 5.1. The reason why the y-axis is more prone to this type of the error is related to the lighting conditions. For instance, Figures 5.2c and 5.2d are examples of shadows

from the box over the item. These dark areas mislead the algorithm to think that they belong to the axis and consequently the walking goes over the item.

Table 5.1: Mean absolute error (cm) caused per each problem raised in the analysed outliers

Problem	Width	Depth	Height
Grid line misled algorithm	5.99	8.42	—
Stopped due to plastic	7.51	9.19	2.79
Walk over item	8.02	12.31	2.89
Walk stopped before item	10.68	9.25	3.56

Nevertheless, the root causes raised in this section can be tackled. In the following paragraphs, each problem is individually addressed by explaining what is currently being done or what can be done in order to attenuate each of them.

First, the problem of walking over the items, either because the item is too shallow or because of shadows on the item, is corrected through the reduction of the algorithm dependence over the axis walk. This is done by measuring the item width and depth considering not only the distance between the end of the axis walk and the box origin, but also the distance from the end of the grid lines walk to the orthogonal axis (method thoroughly explained in Section 4.2.4). The walk over items problem produces the most significant errors and this grid line integration in the measurement had an enormous impact on reducing its cadence but there are two complications with this approach: (i) it led to the appearance of an occasional problem related to the grid line being responsible for wrong measurements (problem exemplified in Figure 5.2e); (ii) the algorithm has problems sometimes in identifying grid lines that would be crucial for the correct measurement and these cases are the ones leading to the complications in Figures 5.2b and 5.2c.

The grid line problem of misleading the algorithm in the measurement happens because the algorithm failed to walk over a given grid line on its fully extent. The grid mask generation was not able to catch specific parts of the grid and, consequently, the walk stopped. Therefore, the direct solution for this problem is to improve the grid mask generation process. However, since the colour selection rules are not applicable due to the lighting conditions variation from picture to picture, the current method is so far the best approach.

Regarding the issue of the algorithm stopping due to the plastic bag, the ideal solution is to make the picture to be captured with the item out of the wrapping. However, this is not feasible as the item unpacking and subsequent wrapping would significantly slow the photographing process.

Finally, the walk stopping before the item occurs due to the appearance of bright zones caused by light reflections. Therefore, this could be solved by integrating the colours of these bright zones in the spectrum considered by the axis colour selection rules. However, as the colours take light tones far from the intended black from the axis, imply broadening the allowed spectrum too much, jeopardising the black colour selection in the remaining dataset pictures.

5.2.2 Model II

The histograms in Figure 5.4 show how the CNN performs on predicting the depth, width and height of an item. The absolute errors distribution is similar for the depth and width, whereas the height tends have lower values. Nevertheless, all the three dimensions have a very satisfactory performance level.

