Tampereen yliopisto

Anni Hakola

DEEP NEURAL NETWORK FOR AUTOMATIC VEHICLE DETECTION

Master of Science Thesis

Faculty of Information Technology and Communication Sciences
Master's Thesis
June 2019

ABSTRACT

Anni Hakola: Deep Neural Network for Automatic Vehicle Detection
Master of Science Thesis, 65 pages
Tampere University
Master's Degree Programme in Electrical Engineering
June 2019
Examiners: University Lecturer Erja Sipilä and Associate Professor Heikki Huttunen

---

Machine learning has achieved an important role in research, business and everyday life in the form of, for example, automatic aviation, face and speech recognition and virtual reality games. Visy Oy, a company in Tampere, Finland, is developing various tools for automatic traffic control. The tools include an access gate consisting of an inductive loop or a laser scanner, a barrier and a camera. The purpose of a loop or a scanner is to trigger a camera to take an image when a vehicle is in the correct spot to which the camera is zoomed and focused. The image is fed to a license plate recognition software and a permit decision is made according to the recognized plate. If the access is accepted, the barrier will open.

This Thesis has two aims regarding machine learning combined with automatic traffic control. The first aim is to search, study and test high-image-quality cameras and decide, whether they are suitable for Visy projects or not. The high image quality is motivated by the customers' need for recognizing small details, such as seals and dangerous goods labels, from an image that is taken of a whole container. The current cameras that Visy Oy is using are not sufficient for this purpose.

Three cameras are chosen for the camera tests including Sony's video surveillance camera, Canon's digital single-lens reflex camera and the current camera used in the projects, Basler's video surveillance camera. Only Sony and Basler are included in the final tests because of a problem in software support in Canon's camera. The tests are performed in Visy Oy's perspective and for Visy Oy's needs in the office of Visy Oy, and the results are observed and estimated visually. In the tests, the cameras shoot images every 15 minutes during the night also and the images are saved to a folder on a computer.

Sony is found to have significantly higher image quality, especially at night, compared to Basler. Sony fulfils Visy's requirements and is found to be suitable for Visy's projects. It has already been proposed to a potential project where small details need to be recognized, but no confirmation has been received for the project while writing this Thesis.

The second aim of this Thesis is to implement a deep convolutional neural network for automatic vehicle detection, called a virtual trigger. Its purpose is to replace inductive loops and laser scanners in Visy projects. In other words, image frames are captured from a camera and each frame is classified to contain a vehicle on the correct spot or not. If the image is classified to have a vehicle on the correct spot, an image for license plate recognition is triggered. Three different network models are implemented, trained and tested, including two pre-trained models and one model that is created from scratch.

The requirements for the virtual trigger network are that it is fast and classifies the images with a high classification accuracy, meaning over 99 %. The neural network tests show that one of the pre-trained network models achieves almost all the goals and is chosen for real-life tests, which are not a part of this Thesis. Virtual trigger is operating on a real installation now. The results are promising, but further improvements are needed for obtaining over 99 % accuracy in real life.

Almost all the goals were achieved, a suitable camera was found, and virtual trigger obtained over 99 % validation accuracy. Camera tests were slightly one-sided and virtual trigger did not exceed the aim on the test data, but the future for both parts looks promising.

Keywords: automatic vehicle detection, machine learning, deep convolutional neural networks, image classification, cameras, image quality, image sensor

TIIVISTELMÄ

Anni Hakola: Syvä neuroverkko automaattiseen ajoneuvontunnistukseen
Diplomityö, 65 sivua
Tampereen yliopisto
Sähkötekniikan tutkinto-ohjelma
Kesäkuu 2019
Tarkastajat: Yliopistonlehtori Erja Sipilä ja Associate Professor Heikki Huttunen

---

Koneoppiminen on saavuttanut tärkeän roolin tutkimuksessa, yrityselämässä sekä ihmisten jokapäiväisessä elämässä esimerkiksi automaattisen lentokoneiden ohjauksen, kasvojen ja puheen tunnistuksen sekä virtuaalitodellisuuden pelien muodossa. Visy Oy, joka toimii Suomessa Tampereella, kehittää erilaisia työkaluja automaattiseen liikenteen ohjaukseen. Nämä työkalut pitävät sisällään automaattisen portin, joka koostuu induktiosilmukasta tai laserskannerista, puomista ja kamerasta. Induktiosilmukan tai laserskannerin tarkoituksena on käskeä kameraa ottamaan kuva, kun ajoneuvo on oikealla kohdalla kameraan nähden, eli siinä, mihin kamera on kohdistettu ja tarkennettu. Kuva lähetetään rekisterinkilpitunnistusohjelmalle, ja lupapäätös tehdään tunnistetun kilven perusteella. Jos lupa on kunnossa, puomi aukeaa.

Tällä työllä on kaksi tavoitetta liittyen automaattiseen liikenteenohjaukseen ja koneoppimiseen. Ensimmäinen tavoite on etsiä, tutkia ja testata korkean kuvanlaadun omaavia kameroita ja päättää, sopivatko ne Visyn projekteihin. Korkea kuvanlaatu on lähtöisin asiakkaiden toiveesta tunnistaa pieniä yksityiskohtia, kuten sinettejä ja vaarallisten aineiden merkkejä, kuvasta, joka on otettu kokonaisesta kontista. Visy Oy:n nykyisin käyttämät kamerat eivät ole riittäviä tähän tarkoitukseen.

Kolme kameraa valittiin kameratesteihin. Nämä ovat Sonyn videovalvontakamera, Canonin järjestelmäkamera ja tällä hetkellä käytössä oleva Baslerin videovalvontakamera. Vain Sony ja Basler olivat mukana testeissä johtuen Canonissa ilmenneestä ohjelmistotuen ongelmasta. Testit toteutettiin Visyn näkökulmasta ja Visyn tarpeita ajatellen Visy Oy:n toimistossa ja tulokset arvioidaan visuaalisesti. Kamerat ottivat 15 minuutin välein kuvia testeissä, myös yöaikaan, ja kuvat tallennetaan kansioon tietokoneelle.

Sonyn kamerassa todettiin olevan huomattavasti korkeampi kuvanlaatu kuin Baslerissa, erityisesti yökuvissa. Sonyn kamera täyttää vaatimukset ja soveltuu Visy Oy:n projekteihin. Sitä on jo tarjottu korkean kuvanlaadun kameraksi yhteen mahdolliseen projektiin, jossa on tarkoitus tunnistaa pieniä yksityiskohtia kuvista, mutta projekti ei ole varmistunut tätä työtä kirjoitettaessa.

Toinen työn tavoite on toteuttaa syvä konvoluutioneuroverkko automaattiseen ajoneuvontunnistukseen, jota kutsutaan nimellä ”virtual trigger”. Sen tarkoituksena on korvata induktiosilmukat ja laserskannerit Visyn projekteissa. Toisin sanoen, kameralta napataan kuvia ja jokainen kuva luokitellaan sen mukaan, onko siinä ajoneuvo oikealla kohdalla vai ei. Kun ajoneuvon havaitaan olevan oikealla kohdalla, käsketään kameran ottaa kuva rekisterinkilpitunnistusta varten. Kolme eri neuroverkkoa toteutettiin, opetettiin ja testattiin tässä työssä. Näistä kaksi on esiopetettuja verkkoja ja yksi rakennetaan itse tyhjästä.

Neuroverkon vaatimukset ovat, että se on nopea ja luokittelee kuvia korkealla luokittelutarkkuudella, tarkoittaen yli 99 prosentin tarkkuutta. Neuroverkkotestit osoittivat, että yksi opetetuista verkoista toteuttaa lähes kaikki vaatimukset ja kyseinen verkko valittiin tosielämän testeihin, jotka eivät ole osa tätä työtä. Virtual trigger on toiminnassa eräässä projektissa tällä hetkellä. Tulokset ovat tähän asti olleet lupaavia, mutta verkko vaatii vielä parannuksia saavuttaakseen yli 99 % tunnistustarkkuuden.

Lähes kaikki tavoitteet saavutettiin työssä: löydettiin Visyn projekteihin soveltuva korkean kuvanlaadun kamera ja virtual trigger ylitti 99 % luokittelutarkkuuden validointidatalle. Kameratestit jäivät hieman yksipuolisiksi, eikä virtual trigger ylittänyt toivottua tarkkuutta testidatalle, mutta tulevaisuus näyttää lupaavalta molempien osioiden osalta.

Avainsanat: automaattinen ajoneuvontunnistus, koneoppiminen, syvät konvoluutioneuroverkot, kuvien luokittelu, kamerat, kuvanlaatu, valoherkkä kenno

PREFACE

First, I want to thank my supervisor D. Sc. Tech. Jyrki Selinummi for helping me to find an interesting topic for my thesis and supporting and guiding me through this challenging project. I want also to thank the executive manager of Visy Oy, Petri Granroth, for making this project possible by supporting it. I want to thank my inspectors, Erja Sipilä and Heikki Huttunen, who made the effort to check and grade my thesis. I want to also thank my family, who supported me through all my studies including this thesis.

In Tampere, 17.5.2019

Anni Hakola

CONTENTS

LIST OF SYMBOLS AND ABBREVIATIONS

ADC	Analog to digital converter, converts analog signals to digital
ANN	Artificial neural network
APS	Active pixel sensor, a CMOS image sensor architecture
API	Application programming interface
AUC	Area under (ROC) curve
CCD	A charge-coupled-device technology, used in image sensors
CFA	A colour-filter array used in image sensors to form colour images
CMOS	A complementary metal-oxide semiconductor transistor technology used in image sensors
CNN	Convolutional neural network
CPU	Central processing unit
DOF	Depth of field
DPS	Digital pixel sensor, a CMOS image sensor architecture
DSLR	Digital single-lens reflex camera
DR	Dynamic range in photography
EDSDK	EOS Digital Software Development Kit made by Canon for EOS cameras
f-number/f-stop	Ratio between the focal length and the diameter of an optical system
FN	False negative
FNR	False negative rate
FP	False positive
FPR	False positive rate
FPS	Frames per second
GPU	Graphics processing unit
HTTP	Hypertext transfer protocol
ISO	International Organization for Standardization
JPEG	Joint photographic experts group
LCD	Liquid crystal display
MAE	Mean absolute error
MLP	Multi-layer perceptron
MSE	Mean squared error
PoE	Power over Ethernet
PPS	Passive pixel sensor, a CMOS image sensor architecture
ReLU	Rectified linear unit or rectified linear function: an activation function used in convolutional neural networks
RGB	Red Green Blue
ROC	Receiver operating characteristic
RTP	Real-time transfer protocol
SDK	Software development kit
SGD	Stochastic gradient descent
SNR	Signal-to-noise ratio
TN	True negative
TNR	True negative rate, a.k.a. specificity
TP	True positive
TPR	True positive rate, a.k.a. sensitivity or recall
UI	User interface
USB	Universal serial bus

1. INTRODUCTION

Machine learning has achieved an important role in research, business and everyday life. Different machine learning algorithms are also used for fun, like virtual reality games and experiences. Algorithms are also developed for making the life of people easier and for decreasing human errors in the fields where it is possible. For example, the automatic aviation in airplanes decreases the possibility of a human error made by a tired pilot. However, some people are worried about being replaced by robots and don't still believe that a machine could perform tasks better than a human.

Visy Oy has developed automatic access and traffic control systems for industry. The systems are globally used in, for example, shipping terminals, border control and factories. It includes machine learning: each time a vehicle wants to enter a certain area, the license plate is recognized, and this is performed with an optical character recognition machine learning algorithm. In addition to license plates, Visy Oy implements, for example, container and wagon number recognition and seal and hazardous materials sign recognition.

Image quality plays a major role in machine learning systems where the algorithms are supposed to recognize small details from images, like in Visy projects. If the image quality is low, it is difficult or even impossible for human eye and for a machine to recognize these details in an image. Therefore, one part of this thesis focuses on the basic principles of cameras and on which factors affect image quality. One aim of this Thesis is to find a camera that is suitable for Visy projects, and offers higher image quality than the current cameras used in the projects. In this Thesis, few cameras are investigated and tested and the use cases for a high-quality camera are considered.

One part of this Thesis consists of machine learning. The cameras in Visy's traffic control systems are zoomed at a certain point, and when a vehicle drives to a gate, it is important to take the image at the correct spot. Currently this is performed with inductive loops, which recognize large amounts of metal over them, or with laser scanners, which alert, when something passes a location that is configured to be on an alarm area. The difficulty of inductive loops is that they are dug to the ground, which makes them difficult to move in cases where gates are relocated. Digging is also expensive. In cases, when there is something magnetic nearby, the loops don't work correctly, because they react

to the change in the magnetic field caused by metal in a vehicle. Laser scanners, on the other hand, are easier to move and not that expensive because no digging is needed. However, they react to everything that passes the alarm area, like rain, snow, animals and humans. And because of this they are not 100 % reliable when it is raining or when a moose decides to pass a gate.

The second aim of this Thesis is to implement a mechanism for vehicle detection in software. We call this algorithm a *virtual trigger* because its purpose is to trigger images exactly like loops and scanners do and to be nearly as reliable without causing too much extra photo shooting. The virtual trigger is implemented as a deep convolutional neural network that recognizes from the image if there is a vehicle on a certain spot. So, the purpose is not to locate vehicles but to trigger an image when a vehicle is on a desired spot. The idea is to capture frames from the camera's video stream and perform a classification with two classes (vehicle or no vehicle) for each frame. When the frame is classified to the vehicle class, the actual permit image is taken and the license plate recognition for the image performed.

We could also just detect license plates from the frames instead of vehicles, but this would mean that vehicles with no plates (i.e. snowploughs and the vehicles owned by ports and factories) would be unrecognized. Also, detecting license plates might be slower because it is performed block by block from the frame while virtual trigger will just perform classification to the whole frame.

This Thesis consists of seven chapters. Chapter 2 gives theoretical background information about the principles of cameras: their structure and function and different parts of the camera. The focus is in the properties of the camera that affect image quality. Chapter 3 introduces the theory of machine learning focusing on convolutional neural networks that are the main part considering this work. Chapter 4 explains how the camera tests were implemented and which cameras were chosen to be tested and why. The implementation of the virtual trigger is presented in Chapter 5. This includes the data, the neural network models that were implemented and a short explanation of the programming tools that were used. The results of both camera and virtual trigger tests are collected to Chapter 6 with discussion about them. Chapter 7 concludes this work and gives some ideas about the future for both cameras and virtual trigger.

2. PRINCIPLES OF CAMERAS

Cameras are optical devices developed for capturing images and videos. Many different camera types have been developed, and they have slightly different functions. The simplest structure of a camera, a pinhole camera, which is introduced in Chapter 2.1, has been known since ancient times. Since the 12th century a useful way of using a lens in image formation has been known, but the photosensitive components for saving the image has been known only since the 17th century, still unable to use them properly. The story of the digital cameras began in 1975 in Kodak laboratories. [1]

The main idea of this Chapter is to introduce basic structure and functions, and the main parts of cameras, mainly focusing on digital cameras, and particularly on *digital single-lens reflex* (**DSLR**) *cameras*. First, the basics of camera functions, parts and focusing are introduced and then some details about aperture, shutter, image sensor, mirror and pentaprism are discussed. Camera interfaces are discussed in Chapter 2.6 and suitable camera types in the perspective of Visy Oy are considered in Chapter 2.7.

2.1 Basic structure and function

As introduced previously, in the simplest case, a camera is a box with a hole. This is called *camera obscura* or a *pinhole camera*. Figure 1 shows an example of a pinhole camera. Light comes through the hole and image is projected on the wall opposite the hole. For saving the image, the wall must be a film or an image sensor with a chemically processed surface. [2] The scale of the formatted image is the ratio between the depth of the box *d'* and the distance *d* of the object to the hole: $\frac{d'}{d}$. So, increasing *d'* or decreasing *d* will increase the image size and increasing *d* or decreasing d' will decrease the image size. The smaller the hole is, the sharper the image will be, but also the image will be darker. The smaller the hole is, the brighter the image will be, so one must find the balance between sharpness and brightness. [1] These properties are discussed later in Chapter 2.2 and 2.3. For increasing the amount of light, a lens is installed to the hole [2].

There are many different parts in a digital camera body. The main parts are *lens*, *aperture*, *shutter*, *image sensor*, *pentaprism* and *mirror*. Pentaprism and mirror are used for *viewfinder*, which is introduced in Chapter 2.5. This Thesis will concentrate on the main parts leaving the other parts of a digital camera body out of the scope. Figure 2 shows an example of DSLR camera parts.

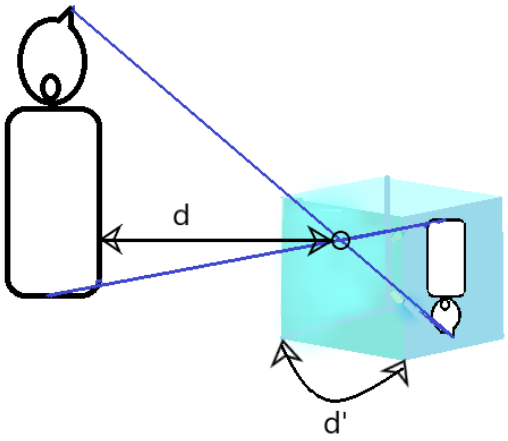

Figure 1: *An example image of a pinhole camera and image formation*

The lens gathers light and focuses the light rays from an object through the aperture to the image sensor (or, in some cameras, film). The image is formatted on the image sensor, reversed and turned upside-down, as shown in Figure 1. The aperture is usually adjustable and controls how much light is admitted to the camera's sensor in a certain time period. The shutter is opened when an image is taken, and the time that the shutter is open, and the image sensor is exposed to light, is called *exposure time*. Together, the shutter and the aperture control the *exposure*, i.e. the amount of light and the exposure time. If the light of the image is desired to stay stable, the shutter needs to be open longer with smaller aperture and vice versa. [2] A colour filter is used for showing the images in **RGB** (Red Green Blue) space.

