Object Detection for Analysis of Piglets Feeding Behaviour in an Incubator

Master's Thesis
VGIS1043

Jonatan Emil Simonsen

Electronics and IT
Aalborg University
10th Semester Vision, Graphics and Interactive Systems
Frederik Bajers Vej 7
DK-9000 Aalborg

AALBORG UNIVERSITY

STUDENT REPORT

10th Semester
Vision, Graphics and Interactive Systems
Electronics and IT
Aalborg University
Frederik Bajers Vej 7
9000 Aalborg
http://www.aau.dk

Title:
Object Detection for Analysis of Piglets Feeding Behaviour in an Incubator

Theme:
Vision, Graphics and Interactive Systems

Project Period:
Spring Semester 2022

Project Group:
VGIS1043

Participant(s):
Jonatan Emil Simonsen

Supervisor(s):
Thomas B. Moeslund

Page Numbers: 65

Date of Completion:
June 2, 2022

Abstract:

The piglets in the pig industry have multiple difficulties in the beginning of their life. SEGES Innovation is working on an incubator using artificial teats to reduce the number of piglets dying shortly after birth. This thesis proposes the usage of object detection by using variations of a Faster R-CNN. The architectures are trained and tested using a custom dataset, which have been gathered in collaboration with SEGES Innovation. The models strive towards detecting piglets using the artificial teats to feed in the incubator, and determine the piglet drinking. The results shows, that the piglet could be detected drinking with an AP of 39.5% using a threshold of 0.7, and distinguishing between the piglets had an AP of 27.4%. Overall the models did not achieve to detect a piglet feeding using the artificial teats with high precision.

Preface

This report is submitted as a thesis project for the Master's Programme in Vision, Graphics and Interactive Systems at Aalborg University under the supervision of Thomas B. Moeslund. The focus of this thesis has been to develop a computer vision system that utilises several modules to acquire information of piglets in an incubator.

Citations done in this report follows APA-style, which means that the reference list is alphabetised, and citations are (Author's surname / Name of source, year).

This report is intended for readers who have an interest within the fields of computer vision and livestock monitoring. Moreover, it is assumed that the reader has basic knowledge of computer vision and deep learning. However, since many techniques within deep learning are moderately new, these techniques will be covered in the thesis.

I would also like to thank my supervisor Thomas B. Moeslund for guidance during the thesis. Thanks to Vivi Aarestrup Moustsen from SEGES Innovation for providing insight into the incubator, and help in regards of understanding piglets. Thanks to SEGES Innovation for great company guidance during my thesis.

Aalborg University, June 2, 2022

Contents

Chapter 1

Introduction

The production of pigs in Denmark is a large industry, which has existed for more than 100 years [9]. The industry is still increasing and is expected to continue to increase. In 2021 Denmark produced around 33 million pigs, which is an increase of 0.4 – 0.5 million compared to 2020 [11].

One of the major issues within the industry is the mortality rate of piglets, which has been a problem for many years. In 2020 the mortality rate was 23,1 percent, which is the second highest between 2011 and 2020. This problem has occurred during the process of making hybrid pigs, even through the number of live pigs five days after birth has been a prioritized aim more than just increasing the litter size. The number of teats on the sow and her milk production for pigs do not correspond to the number of piglets farrowed. This means the piglets do not obtain enough colostrum to be able to fight off bacteria and protect themselves against starvation and hypothermia [18].

The company SEGES Innovation has tried to solve this issue and reduce the death rate of piglets by creating an incubator with artificial teats. Since the summer of 2021 SEGES Innovation have been working on this project and tested it. SEGES Innovation have obtained good results and have been able to observe, that the piglets drink and can therefore eat more, and it helps them gain weight [17].

One of the problems SEGES Innovation encounters is the testing of their incubator project. While testing, technicians from SEGES Innovation must be present, so they can measure the weight of the piglets during the six-to-eight-hour period the piglets are in the incubator. This is the only way for SEGES Innovation to tell how well the incubator works. This approach does not allow for mass testing of the incubator, since there is a limitation in workers. This also means the testing of the incubator and development becomes more cost heavy, and it will take more time.

The overall goal for this thesis is to look into Computer Vision (CV) based solutions, which can provide assistance in overseeing the piglets in the incubator

and provide information regarding their drinking behaviour. This would reduce the required manpower and it would allow multiple incubators to be utilized at the same time. This would eventually over time speed up the development and research process of the incubator, so a final product can be produced and help reduce the mortality rate of piglets.

Chapter 2

Analysis

This chapter will cover the general information of pig production in Denmark to reach a fundamental understanding of producing piglets and discover the current issues within the topic. Afterwards, an exploration of existing systems will be covered within the field of computer vision. Furthermore, topics within recognition and computing will be introduced, that could provide a solution to the discovered issues. At the end, possible problems are introduced, which should be taken into consideration for the computer vision system for piglets in Denmark.

2.1 Pig Production in Denmark

The production of pigs has been around for 100 years in Denmark, and in 2020 the export of pigs was worth more than 35 billion kroners and exported to 124 countries. The Danish pig farmers are leading within quality, animal welfare and traceability, which are the main reasons for the Danish farmers to be among the leading farmers in the world [9] [10]. In 2021 there were 210.000 lactating sows in Denmark, which on average nursed 12,5 piglets, i.e., 2.631.000 piglets in total [35]. To acquire fundamental knowledge about piglets, the following section will cover the general piglet production in Denmark, and the current issues related to piglets.

2.1.1 Production Cycle

The process of producing pigs in Denmark is a five-step process. This consists of the mating area, gestation barn, farrowing barn, weaners and finishers.

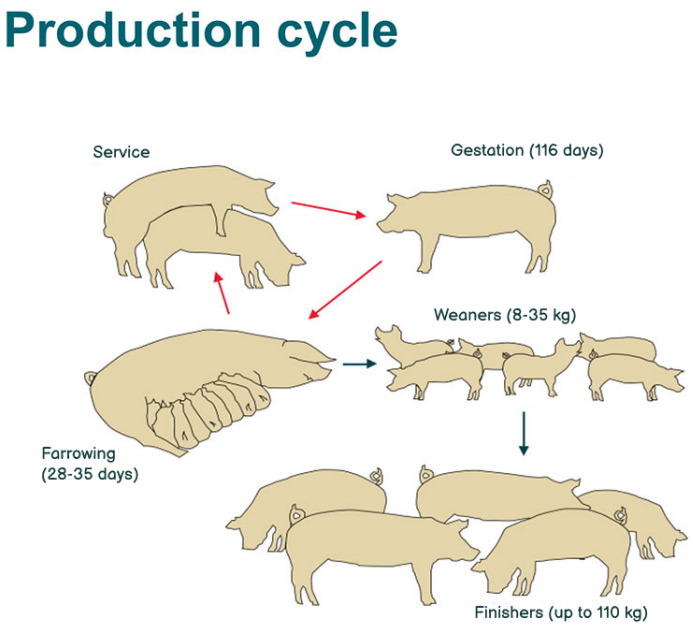

Figure 2.1: The production cycle of pigs. The original image is taken from [7] and have been adjusted.

Mating Area

In the mating area the sow is inseminated with semen from a boar. The semen is gathered at a boar station, which aims to provide the best quality semen to produce pigs with higher quality genes. This is done by using different pig races and combining the traits from the different pigs. Resulting in healthier, faster growing, and higher meat quality pigs. The general mating area provides space for the sows, so they have the possibility to walk around in more open spaces, but it is not compulsory [9].

Gestation Barns

After the sow has been confirmed to be pregnant, it will be moved to a gestation barn within four weeks. The sows will then spend their time in this type of area until five days before the expected delivery [9].

Farrowing Barns

The sow delivers the piglets in a farrowing barn. The area is designed in a way, so each sow and their piglets have their own pen. The sow is contained in a crate, which is implemented to prevent her from laying on the piglets. The sow stays with her piglets in the farrowing barn until the piglets reaches the age of four

weeks. Then the sow returns to the mating area, and the piglets are moved to the weaner area [9].

Weaner Area

For the weaner area, the small pigs are now in groups of 30 to 35 pigs. The goal of the weaner area is for them to gain weight. The areas are designed with small shelters in the back, so the temperature can be kept higher, since the small pigs needs a warmer environment, than the larger pigs. The weaners stay in this area until they weigh around 25-30 kilograms. They will then be moved to the last area, which is the finishers [9].

The Finishers

This area contains around 15 pigs per area, and the goal is the same as the weaner area. The pigs are now bigger, and do not need the same living criteria as for smaller pigs. The pigs will be gaining weight until they reach around 105 kilograms. The pigs are then transported to slaughterhouses, where they then will be turned into meat. By the end, the pigs are between five to six months old [9].

2.1.2 Pig Breeds

Multiple varieties of pig breeds are used within production in Denmark. There are three main breeds used in the industry, which have been picked to produce healthy, bigger and fast-growing pigs with a good meat quality. The three breeds are usually cross-bred to achieve the best qualities from the different breed, which creates hybrid pigs. The most commonly used hybrid is a mix between the Danbred Landrace and the Danbred Yorkshire, which is used to create the sow and usually the boar is a Danbred Duroc [9].

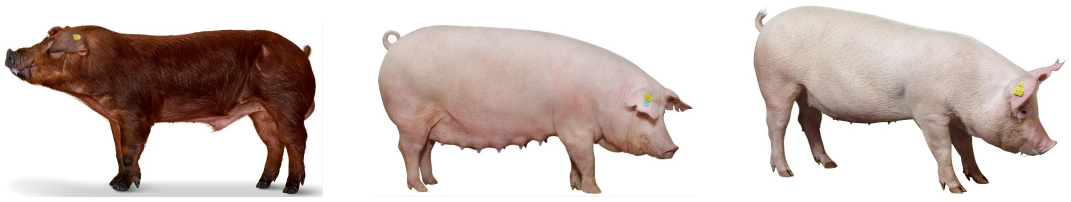

Figure 2.2: The images show the three breeds of pigs used within the Danish pig industry [8]. Left: Danbred Duroc; Middle: Danbred Landrace; Right: Danbred Yorkshire.

Danbred Duroc

The Danbred Duroc originates from USA and Canada, where it was exported to Europe between 1977 to 1979. The Danbred Duroc is primarily used for boars used

for breeding. This breed is known for a high daily growth and a high meat quality, while it does not eat as much as other breeds. In relation to breeding, it has also proved to provide piglets with a higher viability compared to other breeds [8].

Danbred Landrace

The Danbred Landrace is one of the main breeds used to produce sows for breeding in Denmark. This breed is known for proving a larger litter of piglets and has proven to have good motherly instincts. Besides the good genetics for breeding, the breed also produces a strong pig with good legs, and it also has a high meat percentage [8].

Danbred Yorkshire

The Danbred Yorkshire originates from England, and it is the other main breed used to produce sows for breeding. This breed has many of the same qualities within breeding as the Danbred Landrace, since it produces larger litters and has good motherly qualities. The Danbred Yorkshire additionally provides fast growth, low food intake and good meat quality. The Danbred Yorkshire provides multiple good production values, which the Danbred Landrace and Danbred Duroc also provides [8].

2.2 Piglets in Production

A newspaper article from 2021 touches on a problem within the pig industry, which is the mortality rate of piglets. A study has shown that 26.575 piglets die each day and the reason for the high mortality rate is still unknown.

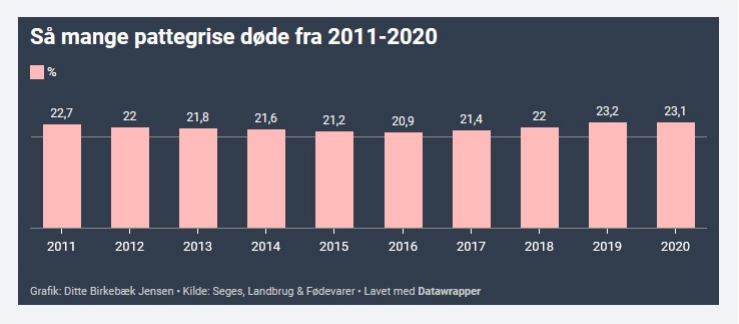

Figure 2.3: The figure presents the development in the mortality rate of piglets in the Danish pig industry from 2011 - 2020 [18].

There could be multiple reasons to this problem, the most common issues for piglets are described in section 2.2.1. One of the main issues is the number of

piglets born from one sow. A sow can give birth to more than twenty piglets, but the standard sow only has fourteen teats. An approach to this issue has been using nursing sows, but this only solves part of the issue. This is due to the colostrum intake of the piglets. The colostrum is very essential for healthy piglets, since it provides antibodies. The other part of the issue, its that the sow does not produce colostrum depending on the number of the piglets birthed. This results in less antibodies for the piglets in a larger litter, which also collides with larger litter piglets having a lower birth weight [18].

2.2.1 The Challenge with Piglets

There are five main challenges with piglets, which all can result in a higher mortality rate among piglets. It is therefore important to be aware of these.

Overlay

This refers to farrowing barns, since it is a crate restraining the sow. This is a method, which was implemented in the 1960s. The implementation was due to a higher mortality rate in piglets. The farmers consistently experienced a mortality above 15% in some litters, which was a result of the sow stepping or laying on the piglets [22].

Chilling

The problem of having cold farrowing pens typically affects the piglets, since they are weak, slow-moving, and huddle in groups. This issue can typically be seen affect piglets below the age of one month, since they do not have the capability to produce heat themselves. The piglets require a room temperature of 20 degrees Celsius, and they are typically provided a brighter area, that provide 35 degrees Celsius. The warmer area is provided to make the piglets avoid huddling below the sow [22].

Failure of Colostrum Intake

Piglets are born without any protective materials. It is important for the piglets to partake in drinking milk from the sow. The first milk from the sow is called colostrum, since it differs in color and provides the protective materials for the piglets. The farmers can also typically see a higher number of sick piglets the following days, if several piglets did not eat enough colostrum. The affected piglets may experience a form of blood poisoning. This result in the piglets being dull and seeming weak while walking. These piglets also appear cold, shivering, while portraying poor breathing and blue extremities. They do not eat properly, hurdle

in groups and may die. It is therefore important for the piglets to gain access to the colostrum from the sow [22].

Diarrhea

This problem usually occurs for one-to-seven-day old's. The problem stems from the piglet eating Escherichia coli (ETEC), which is in the farrowing pen. This is usually done soon after the birth, and the bacteria then attaches to the piglets' intestines. The issue can be resolved by treating the piglets with oral antibiotics and re-hydration electrolytes, which will replace the body fluids [22].

Birth Defects

A range of different birth defects can occur. Some examples could be one testicle for male piglets, fewer teats or small vulvas for female piglets. Another birth defect for male piglets is a condition called inguinal, which is the case where male piglet's intestines are mixed with the testicles. Another birth defect is called spray-legs, where the piglets stand with its back legs splayed out sideways and forward. This result in difficulties for the piglets, since they have problems standing on their hind legs [22]. An employee from SEGES Innovation have stated, that the issue of birth defects is not as common in Denmark as elsewhere.

