Pig Localization using Computer Vision

T.J.J. Wittendorp

S1739085

First Supervisor University of Twente: dr.ing. G. Englebienne

Second Supervisor University of Twente: dr. M. Poel

Supervisor Nedap: dr. R. Aly

UNIVERSITY OF TWENTE

Master Interaction Technology

nedap

UNIVERSITY OF TWENTE.

Abstract

Finding a pig in a large herd is tedious as pigs are housed in large groups and look-alike for humans. This project aims to solve the problem of finding pigs in a shed using computer vision. The problem is split up into acquiring an image of pigs, detecting, identifying and locating pigs. The focus is on the identification of pigs. A method similar to human face recognition is proposed where new pigs can be added without retraining. A dataset of 105 individual pigs is collected to train the network and evaluate the method. The evaluation shows that ten pigs seen during training of the embedding network can be identified with an accuracy of 87.8%. Identifying pigs not seen during training of the embedding network can be identified with an accuracy of 55.5%. There are still limitations when detecting pigs in another perspective or later in time than added to the identification system.

Acknowledgements

During the project, I received support from several people whom I would like to thank. I would like to thank my first supervisor Gwenn Englebienne for his guidance and supervision throughout the project. The weekly meetings were always helpful. I also want to thank my second supervisor Mannes Poel for his feedback. The project was carried out on behalf of Nedap Livestock Management. I would like to thank the innovation team members at Nedap for their support. I want to thank Robin Aly, my supervisor from Nedap, for his guidance and feedback during the project. I also want to thank my girlfriend Veerle and my family for their support during the project and the positive distraction.

Table of Contents

Abstract *2*

Acknowledgements *3*

1. Introduction *7*

1.1 Context **7**

1.2 Problem Definition **7**

1.3 Challenge **8**

1.4 Research Questions **8**

1.5 Structure **9**

2. Literature Review *10*

2.1 Capture **11**

2.2 Detection **13**

2.3 Identification **16**

2.4 Localization **19**

2.5 Conclusion **21**

3. Technology Exploration *23*

3.1 Detection **23**

3.2 Identification **26**

3.2.1 Triplet loss function 26

3.2.2 Triplet loss on pigs 30

3.3 Conclusion technology exploration **35**

4. Data Collection *36*

4.1 Environment **36**

4.2 Raspberry Pi Camera **37**

4.2.1 Collected Videos 38

4.3 Security Camera 39

4.3.1 Collected Videos 39

4.4 Video Preprocessing 40

4.4.1 Labeling raspberry pi videos controlled perspective 40

4.4.2 Labeling raspberry pi videos overview perspective 40

4.4.3 Labeling security camera videos 41

4.4.4 Cropping head in videos 42

5. Method 43

5.1 Comparison with existing methods - Method 45

5.2 Identify and verify pigs not trained on - Method 46

5.3 Identify and verify pigs in later data 48

5.4 Identify and verify pigs in overview perspective 49

6. Results 51

6.1 Compare with the existing methods 51

6.1.2 Results Standard Classification Network 51

6.1.2 Results Embedding Network with KNN Classification 52

6.1.3 Results Embedding Network with Distance Threshold Verification 52

6.2 Identify pigs not trained on 54

6.2.1 Pigs not trained on - identify 54

6.2.1.2 varying gallery size 55

6.2.2 Pigs not trained on – verify 55

6.3 Identify pigs in later data 57

6.3.1 Results experiment 1 later - identify 57

6.3.1 Results experiment 1 later - verify 58

6.3.2 Results experiment 2 later - Identify 59

6.3.3 Results experiment 2 later – Verify 59

6.4 Identify pigs from in overview perspective 61

6.4.1 Results experiment 1 overview perspective - identify 61

6.4.2 Results experiment 1 overview perspective – verify 62

6.4.3 Results Experiment 2 overview perspective - Identify 63

6.4.4 Results Experiment 2 overview perspective – Verify 64

7. Discussion 65

8. Conclusion and Recommendation 69

References 70

1. Introduction

1.1 Context

Pig farms can nowadays hold thousands of pigs. These large amounts of animals bring challenges. As pig herds grow even further, it is almost impossible for farmers to assess individual animals to assure their well-being. The farmer might need to know where a particular pig is when the pigs need to get diagnosed or treated for an illness. At this moment, finding a pig in a barn is cumbersome since someone has to search for the animal himself, and to humans, pigs look alike. Automating this task could reduce a person's workload. A system that would find a particular animal should know which pig is which and where the pigs are located in the bard. The system should thus be able to detect, identify, and locate the pigs inside the barn and navigate the farmer to this pig. This thesis focuses on the technology that can assign a location to an identified pig and not on navigating the farmer to a pig.

There are multiple technologies used for locating animals on farms. RFID sensors are one of these technologies used to identify pigs from short distances [1]. Wireless technologies are another type of technology used for identification and localisation. Bluetooth [2] and WLAN [3] are wireless technologies that can be used from further distances to locate cows inside a barn. Besides wireless technologies, computer vision can also be used to identify and locate pigs. Using cameras over wireless technologies have the advantage that it is non-invasive. For this reason, this thesis focuses on computer vision-based solutions.

1.2 Problem Definition

The problem of locating pigs will be split up into multiple steps. The steps are the different steps a system has to do to locate pigs. The different steps are visualised in Figure 1. The first step is capturing images or videos of the pigs. Different types of cameras can be used for this. The second step is detecting each pig in the video. Each pig in the video should be isolated so that only one pig is identified at a time. Identifying pigs

is the third step of the process. The last step is obtaining the actual location of the pigs inside the shed.

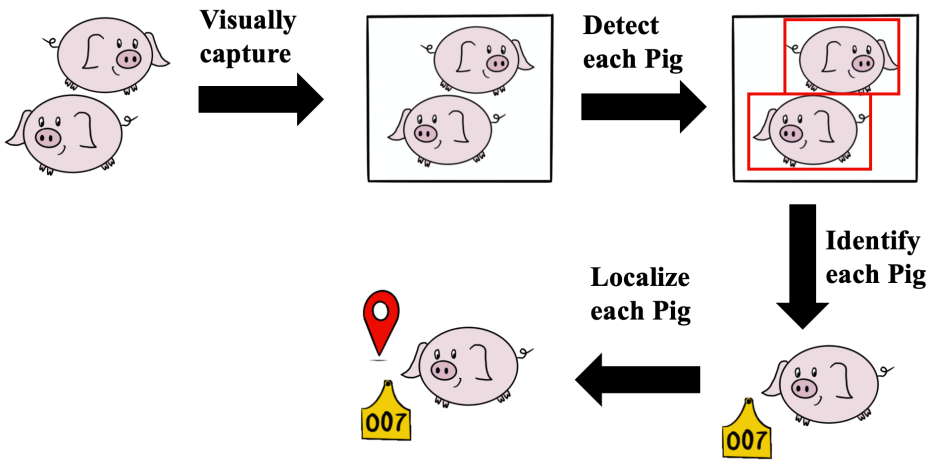


Figure 1: Visualisation of four steps

1.3 Challenge

The literature shows that the existing methods used for identifying pigs using computer vision can only identify pigs that it has been trained on. This is unsuitable for real-world applications where pigs are continuously added and removed from the group. The main challenge of identifying pigs using computer vision is that it should work in a setting where pigs are continuously added and removed from the group. The main challenge is to add pigs to the identification system without training again.

1.4 Research Questions

The main research question of this project is:

"How can pigs be identified in a real-world application where pigs are continuously added and removed from the group?"

The main research question is split up into multiple research questions. These research questions will be answered using a prototype of the proposed methods. During the evaluation of the prototype, the following research questions will be answered:

"How does our identification method compare to the existing method?"

"How well can our identification method identify pigs not seen during training?"

"How does our identification method perform on newer data?"

"How does our identification method perform in other camera perspectives?"

1.5 Structure

This thesis starts with a literature review to identify what has already been done and the shortcomings. A prototype based on this is made to identify if the technologies found suit our purpose. This is described in the technology exploration chapter. A dataset of pig heads is collected for evaluation of our identification method. The data collection is described in the data collection chapter. Experiments are done using this collected dataset to answer the research questions. These experiments are described in the method chapter and their results in the results chapter.

2. Literature Review

A literature review is done to review what prior work has been done. There is not yet a system that can identify and locate pigs using computer visions. However, the different parts of the problem are done. This literature review examines the four steps explained in the introduction and evaluate their suitability for our application. The five questions that are answered in the literature review are:

"What cameras are used to capture the pig?"

"How can pigs be detected in an image?"

"How can individual pigs be identified?"

"How can the location of an individual pig be determined?"

"Are the existing methods good enough for our purpose?"

The type of camera that should be used will be determined by reviewing what type of camera other studies used for pig detection and identification. The pig detection question will be answered by looking at studies that focused on detecting pigs in a video or image. Identification and verification will however review studies of pigs, other livestock animals and humans. Localizing the pigs focuses on calibrating the cameras. Search terms are chosen based on these topics. The following search terms are used: PIG DETECTION, PIG FACE RECOGNITION, ANIMAL FACE VERIFICATION, ANIMAL BIOMETRIC VERIFICATION, LIVESTOCK BIOMETRICS, LIVESTOCK BIOMETRIC VERIFICATION, CAMERA CALIBRATION, AUTOMATIC CAMERA CALIBRATION. Google Scholar is used as database to search in. The articles of the first five pages are selected on the title, abstract, peer review status and availability in the University of Twente library. A snowball method is used, with the starting point being the literature selected using the search terms. The literature review should indicate how individual pigs can be captured, detected, identified and located and if these methods are good enough for our purpose.

2.1 Capture

Capturing the pigs in digital content is the first step in identifying pigs using computer vision, and different types of cameras can be used for this. This section will review different cameras. The technology chosen to capture the pigs influences the performance of the identification. There are multiple cameras besides the standard RGB camera that can be found in many consumer products. These RGB cameras can capture images very similar to what the human eye can see. However, some cameras can capture depth, thermal information, or perform very well in low light conditions. Each type of camera has its pros and cons, and every camera is suited for different applicants. A camera will be selected to identify pigs in a pig shed. The cameras that will be considered are an RGB camera, depth camera, infrared camera for thermal imaging and an infrared camera for night vision.

The environment in which the camera will be used affects the camera choice. The camera will be used in a pig shed which can have some problematic conditions for capturing the pigs. According to [4] and [5], light conditions inside a pig shed are challenging. Sudden light fluctuations, direct sunlight and low light conditions can affect how well the camera can capture the pigs. The second major challenge mentioned in paper [4] is object deformations and occlusions. Insects might corer the lens, or pigs might occlude each other when they are close together. Selecting the right camera might help overcome these challenges.

Several studies are done where an image or video is used to detect pigs. The camera used for capturing the data in these studies and their results can help determine a suitable camera. Most of the cameras used in pig detection and identification papers are RGB cameras. Thermal cameras are not used in pig detecting papers. However, thermal cameras are used to measure the pigs' temperature to detect diseases [an overview of]. Thermal cameras for detection is, in contrast to pigs, used for detecting cows [6], [7]. Detection cows using thermal cameras seem to perform well. However, it is expected that the performance for pigs is less. This is because pigs tend to get closer to each other than cows. This might result in the pigs becoming a blur on the thermal camera. Depth

cameras are used quite frequently, especially in recent years. All depth cameras are used in combination with either an infrared camera or an RGB camera. The depth information is used for both detecting pigs using a threshold or when in combination with an RGB camera for vision in the dark. The depth camera's used in the studies were either a Microsoft Kinect or an Intel RealSense. Two studies that used a Kinect [8], [9] mentioned that the Kinect has poor depth measurements at distances larger than 4 meters. Depth information is also used to overcome low light conditions, as seen in [10], which is used in combination with an infrared camera. Challenging light conditions can also be overcome using a somewhat simpler solution which is an infrared security camera. The data from an infrared camera, which is a grayscale image, is easier to process while still performing well at detecting pigs in low light conditions [5]. The Infrared cameras used in the two studies are security cameras that automatically switch between RGB and infrared, based on the lighting conditions. Although the camera switches between RGB and infrared, a color image might not be needed at all. Pig face recognition, as proposed in[11], performed significantly better on grayscale images compared to RGB images. It is expected that the Kinect and RealSense depth cameras, and thermal cameras are not suitable for identifying pigs. The depth resolution of the Kinect and RealSense are not high enough to see detail about the pigs [8], [9].

Lighting conditions inside a pig shed seem to have a strong influence on the camera choice. Some studies use depth cameras to overcome this, while others use infrared cameras. Footages of both camera types seem to perform well when it comes to detecting pigs in low light conditions. RGB cameras perform well in a room with good lighting. However, they perform poorly in difficult lighting conditions. The performance of thermal cameras in detecting pigs is unknown; however, it is expected to perform poorly since pigs tend to be close together. While depth cameras perform equally as infrared cameras, they might not be the best option. Accessible depth cameras such as the Kinect or RealSense are not designed for harsh conditions such as a pig shed, in contrast to security cameras. Using a security camera that can capture RGB and infrared for pig identification seems to be the best solution.

2.2 Detection

The pigs or part of the pigs in the captured image have to be detected before they can be identified. This is because later on, the identifier should identify each pig separately. This section will review different detection and segmentation methods. The image has to be cropped in such a way that there is only one pig in the image. This can either be done using detection or segmentation. Detection deals with detecting every instance of a pig in an image and drawing a bounding box around every pig. This bounding box can be used to crop the image. Segmentation identifies every pixel if it belongs to an instance and cuts the pig out along its contours. There are numerous ways to segment or detect pigs in an image, for example, by thresholding, edge detection or object detection using machine learning. There have been done quite some attempts at segmenting and detecting pigs, and they will be reviewed to choose the right method to identify later and locate pigs.

Background subtracting can be used to separate pigs from the background and uses multiple images to separate the foreground from the background. McFarlane and Schofield [4] used image differencing with respect to the background and a Laplacian operator to detect pigs in 1995. Although the method used a low-resolution camera, it still performed well on single piglets.

Thresholding and the gaussian mixture model are two other techniques that can be used to separate the foreground and background. The thresholding technique either can threshold the color intensity or depth in case a depth camera is used. Thresholding the color intensity will block darker colors. In the case of pigs in a pig shed, the pigs will be segmented from the background since pigs are usually brighter than the background. This thresholding technique based on color intensity can be seen in [12] and [13]. Both papers use a dynamic threshold value since lighting conditions have much influence on the threshold value. After thresholding the image, an ellipse fitting method is done in [13] to actually detect a pig, while in paper [12], only the center of a blob is calculated.

Depth information from a depth camera can also be thresholded to detect and segment pigs [9], [10]. The threshold is set between the distance between the camera and the pig, and the camera and the floor. This method is relatively simple compared to

thresholding the colour intensity and works excellent in various lighting conditions. The accuracy of detecting pigs using a depth threshold performs well and can reach an accuracy of 94% [9], although that method combined depth thresholding with a convolutional neural network. The method proposed in [10] only used depth information and reached an accuracy of 79%.

The thresholding method seems to perform well at separating pigs from the background, however, there can be problems when pigs get too close to each other, and multiple pigs are detected as one [12].

A gaussian mixture model can be used to separate the foreground and background besides thresholding. The Gaussian mixture model, or GMM, can, for instance, be seen in [14], where it is combined with a dyadic wavelet transform. This segmentation was then used to track the pigs with a tracking error of 5 percent. The Gaussian mixture model is used as well in paper [8], where it is used in combination with a convolutional neural network. This paper provided a method on how two pigs that are close to each other can be separated. The problem with background subtraction is that if two pigs are close to each other, they will be segmented as one pig. The method in paper [8] first subtracts the background with a GMM, after which a convolutional neural network detects the individual pigs. It is expected that all methods that use a foreground and background separation technique suffer from the problem that multiple pigs that are close to each other will be segmented as a whole and thus be detected as one pig.

Another method to detect and segment pigs in an image is by using a convolutional neural network, or CNN in short. This learning-based approach uses an artificial neural network to analyse images. There are different approaches for object detection, for example; Single Shot Detector (SSD), You Only Look Once (YOLO), RefineDet, RetinaNet, R-FCN, R-CNN, and Faster-R-CNN. Approaches like SSD and YOLO give a bounding box around the detected object, while R-CNN and Faster-R-CNN provide a region of interest. These approaches have different accuracies and speeds between which a trade-off has to be made. A higher accuracy generally means a lower speed. In paper [15], an approach for detecting pigs and their posture is explained using a Faster-RCNN

method. This approach performed well at detecting pigs with a 97% precision. However, it did not perform well at detecting the pigs' posture. A comparison between R-FCN, Faster-RCNN and SSD is made in [5] regarding speed and accuracy while detecting pigs. SSD has the highest accuracy of the three while also being the quickest. A custom convolutional neural network is proposed in [16] and which has a precision of 99% at detecting pigs and a recall of 95%. This approach detects not only pigs but also four different parts of the pig's body, namely both ears, shoulder and back. This might be beneficial if, for example, the identification only needs a pigs face. The CNN based pig detection approaches seem to perform really well, even if pigs are close to each other or partly obstructing each other [5]. The downside of these CNN based approaches is that they are more complex compared to the previous methods and need to be trained.

Having a good pig detector is crucial for identifying pigs. If the system can't detect pigs and segment them, identifying will also not be possible. The different methods all have their pros and cons. Threshold methods are one of the easiest methods of detecting pigs in an image. Thresholding does show great performance with single piglets, however, it has difficulties to detect individual pigs when they are close. This problem occurs on both thresholding depth and color information. Background subtraction using GMM has the same problems as thresholding. Convolutional neural networks perform great and can even detect different parts of the pig's body. The advantage of this is that only the pig's face can be detected, which might be beneficial for identification. CNN's still perform well in different lighting conditions and have much fewer difficulties if pigs get close to each other. The downside of a CNN is that it is a much more complex method of detecting pigs. Considering the importance of detecting in the whole system, a CNN will most likely be the best method.

