Master's Thesis

Anonymisation of Image Data Using Deep Learning

Jonna Johansson and Jesper Lundberg

September 2020

Supervisors:
Mikael Nilsson, LTH
Håkan Ardö, Axis Communications AB

LUNDS UNIVERSITET

Abstract

Collecting and storing data is easier and more common than ever before. A lot of this data is personal data, which is problematic to store and use both for ethical reasons and because of legislations. In some scenarios the personal information in the data is not interesting or relevant for the given task but it is still collected and stored as a byproduct of the data collection. In these scenarios it would be much better if one could use anonymised data instead.

This report presents a neural network approach for creating an anonymisation filter for image data, specifically for images depicting humans taken from above. A convolutional neural network is used as the filter. It is trained together with a person detector and ReID in order to generate images where it is possible to detect people in the images but impossible to identify them. The training process is similar to that of a generative adversarial network since the goal of the filter was to construct an anonymisation that makes it easy for the detector but hard for the ReID and the aim for the detector and ReID was to become as good as possible. Using the presented method a filter was created which succeeds in anonymising the images whilst almost maintaining the performance of the person detection. Using the metrics AP for the detector and AUC-ROC for the ReID this filter yielded the results 0.864 and 0.540 respectively. However, several of the parameters were very sensitive to changes, yielding results that varied widely when small changes were made, making the network fairly hard to train.

Acknowledgements

We would first and foremost like to thank our two supervisors Mikael Nilsson and Håkan Ardö for helping with the structure of this master's thesis and for giving great advice. To Håkan we also would like to give our thanks for providing and explaining the detector as well as for always being ready to answer questions and help with practical issues. We would also like to thank Axis, especially the team working at the Cognimatics department, for giving us this opportunity and for providing us with a work space even though there was already no space left. It is also important for us to acknowledge the care that has been taken by all parties to make sure that our work could be continued despite the covid-19 pandemic situation. Also, we would like to thank Gustaf Oxenstierna for providing and explaining the ReID code to us and for helping with technical complications.

Furthermore we would very much like to thank our family and friends who supported us throughout this thesis and our education. Special thanks to our friend Emil Johansson for all the helpful discussions regarding the report and for his immaculate proof reading skills. For helping with proof reading we would also like to thank Marie Lundberg. Thanks to our friend Desiré Nilsson for offering us the stylish layout for the report previously used by herself. Finally, to Magnus and Åsa Johansson, thank you for the extra support, always wanted, sometimes needed.

Contents

1 Introduction **4**
1.1 Background . . . 4
1.2 About Axis Communications . . . 4
1.3 Aim . . . 5
1.4 Related Work . . . 5
1.5 Thesis Outline . . . 6

2 Artificial Neural Networks **7**
2.1 Traditional Artificial Neural Networks . . . 7
2.2 Convolutional Neural Networks . . . 9
2.3 Generative Adversarial Network . . . 10

3 Data **11**
3.1 Simple Data Set . . . 11
3.2 Real Data Set . . . 12

4 Method **14**
4.1 Network Design and Training . . . 14
4.1.1 Object Detector . . . 17
4.1.2 ReID . . . 17
4.1.3 MobileNet . . . 18
4.1.4 Filter . . . 18
4.1.5 The variations of filters and training methods tested . . . 19
4.2 Evaluation Methods . . . 19
4.2.1 Precision, Recall and False positive rate . . . 20
4.2.2 Detector Metric: Average Precision . . . 21
4.2.3 ReID Metric: AUC-ROC . . . 22

5 Results **24**

6 Discussion **29**
6.1 General discussion . . . 29
6.2 What is Anonymity? . . . 30
6.3 Future Work . . . 31
6.4 Conclusions . . . 31

References **32**

Appendix A **35**

1 Introduction

1.1 Background

The issue of handling personal data is growing as data becomes easier and easier to collect and store. If the data collected is personal data it must be protected and handled securely, mainly for ethical reasons but also to abide by current legislations such as GDPR [16]. If certain sensitive personal data falls into the wrong hands it could have disastrous consequences. However as good as it is that these regulations exist, they are also restricting. Machine learning solutions can for example be used for great innovation and to improve society but they are also often privacy-invasive. The most common issues are when image data containing people are collected and annotated to use during the training phase and when the finished network is put to use. The reason why using the finished network is problematic is that the data it has learned to process is the same kind of privacy-invasive data it has trained on and thus has to keep being provided the same kind of troublesome data. In a lot of applications however, the moral and logistic issues could be greatly reduced if not avoided completely. This is true for every application where only anonymous statistics are needed and information regarding individuals are irrelevant, such as counting or locating people. Applications like these could provide a lot of benefits within society and improvements to daily life, but are at risk of not being utilized due to complications with handling nonanonymous data. Applications where these kinds of solutions could be used include counting the number of people in a building (which could be useful in case of emergencies), analysing people flow in stores and city centres or in self driving cars.

The issues with handling privacy-invasive image data where the personal information is unnecessary can be mitigated by using an anonymisation filter which preserves the information needed to detect people but distorts them enough to make them unidentifiable. One of the most common ways of identifying people is by analysing their faces. There already exist anonymisation solutions in which faces are detected and distorted while keeping the rest of the image untouched, for some examples see section 1.4, Related Works. However these current solutions are completely dependent on the performance of a face detector and therefore so would any future applications using the anonymised images produced by these filters. A problem with this is that detectors sometimes fail to detect faces. Even today's state of the art detectors do not perform perfectly, see for example the results produced by the Cascade R-CNN which is one of the top performing detectors [2]. This would leave some persons completely identifiable. Face detection also requires a lot of computing power which makes it unsuitable for most cameras, therefore requiring privacy-invasive images to be saved in order to later be anonymised. One could also question whether anonymisation of the face is adequate for making a person unidentifiable, most likely there are other possible features that could be used for identification. A better solution would in many cases be an anonymisation filter which do not rely on any semantic understanding and through which entire images could be filtered before they are ever saved, so that the issues with privacy-invasive images could be avoided.

1.2 About Axis Communications

This project has been completed in collaboration with Axis Communications AB, the company at which the premise of the thesis was created. Axis produces surveillance cameras of various kinds for various purposes. Some are used for people counting, where the identity of the people captured on film are irrelevant. One example of this is when keeping track of the amount of people in a building by having cameras over every entry and exit. This is the kind of application our thesis work and filter will be directed towards.