2.2.2 Piglet Feeding Behavior

When the piglets are in the farrowing pens, they do not have constant access to milk. The milk is only available to the piglets during milk letdown. The milk letdown is when the milk is present in the teats of the sow, and for this to occur the teats of the sow must be stimulated by the piglets. The process of feeding is split into five phases. During the first phase the piglets gather around the sow's udder and find their respective teat. For the piglets there is a hierarchy, which is usually based on their weight and strength. The piglets have chosen a teat based on this hierarchy. During phase two they massage the udder, and in phase three they suckle on the teats with slow mouth movements. The milk letdown then happens during the fourth phase, and the milk letdown lasts for ten to twenty seconds. The piglets will then massage the udder again during the fifth phase, which pre-establishes the right amount of milk for the next milk letdown. These five phases will happen every 40 – 60 minutes, and the piglets obtain 45 – 55 g of milk per milk letdown. The moment of the milk letdown can be determined by observing the piglets, since they will start having rapid mouth movements during the milk letdown [27].

2.3 Existing IT Solutions

Within the recent years, where the animal welfare has been put more into focus, there has been developed IT solutions to some of the issues within the pig industry. This is changing the industry and providing a more tailored treatment for the individual pig.

An example of this is an article from 2021, where the focus is to obtain tracking of piglets. In order to track health, welfare and productivity. The information can then be utilized to gain a deeper understand of the piglets and overtime help farmers to provide the needed requirements of the individual piglets. This solution is produced from a combination of a two-stage CNN-based detector, by using a ResNet-101 CNN model and a Faster R-CNN network. This method achieved an average precision (AP) of 0.5 and recall values of 0.978 and 0.987, and this was achieved by discarding detection with a classification lower than 0.9 [12].

Another example could be DigiPig, which utilizes an automated monitoring system to track of the body, head and tail of pigs. This system aims towards using detection as a welfare evaluation system for the farmers. The detection is based on the shape of the pigs instead of the faces, since a video setup to recognize faces would prove to be difficult. The system also strives towards analyzing pigs' behavior towards each other from head to tail. The DigiPig has also been utilized for piglets, where fully convolutional networks were used, and they achieved an accuracy of 98% for drinking, 95% for feeding and 88% for nursing. These numbers are based on the accuracies for behavioral classification compared to manual scoring [24].

Other approaches, besides CV based solutions, are with the usage of sensors. In an article from last year a sensor-based solution for tracking the individual pig's activity, vocalization and temperature was developed. The solution utilizes a sensor board in an ear tag, which then sends information to a deep learning model to evaluate the provided information. The current goal of the technology is optimized management of farm animals, detecting sick animals, reduce the medical cost and reduce the used amount of antibiotics [26].

The currently primarily utilized tool within pig farming is sensors, Closed Circuit TeleVision cameras and chips. These technologies are provided by multiple companies like SKOV, FarrowTech, SEGES Innovation, etc. These different companies provide products to track the room temperature, water and feed intake, pigs' growth rate and activity level [13].

2.4 Computer Vision Based Approaches

Based on the existing IT solutions found within the field of the pig industry, CV was utilized multiple times. CV is a concept within artificial intelligence (AI), that allows computers to gather information from digital images, videos and other types of visual inputs. The goal of CV is to imitate the human perception, so a computer based on visual input can perform pre-defined tasks. Since the visual input is essential within CV, multiple techniques of Image Processing can be used for image analysis. CV is a subfield of Deep Learning and AI, and CV is split up into multiple topics like object identification, facial recognition, image segmentation and image classification [2].

The following subsections will cover CV theory and methods to determine the possible approaches of solving the current issues of piglet's eating behavior.

2.4.1 Recognition

Recognition is one of the tasks, which can be conducted with CV to recognize and get information from visuals in an image. There are three common tasks within CV in combination with deep learning. The tasks are image classification, object detection, and image segmentation, and they will all be covered in the next subsections. The task of annotating data will also be covered in one of the following subsections.

2.4.2 Image Classification

Classification is part of almost all CV tasks, since the purpose of classification is to categorize and assign labels to groups of pixels. Classification can be done in two ways: unsupervised and supervised classification.

Unsupervised classification is a method, that does not require annotations, since it allows the algorithm to detect patterns by itself during training. An example of this could be K-means, where the algorithm groups objects into k groups based on extracted features from the training.

Supervised classification uses annotations and learns from the groundtruth, which is the images where a person has defined objects and groups beforehand [3].

2.4.3 Object Recognition

Object recognition is a term used to describe several CV tasks, where the task is to identify objects in images. There are three general terms, which are covered

by object recognition: image classification, object localization and object detection. Image classification is defined in the section above. Object localization strives to locate an object in an image and define the location in the image of the location. This is usually done with bounding boxes, or it can be like the existing solutions, see section 2.3, and be several points, which defines the location of an object. Object detection is trying to locate an object in an image like object localization, but with the addition of detecting the type or class of the object [4].

2.4.4 Image Segmentation

Image segmentation is a method used to reduce the complexity of an image. This is done by grouping pixels in an image into subgroups, which is called image segments. This process also makes the analysis of the image easier, since pixels are already grouped. This can be utilized for other deep learning models like object detection. The image segmentation can pass regions of an image, based of the different segments in the image, which reduces the processing time for the object detection model. There are two approaches within image segmentation: similarity approach and discontinuity approach.

Similarity approach is detecting similarities between pixels in the image and will create a segment based on it. These similarities are based on a predefined threshold, and this approach is often used for clustering algorithms.

Discontinuity approach uses the discontinuity of a pixel's intensity values in an image. This is often used in edge detection techniques, where it makes intermediate segmentation results at first, and then later continue the process to achieve the final segmented image [37].

2.4.5 Image Annotation

Annotating images is a commonly used method for deep learning models. Labeled data is used for supervised learning, so the training of the model is based on the annotations. This also means, that the quality of the model relies on the quality of the annotations. There are multiple ways of annotation images, which all have different pros and cons related to them [28].

Bounding Boxes

Bounding boxes are the most used type of annotations in CV. The purpose of the bounding boxes is to define the location of the targeted object. The localization of the targeted object is defined by the x and y axis coordinates in the upper left corner and the x and y axis coordinates in the lower right corner. The bounding boxes are

primarily used for tasks related to object detection and image localization. This method is time effective, since the targeted object can be defined by defining two to four locations to form the bounding box [28].

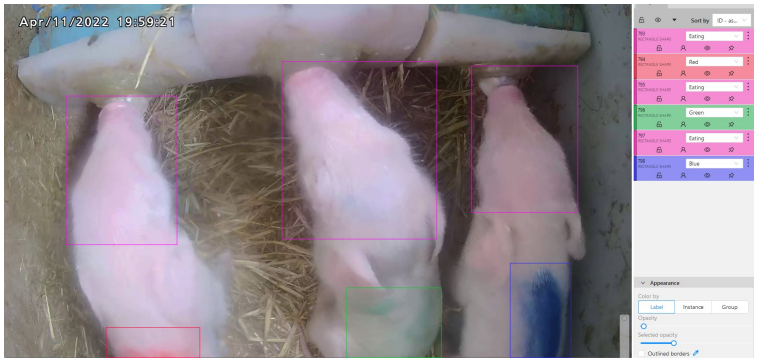

Figure 2.4: An example of the usage of bounding boxes.

Polygonal Segmentation

Polygonal segmentations are primarily used for more complex objects, or if higher precision is preferred. This allows the annotator to precisely define the shape of the targeted object, but it comes with the downside of time, since it is a time-consuming procedure [28].

Semantic Segmentation

Semantic segmentation is using image segmentation, but with the purpose of labeling, see section 2.4.4. It uses pixels for annotations by assigning each pixel to a class, which is done by using color value or pixel intensity value. This method is utilized in cases, where the entire environment is important. Examples of this could be robotics or self-driven cars. Since the method works on a pixel-based method, it would be very time consuming, but it provides a lot of information [28].

3D Cuboids

3D cuboids are like bounding boxes, but with the addition of a third dimension. This provides depth information as well as the size of the targeted object. This method is used within self-driven cars, since it allows the car to measure the distance of objects from the car based on 3D positions. This method is difficult to utilize, since a 3D cuboids must be defined on a 2D image, and it might cause some trouble, which will decrease the efficiency of annotating [28].

Key-Point and Landmark

Key-points and landmark is a method, where multiple points are positioned, and the dots are connected to define objects. This method is used for smaller objects, which differs in their shape. This type of annotation is used for facial feature detection, poses, and it is also utilized for some of the existing IT solutions for pigs. It is done by defining the head, back and tail, see section 2.3. The method takes longer to execute compared to bounding boxes and will be about the same time efficiency as polygonal segmentation [28].

Lines and Splines

This method uses lines and splines to define objects. It is primarily used within the field of autonomous vehicles, since it is used to define roads, and the lanes on the road. This method is very time effective, but it is very limiting, since it requires very specific requirements for the targeted object [28].

2.5 Multiple Ways of Solving the Issue

Based on the covered material, there could be multiple ways of solving the issue of detecting when a piglet is eating. The main issues for the project is the limited space in the incubator for placing a camera, and that there is no existing dataset.

One way of approaching the problem would be to use tracking. Object tracking is the process of detecting an object in a video, and then continue to track the object, which is provided with an unique ID. There are four components of a tracking algorithm, which is target initialization, appearance modeling, motion estimation, and target positioning. These four components allow us to estimate the position of the target in the video, adapt to changes in the target due to motion blur, lighting, etc., predict the movement of objects, and motion estimation. Tracking can also be used on multiple objects, which in this case allows for tracking of multiple piglets.

Object tracking can be implemented as online or offline trackers. Offline trackers do not run in real time, and it is conducted by analysing pre-recorded video. This also means that, while tracking objects, past and future frames can be utilised to achieve a higher precision. Online tracking runs in real-time, and it can therefore only utilise past frames to achieve a precise tracking, but it allows the predictions to be available at once.

For the case of this thesis, it would only be possible to use offline training trackers, since the data for the thesis have been gathered. One of the most used

architectures for offline tracking is a convolutional neural network based one.

The approach is to train the CNN on the dataset, which would consist of videos. During testing the CNN will then track an object by initial detection, and then proceed to use the position provided by the past frame and the current frame to conduct tracking of objects. An example of this is the GOTURN model, which has a high performance, and it is able to process 100 frames per second (fps) on GPU powered machines [33].

2.5.1 Convolutional Neural Network

A Convolutional Neural Network (CNN) differs from a traditional Artificial Neural Network (ANN), which is inspired by how the human brain is functions. An ANN is a network of connected nodes, which is referred to as neurons, which work together to learn from an input to optimise the output.

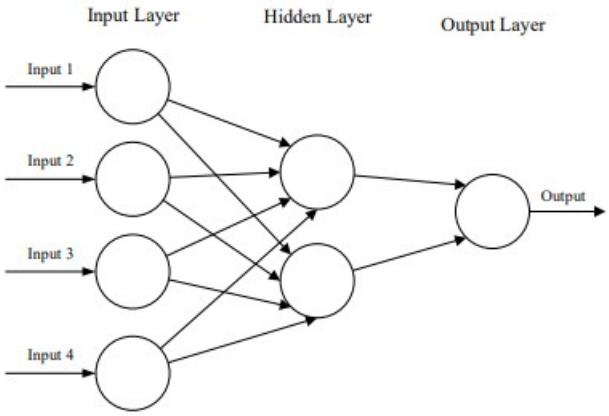

Figure 2.5: A simple three layered feedforward neural network (FNN), which consists of a input layer, a hidden layer, and an output layer [25].

A typical ANN consist of a input layer, hidden layers, and an output layer. The input layer receives a input, which could be in form of an image. The image is then distributed to the hidden layers, that look at features in the image. The weight that is present at each node will then adjust accordingly to achieve a higher output. The output is the result of the model. In a model there is only one input and output layer, but there can be multiple hidden layers. Having multiple hidden layers are usually referred to as deep learning.

A CNN differ from an ANN in its architecture, since a CNN consists of an input layer, convolutional layers, ReLU, pooling, flatten, fully connected and softmax

layers [25].

The following sections will explain the different aspects of a CNN.

Convolutional Layers

The convolutional layer is the main computational workload of a CNN. The convolutional layers performs dot product computation between two matrices, which is the matrix of the kernel and the restricted portion of the input. During the forward pass, the kernel slides across the height and width of the input image, and produces a representation of the image called an activation map. The movement of the kernel is based on the stride and the padding. The stride decides how many pixels the kernel moves across for each action. Padding is the action of adding layers of zeros to the input image. This method can be useful to be able to analyse edges of an image, since the kernel might not be able to process the edge of an image otherwise.

Below, the equation for a convolutional layer can be seen.

$$W_{out} = \frac{W - F + 2P}{S} + 1$$

For the equation W_{out} is the output volume of the input of size $WxWxD$. The F is the spatial size of the kernel, S is the stride and 2P is the padding multiplied with two [23].

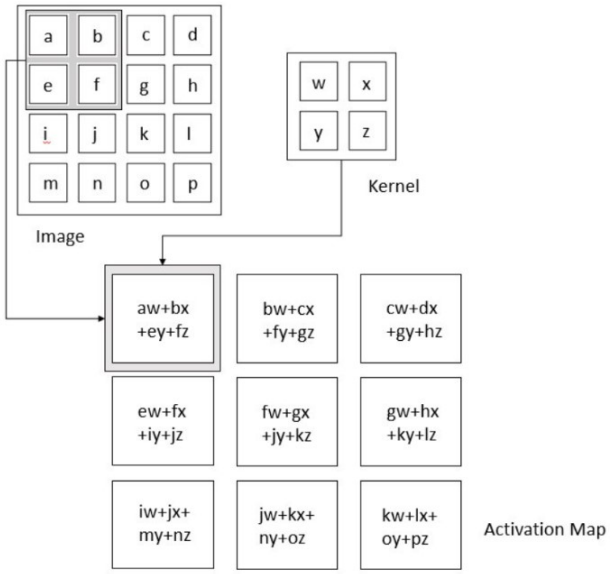

Figure 2.6: The convolutional layer using the kernel, stride and padding to produce the activation map [23].

As seen in figure 2.6, the kernel is smaller than the input image. An input image can consist of thousands of pixels, and the kernel creates the activation map by computing the dot product and detecting meaningful information.

Pooling Layer

After the convolutional layer the next step is usually a pooling layer. The purpose of the pooling layer is to reduce the spatial size of the activation map, which reduces the computations and the weights. The pooling operation processes the activation map on a individual level. There are multiple pooling operations, but the primarily used method is max pooling, which produces the maximum output from a neighborhood.

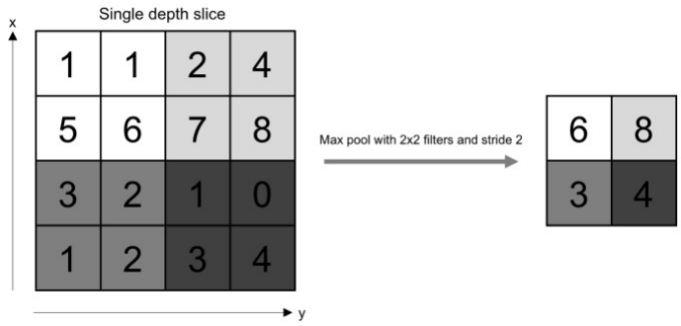

Figure 2.7: An example of the pooling operation, which uses a $2x2$ filter and stride set to 2. This example shows the reduction of the activation map created from the convolutional layer [23].