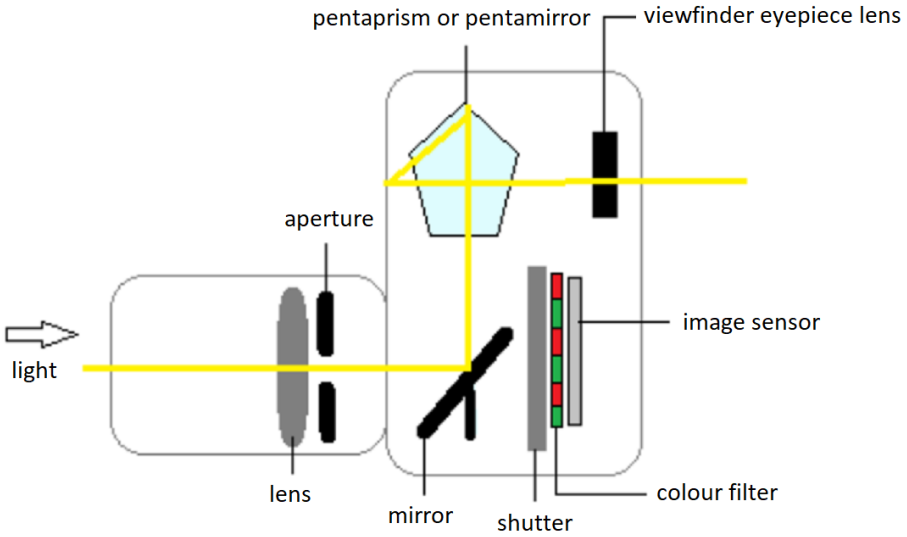

Figure 2: *Digital camera (DSLR) parts*

The lens that is added to the pinhole cameras enables more light but lacks the possibility of focusing. The objects closer the lens will be sharper, and the others are blurrier, depending on the lens. Because of the lack of focusing, DSLR cameras need a separate photographic objective, which means a lens or more commonly a system of lenses, to be able to function properly. [1] Figure 3 shows a simplified diagram of focusing. The *depth of field* (**DOF**) shown in Figure 3 is discussed in Chapter 2.2

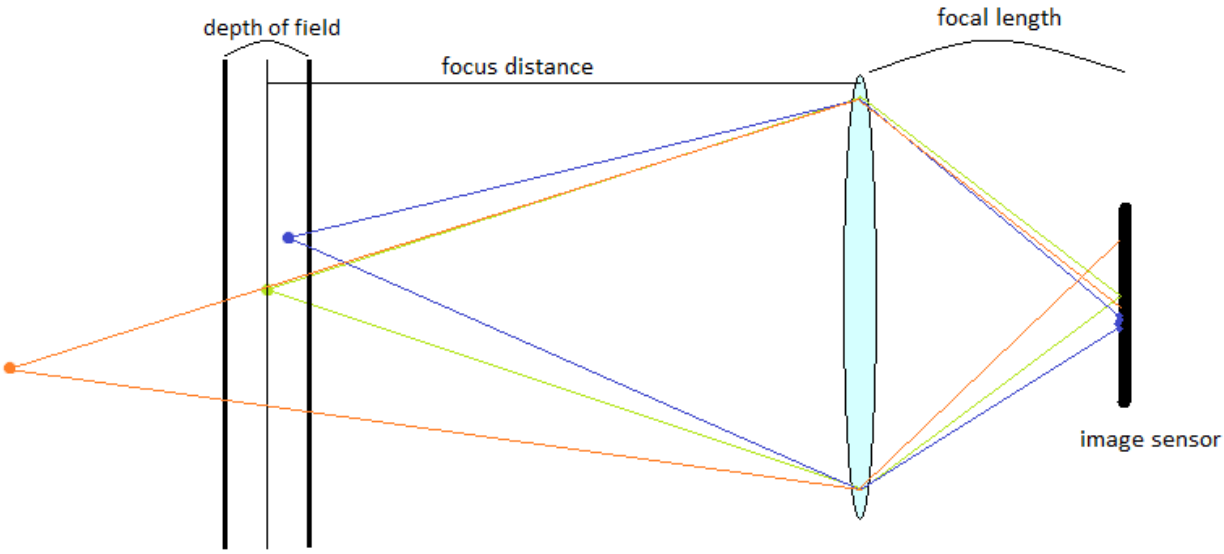

Figure 3: *A simplified diagram of focusing of cameras*

There are three colored points in Figure 3. Let's imagine an object to each of these points. The green one is perfectly in focus and it will be seen sharp on the image. The blue point is said to be marginally in focus. It is not totally out of the focus, but it is located in the acceptable area of depth of field (see Chapter 2.2), so it will be acceptably blurry. The orange point is out of focus so this object will be blurry in image.

2.2 Aperture

As discussed earlier, the *aperture* controls the amount of light in a certain time interval: the bigger the aperture, the brighter the image will be because of more light coming to the image sensor. The separate photographic lenses have their own aperture sizes, which affect the adjustability and the image quality of the camera (camera body + photographic objective). Let's now study the standard scale of the aperture sizes, called *f-numbers* or *f-stops*. [2] Some examples of these are shown in Figure 4 [3]. F-numbers are related to the properties of an optical system. In cameras, the explanation for the f-numbers can be started from the lenses.

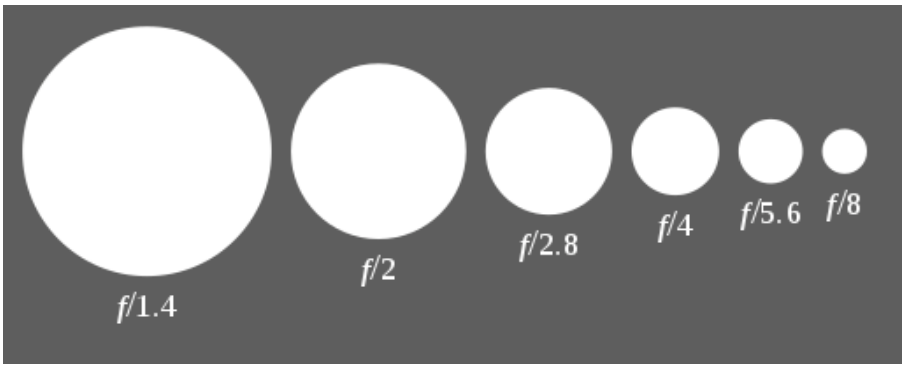

Figure 4: *Examples of standard aperture f-number scale* ***[3]***

Lenses are usually convex or concave and they have a focal point. *Focal length* is the name for the distance between the focal point and the lens. [4] In optical systems the f-number describes the ratio between the focal length of the system and the aperture diameter [5]. Let's mark the focal length as *l* and the aperture diameter as A, and we obtain the following formula:

$$f_{number} = \frac{l}{A}. \tag{2.1}$$

If the focal length is 28 mm and the aperture is 10 mm, the f-number is marked as f/2.8 or f2.8. The bigger the f-number, the smaller the aperture and therefore, the less light will be let in. Few DSLR cameras have the possibility to use the whole f-number scale, but only a range of it. Therefore, when choosing the camera and the photographic objective, it is important to check the available f-number values. [2]

Besides affecting the amount of light in a certain time period, the aperture size affects the depth of field. The bigger the aperture, which means a smaller f-number, the narrower the depth of field. The depth of field means the area, which is sharp in the image. So, the camera is focused only on the objectives at a certain distance. With a smaller aperture size, a larger depth of field is obtained. This means that a bigger area in front of and behind the focus point will be sharp in the image. [2]

The depth of field is demonstrated in Figure 5. The upper image in Figure 5 demonstrates the result with a bigger aperture causing a narrow depth of field. Lower image demonstrates the larger depth of field with smaller aperture. The depth of field is actually a result of four different parameters. One is the aperture (f-number), second is the focal length of the optical system, third is the object-to-lens distance, also known as focus distance, and fourth is the criterion chosen for sharpness, called the *circle of confusion*.

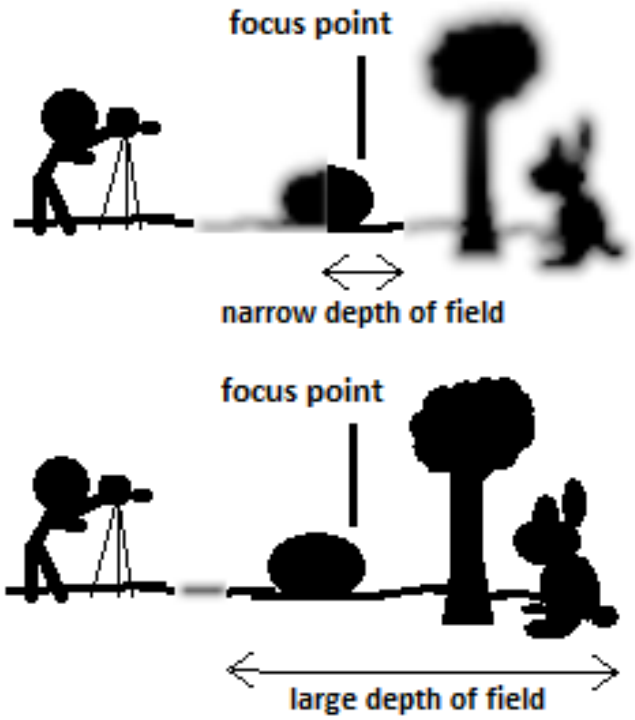

Figure 5: *Understanding the depth of field*

Circle of confusion means an optical spot, which is caused by light rays that are not coming to a perfect focus from the lens as seen in Figure 6. Let's mark the focal length as *l*, the focus distance as *D*, aperture as f_{number} and circle of confusion as *C*. The approximation for DOF is then

$$DOF \approx \frac{2D^2Cf_{number}}{l^2} \text{ [6]} \qquad (2.2)$$

given in meters.

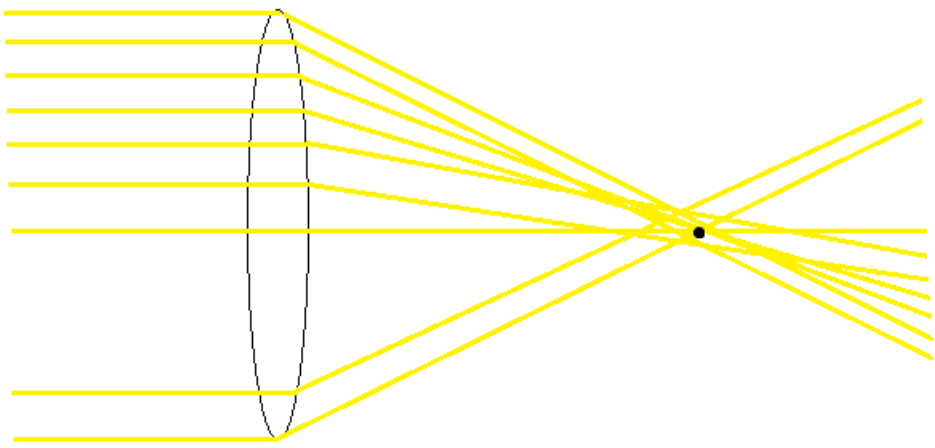

Figure 6: *Imperfect lens*

Let's calculate an example for DOF. If the focus distance is *D* = 5 m, circle of confusion is decided to be *C* = 3 m, the focal length is *l* = 50 mm and the f-number is 1.4, we get the following result for DOF:

$$DOF \approx \frac{2 * (5)^2 * 1.4 * 3}{(50)^2} \approx 0.084\ (m).$$

The black point in Figure 6 describes the focal point of the lens. Yellow lines describe the light rays coming to the lens and refracting when reaching the lens. As in Figure 6, in an imperfect real-life lens not all the light rays go through the focal point of the lens after refracting. This causes the blurry spots in the image.

2.3 Shutter

The *shutter speed* affects the *exposure time*. In addition to that, the shutter speed can be considered as the controller of motion. The two main types of shutters are mechanical and electronic. Mechanical shutter includes two types: a leaf shutter and a focal-plane shutter. The leaf-shutter comprises of overlapping metal blades that open and close by a spring. The focal-plane shutter is in general two overlapping curtains, which move to one direction over the film or the image sensor. The curtains work as an adjustable window, which expose the film or the image sensor one section at a time. [2] Rolling, global and hybrid shutter are electronic shutters. Rolling and global shutter are controlled by the sensor itself. Rolling shutter scans the image plane row by row and the exposure takes place in the time interval between the first and the last row illumination. Global shutter illuminates the whole image plane at the same time. Hybrid shutter combines mechanical and electronic shutter functions. Electronic shutters don't have any moving parts, which is an advantage since their operation is silent and there are no mechanical shutter parts that break easier. Another advantage of electronic shutters is their high shutter speed. [7]

As demonstrated in Figure 7 [8], a high shutter speed eliminates blurring from an image. This ability depends on the speed of the objects also. When the shutter speed is low or the object moves too fast compared to the shutter speed, the object moves before the image is fully formatted on the film or the sensor and this causes blurring. The blurring occurs more, when the object is moving horizontally in front of the camera than when the object is moving directly towards or away from the camera. [2]

Figure 7**: Shutter speed controlling blurring **[8]

Aperture diameter and shutter speed are related to each other: when the f-stop is increased by one (the aperture is smaller), the amount of the light is half of the previous value and, due to this, the shutter speed needs to be doubled. When the f-stop is decreased by one, the amount of light is doubled, and therefore, the shutter speed may be reduced to half.

2.4 Image sensor

Image sensor is considered as the most important part of the camera in terms of the image quality. Image sensor consists of pixels that are usually square. The common technology used in image sensors in DSLR cameras these days is complementary metal-oxide semiconductor (**CMOS**) transistor technology. This technology has almost fully replaced the charge-coupled-device (**CCD**) technology, which used to be the most common technology in DSLR cameras. The basic idea of image sensors is that first, the light rays are focused on the sensor. Image sensor converts light into an array of electrical signals. Usually, the sensor uses a *colour-filter array* (**CFA**) to make each pixel to produce a signal that corresponds to red, green or blue colour i.e. the pixels show the image in RGB colour space. The sensor itself does not produce colours, it sees only the black and white data and therefore a CFA is needed. A common CFA is a Bayer filter, which is demonstrated in Figure 8. One pixel does not store all the RGB values. The RGB values are stored according to the Bayer filter array. The analog pixel data i.e. the electrical signals are converted to digital with an analog to digital converter (**ADC**). Then a spatial interpolation operation is performed to form a full colour image and usually some further digital signal processing is used to improve the image. Interpolation completes

the image, which is formed with the Bayer filter. Finally, the image is compressed and stored to reduce the file size. [9]

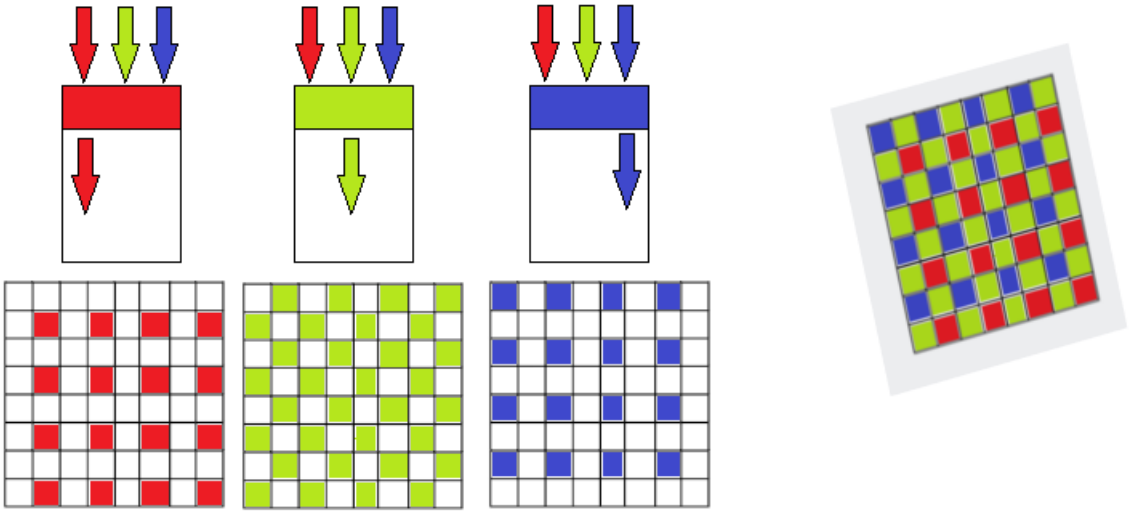

Figure 8: *A Bayer pattern colour filter*

In CMOS technology, each pixel of the image sensor contains a photodetector, which converts the light into photocurrent. Then the photocurrent is converted into voltage and readout or vice versa. The most general types of photodetectors in CMOS technology are reverse-biased PN junction photodiodes and PIN diodes. The photocurrent produced by photodetectors is usually too low, which means current from femtoamperes to pico-amperes. Therefore, in CMOS technology the current is first integrated as shown in Figure 9, and then read out. Figure 9 shows, that the voltage over the photodiode is reset to voltage V_{dd} after which the switch is opened and the current flow through the diode is integrated over the diode capacitance (C_d). [9] The PN junction photodiode is a semiconductor, which contains negatively charged electrodes (n-type region) and positively charged holes (p-type region) and those are fused together. Reverse-biased means that the voltage over the diode is negative, and the n-type region of the diode is connected to positive terminal of a source and p-type region is connected to negative terminal of the same source. The greater the light intensity, the smaller the diode resistance and therefore the greater the current. The PIN diode contains n-type and p-type regions and, a slightly doped semiconductor region between them. [10]

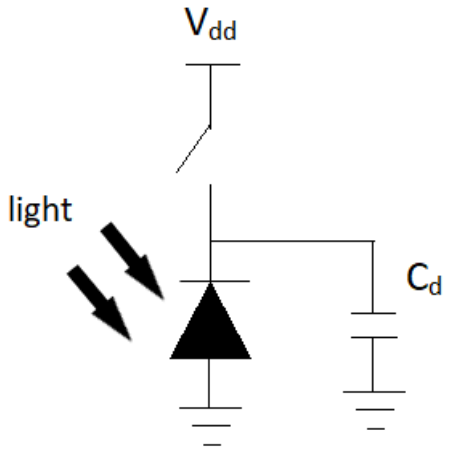

Figure 9: *An example of direct integration of the photocurrent*

There are three main readout technologies in CMOS image sensors. Different versions of active pixel sensor (**APS**) are the most common ones. This means a technology where each pixel contains a photodiode, one or more transistors and an amplifier, which makes the imaging process faster and increases signal-to-noise ratio (**SNR**). The photocurrent is first converted into voltage and then read out from the pixel array. Digital pixel sensor (**DPS**) means that each pixel contains a photodiode, few transistors, an ADC and some memory for temporary storage of the digital data. So, the current is changed into digital data and then read out from the pixel. Passive pixel sensor (**PPS**) is the oldest one of the main readout technologies. In PPS, each pixel contains only one diode and one transistor, and the current is first read out and then changed into voltage. [9]