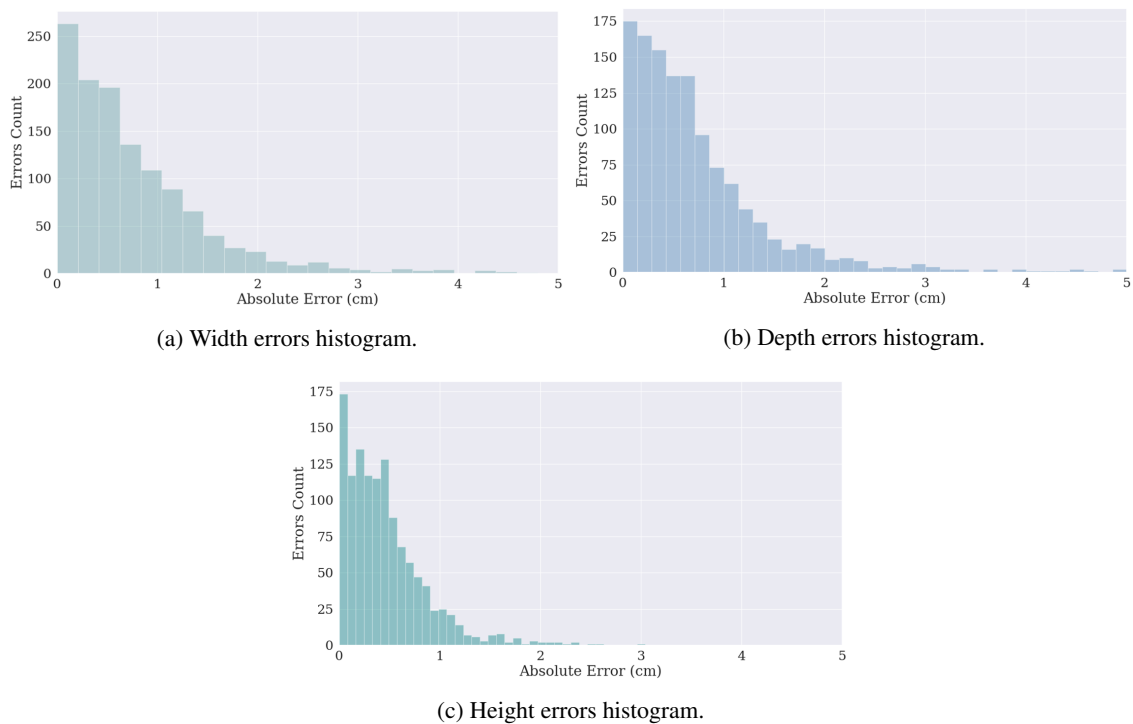


Figure 5.4: Individual histograms for each of the three dimensions' errors.

Regarding the errors analysis for the three dimensions, a study is executed aiming to identify some patterns leading to higher values. The examination happens by analysing the pictures corresponding to each error and checking specific problems such as the presence of shadows; the items ground truth measurements; or if the item has differentiating properties such as a very distinct color or shape.

In order to test the influence of shadows in the errors distribution, the dataset was split into pictures with shadows and pictures with no shadows, similarly to the work performed by the Shallow CNN in the CV Algorithm. The amount of images of each type are 584 and 640 respectively. To compare the two groups mean error, a bilateral two-sample t-test was done for each dimension with a significance level of 5%. The requirement of homogeneity of variances is tested resorting to the Levene Test. Regarding the normality assumption, the Kolmogorov-Smirnov test is performed, where the normality is rejected for the six distributions. However, as the samples are significantly large ($N > 30$), the central limit theorem allows the relaxation of the normality assumption (Cabral and Guimarães, 2010). The results for each Levene test and two-sample t-test are present in Table 5.2.

Table 5.2: Results obtained by applying the Levene test and the two-sample t-test to compare the impact of shadows on each dimension.

Errors Distribution	p-value Levene	p-value t-test
Width	0.249	0.190
Depth	0.293	0.291
Height	0.133	0.01

Concerning the width and depth, no patterns were found regarding the presence of shadows as the null hypothesis is not rejected (p-value higher than 0.05). Moreover, by individually analysing the pictures providing the higher errors for these dimensions, nothing peculiar in the photographed items is identified regarding the item colour or shape.

From Table 5.2 there is statistical evidence rejecting the null hypothesis of equality of the mean error for the height dimension between the two samples, as the p-value is smaller than 0.05. In other words, the height prediction of an item is impacted regarding the presence of shadows. Figure 5.5 shows a box plot comparing the height errors between pictures with and without shadows.

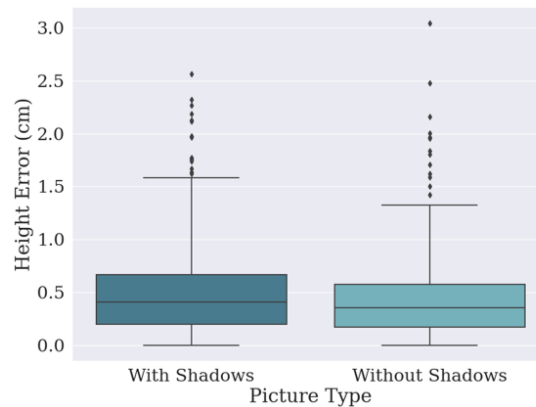


Figure 5.5: Height errors distribution in pictures with and without shadows.

Figure 5.5 suggests the error is higher for items present in pictures with shadows. Moreover, the analysis of the means of each group shows that the mean error is equal to 0.42cm for pictures with no shadows, against a mean error of 0.49cm from pictures with shadows. Therefore, it is possible to conclude that images with shadows lead to larger errors in height prediction.