The output of the pooling operation is based on the size of the activation map $WxWxD$, the spacial size F of the kernel and the stride S. This produces the equation:

$$W_{out} = \frac{W - F}{S} + 1$$

The reduction of the activation map will reduce the amount of computational operations and result in a more efficient model.

Fully Connected Layer

The neurons in this layer is fully connected with all neurons, which means it cannot be computed by using matrix multiplication. The goal of the fully connected layer is to conduct classification. Typically the activation map is flattened and forwarded to the fully connected layers. The dot product is then computed between the flattened input and the weights, and the output is produced. Since the fully

connected layers uses a linear transformations, a non-linear activation function is applied to achieve a non-linear transformation of the weights and flattened input vector.

The most commonly used non-linear transformations are sigmoid and ReLU. Sigmoid takes a real number and converts it to a value between 0 and 1. This is done for computational purposes, since smaller values makes the computational workload lessened [38].

$$\sigma(\kappa) = \frac{1}{(1 + e^{\kappa})}$$

The usage of sigmoid provides some undesired results, since the gradient becomes almost zero, which means the backpropagation will effectively "kill" the gradient.

ReLU, which is short for The Rectified Linear Unit, which is a very popular non-linear transformation also have some downsides. ReLU outputs the maximum value between zero and the input value, which means all negative values are set to zero.

$$f(x) = max(0, x)$$

In the equation above x is the input value, and based on this the function can be rewritten to the figure 2.8.

$$\boldsymbol{f(x) = \begin{cases} 0, if\ x < 0 \\ x, if\ x \geq 0 \end{cases}}$$

Figure 2.8: The rewritten version of the ReLU equation [6].

The downside of this approach is the possibility of ReLU being fragile during training. This is due to the potential of large gradient values, which can result in no further updates of the neurons. This issue can however be resolved by using a fitting learning rate [23].

Backpropagation

The common ANN's are FNNs, but another type is using backpropagation. It is a method to fine-tune the weights of a neural network based on the error rate from the previous iteration. This can reduce the error rates and provide an increase in generalization. Backpropagation starts at the output layer and computes the gradient of the loss function for a single weight. It will then work from back to front and compute one layer at a time, and then adjust the weights, which is random from the start [19].

2.6 Multiple Ways of Solving the Issue

Another method for solving the problem of detecting the piglets eating could be to utilise object detection, see section 2.4.3. By detecting objects and classes in an image it could provide a usable approach. A way of solving the problem could be to create an architecture, which is trained to detect piglets drinking. This would only provide information for a feeding, and would not provide information about the specific piglet drinking. This means another model would be required, which would pass the information of the detected bounding box of the piglet drinking, and from there determine the specific piglet, who is drinking.

A way of solving could also be related to tracking, see section 2.5, where tracking of the snout could be conducted. This would provide information about each piglet and their position in the incubator. A 2D point could be defined at each artificial teat. A feeding could then be determined as a snout lining up with the 2D point.

2.6.1 Pros and Cons of Possible Approaches

In the former sections, multiple approaches to solving the issue have been mentioned. This section will cover the pros and cons of each method. The first idea provide tracking of each individual piglet, which removes the issue of determining which piglet is drinking. The con of this method would be that tracking does not distinguish between multiple classes, so the piglet would be detected as a piglet, and the piglet drinking would not make a difference.

The other method related to tracking of the snout could solve this issue, since a feeding would be the tracked snout matching up with the pre-defined 2D point. The main con with this approach would be the 2D point, since a stationary 2D point cannot be used. It was observed during the data gathering, that the piglets move around a lot. The piglets also stand in multiple positions while eating, and therefore a 2D point would either miss feedings, or it would register massage of the artificial teats as eating.

Figure 2.9: The images shows an example of a case, where the piglets after some time in the incubator no longer have any number or color indication on their back.

The approach about using object detection also creates it's own challenges. Through image annotations a piglet drinking and not drinking could be established, and learned by the used model. The issue of registering the specific piglet drinking might provide a challenge, which cannot be solved currently. During the data gathering the piglets had the numbers 1 to 6 on their back in the colors red, blue, and green. The main issue is that the piglets play with each other, massage the artificial teats, show dominance, etc. Over time the numbers on the back of the piglets disappeared, which in some cases leaves no indication of which piglet it is. This issue would not be present for tracking, since the piglets are tracked individually, and are assigned a unique ID. In the case of using object detection, and the color is not removed from the back of the piglet, then occlusion is still a problem, since object detection is operating on a single image level. Since the detection of the drinking is the focus, there would not be any past frame to utilise to determine the piglet, which is occluded by another piglet.

Based on the gathered material and the pros and cons, the thesis will do an implementation of an architecture utilising object detection. This decision is based on the fact, that the piglets feeding are the focus, and not tracking of the piglets. Another reason would be that the color on the piglets back can be replaced in the future. During the data gathering, I was informed by a SEGES Innovation employee that ear tags in multiple colors could have been used as well. So in the case of a low detection of the individual piglets, an improved could potentially be made in the future by using ear tags in different colors instead.

2.7 State-Of-The-Art Object Detection

The following section will cover some of the milestones in state-of-the-art object detection. The methods covered are from after 2014, where two-stage object detection algorithms was used, and deep learning detection was used too.

2.7.1 Faster R-CNN (2015)

The Faster R-CNN architecture was created in 2015, and it has since been seen as one of the most reliable deep learning models, which provides high performance. The Faster R-CNN consists of two modules, the first one is a deep fully convolutional network and the second one is a Fast R-CNN detector. The two modules are a single unified network used for object detection.

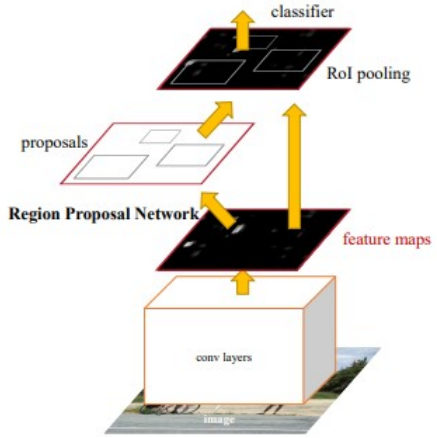

Figure 2.10: The architecture of a Faster R-CNN [31].

As seen in figure 2.10, the input is fed into the backbone CNN, the CNN outputs a activation map, which is then processed by the Regional Proposal Network (RPN).

The RPN takes in a feature map as input and outputs sets of rectangular object proposals, and each have been given an objectiveness score. When conducting convolutional operations on the input image, multiple region proposals are predicted. The maximum number of predictions for each location is κ. This means, that the regression layer has 4κ outputs encoding coordinates of κ boxes. The classification layer outputs 2κ scores, that estimates the probability of if there is an object for each proposal. The κ proposals are relative to κ reference boxes, which is called anchors. The anchor is placed at the center of the kernel. The anchors indicate the sizes and aspect ratios of a possible object in the location [31].

When the network goes through each pixel in the output feature map, it is checking if the anchors actually contains objects. The anchors' coordinates are then refined to provide regions of interest. The output feature map consists of about 20.000 anchors in total. Not all these anchors are used, since an anchor is considered acceptable under two conditions. The anchors needs to have a high Intersection over Union (IoU) with a groundtruth bounding box, or that the an-

chors have a IoU higher than 0.7 with any groundtruth bounding box. An anchor is discarded if the IoU is below the set threshold, and will then not be part of the RPN training.

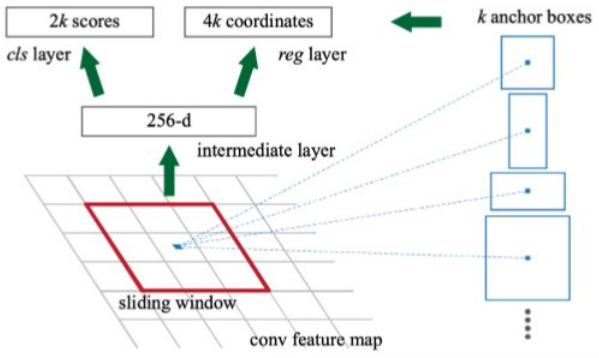

Figure 2.11: Region Proposal Network displaying the process of making predictions in a location [31].

The anchor works on a binary level, where the log loss is over two classes, which is object vs no object.

$$L_{Cls}(p_i, p_i^*)$$

This is the classification loss, where p_i is the output classification score of an anchor i, and p_i^* is the groundtruth label. If the anchor contains an object, then the regression loss function is activated.

$$L_{reg}(t_i, t_i^*)$$

For the regression loss function t_i is the prediction of the regression layer, which contains four variables: $[t_x, t_y, t_w, t_h]$. The regression target t_i^* is calculated with the formula below.

$$t_x^* = (x^* - x_a)/w_a, t_y^* = (y^* - y_a)/h_a, t_w^* = log(w^*/w_a), t_h^* = log(h^*/h_a)$$

For this formula the variables x, y, w, and h corresponds to the (x, y) coordinates of the box centre. h is the height, and w is the width of the box. x_a, x^* represent the coordinates of the anchor box and its corresponding groundtruth bounding box.

When the RPN has finished producing object proposals, the Region of Interest (ROI) Pooling is conducted. The feature map produced by the backbone CNN and the boundary box proposals are passed to the ROI Pooling layer. The object

proposals from the RPN, which have been combined with the feature map from the backbone are forwarded to the ROI layer. The region is then divided into a fixed number of sub-windows, and max pooling is performed on the sub-windows to produce a fixed size output. The fixed output feature maps are then forwarded to the ROI head, which consist of fully connected convolutional layers, a non-linear transformation functions like softmax and a regressor, which provides the output results. The softmax layer is used to predict the class scores and the fully connected convolutional layers are used to predict the bounding boxes of the detected objects [1].

2.7.2 G-RCNN (2021)

The Granulated R-CNN (G-RCNN) is an advanced version of a Faster R-CNN. When looking at the feature maps with the ROIs from a Faster R-CNN, the background might affect the detection accuracy and result in wrong classifications. The granules solves this issue, since they focus on the foreground, where there is a higher chance of a single or multiple objects to appear. This should not result in desired regions being removed, and it also reduces future operations. This process results in a reduction of cost-complexity for video object detection. The granules are clusters, which uses neighborhood, similarity, nearness or functionality. Tested on supervised video the neighborhood granules showed good results by using spatial and temporal similarity to localize object's regions in video frames. The granules are created based on the results from the pooling feature map from the CNN. By using granules, the classification task is only conducted on objects present in ROIs, which has shown higher detection accuracy, and performance superior to Fast R-CNN and comparable to Faster R-CNN [29].

2.7.3 YOLOv3 (2018)

YOLO stands for "You Only Look Once" and v3 is the third version of the architecture. The model is not as accurate as Faster R-CNN, but has show great performance for real-time detection without losing too much accuracy.

YOLOv3 uses Darknet-53 as the feature extractor, which contains 53 convolutional layers, batch normalization and Leaky ReLU. Leaky ReLU is a variation of ReLU, but it has a small slope of negative values instead of a flat slope. The slope coefficient is determined before training, which works well in cases of sparse gradients. Darknet-53 does not use pooling layers, and has a stride of 2 instead to reduce the feature maps. This helps to prevent loss of low-level features.

When working with YOLOv3, it is necessary to have a fixed height and width of the images, since the YOLOv3 model concatenates images into large batches,

and therefore all images should have a fixed height and width.

The output of the YOLOv3 model is a list of bounding boxes with the recognized classes. The detections are conducted by a classifier, which is a convolutional layer of 1x1 convolutions. Each detection in the YOLOv3 model has a six number representation $p_c, b_x, b_y, b_h, b_w, c$, which describes center coordinate, dimensions, objectiveness score, and C class confidence.

When YOLOv3 is operating on an image, it is only using one bounding box, which is used to detect any object. This is done by dividing an image into a grid equal to the final feature map.

The YOLOv3 model inputs an image into the model, and reduces the image using stride. The output is an encoded image, which is used to place anchor points. The model then goes through each box, and check the probability of that box belonging to a specific class. The model then searches for the highest probability score for a class, and places an anchor point. This may result in many overlapping bounding boxes, which is why non-max suppression is used. This is done through IoU, where the bounding boxes with a lower score are removed, and with boxes overlapping only one box is kept [30].

2.8 Summary

This chapter has covered general information in regards to pig production in Denmark. In relation to pig production, the common reasons for a high mortality rate in piglets have been described.

The chapter then proceeded to look into existing IT solutions utilising methods like tracking, and sensor based solutions. The following section covered material of different computer vision methods, which could be used to provide a solution.

Based on the material covered about computer vision, the information was used as a foundation for a discussion about different approaches to register a piglet drinking using the artificial teats. The methods based on tracking and object detection was then discussion with pros and cons. This resulted in an object detection approach, and the state-of-the-art object detection architectures was covered.

Chapter 3

Problem Statement

Based on the topics covered in the analysis, there can be multiple approaches to determine an active feeding. During the analysis, it was also determined that a CV approach could provide a solution, since CV provides many options in regards to detection. Based upon the previous chapter 2 and meetings with SEGES Innovation, it was decided that a high accuracy detection of an active feeding and tracking drinking time for each piglet could provide useful information in regards to the piglets feeding habits. This could also provide information for SEGES Innovation in further research in relation to optimizing the artificial teats. A system which can detect an active feeding and assign feeding time to the piglet drinking is therefore desired. The focus of this thesis will be on developing a model tailored to the SEGES Innovation incubator setup, and future studies within the scope of artificial teats with the goal of decreasing the mortality rate of piglets.

The majority of the existing systems are based on top-view cameras, but since this case is connected to a new study, the environment cannot be compared to other top-view camera setups. Besides the environment, most of the existing systems provide analysis of finishers and not piglets.

To achieve a precise and accurate model, methods within deep learning and CV will be utilized. This is based on the methods ability to solve complex detection and classification tasks.

This thesis will be based on a detection-based solution, where the detection algorithm is the priority, and it provides the essential information to determine an active feeding. Another goal is to define the specific piglet, who is drinking. This will also be done using a detection-based solution, with the purpose of assigning feeding time to individual piglets.

Success criteria have been utilized to provide objectives for the system and set criterion's to evaluate the model.

3.1 Success Criteria

Based on the covered material and meetings with SEGES Innovation, the following success criteria are created for the model.

SC1 During inference, the model should be able to process at least 5 images per second.

SC2 The model should achieve an Average Precision above 75%.

SC3 The model should detect the three different piglets with a Average Precision above 75%.

SC4 The model should output detected feedings and label a feeding to the correct piglet.

These success criteria will be summarized in chapter 7

Chapter 4

Data Analysis

The following chapter describes the gathered data for this thesis. All the data has been collect in collaboration with SEGES Innovation, and this thesis has contributed by pre-processing and annotating the data. This chapter will describe the process of gathering the data, the setup for the data collection, and the environment, that the piglets have been in while collecting the data. The pre-processing and the annotation of the data will be explored in a later chapter.