Image sensor size affects the image quality. The bigger the image sensor, the bigger the resolution and the more detailed the image if the sensor contains more pixels. This means also better image quality and larger image area. If the image sensor size increases, but the number of pixels stays the same, the aim is to have bigger pixels and a better *dynamic range*, which is introduced later. Figure 10 [11] shows different sensor sizes. Some camera manufacturers have own names for specific image sensor sizes, but usually image sensor sizes are expressed in inches. Figure 10 shows both: sizes in inches and few examples of sizes of camera manufacturers. The full frame image sensor is currently the largest available in basic consumer cameras. It is 36x24 mm. [12]

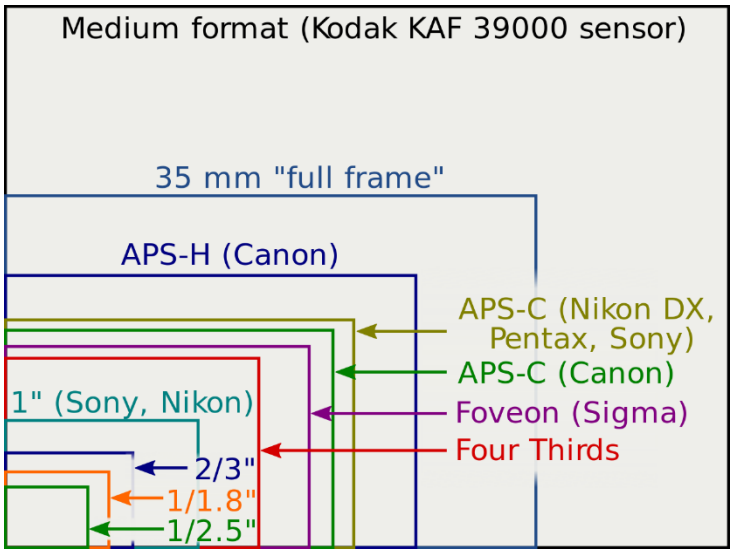

Figure 10: *Different sensor sizes* ***[11]***

One important character of an image sensor is called *light sensitivity* or ***ISO*** (International Organization for Standardization) *speed*. It is a camera setting controlling the brightness of photos. The higher the ISO speed, the brighter the photo. The brightness is controlled by amplifying the output signal, and therefore the image quality does not necessarily improve with a higher ISO value since the noise in the photo will also increase when the ISO number increases. ISO value should only be used if the photo cannot be brightened with aperture or shutter instead. When adjusting aperture, shutter and ISO number it is possible to keep the exposure time the same. With smaller aperture, the shutter time needs to be increased, or, if those need to be stable, then the ISO number is increased. [13]

Related to the characters introduced above, *dynamic range* (**DR**) of a camera is an important term regarding photography and image quality. It describes the ratio between the maximum and the minimum light intensities at each ISO stop. The maximum light intensity or signal is at the pixel saturation point, and the minimum light intensity is the noise floor of the signal. The sensor pixel size affects the camera's dynamic range. Because only a certain number of photons fit to the area of a pixel, smaller pixels have a smaller dynamic range and DR is defined by dividing the maximum number of photons in the area of one pixel by the minimum amount, which is one. In real life, it is not possible to count the actual number of photons. For example, f-stops, which were introduced in Chapter 2.2, can be used as a measure for the DR of a camera. In this case, increasing the DR by one stop means doubling ratio between maximum and minimum light intensity, and therefore twice the details in dark and light areas can be seen. [14]

2.5 Viewfinder

A *viewfinder* allows the photographer to check the cropping and the focus of an image before taking it. There are mainly two technologies: an optical viewfinder and an electronic viewfinder, which is a liquid crystal display (**LCD**). There may also be an additional LCD in the cameras, especially if the viewfinder is optical. In these cases, the extra LCD screen is meant for live view and for showing the image right after taking it. [2]

The optical viewfinder is more common in DSLR cameras than electronic and it is implemented with a mirror and a pentaprism. Before the shutter is opened, the light reaches the mirror instead of the image sensor. After the light rays reach the mirror, the mirror reflects them to the pentaprism. The pentaprism turns the image to the correct position. This means that the photographer sees the actual image, which will be saved by the image sensor, through the optical viewfinder. [2] When the shutter button is pressed the mirror rises to let the light rays to expose the image sensor. [15]

In the electronic viewfinder, the image is electronically projected into the small LCD. Therefore, the view is not exactly the same as it will be on the image sensor, but a projection of it. [2]

2.6 Camera interfaces

For this project, it is important to know, what kind of *interfaces* the cameras have, concerning the software and hardware interfaces. The power cabling for different cameras is different. Some cameras work with a Power over Ethernet (**PoE**) cable, some work with a VDC power cable and some need a battery. Some of the cameras may have several options for cabling and a battery system. For example, Basler IP cameras can be powered with a PoE cable, but also with a 12 to 24 VDC cable [16]. DSLR cameras like Canon's are powered with a battery, but in some of their cameras, the battery is replaceable with an AC power adapter and a DC power connector [17].

If the camera is an IP camera, it needs a network connection also. Those cameras are connected to Internet via Ethernet cable. In these cases, the camera usually has a web user interface (**UI**), from where the settings are configured. Some cameras may also have wireless connection possibilities, for example, wi-fi and Bluetooth connections. Wi-fi and Bluetooth connections may also be used for accessing the camera from a mobile phone or a laptop. The camera may be connected to a laptop or a computer with a **USB** (universal serial bus) cable for, e.g., transferring images. This is a common way in DSLR cameras.

Software interfaces are important, when one needs to control the camera automatically from a PC and transfer and process images automatically. Some cameras are accessed from a computer through a web UI and some camera manufacturers have made their own application that needs to be downloaded and installed before using the camera remotely. Some manufacturers have also made a software development kit (**SDK**) for controlling and using the camera remotely. Video surveillance cameras are usually IP cameras, and therefore, there is usually a web UI, which allows watching a live image from the camera and changing the camera settings. For accessing the camera through a web UI, an IP address is needed. It is usually obtained through a finder program provided by the camera manufacturer.

In IP cameras, still images are obtained by using hypertext transfer protocol (**HTTP**) for transferring information or by using real-time transport protocol (**RTP**) and capturing frames from the video stream. HTTP is a request-response protocol, where the HTTP request is sent from a client to a server. The server sends a response, which in this case is an image. RTP is a protocol for transferring real-time data, for example, images and audio, over IP networks.

2.7 Properties of suitable camera types

Above we introduced different camera types and their properties. When considering this Thesis and the requirements that Visy Oy has for the cameras, next we figure out, which kinds of cameras would be suitable for the future projects.

First, the aim is to have a higher image quality, so the image sensor needs to be large and contain at least 10 megapixels. The pixel size needs to be also big for obtaining a high dynamic range and due to this, high image quality. The type of the sensor is not that important, but the size of it.

From two different shutter types, an electronic shutter would be better when compared to mechanical. In Visy projects the cameras may take thousands of images per day, which would most likely consume a mechanical shutter more than an electronic one.

A suitable camera needs to be powered with a PoE or a VDC power cable. In Visy projects the camera must take images all the time every day so loading a battery is not possible, and because automation is the purpose of Visy's projects, changing the battery by people would not be desired.

There has to be a way to control the camera remotely and automatically, because our software needs to be able to tell the camera, when to take an image. Also, related to the

remote control, a possibility to use HTTP for transforming images is considered as an advantage. This is not critical, if there is another fast way to transfer the images from the camera to the computer, but since Visy Oy has already implemented code for HTTP image transferring, it would make it easier for us to use HTTP also in the future. Also, if the camera has a web UI for changing settings, it is considered as an advantage. A web UI enables configuring the camera remotely, which we currently do in projects.

3. MACHINE LEARNING THEORY

Machine learning consists of algorithms and statistical models for automated data analysis methods. The idea is to implement mathematical models that can learn from the data by detecting patterns and using these patterns to predict from new data. For example, classification, regression and feature learning are types of machine learning methods. The idea of machine learning is to find an output Y for some input variable X in $\mathbb{R}^{mxn}$. This can be written mathematically as following:

$$F: \boldsymbol{X} \rightarrow Y, \tag{3.1}$$

where **X** is a matrix of the input variables and Y is an output variable. The aim is to find the function F that best maps **X** to Y. One input for a machine learning model is x_i and the model will produce an output Y by computing it from the input with a function F.

The models of machine learning are trained with *data*. Data can be, for example, images, audio or data points and the dataset used for training a machine learning model is called *training data*. Three main types or training are *supervised, semi-supervised* and *unsupervised* training. [18] These are introduced later, especially supervised learning, which is in the focus of this Thesis. There are also other types of learning, for example, reinforcement learning, but those are out of the focus of this Thesis, so they are not discussed.

Supervised learning is the most common concept of machine learning [19]. In supervised learning, the model is trained with a labelled or a classified training dataset. Labelling or classifying the training data is called *annotating* it. Annotating is an important part of the machine learning process because in order to obtain good results, there must be a large amount of annotated data and annotations need to be correct. The training data is usually a vector of inputs **x**. It is shown to the model one by one with the desired output Y. According to this knowledge, the model is supposed to find parameters that map the input **X** into the desired output Y. The more inputs the model is able to map into correct outputs, the better the found parameters. So, supervised learning means that the model is trained with a set where the correct output is known for each input and during the learning process, the model modifies its parameters in order to obtain better results. [20]

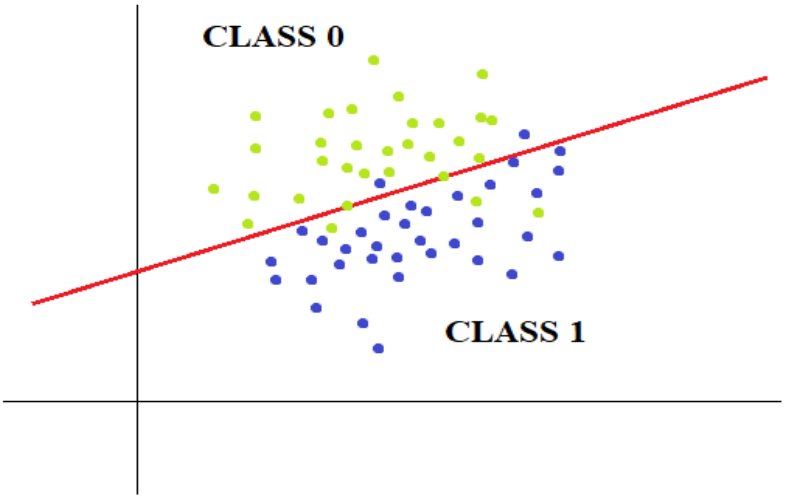

Figure 11: *An example of linear regression*

Classification and regression are types of supervised learning. In classification, the outputs belong to a limited set of values, and the target outputs are categorical. For example, in binary classification problems there may be two possible output classes like cats and dogs, and they can correspond to, for example, classes 0 and 1 in the model. In regression, the outputs may be any numerical values in a range and the model tries to find thresholds or boundaries to divide the data. [18] Figure 11 shows a 2D linear regression example with two classes. Green points belong to one class and blue points belong to other class, for example, classes 0 and 1. Red line is the result of linear regression, called a decision boundary. It is a function $f = w_a x + w_b$ and the parameters, also known as *weights* w_a and *bias* w_b are chosen to be the ones that produce a decision boundary that separates the two classes most accurately. Whenever a new point is added to the samples, it can be classified with the found decision boundary.

In *unsupervised learning*, the dataset is not annotated, so the correct outputs for the given inputs are not known. Instead, the model is supposed to divide the dataset into outputs itself by studying the features of the data. Unsupervised learning tasks can be divided into clustering, density estimation and visualization problems. The most common one is clustering, where the data is organized to groups by the features of it. The model finds the commonalities or their absence and groups or, in other words, clusters the data according to those. In density estimation, the model tries to find the density distribution of the data. The aim in visualization problems is to project high-dimensional data into two or three dimensions to be able to visualize it. [18]

Semi-supervised learning combines supervised and unsupervised learning methods. A part of the training data is labelled. In semi-supervised learning problems, there is usually

a large amount of training data available, but annotating is not possible, or it is too expensive, which leads to training the model first with supervised methods and continuing with unsupervised learning. [21] However, this Thesis concentrates on supervised learning methods, and, more specifically, on *deep convolutional neural networks*.

First, this Chapter introduces artificial neural networks generally. Then single-layer perceptron, multi-layer perceptron and convolutional neural networks are presented. Finally, training and evaluating neural network models is discussed. Model evaluation includes presenting error metrics, overfitting, cross-validation, data augmentation and regularization.

3.1 Artificial neural networks

Artificial neural networks (**ANN**s) are a group of machine learning methods. The basic component of ANNs is an artificial neuron, which is loosely based on the biological neuron. Figure 12 [22] shows one biological neuron that would be connected to another from the axon terminal. Electrical signals are transmitted from neuron to another via axons.

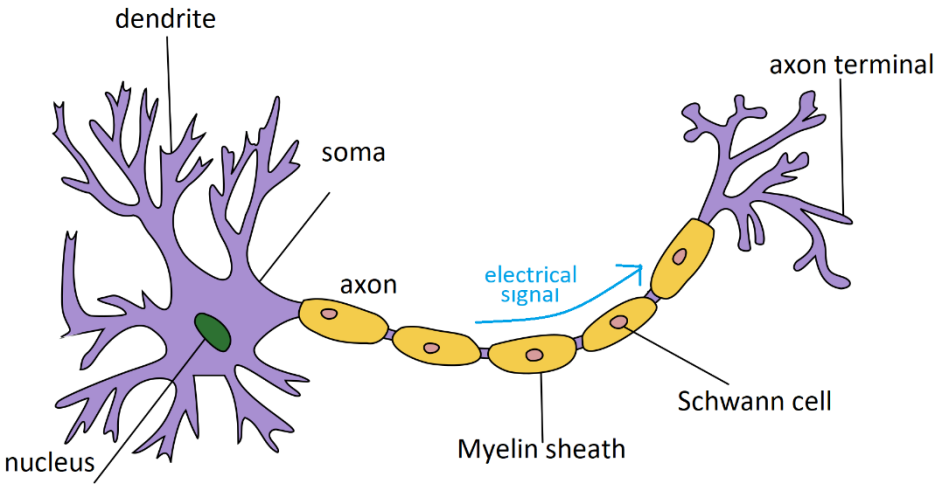

Figure 12: *Biological neurons, modified from* ***[22]***

The artificial neural network itself is not an algorithm. It is a structure for different machine learning algorithms to learn patterns from the data and produce desired target outputs. The simplest type of a *feedforward* artificial neural network is a *perceptron*. It was invented in 1960s by Frank Rosenblatt to solve binary classification problems. The function of only one neuron is called a single-layer perceptron, and it is introduced below. A feedforward network means that the data is only transmitted to one direction and the neurons do not form cycles. [18]

3.1.1 Single-layer perceptron

Single-layer perceptron is presented in Figure 13. Inputs **X** are fed to through weights W and the outputs Y are calculated as a sum of dot products of the weights and the inputs as the following formula shows.

$$y = f(\sum_{i=1}^{N} w_i x_i + b), \tag{3.2}$$

where b is a bias for shifting an *activation function f* and w_i describes the *i*th weight of W. The output *y* is the resulting class from two options. Single-layer perceptron can only be applied to linearly separable data and only in the case of binary classification. If the problem is more complicated or includes more classes, the single-layer perceptron needs to be developed into a non-linear *multi-layer perceptron*, which is introduced later in Chapter 3.1.2. [23]

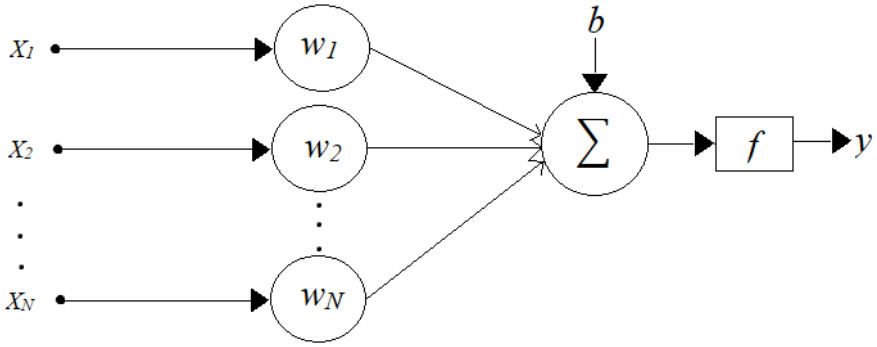

Figure 13: *A simple single-layer perceptron, also known as a neuron*

The purpose of the activation function is to produce a *decision boundary*, also known as a *threshold*, which defines the resulting class. If the output exceeds the threshold, the neuron is activated and if it the threshold is not exceeded, the neuron is not activated. [23]

Rosenblatt's algorithm defines the non-linear activation function is as a step function:

$$f(s) = \begin{cases} 1, if\ s \geq T \\ -1, if\ s < T \end{cases} \text{ [23].} \tag{3.3}$$

Rosenblatt's theorem says that a perceptron can learn anything that it represents or simulates. However, the Rosenblatt's theorem is already out of date, so to say, and the activation function can also be something else, when the Rosenblatt's theorem does not even hold up anymore [23]. Rosenblatt's theorem is only introduced in this Thesis, because it is a simple example and introduction to machine learning, artificial neurons and ANNs and it makes it easier to understand the rest of this Chapter.

3.1.2 Multi-layer perceptron

A *multi-layer perceptron* (**MLP**) consists of multiple layers of neurons [24]. A multi-layer perceptron is shown in Figure 14. The Figure shows an example of N inputs and C outputs.