1.3 Aim

The aim is to evaluate a deep learning method and variations of it to anonymise image data while still preserving enough features for person detection, without using a detector to target certain parts of the image to modify. We will also reflect over what anonymity is and how it is measured.

1.4 Related Work

Since anonymisation is a growing problem in today's society, many different solutions have been presented. A common feature amongst them is that a filter is only applied on the parts of images depicting faces, which is accomplished using a face detector. Often the faces are pixelated, blurred or blacked out while the rest of the image is untouched. These methods have been used for example by Google street view, tv programs and magazines [23]. However it has been disputed whether approaches like these are in fact anonymous. In a publication by Natacha Ruchaud and Jean-Luc Dugelay [23] they argue that many of the common anonymisation techniques used today do not anonymise very well. They demonstrate this by making a network that identifies the anonymisation method used and then reverses it to get the original images. It was tested on blurring, blackening, pixelation and noise filters and got good results in all instances.

There also exist solutions which still use face detection but where the faces are anonymised by neural networks, with architectures similar to generative adversarial networks, GANs, that generate new anonymous facial features. One example of such a solution has been created by Zhongzheng Ren, Yong Jae Lee and Michael S. Ryoo [21], wherein faces are detected and their features distorted to make new unrecognisable faces. This is done by constructing a network consisting of a face detector, a face modifier, an action classifier and a person identifier. The training alternates between all components except for the face detector in order to make them compete against each other, similar to a GAN. The function of the face detector is to provide a cropped image consisting of only the face to the face modifier. A prototype for another solution was created by Nikhil Chhabra [3] where a GAN is constructed and used together with a face detector. Given an image the resulting filter creates an anonymous version of the face and replaces the original with the anonymous version. The GAN consists of a generative part and an adversarial part. The generator comprises an encoder and a decoder part. In the encoder a representation of a face is made using learned features; the input is an image with a face partly covered by a gray box. The location of the gray box is determined by the face detector in order to cover the center of the face. Using the given representation the decoder tries to reconstruct a face. The adversarial part of the network consists of a face classifier which compares the generated face and the original image and tries to identify which one is the original image and which one is the generated image. After training is completed the generator functions as the anonymisation filter.

A problem with using face detection as a basis for anonymisers is that today's state of the art face detectors do not work in every case, see for example the results produced by the Cascade R-CCN which is one of the top performing detectors [2]. There are circumstances that make face detection harder, for example if a part of the face is obscured or if the image is taken at a strange angle. With such complications and sometimes even without complications the face detectors might fail. If these face detectors are then used as a basis for anonymisation filters the faces that the detector misses will be identifiable.

Therefore an approach which filters the entire image instead of only disrupting faces will be explored, on which previous work has not been found. However, in order to create a useful filter, the ability to detect people and the inability to recognize them must be tested. How easily people are detected in the filtered images is tested by studying how well an object detector, of which there are many, performs. One example of a common object detector is YOLO, You Only Look Once, made by Joseph Redmon et al. in 2015 [17]. YOLO consists of a single CNN and one of its primary features is that it is fast. Several later versions of this detector has also been constructed [18, 19]. Another common object detector is R-CNN, which also utilises a CNN as well as using selective search to propose potential parts of the image that might contain entire objects [6]. This object detector has too been developed further into other versions [8, 20]. A third detector is the MobileNet developed by Andrew H. Howard et al. in 2017 [9]. It is a CNN network which is lightweight and efficient. The person detector which will be used in this project is based on the MobileNet. It was created by Håkan Ardö at Axis but is not published.

In order to test the inability to recognize persons after the filter has been applied to an image a reidentification solution, often referred to as a ReID, can be applied. If the ReID performs badly when trained on filtered images, people are at least to an extent anonymous. Many ReID solutions exists, for example the spatial-temporal ReId presented by Guanggeong Wang et al. which uses spatial and temporal information as well as appearance for classification [26]. Another solution is Robust ReId created by Hussam Lawen et al. This solution is created specifically so that no large pretrained part is needed which makes exploration of different architectures simpler [12]. Niki Martinel, Gian Luca Foresti and Christian Micheloni has presented the ReId solution pyrNet which has a structure resembling a pyramid and considers features at different depths [14]. In this project a ReID solution constructed during a previous master's thesis at Axis by Johan Rodhe and Gustaf Oxenstierna will be used [22]. The focus of the earlier master's thesis was to identify people from a top down perspective in order to estimate mean visiting times in stores. This network is based on a MobileNet just like the object detector which will be used in this project.

1.5 Thesis Outline

The aim of this report is to give the reader an understanding of how the anonymisation filter in this project was created and how it works. This is done by first giving a gentle introduction to artificial neural networks. In this section neural networks will be discussed in general. In addition an introduction to convolutional neural networks will be given since it is the kind of network that will be used for the filter. Generative adversarial networks will also be explained since the structure of such a network is similar to the structure that will be used to train the filter. After this section, a section about the data that has been used during training will follow. Then, details about the construction of the networks and the training algorithm will be given in the section called Method. This section will also include an explanation of the evaluation methods used for the different parts needed. Thereafter follows the section where the results will be presented. Lastly there will be a discussion in which the results will be analysed. This is also the section in which the concept of anonymity is examined and where it will be discussed how well the evaluation methods in this project can be considered to measure anonymity.

2 Artificial Neural Networks

2.1 Traditional Artificial Neural Networks

An artificial neural network, hereafter referred to as an ANN, is a method for computers to solve problems which is inspired by how the brain works. The idea behind ANNs is that there are artificial neurons called nodes connected in a graph, where each node contributes to the collective goal of learning to solve one type of problem. Nodes, like neurons, take inputs and produce an output, see Figure 2.1. They do so by summing all inputs and applying an activation function, a function which determines if an output should be produced and how large it should be, depending on the sum of the inputs. A node can solve very simple problems, for example the logic operation 'and'. Let 1 represent a true statement and 0 a false one. A node with two inputs which has been trained to perform the AND operation should create an output representing true if both input1 AND input2 are true. If both inputs are true, both inputs will be 1 which means that they will sum to 2, if at least one of the inputs is not true the sum will be smaller than 2. By having an activation function for that neuron which only produces an output of 1, representing true, for sums which are 2 and an output of 0 otherwise, it is clear that the neuron solves the problem. Sometimes though, the importance of the different inputs can vary or be scaled very differently. Therefore the inputs are weighted before they are summed, which can be thought of as scaling the inputs so that each of them contribute the right amount to the output. It is the value of these weights that change when training the network. A mathematical formula representing a neuron, where n is the number of inputs, x_i are the inputs, y is the output, w_i are the weights and b the bias, is