4.1 Data

Two cameras are utilized to capture the data of the incubator. The incubator is split into two sections, so that there are three piglets in each section of the incubator with access to one artificial teat each. The incubator is placed next to the farrowing pigsty. The data has been collected over a period of three days.

The initial data was collected during the month of February, and the remaining of the data was collected from the 11th – 13th of April. The initial data contains eight hours of recorded data. The amount of recording was reduced for the second collection, since SEGES Innovation discovered from a former study, that the active time period of the piglets is up to six hours. The remaining time from six to eight hours did not provide a lot of feeding data.

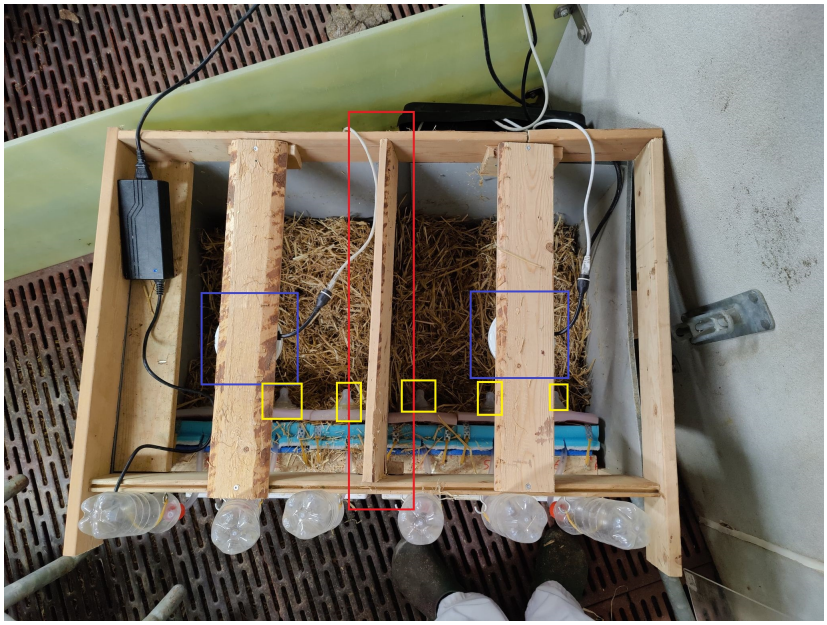

Figure 4.1: The setup of the incubator seen from above. Here the split of the incubator into two sections can be seen (red box), the artificial teats (yellow boxes) and the board with the cameras mounted on them (blue boxes).

This data also discovered a connection between the piglets using the artificial teats and the sow doing the call for milk letdown. The piglets have been seen to push right below the artificial teats, which is part of the massaging of the sow's teats, see section 2.2.2, and will then after start using the artificial teats. After the piglets first usage of the teats, the general behavior can no longer be followed, since the piglets have free access to milk, and are not limited to the ten to twenty seconds. The current only way SEGES Innovation can measure the amount of milk consumed by each piglet is by measuring their weight. An observation of the milk put into the incubator is not accurate, since the massaging of the artificial teats creates a loss of milk.

The images are extracted from the videos for the piglets in the two sections of the incubator. The two cameras have been set to two different recording settings. This means, that for incubator one the videos are running at 25 frames per second (fps) and the camera in incubator two have been running at 10 fps. The images have been extracted using VLC media player. This have been done using it's video filter feature, which provides the opportunity of saving images from a video.

Incubator	Images	Time Duration
Incubator-1-1	19.558	6 hours
Incubator-1-2	19.569	6 hours
Incubator-1-3	19.531	6 hours
Incubator-1-4	19.000	6 hours
Incubator-1-5	19.226	6 hours
Incubator-2-1	13.731	6 hours
Incubator-2-2	13.360	6 hours
Incubator-2-3	13.546	6 hours
Incubator-2-4	13.057	6 hours
Incubator-2-5	9.790	6 hours

Table 4.1: The table contains the amount of data gathered for the thesis, which have been gathered by myself in collaboration with SEGES Innovation. The images is the video data split into images.

The images have been extracted for every thirty frames, which is also the reason for the image difference from incubator one and incubator two. The images are then saved into the provided folder, where in this case every video file have been converted into images and saved in it's own folder. The chosen image format is .png and the files are named vlc-extracted and then starts from 00001. Each file is then saved according to their frame, so the image names will jump with thirty every time, since every thirtieth frame is saved as an image.

4.2 Piglet Incubator

The setup of the incubator is not a pre-setup, and it must be assembled every time it is used. The foundation of the incubator is a wooden crate, which has a heating mat on the bottom of the crate to keep a warmer climate for the piglets. This heat mat is then covered by straw, and this is to try to emulate the warmer area in the farrowing pen. The artificial teats are mounted on the side of the crate, which is connected to power. The power requirement is due to the heating element in the artificial teats, which makes them appear closer to reality. The cameras are then mounted on the top of the crate to provide a top view of the artificial teats. The cameras are setup with a LAN cable, which is needed to create a stable connection to the receiver, that is connected to the computer. The video data will then be transferred to an external hard disc. At the end, the bottles are setup on the side of the crate, and are connected to the artificial teats, so milk can be provided. Two plastic plates are then laid on top of the crate to make a warmer environment for the piglets.

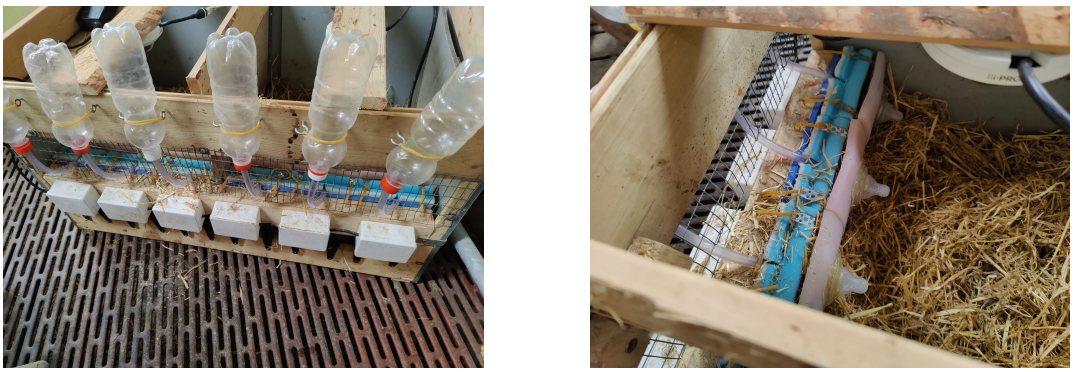

Figure 4.2: The setup of the incubator seen from the outside and the inside.

The milk used while gathering data is cow's milk. This is due to the limited milk produced by sows, read more in section 2.1. The cow's milk used is colostrum and milk produced after colostrum.

4.3 Equipment for Data Collection

The data has been collected using two cameras, where they were placed in each of the two sections of the incubator. The cameras are Panasonic WV-U2530L, which is a network camera. The camera is utilizing Intelligent Auto (iA) to adjust the camera settings to changing dynamics and motion to reduce distortion such as motion blur. The camera is also made to capture longer recordings, while taking up less storage space. The camera captures videos in full HD with 1080 pixels at 30 frames per second. The camera can work outdoors and has a protective glass around the lens to protect the lens. The camera uses super dynamic 120 dB to provide a high range capability, which helps for clear data in brighter or darker areas. To reduce the storage space, the camera uses a compression technique, which is based of self-learning region of interest (ROI) encoding. The ROI is based on movement and the camera will, based on detected movement, compress the areas of the image with little to no motion in it. This results in reduced transmitted data, while maintaining a high quality of the images [5].

Figure 4.3: The Panasonic WV-U2530L are used in both sides of the incubator and have been used to capture all the gathered visual data for this thesis [5].

4.3.1 Side View vs Top View

During the initial data gathering a side view and a top view was utilized to create comparison of the data. Based on the theory from section 2.2.2, it is established that the main sign of feeding in piglets is movement of the jaw. From the two different viewpoints, a side view can provide a better angle for viewing the jaw, but it comes with the downside of less artificial teats in focus. The top view result in a good overview of all three artificial teats, but it does not provide a good view of the jaw. The theory discovered in section 2.3 shows, that the existing IT solutions uses top view for tracking and detection of pigs and piglets. This is done to provide a good overview and with the possibility of making multiple detections with less occlusions.

Figure 4.4: The image shows an example of a top view of the artificial teats and the piglets in the incubator.

Based on the amount of collected data, and conversations with SEGES Innovation, it was decided to go with the top view. This decision was based on the visibility of the artificial teats, but it was also heavily based on the fact, that an expert would still be able to distinguish a piglet drinking with the top view. Based

on former studies and the first data collected, there was a tendency in the videos, that the piglets preferred one artificial teat over the rest, and the piglets would proceed to stand on top of each other. Based on the gathered video the top view provided a better angle, so a feeding could be seen, where the occlusion from the side view did not provide conclusive results.

Figure 4.5: An example of a side view in the incubator of the piglets and artificial teats.

4.4 The Data Visuals

The piglets in the gathered data have been marked with colored numbers, which makes them differ from one another. The piglets are a hybrid breed, where the most common pig types in Denmark have been mixed. This provides better quality pigs, but it also results in piglets and pigs, which do not have any visual distinction. The numbers can therefore provide a visual to distinguish the three piglets from each other, which can be seen in the figure below.

Figure 4.6: The image shows the quality and how the piglets are visualized in the incubator.

There are three piglets in each image, unless they are in the back of the incubator, where they cannot be seen by the camera. The piglets have a number on their

back, which differs from one another. The number are also written in different colors, so in the case of the numbers being unrecognizable, there would still be some sign in red, blue or green. The images also provide a top view of the artificial teats in the incubator, which then can be utilized during the annotation of the visual data.

The piglets used for the captured data is hybrid pigs. The piglets are a combination of Danbred Landrace, Danbred Duroc, and Danbred Yorkshire. The piglets have been bred after the most common hybrid combination, see section 2.1.2. The piglets were put into the incubator shortly after being born (within 10 min), since the piglets in former experiments have shown no interest in the artificial teats after feeding from the sow.

4.5 Summary

This chapter provided an overview of the data collected in collaboration with SEGES Innovation, a description of the setup of the incubator, a description of the type of camera, and a visual representation of what the collected data looks like. The collected data will be used for training and evaluating the system. The placement of the camera in the incubator was also discussed and a final setup was decided upon.

Chapter 5

Implementation

This chapter covers the hardware used for the training and inference of the model. The chapter will also cover the theory of the used models, and a description of the dataset created for the testing of the model.

5.1 Hardware

The hardware that will be used for training the model and testing the performance of the model has the specifications seen in the table below.

OS	Microsoft Windows 10 Home
GPU	Nvidea GeForce GTX 1060
CUDA Version	11.1
Nvidea Driver	512.95
CPU	Intel(R) Core(TM) i7-7700HQ CPU 2.80GHz
RAM	16 GB DDR4

Table 5.1: The specifications for the computer utilised for the experiment.

The GPU Geforce GTX 1060 has 6GB GDDR5 VRAM, which is insufficient for large deep learning models. An alternative could be using services like Kaggle or Google Colab, which provides the possibility of training models on GPU. Since the thesis is written in collaboration with SEGES Innovation, and have connections with a current research project, public services do not seem safe, with the risk of it becoming public.

5.2 The Models

For the implementation, multiple models have been implemented with the aim of achieving a high precision and recall value, while also being able to have a high performance. Based on the state-of-the-art computer vision models, see section 2.7, the Faster R-CNN shows great potential, and it is known to provide good results, while being able to run in real-time. Therefore multiple versions of the Faster R-CNN have been implemented. The implementations uses the torchvision.models library, and have then been modified. Four different models have been implemented: fasterrcnn_resnet50_rpn, fasterrcnn_resnet18, fasterrcnn_squeeznet1.0, and fasterrcnn_squeeznet1.1. These four versions of a Faster R-CNN all uses a different backbone, which is followed by a RPN and ROI Pooling. The following sections will go in depth with the four models.

5.2.1 Faster R-CNN ResNet-50

The Faster R-CNN implementation using the ResNet-50 as a backbone with FPN is a PyTorch model, which is accessed through the torchvision.models.detection library.

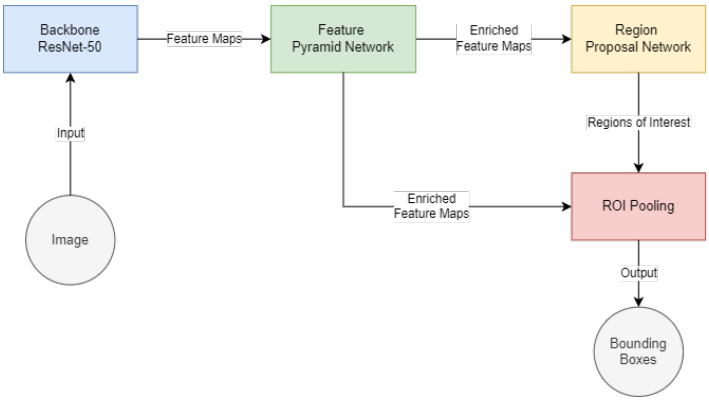

Figure 5.1: An overview of the Faster R-CNN implementation, which uses ResNet-50 and FPN as a backbone.

Backbone

As seen in figure 5.1, the backbone of the Faster R-CNN is the ResNet-50 architecture. As described in the theory in section 2.7.1, the purpose of the backbone is to take the input image or video and forward a feature map. The feature maps are created by the ResNet-50 extracting useful features from the input images related to objects.

The ResNet-50 is a CNN model, that contains 50 layers. ResNet-50 is a deep neural network, which typically result in difficulties, since it is more difficult to train large deep networks. This is due to degradation, which can happen to deep CNNs, since the accuracy levels might get saturated and degrade after a certain point.

The ResNet models avoid this issue, since they utilise a identity function, which means the higher layers of the model do not perform worse than the lower layers. This is done through residual blocks, which improves the efficiency of deep neural networks, and minimize the percentage of errors. This is also known as skip connections, where the outputs from previous layers are added to the outputs of the stacked layers.

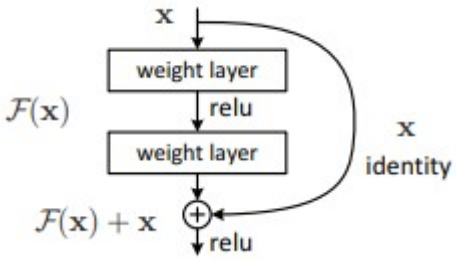

Figure 5.2: A visual of residual learning [14].

The residual learning aims to fit a desired underlying mapping. The desired underlying mapping is expressed as $H(x)$, which is then fitted with the non-linear layers.

$$F(x) := H(x) - x$$

The original mapping is then recast into $F(x) + x$. The process of conducting identity mapping do not add any extra parameters or computational complexity.