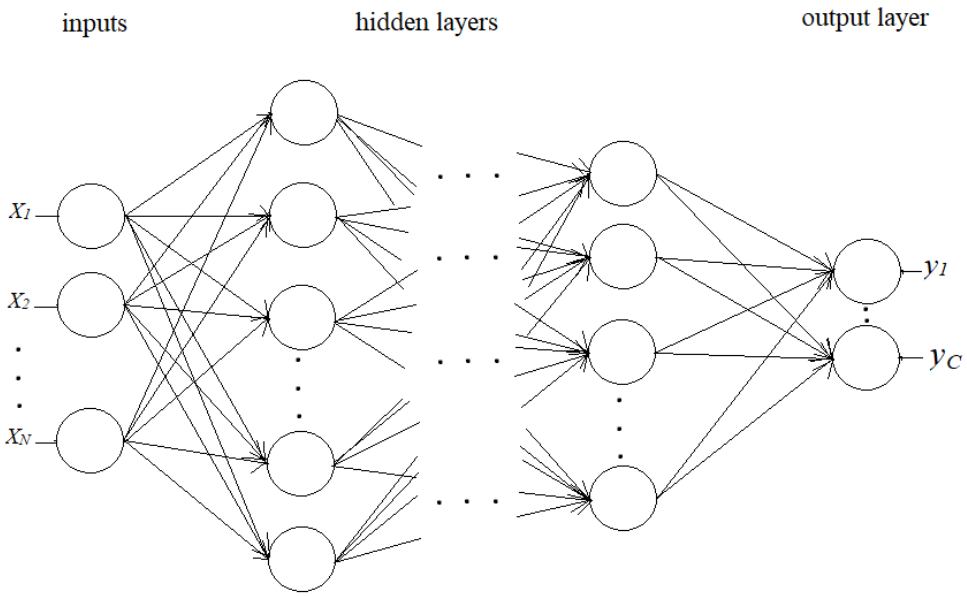

Figure 14: *Multi-layer perceptron*

The neuron layers between input and output layer are called hidden layers. Hidden means that they have no contact with the outside – the input data is given to the first layer and the output layer gives the results. The layers in a multi-layer perceptron are fully connected which means that each neuron in one layer is connected to every unit on the subsequent and previous layer. Multi-layer perceptron is a feedforward network. [25]

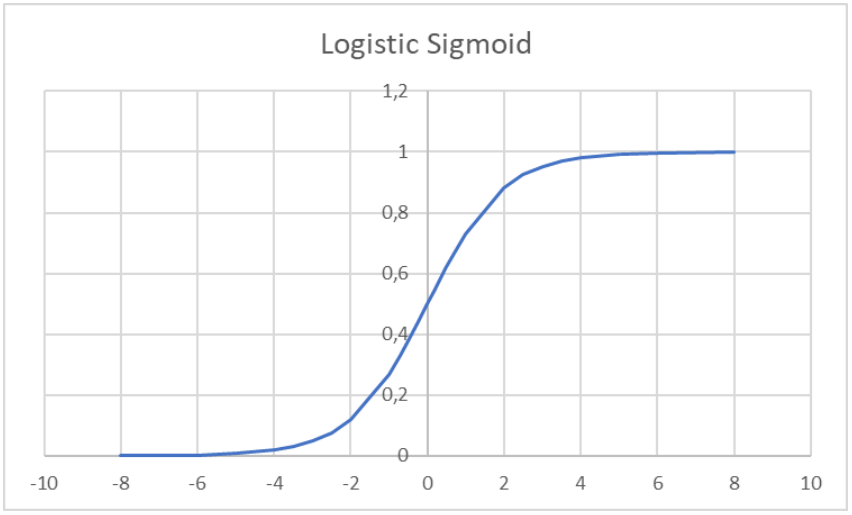

Figure 15: *Logistic sigmoid curve*

Multi-layer perceptron needs also an activation function to decide whether a neuron is activated or not. In multi-layer perceptron, commonly used activation functions are called *logistic sigmoid*, *hyperbolic tangent* and *rectified linear unit* (**ReLU**). They are all mathematical functions that are applied to machine learning. [25]

A non-linear logistic sigmoid function is shown in Figure 15, and it is defined as following:

$$\frac{1}{1+e^{-s}}, \tag{3.4}$$

where $s = \sum_{i=0}^{N-1} w_i x_i + b$ [23].

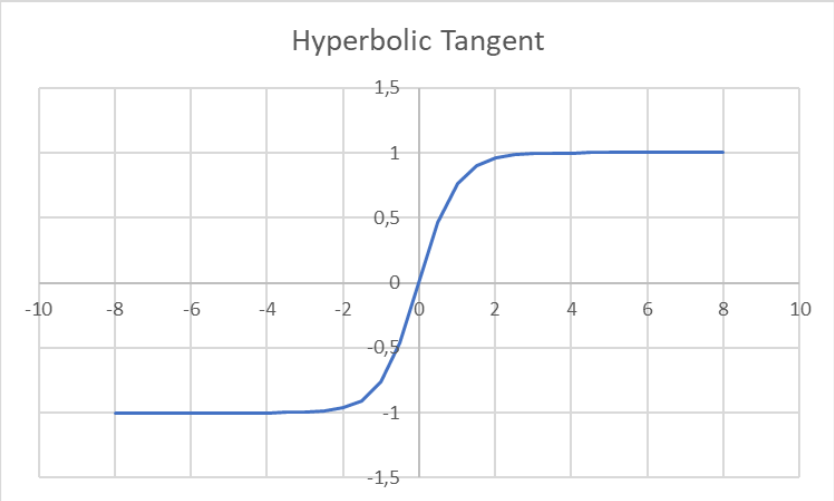

Figure 16: *Hyperbolic tangent*

Hyperbolic tangent in Figure 16 is defined as:

$$\tanh(s) = \frac{\sinh(s)}{\cosh(s)} = \frac{1-e^{-s}}{1+e^{-s}}. \text{ [26]} \tag{3.5}$$

Definition for *s* is as above. Rectified linear unit curve is shown in Figure 17.

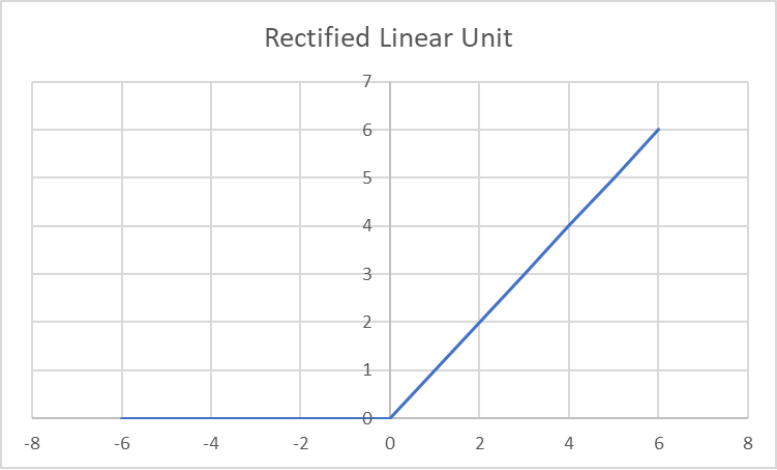

Figure 17: *ReLU curve*

ReLU is congruent to a half-wave rectifier in electronics. A half-wave rectifier circuit includes a diode for allowing the current to only flow to one direction. This means that the current to the other direction is 0 and to the other some value *I*. [27] The definition for ReLU is

$$\mathrm{x} \rightarrow x_+ = \max(0, s) \quad [28] \tag{3.6}$$

and *s* is defined as above.

3.1.3 Convolutional neural networks

Convolutional neural networks (**CNN**s) are deep neural network variations of the multi-layer perceptron structure, using a convolution instead of a general matrix multiplication at least on one layer. In difference to multi-layer perceptron, CNNs have sigmoidal non-linearity in hidden layers whereas MLP has step-function non-linearities. [18]

The convolution itself is mathematically written as the following formula

$$s(t) = (w * x)(t), \tag{3.7}$$

where *x* and *w* are functions describing the input (*x*) and the function modifying the shape of the input (*w*). In this work, we focus on CNNs, and therefore we are interested in the discrete case of convolution. This is written for a 1D input vector **x** as

$$s(n) = \sum w(t-n)x(n), \tag{3.8}$$

where *x(n)* is the *n*th value of vector **x** and *w* describes a so-called *kernel* in CNNs. The kernel, in this case, is also a 1D vector, and the values of it are adjusted by a learning algorithm. [25]

Convolutional neural networks typically consist of pairs of one convolutional layer with activation function and one pooling layer. The last layer or layers are usually fully connected layers. Figure 18 shows an example of this kind of typical structure of a CNN, but also other kinds of structures have been proposed to improve the performance of the networks. [24] The example of CNN in Figure 18 is for image classification.

A convolutional layer computes the convolution of the inputs and the kernel, and produces a feature map, which is passed to the subsequent layer. Neurons in one convolutional layer are organized into these feature map planes. The neurons in the same plane use the same weights. Each neuron of the convolutional layer takes a subregion of the input (e.g. image). This input area for the neuron is also called a receptive field. In a fully connected layer, each input is connected to each neuron and the receptive field is the entire previous field. [24]

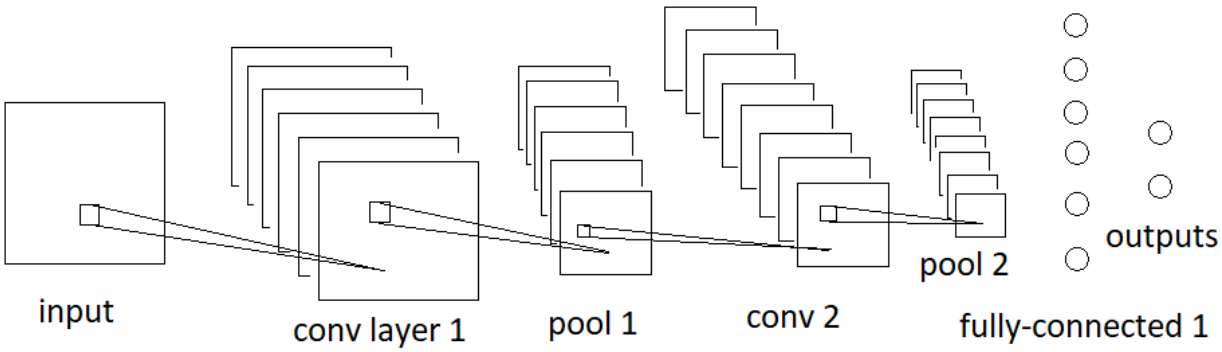

Figure 18: *An example of a convolutional neural network architecture*

CNNs are especially used for processing 2D grid-like data, like images, which are also the focus of this project. Images can also be considered as 3D data, if colour channels are added. Next, we want to apply the Formulas 3.7 and 3.8 to get a convolution for 2D cases. When the input is a multi-dimensional array, we also want to use a multi-dimensional kernel. For a grayscale image of the size *M x M* with an *N x N* kernel, the output feature map for one neuron in convolutional layer is computed as

$s_{u,v} = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} w_{i,j} x_{i-u,j-v}$ [25]. (3.9)

The resulting feature maps are passed through a non-linear activation function. Commonly used activation functions have been logistic sigmoid and hyperbolic tangent. Recently, ReLU has become a popular option for the activation function, but also other activation functions are used. [24] ReLU, logistic sigmoid and hyperbolic tangent were introduced in Chapter 3.1.2.

If we follow our example in Figure 18, the next step is a pooling layer after the feature maps are passed through an activation function. The pooling layer performs a down-sampling to the feature maps. Each unit of the pooling layer takes an *N x N* disjoint block of the feature map and reduces that to one single pixel. Two different examples of how pooling can be performed, are presented in Figure 19. These are called max pooling and average pooling. Max pooling on the left chooses only the maximum value of the block and passes that to the next layer. In average pooling, the average of the block is computed and that is passed on. [24]

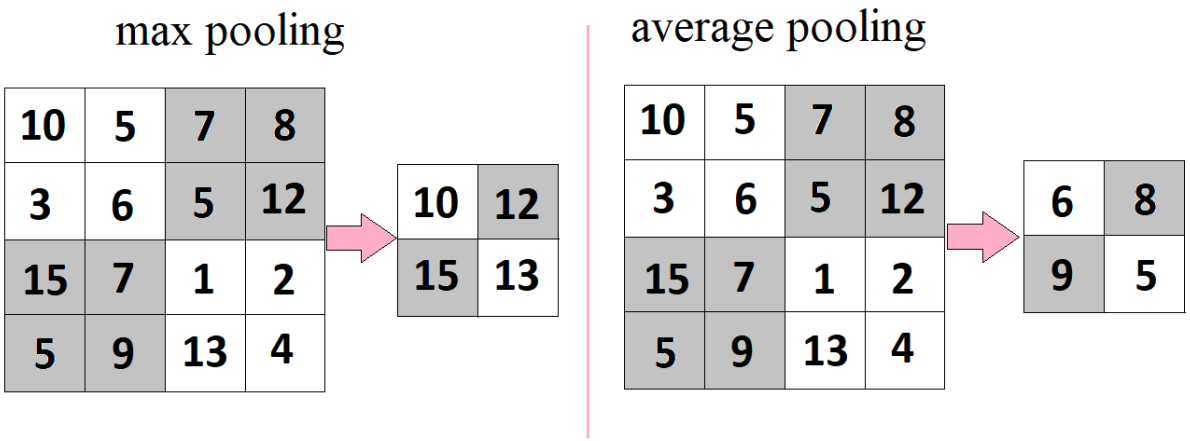

Figure 19: *Examples of how max and average pooling are performed*

In our example, the last layer is a *fully connected layer*. It means that each neuron on this layer is connected to each unit of the output of previous layer. In CNNs, the high-level reasoning happens in fully connected layer and this is where the feature maps are interpreted. A fully connected layer is followed by an *output layer*, which includes as many neurons as there are possible output targets. In the example in Figure 18 we have two neurons in the output layer corresponding to two possible output classes. Each input image is classified into one of these. [24]

3.2 Network training

In general, *training* a deep convolutional neural network means using *learning algorithms* for adjusting the free parameters of the network model, meaning the weights and the biases [24]. The weights are related to the convolution, which is computed in each convolutional layer. The convolution was introduced in Chapter 3.1.3. This Thesis focuses on image classification and for that purpose, the convolution is performed with the Formula 3.9, where w_i describes the ith weight. The neurons in the same feature map plane (introduced in Chapter 3.1.3) use the same weights for calculations.

For training a deep CNN model with supervised learning, the training data needs to be annotated. For image classification training dataset means a set of classified images. In the first phase of training, the weights and biases are initialized randomly, and the training data is fed to the network model. The network with randomly initialized weights and biases processes the data and produces the result vector $\hat{\mathbf{y}}$ consisting of the predicted outputs. This phase is called forward propagation. Because of annotated training data, the desired vector $\mathbf{y}$ with the correct output labels is also known. Therefore, it is possible to calculate the difference between $\hat{\mathbf{y}}$ and $\mathbf{y}$. [25]

For minimizing the difference, also known as the error, between $\hat{\mathbf{y}}$ and $\mathbf{y}$, a so-called *cost function J(w)* is needed. The cost function measures the performance of the neural network model. Commonly used cost function is, for example, a mean squared error (**MSE**):

$$J(w) = \frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2, \tag{3.10}$$

where *N* is the number of samples, y_i the *i*th correct label and $\hat{y}_i$ the *i*th predicted label. The labels are in image classification problems integers each corresponding to one class. Another cost function used in DNNs is a mean absolute error (**MAE**):

$$J(w) = \frac{1}{N}\sum_{i=1}^{N}|\hat{y}_i - y_i|, \tag{3.11}$$

where the variables are defined as above. A third example, with the same variable definitions, of cost function options is the *cross-entropy function*, which is given as

$$J(w) = -\sum_{i=1}^{N}[y_i ln\hat{y}_i + (1 - y_i)\ln(1 - \hat{y}_i)]. \text{ [18]} \tag{3.12}$$

The next step after deciding a cost function is to perform *backpropagation.* This starts with an optimization problem: we want to minimize the cost function. In backpropagation, the gradients for all the outputs in the previous layer are computed. These gradients then show, how much and to which direction the adjustable parameters of the neural network affect the cost. The weights and the biases are then updated according to the result of the cost function. [18]

Backpropagation is performed by using the chain rule of calculus and an optimization algorithm. The optimization algorithm is used for computing the gradients. Most of the deep learning optimization algorithms are based on an algorithm called *stochastic gradient descent* (**SGD**) [25]. It is a stochastic approximation of a gradient descent algorithm, which is a first-order iterative algorithm defined as

$$w^{\tau+1} = w^{\tau} - \eta\nabla J(w^{\tau}). \tag{3.13}$$

On each parameter update, a step η, also known as the *learning rate*, towards the negative gradient is taken. After each update, the gradient is re-evaluated. [18]

In gradient descent, the cost function is defined to the whole dataset. Therefore, the whole dataset is processed at once and the adjustable parameters are only updated according to that. These methods are called *batch methods.* In SGD, only a *mini batch* of samples is processed at once and the gradient is computed for that as following:

$$w^{\tau+1} = w^{\tau} - \eta\nabla J_n(w^{\tau}). \text{ [18]} \tag{3.14}$$

In order to find the minimum of the cost function the gradient is computed and a step towards the negative gradient is performed. It is good to keep in mind that finding a

minimum of the cost function does not necessarily mean that it is the global minimum, because usually the cost functions have many local minimums in addition to the global minimum. In most cases, however, the local minimums will give results close enough to the global one [19] and therefore, the training of the neural network is stopped whenever a minimum, local or global, is found.

3.3 Model evaluation

While and after the training of a CNN model, it is important to validate and test it. The aim is that the model *generalizes*, which means that it successfully classifies unseen data (data outside the training set) in the future [25]. Therefore, it is reasonable to split the training set into three different datasets in the training phase. These sets are called *training*, *validation* and *test sets.* Training set is used for training the model. Validation set is for testing during the training, how well the adjustable parameters work and when the model's performance does not improve anymore, and it is no use to continue training. Test set is used after training for testing the generalization of the model. Test set can also be collected separately and outside the training set, but the main point is that it consists of data that is not yet shown to model during training.