$$y = f\left(\sum_{i=1}^{n} w_i x_i + b\right). \tag{1}$$

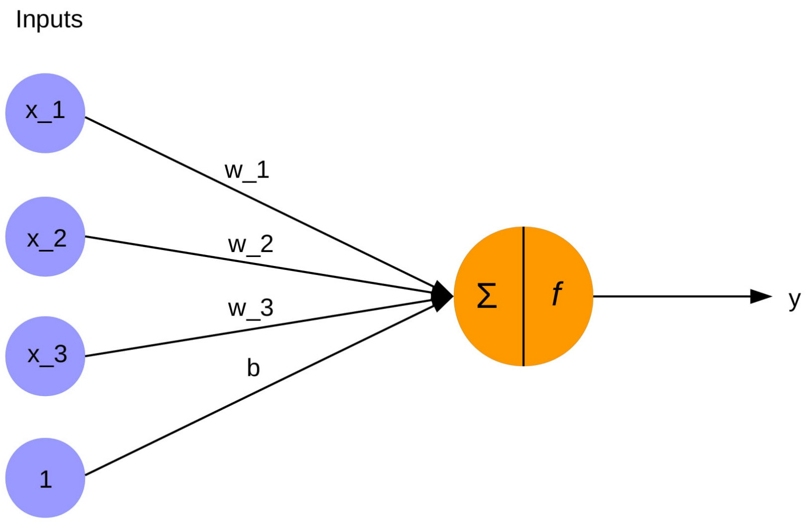


Figure 2.1 – A node having three inputs with a bias and an activation function f.

In the training process the network is repeatedly fed data as inputs for which the correct outputs are known and then the weights are updated accordingly. The values for the weights are updated based on how much they contributed to the error between the produced output and the desired output. The weights are often referred to as parameters and the process of changing them is often called tuning the parameters.

Typical ANNs consists of a lot of connected nodes, see Figure 2.2. If they are connected into several layers the network is called a deep neural network. Since nodes can solve small problems it is not hard to imagine that in a larger structure of them each node can be used to solve a subtask which can result in a solution to a more complex problem. Examples of such problems is finding objects in images or recognising whether two photos are of the same person.

One issue when training a network is that it might get overtrained which means that it has become too specialised on the exact data it has been provided during training. Therefore networks are tested with data from a different data set than the training data. This new data is not used to modify the network by updating any weights, it is only used to get a more realistic estimate of the performance for future unseen data. For the interested reader a much more thorough and rigorous explanation can be found in the book *Deep Learning* by Ian Goodfellow, Yoshua Bengio, and Aaron Courville [7].

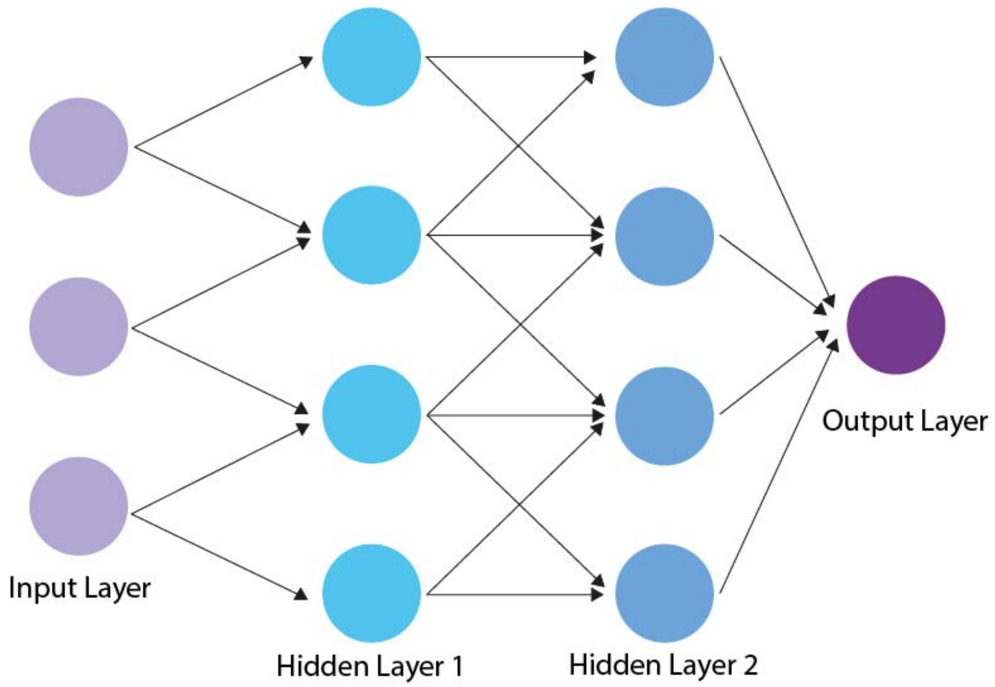


Figure 2.2 – An example of a neural network structure.

2.2 Convolutional Neural Networks

A common type of network is the Convolutional Neural Network, often refereed to as a CNN. For an illustration of how a simple CNN works, see Figure 2.3. CNNs are primarily used when the data consists of images. It has layers, called convolutional layers, in which it only looks at small portions of the images at a time trying to find features. Examples of features it might learn to find are edges, corners, colours or textures. The small parts that find features in the images are called kernels and they slide over the entire image looking for the exact same feature in every portion of the image. If the feature is found it is stored in a new image, called a feature map. In every layer one can have several kernels looking for different features resulting in more than one feature map in that layer. In the next convolutional layer the kernels looks for features within these feature maps, and so on; as a result it is usually possible to find larger and more complex structures for every layer. This is much more efficient than having a node for every pixel in the image connect to all the nodes representing pixels in the next layer. Such a structure would often result in too many connections to train and too much training data would be needed.