In pictures with shadows, the box region around the origin generally contains more shadows. These regions can be the reason behind the worse performance of the model in the height measurement. Therefore, as the item is closer to the box origin or, in other words, as the item is shorter, it is possible that the error increases. In order to study if the shadows affect the height prediction depending on the items' real height, the errors were grouped into four balanced categories according to their ground truth height: $[0.5cm, 1.5cm]$; $[1.5cm, 2.5cm]$; $[2.5cm, 4cm]$; $[4cm, 12cm]$.

A two-sample t-test intending to compare the impact of shadows on each heights group was done with a significance level of 5%. This time, the test is unilateral where the null hypothesis

states the mean of the errors is equal for both groups, whereas the alternative declares the mean of the errors provided from pictures with shadows is the largest.

Table 5.3: Results obtained by applying the Levene test and the two-sample t-test to compare the impact of shadows on each height's group.

Dimensions Group	p-value Levene	p-value t-test
[0.5, 1.5[0.061	0.000
[1.5, 2.5[0.359	0.534
[2.5, 4[0.981	0.698
[4, 12]	0.760	0.507

From 5.3, a conclusion can be drawn: statistical evidence shows the shadows presence leads to worse height estimates in items belonging to the heights group of $[0.5\text{cm}, 1.5\text{cm}[$. Regarding the other groups, the study returned inconclusive as the corresponding p-values are greater than 0.05. The different distributions are visible in Figure 5.6.

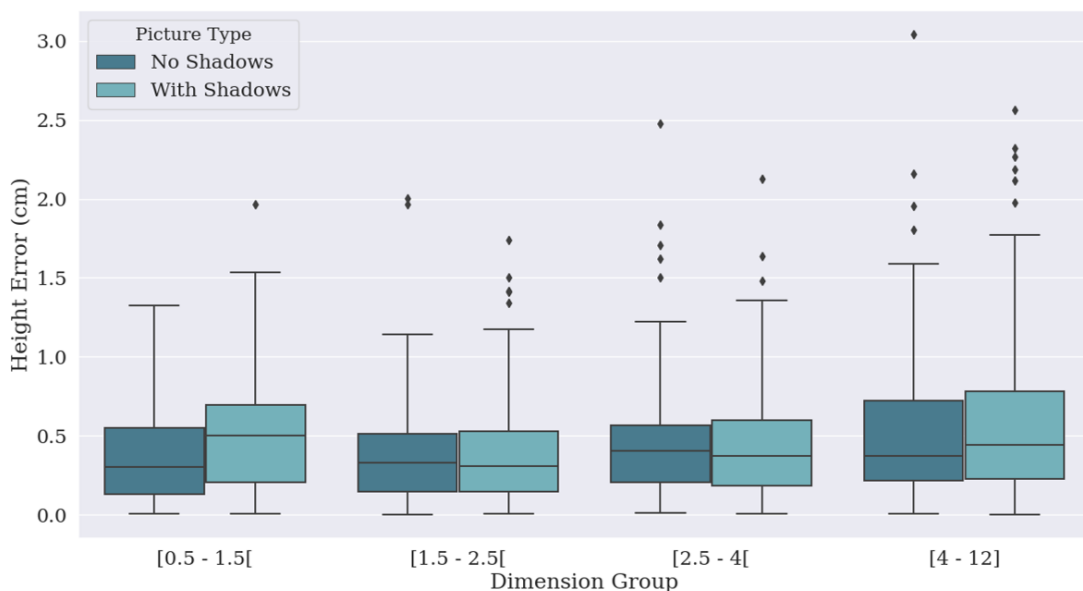


Figure 5.6: Box plot showing the influence of shadows per dimensions groups

In order to understand if shadows are truly impacting the different heights groups in a distinct way, an one-way ANOVA test is performed to the pictures with shadows, where the factor is the height group and the response is the error. The ANOVA has three important requirements: (i) errors normality, which is not met but, due to the significant size of each sample, the central limit theorem allows the relaxation of this requirement; (ii) homogeneity of variances of the errors, which is also not met but, as this assumption is only important when the samples are heavily unbalanced (i.e., dimension of the largest sample twice larger than the smaller sample) and this is not the case, it can be relaxed; (iii) errors independence, assumption fulfilled (Cabral and Guimarães, 2010). The test is done with a significance level of 5% and the null hypothesis stating all means are equal and the alternative declaring at least one of those is different.