The purpose of this Chapter is to introduce ways to measure and improve the performance of a deep CNN model, and finally to discuss about a common problem in network training called *overfitting*. The more we have the data, the higher the performance of the model usually is [29].This can be performed with an algorithm called *cross-validation*, which is introduced in Chapter 3.3.3. Another way to gain more data is *data augmentation*, which is discussed in Chapter 3.3.4. For evaluating the model, Chapter 3.3.1 introduces *error metrics*. Chapter 3.3.2 discusses about overfitting. Finally, Chapter 3.3.5 presents *regularization* and *dropout*, which are used for preventing overfitting along with data augmentation and cross-validation.

3.3.1 Error metrics

Error metrics are used to measure the performance of the deep CNN models and with them, different models can be compared to each other and the model with the highest performance can be chosen. *Accuracy* is one metric for measuring the performance of a neural network model. It simply provides the proportion of how many samples were predicted correctly out of the total number of samples. *Error rate* is the opposite metrics for the accuracy. It presents the proportion of incorrectly predicted samples. [25]

When evaluating the deep CNN models, it is reasonable to think about what kind of errors are acceptable and which errors should not occur at all. If we take a cancer detector example, it is far more dangerous to obtain an output class 'no cancer', when there is cancer than getting a result 'cancer', when there actually is not cancer. In our virtual trigger case, it is also better to get *false positive* (**FP**) results, which mean that the classifier gives an output 'car' when it should give 'no car' than *false negatives* (**FN**) results meaning an output 'no car' when there actually is a car. In the false negative virtual trigger case, the vehicle would not be able to pass since it is categorized to be a 'no car' and therefore no license plate recognition and permit check is performed for it. The results for four different error metrics are collected to a confusion matrix shown in Table 1. These include FP and FN, but also the positive cases, where the output is correct, *true positive* (**TP**), when a car is classified to a 'car', and *true negative* (**TN**), when a no car is classified to 'no car'. [30]

Table 1: *Confusion matrix*

	predicted positive	*predicted negative*
true positive	TRUE POSITIVE	FALSE POSITIVE
true negative	FALSE NEGATIVE	TRUE NEGATIVE

Based on the confusion matrix, it is possible to compute *true positive rate* (**TPR**) for the deep CNN. This is also called *sensitivity* or *recall* of the model, because it describes how many of the samples belonging to the positive class have been classified correctly. It is computed as

$$TPR = \frac{TP}{TP+FN}. \text{ [30]} \quad (3.15)$$

True negative rate (**TNR**) describes how many of the samples of negative class (in virtual trigger case 'no car') have been predicted correctly by the CNN model. TNR is also called the specificity of the model, and the formula for computing it is

$$TNR = \frac{TN}{FP+TN}. \text{ [30]} \quad (3.16)$$

False positive rate (**FPR**) indicates how many of the samples belonging to negative class (in virtual trigger 'no car') have been falsely predicted to the positive class ('car'). FPR is computed with the following formula:

$FPR = \frac{FP}{FP+TN}$ [30]. (3.17)

False negative rate (**FNR**) shows how many of the samples belonging to the positive class ('Car') have been predicted to the negative class. The following formula gives us FNR:

$FNR = \frac{FN}{TP+FN}$ [30]. (3.18)

One other useful measure for indicating model performance is *F1 score*. It is defined to be the harmonic mean of recall and *precision*. Precision is given as

$p = \frac{TP}{TP+FP}$ [31] (3.19)

and F1 score is computed as

$F1 = \frac{2}{\frac{1}{TPR}+\frac{1}{p}} = \frac{2TP}{2TP+FP+FN}$ [31]. (3.20)

We can now write the accuracy of the model, which was defined in the beginning of this Chapter, with the parameters introduced above as

$ACC = \frac{TP+TN}{TP+TN+FP+FN}$. (3.21)

A *receiver operating characteristics* (**ROC**) *analysis* is developed for measuring the performance of a deep CNN model. ROC analysis is based on measures introduced above: a ROC graph is a plot of TPR on y-axis against FPR on x-axis. The shape and location of the ROC curve indicate the performance of a classifier. [30] Examples of ROC curve shapes and locations are shown in Figure 20.

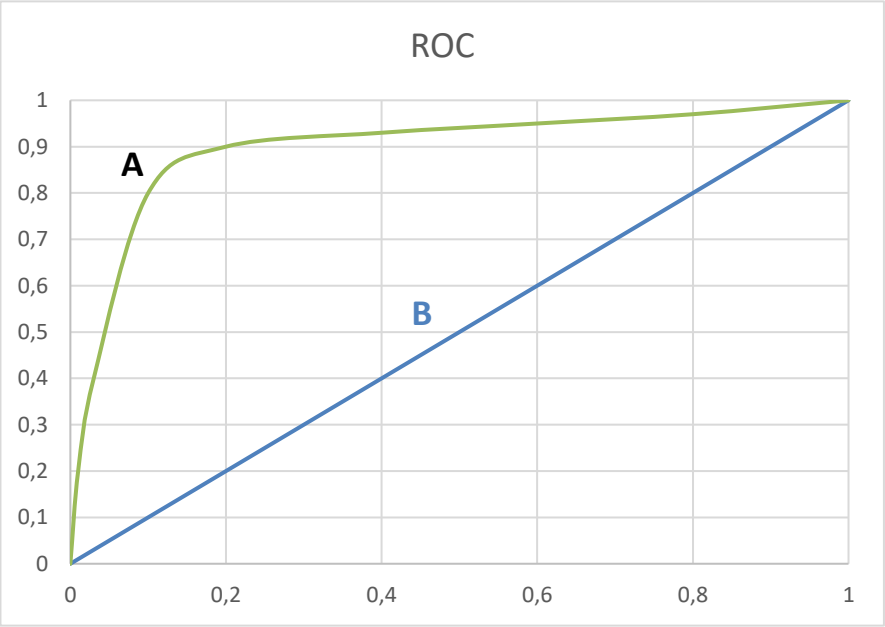

Figure 20: *Examples of ROC curves*

Figure 20 introduces two different examples of ROC curves on a ROC graph. Curve B is located on the diagonal of the graph. The classifier with a ROC curve like this is a random guesser with an accuracy of 50 %. So, the performance of a model with a ROC curve like B is not very high. The location for an ideal classifier would be at the point (1,0), which means that TPR = 1 and FPR = 0. Curve A follows the shape of an ideal ROC curve.

Area under curve (**AUC**) is a measure for comparing different ROC curves in numerical form. For the diagonal ROC curve AUC = 0.5, and therefore, any reasonable classifier would have an AUC close to 1.0. An ideal classifier would have AUC = 1.0.

The original ROC analysis is developed for two-class classification problems [30], and since the aim of this project is to implement a two-class classifier, this Thesis introduces only the basic ROC analysis and will not go deep into the multi-class applications of ROC.

3.3.2 Overfitting

The problem that may occur in training the deep CNN models is called *overfitting.* Overfitting means that the model learns the training data too well, so to say, and it begins to learn also the noise in the training data in addition to the actual signal. This also leads to a model that does not generalize. It may classify the training data perfectly but fails in classifying the test data.

Many ways to reduce overfitting have been proposed. One way is to keep the CNN model simple. The complexity of the model should correspond to the number of the training data available [32], and therefore a model that is too simple or too complex will not have a high performance. Data-related approaches for preventing overfitting, *cross-validation* algorithm and *data augmentation*, are introduced in Chapters 3.3.3 and 3.3.4. Chapter 3.3.5 presents two model-related ways to reduce overfitting, *regularization* and *dropout*.

Besides overfitting, also *underfitting* may occur when training machine learning models. Underfitting means that the model is unable to reach a relatively low error [25], i.e. minimizing the cost function is unsuccessful. This occurs in, for example, situations, where a linear model is attempted to train with non-linear data or the other way around. However, because this Thesis concentrates on deep CNN models, which overfit more often than underfit, we will not go deeper into underfitting.

3.3.3 Cross-validation

Cross-validation algorithm is based on the idea of performing training and validation several times with different training and validation data. For each time, the training dataset is split randomly into subsets. One set is used for validating the model and all the other parts for training it. For *k-fold cross-validation*, this is performed by dividing training dataset into *k* equal-sized subsets. One set is used for validation and others for training. This process is repeated *k* times and the final model is a combination of the results. Figure 21 shows the idea of k-fold cross-validation.

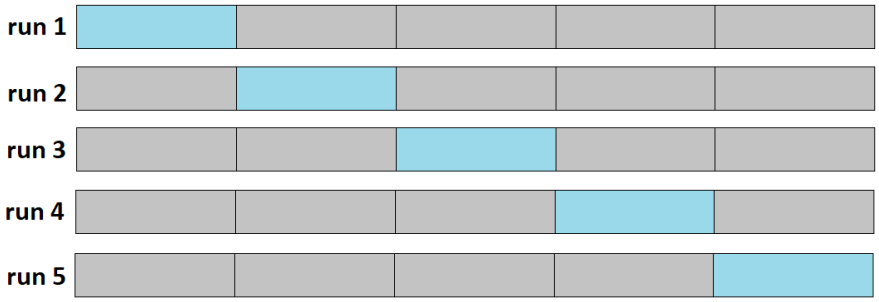

Figure 21: *K-fold cross-validation*

In Figure 21 the training dataset is divided into five subsets, so $k = 5$. This means that the training is performed five times. Light blue block means a subset that is used for validation in that run and grey blocks are the subsets used for training. [18]

If the size of the neural network model and the training dataset allow, we may mark that $k = N$, where N is the total number of the training samples. In this case, the algorithm is called *leave-one-out* cross-validation. The training is performed as many times as there are samples and each sample is used as validation data on its turn. Usually this is not required or even recommended, because the number of samples and the size of the network are so large that it would not be reasonable. [18]

3.3.4 Data augmentation

Data augmentation means increasing the number of the datapoints in the training data. In image classification, data augmentation means increasing the number of images. This can be performed by rotating, mirroring and translating the images. For example, rotating and translating an image of a tennis ball, which is demonstrated in Figure 22, may seem to human eye like a minor change, but a neural network won't know anything about tennis

balls before it sees the training data and therefore it is better to show as many different positions of the objects to be recognized as possible [33].

Figure 22: *Rotating and moving an image of a tennis ball*

There is a library [34] for programming language python called *imgaug* that provides several data augmentation techniques. With imgaug, it is possible to perform all the above-mentioned image transformations, but also other image processing like filtering. Different filters include adding noise to images like salt, pepper or salt and pepper noise. Blurring the images can be done with i.e. Gaussian and median filters. Adding noisy and blurry images to the training dataset improves the model's ability to separate noise and signals from each other [29]. Imgaug also provides tools for modifying the colors and the contrast of the images.

3.3.5 Regularization

In many approaches, *regularization* is performed by adding a penalty (a regularization term) to the cost function (introduced in Chapter 3.2) as $J(w) = J(w) + r$, where *r* describes the regularization term. The aim of the penalty term is to limit the capacity of the CNN model and therefore to prevent overfitting, because the simpler the model, the less it will be prone to overfitting. Commonly used regularization methods are L_1 and L_2 (also known as *weight decay*) regularizations. [25] *Dropout* is also one regularization method, proposed recently [19].

L_2 adds a quadratic penalty to the cost function *J(w)* as following

$$J(w)_{regularized} = J(w) + \frac{\lambda}{2}\boldsymbol{w}^T\boldsymbol{w}, \tag{3.22}$$

where λ is the chosen regularization coefficient and w describes the weights of the model. [18]

L_1 adds the absolute value of the weights *w* of the model to the cost function as following:

$$J(w)_{regularized} = J(w) + \frac{\lambda}{2}|\boldsymbol{w}|. \text{ [18]} \tag{3.23}$$

Dropout means dropping out units from the deep CNN model during training. The dropped units may be hidden or visible, but they are temporarily deleted. In the simplest case, the dropout is a probability between [0,1] referring to the proportion of dropped units out of all units. [35]

4. CAMERA TESTS

This Chapter introduces the camera tests. First, the requirements for cameras in the perspective of machine learning in access control system are discussed, including a glance into a basic Visy access gate. Secondly, the chosen camera models are introduced, and the choices justified and discussed. Also, the most interesting properties of the cameras from Visy Oy's point of view are presented. Then the implementation and environment of the tests are introduced, and finally, the use cases for a high image quality camera in Visy projects are considered.

4.1 Requirements for cameras

The main idea for these camera tests is to find out whether the cameras with a sufficient image quality to recognize small details from large images are suitable for Visy projects. Visy projects include machine learning in traffic control systems. The basic gate system is like in Figure 23. A truck drives to a gate that consists of a camera and a barrier. Few meters before the gate, there is an inductive loop on the ground that notices the truck driving and triggers the camera to take images. The purpose is that an image is taken and sent to a computer, license plate is recognized from the image and the permit decision made (access OK or denied) and the barrier starts opening, if the permit is fine, before the truck reaches the barrier.

Figure 23: *An example of access gate made by Visy Oy*

In the perspective of machine learning, the most important property of the camera is the image quality. The image quality is important, because the aim is to recognize details from image, and this is not possible if the images are blurry. When high image quality is desired, the focus needs to be on the camera's image sensor and pixel sizes. Image sensor was introduced in Chapter 2.4. To obtain high image quality, the sensor needs to be big and include many big pixels. In this case we search for cameras with a sensor with at least 10 megapixels, because it would guarantee a significantly higher image quality than currently in our projects, and more detailed images. Also, the aperture and the shutter, which form the exposure time, affect the image quality of a camera. These were introduced in Chapter 2.2 and 2.3.

When the machine learning is combined with an access control system, where the cameras need to be outdoors all the time every day, also speed, size, durability and usability of a camera become points of interest. The speed is important because the vehicles approaching the gates are not supposed to wait for long times before entering, and usually they are moving when the images are taken. Therefore, the camera needs to be fast enough to capture a sharp image while the truck is moving and transferring images from the camera to a computer for performing recognition needs to be fast also. Size, durability and usability are all related to each other, because, for example, durability can be improved with a camera outdoor housing, but a housing increases the camera size and possibly makes the usability more difficult.

The durability of the camera consists of the operating temperature and the mechanical durability. The operating temperature is important, because the camera needs to work in all possible weather conditions. However, as discussed, an outdoor housing can be added to increase the operating temperature. In the housing, it is possible to include heating and cooling and increase the mechanical durability because if a vehicle hits only the housing, the camera does not break. The outdoor housing needs to include the power and ethernet cabling, but it also needs to be small enough, because of the usability of the camera.

In traffic control systems, the size of a camera is important, because cameras need to be easy to attach to the gates and portals, and small enough not to disturb the vehicles driving by. So, when the outdoor housing is added, this needs to be kept in mind. The outdoor housing, the functions and user interfaces belong to the usability of the camera. If the camera breaks, it needs to be easy to switch. The outdoor housing cannot prevent electricians from checking and measuring electrical properties of the camera, the couplers or the cables. It would also be good to have an easy access to the buttons of the camera to zoom and focus after installing the camera, when needed. The UIs include,

besides cabling, software and data transferring protocols etc. We need to be able to transfer images from a camera to a computer, where the recognition is performed, fast, so vehicles do not have to wait a long time at the gate.

4.2 Camera models

When starting this project, a DSLR camera was considered as the only choice for improving the image quality, because of the image sensor and pixel sizes were thought to be large enough only in those. The negative side in DSLR cameras is that they are usually designed for consumers and not for business, and therefore the needs of business and 24/7 outdoor use are not necessarily considered. However, when the project proceeded, it turned out that there are also few video surveillance camera manufacturers, which offer cameras with sufficiently big image sensors and enough pixels. In this project, three different cameras are supposed to be tested. These include Basler BIP2-1920-30c (video surveillance camera), Sony SNC-VB770 (a video surveillance camera) and Canon EOS 6D Mark II (a DSLR).

The camera needs to fulfil all or almost all the requirements that Visy Oy has for the camera, but camera manufacturers don't have all the details that are needed to be studied on their websites, so comparing two different cameras on paper is not that straightforward. When considering DSLR or video surveillance cameras, also the photographic objective lens is an important part of the image quality. The availability of the cameras affected the choices also.

Canon was chosen for the manufacturer of the DSLR camera because it turned out that Visy Oy already had implemented some code for Canon cameras for other kind of purposes, and therefore the aim was to utilize that code when implementing the remote control of the camera (sending the message to shoot an image and transferring the image to a computer). Sony and Nikon were also considered as the manufacturer, mainly because of their reputation. No other major differences than the existing code between Canon, Sony and Nikon DSLR cameras were found. The model of Canon DSLR camera, EOS 6D Mark II, was chosen because it has a sufficient image sensor and pixel size, which means a high image quality, it is relatively new model, so it is expected that Canon is not going to finish manufacturing them in the near future, which would require new tests for a new camera model, and according to the documentation of Canon, this model has support for Canon EOS Digital SDK [36], which is needed for the Visy's code to work.

Sony's video surveillance camera was chosen because of the expected high image quality resulting from the image sensor and pixel sizes. A video surveillance camera was

chosen because of their reliability. They are usually designed for outdoor use all the time every day and for business, and therefore it is justified to expect that their durability, usability and speed fulfil the requirements that we have for cameras. Sony was also very cooperative, and therefore their cameras were easily available. Currently used Basler's video surveillance camera was included in tests to be able to compare the results and decide, if the goals are achieved with the other two cameras (Canon and Sony). Figure 24 shows all the three models included in tests. The sizes in Figure 24 do not correspond to the actual sizes and while Basler and Canon are shown without an objective lens, Sony is shown with one.

Figure 24: *Canon EOS 6D Mark II DSLR camera (on the left)* ***[17]****, Basler's BIP2-1920-30c camera (middle)* ***[16]*** *and Sony SNC-VB770 full frame video surveillance camera (right)* ***[40]***

Because of the requirements introduced in Chapter 4.1, we are interested in the physical size and weight of the camera, the size of the image sensor and the amount of the pixels, the maximum aperture size, the user interfaces and the power cabling of the camera. The properties of interest of all the three cameras chosen for tests are collected to Table 2. Visy Oy uses HTTP grabbing to get images from cameras, and therefore if HTTP grabbing is an option in a camera, it is considered as an advantage. From image compression formats, we are interested in **JPEG** (joint photographic experts group) to save network bandwidth in comparison to RAW in multiple camera configurations.

The cameras that Visy Oy uses now are Basler BIP2-1920-30-c cameras with an Azure 12-36 mm objective lens [16] [37]. The aperture is manually adjusted and used fully open in Visy Oy projects. Also zoom and focusing are done manually from the objective lens. These cameras are network cameras and their IPv4 addresses are obtained with Basler's BIP Finder software. Then the web UI can be accessed with the IP address and the camera can be configured.