2.3 Identification

After a pig is detected in an image, it has to be identified. This way, the system knows which pig is which. This can either be done by identification or verification method. This information can later be used to determine the location of a particular pig. There are multiple ways of identifying and verifying a pig. While some methods use markings or ear tags to identify [17], others use face recognition to identify a pig [11], [18]. Identifying a pig using face recognition, or using the pig's whole body, is the preferred way to identify pigs. Since pig identification using computer vision is not very common, methods for identification and verification of other livestock animals will be reviewed as well as identification and verification for people.

Pig face recognition systems seem to perform great although pigs look the same for people. A comparison between three different methods is made in paper [18]. The comparison between fisher face, VVG and a custom CNN is made. FisherFace and VVG are two methods that are used for human face recognition. They showed that a custom CNN outperformed the two human face recognition methods, and the CNN reached an accuracy of 97%. The CNN was trained on ten pigs using over 1500 images. These images were closeups of the pig's faces. A visualisation of what regions the network learned most shows that the system looks primarily at the forehead and nose. A visualization of the learning regions can be seen in *Figure 2*.

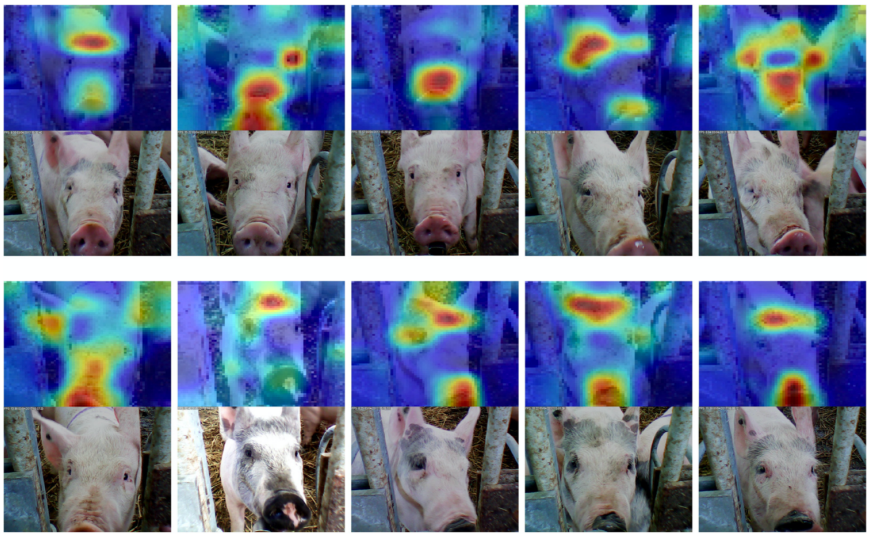


Figure 2: Close up of pig face and visualization of learning regions [11]

Another study [11] that also trained a CNN to recognize pig faces show a similar map of what the network looks at. These regions can also be visible from a top view, which might allow for face recognition using a top view. Using a CNN to identify pig faces seems to work well. The papers show that face recognition for pigs is possible. The networks of both papers were trained on ten particular pigs. This means that these networks won't allow for more pigs to be added or changed for different pigs. For every new pig added, the network has to be trained again, which is not feasible for real-world usage. A verification method should be able to verify pigs without training again, however there are no studies found that focus on pig verification.

A different approach for pig face recognition is presented by Wang and Liu [19]. Instead of using a network with an output for each pig, they trained a network that outputs embeddings where similar pigs are grouped together. This is done using the triplet loss function in combination with a full-connected layer. In this study 28 pigs are used to train the network and these 28 pigs are also used to test it. This is done by splitting it into 200 train images per pig and 50 test images. The grouped embeddings are classified using a k-nearest-neighbour (KNN) classifier. Six different networks are tested, and the classification accuracy for the different networks range between 82% to 97%. This is a similar accuracy as the previous two pig face recognition papers. While the performance is similar, this approach does have an advantage over the previous two. It might be possible to add new pigs to this network because it does not have a fixed network output for each pig or use it as a verification method. It should be noted that this research was published during the project.

Verification of cows can be done using the cow’s muzzle because the muzzle pattern of a cow is a unique biometric characteristic for cows[20]. The method proposed by Kumar et al. [20] matches the muzzle features from test images with the features from a stored image. Matching is done using a one-shot similarity and incremental first-class SVM learning model. The method yields an accuracy of 96.87% tested on 500 cows. It is not expected that muzzle images can be used for recognizing pigs. Another technique of identifying cows is using face recognition. A 3D face recognition system is proposed by

Yeleshetty et al. [21]. A point cloud of a cow's face is captured using two 3D cameras. A point cloud of a test image is compared with a point cloud in the point gallery using the RMSE after aligning the two point clouds. A set of 32 cows with 5 point cloud per cow can be identified with an accuracy of 99.53%. This method seems to work well for identifying cows nearby. However, it is expected that the resolution of a point cloud obtained in an overview perspective of a barn is not high enough for identification. A 2D face reidentification is demonstrated by the authors of [22], where a single view reidentification is compared with a multi-view reidentification. They trained a CNN that outputs embeddings. These embeddings can be classified using a KNN with an accuracy of 74.8% for a single view and 89.1% for a multi-view. This is lower than 3D face recognition. However it does not require a 3D camera.

Face identification and verification is most seen for humans. One of the better face recognition methods is Google's Facenet [23]. FaceNet uses a deep learning method that calculates embeddings from face images. Embeddings of similar identities are grouped together. This is done by using a triplet loss function where similar identities are brought close together while different identities are brought further apart. These clusters are created by training a network using triplet loss. Images of similar identities are brought close together, while images of different identities are brought further apart. This method is used in Wang and Liu's pig face recognition paper [19]. However, FaceNet is trained on 100 to 200 million images of 8 million different identities. This results in a network that can distinguish images not seen during training. The embeddings are verified by thresholding the Euclidian distance between two embeddings. Identification is done by using a k-nearest-neighbour classifier. FaceNet reaches an accuracy of 99.6% on the Labeled Faces in the Wild (LFW) dataset. Face verification using the triplet loss function is also presented by Parkhi et al. [24] with similar results. However, a much smaller dataset of 2.6K identities is used instead of the 8M identities used for facenet. Another face verification approach, called deepface, is presented by researchers at Facebook [25]. Deepface uses 2D and 3D face alignment to get a frontal view of a person's face. This frontal view is verified using a Siamese neural network, two similar neural networks that

verify two images based on the similarity. Deepface reaches an accuracy of 97.35% on the LFW dataset, which is lower than facenet. The deep learning verification methods of livestock animals and humans have in common that embeddings are calculated from two images that are compared for verification.

The reviewed pigs' identification methods can identify pigs with high accuracy. However, these methods can only identify pigs that are seen during training. Verification methods for livestock animals and humans can identify identities not seen during training with high accuracies. Using triplet loss to train an identification or verification network seems to be one of the better methods for human face verification. The triplet loss function is also used for pig face recognition [19], but it was tested on the same pigs as trained. It is expected that the triplet loss method can also be used for identifying or verifying pigs not seen during training.

2.4 Localization

The position of a pig in the barn has to be determined once a pig is identified. After identification, the system knows where each pig is on the image. This pig's location on the image is however not the same location as in the shed. An understanding of the camera and how the camera is positioned in the world is needed to calculate the actual position of the pigs inside the shed. Methods to get this understanding will be reviewed in this section.

The camera can be modelled using the camera pinhole model to get a better understanding of the camera and how it relates to the world. This camera pinhole model describes the mathematical relationship between a point in 3D space and its projection into the image plane. This pinhole camera model both has intrinsic and extrinsic parameters. The extrinsic parameters describe the rotation and translation of the camera. The intrinsic parameters include camera characteristics such as the focal length, optical centre and skew coefficient. Both the extrinsic and intrinsic parameters can be expressed as matrixes that can be used to calculate world coordinates from pixel coordinates and vice versa. Lens distortion is missing in the pinhole camera model. A perfect lens is

assumed in this model that has no distortion. However, not all lenses are perfect, and lens correction has to be applied if the distortion is too much and affects performance.

Estimating the intrinsic parameters is also known as camera calibration. The most used method is Zhang's method [26]. This method is designed to be used by people without special knowledge of camera calibration. The camera can be calibrated using a picture of a checkerboard with a known size. At least two images are used to calculate the intrinsic parameters. The extrinsic parameters can be calculated using the checkerboard as well. In this case, the extrinsic parameters express how the camera is positioned in relation to the checkerboard. Zhang's method is implemented in the Matlab camera calibrator in combination with a model for lens distortion.

While Zhang's method is the most used, there are other methods to calibrate a camera. Vanishing points can be used to determine both the intrinsic and extrinsic parameters [27]. These vanishing points are obtained by selecting parallel lines in an image. The perspective view will cause parallel lines to intersect. Intersections in three orthogonal directions are needed to calculate the intrinsic or extrinsic parameters. Calibrating the camera using vanishing points is less accurate than Zhang's method. However, the advantage is that there is no need for a checkerboard and calibration can be done after taking the images. Calibrating a camera using vanishing points does mean that parallel lines in three orthogonal directions should be visible in the image. This might not be the case if a top view which means that the vanishing point calibration might not be feasible. The vanishing point method also assumes that there is no distortion in the image.

The previous two methods require manual interaction to calibrate the camera. Automatic calibration of the intrinsic parameters can be achieved using a convolutional neural network. DeepCalib can calibrate wide field-of-view cameras automatically [28]. This method does perform not as good as Zhang's method, but it can be used if manual calibration using a checkerboard can not be done.

There are multiple ways to calibrate a camera manually or automatic. However, there is still a challenge when determining the position of pigs. That challenge is determining the coordinate system in which the location of the pigs will be expressed.

The methods described previously choose a world coordinate system somewhere on the image. However, in the case of locating pigs, the world coordinate system might be somewhere not visible on the image. If multiple cameras are used they should use the same coordinate system. A ground plan of the pig farm might be needed with the location of each camera and a coordinate system.

The camera used to identify pigs has to be calibrated to obtain the position of the pigs. This calibration consists of the intrinsic and extrinsic parameters. There are multiple methods of calibration of the intrinsic parameters. Zhang's method is the most popular and seems to estimate the intrinsic parameters with high precision compared to others. Calibrating the extrinsic parameters are a bit more challenging because a world coordinate system has to be chosen somewhere in the pig farm that might not be visible on the image. Extrinsic parameter calibration is dependent on where the camera is installed in the farm, while the intrinsic parameters stay the same for every camera. Calibration of the extrinsic parameters can be done with Zhang's method in combination with a linear transformation to set the coordinate system to somewhere in the pig farm. Once both the intrinsic and extrinsic parameters of the camera are calibrated, the identified pigs can be located in the shed.

2.5 Conclusion

This literature review aimed to understand what methods are available that can identify and locate pigs using computer vision and assess whether these methods are suitable. Four sub-questions are answered to understand how the system can be made. Capturing pigs can be done using different cameras. Lighting conditions have an influence on how well the cameras can capture pigs. Security cameras with an infrared camera can capture well in difficult lighting conditions and the footage can be used to detect and identify pigs. This is why security cameras seem to be the best camera to captures pigs for detection and identification. A convolutional neural network can be used to detect the individual pigs in a video. CNNs can still detect pigs in different lighting conditions and if pigs get close to each other, which is why a CNN is the best technique to detect pigs. Besides detecting

pigs, CNNs can also be used to identify pigs. The proposed methods in the literature for capturing, detecting, and localization seem to be suitable for our application. However, the proposed methods for pig identification do not seem to be suitable. The proposed methods can only identify pigs that it has been trained on. The network has to be trained again in case new pigs are added, which is not feasible for a real-world application. Identification methods for other livestock animals and humans do seem to be suitable for identifying pigs. Triplet loss is a training technique seen in human identification and verification which is also used for identifying the same pigs as trained on. It is therefore expected that triplet loss can be used to identify and verify pigs not seen during training.

3. Technology Exploration

The technology found in the literature will be explored with the aim of identifying if and how they are suitable for our application. The literature review showed that identification of pigs is where could be improved upon the most. Pig detection is an essential step before identification can be done on our dataset which is why this chapter will also focus on pig detection. The research question if the technologies are suitable for our application will be answered using a research-through-design methodology [29]. Research through Design is a method that attempts to discover knowledge by doing design work. A prototype of pig detection and identification is made and evaluated. The outcome of the technology exploration is a detection and identification technique suitable for our application with the available data and hardware.

3.1 Detection

The literature review showed that a convolutional neural network seems the best option to detect pigs. There are many types of convolutional neural networks that can detect objects. For a detection method to be suitable for our application, it should be able to detect every pig in an image or video and should be able to do this real-time or close to real-time. A fast detector is eventually wanted to identify and locate the pigs in real-time.

A comparison between different object detectors detecting pig faces is made in [19] between efficientDet, Faster RCNN, MobilenetV3-SDD, SSD, Tiny-YOLO v3 and YOLOv3. This comparison showed that tiny-YOLOv3 is the best balance between accuracy and speed. Tiny-YOLOv3 runs at 7.9ms with an average precision of 98.86%. An improved version of YOLO v3, YOLO v4, is presented in [30] and improves YOLO v3's AP and FPS by 10% and 12% respectively. Yolov4 tiny, the faster version of yolov4 seems to be the best object detector for our purpose. Yolov4 tiny is trained using the existing dataset to evaluate if it is suitable for our application using the available data.

Five existing datasets are used for the detection experiment. These datasets are all the available datasets found in the literature. All datasets consist of a top view or slightly angled view. An overview of the datasets and information can be seen in *Table 1*. An

example of each of the five datasets can be seen in Figure 3. The datasets are referred to using the numbers seen in table 1. This dataset consists of both RGB and infrared videos. The videos are annotated with points on the shoulder and tail of this pig and the pig's ear tag number. Dataset 2 has the same shoulder-tail annotations as dataset 1 but has a point on each ear. Datasets 1 and 2 do not contain bounding box annotation. A bounding box could be calculated using the shoulder-tail annotation but is not done for training the detector. Datasets 3, 4 and 5 contain bounding box annotations which are used to train a custom detector. There are no datasets available with images of pigs' faces.

Table 1: Datasets used for pig detection

Dataset Number	**Data type**	**Number of annotated images/frames**	**Annotation type**	**Reference**
1	Videos	15 videos of 30m at 5fps	Shoulder point, tail point, ID	[17]
2	Images	2000 images	Shoulder, tail, and ears point	[16]
3	Images	297 images	Bounding boxes and standing/lying position	[15]
4	Videos and images	380 images	Bounding boxes	[31]
5	Videos	12 videos of 1m at 30fps	Bounding boxes, behaviour and ID	[32]

Figure 3: Examples of datasets. Top left) dataset 1, Top middle) dataset 2, Top right) dataset 3, Bottom left) dataset 4, Bottom right) dataset 5

The data of datasets 3, 4, 5, and 8 manually annotated images of dataset 1 will be used to train a custom YOLO v4 tiny object detector. Bounding boxes of datasets 1 and 2 could be calculated but are inaccurate and will not be used. In total, 1047 images are used containing 13.558 individual pigs. The images are split into 95% train and 5% test images. This split is done for each dataset separately. The detector is trained for 6000 iterations. The mAP is calculated by dividing the intersection over the union of the predicted bounding box and labelled bounding box and thresholding this at 50%. The highest mAP is 82.1%. This is lower than reported in [19], however it seems to predict every pig correctly in Figure 4. The yolov4 tiny detector ran at 1.5 fps without the use of a GPU on a 2019 MacBook.

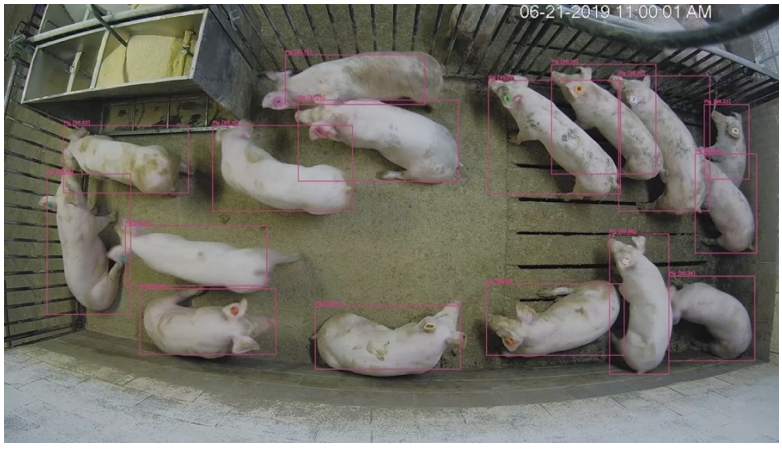


Figure 4: Pig detection using yolov4 tiny

3.2 Identification

Literature showed that the triplet loss could be used to identify or verify pigs not seen during training. This allows for identifying and verifying newly added pigs to the group without retraining a network. This section explains how triplet loss can be used, how different types of data influence a network trained using triplet loss and how various networks behave on data of pigs.

3.2.1 Triplet loss function

The identification system strives for an embedding where the distance between similar identities is small, and the distance between different identities is large. A prototype of such a system will be made using the MNIST handwritten database. The prototype aims to understand how triplet loss works and evaluate the feasibility of using the triplet loss function.