The null hypothesis is rejected with a p-value inferior to 0.050. As suggested by the Figure 5.6, there is statistical evidence showing the mean error of the groups are different. Nevertheless, the ANOVA does not state which groups are statistically different. Therefore, a post-hoc analysis comparing the means of the groups pairwise using two-sample t-tests is done. The tests are bilateral and executed with a significance level of 5%.

Table 5.4: Pairwise comparison of the heights groups regarding the errors in pictures with shadows.

Height Group I	Height Group II	p-value t-test
[0.5, 1.5[[1.5, 2.5[0.017
[0.5, 1.5[[2.5, 4[0.022
[0.5, 1.5[[4, 12]	0.207
[1.5, 2.5[[2.5, 4[0.430
[1.5, 2.5[[4, 12]	0.000
[2.5, 4[[4, 12]	0.001

From the results tabled in 5.4 and by visualising Figure 5.6, there is statistical evidence showing the mean of the errors from groups [0.5cm, 1.5cm[and [4cm, 12cm] are larger than the remaining groups (p-values smaller than 0.050) but not different between each other, as the p-value is larger than 0.050.

To provide a term of comparison, a similar ANOVA test is done in pictures without shadows. The null hypothesis is rejected with a p-value of 0.002. Therefore, the post-hoc analysis is done and the results are stated in Table 5.5.

Table 5.5: Pairwise comparison of the heights groups regarding the errors in pictures without shadows.

Height Group I	Height Group II	p-value t-test
[0.5, 1.5[[1.5, 2.5[0.227
[0.5, 1.5[[2.5, 4[0.487
[0.5, 1.5[[4, 12]	0.000
[1.5, 2.5[[2.5, 4[0.087
[1.5, 2.5[[4, 12]	0.000
[2.5, 4[[4, 12]	0.003

The results are distinct for pictures without shadows. The statistical analysis allows to infer that taller items are significantly more subject to error than the other groups (p-value inferior to 0.050 in every test [4cm, 12cm] is present). In contrast, the error from shallow items is not significantly different as in pictures with shadows.

At this point, taking into account the entire statistical study executed, it is possible to infer two conclusions: (i) shadows impact items in a distinct way depending on the real height, where shallow items are more prone to error; (ii) taller items from group [4cm, 12cm] have larger error associated regardless of the picture type.

Regarding the first conclusion, there are signs that the previously raised hypothesis stating the shady regions generated around the box origin affect the height measurement for shallow items may be actually the cause of the larger errors. In Figure 5.7 is present an item where this exact problem occurs and whose height measurement error was 1.2cm.

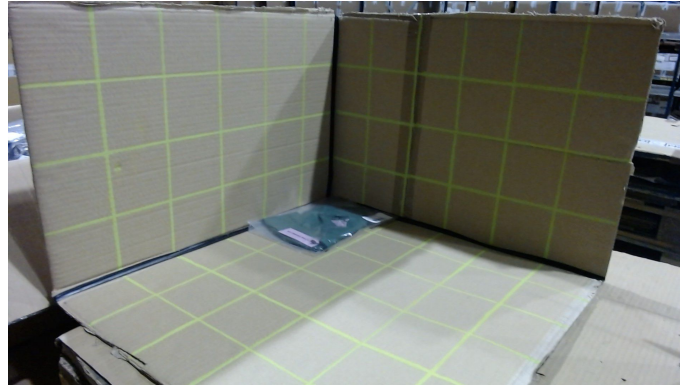


Figure 5.7: Picture with a dense dark region around the box origin.

Hence, in order to tackle this particular problem, the recommendation is to photograph these short items in a warehouse location where the lighting conditions generate a clean picture without shadows.

Concerning the second inference, the root cause of the error is related with the nature of each height group. The four heights categories were built in order to make the four classes to have a similar sample size. However, due to the dataset unbalance regarding heights, the range of each class in cm is very different: for instance, whereas $[1.5\text{cm}, 2.5\text{cm}[$ has a range of 1cm, the $[4\text{cm}, 12\text{cm}]$ has 8cm. Consequently, the amount of items per each cm within each category varies considerably as seen in Table 5.6.

Table 5.6: Items distribution per cm inside each height group.

Height Group	Range	Number of Items	Items per cm
$[0.5, 1.5[$	1cm	308	308.0
$[1.5, 2.5[$	1cm	275	275.0
$[2.5, 4[$	1.5cm	325	205.3
$[4, 12]$	8cm	316	38.5

The model has significantly less items per cm inside the group $[4\text{cm}, 12\text{cm}]$ to be trained with. Inevitably, the model struggles to distinguish the heights inside this group due to the lack of training data, leading to larger errors.