The objective lens chosen for Canon EOS 6D Mark II camera is Canon EF 24 - 105 mm f/4L IS II USM lens [38] . The viewfinder is based on pentaprism and the camera has also an LCD screen. This camera has a live view-mode where an electronic projection

of the image is shown on the LCD screen for viewing it before taking the image, if one does not want to use the pentaprism viewfinder. [17] The aim with this camera is to stream the viewfinder image in live-view mode and capture images from that. One note for this camera model is that the maximum length of a video is 29 minutes and 59 seconds [17], so using the video property in the future is not possible.

Canon offers two different ways to control DSLR cameras remotely. EOS Utility software allows to control the camera remotely, for example, shoot images. The software user interface consists of different buttons, and therefore EOS Utility needs a person to use it. EOS Utility is available for free download from Canon website. Canon has also made an SDK called Canon EOS Digital SDK (**EDSDK**). The access to SDK can be obtained by applying for it on the internet. [17] EOS Utility software does not use EDSDK [39]. In this project, EDSDK was utilized because the goal is to have a camera, that is functioning automatically. Therefore, EOS Utility was not considered as an option.

Table 2: *Summary of the properties of the cameras and their objective lenses*

	Basler	Canon	Sony
Physical camera size (L/D x W x H, mm)	109.7 x 44 x 29 [16]	74.8 x 144 x 110.5 [17]	118 x 104 x 84.6 [40]
Camera weight (g)	210 [16]	765 [17]	720 [40]
Sensor type	CMOS [16]	CMOS [17]	Exmor CMOS [40]
Sensor size (mm x mm)	4.8 x 3.6 [16]	35.9 x 24.0 [17]	36 x 24 [40]
Maximum image resolution (px x px)	1920 x 1080 [16]	6240 x 4160 [17]	4240 x 2832 [40]
Number of pixels (Mpx)	2.1 [16]	26.2 (effective) 27.1 (total) [17]	12 [40]
Pixel size (µm x µm)	5.86 x 5.86 [16]	-	-
Objective lens size (D x L, mm)	38.5 x 53.97 [37]	83.5 x 118 [38]	64.4 x 70.5 [41]

Objective lens weight (kg)	0.150 [37]	0.795 [38]	0.281 [41]
Objective lens aperture size	f/2.8 [37]	f/22-f/4 [38]	f/22-f/1.8 [41]
Objective lens focal length (mm)	12-36 [37]	24-105 [38]	55 [41]
Operating temperature (°C)	-10 – +50 [16]	0 – +40 [17]	-5 – +50 [40]
User interfaces	BIP Finder + Web UI	EOS Utility, Canon EOS SDK	Web UI
Shutter	Global shutter [16]	Focal plane shutter (electronic control) [17]	Electronic [40]
HTTP grabbing	Yes [16]	No [17]	Yes [40]
Video compression formats	MJPEG, MPEG-4, H.264 [16]	MPEG-4, H.264 [17]	JPEG, H264 [40]
Max. video rate (fps)	30 [16]	-	2.5, 30 [40]
JPEG format	Yes [16]	Yes [17]	Yes [40]
Power cabling	PoE or 12 to 24 VDC [16]	Battery or an AC Adapter E6(N) with a DC coupler: DR-E6 [17]	12 VDC or 24 VAC [40]

The objective lens for Sony's full frame video surveillance camera SNC-VB770 is Carl Zeiss Sonnar FE 1.8/55 ZA. The image sensor of this camera is a 35 mm full frame exmor CMOS sensor [40]. Exmor CMOS image sensor technology is developed by Sony. [42] Sony's special exmor sensor technology does not have a conventional fundamental pixel structure, which is called *front-illumination*. Sony uses a *back-illumination* pixel structure. Back-illumination provides improved sensitivity and noise reduction, but causes dark current, defective pixels and colour mixture that lead to decrease in SNR. To solve these problems, Sony has a photo diode structure and on-chip lens, which both are specifically developed for back-illumination pixel structure. [43] This camera is a net-

work camera, which is considered as an advantage because we are currently using network cameras and therefore it would be really easy to integrate this camera to Visy's projects. Sony offers a web UI for configurating the camera, watching the live image from it and taking evidence shots. The IP address for accessing the web UI can be obtained, for example, with SNC Toolbox provided by Sony. [40]

From the perspective of durability and usability, all the three cameras have almost the same operating temperature as shown in Table 2. This means that they all need an outdoor camera housing to be able to work in less than -5 °C temperatures. Basler camera is significantly smaller and lighter than Sony and Canon, which are about the same size with each other. However, Sony and Canon are still small enough for Visy's purposes. The objective lens used in Canon's case is bigger than in Sony's but does not offer as wide aperture size range. Objective lens can be, however, chosen from many different options. One just needs to make sure that it is suitable for the camera. With Sony's camera we in Visy would not need to change almost anything in our code, because the user interface, accessing the camera configuration, telling the camera to shoot and transferring the images to a computer are implemented in the same way as in our projects currently. But, the software for Canon is also implemented, it should just be modified for this particular use. Basler and Sony are both powered with a power cable. For Canon to work all the time every day, we would need a separate AC adapter and DC coupler, which were difficult to find and buy, as it turned out during this project.

Considering the image quality, Basler has significantly smaller image sensor and it produces significantly smaller image resolution as can be seen in Table 2. When considering object detection cases, which are introduced in Chapter 4.4, Basler's IP camera is not enough because its image sensor is too small and therefore the image quality is not high enough. Sony has only half of the image sensor size of the Canon's camera, but it still exceeds the aim, which is over 10 megapixels. As shown in the Table 2, the image sensor in Sony has 12 megapixels. Therefore, both Sony and Canon have image sensor big enough to produce a sufficient image quality.

4.3 Test implementation and environment

The camera tests were performed in the office of Visy Oy. The cameras were placed on a table in front of a window next to each other. Figure 25 shows approximately how the cameras were located on the table, but not the final test placements. The cameras were not moved or touched during the tests. The idea was to get images from roughly the same place with the same view at the same time to be able to compare the test results. In the tests, the cameras were shooting images every 15 minutes to obtain images at

night also, because the importance of the exposure time and image quality increases at night.

Figure 25: *The placement of the cameras*

For shooting the images with a specific time interval, a software made by Visy Oy was used. The software tells the camera to shoot an image every 15 minutes (or as often as it is configured) and after each shooting, it transfers the image from the camera to a computer and saves it into a folder. The image transferring is designed in these tests as if it would be performed in real life. In real life the camera would shoot only when it is triggered with a message from Visy software, but in this case, images are only needed to compare the image quality. The image transferring was performed with HTTP grabbing, because only Sony's and Basler's cameras were included in the tests and both offer this option. Sony and Basler were taking photos during the following nights: 6.3-7.3.2018, 7.3.-8.3.2018 and 8.3.-9.3.2018.

The promised support in Canon camera for EDSDK turned out not to be working. This was discussed with Canon contacts and they promised to study the problem [39], but no solution was found in a reasonable time, so Canon's camera was left out of the tests. This also showed that maybe DSLR cameras are not reliable enough for our projects. We have to sometimes switch to a newer model in cameras, because, for example, the manufacturer does not produce the model used by us anymore. In these situations, we have to be able to be sure that the support for an SDK that is promised, actually works.

The speed is only shortly tested by holding a hand in front of the camera and pressing the shutter button at the same time when taking the hand away and checking, whether the hand is still in the image or not. We made a decision not to follow official standards when comparing the image quality, but to test the cameras specifically for Visy project

needs. Many of the standardized tests are designed for cameras that are used in photo shooting, professional or individual use and mainly in indoors. In Visy projects, the cameras are outdoors all the time every day, so the properties of the cameras were studied in that point of view. The focus in Visy projects is on machine learning, and more specifically on image or object recognition and not so much on the visual result of the image.

4.4 Use cases

As mentioned, a better image quality is obtained with a sensor with more pixels than in the current cameras. Then, the sensor also needs to be bigger to keep the DR value high despite the bigger number of pixels. There are two main use cases, where the cameras with better image quality could be used. Both cases include recognizing small details of containers. An example image of a container is in Figure 26. From images of whole containers, we want to recognize, in addition to container numbers, some smaller details too, because it is a request of the customers.

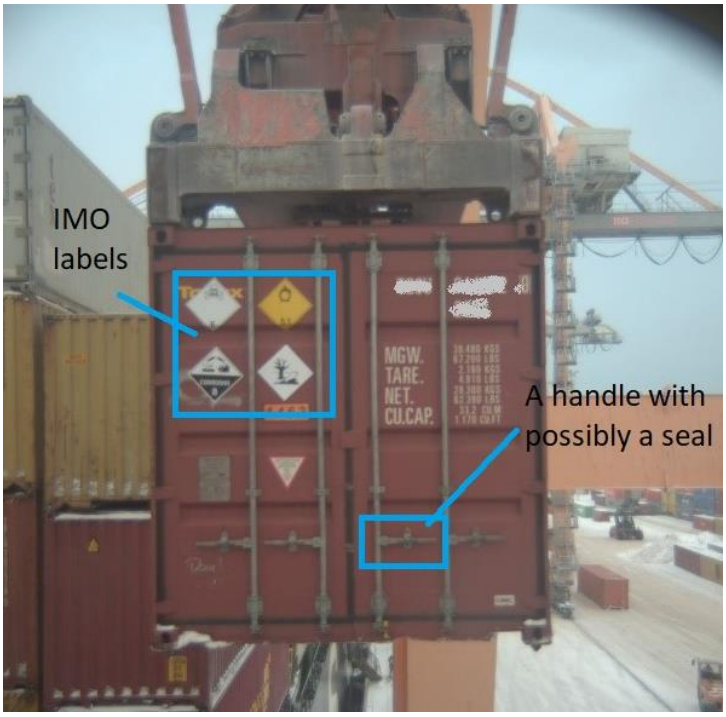

Figure 26: *An example of a container image*

One of the use cases for the high-quality camera is recognizing IMO labels, which mean the labels for dangerous goods that are mandatory for trucks and train wagons transporting them. In Figure 26 can be seen examples of those too. For machine learning purposes, the image is too blurry to detect and classify the IMO labels. Figure 27 shows an example of the image quality in practise, taken with a Basler camera, zooming to the

labels. One of the reasons for this quality is that the camera needs to take an image of the whole marine container, as shown in Figure 26, but the IMO labels are small compared to the container with the image quality that Basler BIP2-1920-30c cameras offer. With a better image quality, the image taken from the whole container could look clear enough for detecting and recognizing these labels too.

Figure 27: *Example of the image quality in IMO recognition in practise*

Another case would be seal recognition. Marine container handles should usually be sealed with a seal to ensure that the goods they are transporting are still the same as when they collected them, and nothing has been removed or added. Seals are relatively small objects and with the present cameras, almost impossible to recognize even with a human eye. Figure 26 shows how small the handles are on the images when there is the whole container on the image. Figure 28 shows two example images of seals.

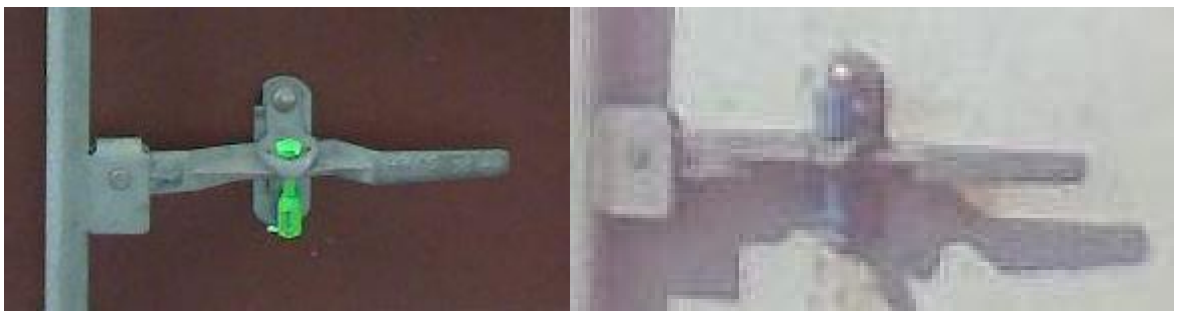

Figure 28: *Two examples of image quality in seal recognition in practise*

The image on the left in Figure 28 is not a zoomed in version of an image with a whole container. This image is taken with a camera, which is dedicated only to take images from seals. The purpose is to have photos of as high image quality as this but taking the whole container in the image and not just handles. Most of the images containing seals look like the photo on the right in Figure 28, so not clear enough. In addition to recognizing if there is a seal, we should be able to recognize, if the seal is broken or not. If it is very difficult even to recognize if there is a seal, it seems impossible to tell, if it is broken. With the image quality of Basler BIP2-1920-30c cameras, we have not been able to implement this, and the aim is to choose a high-quality camera, which is suitable for Visy projects, to be able to implement seal and IMO label recognition in the future.

5. USE CASE: A VIRTUAL TRIGGER IMPLEMENTATION

In this work, a use case of machine learning, and more precisely, of deep convolutional neural networks introduced in Chapter 3, is implemented. It is called a virtual trigger. The idea is that the convolutional neural network takes an image as an input and classifies it into one of two classes: car or no car. “Car” in this case means any vehicle trying to enter a Visy access gate like one in Figure 23. Currently, laser scanners and inductive loops are used for triggering the cameras to take images in Visy projects, when a vehicle arrives at the gate. The aim of this DNN implementation is to replace scanners and loops with virtual triggering, so only the cameras would be needed in the projects in the future. The idea of how virtual trigger would operate can be seen in Figure 29. The output means the class probability: when the output is close to zero, it will be classified as “No car” and when a vehicle is present, the output will be close to one.

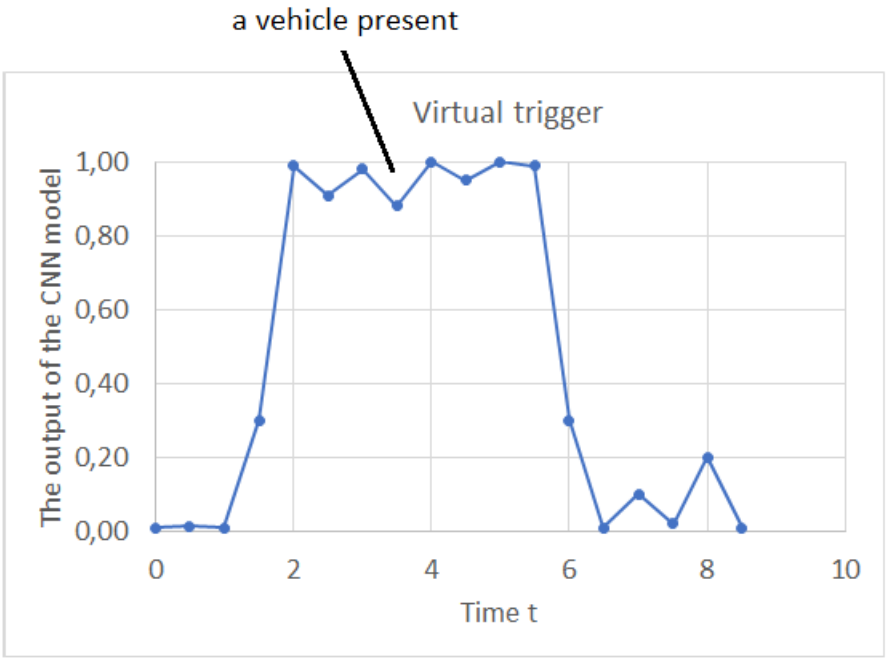

Figure 29: *The idea of how virtual trigger would operate*

First, this Chapter presents what kind of data was collected to train and test the network. Secondly, the used programming environments, platforms and languages are discussed and, finally, the implemented models and the ways of comparing them are introduced.

5.1 Data

The training and the testing data consist of images of vehicles, and the sets were collected from different places and different lanes around the world. Train and test sets are collected from different places and kept separate throughout the process to be able to use unseen data for final testing of the neural network. All the places, from where the images were collected, have Visy's traffic control system. The purpose was to include images from different times of the day and different times of the year to include as many different scenarios in training and testing as possible, because the classifier needs to work properly all the time in different conditions. Different scenarios include everything where the images may differ from each other, for example sunshine, cloudy weather, raining, snowing, day and night. Figure 30 shows examples of different lighting and weather conditions. Images in Figure 30 are taken from the actual training dataset.

The images in the datasets are taken with a Basler BIP2-1920-30c cameras with Azure optics, which are introduced in Chapter 4.2. The settings, zooming and focusing of the cameras differ slightly from each other in different access control systems because the lanes in real life can be very different from each other and, regardless of the lane, it is important to get a clear and sharp image for license plate recognition. Also, depending on the lane, the angle in which the vehicle drives to the gate may differ. The highest image resolution that can be obtained from Basler cameras is 1920 x 1080 pixels, and therefore all the images in the training set are of this resolution or lower. All the neural networks in this project resize the images for speeding up the training. Few image sizes were tested in training and according to these test results, the image sizes resulting in the highest classifying accuracies were chosen for further network training and testing.

The top image on the left in Figure 30 shows an example were the sun is shining and its rays are reflecting from the car and disturbing the visibility of the details. The top image on the right shows an example of the image quality at night, the visibility of the details is low again. The middle image on the left shows a situation when it is snowing, and the snow is disturbing and covering the details in the image. The bottom image on the right shows a situation where the environment looks different because of snow and disturbs the classifier because the classifier needs to learn the environment too to be able to separate it from a vehicle. The middle image on the right shows a cloudy environment, where, however, the details should be sharp because there are no shades or extra objects (like snowflakes) disturbing the visibility. The bottom image on the left shows a situation where there is a vehicle in the image, but it is not at the correct spot for license plate recognition, and therefore it should be classified as a 'No_car'.

Figure 30: *Examples of different weather and lighting conditions*

When comparing the images including a vehicle in Figure 30, it is visible that the vehicles come in an angle to the camera, from left or from right. This also complicates the CNN model training, because the model needs to learn both angles, with a vehicle and without.

The training set was modified during the training of the CNNs, mainly by adding more images from new places for obtaining higher classifying accuracies and avoiding overfitting, which was introduced in Chapter 3.3.2. The final training set consists of 1145 images belonging to class ‘Car’ and 1213 images belonging to class ‘No_Car’. So, in total 2358 images were used for training the network.