The triplet loss function uses three images to calculate the loss, hence the name triplet loss. The triplet loss function tries to bring an image of a pig (Anchor) closer to another image of the same pig (Positive) than an image of a different pig (Negative). This is visualised in Figure 5. This way, the network is trained to group embeddings of images with a similar identity.

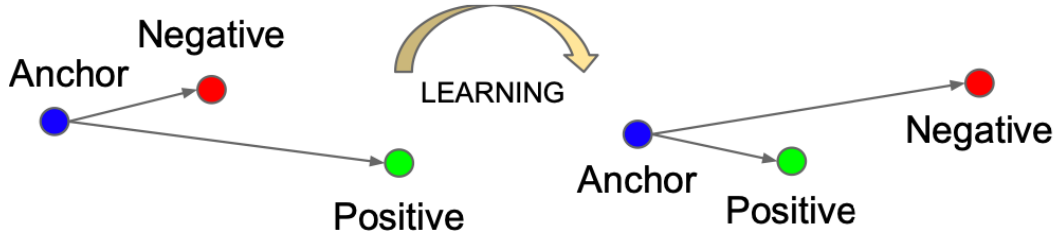


Figure 5: Triplet loss decreases the distance between Anchor and Positive, and increases the distance between Anchor and Negative [23]

Bringing the positive closer to the anchor than the negative is formalised as follows, where alpha is a margin between the positive and negative pairs:

$$||f(Anchor) - f(positive)||^2 + \alpha \leq ||f(Anchor) - f(Negative)||^2$$

Three types of triplets could be generated: easy triplets, hard triplets, and semi-hard triplets. The three different triplets types are visualised in Figure 6. Moindrot [33] describes these three types of triplets as follows.

- Easy triplets are triplets where the negative plus a margin has a larger squared distance to the anchor than the positive to the anchor and thus satisfy the constrain.
- Hard triplets are triplets where the negative is chosen to have the minimum squared distance to the anchor and the positive to have the maximum squared distance to the anchor.
- Semi-hard triplets are triplets where the squared distance between the negative and anchor is larger than the distance between the positive and anchor, but smaller than the distance between the positive and anchor plus a margin.

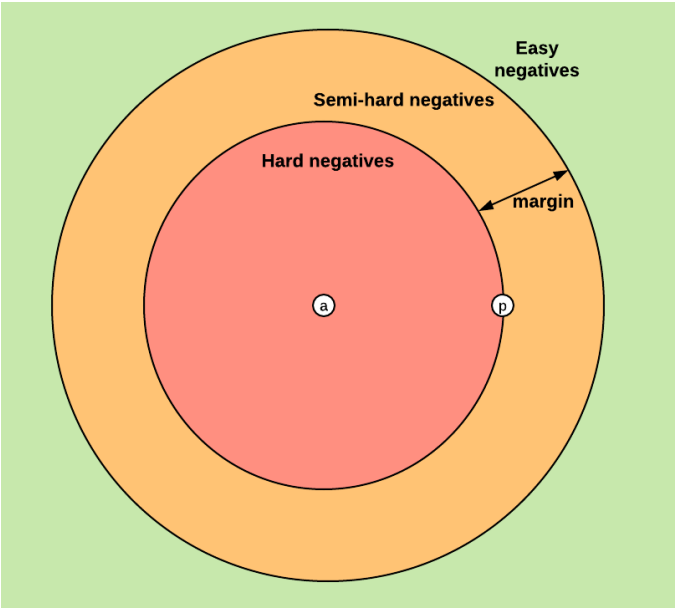


Figure 6: Easy, Hard and Semi-hard triplets visualization [33]

As shown in [23], the best results are from training using the semi-hard triplets. These triplets are selected using an online learning method because it is more efficient. The semi-hard negatives will be selected for all anchor positive pairs in a mini-batch. The semi-hard negatives are selected by creating a pairwise distance matrix of the embeddings of all images in a mini-batch. The loss value for a mini-batch is the average loss of all semi-hard triplets. The loss function for one semi-hard triplet is as follows:

$$\mathcal{L}(Anchor, Positive, Negative) = \max(||f(Anchor) - f(Positive)||_2^2 - ||f(Anchor) - f(Negative)||_2^2 + \alpha, 0)$$

The max function combined with the margin ensures that only semi-hard triplets are used for calculating the loss. The mini-batches should be chosen large enough such that enough semi-hard triplets can be selected. Small batch sizes might result in a division by zero because there are no triplets of which the average loss could be calculated.

3.2.1.1 Triplet loss example

The triplet loss function as explained above is implemented in TensorFlow and will be used to create a demonstrative example. The implementation of the triplet loss function in TensorFlow includes both selecting triplets as well as calculating the loss. TensorFlow's documentation about the triplet loss function [34] will be followed mainly for this example. The MNIST handwritten digit database will be used instead of pigs because the MNIST database is a bit less complex and contains many images. The dataset is split into 80% train and 20% test images. These images are normalised and batched in groups of 32. A relatively small network is used and consists of two convolutional layers each followed by a max-pooling and dropout layer. This network has a 256-dimensional output vector which is L2 normalised. The network is trained for five epochs using the Adam optimiser with a learning rate of 0.001. Embeddings of the test set are obtained using the trained network. Principal component analysis is used to reduce the dimensions of the test embeddings from 256 to 2, such that the embeddings can be visualised. The

PCA visualisation of the test set embeddings before and after training the network can be seen in Figure 7.

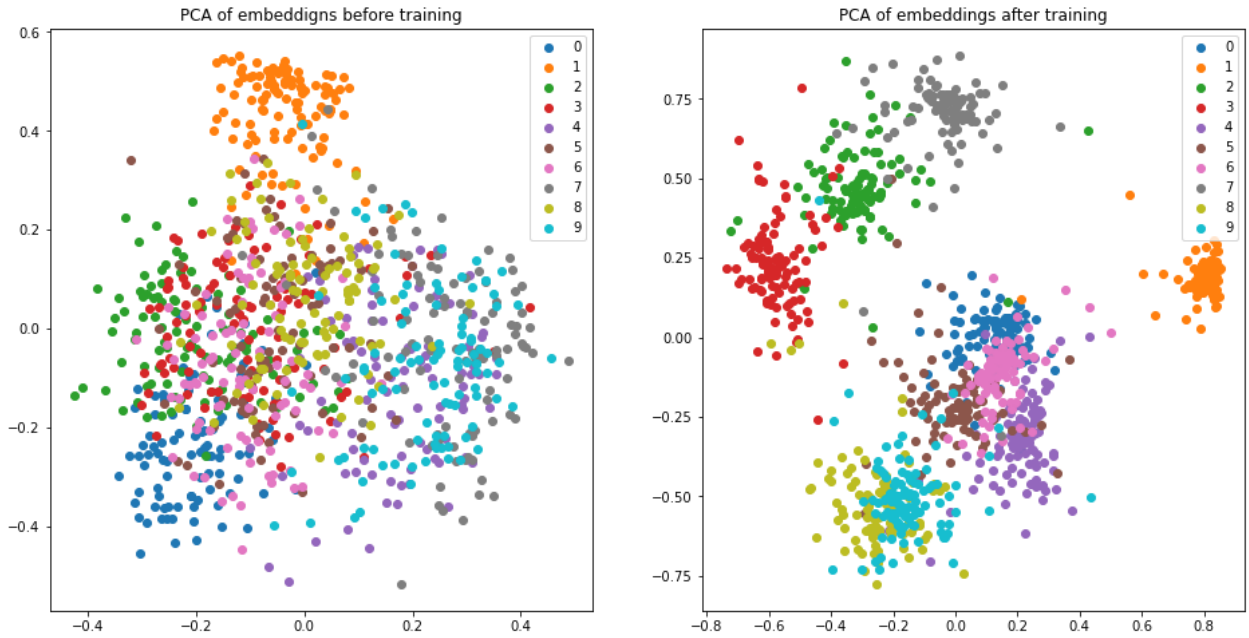


Figure 7: PCA of the test set embeddings before and after training of the network. Each colour represents a digit class.

It can be seen in Figure 7 that similar digits of the test set are grouped after training. Note that the colours for each class are not the predicted classes but their actual class and is only used to evaluate the clusters.

Although not all digits are grouped perfectly, it still does show that the triplet loss works as expected. The groups are much more distinct compared to before training. Actual performance measures are not calculated for this prototype because that is not the goal. Although the MNIST dataset is simple compared to pigs, it is expected that using the triplet loss function on pigs is feasible based on this triplet loss example.

3.2.1.2 Identification and Verification

The embeddings from the network trained using triplet loss do have to be identified or verified to assign identity to an image of a pig. The embeddings of the same class or identity are closer together than embeddings of a different class or identity. This can be used to identify or verify the images. In the facenet paper [23] k-nearest-neighour (KNN)

is proposed for identification and a threshold on the distance between two embeddings for verification. These two techniques are possible because the Euclidian distance is reduced for similar identities and increased for different identities during training. KNN will assign the identity of the closest group to an embedding of unknown size. The main advantage of using KNN as classifier is that new instances can be added and removed easily without training, as KNN does not require training. Verification can be used to verify if the images of two pigs have the same identity. This can be done by settings a threshold on the distance between the embeddings of the two images. If the distance between the two is below the threshold, the pigs in the images are of the same identity. Opposite, if the distance is above the threshold, the pigs in the images have a different identity.

3.2.2 Triplet loss on pigs

The triplet loss example showed that the triplet loss function could be used to distinguish handwritten digits from each other. Distinguishing written digits is very different from distinguishing pigs from each other. Multiple experiments are done using an existing dataset found in the literature. These experiments aim to learn how different types of data and various types of networks influence the performance. The performance is evaluated using the result of a KNN classifier. This classifier uses the output of the triplet loss network.

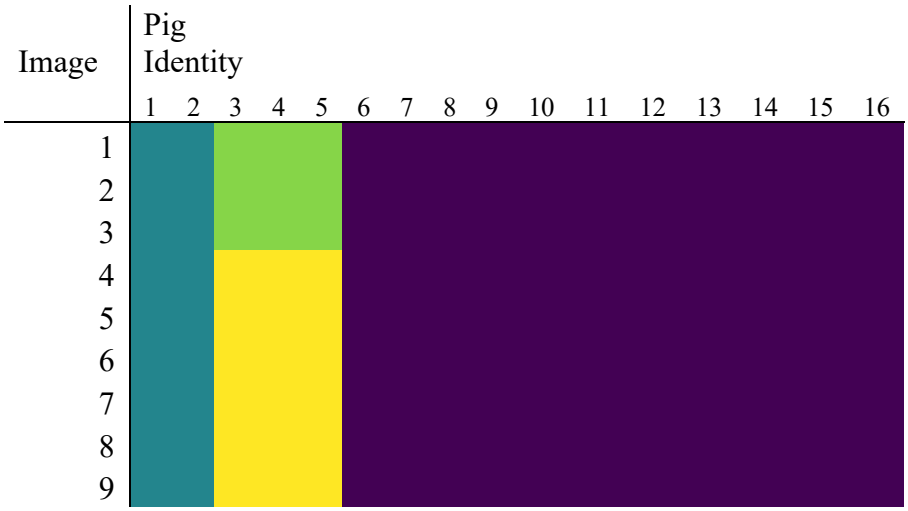
The data used for these experiments are videos from dataset 1. This dataset is used because it contains the identity of the pigs in each frame of the video. The videos of dataset 5 contain identities as well, but this dataset was not found yet during these experiments. Every 1000 frames, or 200 seconds, a snapshot of the video is made such that there is enough difference between each snapshot. The snapshots are cropped to each bounding box resulting in a separate image for every pig. The dataset is annotated using shoulder and back points from which the bounding boxes are calculated. In total, there are 9 images per pig per video.

The images of video 6 are edited to evaluate different types of data. Only the images of one video are chosen such that quick iterations can be made. Four alterations are used. Namely, the calculated cropped images, manual cropped images, colored ear tags erased, and head cropped. Manual cropped images are used because the calculated bounding boxes are spacious. The ear tags are erased to test whether the networks can learn from the colored ear tags. In previous work [17], colored ear tags are used to identify and track pigs. Lastly, the pigs' heads are cropped instead of the whole body. Not all frames are used for the head crop because the head is not always visible. An image of each of the four alterations can be seen in Figure 8.

Figure 8: Four data alterations. From left to right: calculated bounding box, manual cropped bounding box, ear tag erased, head cropped

Video 6 consists of 16 pigs, of which 9 images are used per pig. These images are split into a train, test and validation set. The train set is used for training the triplet loss network, and the validation set is used to monitor the training. The test set will again be split into a gallery and probe split. The gallery set is used to for the KNN identification of the probe set. The gallery and probe are splits are done per image while the train, test, and validation splits are done per identity. The splits can be seen in Table 2.

Table 2: Dataset split. Blue: Validation split, Green: Probe, Yellow: Gallery, Purple: Train



The experiments with different types of networks use the images of three videos. Videos 1, 6, and 11 are chosen because this includes all three age types: nursery, early finishers, and late finishers. These videos are recorded during the day, and the pigs have a high activity according to the annotations of the dataset. In total, 27 pigs are used, each with 9 images per pig. The dataset split is done similarly as the previous set. 25 pigs are used for training, 8 for the test set and 6 for the validation set. The test set is again split into a gallery and probe set.

Similar to the previous triplet loss prototype, this prototype is made in TensorFlow and Keras. The images are loaded into TensorFlow and labelled according to the images' folder. All images are resized to 96x96 pixels. This size is in the similar range as the other three studies about pig face recognition which used sizes of 64x64 [18], 96x96 [19], and 128x128 [11]. The images are normalised to the expectations of the network. The different backbone networks that will be used are DenseNet, MobileNetV2, Inception, VGG, and ResNet. The data alteration experiments are done with the network that performs best. The convolutional neural networks used are pre-trained networks on ImageNet. These networks are available in Keras. The networks are used without a fully-connected layer but use a 128-dimensional dense as the final layer, which is L2 normalised. The semihard triplet loss function is used which is available in the

TensorFlow Addons library. The loss function handles both the triplet selection as well as loss calculation.

After training, the embeddings of the gallery and probe sets are calculated. The gellery set is used to 'train' the KNN classifier. The number of neighbors used for the classification is set to 5. Evaluation will be done using the classification's accuracy, precision and recall metrics as well as the triplet loss of the train and validation set. The results of the experiments with the different networks can be seen in Table 3.

Table 3: Results of network experiments

Network	**Data**	**Triplet loss train after training**	**Triplet loss validation after training**	**KNN Accuracy**	**KNN Precision**	**KNN Recall**
DenseNet121	Calculated crop Video 1, 6, 11	0.064	0.978	0.46	0.63	0.46
MobileNetV2	Calculated crop Video 1, 6, 11	0.044	0.998	0.4	0.63	0.4
InceptionV3	Calculated crop Video 1,6, 11	0.757	0.999	0.27	0.19	0.27
VGG16	Calculated crop Video 1,6, 11	0.999	0.999	0.2	0.14	0.2
ResNet50	Calculated crop Video 1,6, 11	0.056	0.979	0.46	0.6	0.46

The results of the network experiments show that the triplet loss of the validation set is higher than the triplet loss of the train set for all the networks, except for VGG16. The triplet loss of the train set is similar to the loss for the validation set for VGG16. The accuracy and recall are the highest for DenseNet and Resnet50. However, Densenet has a slightly higher precision. VGG16 and InceptionV3 performed worse than the other three in all evaluation metrics. DenseNet and ResNet perform similarly, but DenseNet has fewer parameters than ResNet. DenseNet seems to be the best network based on these results. Wang and Liu [19] made a similar conclusion where they compared 6 networks trained using triplet loss on pigs. DenseNet121 will be used for the experiments with

image alterations. This experiment only uses the pigs from video 6. The results of the image alteration experiments can be seen in Table 4.

Table 4: Results of data alteration experiments

Network	Data	Triplet loss train after training	Triplet loss validation after training	KNN Accuracy	KNN Precision	KNN Recall
DenseNet121	Calculated crop Video 6	0.0003	0.933	0.83	0.91	0.83
DenseNet121	Manual crop Video 6	0.0003	0.932	1	1	1
DenseNet121	Ear tag erased Video 6	0.0003	0.883	0.66	0.70	0.66
DenseNet121	Head crop Video 6	0.0009	0.948	1	1	1

The KNN classification metrics of the data experiments are higher than in the network experiments. This is because less data is used for the data experiments. There are only 9 images in the probe set, which might not be enough to draw actual conclusions. The validation loss is the lowest for the dataset with ear tags erased, while it performed worst in the KNN classification. The opposite can be seen in the network experiments, where a lower validation loss results in a higher KNN calcification performance. Nevertheless, it can be observed that the manual crop and head crop performed best in the KNN classification. The pig's head is used to recognize them is used in all three previous studies [11], [18], [19]. Besides that, it is expected that the body does not contain enough biometric information and that the pigs are identified based on the mud patterns. These mud patterns can rapidly change over time. Using the pig's faces for identification seems to be the best option based on the previous work and our experiments.

3.3 Conclusion technology exploration

This technology exploration chapter aimed to identify if the technologies found in the literature are suitable for our application. The pig detection should be able to work real-time and detect all pigs in an image or video. The YOLOv4-tiny detector did not run real-time on a macbook without GPU, but should be able to run at framerate higher than 300 with the use of a GPU [30]. The detector was able to detect all pigs in an image with an mAP of 82.1%. Based on this the detector seems to be suitable for our application. The identification showed that using triplet loss could be suitable for our application. The KNN identifier can identify 9 images of pig heads with an accuracy of 100%. Although this is tested with 9 images in the same perspective and over a time span of 30 minutes, it is expected that using a triplet loss function is suitable for a real-world pig verification or identification system. A dataset with pigs' heads in different perspectives over a longer time period should be collected and used for evaluation to answer the research questions.