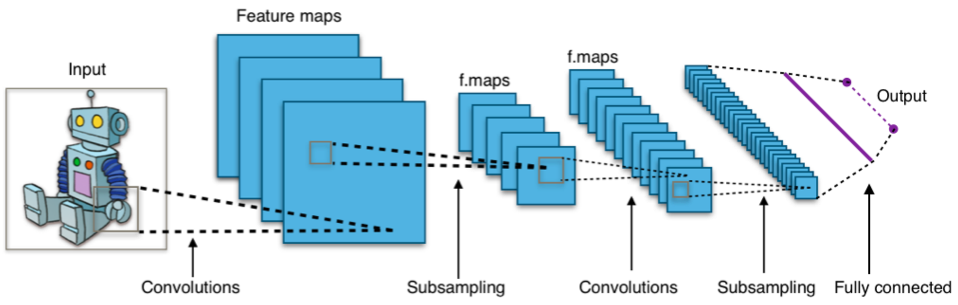


Figure 2.3 – An illustration of how a simple CNN works [1].

CNNs can for example be used when trying to find people in images or when trying to identify them, both of which will be needed to create the filter in this project. However, something we also want to do is to modify existing images, since we are creating a filter, which also can be done by using convolutional layers. More information about CNN’s can be found in the book *Deep Learning* [7].

2.3 Generative Adversarial Network

Generative Adversarial networks, also called GANs, are a type of network which usually consists of two parts, see Figure 2.4. A generator which has the task of generating fake but realistic data, for example images of faces or artworks, in order to fool the discriminator. The discriminator, the other part of the network, is assigned to determine whether the input it gets is real or synthesised by the generator. The idea is to make the two parts compete against each other by updating their weights alternately so that the generator gradually gets better at fooling the discriminator at the same rate as the discriminator learns how to not be tricked by the generator. The more interested reader can find a more detailed explanation in the book mentioned before [7].

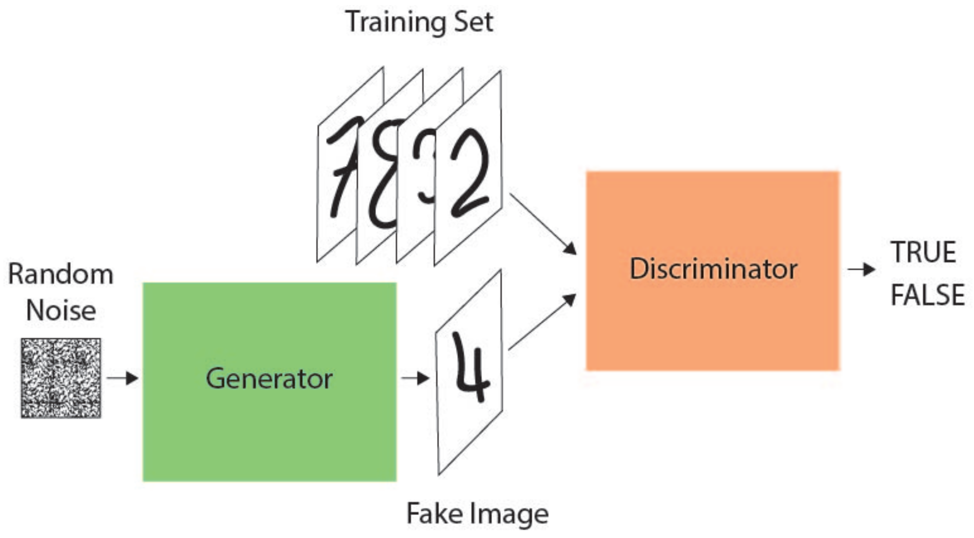


Figure 2.4 – A typical GAN trained to produce images of numbers.

We will not be creating or using any GANs in this project since there is no anonymised data that we would like to mimic, like the generator in a GAN would. What we intend to do is to *find* a distribution for well anonymised data. This means that the generator tries to come up with an anonymisation instead of imitating data. Hence the role of the discriminator is also slightly different, it needs to judge the generator not by how well it creates fakes but how well it anonymises data. The extent to which the data is anonymised will be measured by the detector and the ReId. Even though the network needed to create the filter is not a GAN it is similar in many ways, for example in that it will have different competing parts.

3 Data

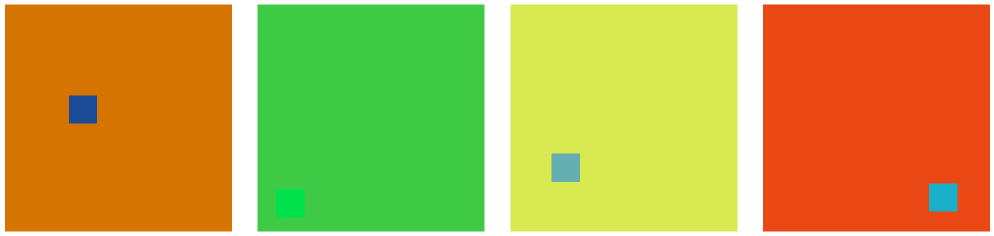
During this project two data sets were used. One simpler generated data set, in order to investigate whether or not the method works, and one using real life data. The data set consisting of real images is in top-down view so that the resulting filter could be used for counting people in buildings which is the most relevant for further research within Axis. Therefore the simple data set was constructed to mimic a top-down view as well.

3.1 Simple Data Set

The simple data set was generated and consists of two parts. One of the parts is data for the detector consisting of images of size 160×160 pixels. The images contain a coloured square of size 20×20 pixels on a coloured background. Both the colour of the square and the colour of the background were randomised for each image. For each colour randomised for the squares, five images were generated representing five images of the same identity. The location of the squares were also randomised, with the limitation that every square had to completely fit within its image, for examples see Figure 3.1a. The ground truth for this data is the coordinates for the center of the square. These were used to create heatmaps as targets for the detector. A heatmap is an image where the colour of every pixel describes how close it is to the pixel where the center of the square would be.

The other part is data for the ReID and consists of images with the same size as the images for the detector. Since this data set was used to try the method and the images could be generated to any size while still serving their purpose, the same size was chosen for all images for a slightly simpler implementation. However these images only contain solid randomised colours, for examples see Figure 3.1b. Several of the images have the same colour representing the same individual. The individual in an image is also given by an ID number which is a unique number for every colour. These images were matched together in pairs during training. The label became 1 if the two images represented the same individual and 0 otherwise. The reason the data needs to consist of two different parts is because the detector and the ReID take different kinds of images as input. In order to train correctly the detector needs images where the object it tries to locate is only a small part of the image while the ReID used in this project needs cropped images displaying only the objects whose identities it has to decide.