The test set was collected similarly to training set: as many different weather conditions as possible and vehicles of different colours and angles etc. The final test set consists of 62 images belonging to class ‘Car’ and 63 images belonging to class ‘No_Car’. So, 125 images in total belong to the test dataset.

5.2 Programming tools

The code is implemented with an application programming interface (**API**) called Keras, which is developed for neural networks and written with a programming language called Python. It is also usable with Python versions 2.7-3.6, and therefore all the code is implemented in Python. It can run on different platforms. In this work, a Tensorflow backend is used with cuDNN dependency. Keras offers tools for neural network development and those are utilized in this project. [44]

CuDNN is a library accelerated with a *graphics processing unit* (**GPU**), developed for running codes on a GPU [45] instead of a *central processing unit* (**CPU**). Teaching deep neural networks may be very slow depending on the size of the network and the amount of training data. To obtain high recognition or classification accuracies, a large amount of training data is usually desired. Running code with GPU-acceleration makes teaching deep convolutional neural networks faster. CuDNN needs to be installed with CUDA Toolkit, which is a programming model and computing platform [46]. In this work, the virtual environment running the code on GPU was used for training the models, because it is faster and the environment running the code on CPU was used for testing the models, because in real-life situation there are rarely GPUs available and the aim was to get as reliable and realistic test results as possible.

In this work, the code is implemented using Anaconda Distribution, which is meant for performing code in R or Python programming language. In this work, we are interested in Python, so Anaconda works as a Python package manager. Anaconda offers a possibility to create different virtual environments and install different Python packages to each of them. This enables, for examples, creating two virtual environments, which are copies of each other, but one runs the code on CPU and the other one on GPU. Anaconda is also compatible with Tensorflow, which is necessary to be able to use it since we want to use Keras with Tensorflow backend, because it is the standard way in Visy Oy. Using Keras with Tensorflow backend enables other workers in Visy to run the codes also on their computers. [47]

5.3 Neural network models

Three different deep CNN models were trained. Two of them use a pre-trained Keras network model as the basis for the neural network, and only the last layer is added or modified. One neural network model was implemented from scratch. The purpose of testing many different networks was to find one with a high classification accuracy and speed. Because the aim is to replace inductive loops and laser scanners, the virtual trigger should be almost as accurate, which means roughly over 99 % recognition accuracy. The loops and the scanners trigger an image when the vehicle is about seven meters from the camera and the aim is that the virtual trigger would work likewise. The speed is related to the real-life use of this application. The vehicles are supposed to pass the gates fast, assuming they have a permission to access, and therefore the classification should happen practically in no time. Something around 200 to 300 milliseconds is a maximum for classifying one frame, and therefore it would be possible to capture and classify 3 to 5 frames per second. After the classification the access control system still needs to capture the permit image and do the license plate recognition, and because of this, the virtual trigger cannot consume a lot of time.

One of the pre-trained models is called VGG16 [48]. It is a deep neural network proposed by Simonyan and Zisserman. It achieved 92.7 % accuracy on ImageNet database. ImageNet database includes over 15 million images, which are labeled. [49] The default input size for VGG16 is 224 x 224, but this can be modified, and, in this work, it is decreased, because the smaller the input image size the faster the training and the classification, and speed is one criterion for the CNN in this Thesis. The input images need to be in RGB color model for VGG16. However, VGG16 has shown higher results comparing to previous pre-trained models [49] and therefore it was chosen to be one of the models to be tested in this work also. The last layer of this structure is removed and replaced with one corresponding to the virtual trigger, which has only two classes: vehicle or no vehicle.

The second pre-trained neural network is called MobileNet [48]. MobileNet was proposed by a group of eight people working for Google. Their motivation was to implement deep convolutional neural networks small enough to be used on, for example, mobile phone applications, but still efficient. Since we are interested in fast and efficient classification, this pre-trained model was chosen to be tested for virtual trigger. MobileNet has 28 layers. Like with VGG16, the default input size for MobileNet is 224 x 224, and input images need to be in RGB. MobileNet is also pre-trained with ImageNet weights and the last layer needs to be replaced with one corresponding to vehicle/no vehicle model. In a classification task, which was performed by the Google group that invented MobileNet,

MobileNet obtained slightly lower classification accuracy compared to VGG16, but the difference is relatively small (70.6 % vs. 71.5 %) and the speed of the classification in this project is very important, and therefore MobileNet is chosen to be tested. [50]

The structure of the model implemented from scratch is shown in Figure 31. The motivation to implement a deep neural network model of my own is to obtain even faster classification than with MobileNet, which was introduced above. This model created from scratch was chosen to have only two convolutional layers because that was the number of the convolutional layers used in a project [51] which is slightly similar to this task. In that project, the aim was to classify the vehicle on an image to one of four classes: truck, van, car and bus [51].

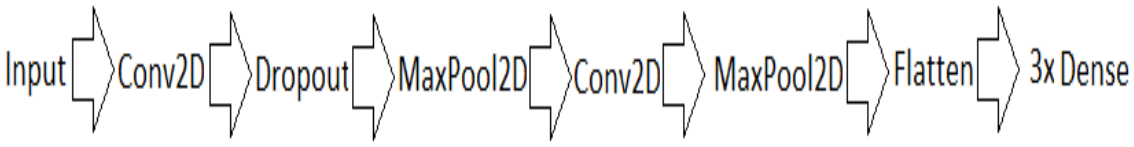

Figure 31: *The structure of the CNN implemented from scratch*

Figure 31 shows that the model created from scratch has an input layer, the first convolutional layer followed by dropout and max pooling and the second convolutional layer followed by max pooling. Then it has a layer called *flatten*, which is used for flattening the input [52], which means transforming it into a 1D vector, and finally, three fully-connected regular layers called *dense* layers in Keras [52].

Table 3 presents different parameters chosen for the models. Input image size is chosen to be the same for each model to be able to compare the classification speed. Loss, optimizer and error metrics are also the same. Batch size is chosen to be 32 for MobileNet and the model from scratch. This batch size was noticed to be too large for the VGG16 model, because it led to an *exhausting error*, which means that the GPU used was not efficient enough to perform the training. Therefore, the batch size is chosen to be 16 for the VGG16 model. Learning rate is different for each of the models. Different learning rates were tested and the one resulting in best validating accuracies is chosen. Epochs for MobileNet and the model from scratch are 60, but 50 for VGG16. It was noticed that training and validation accuracy did not increase during the last 10 epochs for VGG16 and it was noticed to be slower to train than the other two models, so therefore the epochs for it were decreased by 10.

Table 3: *Parameters chosen for the deep CNN models*

	Input image shape (pixels, pixels, color channels)	Loss	Batch size	Learning rate	Optimizer	Error metrics	Epochs
VGG16	(128,128,3)	categorical cross entropy	16	1e-4	SGD	Accuracy	50
MobileNet	(128,128,3)	categorical cross entropy	32	3e-4	SGD	Accuracy	60
From scratch	(128,128,3)	categorical cross entropy	32	1e-2	SGD	Accuracy	60

All the three models are trained with the same training data introduced in Chapter 5.1 and the training and validating results are plotted using the same code to be able to compare them. The models are also tested with the same test dataset, which was also introduced in Chapter 5.1. The speed is tested by classifying the test set, and measuring, how long time it takes from the model to perform classification. From this it would be possible to calculate, for example, the average time for classifying one image and the frames per second (**FPS**) speed. The classification test is performed in the virtual environment, which runs the code on CPU, because in real-life projects the classification will be run on CPU, at least for now.

For each model, we plot an accuracy and loss plot showing the training and the validation accuracies. Accuracy was introduced in Chapter 3.3.1. The training data is split to train and test parts. Also, a confusion matrix (introduced in Chapter 3.3.1 and shown in Table 1) and a ROC curve (introduced in Chapter 3.3.1 and shown in Figure 20) are plotted for all the models and few measures computed from confusion matrix. Then, the results are compared and discussed. The training, validating and test results are collected and discussed in Chapter 6.2.

6. RESULTS

The purpose of this Chapter is to introduce the results of the camera tests and the virtual trigger tests separately. The camera tests include few examples of the images taken during the tests according to Visy's own purposes as introduced in Chapter 4. The images are followed by some discussion about the image quality and the properties of the cameras. The aim is to decide, whether the tested cameras have a significantly better image quality than the current cameras and if they can be used in Visy's projects due to their properties or not. Secondly, the results for the three different virtual trigger deep CNN models are introduced and discussed and one of them is chosen for real-life tests that are not a part of this Thesis.

6.1 Camera tests results and discussion

The camera tests implementation, motivation and the camera models chosen for tests were introduced in Chapter 4. The purpose was originally to include all the three cameras introduced in Chapter 4.2 to the tests. But because of problems, also discussed in Chapter 4, that occurred in implementing the test code for Canon EOS 6D Mark II, only Basler's BIP2-1920-30c and Sony's SNC-VB770 video surveillance cameras were finally included in the tests.

Figure 32: *Basler test image on 9th of March 2018 at 1:45AM*

Figure 32 shows a test image taken with a Basler IP camera (the present model used in Visy projects). This image is taken at night, because the importance of image quality increases when it's dark. From Figure 32 it can be seen that the image is quite blurry, and no small details can be detected. Figure 33 shows image taken with Sony's camera approximately at the same time as the Basler's image above. When comparing Figure 32 to Figure 33, it can be evaluated with human eye that the image quality is significantly higher in Figure 33. So, Sony's camera offers a better image quality than Basler's.

Figure 33: *Sony test image on 9th of March 2018 at 1:45AM*

The test results show clearly that 12 megapixels is enough to increase the image quality from 2 megapixels. Because the Canon's DSLR camera has over 20 megapixels, it is likely that also the image quality of it would be enough, but the reliability, like the support for SDK, is not enough, and therefore it is better to use video surveillance cameras. Important properties of Sony's camera were similar to the Basler's camera. For example, the messages can be sent to the camera as to the present Basler's camera. Images can be transferred via HTTP grabbing, which is the case with the Basler camera also. Therefore, there is no need to change anything in Visy software, which is a significant advantage. Sony's camera is powered by a power cable which is also important, because in Visy projects, using a battery is not possible.

The speed was tested as described in Chapter 4.3, and no major differences between the cameras were noticed. The speed tests did not follow any standards and therefore, it is to question, if the speed of the cameras actually differ or not. However, in Visy projects the most important point concerning the speed is that the camera takes the images fast enough after it receives the message to do so, and both Basler and Sony fulfil this criterion.

However, Sony's camera would need some outdoor housing to work in less than -5°C, which needs a bit of consideration. But all the cameras that were studied for this Thesis have this disadvantage, and therefore it does not affect the choice. The most important property, image quality, seems to be significantly better in images taken with Sony than taken with Basler, and due to all these observations, Sony is chosen to be tested in future projects, where we need to recognize small details from images.

6.2 Virtual trigger results and discussion

Three different deep CNN models were implemented and tested. Two of them are pre-trained models. We start with the results from training and validation and continue to test results. The results are compared to each other, discussed, and the most suitable model for virtual trigger is chosen according to this. For each model, there is shown an accuracy plot, a confusion matrix and a ROC curve. These were introduced in Chapter 3.3. The parameters for each model were presented in Chapter 5.3.

Let's first compare training and validation accuracies and losses of the models. The accuracy and loss plots for all the three models are shown in Figure 34. The accuracy is presented as a probability between [0, 1]. It can be changed to percentage by multiplying with 100. The green curves on these figures are named as "test accuracy" and "test loss", but what they mean is the testing performed during the training, so they can also be called validation accuracy and validation loss. Training and validation accuracies for all the models seem to be smooth and steady, and the losses smoothly decreasing during the training. Except for model created from scratch, the loss seems to be slightly increasing towards to end of the training. Both training and validation accuracy seem to be quite high for all the models.

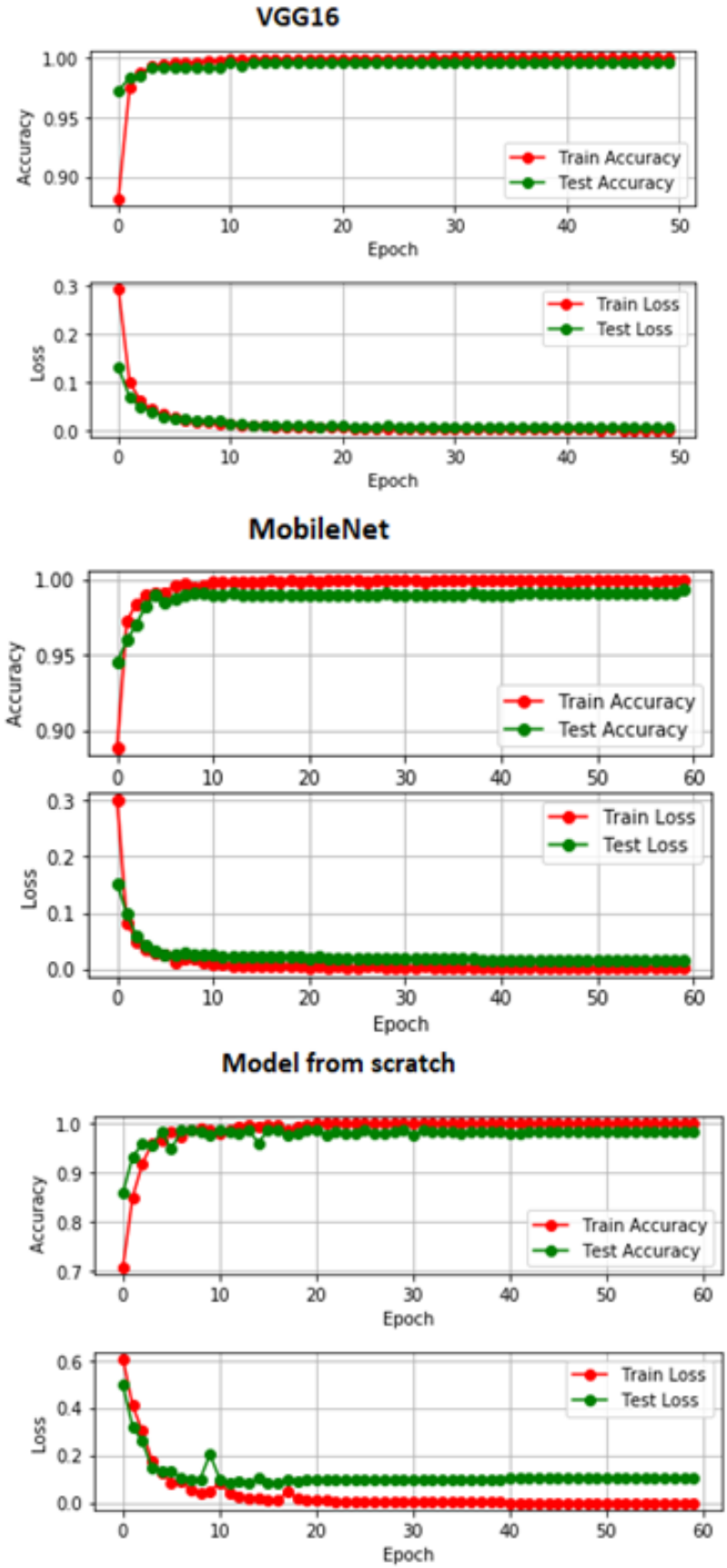

Figure 34: *Training and validation accuracies and losses plots for all the three models*

For VGG16 model, training and validation accuracies seem to be a little closer to each other according to the plots in Figure 34, which would indicate that this model has overfitted the least. This is, however, better studied with the test data. Also, the differences between the performance of the models is difficult to tell only by Figure 34, because we see the values only approximately and all the models seem to have quite similar performance according to these result plots. The actual values for validation accuracies are collected to Table 5, and these tell more accurately, which model obtained the best validation accuracy during training.

VGG16	Pred: Car	Pred: No car
True: Car	230	2
True: No car	0	240

Mobile Net	Pred: Car	Pred: No car
True: Car	231	2
True: No car	1	238

Scratch	Pred: Car	Pred: No car
True: Car	220	4
True: No car	4	244

Figure 35: *Confusion matrices for all three CNN models*

Figure 35 shows the confusion matrices for all the three models. “Pred” means the predicted class and “True” is the class to which the image actually belongs. From these matrices, we compute FPR, FNR and F1 score for each model. These were all introduced in Chapter 3.3.1. The results are collected to Table 4.

Table 4: *Results for CNN models*

	FPR	FNR	F1 score
VGG16	0	0.0086	0.9957
MobileNet	0.0004	0.0086	0.9935
Scratch	0.0161	0.0179	0.9821

The ROC curves for all the model is presented in Figure 36. The plots show also the AUC values for each model.