Residual learning is adopted to every few stacked layers. One of the blocks seen in fig 5.3 would be defined as:

$$y = F(x, W_i) + x$$

For this equation x and y are the input and output vectors of the layers. The F function represents the residual function. In the case seen in image 5.3, there are two layers, which would be written as:

$$F = W_2\sigma(W_1x)$$

Here σ represent the ReLU function and the biases. For this equation the input and the residual map have to contain the same dimensions. In the case of hanging the input and output channels, then linear projection W_s is used:

$$y = F(x, W_i) + W_s x$$

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		[3×3, 64; 3×3, 64] ×2	[3×3, 64; 3×3, 64] ×3	[1×1, 64; 3×3, 64; 1×1, 256] ×3	[1×1, 64; 3×3, 64; 1×1, 256] ×3	[1×1, 64; 3×3, 64; 1×1, 256] ×3
conv3_x	28×28	[3×3, 128; 3×3, 128] ×2	[3×3, 128; 3×3, 128] ×4	[1×1, 128; 3×3, 128; 1×1, 512] ×4	[1×1, 128; 3×3, 128; 1×1, 512] ×4	[1×1, 128; 3×3, 128; 1×1, 512] ×8
conv4_x	14×14	[3×3, 256; 3×3, 256] ×2	[3×3, 256; 3×3, 256] ×6	[1×1, 256; 3×3, 256; 1×1, 1024] ×6	[1×1, 256; 3×3, 256; 1×1, 1024] ×23	[1×1, 256; 3×3, 256; 1×1, 1024] ×36
conv5_x	7×7	[3×3, 512; 3×3, 512] ×2	[3×3, 512; 3×3, 512] ×3	[1×1, 512; 3×3, 512; 1×1, 2048] ×3	[1×1, 512; 3×3, 512; 1×1, 2048] ×3	[1×1, 512; 3×3, 512; 1×1, 2048] ×3
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

Figure 5.3: An overview of the ResNet architecture. [20].

The information related to residual learning is written for the ResNet-34 from 2015, since the residual skip have changed from two to three layers for ResNet-50 and other ResNet architecture with more layers [14].

Feature Pyramid Network

The Resnet-50 produces the feature map, which is then passed on to the Feature Pyramid Network (FPN). The purpose of the FPN is to do refinements of the feature maps from the ResNet-50 and then forward it to the RPN and the ROI. The FPN enhances features in the feature map, so small features are easier to recognize. This is done through creating multiple feature map layers.

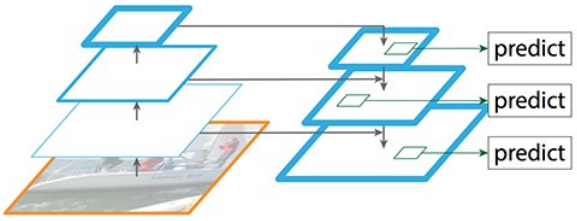

Figure 5.4: A visual representation of the feature maps through the bottom-up and top-down pathway [15].

FPN uses bottom-up and top-down pathway to refine feature maps. The bottom-up method is utilised by general CNNs for feature extraction. This is the process of reducing the image size and keeping key features, which may result in losing smaller features. This process is described in section 2.5.1. FPN then uses the spatially reduced feature map and constructs higher resolution layers based on the feature map. Due to the downsampling and upsampling, the locations of the objects are not precise, and therefore lateral connections are used between the reconstructed layers and the corresponding feature maps. This method resembles the skip connections from a ResNet model.

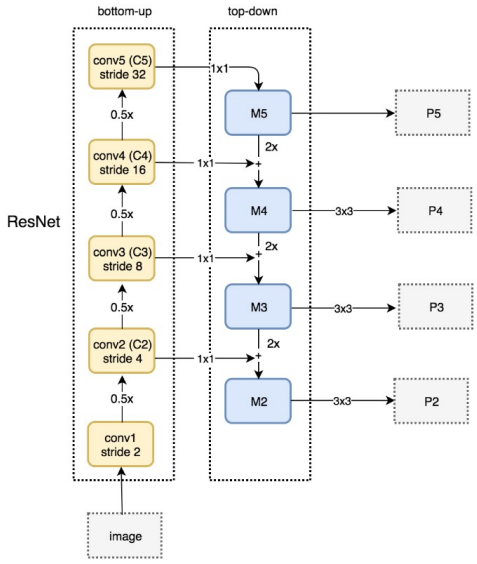

Figure 5.5: An overview of the process of bottom-up and top-down pathway [15].

The last convolutional layer is applied with a 1x1 convolutional filter, which is used to create the first top-down feature map. Each feature map are then up-scaled by 2, since the feature maps are reduced to half the size during the bottom-up pathway. This happens since the stride is set to 2. When up-scaling the feature maps by 2, a 1x1 convolutional filter is applied to the up-scaled feature map, which is done to reduce the depth of the feature map. This process is continued except for the first convolutional layer in the bottom-up pathway, since the spatial dimension of C1 is too large [15].

RPN

The refined feature maps are then passed to the Regional Proposal Network (RPN), that is a feature extractor and to the Region of Interest (ROI) Pooling. The RPN

then applies a sliding window over the feature maps to make predictions about the possibility of there being an object and object boundary boxes. This process is explained in section 2.7.1.

When used in combination with a FPN, a 3x3 convolution filter is applied over each scaled feature map. This process is then followed by a 1x1 convolutional filter, which is used to make predictions in regards to objects and boundary box regression. This is referred to as the RPN head, and the process is applied to all scaled feature maps from the FPN [15].

ROI pooling

After the RPN, the proposed regions are combined with the feature maps from the FPN. Multiple feature maps have been created by the FPN, and based on the width and height of the ROI from the RPN, a feature map from the FPN is selected.

$$k = \lfloor k_0 + log_2(\sqrt{wh}/224) \rfloor$$

In this function k_0 is equal to the selected feature map. k is the P_k layer in the FPN, which is used to generate the feature patch. The ROI pooling produces a fixed size feature maps of the proposed regions made by the RPN. This fixed size output is achieved by using max pooling [15]. This is described in section 2.5.1.

As described in section 2.7.1, the fixed size ROI feature maps are forwarded to the predictor, which consist of fully connected convolutional layers, which is used to predict the bounding boxes of detected objects, and a softmax layer to predict the score of the classes.

Softmax is a function to turn a vector of K real values into a vector of K real values, that sums to 1. The softmax values transforms values of positive and negative values to be a value between 0 and 1.

$$\sigma(\vec{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$$

Figure 5.6: The function of Softmax [39].

In the function above, the $\vec{z}$ is the input vector to the softmax function. z_i is all the elements of the input vector, which can be real values, positive values, negative values or zero. e^{z_i} is the standard exponential function, which is applied to every element of the input vector, which makes all parameters of the input vector positive. The term in the bottom of the formula is the normalization term. The normalization ensures that all the output values sum to 1 and are in the range

between 0 and 1. The K variable used in the normalization term is the number of classes in a multi-class classifier [39].

5.2.2 Faster R-CNN ResNet-18

The Faster R-CNN using the ResNet-18 as a backbone is a classic implementation of the Faster R-CNN, since it uses a CNN backbone followed by a RPN, and the ROI Pooling.

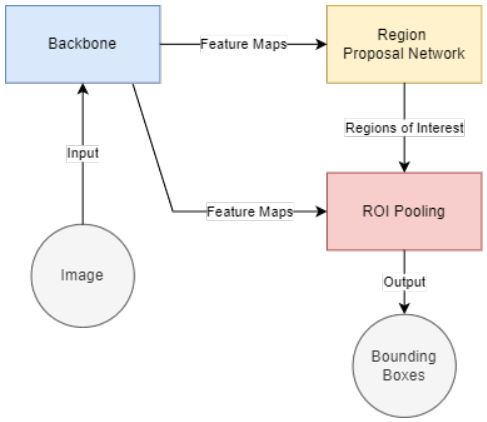

Figure 5.7: An overview of the classic Faster R-CNN using a CNN as a backbone, RPN and ROI Pooling. This architecture is used for the following three model implementations.

This model follows the general Faster R-CNN architecture without the FPN. This means, that the feature map made by the backbone is forwarded to the RPN and the ROI Pooling layer. The RPN will then based on the feature map create ROIs, which will be extracted and forwarded to the ROI Pooling layer, and be given a fixed size, which is then used to predict the boundary boxes of detected objects and the classes score. This is explained in the previous section 5.2.1.

The ResNet-18 architecture is also very similar to the ResNet-50, but it is a more simple architecture with fewer layers. As established in the section above, the ResNet-18 is only using two layers for the residual skip, where the ResNet-50 uses three layers.

5.2.3 Faster R-CNN SqueezeNet1.0

The Faster R-CNN using the SqueezeNet1.0 as a backbone is a similar implementation mentioned above. The difference between the two models are the backbone, where this version uses the SqueezeNet1.0 architecture.

The architecture of the model can be seen in figure 5.7, and it follows the same procedure as the Faster R-CNN using the ResNet-18 as backbone.

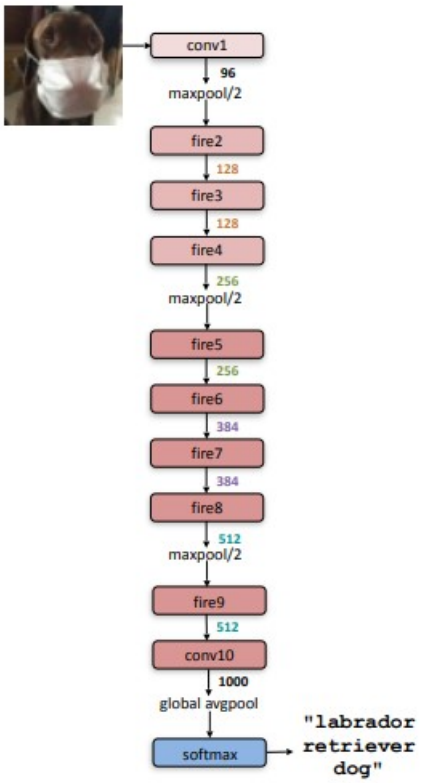

Figure 5.8: An overview of the SqueezeNet architecture [16].

SqueezeNet is an architecture, which strives towards preserving competitive accuracy with fewer parameters. This is done by using three different strategies.

The first strategy is to replace 3x3 filters with 1x1 filters, which reduces the amount of parameters by 9 times for every 3x3 filter.

The second strategy is to reduce the input channels to 3x3 filters, which is done using squeeze layers. A squeeze layer is a convolutional layer, which only has 1x1 filters.

The third strategy is to downsample late in the network, so the convolution layers have large activation maps. The most common strategy is to use a stride above one, but the strategy with the SqueezeNet is to use a stride of one, and the layers

having a stride above one is centralised around the last layers, then the majority of the activation maps will be large. This will provide a higher classification accuracy even though less parameters are available.

The majority of the architecture of the SqueezeNet consists of Fire modules. A Fire module consists of a squeeze convolution layer, which feeds into an expanding layer consisting of 1x1 and 3x3 convolutional filters. A Fire module consists of three tune-able dimensions. s_{1x1} is the number of filters in the squeeze layer, e_{1x1} is the number of 1x1 filters in the expanding layers, and the e_{3x3} is the number of 3x3 filters in the expanding layers.

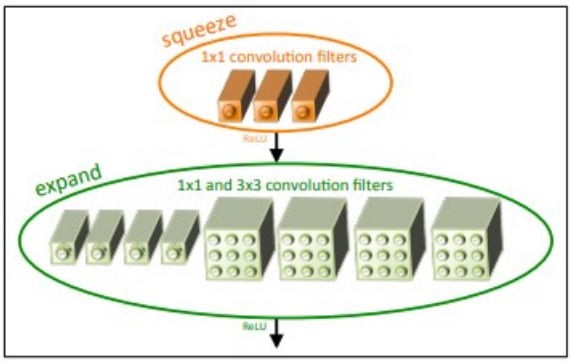

Figure 5.9: A visual representation of the Fire module using a squeeze convolutional layer and feeding into an expanding layer [16].

The Fire modules set the s_{1x1} to be less than $e_{1x1} + e_{3x3}$, which makes the squeeze layers keep the number of input channels low. The SqueezeNet uses the ReLU activation function on squeeze and expand layers.

The SqueezeNet architecture performs as well as the AlexNet, but with a 50 times reduction in the model size [16].

5.2.4 Faster R-CNN SqueezeNet1.1

The Faster R-CNN using the SqueezeNet1.1 as a backbone is very similar to the SqueezeNet1.0 implementation above. This Faster R-CNN implementation follows the architecture seen in figure 5.7.

The SqueezeNet1.1 is an optimized version of the SqueezeNet1.0, where the conv1 layer have gone from 96 filters of resolution 7x7, to now having 64 filters of resolution 3x3. The pooling layers have also been reduced. This has reduced the

computation by 2.4 times, while still achieving the same top-5 accuracy [34].

There is no other differences between the SqueezeNet1.0 and SqueezeNet1.1 besides the few optimization changes. The result of the Faster R-CNN implementation follows the Faster R-CNN using ResNet-18 as a backbone in section 5.2.2.

5.3 The Custom Dataset

This section will go through the created dataset for the training and testing of the four formerly mentioned models. The section will describe the annotation tool used to conduct image annotations. The dataset will then be presented with annotation examples from the dataset. At the end of the section some discoveries found during the annotation process will be presented, which has a major influence on the training and test results of the models.

5.3.1 Annotation Tool

For the annotation of the data gathered in section 4 the open-source image and video annotation tool Computer Vision Annotation Tool (CVAT) has been used. CVAT has been developed by Intel and it can be accessed online or run on your local machine.

When creating a project in CVAT, only the name and the labels are decided. When creating tasks, the images or video are uploaded to the task, where additional labels can be added. In advanced configurations it is possible to decide the sorting method, image quality, format, etc.

Figure 5.10: An example of bounding box annotations using CVAT.

The annotation process can then begin, where a variety of possible annotation

types can be utilised. These types are described in section 2.4.5.

After the annotation process the dataset can be extracted in multiple formats like COCO, Pascal VOC, YOLO, and more [32].

For this thesis, the annotations have been exported in the Pascal VOC format. The Pascal VOC format consist of XML files, which contains information about filename of the image containing the annotations, and a path for the image. The XML file then describes the dimensions of the image, and proceeds to describe the annotation. The annotation part of the XML file has the name of the object, pose, truncated, difficult, and then the bounding box coordinates. The dataset used for this thesis has pose set to "Unspecified", and truncated and difficult set to zero.

Truncated means the bounding box specified for the object does not contain the entire object. This can be used in case of occlusions, where the value would be set to one.

The variable difficult refers to an object being perceived as difficult to recognize, and in those cases the value would be set to one [21].

5.3.2 Annotations

For the annotations, the boundary box method was used, which was positioned based on two points, with smaller adjustments afterwards. As seen in the table below, the amount of NotEating annotations have more that twice as many annotations as Eating. This is due to the piglets moving around causing occlusions of the artificial teats, which can be seen in fig 5.11.

While diving more into the data while annotating, it was discovered that 9 out of the 24 piglets did not feed at all. This removes 37.5% of the piglets from being annotated as eating. The dataset also contains images of the piglets sleeping, which also provides no annotations.