4. Data Collection

The literature and triplet loss experiments showed that a dataset of pigs' faces is needed for identification or verification. There are no publicly available datasets of pig faces which is why our own dataset is collected. The data is captured in two perspectives, a controlled perspective, and an overview of the shed perspective. This simulates a real-world application where pigs are enrolled in the controlled perspective and identified in the overview perspective. Eventually, the identification must be working in real-time. In this project, we will not be focusing on real-time data, but collect a dataset on which experiments can be done. This section will describe how this dataset is collected and preprocessed.

4.1 Environment

The farm where the data is collected has multiple sheds, one of which will be used to collect the data. The used barn consists of approximately 100 sows. An impression of the shed can be seen in image Figure 9.

Figure 9: Pig shed where the data is collected

The shed is installed with Nedap SowSense. This is an intelligent system for individual animal management. It can identify each sow, track her feed consumption and weight and

separate each animal if needed. A schematic of the installed SowSense components can be seen in Figure 10.

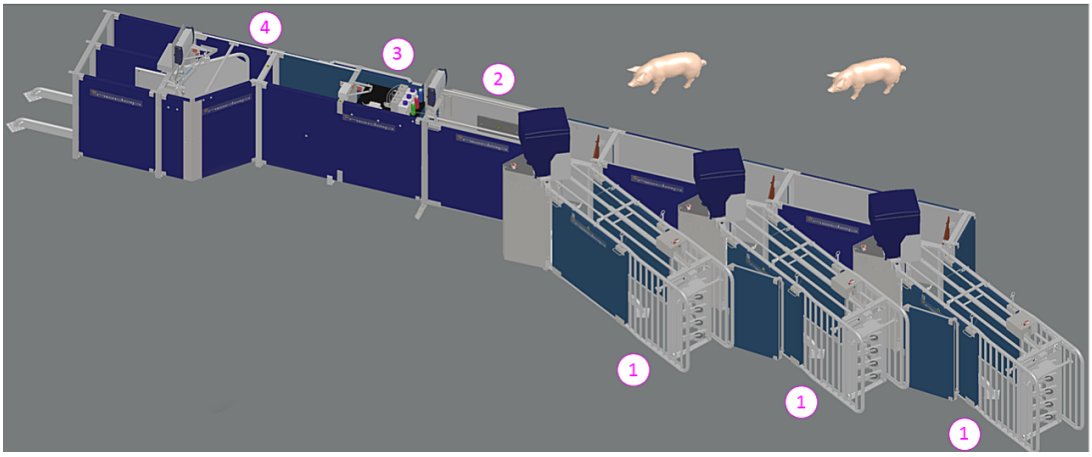


Figure 10: SowSense. 1: Electronic Sow Feeders. 2: Weight sampler. 3: Color Spray Markers. 4: Separation unit

The pigs' identity is checked multiple times when they walk through the SowSense units using an RFID tag. The identity information of the SowSense will be used to label the videos of both perspectives. The controlled perspective will be installed on the SowSense itself, and the overview perspective will be pointed towards the exit of the SowSense. This way, both perspectives use the identity information of the SowSense to annotate the images with a pig identity.

4.2 Raspberry Pi Camera

The camera used to record the pigs is a Raspberry Pi V2 camera. Although the literature showed that a security camera seems the most suitable, the pi camera is chosen for its price, availability and simplicity. The camera has a resolution of 1080x1920, but the videos are recorded in 480x640 pixels at 15fps. Reducing the resolution is done to avoid video stuttering. An 800-lumen led light is used for lighting the environment. The videos are captured only when motion is detected using MotionEye [35]. The camera's shutter speed is reduced in MotionEye’s settings to minimise motion blur.

4.2.1 Collected Videos

The data is captured for five days using the Raspberry Pi camera. A snapshot of the captured videos of both perspectives is shown in Figure 11. During the five days, 2175 videos were taken in the overview perspective, and 3776 videos were taken in the SowSense perspective. This is not the same amount of individual pigs because one visit might be split up into multiple videos, or the motion detection is wrongly triggered. The data from the weight scale shows that there were 1454 weighing events over the 5 days. There are approximately 100 sows in the shed, which mean that, on average, the pigs walk three times a day through the SowSense. The captured videos of the controlled perspective are dark despite a light is used as a result of the reduced shutter speed. Increasing the shutter speeds leads to videos with a lot of motion blur. The images of the overview perspective as much brighter as there is more light than in the SowSense.

Figure 11: Left) Video captured in controlled perspective. Right) Video captured in overview perspective

4.3 Security Camera

A security camera is used to capture brighter videos from the controlled perspective. The literature showed that security cameras are suitable for the pig shed environment. They are designed to be used outdoors and in dark environments. The security camera used is an Axis M3058-PLVE. It is a 12MP dome camera with a 360-degree view. This security camera is chosen because it is available. Only one camera is available which is why only the controlled perspective is captured using this camera. A 360-degree view is not necessary, but a partition of the 360 view can be selected to be similar to a regular lens. Digital pan-tilt-zoom controls are used to select a view that is automatically rectified. The camera is set up to record infrared videos, resulting in brighter and cleared videos.

4.3.1 Collected Videos

The data is captured for three days from 30-04-2022 until 02-05-2022. The continuous recording is split up into videos of 10 seconds. The resolution of the videos is 2048x1536 at a frame rate of 10 fps. Videos of 105 individual pigs are captured during the three days. Besides the three days of recording, there is another day of recordings 18 days later on 20-05-2022. A snapshot of one of the captured videos is shown in Figure 12. The videos of the security camera are much brighter than the raspberry pi camera. The downside of the security camera is that it has a larger lens which collects more dust. This dust results in blurry videos.

Figure 12: Video captured in controlled perspective using the security camera

4.4 Video Preprocessing

The videos have to be annotated first, and the pig's head has to be cropped from each frame in the video before it can be used for identification. This section will explain how the videos are cleaned, labelled with the pig's identity and cropped in each frame.

4.4.1 Labeling raspberry pi videos controlled perspective

The weighing data includes the identity of the weight pig and the time of weighing, which can be used to label the videos. The weight data is first cleaned by removing the events where multiple pigs walk closely behind each other. These events have multiple pig identities for one weighing event. This is less than 10% of all weighing data. This avoids selecting videos with multiple pigs visible in one video. After that, the videos are matched to a weight event by checking if there is a weight event between the start and end of the video. If there is only one weight event during the video, the identity for the weight event is matched to the video. This is done by moving all videos of one identity into one folder, which is named to the pig's identity. There are 1331 videos of 105 unique pigs from the controlled perspective. The videos are captured from 07-04-2022 until 12-04-2022.

The videos of one day between 8:00 and 20:00 are manually checked to see if they are correctly matched to a weighing event. There were 179 weighing events between this time, of which 133 videos could be matched correctly. Most of the videos that are not matched are because the videos started when the pig was already on the scale and thus could not be matched to a weight event. There are no videos that are incorrectly matched to an identity.

4.4.2 Labeling raspberry pi videos overview perspective

Ideally, the pigs of the overview perspective would be grouped similarly to the controlled perspective. This would be possible if the pigs walked out of the SowSense immediately after they stepped on the weight scale. However, the pigs stay in the SowSense for some time and come out of it in groups. MotionEye detects multiple pigs exiting as one motion event. This results in a video where multiple pigs are exiting the SowSense. The

automatic video and weight event matching do not work because there are multiple pigs in one video. The videos are manually grouped by identity; if needed the videos are split into multiple videos. The videos are matched to an identity by using the weight information. The number of pigs walking through the SowSense between two breaks of at least 10 minutes is counted in the weighing data. The pigs are matched to these weighing events if the number of pigs walking out of the SowSense in the videos is the same. The videos are discarded if this number is not the same. Only the videos of one day are labelled because this is a tedious task. In total, 196 videos of 77 unique pigs are labelled.

4.4.3 Labeling security camera videos

The data preprocessing is similar to the previous dataset. The difference is that the video of this dataset is recorded continuously. The videos can be matched to a weight event using the start and end times of that weight event. This is possible because the recording is split into videos of 10 seconds. Weight events with more than one pig on the scale or where a pig steps on the scale before the previous video is ended are not used. In total, there are 814 weight events used over the three days. The videos taken 18 days later contain 240 weighing events.

4.4.4 Cropping head in videos

The head of the pigs must be cropped from the videos so they can later be used for identification. An example of the cropped heads is shown in Figure 13. Cropping the videos of the close-by perspective is done using a yolov4 tiny object detector. This object detection is trained on snapshots of the dataset itself. 253 images are manually annotated with bounding boxes around the pig's head, which are used to train the yolov4 tiny object detector. The bounding boxes do not include the pig's ears. The object detector detects the pig's head in each frame and saves this as an image. The raspberry pi controlled perspective dataset consists of 111.968 images of 105 individual pigs. The security camera dataset contains 25.097 images. Cropping the head from the overview perspective is done manually although the initial idea was to automatically detect the pigs using the detector of the previous chapter. Pigs are in front of each other regularly, making it hard for object detection and tracking to know which pig is which. In the tracking experiments, it could be seen that identity swapping happens when pigs get close to each other. To prevent this, the heads are cropped manually. In total, 1052 images of pig faces are collected from 77 individual pigs.

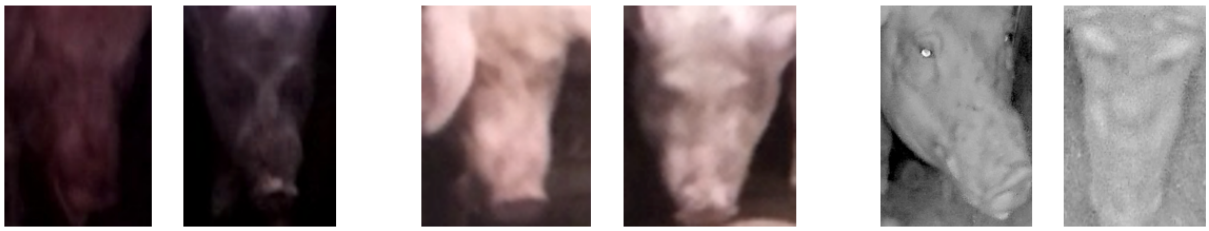


Figure 13: Left Two) Controlled perspective rpi. Middle Two) Overview Perspective. Right Two) Overview perspective security camera

5. Method

The evaluation methods are explained in this chapter. Each of the research questions will be answered using an experiment. The different experiments compare with existing methods, identifying and verifying pigs not trained on, identifying and verifying pigs on later data, and identifying and verifying pigs in a different perspective. The experiments are explained in detail below and the network used for these experiments.

The network used in the experiments is densenet121, which showed the best results in the previous chapter. After the output of the embedding, extra layers are added, which are used for a SoftMax classification, as shown in Figure 14. These layers can be used for multitask learning which makes the network converge faster [19] and can be used in the experiment that compares our method to the existing. The extra layers after the embedding output are removed after training for the other experiment.

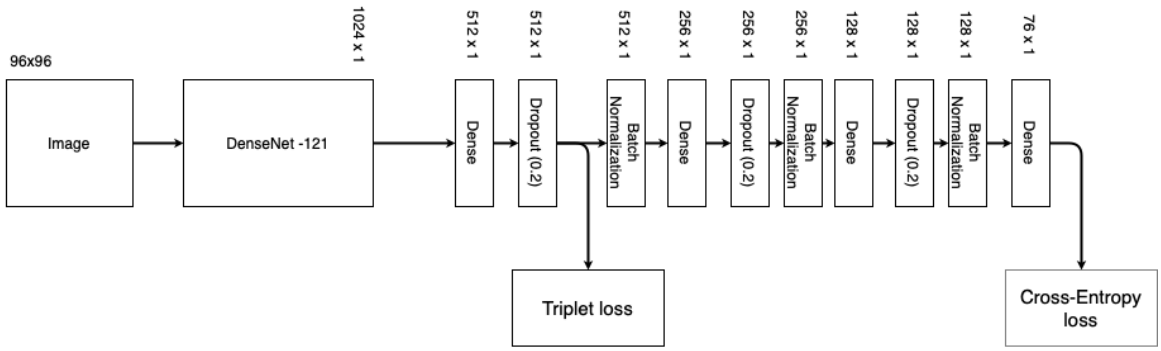


Figure 14: Visualization of embedding network

Except for the experiment that compares our method to the existing ones, the network is trained on different pigs than tested. After training the network using triplet loss and the cross-entropy loss, pigs are enrolled in the system. Embeddings of the pig's images, including their identity, are stored in the gallery. The embeddings with their identity will be used to identify or verify new images. This process is visualized in Figure 15.

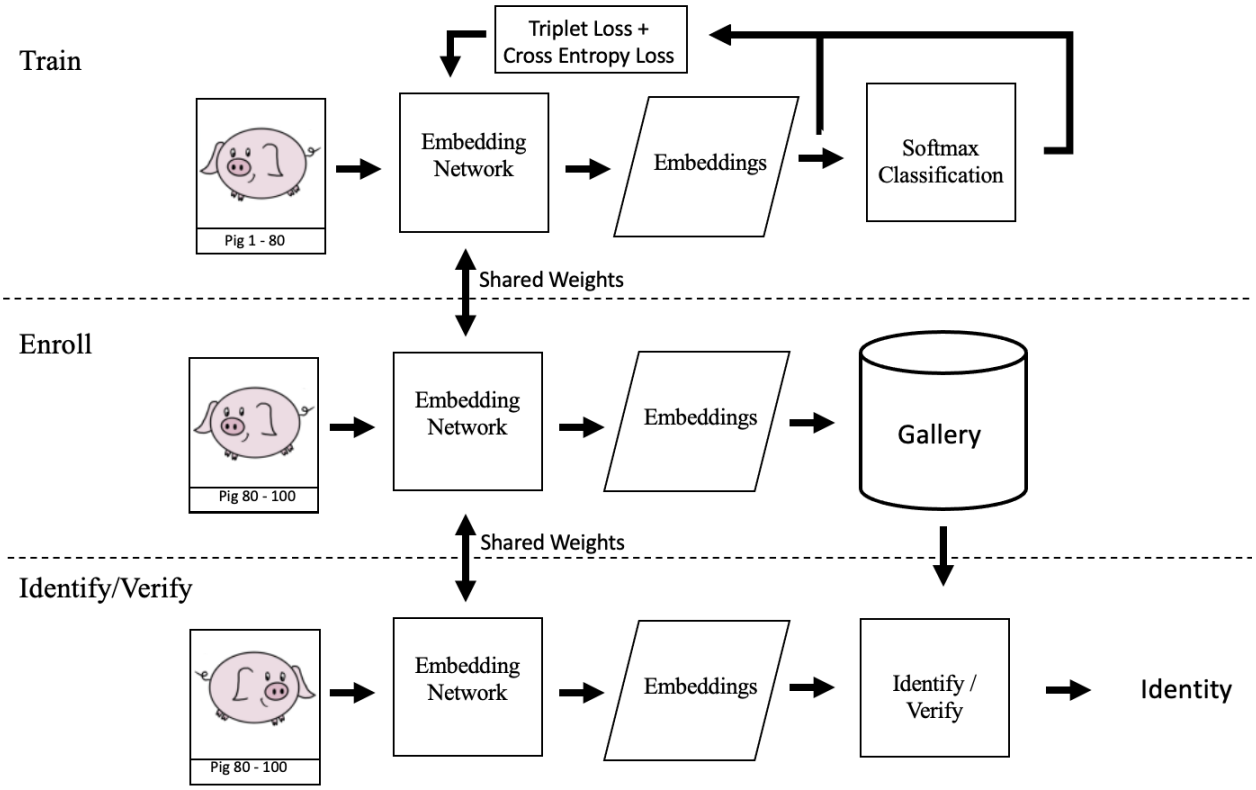


Figure 15: Training, enrolling and identifying/verifying pigs using the embedding network

The fact that embeddings are trained such that the Euclidian distance is small between embeddings of similar identities and large between embeddings of different identities will be used to identify or verify the images. Identification is done using k-nearest-neighbour because this uses the distance between embeddings. Besides that, the main advantage of using k-nearest-neighbour as classifier is that is does not have to be trained again if new identifies are added or removed from the gallery. Verification will also be using the Euclidian distance between two embeddings. A threshold will be determined by making all combinations of the gallery set. If the distance between two embeddings is below the threshold, they are verified to be the same. If the distance is above the threshold they are not verified. Embeddings with unknown identity are compared to the embeddings in the gallery. The embeddings that are compared with the gallery is called the probe set.

5.1 Experiment 1: Comparison with existing methods

This experiment will compare our network and data to the existing methods found in the literature. This test will use the same number of pigs during training as the existing methods and compare their classification method to ours. The datasets used in previous work are unavailable, which is why our dataset is used. Two experiments with different classification methods will be used during this test. The first method reproduces the state of the art as best as possible, and the second is our proposed method. This will show how KNN and distance threshold compares to the SoftMax classification, as seen in the literature. Besides that, a comparison between our network and data can be made to the ones in the literature. This test answers the following research question.

"How does our identification method compare to the existing methods?"

The first experiment uses a SoftMax classification to identify the pigs using the cross-entropy loss. This is similar to the two methods in the literature [18] [11]. The second experiment uses KNN for identification and distance threshold for verification. The following parameters are used for training the network.