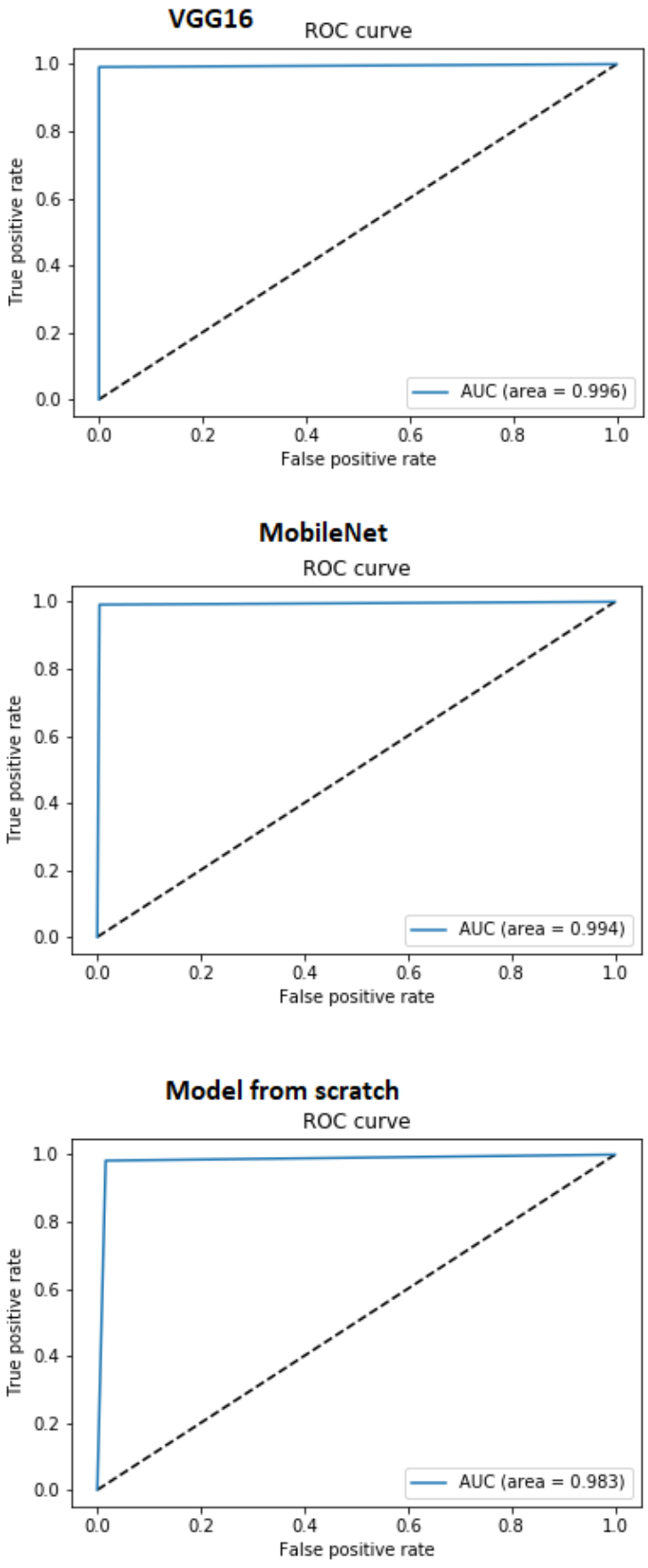

Figure 36: *ROC curves for all the CNN models*

The shape of the ROC curves for all the models seems to follow nicely the shape of the ideal ROC curve. The AUC value for VGG16 is 0.996, which is close to ideal value 1. For MobileNet, the AUC value is 0.994, which is also close to the ideal value. The AUC for the model created from scratch is 0.983, which is the lowest one of these three models, but still, in general, relatively close to the ideal value.

Training, validation and testing accuracies, and the classification time for the entire test dataset with 125 images are collected to Table 5. The tests were run more than once, and CPU time in milliseconds was measured to be in the time interval of 9 and 15 milliseconds for all three models. No major differences between the classification time was found, so it seems that the test dataset may be too small to test the accurate classification speed.

Table 5: *Test results for three deep CNN models*

	Training accuracy (%)	Validation accuracy (%)	Test Accuracy (%)	CPU time (milliseconds for 125 images)
Model 1: VGG16	100	99.58	92.8	13.99
Model 2: MobileNet	99.95	99.36	94.4	10.99
Model 3: From scratch	100	98.31	83.2	12.59

When comparing all these results introduced above, it seems that the model from scratch is slower than MobileNet, however, the speed tests may have not been reliable. Model from scratch also obtains a significantly lower classification accuracy on the test dataset than the pre-trained models, which would indicate that the model has been overfitting. The difference between VGG16 and MobileNet models is not that significant. MobileNet seems to be faster but obtains slightly lower validation accuracy. However, it obtains a higher test accuracy. Because of the knowledge that VGG16 is a larger and therefore slower network and that it obtains lower test accuracy than MobileNet, MobileNet model is chosen as the deep CNN model to be tested in a real-life case. The real-life case is

outside of the focus of this Thesis, because the aim of this project was to test different deep CNN models and choose the most suitable one.

The training dataset in this project was stable, and therefore the results may be optimistic compared to a real-life situation, where the angles of the vehicles differ from each other, the weather conditions differ, and the camera is zoomed and focused differently. However, improving the results in a real-life site can be performed by adding images from that site to the training set.

Virtual trigger is now operating in one real-life site. For now, it seems to be working with a slightly less accuracy than in these tests and it triggers the image too early, when the license plate is not yet on the correct spot. But updates are under implementation, and the tests in the real site show a positive future and possibilities for improvements.

7. CONCLUSIONS

Visy Oy is a company producing machine learning software to traffic control systems. Image quality plays a key role in machine learning projects where the purpose is to detect and classify small details from images. There were two main goals in this Thesis related to traffic control systems which include machine learning. One goal was to find and test high-quality cameras, study their properties and decide according to the results whether they can be used in Visy's projects or not. The second goal was to implement a deep convolutional neural network, called a virtual trigger, for classifying images into two classes: cars and no cars, where class 'Car' means any vehicle on a specific spot in the image. A camera meant for license plate recognition is zoomed and focused on this spot.

The motivation to test high-quality cameras was to be able to recognize small details, such as seals or dangerous goods labels, from an image that is taken of a whole container. Basler's BIP2-1920-30c cameras are currently used and suitable for license plate and container number recognition, but they do not offer an image quality high enough to recognize seals and dangerous goods labels. In the images taken with a Basler, seals and IMO labels are too blurry to be recognized correctly or even detected. To improve this, two cameras were chosen for tests, Canon EOS 6D Mark II, which is a DSLR camera and Sony's SNC-VB770 video surveillance camera. Also, the Basler's video surveillance camera was included in the tests for comparing the image quality. Finally, Canon's camera was left out, because of a problem with its support for Canon's remote camera control software called EDSDK.

The image quality tests were performed in Visy's office and they were planned in the perspective of Visy's needs and purposes. The image quality was estimated visually. We in Visy decided not to perform standardized tests, because standard tests were not found to measure our needs.

It took a long time to realize that the problems in developing remote control for Canon's camera were not in Visy's software but in the EDSDK support. When this was found out and discussed with Canon, it would have still been possible to ask for another camera from Canon to include a DSLR camera into tests in addition to the video surveillance cameras. Now the tests included only video surveillance cameras and therefore seem to be slightly one-sided. There is still the possibility to test some other DSLR model from Canon or from another manufacturer, if DSLR cameras are considered reliable enough

in the future. According to these tests, DSLR cameras are not yet reliable enough for Visy's projects.

After all, the goal was achieved. Sony's camera was found out to fulfil the requirements that Visy Oy has for cameras. It can be used similarly to the current Basler cameras in projects, so software changes are not required. It also fulfils the image quality requirement with its 12 megapixels image sensor. After the tests this camera model was proposed to a potential customer when making a project offer, but the decision has not been made when writing this Thesis. It is likely that we will use this camera model in some projects that require high image quality in the future.

The motivation for the virtual trigger part was to replace inductive loops and laser scanners that are currently used to trigger images at traffic control gates in Visy projects. This would save money and time, when there would not be the need to dig loops into the ground or configure laser scanners. The idea is to trigger permit images as they are triggered with loops and scanners. Whenever the output from the neural network is 'Car', an image will be taken.

Training and test datasets were collected separately for implementing deep convolutional neural networks. One virtual environment was created to train the neural networks on a GPU, and one was created to test them on a CPU, because in projects there usually is not a GPU available. Two of the deep CNNs are pre-trained network models, called MobileNet and VGG16 and one model is created from scratch.

MobileNet classifies fast enough and has over 99 % accuracy on validation data. The aim was to obtain over 99 % accuracy also on test data, but this was not achieved while making this project. VGG16 reached over 99 % validation accuracy and model from scratch reached over 98 % validation accuracy, but both obtained a lower test accuracy compared to MobileNet. We found out that there was not enough data for getting reliable speed measurement results, but the speed tests proved that all the three models classify fast enough on CPU. Because the speed test results are not reliably comparable to each other, the final model for real-life tests was chosen due to the documentation and research about MobileNet and VGG16. MobileNet was chosen, because it has a lighter and smaller structure than VGG16, which would indicate that MobileNet is a faster classifier, and it also obtained a higher classification accuracy on test data than VGG16.

At first, there were many problems with the virtual environments. Whenever one library is updated to a version that has some bugs, the whole virtual environment breaks down so there were a couple of occasions where the virtual environments needed to be created all over again. Also plotting the results had some problems. This was solved by using

and modifying some ready-made code. The number of images belonging to training dataset was increased when it seemed that accuracy will not be enough. Increasing the number of training images helped.

Virtual trigger is running at a customer site at the moment. The results are very promising but further improvements are needed to obtain the required 99 % in real installations. This is mainly because the lanes in the customer sites look so different to each other and the cameras are focused and zoomed differently. For example, the triggering happens too early in the current real-life test environment because the camera is not as zoomed in as in most of the training data cases. Some images were collected from this real test lane and the deep CNN model based on MobileNet has been trained again to obtain higher accuracies, but the network model has not yet been updated to the site. Therefore, when writing this Thesis it is still unknown how the improvements will affect.

All in all, almost all the goals were achieved and there seems to be future use for both Sony's camera and virtual trigger.

REFERENCES

[1] H. Maître, From Photon to Pixel: The Digital Camera Handbook, John Wiley & Sons, Incorporated , 2015, 403 p., ISBN: 9781119238645

[2] The Camera, Life library of photography,Time-Life Books, Time Inc., London, 1974, 236 p.

[3] Buckely, C., Aperture Diagram, Image, License: CC BY-SA 3.0, 2006. Updated by Glrx, 5 September 2018.

[4] H. Lehto, R. Havukainen, J. Maalampi and J. Leskinen, FYSIIKKA 3 Aallot,Tammi, Helsinki, 2009, 159 p., ISBN: 9789513147921

[5] J. Law and R. Rennie, A Dictionary of Physics (7 ed.): Aperture, Oxford University Press, 2015, Available (accessed 15 June 2018): https://www-oxfordreference-com.libproxy.tuni.fi/view/10.1093/acref/9780198714743.001.0001/acref-9780198714743-e-133#

[6] E. Allen and S. Triantaphillidou, The Manual of Photography and Digital Imaging, 10th edition, Elsevier Ltd., 2011, 585 p., EBOOK ISBN: 9780080926803, Available (accessed 13 May 2019): https://ebookcentral.proquest.com/lib/tampere/reader.action?docID=647538

[7] M. Brown, Mechanical vs electronic shutters, Photo Review Magazine, Excerpt from Iss 75, Available (accessed 17 March 2019): https://www.photoreview.com.au/tips/shooting/mechanical-vs-electronic-shutters/

[8] N. Dilmen, Windflowers, Image, License: CC BY-SA 3.0, 2004.

[9] A. El Camal and H. Eltoukhy, CMOS Image Sensors, IEEE Circuits and Devices Magazine, Vol. 21, Iss. 3, 2005, pp. 6-20.

[10] F. Tavernier and M. Steyaert, High-Speed Optical Receivers with Integrated Photodiode in Nanoscale CMOS, Springer Verlag, New York, 2011, 230 p., EBOOK ISBN: 9781441999252

[11] Moxfyre, Sensor sizes overlaid inside, Image, License: CC BY-SA 3.0, 2009. Updated by Trammell Hudson, 29 May 2009.

[12] S. Crisp, Camera sensor size: Why does it matter and exactly how big are they?, New Atlas, 21 March 2013, Available (accessed 27 July 2018): https://newatlas.com/camera-sensor-size-guide/26684/

[13] M. Pylkkö, Valokuvauksen perusteet, Docendo, Jyväskylä, 2017, 192 p., ISBN: 9789522913319

[14] S. McHugh, Dynamic range in digital photography, Cambridge in color, Available (accessed 21 March 2019): https://www.cambridgeincolour.com/tutorials/dynamic-range.htm

[15] K. Kuiper, Viewfinder, Encyclopædia Britannica, Inc., 2009, Available (accessed 15 May 2019): https://academic-eb-com.libproxy.tuni.fi/levels/collegiate/article/viewfinder/75322

[16] BIP2-1920-30c - Basler IP Fixed Box, Basler AG, Available (accessed 16 July 2018): https://www.baslerweb.com/en/products/cameras/network-cameras/ip-fixed-box-cameras/bip2-1920-30c/

[17] Canon EOS 6D Mark II Tekniset Tiedot, Canon Oy, Available (accessed 21 August 2018): https://www.canon.fi/cameras/eos-6d-mark-ii/specifications/

[18] C. M. Bishop, Pattern Recognition and Machine Learning, Springer Science+Business Media, LLC, 2006, 758 p., ISBN: 9780387310732

[19] Y. LeCun, Y. Bengio and G. Hinton, Deep Learning, Nature, Vol. 521, Iss. 7553, , May 2015, pp. 436-444.

[20] T. Hastie, R. Tibshirani and J. Friedman, The Elements of Statistical Learning: Data Mining, Inference and Prediction, Springer Science+Business Media, LLC, 2009, 745 p., ISBN: 9780387848570

[21] D. Kingma, D. Rezende, S. Mohamid and M. Welling, Semi-supervised Learning with Deep Generative Models, June 2014, Available (accessed 5 May 2019): https://arxiv.org/abs/1406.5298

[22] Q. Jarosz, Neuron-no labels2, Image, License: CC BY-SA 3.0, 2009.

[23] D. Graupe, Principles of Artificial Neural Networks, 2nd edition, World Scientific, Singapore, 2007, 303 p., ISBN: 9789812770578

[24] W. Rawat ja Z. Wang, Deep Convolutional Neural Networks for Image Classification: A Comprehensive Review, Neural Computation, Vol. 29, Iss. 9, pp. 2352–2449, The MIT Press Journals, 2017.

[25] I. Goodfellow, Y. Bengio and A. Courville, Deep Learning, MIT Press, 2016, 775 p., Available (accessed 7 May 2019): http://www.deeplearningbook.org/

[26] M. Kordos, M. Blachnik, and T. Wieczorek, Temperature Prediction in Electric Arc Furnace with Neural Network Tree, Artificial neural networks and machine learning - ICAN 2011, pp. 71-78, Springer Berlin Heidelberg, 2011, EISBN: 9783642217388

[27] D. L. Waidelich, Rectifier, AccessScience, 2014.

[28] H. N. Mhaskar, Function approximation with zonal function networks with activation functions analogous to the rectified linear unit functions, Journal of Complexity, Vol. 51, pp. 1-19, 2019.

[29] L. Jiang-Jing, S. Xiao-Hu, H. Jia-Shui, Z. Xiang-Dong and Z. Xi, Data augmentation for face recognition, Neurocomputing, Vol. 230, pp. 184–196, 2017.

[30] M. Majnik and Z. Bosnic, ROC analysis of classifiers in machine learning: A survey, Intelligent Data Analysis, Vol. 17, Iss. 3, pp. 531–558, 2013.

[31] Z. C. Lipton, C. Elkan and B. Naryanaswamy, Thresholding Classifiers to Maximize F1 Score, University of California, 2014, Available (accessed 10 May 2019): https://arxiv.org/abs/1402.1892

[32] A. Piotrowski and J. Napiorkowski, A comparison of methods to avoid overfitting in neural networks training in the case of catchment runoff modeling, Journal of Hydrology, Vol. 476, pp. 97-111, 2013.

[33] B. Raj, Data Augmentation | How to use Deep Learning when you have Limited Data—Part 2,11 May 2018, Available (accessed 9 May 2019): dhttps://medium.com/nanonets/how-to-use-deep-learning-when-you-have-limited-data-part-2-data-augmentation-c26971dc8ced

[34] A. Jung, A Python library: Imgaug, Github, 2019, Available (accessed 9 May 2019): https://github.com/aleju/imgaug

[35] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever and R. Salakhutdinov, Dropout: A Simple Way to Prevent Neural Networks from Overfitting, Journal of Machine Learning Research, Vol. 15, pp. 1929-195, 2014.

[36] EDSDK3.8.0 API Programming Reference, Canon, 2018.

[37] AZURE-1236ZM, AZURE Photonics U.S.A, Inc., 2000-2013. Available (accessed 16 July 2018): http://www.azurephotonicsus.com/products/azure-1236ZM.html

[38] Canon EF 24-105mm f/4L IS II USM, Canon, 2019, Available (accessed 12 March 2019): https://www.canon.fi/lenses/ef-24-105mm-f-4l-is-ii-usm-lens/specifications/

[39] O. Turtiainen, Product Intelligence Professional, Canon Oy, Interview 31 May 2018.

[40] SNC-VB770, Sony, 2019, Available (accessed 8 March 2019): https://pro.sony/en_NI/products/fixed-cameras/snc-vb770

[41] SONNAR® T* FE 55 MM F1.8 ZA: Full Specifications and Features, Sony, 2018, Available (accessed 12 March 2019): https://www.sony.com/electronics/camera-lenses/sel55f18z/specifications

[42] Exmor R 2009: Back-illuminated CMOS Image Sensor, Sony Semiconductor Solutions Corporation, 2019, Available (accessed 15 March 2019): https://www.sony-semicon.co.jp/products_en/IS/sensor2/technology/exmor-r.html

[43] Sony develops back-illuminated CMOS image sensor, realizing high picture quality, nearly twofold sensitivity(*1) and low noise, Sony, 11 June 2008, Available (accessed 16 March 2019): https://www.sony.net/SonyInfo/News/Press/200806/08-069E/index.html

[44] Keras-team, Keras Documentation: The Python Deep Learning Library, Available (accessed 5 May 2019): https://keras.io/

[45] NVIDIA cuDNN, NVIDIA Corporation, 2019, Available (accessed 5 May 2019): https://developer.nvidia.com/cudnn

[46] CUDA zone, NVIDIA Corporation, 2019, Available (accessed 5 May 2019): https://developer.nvidia.com/cuda-zone

[47] Anaconda Distribution, Anaconda Inc., 2019, Available (accessed 5 May 2019): https://www.anaconda.com/distribution/

[48] Keras-team, Keras Documentation: Keras Applications, Available (accessed 14 May 2019): https://keras.io/applications/

[49] S. Masood, A. Rai, A. Aggarwal, M. N. Doja and M. Ahmad, Detecting distraction of drivers using Convolutional Neural Network, Pattern Recognition Letters, 2018.

[50] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto and H. Adam, MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications, Google Inc., 2017, Available (accessed 6 May 2019): https://arxiv.org/abs/1704.04861

[51] H. Huttunen, F. S. Yancheshmeh and K. Chen, Car Type Recognition with Deep Neural Networks, Tampere University of Technology, Visy Oy, Tampere, 2016, Available (accessed 6 May 2019): https://arxiv.org/abs/1602.07125

[52] Keras-team, Keras Documentation: Core Layers, Available (accessed 14 May 2019): https://keras.io/layers/core/