Incubator	Images	Total Annotations	Eating	NotEating	Blue	Red	Green
Incubator-1-1	19.558	3008	719	1643	439	241	2
Incubator-1-2	19.569	1387	460	475	60	95	297
Incubator-1-3	19.531	167	12	143	0	6	6
Incubator-1-4	19.000	0	0	0	0	0	0
Incubator-1-5	19.226	0	0	0	0	0	0
Incubator-2-1	13.731	283	0	283	0	0	0
Incubator-2-2	13.360	1405	249	912	195	0	49
Incubator-2-3	13.546	674	212	251	78	0	133
Incubator-2-4	13.057	0	0	0	0	0	0
Incubator-2-5	9.790	0	0	0	0	0	0
All incubators	150.578	6924	1652	3707	772	342	487

Table 5.2: The table above contains the number of annotated images from each incubator recording. The table also contains the number of annotation for the classes Eating, NotEating, Blue, Red, and Green.

During the task of annotating, it was discovered that multiple of the piglets in the incubator uses the artificial teats to seek comfort. An example of this can be seen in fig 5.12. The piglets seeking comfort followed two different patterns, where one case was easily noticeable, but the other case proved to be difficult.

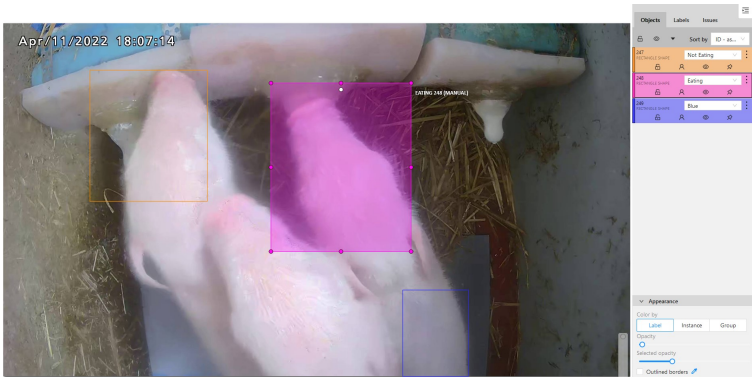

Figure 5.11: An example of a piglet eating, while another piglet is causing an occusion of one of the artificial teats.

The easy to notice comfort seeking pattern involved a piglet, which had eaten using the artificial teats. The piglet was sitting in front of one of the artificial teats, and then kept the teat in its mouth, while it was falling asleep. The teat was slowly falling out of the piglet's mouth, which eventually slightly woke up the piglet,

which then attached itself to the artificial teat and fell asleep. This pattern keeps repeating for the period where the piglet is sleeping. This behavior results in many NotEating annotations, since it can look like a piglet is feeding.

The more difficult case of piglets seeking comfort is also a piglet sitting with the artificial teat in its mouth, but in this case the teat stays in the piglets mouth. This results in a case, where it is extremely difficult to determine with a top view camera, if a piglet is eating or sleeping, since the eyes cannot be seen in certain positions in the incubator.

The total amount of piglets seeking comfort from the artificial teats is 9 out of the 24 piglets, which is 37.5% of the data, which is being labeled as NotEating or human error resulting in cases being labeled as Eating, when it should have been labeled as NotEating, or the other way around.

Out of the 24 piglets only 6 are feeding without seeking comfort, which is 25% of the subjects used to gather data for the dataset.

Figure 5.12: An example of a piglet eating, and another piglet using the artificial teat to seek comfort.

Chapter 6

Evaluation of Models

This chapter will go through the evaluation of the four versions of the Faster R-CNN. The models will be evaluated based on Precision and Recall from the training of the models. Each of the models have also been tested on images from the incubator, where a speed performance evaluation will be conducted in the form of Frames Per Second (fps).

6.1 Precision and Recall

The four variations of a Faster R-CNN have been trained and tested using the custom dataset, see section 5.3, where they have been training for 50 epochs, with images of size (300, 300), and a batch size of 4.

For the evaluation Intersection over Union (IoU) has been used to evaluate the boundary boxes. The method describes the extend of the overlap of the proposed boundary boxes and the groundtruth bounding boxes.

$$IoU = \frac{Area of Intersection of two boxes}{Area of Union of two boxes}$$

The intersection of the two boxes is computed by comparing the coordinates of the corners of the two boxes. The Union of the two boxes is found by adding the area of the two boxes together, and then subtracting the area of intersection. As mentioned in the equation above, the IoU can be calculated by dividing the area of intersection with the area of union for detected bounding boxes and the groundtruth bounding boxes [36].

The IoU is then used with Precision and Recall to evaluate the detection of the class Eating. Precision measures the accuracy of the models predictions, which is

done using True Positive (TP) and False Positive (FP). A TP is a correct prediction, and a FP is an incorrect detection. The Precision is then found with the equation below.

$$Precision = \frac{TP}{TP + FP}$$

A perfect Prediction is equal to one. When calculating the Prediction for object detection models, the FP and TP are based on the IoU, and the set threshold. For the results of this thesis, the threshold of 0.5 and 0.75 will be used. This means, that if the IoU is below 0.5 or 0.75, it will be determined to be a FP, and if the IoU is above 0.5 or 0.75 it is a TP.

The other part of the evaluation is the Recall, which is a measure of how well the positives are detected. To calculate the Recall value, the TPs is used with False Negatives (FN), which is non-predicted objects. This is calculated with the equation below.

$$Recall = \frac{TP}{TP + FN}$$

The Average Precision and Recall for the class Eating will be calculated to evaluate the four models [40].

Model	Average Precision @50	Average Precision @75	Average Recall
ResNet-18	0.469	0.395	0.363
ResNet-50	0.466	0.387	0.364
SqueezeNet1.0	0.453	0371	0.359
SqueezeNet1.1	0.456	0.393	0.364

Table 6.1: The test result of the four different models.

The results of the training and testing of the four models can be seen in the table above. The Faster R-CNN using ResNet-18 as a backbone achieves the highest Average Precision for both thresholds. The highest Average Recall is achieved by the Faster R-CNN models using ResNet-50 and FPN as a backbone and SqueezeNet1.1 as a backbone.

In general, the models using ResNet-18 and ResNet-50 produces almost the same results with the exception of the Average Precision with a threshold of 0.75. For the two SqueezeNet models, there is a better performance with the optimized SqueezeNet1.1, which is seen in Average Precision with a threshold of 0.75 and the

Average Recall.

The model that showed the most potential was trained and tested using the classes Red, Green, and Blue.

Model	Average Precision @50	Average Precision @75	Average Recall
ResNet-18	0.578	0.274	0.375

Table 6.2: The test result of the four different models.

As seen in table 6.2, the achieved Average Precision between the two thresholds of 0.5 and 0.75 are halved. This model was also trained with the same configurations as the models trained and tested on the dataset using the class Eating, while using NotEating ad background.

Examples of detections performed by the four different models, and plots of their training loss over number of iterations can be found in the appendix A.

6.2 Speed Performance

The trained models seen in table 6.1 have been tested on new data to evaluate their performance in form of frames per second (fps), since the model should be able to conduct defections in real-time. The test uses 464 images containing cases of eating, and the average fps is calculated by taking the average amount of images processed per second.

Model	Average fps
ResNet-18	21.097
ResNet-50	6.361
SqueezeNet1.0	23.326
SqueezeNet1.1	29.737

Table 6.3: The performance results of the four different models with their average frames per second.

As seen in table 6.3, the fastest model is SqueezeNet1.1, and the slowest model is the ResNet-50. There is a 23,376 frames processed per second difference between the two models.

Chapter 7

Discussion

This chapter will summarise the success criteria and discuss the evaluation of the models, and the decisions taken throughout the process.

7.1 Success Criteria

In this section, the results from the evaluation of the four models, which can be seen in section 6.1 and section 6.2, will be compared to the success criteria, which can be found in section 3.1. The purpose of this comparison is to discuss the quality of the solution compared to previously established requirements.

SC1 During inference, the model should be able to process at least 5 images per second. - This success criterion was achieved, since the slowest model still processed more than five images per second.

SC2 The model should achieve an Average Precision above 75%. - This success criterion was not achieved, since the Average Precision values with a threshold of 0.75 did not surpass 0.50, see table 6.1.

SC3 The model should detect the three different piglets with a Average Precision above 75%. - This success criterion was not achieved, since the model used did not surpass 0.3 with a threshold of 0.75, see table 6.2.

SC4 The model should output detected feedings and label a feeding to the correct piglet. - This success criterion was not achieved, since detections with a high Average Precision was not achieved, and therefore not attempted.

7.2 Discussion of the Results

One of the main identified problems is in regards to the dataset. During the annotation process, it was discovered that 9 out of the 24 piglets were seeking the artificial teats for comfort, see section 5.3. This has resulted in multiple problems in regards to annotating the data. This has resulted in less usable samples, and probably wrong annotations during the annotation process.

Due to the piglets seeking comfort, multiple images have been annotated as a piglet not eating, even though the piglet has an artificial teat in it's mouth. This can have been part of the reason for the poor results of the models, where only 38% of the cases using a threshold of 0.7 have been correct predictions. SEGES Innovation knew about piglets seeking comfort using the teats, but not to this extend. This knowledge was not shared with me before after the data was collected.

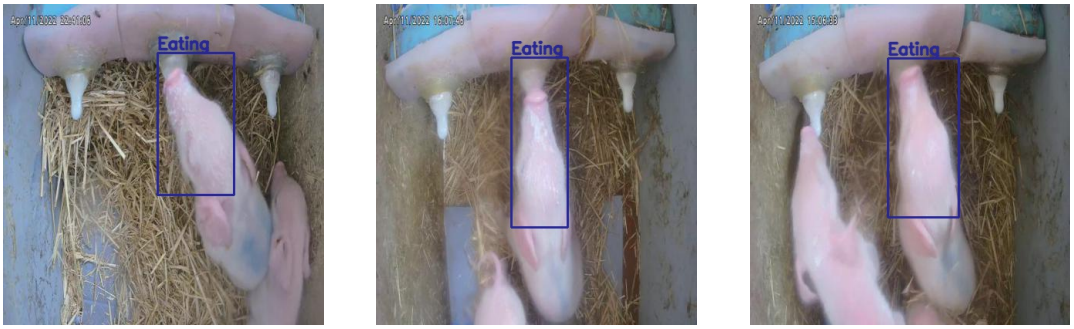

Figure 7.1: An example of False Positive detections made during the speed performance evaluation. These detections are probably due to wrongfully annotations.

The problem of the piglets seeking comfort might be solvable with another setup for the data gathering. For the gathering of the data, see section 4, a top view camera has been used. This resulted in more data, since all three artificial teats were visible at all times. Since the piglets do not provide any signs of eating besides movement of the jaw, it can be difficult to distinguish a feeding pig from one seeking comfort, with a top view. The alternative is the side view, that could provide a clear view of the jaw of the piglets, which could provide more accurate annotations. The side view would not provide a clear view of all three artificial teats, but more precise annotations could be used. Based on the placement of the piglets in the incubator occlusion could happen more regularly in the gathered data, and only usage of one artificial teat at the time might be captured. This could be the case with a side view, since the piglet eating might make it impossible to see the two other teats.

The detection of the colors on the back of the piglets did not achieve great results, which is due to multiple factors. One of the reasons is the dataset itself. As seen in table 5.2 only 1601 annotations were made of the piglets colored numbers. The main focus of this thesis has been to detect a piglet eating with high precision, and therefore the piglets colors were only annotated in the cases of a piglet feeding. This has resulted in a very limited dataset to detect the color on the back of the piglets. To achieve a higher precision more images have to be annotated of the piglets colors on their back.

Another issue with the piglets colors on the back were, that they would disappear over time. The piglets are playing, sleeping, massaging the artificial teats, eating, etc. in the incubator, which over time result in the colors disappearing. During the data gathering, a SEGES Innovation employee stated, that multicolored ear tags could also have been used instead of the colored numbers on their back. Using ear tags in different colors would solve the issue of the colors being removed over time. This would cause potential occlusions of the ear tags at times, but since the ears are laying back for the majority of the piglets the entire duration of the incubator time, it would be limited. This approach would also make the task simplified. The current annotations of a feeding piglet contains it's head. The ear tag would be within the bounding box, which could then be forwarded to another model that performs classification on the bounding box image, and determine the specific piglet based on the color of the ear tag. The model would then be able to provide information like drinking duration, and number of feeding times for each individual piglet.

The selection of the architectures provided multiple models, see chapter 5, which all performed poorly. The Faster R-CNN using ResNet-50 and FPN was selected, since it provided a deeper neural network and the Feature Pyramid Network, since the refined feature maps might provide better results. This was not the case, which is mainly due to the dataset. The addition of the FPN might also have been unnecessary, since the detected objects are of a larger size, and therefore refined feature maps might not be necessary. The ResNet-18 version was to include a Faster R-CNN without the FPN, and to use a less deep neural network using fewer parameters. The same was the case for the two SqueezeNet implementations. The SqueezeNet versions were an attempt to achieve a higher precision using fewer parameters, and with high speed performance. This can also be seen in table 6.3. All of the architectures failed to provide a high precision or recall, but they all proved to be usable for processing in real-time, since all models were able to process more than five images per second.

Chapter 8

Conclusion

The intention of this thesis was to create an object detection model that would be able to detect piglets eating using artificial teats, and determine the individual piglet drinking. It was attempted to achieve this goal by using variations of a Faster R-CNN.

Based on the analysis, see chapter 2, one of the main reasons of piglets dying is due to not eating colostrum provided by the sow. The incubator developed by SEGES Innovation strives to solve this issue, but the current setup requires many staff hours. Based on this the problem statement, see section 3, and the success criteria, see section 3.1, the models was aiming towards solving these issues.

In the evaluation of the models the IoU method was used with Precision and Recall to evaluate the detections of the models, see section 6. All the models were tested on the same train and test data, which was the custom dataset, see section 5.3. Following the training and testing, the models performance were evaluated, since the models should be able to process in real-time, which was done using fps as a measure.

The models did not fulfill the success criteria for the thesis. The first success criteria was fulfilled, since all models speed performance were above five frames per second, see section 6.2. The remaining success criterias were not fulfilled, since the detections of eating did not provide a high precision, see section 6.1.

The main reason for the poor precision results are the custom dataset, since it only provides limited annotations, and annotations might be wrongly annotated, see section 5.3. None of the trained models were able to achieve an Average Precision above 0.40 using a threshold of 0.7.

Based on the decisions made in regards to the data gathering, see section 4, it was not possible to fulfill the success criterias, see chapter 7. To achieve the success criterias it would be necessary to start over with the data gathering, where another camera position has to be considered. The new annotated dataset could then be created, and the Faster R-CNN architectures could potentially fulfill the success criterias, and further development of assigning a piglet to a detected eating could be performed.

Bibliography

[1] Shilpa Ananth. "Faster R-CNN for object detection". In: (2019). URL: https://towardsdatascience.com/faster-r-cnn-for-object-detection-a-technical-summary-474c5b857b46.

[2] Hmrishav Bandyopadhyay. "Computer Vision: Everything You Need to Know". In: (2022). URL: https://www.v7labs.com/blog/what-is-computer-vision.