- Epochs: 35
- Learning rate: 0.001
- Optimiser: Adam

Ten pigs are used in both studies, and the networks have a network output for each pig. This test is also done using ten pigs from the security camera dataset as the literature datasets are unavailable. The ten pigs with the most images are used for this test, and the average number of images per pig is 116. Consecutive frames are checked on similarity using the structural-similarity index measure (SSIM) and removed if they are too similar. Testing is done using different images of the same pigs as in the training set similar to [18] [11]. The images of the ten pigs are split into 80% train images and 20% test images for the SoftMax classifier. The KNN and verification split the test set up again in 80% gallery and 20% probe sets. A 5 fold cross-validation is used. The identification is done such that every image in the probe set is identified. All combinations between the probe

set and gallery set are made for verification. The evaluation metrics used for the SoftMax and KNN identification are the accuracy, precision, and recall. The distance threshold verification is evaluated visually using a kernel density estimation (KDE) plot on the distances of similar and different identities. A threshold is chosen from a ROC curve where the difference between the true positive rate and false positive rate is highest. This is considered the optimal threshold. Using this threshold the F1 score, precision and recall are calculated on the combinations between the probe set and gallery set.

5.2 Experiment 2: Identify and verify pigs not trained on

In this test, the identification method is trained on a different set of pigs than tested. The following research question is answered with this test.

"How well can our identification method identify pigs not seen during training?"

This experiment evaluates how the identification method performs in identifying pigs not seen before during training of the embedding network. Two experiments are done: identification using KNN and verification with distance threshold. Identification considers the problem "which pig is this?" while verification answers "are these two images of pigs the same?". The identification experiment is done using a fixed gallery size and a variable gallery size. This is done to evaluate the influence of the gallery size on the performance. The following parameters are used for training the embedding network.

- Epochs: 3
- Learning rate: 0.001
- Optimizer: Adam

The security camera dataset is used for this experiment where similar frames are removed. The data splits are shown in *Table 5*. Every image in the probe set will be identified using KNN. All combinations between the probe set and gallery set are made for the verification experiment, and these combinations are verified.

The dataset is split into 80% train and 20% test using 5-fold cross-validation. The test set is split up again into a gallery set and probe set. The experiment with fixed size uses an 80%-20% gallery and probe split—the test with variable size from 1%-99% to 99%-1% splits.

Table 5: Data splits pigs not trained on test

Experiment	Train set	Test set	Gallery set	Probe set
Experiment 2.1 KNN identification fixed gallery size	80% of security dataset	20% of security dataset	80% of test set	20% of test set
KNN identification variable gallery size	80% of security dataset	20% of security dataset	Varying 1% - 99% of test set	Varying 99% - 1% of test set
Experiment 2.2 Verification using fixed gallery size	80% of security dataset	20% of security dataset	80% of test set	20% of test set

The KNN identification is evaluated using the accuracy, precision and recall for the fixed gallery size. The experiment with variable gallery size will be evaluated with an interval of 1 percent using the accuracy. The verification experiment is evaluated visually using the KDE plot. The F1 score, recall and precision are calculated for the optimal threshold.

5.3 Experiment 3: Identify and verify pigs in later data

In this experiment, the images in the gallery are older than the images in the probe set. This test aims to evaluate how well the identification system can still identify the pigs after a certain amount of time. The pigs' appearance might change due to ageing or being covered in dirt. The following research question is answered with this test.

"How does our identification method perform on newer data?"

This experiment evaluates how well the identification method can identify pigs 18 days after they are added to the gallery. The datasets used are the security camera dataset, and the security camera dataset collected 18 days later. These will be referred to as the old dataset and the new dataset. The splits of the dataset for both experiments are shown in Table 6. The embedding network is trained on 80% of the identities in the old and new datasets combined. The other 20% is used for the gallery and probe set. A 5-fold cross-validation is used for the train and test splits. The following parameters are used for training the embedding network.

- Epochs: 3
- Learning rate: 0.001
- Optimiser: Adam

Two experiments are done. In the first experiment, images from the old set are added to the gallery. In the second experiment, the images from both the old and new datasets are added to the gallery. Both experiments use images of the new dataset as probe set. Both experiments are done with KNN identification and verification using a distance threshold. All combinations between the probe set and gallery set are made for verification. The identification is evaluated using the accuracy, precision, and recall. The verification is visually evaluated using KDE plots. The optimal threshold is calculated from a ROC curve and this threshold is used to calculate the F1 score, precision and recall. The evaluation metrics are calculated for each fold.

Table 6: Dataset splits for test on later data

Experiment	Train set	Test set	Gallery	Probe
Experiment 3.1 Old dataset gallery. New dataset probe	80% of combined old and new security dataset	20% of combined old and new dataset	Only old images of test set	Only new images of test set
Experiment 3.2 New and old dataset gallery. New dataset probe	80% of combined old and new security dataset	20% of new dataset	80% of test set	20% of test set

5.4 Experiment 4: Identify and verify pigs in overview perspective

This test will evaluate how well pigs can be identified from another perspective. This test simulates adding pigs to the gallery in a controlled perspective where their identity is known and later identified in an overview perspective. The following research question is answered with this test.

"How does our identification method perform in other camera perspectives?"

This experiment evaluates how well the identification method can identify pigs in another perspective than used for adding them to the gallery. Both perspectives are captured using the raspberry pi are used for this. Two different experiments are done for which the data splits are shown in Table 7. For both experiments, the embedding network is trained on 80% of the combined perspectives. The following parameters are used for training the embedding network.

- Epochs: 3
- Learning rate: 0.001
- Optimizer: Adam

In the first experiment the images of the test set captured in the controlled perspective are added to the gallery. In the second experiment, the images of both perspectives are added to the gallery. The probe set contains images captured in the overview perspective. KNN identification and verification with a threshold on distance is done for both experiments. For the verification, all combinations between the probe set and gallery set are made. The identification is evaluated using the accuracy, precision, and recall. The verification is visually evaluated using KDE plots. The optimal threshold is calculated from a ROC curve and this threshold is used to calculate the F1 score, precision and recall. The evaluation metrics are calculated for each fold.

Table 7: Datasplits of identifying pigs in overview perspective experiment

Experiment	**Train**	**Test**	**Gallery**	**Probe**
'Train' KNN on controller perspective, test on overview perspective	80% of combined controlled and overview perspective.	20% of combined controlled and overview perspective.	Only Controlled perspective of test set	Overview perspective of test set
'Train' KNN on overview perspective Test KNN on overview perspective	80% of combined controlled and overview perspective.	20% of combined controlled and overview perspective.	80% of test set	20% of test set

6. Results

The result of the experiments as described in the method chapter are shown in this chapter.

6.1 Experiment 1: Compare with the existing methods

This experiment aims to compare our method to the existing pig identification methods as found in the literature. This is done by comparing the SoftMax classification with KNN identification and distance threshold verification of the embeddings. This is done using 10 pigs.

6.1.2 Results Standard Classification Network

The average accuracy, average precision and average recall of the five folds of the standard classification network can be seen in Figure 16. The average accuracy is 84.4% ±7.7%, the average precision 83.6%±9.2%, and the average recall is 79.8%±13.0%.

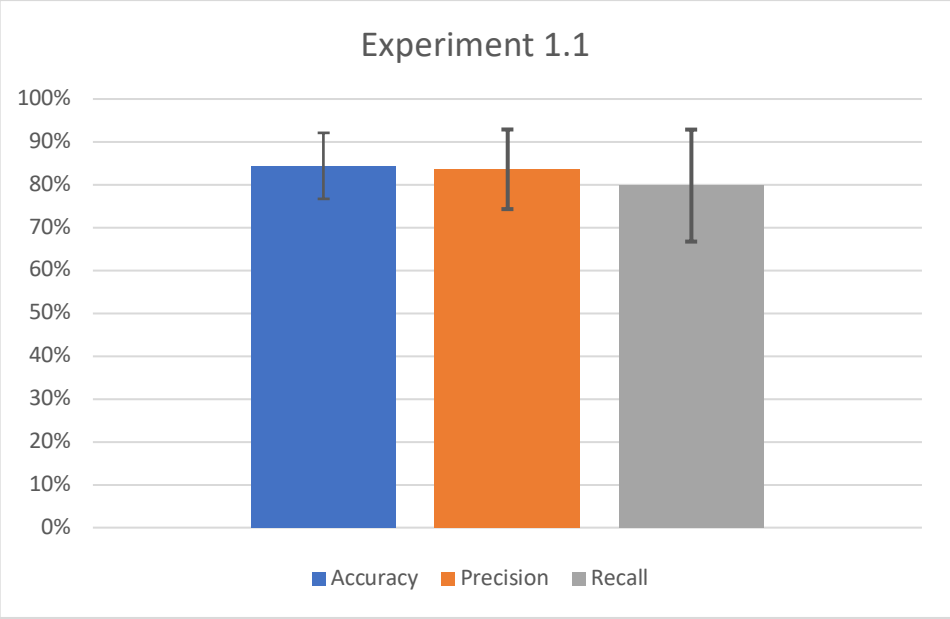


Figure 16: Average Accuracy, Precision and Recall of the standard classification experiment

6.1.2 Results Embedding Network with KNN Classification

The average accuracy, precision and recall of the five fold of the embedding network after KNN classification can be seen in Figure 17. The average accuracy is 87.8%±7.0%, the average precision 89.8%±5.4%, and the average recall is 89.8%±6.2%,

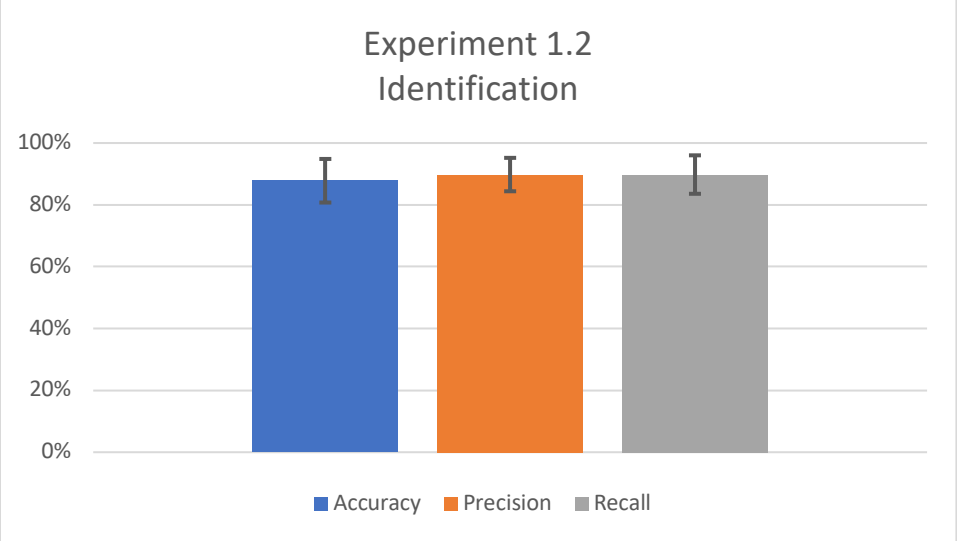


Figure 17: Average Accuracy, Precision and Recall of the embedding network with KNN experiment

6.1.3 Results Embedding Network with Distance Threshold Verification

The KDE plot of the Euclidian distances between similar and different identities is shown in Figure 18 The average optimal threshold of the five folds is 1.25. At this threshold, the F1 score, precision and recall are calculated for all probe gallery combinations which are shown in Figure 19. The average F1 score, precision and recall is 65.2%±12.5%, 55.0%±15.4%, and 83.0%±8.6%,, respectively.

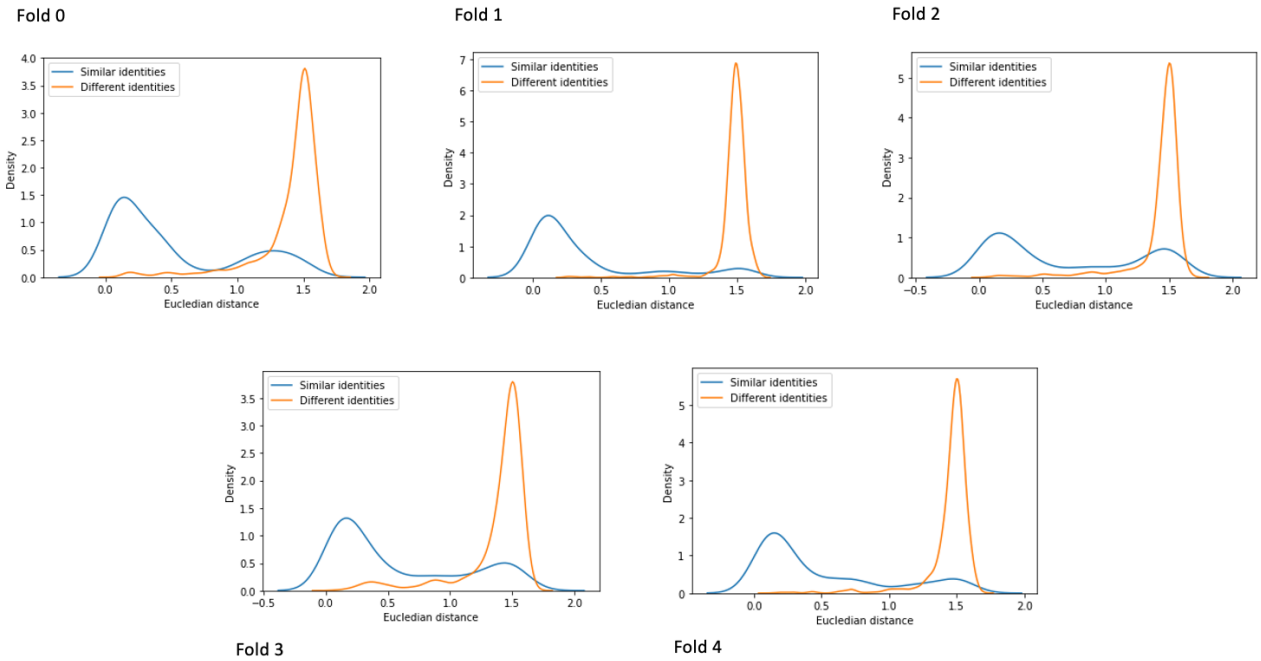


Figure 18: KDE plot for each fold of compare with existing experiment

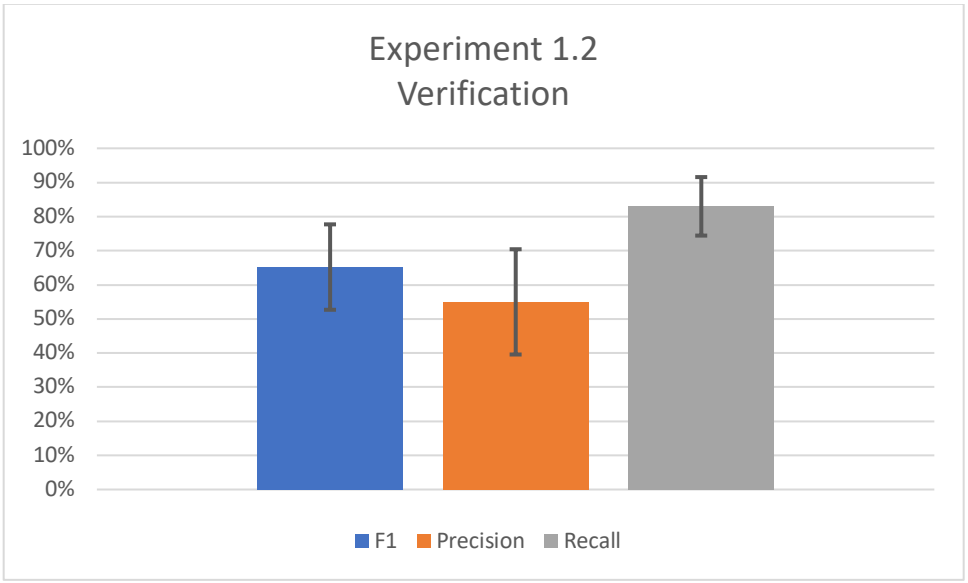


Figure 19: F1 score, Precision and recall for verification of comparison with existing experiment

6.2 Experiment 2: Identify pigs not trained on

This experiment evaluates how well pigs can be identified and verified that are not seen during training. The security dataset in the controlled perspective is used for this experiment. Identification is done with both a fixed and varying Gallery size.

6.2.1 Results Identifying Pigs Not Trained On

The accuracy, average precision and recall for each of the 5 folds are shown in Figure 20. The average accuracy of the five folds is 55.5%±4.8%, the average precision is 48.6% ±4.2%, and the average recall is 47.4%±5.6%.

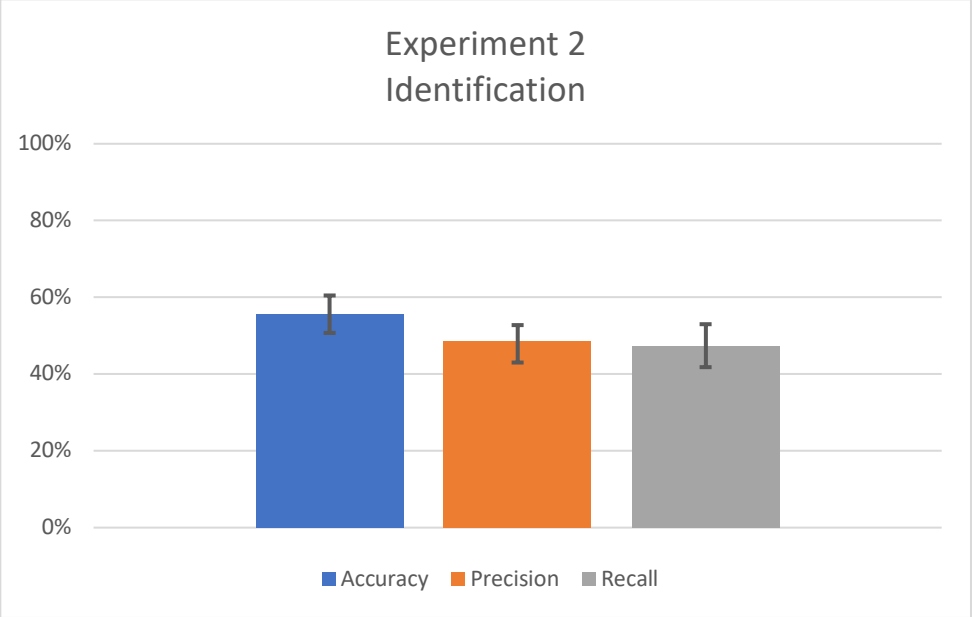


Figure 20: Average Accuracy, Precision and recall of identification experiment of pigs not trained on

6.2.1.2 Results Varying Gallery Size

The accuracy for each probe and gallery split can be seen in Figure 21. The x-axis shows the percentage of the test set used as gallery.