Thus, the second recommendation to improve the model performance is to provide more training data of items with height between 4cm and 12cm in order to balance the dataset. Therefore, the Operations team must now focus on photographing these taller items.

5.3 Models global comparison and Verdict

During the errors analysis in Section 5.2, the Model I manifested its strongest advantage: the transparency. One can easily understand what went wrong in inaccurate estimations just by observing the output image containing the green box. In fact, it was immediately possible to raise several issues directly impacting the model performance. In contrast, analysing the errors in Model II is a much harder task. There are some possible approaches such as the activation maps visualisation used in Section 5.1 or the patterns search utilised in Section 5.2.2 but the interpretation is less clear than in the CV algorithm. Moreover, another feature of Model I is the absence of training which is advantageous for three reasons: (i) the training is a slow process; (ii) the model is independent from real measurements and possible error coming with them. Finally, whereas the CNN model needs a heavy file responsible for saving the values of its 33 million parameters (this file has almost 400MB) in order to be able to operate, the CV algorithm does not occupy more than 33MB and, therefore, is extremely lightweight in comparison.

Some of the pros coming from the Model II are not only the more accurate predictions but also the estimation speed. The CNN model is able to process and estimate over 1.000 pictures in one minute. To put in perspective, the CV algorithm takes over four hours for the same 1.000 pictures, which means the CNN estimation time largely compensates for the time spent in training. Furthermore, Model II can measure any item that fits the cardboard box, whereas the Model I is limited to depth and width under 50cm as well as height under 30cm.

Taking into account the positive and negative points from each model, the CNN model is the approach recommended for the measurement of fashion items in the HUUB context. The high speed in the prediction process and, above all, the smaller estimation error justify this choice.

5.4 Impact in the time spent on the measurement process

Finally, recalling the AS-IS situation exposed in Section 3.1, the manual measurement procedure, which included not only the measurement but also the data insertion, was taking on average 50.8s per item. In the TO-BE scenario, present in Figure 5.8, the manual procedure is replaced by the item photographing. Although the procedure is not yet implemented, a representative simulation was executed and the item photographing process length was measured. The average time taken by the new process is 17.7s per item, which represents a noteworthy reduction of 65% of the time spent on this process.

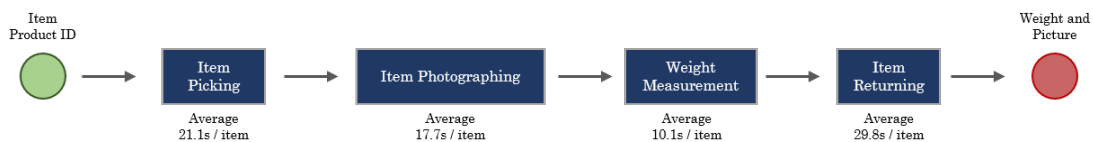


Figure 5.8: The item photographing reduces the measurement procedure by 65%.

Chapter 6

Final Considerations and Future Work

The foremost purpose of the work presented in this dissertation was the development of a solution that would enable HUUB to get the width, depth and height measurements of fashion items resorting to a single picture. HUUB, as a logistics company responsible for managing fashion brands supply chains, needs accurate estimates of dimensions in order to predict transportation costs. As these shipping costs are calculated resorting to the volume and weight to be carried, it is crucial for HUUB to have each item's dimensions available.

In the beginning of the project, the items measurement was being executed through a time-consuming manual procedure, and the idea was to replace this process by a simpler and faster action of taking a picture of the product. Hence, this project intended to build a model to process item pictures resorting to CV techniques and output the measurement of each of its three dimensions (depth, width and height).

The chosen methodology consisted on the complete development of three distinct models: (i) a CV algorithm; (ii) a CNN-based approach (i.e., DL); (iii) a hybrid configuration, where the CV algorithm performs an image segmentation task before the picture is processed by a CNN. These models were then compared, ending up with a choice over the model to be implemented. The choice was made taking into account not only the models accuracy in the prediction task, which was assessed through the mean absolute error, but also other performance metrics such as the time needed to provide a prediction, and the transparency of the model regarding its functioning. In the end, the CNN model was selected to be implemented as it provided the faster predictions with the lowest errors.