[3] Gaudenz Boesch. "A Complete Guide to Image Classification in 2022". In: (). URL: https://viso.ai/computer-vision/image-classification/.

[4] Jason Brownlee. "A Gentle Introduction to Object Recognition With Deep Learning". In: (2019). URL: https://machinelearningmastery.com/object-recognition-with-deep-learning/.

[5] Panasonic BUSINESS. "iA(Intelligent Auto) H.265 Network Camera". In: (). URL: http://www.tos.by/pdf_files/cctv_panasonic/wv-u2530l.pdf.

[6] DeepAI. "What is ReLu?" In: (). URL: https://deepai.org/machine-learning-glossary-and-terms/relu.

[7] Agriculture & Food.co.uk. "Animal welfare". In: (). URL: https://agricultureandfood.co.uk/pig-production/primary-production/animal-welfare.

[8] Landbrug & Fødevarer. "DET DANSKE SVINEAVLSPROGRAM". In: (2022). URL: https://svineproduktion.dk/viden/om-grisen/avl-og-reproduktion/avl/det-danske-svineavlsprogram.

[9] Landbrug & Fødevarer. "Produktion af grise i Danmark". In: (2021). URL: https://lf.dk/viden-om/landbrugsproduktion/husdyr/svin.

[10] Landbrug & Fødevarer. "STATISTIK 2020 Grisekød". In: (2021). URL: https://lf.dk/tal-og-analyser/statistik/svin/statistik-svin/statistik-gris-2020.

[11] Landbrug & Fødevarer. "Udvikling i den kvartalsvise produktion af svin i Danmark samt prognose for de kommende kvartaler". In: (2021). URL: https://lf.dk/~/media/lf/tal-og-analyser/statistikker/aktuelle-svin/kvartalsvis-produktion/2022/kvartalvis-produktion-februar-2023.ashx.

[12] Haiming Gan et al. "Automated piglet tracking using a single convolutional neural network". In: *Biosystems Engineering* 205 (2021), pp. 48–63. ISSN: 1537-5110. DOI: https://doi.org/10.1016/j.biosystemseng.2021.02.010. URL: https://www.sciencedirect.com/science/article/pii/S1537511021000465.

[13] Janne Hansen. "Monitoring pigs to imporve welfare and production". In: (2011). URL: https://www.pigprogress.net/home/monitoring-pigs-to-improve-welfare-and-production/.

[14] Kaiming He et al. "Deep Residual Learning for Image Recognition". In: *CoRR* abs/1512.03385 (2015). arXiv: 1512.03385. URL: http://arxiv.org/abs/1512.03385.

[15] Jonathan Hui. "Understanding Feature Pyramid Networks for object detection (FPN)". In: (). URL: https://jonathan-hui.medium.com/understanding-feature-pyramid-networks-for-object-detection-fpn-45b227b9106c.

[16] Forrest N. Iandola et al. "SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <1MB model size". In: *CoRR* abs/1602.07360 (2016). arXiv: 1602.07360. URL: http://arxiv.org/abs/1602.07360.

[17] SEGES Innovation. "Pattegrise dier nu på livet løs på kunstige kirtler". In: (2022). URL: https://www.seges.tv/video/73946347/pattegrise-dier-nu-pa-livet-los-pa.

[18] Ditte Birkebæk Jensen. "26.500 grise dør hver dag, og landbruget aner ikke hvorfor". In: (2021). URL: https://avisendanmark.dk/artikel/26-500-grise-d%C3%B8r-hver-dag-og-landbruget-aner-ikke-hvorfor.

[19] Daniel Johnson. "Back Propagation Neual Network: What is Backpropagation Algorithm in Machine Learning?" In: (2022). URL: https://www.guru99.com/backpropogation-neural-network.html#3.

[20] Aakash Kaushik. "Understanding ResNet50 architecture". In: (). URL: https://iq.opengenus.org/resnet50-architecture/.

[21] Renu Khandelwal. "COCO and Pascal VOC data format for Object detection". In: (2019). URL: https://towardsdatascience.com/coco-data-format-for-object-detection-a4c5eaf518c5.

[22] Steven Mcorist. "Major problems in piglet health and management". In: (2014). URL: https://www.pigprogress.net/home/major-problems-in-piglet-health-and-management/.

[23] Mayank Mishra. "Convolutional Neural Networks, Explained". In: (2020). URL: https://towardsdatascience.com/convolutional-neural-networks-explained-9cc5188c4939.

[24] Marko Ocepek et al. "DigiPig: First Developments of an Automated Monitoring System for Body, Head and Tail Detection in Intensive Pig Farming". In: *Agriculture* 12.1 (2022). ISSN: 2077-0472. DOI: 10.3390/agriculture12010002. URL: https://www.mdpi.com/2077-0472/12/1/2.

[25] Keiron O'Shea and Ryan Nash. "An Introduction to Convolutional Neural Networks". In: *CoRR* abs/1511.08458 (2015). arXiv: 1511.08458. URL: http://arxiv.org/abs/1511.08458.

[26] Santosh Pandey et al. "Behavioral Monitoring Tool for Pig Farmers: Ear Tag Sensors, Machine Intelligence, and Technology Adoption Roadmap". In: *Animals* 11.9 (2021). ISSN: 2076-2615. DOI: 10.3390/ani11092665. URL: https://www.mdpi.com/2076-2615/11/9/2665.

[27] M.L. Pedersen et al. "Improved udder access prolongs duration of milk letdown and increases piglet weight gain". In: *Livestock Science* 140.1 (2011), pp. 253–261. ISSN: 1871-1413. DOI: https://doi.org/10.1016/j.livsci.2011.04.001. URL: https://www.sciencedirect.com/science/article/pii/S1871141311001429.

[28] Sabina Pokhrel. "Image Data Labelling and Annotation - Everything you need to know". In: (2020). URL: https://towardsdatascience.com/image-data-labelling-and-annotation-everything-you-need-to-know-86ede6c684b1.

[29] Anima Pramanik et al. "Granulated RCNN and Multi-Class Deep SORT for Multi-Object Detection and Tracking". In: *IEEE Transactions on Emerging Topics in Computational Intelligence* PP (Jan. 2021), pp. 1–11. DOI: 10.1109/TETCI.2020.3041019.

[30] PyLessons. "YOLOv3 theory explained". In: (2019). URL: https://pylessons.com/YOLOv3-introduction.

[31] Shaoqing Ren et al. "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks". In: *CoRR* abs/1506.01497 (2015). arXiv: 1506.01497. URL: http://arxiv.org/abs/1506.01497.

[32] Alberto Rizzoli. "The Complete Guide to CVAT - Pros & Cons [2022]". In: (2022). URL: https://www.v7labs.com/blog/cvat-guide.

[33] Ankit Sachan. "Zero to Hero: A Quick Guide to Object Tracking: MDNET, GOTURN, ROLO". In: (). URL: https://cv-tricks.com/object-tracking/quick-guide-mdnet-goturn-rolo/.

[34] "SqueezeNet Github library". In: (). URL: https://github.com/forresti/SqueezeNet/tree/master/SqueezeNet_v1.1.

[35] Danmarks Statistik. "Større svinebestand". In: (2021). URL: https://www.dst.dk/da/Statistik/nyheder-analyser-publ/nyt/NytHtml?cid=31810.

[36] Vineeth S Subramanyam. "IOU (Intersection over Union)". In: (2021). URL: https://medium.com/analytics-vidhya/iou-intersection-over-union-705a39e7acef.

[37] Mrinal Tyagi. "Image Segmentation: Part 1". In: (2021). URL: https://towardsdatascience.com/image-segmentation-part-1-9f3db1ac1c50.

[38] Diego Unzueta. "Convolutional Layers vs Fully Connected Layers". In: (2021). URL: https://towardsdatascience.com/convolutional-layers-vs-fully-connected-layers-364f05ab460b.

[39] Thomas Wood. "What is the Softmax Function?" In: (). URL: https://deepai.org/machine-learning-glossary-and-terms/softmax-layer.

[40] Shivy Yohanandan. "mAP (mean Average Precision) might confuse you!" In: (2020). URL: https://towardsdatascience.com/map-mean-average-precision-might-confuse-you-5956f1bfa9e2.

Appendix A

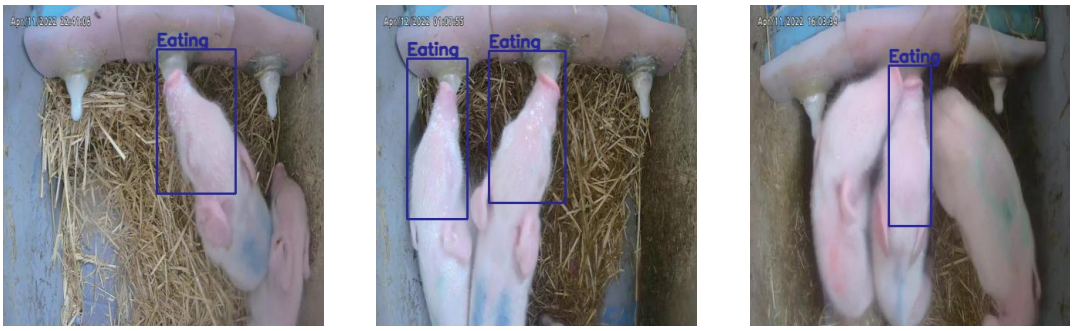

Figure A.1: Examples of detections using the Faster R-CNN with the ResNet-18 as a backbone.

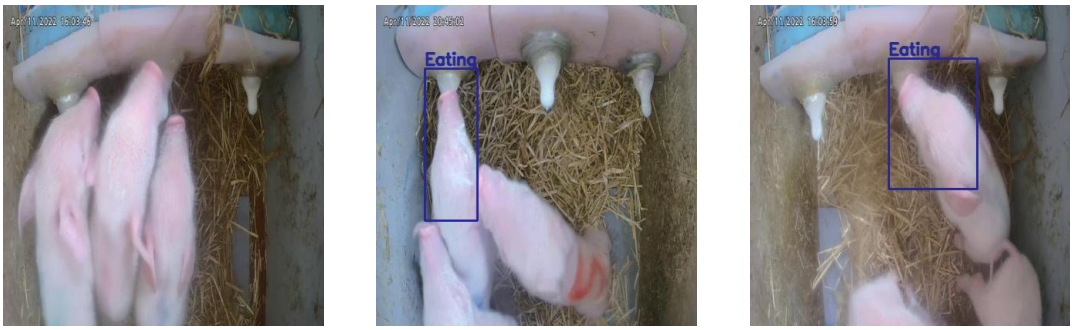

Figure A.2: Examples of detections using the Faster R-CNN with the ResNet-18 as a backbone.

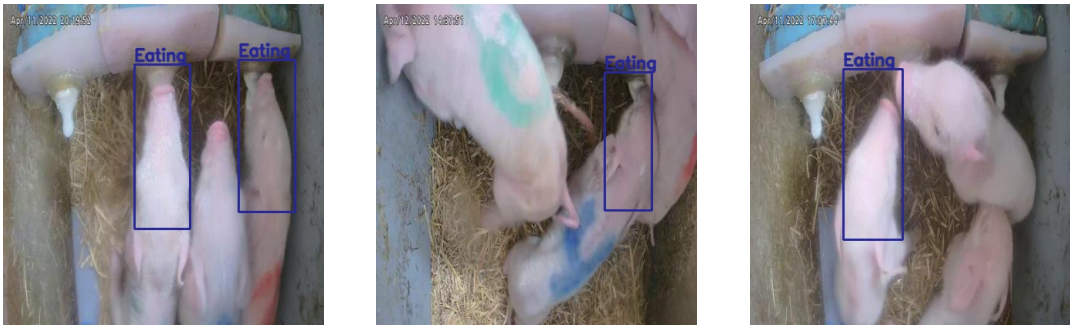

Figure A.3: Examples of detections using the Faster R-CNN with the ResNet-18 as a backbone.

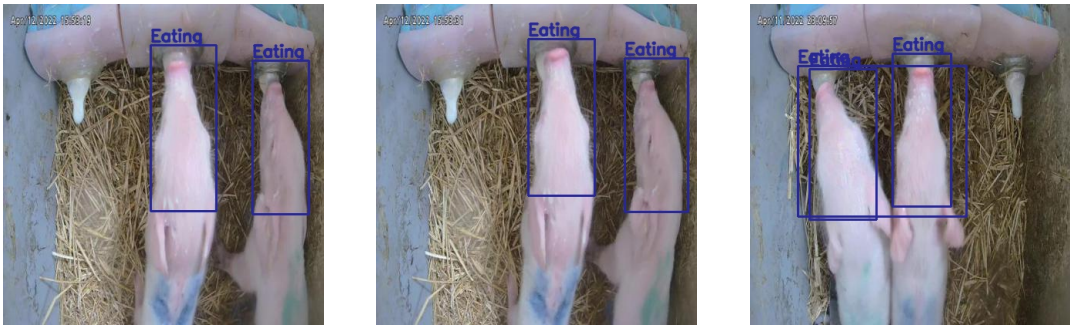

Figure A.4: Examples of detections using the Faster R-CNN with the ResNet-18 as a backbone.

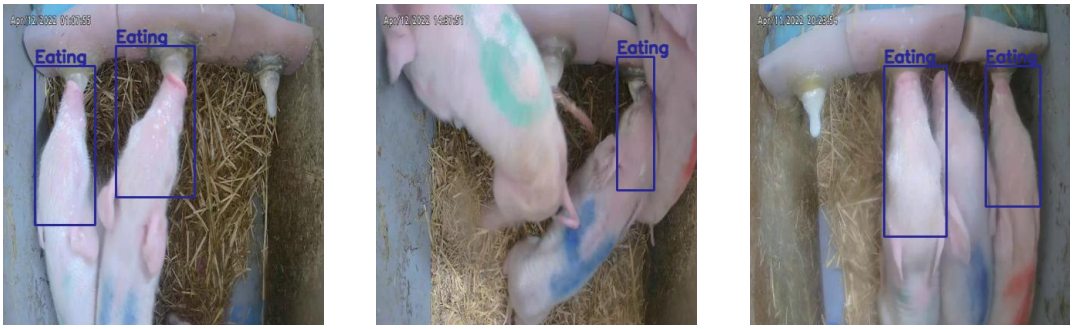

Figure A.5: Examples of detections using the Faster R-CNN with the ResNet-50 + FPN as a backbone.

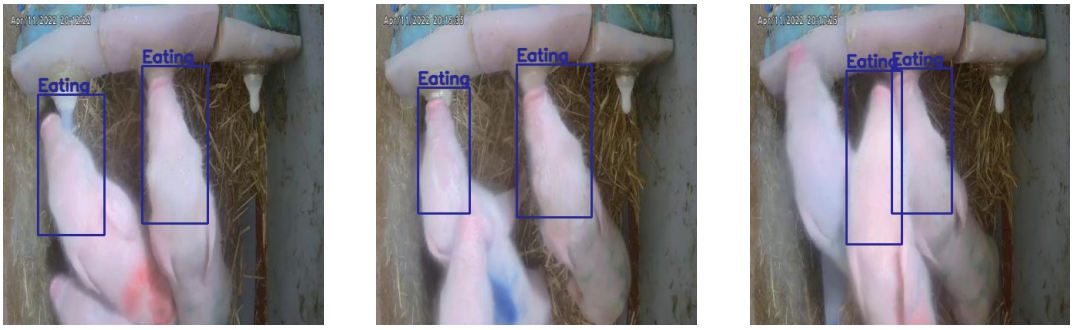

Figure A.6: Examples of detections using the Faster R-CNN with the ResNet-50 + FPN as a backbone.