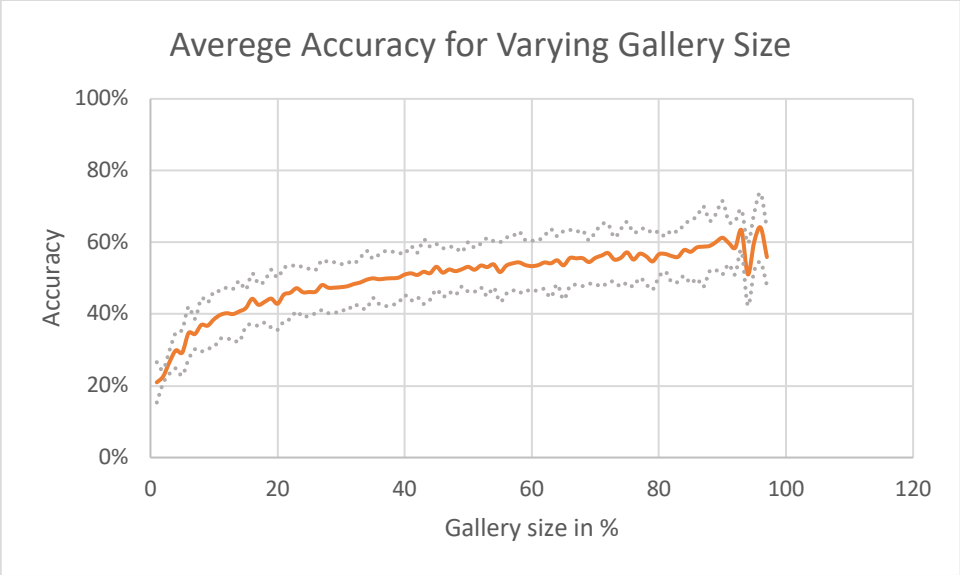


Figure 21: Accuracy of varying gallery size for each fold

6.2.2 Results Verifying Pigs Not Trained On

The probability density of the distances between similar and different identities for each fold is shown in Figure 22The average optimal threshold is 1.09, at which the F1 score precision and recall are calculated for each fold and are shown in Figure 23. The average F1 score, precision and recall are 21.0%±3.9%, 12.8%±3.3%, and 52.0%±1.4%.

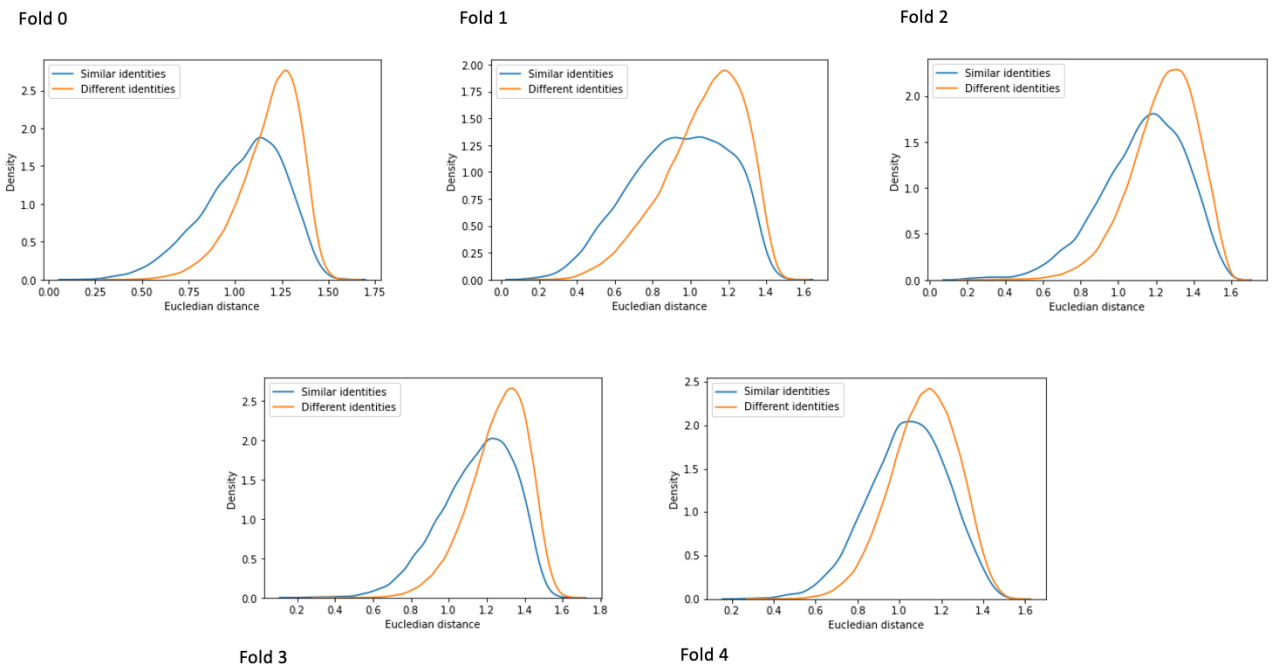


Figure 22: KDE plot of each experiment for pigs not trained on experiment

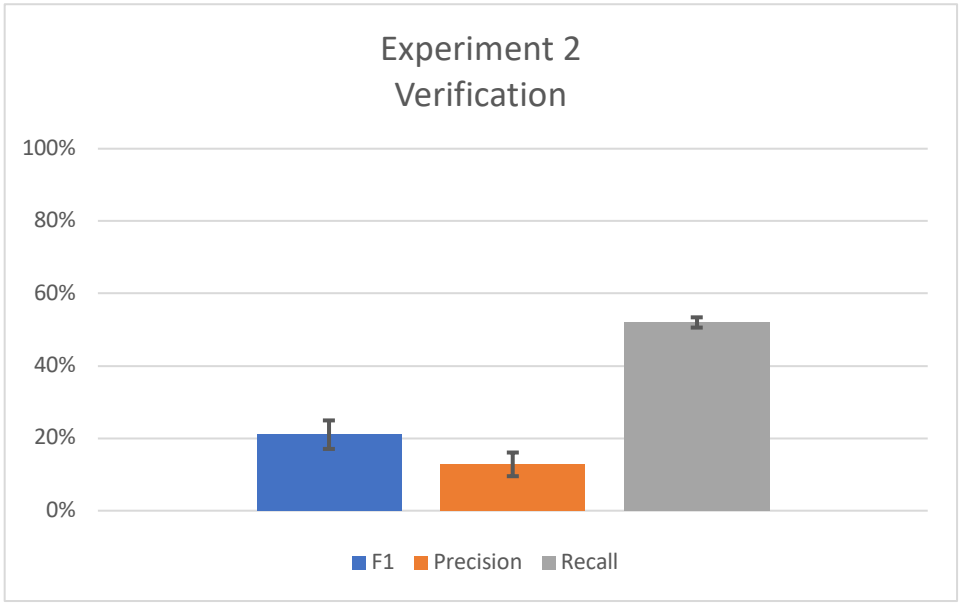


Figure 23: Average Verification F1 score, precision and recall of pigs not trained on experiment

6.3 Experiment 3: Identify pigs in later data

This experiment evaluates how well the identification and verification work using later data. Two experiments are done. The first experiment only has old data stored in the gallery. The second experiment has both old and new data stored in the gallery. In both experiments, new data is used as probe set. The controlled perspective dataset captured using the security camera is used for this experiment.

6.3.1 Results experiment 3.1 later - identify

In Experiment 1, data from the old dataset is used to 'train' the KNN, and the new dataset is used to test. The accuracy, precision, and recall for each fold are shown in Figure 24. The average accuracy, precision and recall over the five folds are 17.6%±1.7%, 13.8±1.6%, and 15.2%±1.6%, respectively.

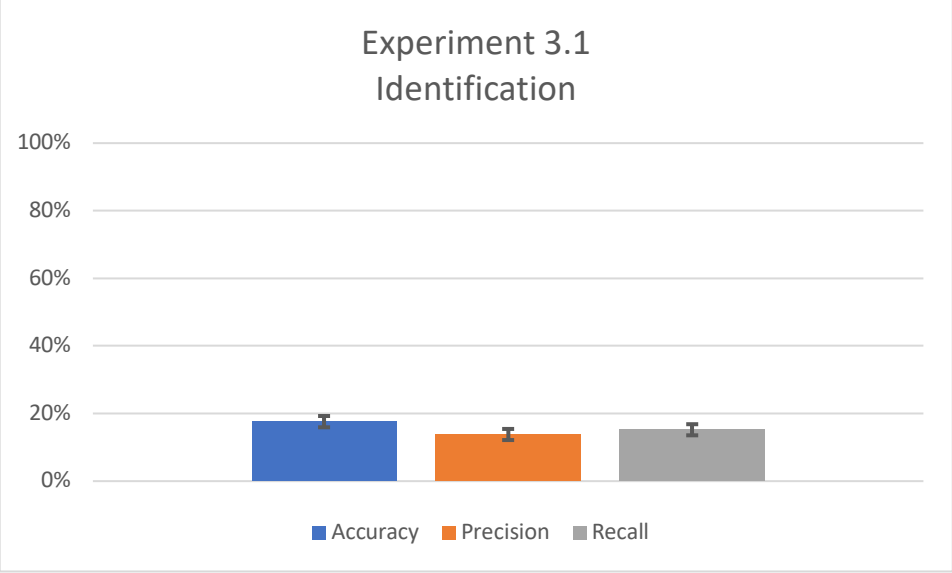


Figure 24:Average Accuracy, precision and recall of experiment 3.1 of the later data test

6.3.1 Results experiment 3.1 later - verify

The KDE plot of the distances between similar and different identities for each fold is shown in Figure 25 The average optimal threshold is 1,11 at which the F1 score precision and recall are calculated for each fold and are shown in Figure 26. The average F1 score, precision and recall are 16.6%±1.8%, 10.4%±1.1%, and 45.6%±7.9%.

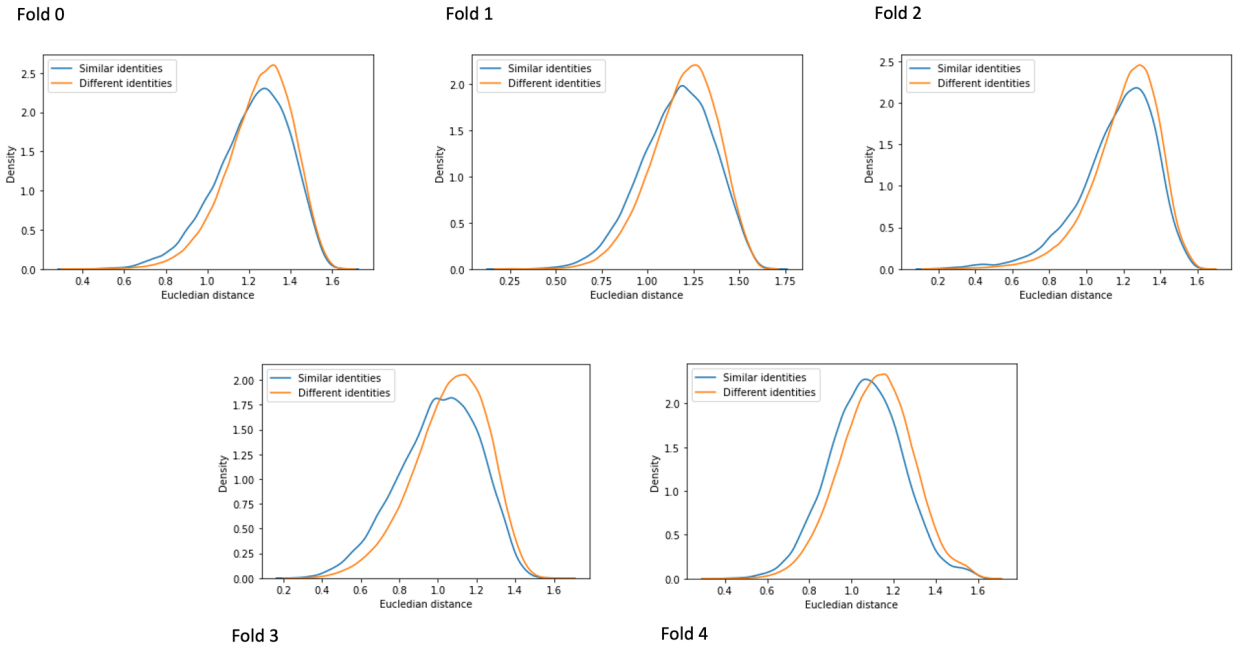


Figure 25: KDE plot for each fold of later experiment 1

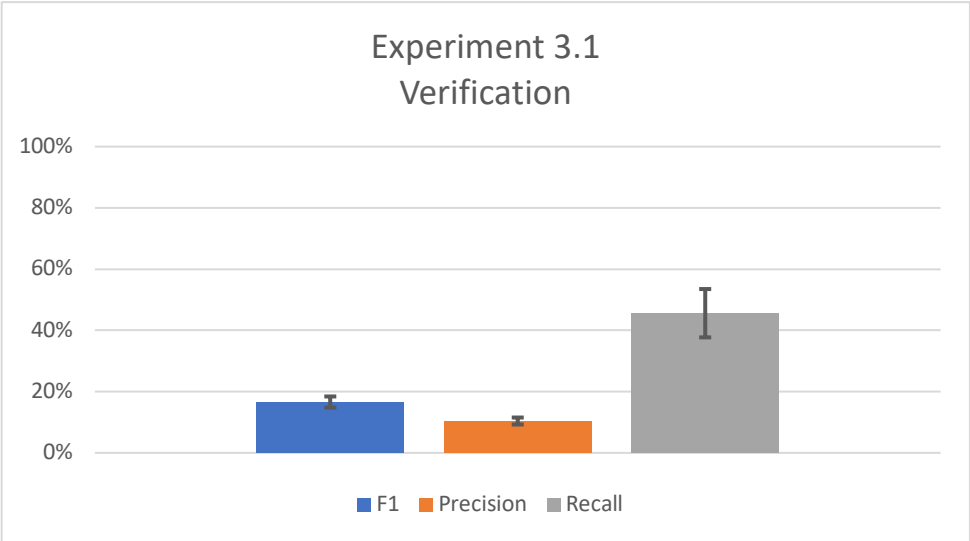


Figure 26: Average Verification F1 score, precision and recall of later experiment 3.1

6.3.2 Results experiment 3.2 later - Identify

In Experiment 2, data from the new dataset is used to 'train' and test the KNN. The accuracy, precision, and recall for each fold are shown in Figure 27. The average accuracy, precision and recall over the five folds are 58.0%±9.6%, 51.6%±8.0%, and 49.0%±5.7%, respectively.

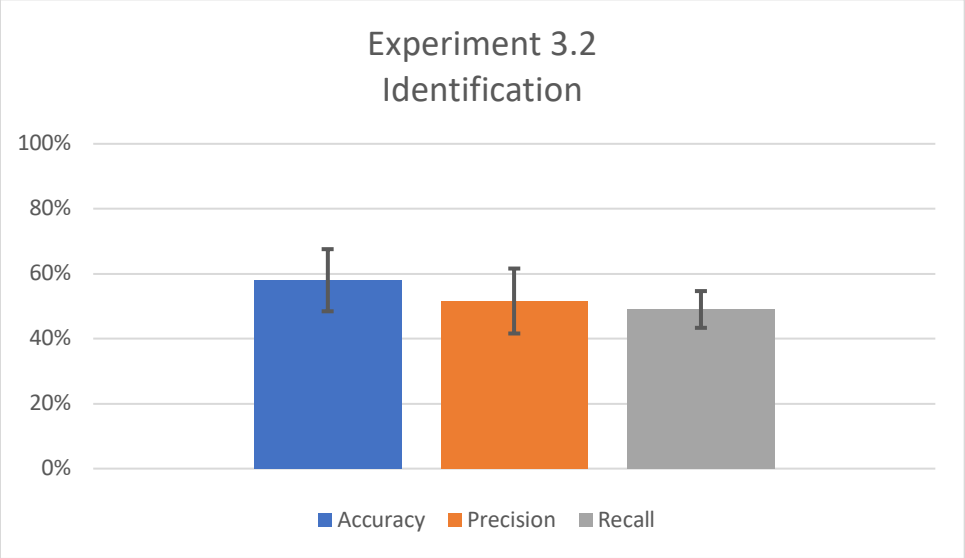


Figure 27:Average Accuracy, precision and recall of experiment 2 of the later data test

6.3.3 Results experiment 3.2 later – Verify

The KDE plot of the distances between similar and different identities for each fold is shown in Figure 28. The average threshold is 1.05 at which the F1 score precision and recall are calculated for each fold and are shown in Figure 29. The average F1 score, precision and recall are 25.4%±1.7%, 16.8%±1.8%, and 56.2±8.7%.