From this point, this chapter intends to wrap up the important key ideas resulting from the project development. In Section 6.1, the contribution from this work to the scientific field of CV is described. Section 6.2 focuses on gathering and presenting the most important conclusions drawn over the course of the dissertation. Finally, the thesis ends with a reflection over further work opportunities identified through the execution of this project.

6.1 The innovative side of the adopted methodology

In literature CV-related works, the model to be developed and deployed was identified from the beginning of the project. Usually, it was well defined if the problem was going to be solved resorting to traditional CV or to DL techniques. In contrast, the methodology adopted in this dissertation is unique as the nature of the model to be implemented to solve the problem was not obvious beforehand.

The direct contributions to the CV field provided by this dissertation are:

- Three novel approaches to the problem of predicting items' dimensions based on a single picture, allowing the direct comparison of the three methods for the same exact problem.
- Creation of a multi-functional picture setup comprised by the grid drawn in the cardboard box that allows the execution of the 3D measurement of the item and the tackle of perspective deformation caused by the camera.
- CV algorithm innovative contributions:
 - Integration of the Shallow CNN on its functioning, which makes a significant difference since it allows the algorithm to work through different light conditions, whereas typical CV algorithms are applied in problems where the picture is always captured in a fixed location.
 - Line detection method by creating the walking process over the lines and integrating regression techniques to get the corresponding line equation.
 - Generation of the grid mask without resorting to colour selection rules.
- CNN model innovative contributions:
 - Usage of a DL approach to CV to address a regression problem, as these methods are usually applied on assignments related to image segmentation, object detection, image classification or image generation.
 - 3D measurement of an item through a CNN, a very unique procedure.
- Direct and original integration of the CV algorithm and the CNN model to generate a hybrid configuration.

6.2 Key Outcomes

6.2.1 Models Summary

Among the three developed models, the CNN-based turned out to perform better than the remaining two and it was chosen to be implemented. The estimation mean absolute errors achieved are 0.79cm, 0.75cm and 0.46cm for the width, depth and height, respectively. The performance of the

model in all dimensions is far superior to the other models. Besides, the prediction speed from the CNN model deserves also an honorable mention as it can estimate the dimensions of more than 1.000 items in one minute. In contrast, the lack of transparency associated with the model functioning is its main weak point. It hampers the process of understanding the origin of its largest errors and therefore it is not straightforward to identify and tackle the root causes. Nevertheless, a careful statistical analysis over the errors distribution of the CNN model allowed to infer two points where the model can be improved:

- The height measurement of items with real height below 1.5cm is affected by shady regions created around the box origin caused by lighting conditions. A performance maximisation is thus expected if pictures to shallow items are from now on taken in good lighting conditions.
- The height measurement of items taller than 4cm is more prone to error due to the lack of these items in the dataset. The direct solution is to promote the photographing of items with real height above 4cm in order to balance the dataset.

The CV algorithm was the runner-up model, providing very interesting results. The main advantage of this model over the other two is the ease on understanding how the algorithm behaves for a given image, which means that any estimation far from the truth measurement of an item can easily be traced and the root cause identified. Besides the worse dimensions predictions, the major problem from this model is the prediction time of 15s per item, significantly more than the CNN model.

At last, the Hybrid model performed the worst. The model configuration implied the CV algorithm was used to perform the segmentation of the grid and the axis in the picture, theoretically helping the CNN to understand the cardboard box location and where each highlighted line finishes. However, that approach did not succeed as the model would solely focus on these accentuated lines and disregard the item. From the scientific point of view, nothing can be concluded about the pertinence of the hybrid models as the CV approach used to complement the CNN functioning was not able to bring the expected benefits from this union. For further work, a hybrid model could be built by using CV techniques to help the CNN model overcome identified problems (as the height estimation in pictures with shadows). In this case, the hybrid model was not configured to address any particular limitation of from the CNN model but to directly integrate the two models and use the CV algorithm to perform image segmentation tasks.

6.2.2 Impact at HUUB

The AS-IS situation, the measurement procedure had two problems regarding the real dimensions obtained: (i) the manual measurements process was a time-consuming operation, leading to training data scarcity; (ii) human error was often present due to wrong interpretations of the measuring tape or mistakes during data registering. This project, by replacing the manual measurement with the item photographing, addresses these two problems.