Since the data was generated, completely new data could be created for every batch during the training. The validation data consisted of the same images every epoch. For the ReID the validation data set consisted of images depicting 20 different individuals and 6 images per individual. Matched into pairs it resulted in 360 pairs, in half of these both images depicted the same identity and in the other half they did not. Augmentation of the data resulted in 720 pairs. For the detector 720 validation images where created with 5 images of each identity. Test data was created in exactly the same way as the validation data.



(a) Example images from the simple data set for the detector.

(b) Example images from the simple data set for the ReID.

Figure 3.1

3.2 Real Data Set

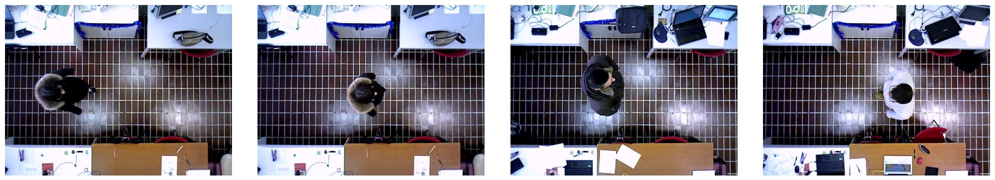
The real data set is based on a data set called TVPR (top view person reidentification) created at the Italian university Università Politecnica delle Marche for scientific and research purposes [13]. The data set consists of 23 videos depicting a total of 100 individuals walking across the frame from a top view perspective. All videos have the same background, an office environment. From these videos six still images depicting each individual were created. This gave a total of 600 images. The creators of the data set gave the following description of the individuals in these videos:

> “The recruited people are aged between 19 - 36 years: 43 females and 57 male; 86 with dark hair, 12 with light hair and 2 are hairless. Furthermore, of these people 55 have short hair, 43 have long hair. The subjects were recorded in their everyday clothing like t-shirts/sweatshirts/shirts, loose-fitting trousers, coats, scarves and hats. In particular, 18 subjects wore coats and 7 subjects wore scarves.” [13].

The data for the detector consists of these images, for examples see Figure 3.2a, and a list of coordinates for the center of the humans, used as ground truth. In order to get the ground truth the images were annotated by hand. To determine the center point of an individual a square bounding box was created which mostly enclosed them. If a person’s limbs were stretched out too far from the body, part of the limb might have been left outside the bounding box in order to keep the majority of the pixels within the bounding box depicting a human. The center coordinates of this box was deemed to be the center of the human and as a result used as ground truth. Just as for the simple data set theses coordinates were used to create heatmaps. Additionally, in order to speed up the training process, the images were downsized to 128×96 pixels from their original size of 640×480.

The data for the ReID consist of the original 600 images, but cropped in order to mostly depict the individuals, for examples see Figure 3.2b. The cropped images were created in exactly the same way as the bounding boxes used to create the detector data described above. They were then resized so that each cropped image were of the size 160×160, which is roughly the median size of the cropped images. Exactly like with the simple data set for the ReID, every identity was given a number and during training the images were matched together in pairs. The pairs function as the data and whether the images in a pair depict the same identity or not was used as ground truth.

The data was divided into training, validation and test data with the proportions 70%, 15% and 15% respectively. This resulted in 630 pairs for the training of the ReiD and 135 pairs each for both the validation and the testing. Since there are fewer data points for the detector, 420 for training and 90 for validation and testing, some images were used more than once each epoch.


(a) Example images from the real data set for the Detector.

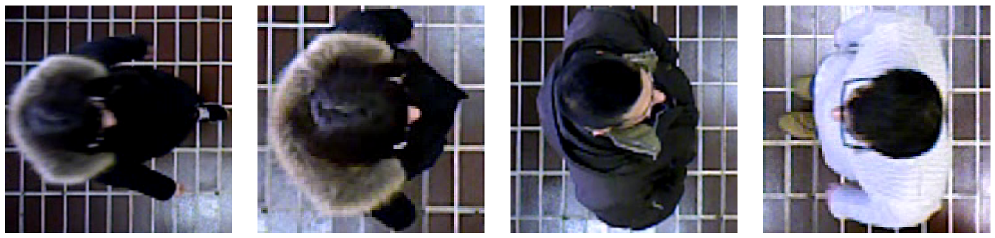

(b) Example images from the real data set for the ReID.

Figure 3.2

4 Method

The main idea behind the method used in this project is to create and train a CNN to become an anonymisation filter. In order to train this filter it is trained together with a person detector and against a ReID. Therefore a bigger network is needed where the three networks are connected even though only the filter part will be used for anonymisation once the training is completed.

In order to see if the method worked it was initially tested using the simple data set. The reason behind using the simple data set was that for this data set the characteristic which separates different individuals is colour alone and therefore it is easy to see if an anonymisation has been accomplished. Once an anonymisation filter had been successfully created using the simple data, validating the method, the data set containing real life data was used instead. For this data several different filters were created and tested in order to produce a well working more applicable filter.

4.1 Network Design and Training

The network needed to train the filter consists of three main parts: an object detector, a re-identification part and the filter, see Figure 4.1. The goal of the filter was to construct an anonymisation that makes it easy for the detector to detect where people are in the image but hard for the ReID to identify who the person is. This means that the filter takes three inputs which it distorts. They consist of a single input image to the detector as well as the two images the ReID takes as input. Technically these inputs were run through three instances of the filter sharing weights which practically is the same thing as running the inputs through the filter one at a time. Once filtered the images were passed to the detector and ReID respectively where the performance of the filter was determined. The network formed when the three parts were joined will be called the combined model. In this combined model the weights for the detector and ReID were untrainable, meaning that they were unable to update. The detector and the ReID were trained separately from the combined model, for a visualisation of the training process see Figure 4.2. By training several filters and varying parameters in both the filter design and the training method it was examined what parameter values, out of the ones tested, are the best to use when creating an anonymisation filter.