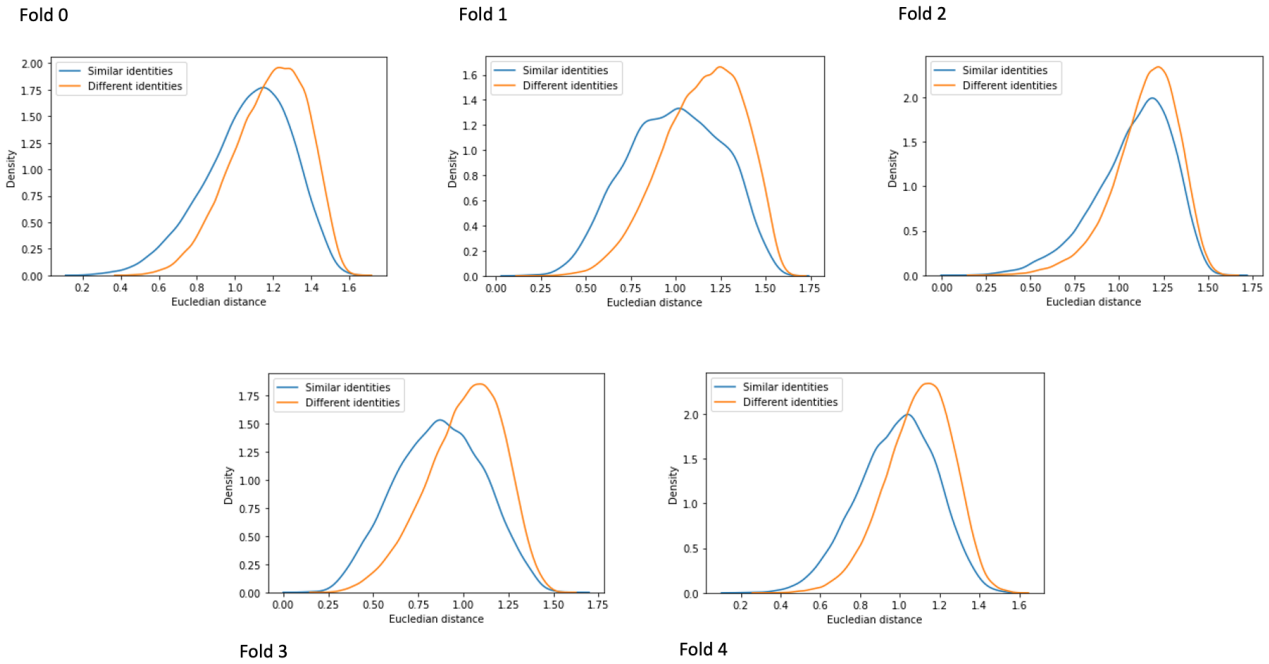


Figure 28: KDE plot for each fold of later experiment 2

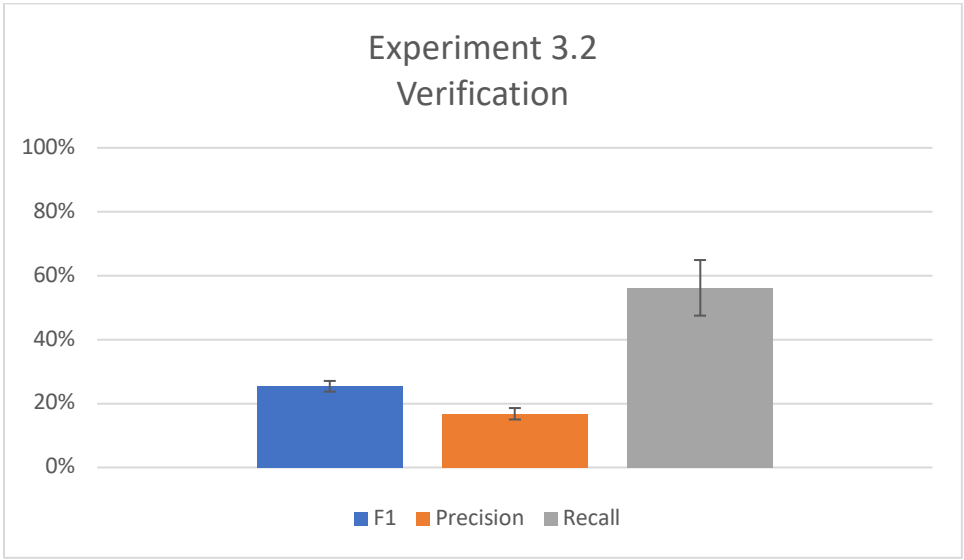


Figure 29: Average Verification F1 score, precision and recall of later experiment 2

6.4 Experiment 4: Identify pigs from in overview perspective

This experiment aims to evaluate how well pigs can be identified and verified in another perspective than the perspective in which they are added to the gallery. Only pigs captured in the controlled perspective are added to the gallery in the first experiment. In the second experiment, the controlled and overview perspectives are added to the gallery. The overview perspective is used as probe set for both experiments.

6.4.1 Results experiment 4.1 overview perspective - identify

In the first experiment, data from the controlled perspective is used to 'train' the KNN and data from the overview perspective are used for testing the classifier. The accuracy, precision and recall of each fold are shown in Figure 30. The average accuracy, precision and recall over the five folds are 19.8%±2.6%, 19.2%±5.2%, and 19.6%±4.8%, respectively.

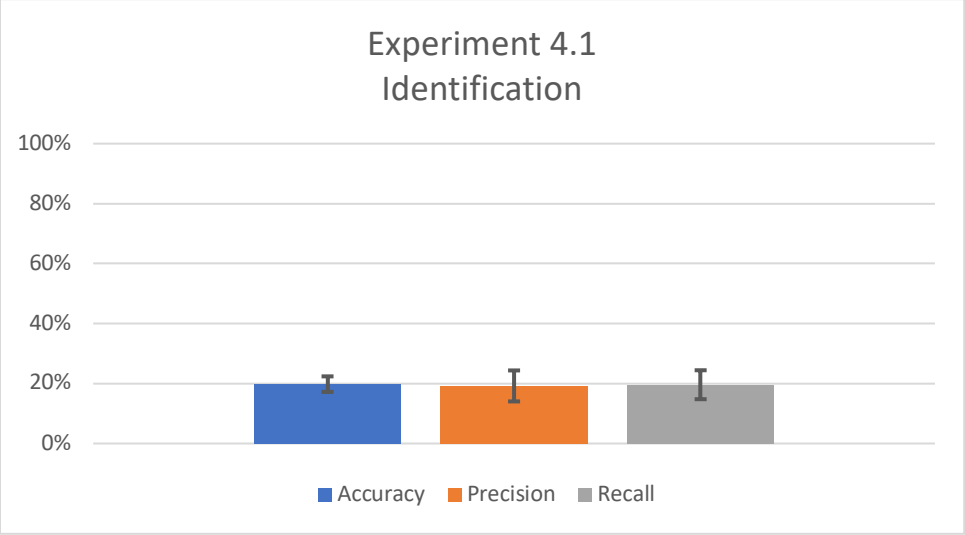


Figure 30: Average Accuracy, precision and recall of experiment 1 of different perspective test

6.4.2 Results experiment 4.1 overview perspective – verify

The KDE plot of the distances between similar and different identities for each fold is shown in Figure 31. The average threshold is 1.11 at which the F1 score precision and recall are calculated for each fold and are shown in Figure 32. The average F1 score, precision and recall are 15.8%±0.8%, 9.6%±0.5%, and 49.2%±9.0%.

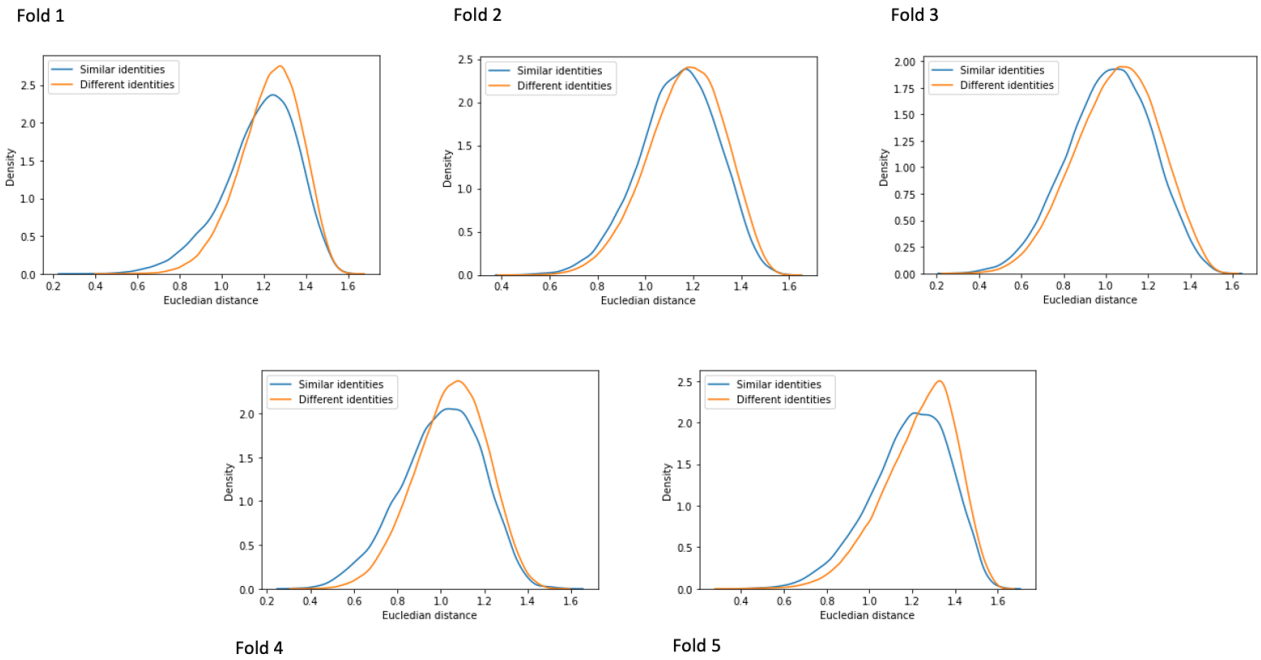


Figure 31: KDE plot of each fold of overview perspective experiment 1

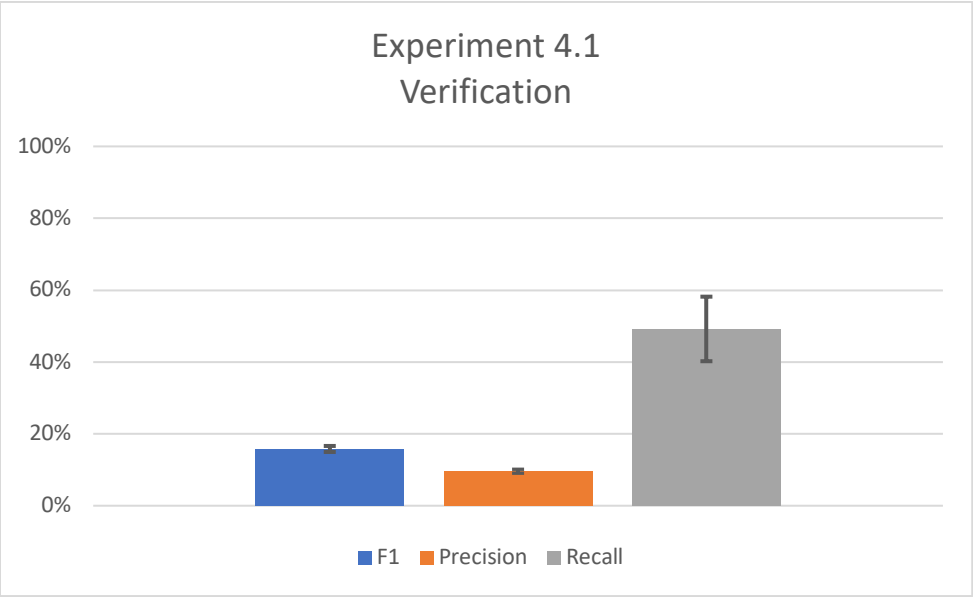


Figure 32: Average Verification F1 score, precision and recall of overview perspective experiment 4.1

6.4.3 Results Experiment 4.2 overview perspective - Identify

In the second experiment, both perspectives are added to the gallery. The accuracy, precision and recall of each fold are shown in Figure 33. The average accuracy, precision and recall over the five folds are 61.4%±8.4%, 64.0%±9.3%, and 62.4%±11.4%, respectively.

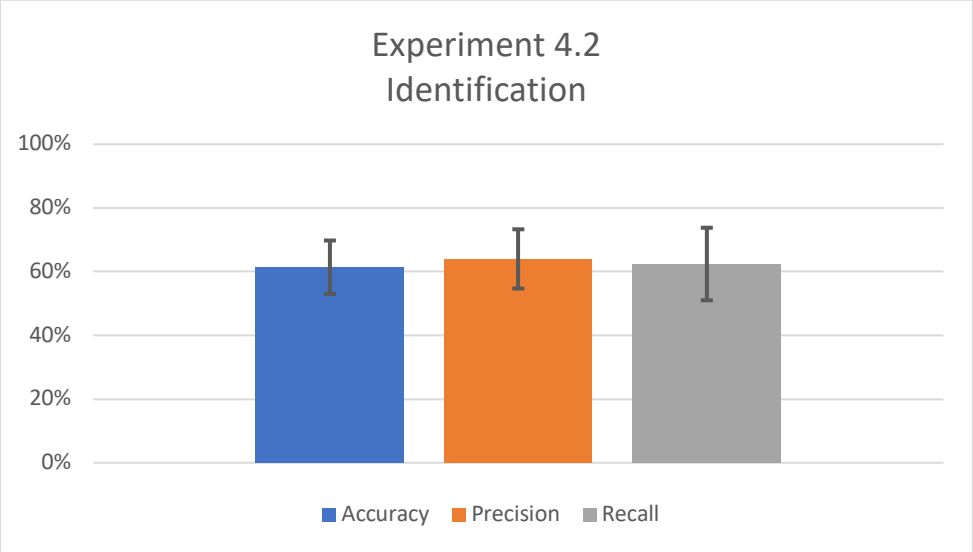


Figure 33: Average Accuracy, precision and recall of experiment 4.2

6.4.4 Results Experiment 4.2 overview perspective – Verify

The KDE plot of the distances between similar and different identities for each fold is shown in Figure 34 The average threshold is 0.91 at which the F1 score precision and recall are calculated for each fold and are shown in Figure 35. The average F1 score, precision and recall are 25.0±1.4%, 16.6%±1.8%, and 56.8%±10.3%.

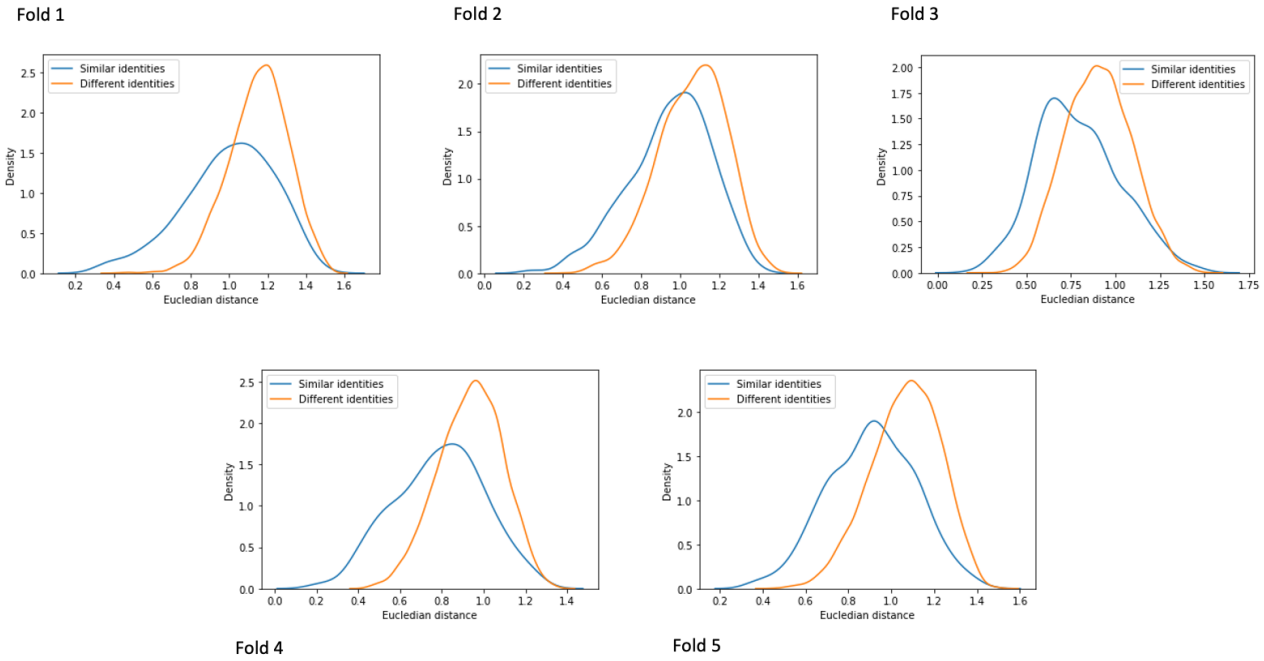


Figure 34: KDE plot for each fold of overview perspective experiment 4.2

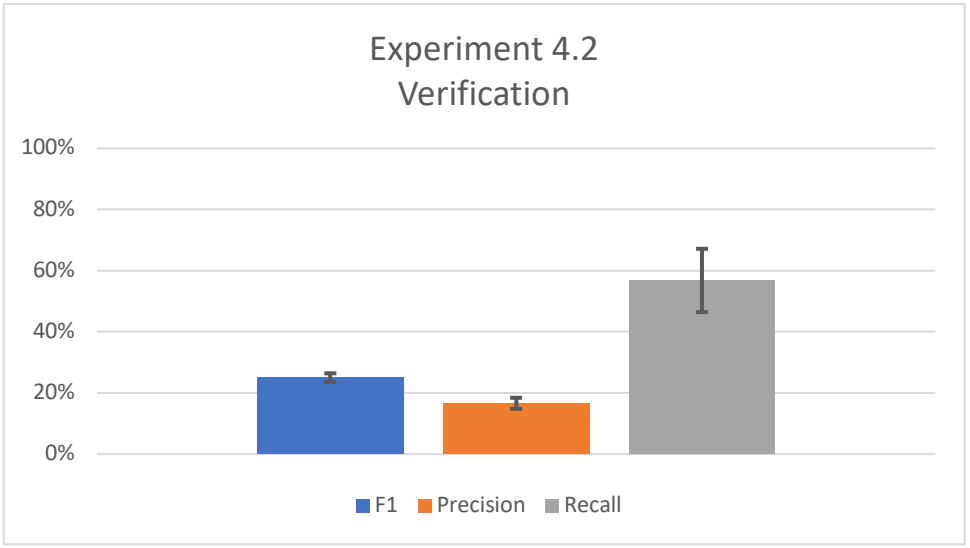


Figure 35:Average Verification F1 score, precision and recall for overview perspective experiment 4.2

7. Discussion

Four tests have been done in the evaluation of the final prototype. The results of the four tests will be discussed.