On the one hand, the hand-operated procedure was taking on average 50.8s per item and, with the project implementation, this duration changes to 17.7s per item, representing an improvement

of 65%. Therefore, the time-consuming problem is attenuated. On the other hand, the human error impact is mitigated as the operator is no longer required to read the measuring tape or to manually insert data. Instead, only a picture must be captured.

HUUB had also the problem of, after the packaging, not being able to anticipate the transportation cost when the pack is a flyer, due to its variable volume. However, with the developed model, the company is now capable of estimate these costs by taking a picture of the flyer to be dispatched.

Moreover, this thesis, by introducing the process of photographing items, enabled the capability of visually compare the obtained measurements and the corresponding item. This is useful as it allows to identify evident errors. Otherwise, only the measurements would be received, leading to an extremely hard task of detecting irregularities.

Finally, the overall goal of improving the accuracy of transportation costs estimation cannot yet be measured. As the project focuses on the improvement of the measurement process, the shipping costs prediction are not immediately impacted by the model deployment. Hence, the project impact in this issue can only be fully appreciated at a later stage.

6.3 Further Improvement Opportunities

The CNN-based model was chosen to be implemented but it is naturally subject to improvement. An idea to enhance the model performance may be a concatenation of two CNN. The first network is intended to perform the item segmentation in the picture which feeds the second CNN, responsible for the estimate. This approach could help the estimating model on focusing exclusively on the item and therefore provide better results.

Furthermore, the developed model may be upgraded to also have as output the weight of an item. However, considering the presence of several types of products dealt by HUUB, a good idea may be to also input the item's family as a complement to the picture. For instance, a blanket and a pillow are both bulky but very different in weight.

Lastly, although outside of the scope of the project underlying the work of this thesis, an analysis of the TO-BE scenario (present in Figure 5.8) allows to infer the need of acting over the time spent in picking and returning items. A possible solution for this problem is to actually eliminate these tasks which can be done by incorporating the measuring process within the normal warehouse operations. In fact, when items are received in the warehouse, the Operations team has to register in the internal system which items were received. In the context of this process, if a picture is taken at this stage, the picking and returning tasks can be eliminated.

Hence, this project was not only useful to help HUUB to address the important problem of the transportation costs estimation, but also to open the improvement doors aforementioned. The seize of these opportunities allow the company to continue its growth through continuous improvement, helping HUUB to aspire even higher flights.





Bibliography

- Cabral, J. A. S. and Guimarães, R. C. (2010). *Estatística*. Verlag Dashöfer Portugal.
- Clevert, D. A., Unterthiner, T., and Hochreiter, S. (2016). Fast and accurate deep network learning by exponential linear units (ELUs). *4th International Conference on Learning Representations, ICLR 2016 - Conference Track Proceedings*, pages 1–14.
- Duchi, J. C., Bartlett, P. L., and Wainwright, M. J. (2012). Randomized smoothing for (parallel) stochastic optimization. *Proceedings of the IEEE Conference on Decision and Control*, 12:5442–5444.
- Glorot, X., Bordes, A., and Bengio, Y. (2010). Deep sparse rectifier neural networks. *Journal of Machine Learning Research*, 15.
- Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press. <http://www.deeplearningbook.org>.
- Karakose, M., Salur, M. U., Othman, N. A., and Aydin, I. (2019). An Embedded Real-Time Object Detection and Measurement of its Size. *2018 International Conference on Artificial Intelligence and Data Processing, IDAP 2018*, (September):1–4.
- Karpathy, A. (2019). Stanford university lecture notes - cs231n convolutional neural networks for visual recognition.
- Keskar, N. S., Mudigere, D., Nocedal, J., Smelyanskiy, M., and Tang, P. T. P. (2016). On large-batch training for deep learning: Generalization gap and sharp minima. *CoRR*, abs/1609.04836.
- Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization.
- Kolesnikov, A., Beyer, L., Zhai, X., Puigcerver, J., Yung, J., Gelly, S., and Houlsby, N. (2019). Large Scale Learning of General Visual Representations for Transfer.
- Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In Pereira, F., Burges, C. J. C., Bottou, L., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc.
- Lecun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *Nature*, 521(7553):436–444.
- Liu, X., Li, S., Kan, M., Zhang, J., Wu, S., Liu, W., Han, H., Shan, S., and Chen, X. (2015). AgeNet : Deeply Learned Regressor and Classifier for Robust Apparent Age Estimation. pages 16–24.