During each epoch of the training either batches were created by synthesising new data (for the simple data set) or data was divided into batches (for the real data set). Then the images in the batch were normalised. For every batch the weights in the detector were updated first, then the ReID's and finally the data was run trough the combined model to update the weights for the filter, see Figure 4.2. When training the detector and ReID the data was first passed through the filter in its current state. The loss function used for the detector was mean square error,

$$L_{\text{MSE}} = \sum_{i=0}^{N}\sum_{j=0}^{M}\frac{(y_{i,j}-\hat{y}_{i,j})^2}{MN}, \tag{2}$$

where M is the number of pixels in an image, N is the number of images in a batch, y is the labels and $\hat{y}$ is the predictions. The loss function used for the ReID was binary crossentropy,

$$L_{\text{BCE}} = \frac{1}{N}\sum_{i=0}^{N}(y_i\log(\hat{y}_i) + (1-y_i)\log(1-\hat{y}_i)), \tag{3}$$

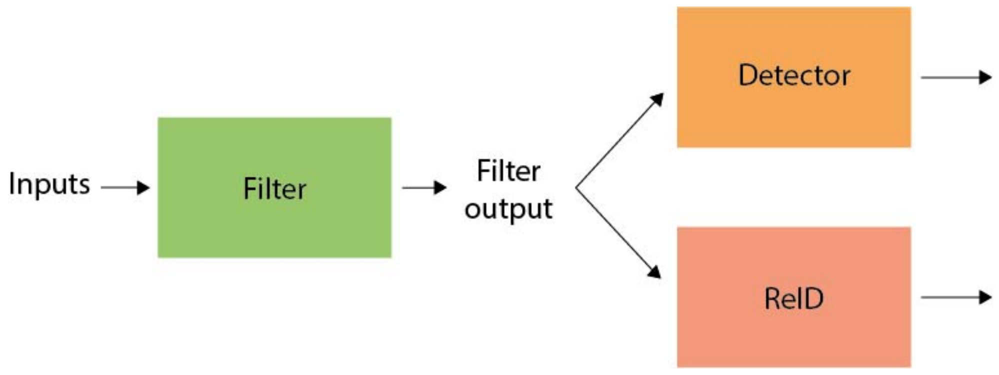


Figure 4.1 – The filter, the detector and the ReID combined into the structure of the combined model. The input to the filter consists of three images. Once filtered one of these images becomes the input to the detector and the other two images become the input for the ReID.

where, exactly as for the formula for L_{MSE}, N is the number of images in a batch, y is the labels and $\hat{y}$ is the predictions. For the combined model a combination of L_{MSE} and L_{BCE} was used. This combined loss, L_{comb}, promotes the filter to help the detector but make it difficult for the ReID. The formula for the loss is given by

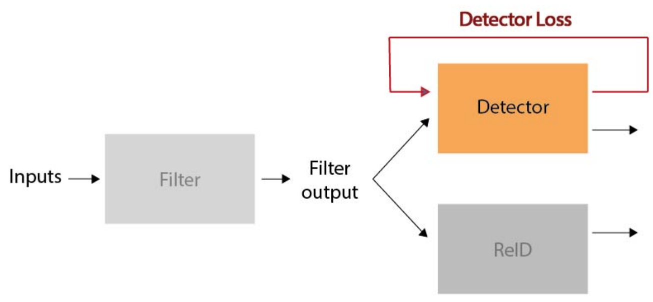
$$L_{\mathrm{comb}} = L_{\mathrm{MSE}} - \frac{L_{\mathrm{BCE}}}{\lambda}. \tag{4}$$

The purpose of parameter λ is to adjust the ratio between L_{MSE} and L_{BCE}. In the network created in this project λ was set to 5, in order for the two losses to contribute more equally to the combined loss.

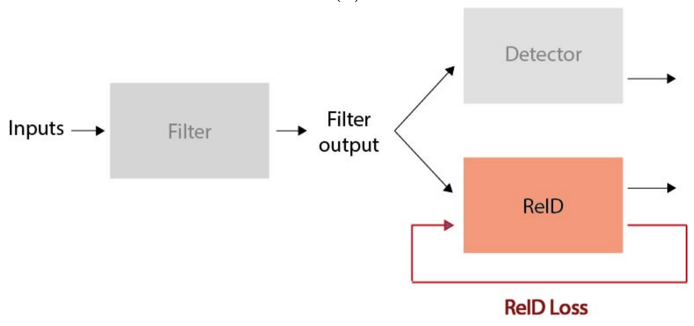
During every training using the simple data set, the optimiser used was the Adam optimiser [4] with the learning rate 0.0005 for the detector and the combined model and 0.001 for the ReID. A more detailed explanation of the different architectures used will follow. Every filter was trained for 12000 epochs and the best performing version among the last 6000 epochs was chosen for testing. The batch sizes used were 60 for training and 90 for validation and testing. This resulted in 12 and 8 epochs per epoch for the training and validation respectively.

When using the real data set the optimiser used was also the Adam optimiser but the learning rates were 0.0001 and 0.001 for the detector and ReID respectively while the learning rate of the filter was varied. Every filter was trained for 200 epochs and the best performing version after at least 20 epochs was chosen for testing. The batch size used was 12 for training, validation and testing. This gives 52 batches each epoch for training and 11 batches for validation.

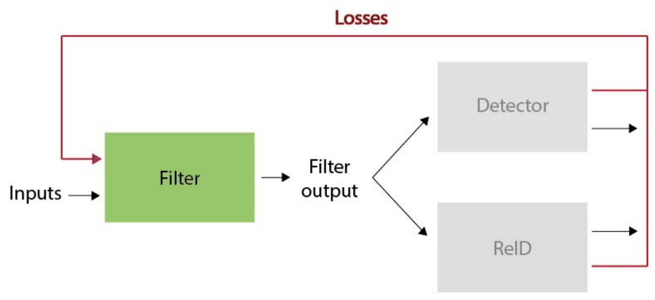
For an overview of the training process, both using the simple and the real data set, see Algorithm 1 or Figure 4.2 below. The code used in this project was implemented in Python 3 using Tensorflow 2.



(a)



(b)



(c)

Figure 4.2 – The training process, where a coloured network represents that the weights are updated and the arrows labeled detector loss, ReID loss and Losses depicts which outputs are used to calculate the new weights. For every batch the training segments occur in the following order: a,b,c.