7.1 Experiment 1

The first test compared the final prototype to the existing methods in the literature. The average accuracy over the 5-folds is 84.4% for our implementation of the existing methods. The average recall and precision are 83.6% and 79.8%, respectively. Marsot et al. [11] reported an average of 83% and Hansen et al. [18] 96.7%, and both used a softmax classification. Our results are similar to Hansen et al. but lower than Marsot et al. Hanset et al. did test on the same pigs as training but tested on data captured 30 days later. It is expected that the data used greatly influences the results. Marsot et al. used pigs with very distinct black skin spots. From their class activation map, it can be seen that their network looks at those black spots which is expected to positively influence the results. The pigs used by Hanset et al. look more like the pigs we used. The results using a KNN classifier are higher than using a softmax classifier. The average accuracy recall and precision of the embedding network and KNN classifier are 87.8%, 89.8% and 89.8%, respectively. This is an higher accuracy than Marsot et al. however it is still not as high as the result of Hansen et al. Verification scored much lower than both identification methods. For verification, the average F1 score, precision and recall are 65%, 55%, and 83%, respectively.

7.2 Experiment 2

The second test evaluated how well the final prototype identified pigs that were not seen during training of the embedding network. The average accuracy, precision and recall for the KNN classification are 55.5%, 48.6% and 47.4%, respectively. This is much lower than identifying pigs which are seen during training as seen in the previous experiment. Verification is even lower with an F1 score, precision and recall of 21%, 13%, and 52%. This is much lower than other livestock verification methods [21][22].

Besides that, is it can be seen that facenet [23], which is similar to the verification method we used, achieves an accuracy of 99.6% on the LWF dataset., Other human face verification techniques perform at similar accuracies as facenet [24], [25]. The verification as seen in the literature only needs less than 10 images in the gallery. From our experiment it can be seen that the KNN identification performs better if there are more images in the gallery. After around 80%, the accuracy does not increase, corresponding to roughly 30 images. This is more than the identification techniques as seen in the literature. There could be less biometric information in a pig's face than other livestock animals or humans and therefore perform worse. However, even if there is less biometric information, it should be able to reach a performance closer to the first experiment where testing and training are done on the same pigs.

7.3 Experiment 3

The third test evaluated the identification performance 18 days after the pigs were added to the KNN classification. This showed that pigs could not accurately be recognised 18 days after being added to the KNN classifier. The average accuracy, precision and recall are 18%, 14%, and 15%, similar to a random classification. After adding the newer data to the KNN classifier, the classification performance is similar to the older data. It is expected that the newer data is in different clusters than the older data because the results are much higher in experiment 2. Hansen et al. tested on data 30 days later than trained and reached a performance of 84%. Although they tested on the same pigs as trained, they showed that it could be done to recognize pigs after a certain amount of time. Verification again performed worse than the KNN classification in both experiments. It is expected that the image quality does influence our results. The security camera attracted a lot of dust which made the videos look different than 18 days earlier.

7.4 Experiment 4

The last test evaluated how the final prototype performs when identifying pigs from another perspective than they are added to the KNN classifier. This test did use the controlled dataset captured using the Raspberry Pi, which was expected to be too dark for recognition. However, it is used such that the same camera type in both perspectives is similar. When testing on images from the overview perspective and the gallery only contains images of the controlled perspective, the KNN accuracy precision and recall is 20%, 19%, and 19% respectively. Verification does not perform better and has a F1 score, precision and recall of 15%, 10%, and 49%. The difference is that the recall for verification is higher than for the KNN identification. When images of the overview perspective are added to the gallery, the performance of both the KNN identification and verification are higher. Again the performance of the KNN identification is higher than that of the verification. The accuracy of the KNN identification is 61% if images of the overview perspective are added to the gallery. This is higher than the accuracy of the second experiment where images of the same perspective are used. The difference between these two experiment is that another dataset is used. The dataset captured using the raspberry pi is used for this. The difference between the two is that the raspberry pi dataset has more images per pig and that RGB images are used instead of infrared. The security camera did also collect more dust which made the images blurry. These differences can all influence the performance. Although the accuracy is higher than that of the second experiment it still not is as high as when training and testing on the same pigs. Bergamini et al. [22] showed that identifying a cow in different angles does lower the performance. The accuracy was 74% when trying to identiy the cow in different angles and 89% when trying to identify the cow using a technique that uses two angles. The effect could also be present for pigs. The overview and controlled view angles might be too different for the identification or verification to match the two.

7.5 Identification and Verification

In the experiments, it could be seen that verification performs worse than KNN classification.

The KDE plots of experiment 2, 3 and 4 show that there is very little difference in distance between similar and different identities. The performance of verification is low for these experiment. The KDE plot in experiment 1 does show a difference between the distances and the verification performance is higher than for experiment 2,3, and 4. In all four experiments, verification performed worse than identification. In the facenet paper [23], it could be seen that verification performed similarly to KNN identification. Based on the KDE plot it seems that the network fails to create one distinct cluster for each identify. This can be observed when used later data or identified in another perspective. Images can only be identified or verified if later images or images in another perspective are added to the galley. There are many different clusters instead of one cluster per identity. The higher performance of the KNN identification is expected from the fact that it only looks at the closest embedding and thus can still identify even if there are many different clusters per identity. This can either be the result of not enough biometric information in the pigs' faces or the fact that the images in the dataset are dark or blurry due to the bad conditions in the shed.

8. Conclusion and Recommendation

The literature review and technology exploration showed that this could be done using a convolutional neural network trained using the triplet loss function. This method is evaluated using a dataset containing 105 individual pigs. The evaluation that the performance is in the same range as the existing identification methods for pigs when tested on ten pigs. However, the performance is much lower when trying to identify or verify pigs not seen during training. The pigs not seen during training can be identified with an accuracy of 56% and verified with an f1 score of 21%. This is lower than verification of other livestock animals and human face recognition, which reach accuracies of 99% [21][23]. Identifying and verifying pigs using later data or in another perspective only performs better than random when data from the same time or same perspective is added to the gallery as used in the probe set. If later images and images in the overview perspective are added to the gallery the identification accuracy is 58% and 61%, respectively. This showed the effect of different perspectives and how KNN can be used to deal with those. Added images over a longer period of time in multiple perspectives to the gallery is not wanted in a real-world application as it is cumbersome. The KNN identification performs better than verification using a distance threshold for all experiments. Time constrains prevented exploring fully why the performance degraded so much in different perspective and over time, but it shows the need for more extensive data collection.

Future work should focus on how much biometric information there actually in in a pig's face and if this is enough to reach similar performances as verification of humans and other livestock animals. A more complete dataset can help with this, especially since there is not yet a public dataset containing images of pig faces. This dataset should be captured over a longer period of time and from multiple perspectives. Most important is that the images should be clear and bright unlike our collected dataset. It is expected that better results can be achieved using our method with a better dataset.

References

[1] C. Tzanidakis, P. Simitzis, K. Arvanitis, and P. Panagakis, 'An overview of the current trends in precision pig farming technologies', *Livest. Sci.*, vol. 249, p. 104530, Jul. 2021, doi: 10.1016/j.livsci.2021.104530.

[2] V. Bloch and M. Pastell, 'Monitoring of Cow Location in a Barn by an Open-Source, Low-Cost, Low-Energy Bluetooth Tag System', *Sensors*, vol. 20, no. 14, p. 3841, Jul. 2020, doi: 10.3390/s20143841.

[3] A. Huhtala, K. Suhonen, P. Mäkelä, M. Hakojärvi, and J. Ahokas, 'Evaluation of Instrumentation for Cow Positioning and Tracking Indoors', *Biosyst. Eng.*, vol. 96, no. 3, pp. 399–405, Mar. 2007, doi: 10.1016/j.biosystemseng.2006.11.013.

[4] N. J. B. McFarlane and C. P. Schofield, 'Segmentation and tracking of piglets in images', *Mach. Vis. Appl.*, vol. 8, no. 3, pp. 187–193, May 1995, doi: 10.1007/BF01215814.

[5] L. Zhang, H. Gray, X. Ye, L. Collins, and N. Allinson, 'Automatic Individual Pig Detection and Tracking in Pig Farms', *Sensors*, vol. 19, no. 5, p. 1188, Mar. 2019, doi: 10.3390/s19051188.

[6] W. Kim, Y. B. Cho, and S. Lee, 'Thermal Sensor-Based Multiple Object Tracking for Intelligent Livestock Breeding', *IEEE Access*, vol. 5, pp. 27453–27463, 2017, doi: 10.1109/ACCESS.2017.2775040.

[7] A. Bhole, S. S. Udmale, O. Falzon, and G. Azzopardi, 'CORF3D contour maps with application to Holstein cattle recognition from RGB and thermal images', *Expert Syst. Appl.*, vol. 192, p. 116354, Apr. 2022, doi: 10.1016/j.eswa.2021.116354.

[8] M. Ju *et al.*, 'A Kinect-Based Segmentation of Touching-Pigs for Real-Time Monitoring', *Sensors*, vol. 18, no. 6, p. 1746, May 2018, doi: 10.3390/s18061746.

[9] J. Kim *et al.*, 'Depth-Based Detection of Standing-Pigs in Moving Noise Environments', *Sensors*, vol. 17, no. 12, p. 2757, Nov. 2017, doi: 10.3390/s17122757.

[10] J. Sa, Y. Choi, H. Lee, Y. Chung, D. Park, and J. Cho, 'Fast Pig Detection with a Top-View Camera under Various Illumination Conditions', *Symmetry*, vol. 11, no. 2, p. 266, Feb. 2019, doi: 10.3390/sym11020266.

[11] M. Marsot *et al.*, ‘An adaptive pig face recognition approach using Convolutional Neural Networks’, *Comput. Electron. Agric.*, vol. 173, p. 105386, Jun. 2020, doi: 10.1016/j.compag.2020.105386.

[12] Y. Guo, W. Zhu, P. Jiao, C. Ma, and J. Yang, ‘Multi-object extraction from topview group-housed pig images based on adaptive partitioning and multilevel thresholding segmentation’, *Biosyst. Eng.*, vol. 135, pp. 54–60, Jul. 2015, doi: 10.1016/j.biosystemseng.2015.05.001.

[13] F. Kang, C. Wang, J. Li, and Z. Zong, ‘A Multiobjective Piglet Image Segmentation Method Based on an Improved Noninteractive GrabCut Algorithm’, *Adv. Multimed.*, vol. 2018, pp. 1–9, Jul. 2018, doi: 10.1155/2018/1083876.

[14] G. J. Tu, H. Karstoft, L. J. Pedersen, and E. Jørgensen, ‘Segmentation of sows in farrowing pens’, *IET Image Process.*, vol. 8, no. 1, pp. 56–68, Jan. 2014, doi: 10.1049/iet-ipr.2012.0734.

[15] M. Riekert, A. Klein, F. Adrion, C. Hoffmann, and E. Gallmann, ‘Automatically detecting pig position and posture by 2D camera imaging and deep learning’, *Comput. Electron. Agric.*, vol. 174, p. 105391, Jul. 2020, doi: 10.1016/j.compag.2020.105391.

[16] E. Psota, M. Mittek, L. Pérez, T. Schmidt, and B. Mote, ‘Multi-Pig Part Detection and Association with a Fully-Convolutional Network’, *Sensors*, vol. 19, no. 4, p. 852, Feb. 2019, doi: 10.3390/s19040852.

[17] E. T. Psota, T. Schmidt, B. Mote, and L. C. Pérez, ‘Long-Term Tracking of Group-Housed Livestock Using Keypoint Detection and MAP Estimation for Individual Animal Identification’, *Sensors*, vol. 20, no. 13, p. 3670, Jun. 2020, doi: 10.3390/s20133670.

[18] M. F. Hansen *et al.*, ‘Towards on-farm pig face recognition using convolutional neural networks’, *Comput. Ind.*, vol. 98, pp. 145–152, Jun. 2018, doi: 10.1016/j.compind.2018.02.016.

[19] Z. Wang and T. Liu, ‘Two-stage method based on triplet margin loss for pig face recognition’, *Comput. Electron. Agric.*, vol. 194, p. 106737, Mar. 2022, doi: 10.1016/j.compag.2022.106737.

[20] S. Kumar, S. K. Singh, R. S. Singh, A. K. Singh, and S. Tiwari, 'Real-time recognition of cattle using animal biometrics', *J. Real-Time Image Process.*, vol. 13, no. 3, pp. 505–526, Sep. 2017, doi: 10.1007/s11554-016-0645-4.

[21] D. Yeleshetty, L. Spreeuwers, and Y. Li, '3D Face Recognition For Cows', presented at the 2020 International Conference of the Biometrics Special Interest Group (BIOSIG), 2020.

[22] L. Bergamini *et al.*, 'Multi-views Embedding for Cattle Re-identification', in *2018 14th International Conference on Signal-Image Technology & Internet-Based Systems (SITIS)*, Las Palmas de Gran Canaria, Spain, Nov. 2018, pp. 184–191. doi: 10.1109/SITIS.2018.00036.

[23] F. Schroff, D. Kalenichenko, and J. Philbin, 'FaceNet: A unified embedding for face recognition and clustering', in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Boston, MA, USA, Jun. 2015, pp. 815–823. doi: 10.1109/CVPR.2015.7298682.

[24] O. M. Parkhi, A. Vedaldi, and A. Zisserman, 'Deep Face Recognition', in *Procedings of the British Machine Vision Conference 2015*, Swansea, 2015, p. 41.1-41.12. doi: 10.5244/C.29.41.

[25] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, 'DeepFace: Closing the Gap to Human-Level Performance in Face Verification', in *2014 IEEE Conference on Computer Vision and Pattern Recognition*, Columbus, OH, USA, Jun. 2014, pp. 1701–1708. doi: 10.1109/CVPR.2014.220.

[26] Z. Zhang, 'A flexible new technique for camera calibration', *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 11, pp. 1330–1334, Nov. 2000, doi: 10.1109/34.888718.

[27] B. W. He, X. L. Zhou, and Y. F. Li, 'A new camera calibration method from vanishing points in a vision system', *Trans. Inst. Meas. Control*, vol. 33, no. 7, pp. 806–822, Oct. 2011, doi: 10.1177/0142331209103040.

[28] O. Bogdan, V. Eckstein, F. Rameau, and J.-C. Bazin, 'DeepCalib: a deep learning approach for automatic intrinsic calibration of wide field-of-view cameras', in *Proceedings of the 15th ACM SIGGRAPH European Conference on Visual Media*

Production - CVMP '18, London, United Kingdom, 2018, pp. 1–10. doi: 10.1145/3278471.3278479.

[29] J. Zimmerman and J. Forlizzi, 'Research Through Design in HCI', in *Ways of Knowing in HCI*, J. S. Olson and W. A. Kellogg, Eds. New York, NY: Springer New York, 2014, pp. 167–189. doi: 10.1007/978-1-4939-0378-8_8.

[30] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, 'YOLOv4: Optimal Speed and Accuracy of Object Detection', 2020, doi: 10.48550/ARXIV.2004.10934.

[31] A. Shirke *et al.*, 'Tracking Grow-Finish Pigs Across Large Pens Using Multiple Cameras', 2021, doi: 10.48550/ARXIV.2111.10971.

[32] L. Bergamini *et al.*, 'Extracting Accurate Long-term Behavior Changes from a Large Pig Dataset':, in *Proceedings of the 16th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*, Online Streaming, --- Select a Country ---, 2021, pp. 524–533. doi: 10.5220/0010288405240533.

[33] O. Moindrot, 'Triplet Loss and Online Triplet Mining in TensorFlow'.

[34] 'TensorFlow Addons Losses: TripletSemiHardLoss'. https://www.tensorflow.org/addons/tutorials/losses_triplet

[35] 'https://github.com/motioneye-project/motioneye'.