- Maas, A. L., Hannun, A. Y., and Ng, A. Y. (2013). Rectifier nonlinearities improve neural network acoustic models. in *ICML Workshop on Deep Learning for Audio, Speech and Language Processing*, 28.
- Myilsamy, R. and Muthukrishnan (2011). Edge Detection Techniques for Image Segmentation. *International Journal of Computer Science & Information Technology (IJCSIT)*, 3(6):259–267.
- O’Mahony, N., Campbell, S., Carvalho, A., Harapanahalli, S., Hernandez, G. V., Krpalkova, L., Riordan, D., and Walsh, J. (2020). Deep Learning vs. Traditional Computer Vision. *Advances in Intelligent Systems and Computing*, 943(Cv):128–144.
- Prechelt, L. (2012). Early stopping - But when? *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 7700 LECTU:53–67.
- Rosebrock, A. (2017). *Deep Learning for Computer Vision*.
- Ruder, S. (2016). An overview of gradient descent optimization algorithms. *CoRR*, abs/1609.04747.
- Shimobaba, T., Kakue, T., and Ito, T. (2018). Convolutional Neural Network-Based Regression for Depth Prediction in Digital Holography. *IEEE International Symposium on Industrial Electronics*, 2018-June:1323–1326.
- Sinha, R. K., Pandey, R., and Pattnaik, R. (2018). Deep Learning For Computer Vision Tasks: A review.
- Song, Y. and Yan, H. (2017). Image segmentation algorithms. *Architectures and Algorithms for Digital Image Processing II*, 0534:172.
- Springenberg, J., Dosovitskiy, A., Brox, T., and Riedmiller, M. (2015). Striving for simplicity: The all convolutional net. In *ICLR (workshop track)*.
- Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958.
- Szeliski, R. (2011). *Computer vision algorithms and applications*. Springer, London; New York.
- Tang, Y. (2013). Deep Learning using Linear Support Vector Machines.
- White, D. J., Svellingen, C., and Strachan, N. J. (2006). Automated measurement of species and length of fish by computer vision. *Fisheries Research*, 80(2-3):203–210.
- Wiley, V. and Lucas, T. (2018). Computer Vision and Image Processing: A Paper Review. *International Journal of Artificial Intelligence Research*, 2(1):22.

Appendix A

One Point Lesson for Photographing

		Items Measurement	
Responsible: Team Leader and team members		Sector: -	NT 13/18
		Machine: -	
Nº	Activity	Picture	
1	<p>Place the item in the box corner, adjacent to the box sides.</p> <p>If a product is clothing, the item should be left folded and inside the plastic bag, just as it is expected to go inside the cardboard box/flyer.</p>		
2	<p>Stand diagonally regarding to the item location as seen in the figure. The three planes of the box should appear on an equal proportion in the resulting picture.</p> <p>Make sure the picture is captured as close as possible to the item but: (i) include the whole item in the picture; (ii) include the 3 black stripes on their full extent.</p>		
2.1	<p>If the item fits the box, capture the item according to the above guidelines. The resulting picture should be similar to the one exemplified on the right.</p>	<p>Webcam picture:</p> 	



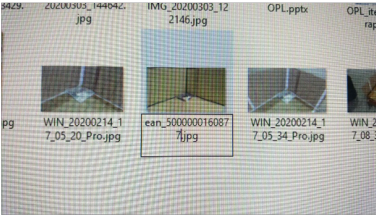
Page 1/2

DATE: 13/02/2020

REVIEWED: Cristina Fernandes

APPROVED:

Figure A.1: One Point Lesson for Photographing (1/2).

		Item Measurement		
Responsible: Team Leader and team members		Sector: -	NT 13/18	
		Machine: -		
Nº	Atividades	Picture		
2.2.	If the item is too large to fit the box, place the item as close as possible to the box corner. Capture the item on its full extent, as well as the black stripes.	Webcam Picture: 		
3.	Save the picture with the name containing the item's EAN. Ex.: "ean_5000000160877.jpg"			

Page 2/2	DATE: 13/02/2020	REVIEWED: Cristina Fernandes	APPROVED: <input type="text"/>
-----------------	----------------------------	--	--

Figure A.2: One Point Lesson for Photographing (2/2).