Algorithm 1 Training

```
for number of Epochs do
    create Batches
    for number of Batches do
        $Y\_detect \leftarrow heatmap$(detectLabel)
        $X\_detect \leftarrow filter$(detectData)
        training Detector on $X\_detect$ with $Y\_detect$

        $X\_reid \leftarrow filter$(reidData)
        training Reid on $X\_reid$ with reidLabel

        training CombinedModel on detectData, reidData with $Y\_detect$, reidLabel
```

4.1.1 Object Detector

The object detector used in this project was developed at Axis by Håkan Ardö. It is based on the MobileNetV2 architecture [24]. An overview of the MobileNet can be found below. This architecture was pretrained on the ImageNet data set for both data sets even though it might not work perfectly for the simple data set. The reason being that ImageNet does not include the kind of images found in the simple data set. In this project the classification needed is slightly different than the classification made by the MobileNetV2 and as a result it needs a different set of final layers. For the detector in this project the aim is to create heatmaps, the purpose of the last layers of the MobileNetV2 is to classify the entire image regarding its content, therefore these layers are removed. After the MobileNetV2, without the last layers, three pairs of layers are instead added. Each pair consists of a convolutional transpose layer followed by a batch normalisation layer and is then followed by a ReLU activation layer. Finally a single convoultional layer is added. The output from the reduced MobileNetV2 is a feature tensor, however after the added layers the result is an image with only one channel that is four times smaller in both width and height than the original input to the detector. The aim of the object detector is to create a heatmap in this output image because the ground truth for each input is a heatmap. They are created from the coordinates in the data set specifying the location of the object. In the heatmaps the warmest part of the map is in the center of the object and then it gradually gets cooler as it gets further away from that center. The loss is calculated by taking the mean square distance for all corresponding pixels between the heatmap created from ground truth with the heatmap produced by the detector. Though this network is not published, a very similar network that is published is the CenterNet for which details can be found here [5]. One of the main differences between the nets are that the CenterNet is based on the ResNet architecture while the detector used in this project is based on the MobileNet architecture. Another major difference is that the dectector in this project produces heatmaps with a higher resolution and therefore it has a few more final layers than the CenterNet. Also the CenterNet produces bounding boxes around the object whilst the detector in this project does not.

4.1.2 ReID

The ReID used in this project was developed by Gustaf Oxenstierna and Johan Rodhe as a part of an earlier master's thesis at Axis [22]. The ReID network is based on the MobileNetV1 architecture [9] without the last layers. An overview of the MobileNet can be found below. Similar to the object detector the MobileNet portion of the network was pretrained on the

ImageNet data set when training on the real data set. However, for the simple data set the MobileNet was not pretrained due to the fact that there are no similarities between the images in ImageNet and the simple ReID images. Because the MobileNet was untrained when creating a filter for the simple data set the network had to train much longer. Since the input to the ReID is pairs of images the first part of the network consists of two instances of the MobileNet sharing the same weights, taking an image each as input. In practice this is equivalent to running the inputs through one MobileNet one at a time. The feature vectors produced by the MobileNet are then subtracted. After this, the resulting vector is squared elementwise which gives the euclidean distance between the feature vectors. This distance is then fed into a dense layer, with a sigmoid activation function, resulting in a number between 1 and 0 representing the likelihood that the images depicts the same identity.

4.1.3 MobileNet

MobileNet is a network which can be trained for a variety of applications, some of which are object detection, object classification and recognising face attributes [9]. Since there exists pretrained weights for the network it can often be useful as a base for a network. In all instances except for when training the ReID using the simple data set, the MobileNets in the detector and the ReID were pretrained using images from the ImageNet database. As a result the detector and the ReID were partly trained once the training in this project begun.

MobileNet is a lightweight and efficient convolutional neural network. It works slightly different from regular convolutional neural networks previously described. Instead of regular convolution it uses something called depthwise separable convolution. In depthwise separable convolution there are two different convolutions: depthwise convolutions and pointwise convolutions. The function of these are explained well in the paper which presented the MobileNet:

> “the depthwise convolution applies a single filter to each input channel. The pointwise convolution then applies a 1×1 convolution to combine the outputs [of] the depthwise convolution. A standard convolution both filters and combines inputs into a new set of outputs in one step. The depthwise separable convolution splits this into two layers, a separate layer for filtering and a separate layer for combining. This factorization has the effect of drastically reducing computation and model size.” [9].

Simply put, in depthwise separable convolution, the channels of an image is filtered and combined in two separate steps instead of one (as in regular convolutions), which makes it faster.

4.1.4 Filter

In this project several different filter structures were tested. Each of the filters consists of five convolutional layers. All layers are comprised of kernels of size 3×3, however, the number of kernels in the layers varies between 3 and 9. In the last layer there are always three kernels, giving three output channels representing the RGB channels of standard images. The activation function in the first four layers is ReLU and for the last layer it is a sigmoid. No biases are used in order to keep the number of weights down, the images are padded and the stride is 1. The reason for having a stride of 1 in every layer of the filter is so that there will be no downsampling throughout the convolutions; simply put, the image will not shrink by being filtered.

4.1.5 The variations of filters and training methods tested

There are a few parameters that were varied in order to examine whether an anonymisation filter can be created as proposed in this project and which set of parameters might be the better approach. The same number of kernels was used for both the filter variations tested on the simple data set, needed to find a working filter to verify the method. In the different layers the number of kernels were 5 in the first layer, 7 in the second and third, 5 in the fourth and 3 kernels in the last layer. The difference between the first and second attempt was that the ReID trained twice for every time the detector was trained in the second attempt instead of once, see Table 4.1. The reason for making this variation out of all the possible ones were that the detector converged twice as fast as the ReID when the networks were trained on unfiltered images. Such a difference was not observed when the detector and ReID were trained on real data. Therefore the number of times the ReID was trained in succession within the training loop was just one, for all of the filters created using the real data set. The parameters that instead were altered were the number of kernels in each layer and the learning rate of the filter. The exact combinations that were tested can be seen in Table 4.2 below.

Name	**Data Set**	**Filter Kernels**	**ReId trainings/ Detector training**
Filter S1	Simple	5,7,7,5,3	1/1
FIlter S2	Simple	5,7,7,5,3	2/1

Table 4.1 – The different filters trained on the simple data set. The learning rate was 0.0005 for each filter.