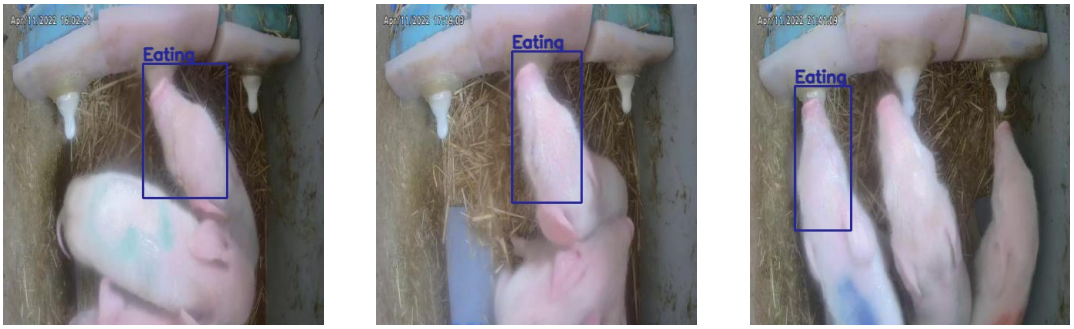

Figure A.7: Examples of detections using the Faster R-CNN with the ResNet-50 + FPN as a backbone.

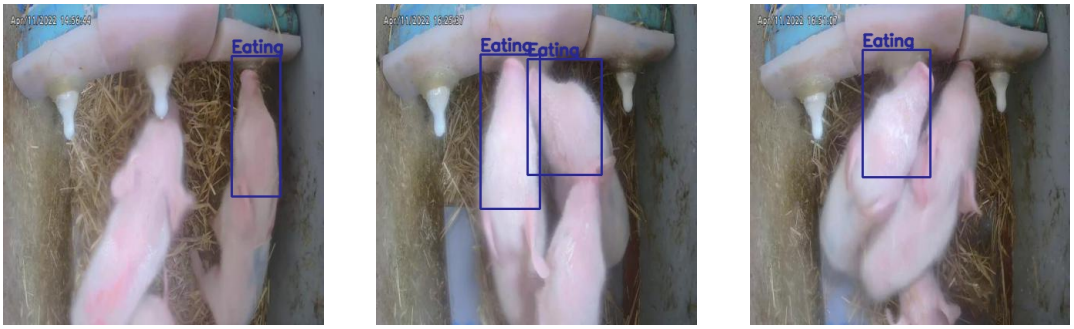

Figure A.8: Examples of detections using the Faster R-CNN with the ResNet-50 + FPN as a backbone.

Figure A.9: Examples of detections using the Faster R-CNN with the SqueezeNet1_0 as a backbone.

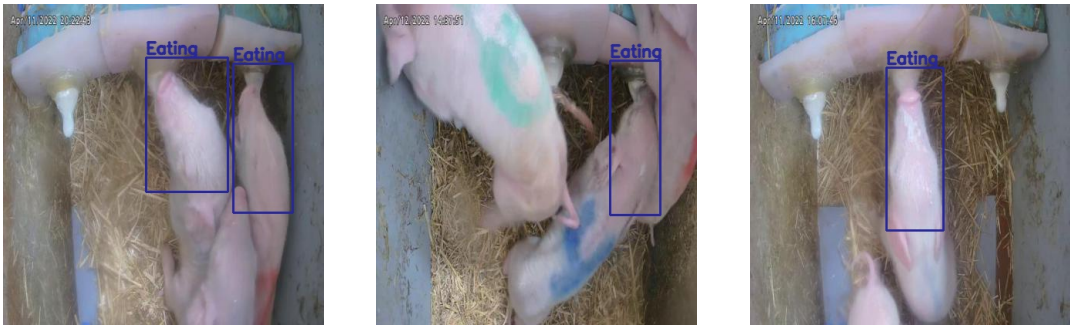

Figure A.10: Examples of detections using the Faster R-CNN with the SqueezeNet1_0 as a backbone.

Figure A.11: Examples of detections using the Faster R-CNN with the SqueezeNet1_0 as a backbone.

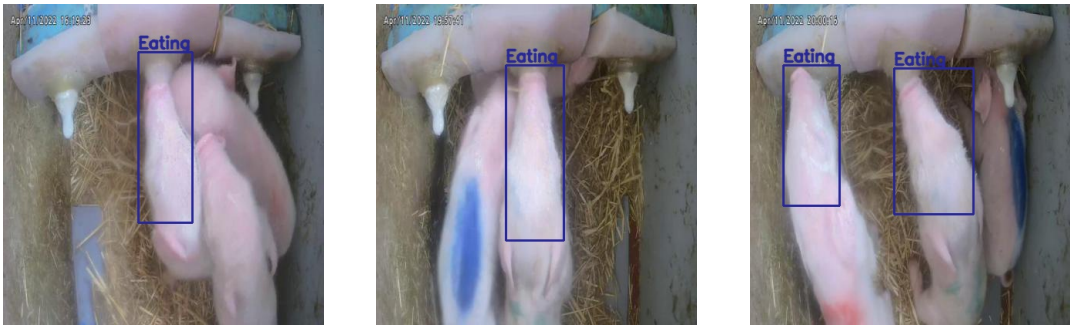

Figure A.12: Examples of detections using the Faster R-CNN with the SqueezeNet1_0 as a backbone.

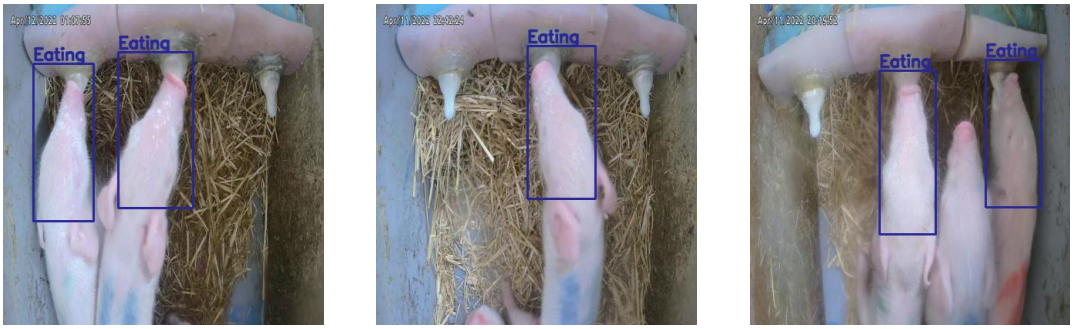

Figure A.13: Examples of detections using the Faster R-CNN with the SqueezeNet1_1 as a backbone.

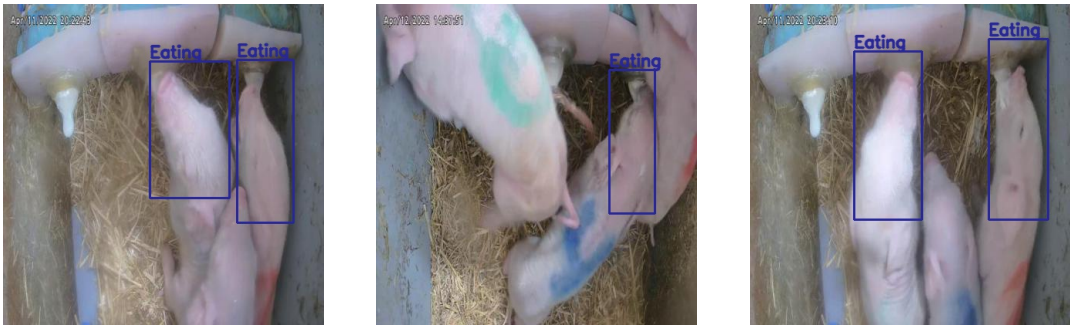

Figure A.14: Examples of detections using the Faster R-CNN with the SqueezeNet1_1 as a backbone.

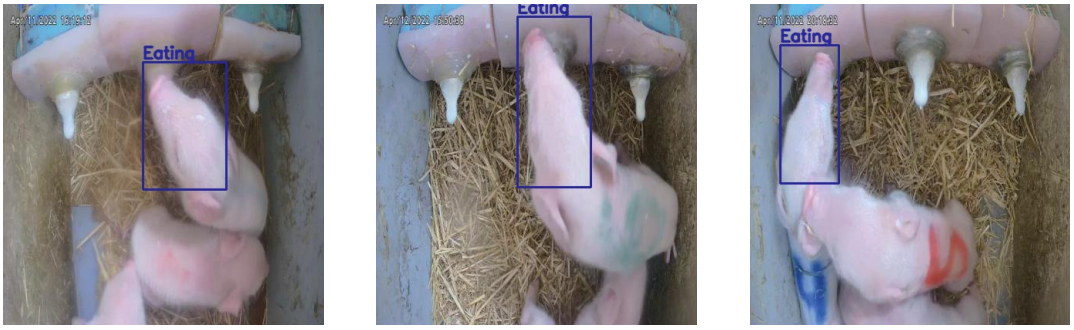

Figure A.15: Examples of detections using the Faster R-CNN with the SqueezeNet1_1 as a backbone.

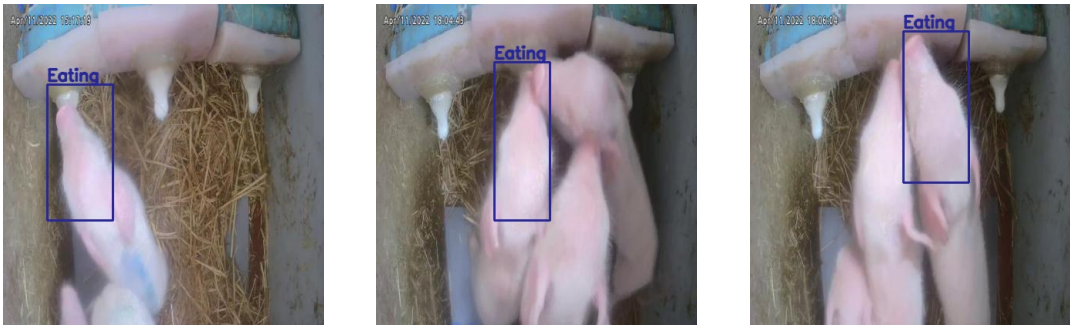

Figure A.16: Examples of detections using the Faster R-CNN with the SqueezeNet1_1 as a backbone.

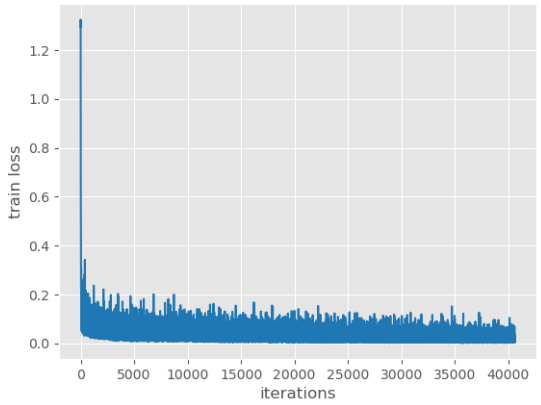

Figure A.17: Plot of loss over iterations during training for the Faster R-CNN using ResNet-18 as backbone on the dataset of piglets feeding.

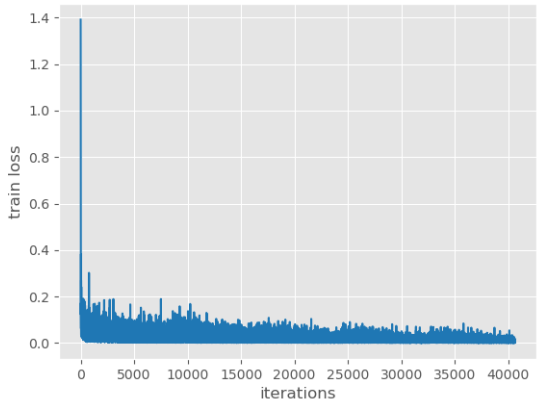

Figure A.18: Plot of loss over iterations during training for the Faster R-CNN using ResNet-50 + FPN as backbone on the dataset of piglets feeding.

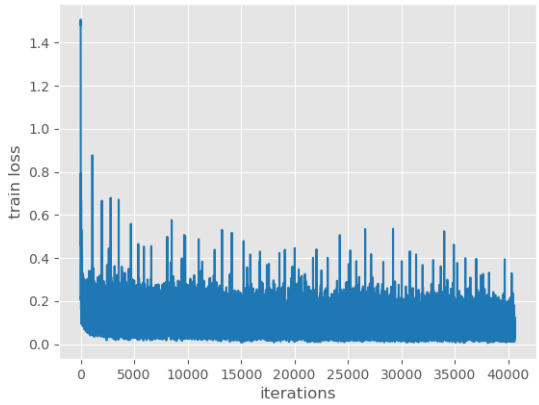

Figure A.19: Plot of loss over iterations during training for the Faster R-CNN using SqueezeNet1_0 as backbone on the dataset of piglets feeding.

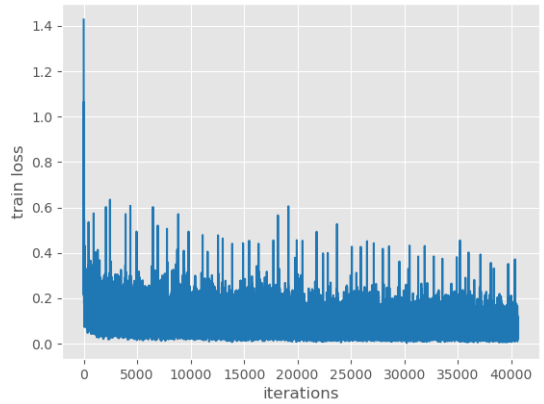

Figure A.20: Plot of loss over iterations during training for the Faster R-CNN using SqueezeNet1_1 as backbone on the dataset of piglets feeding.

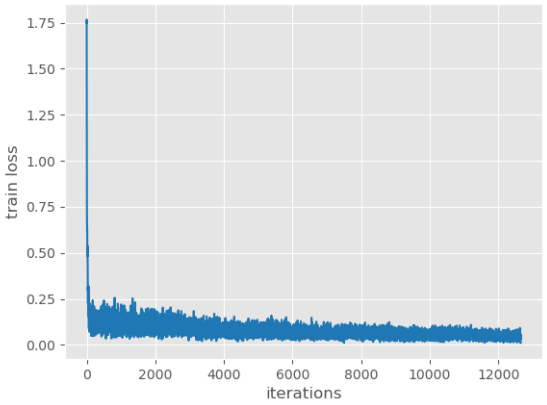

Figure A.21: Plot of loss over iterations during training for the Faster R-CNN using ResNet-18 as backbone on the dataset for the colors on the piglets backs.