Name	**Data Set**	**Filter Kernels**	**Learning Rate**
Filter R1	Real	5,7,7,5,3	0.0005
Filter R2	Real	3,3,3,3,3	0.0005
Filter R3	Real	3,5,5,3,3	0.0005
Filter R4	Real	7,9,9,7,3	0.0005
Filter R5	Real	5,7,7,5,3	0.005
Filter R6	Real	5,7,7,5,3	0.0001

Table 4.2 – The different filters trained on the real data set. The number of subsequent times the ReID trained each loop was 1.

4.2 Evaluation Methods

To measure the performance of the different parts of the network a method of evaluation is needed. The evaluation method used to determine the performance of the filter must evaluate the detector and the ReID since the detector ideally should perform well while the ReID should not be able to. The methods for evaluating these parts differ quite a bit and will be explained below, starting with an explanation of true and false positives and negatives. These concepts are required in order to describe the evaluations used which are based on precision and recall. For the detector performance a method called average precision is used and for the ReID the performance is measured using the area under the ROC-curve.

4.2.1 Precision, Recall and False positive rate

To understand precision, recall and false positive rate one must know what true and false positives and negatives are. These are categories which the networks predictions of the data can be split into. Positive and negative represent what prediction the network makes and are best explained with an example. Using the ReID from this project as the example, a positive prediction would mean that the network guessed that a pair of images depicted the same identity and a negative prediction would mean that it guessed that the images depicted different identities. However, what is considered a positive and negative prediction differs depending on the problem. It also depends on what is being tested, for example one could just as well have used a network which was supposed to find people having different identities, as opposed to the ReID. For such a network which predictions would be considered positive and negative would be reversed. True and false are properties of the data points which depends on the prediction by the network. Correct predictions are called true and incorrect predictions are called false. Thus a positive data point which was predicted to be positive is called a true positive whilst it would be a false negative if it was predicted to be negative. False positives and true negatives are of course created using the same principle.

Precision is the ratio of correctly guessed positives. In other words, the number of true positives divided by the number of true and false positives. Recall is the ratio of correctly predicted positives among all predictions that should be positive. It is calculated by taking the number of true positives divided by the number of true positives and false negatives. See Figure 4.3 for a graphic explaining the definitions of precision and recall.

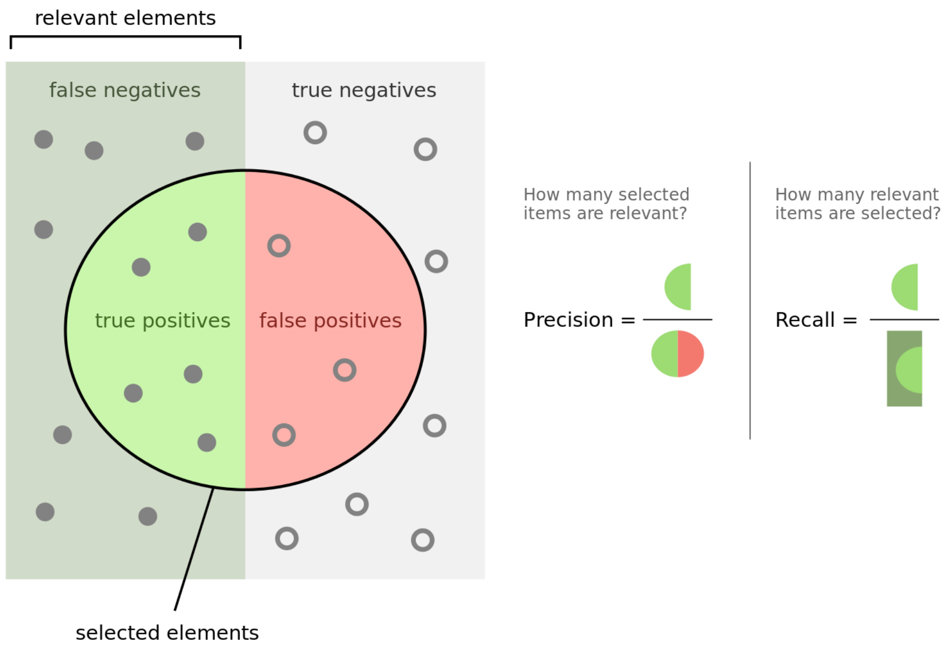


Figure 4.3 – How to calculate precision and recall [25].

The false positive rate, FPR, is the number of false positives divided by the sum of the number of false positives and true negatives,

$$FPR = \frac{FP}{FP + TN}. \tag{5}$$

It could also be represented by the red semicircle divided by the right side of the image in the graphics above, 4.3.

4.2.2 Detector Metric: Average Precision

To determine if a prediction by the detector is a true or false, positive or negative, a comparison is made between the heatmap generated by the detector and the heatmap used as label. Both the heatmaps are first maxpooled, then the maxpooled maps and the corresponding original heatmaps are compared to each other pixel by pixel. To keep track of the largest values in the images two mask images are created. If a pixel has the same value in both the original and the corresponding maxpooled map (indicating a local maximumum), as well as having a value of 0.5 or more, the pixel and all surrounding pixels are marked in the mask. If the euclidean distance between the coordinates of a marked pixel in the label mask and the generated mask is less than 4 pixels it is counted as a correctly detected pixel, these are true positives. The marked pixels in the generated map that are more than 4 pixels away from a marked pixel in the label map are false positives. The false negatives are marked pixels in the label map without any marked pixels in the generated map less than 4 pixels away. Precision and recall are calculated as described above for every batch.

When evaluating the detector the precision and recall is used to calculate the interpolated average precision. Average precision is the area under the curve when precision is plotted as a function of recall using the metric values recorded for each epoch. An image illustrating the interpolation of an average precision curve can bee seen below, see Figure 4.4. Average precision is defined as

$$AveP = \int_0^1 p(r)dr,$$

where r is the recall values and $p(r)$ is precision expressed as a function of recall. In practice the integral is replaced by a sum over the existing measurements. When calculating the interpolated average precision $p_{interp}(r)$ is set to the largest precision value, $p(\tilde{r})$, among all values of $\tilde{r}$ greater or equal to r. Let R be all the measured values of the recall then the interpolated average precision can be written as

$$AveP_{interp} = \sum_{r \in \{R\}} p_{interp}(r).$$

Mathematically $p_{interp}(r)$ can be expressed as

$$p_{interp}(r) = \max_{\tilde{r}:\tilde{r} \geq r} p(\tilde{r}).$$

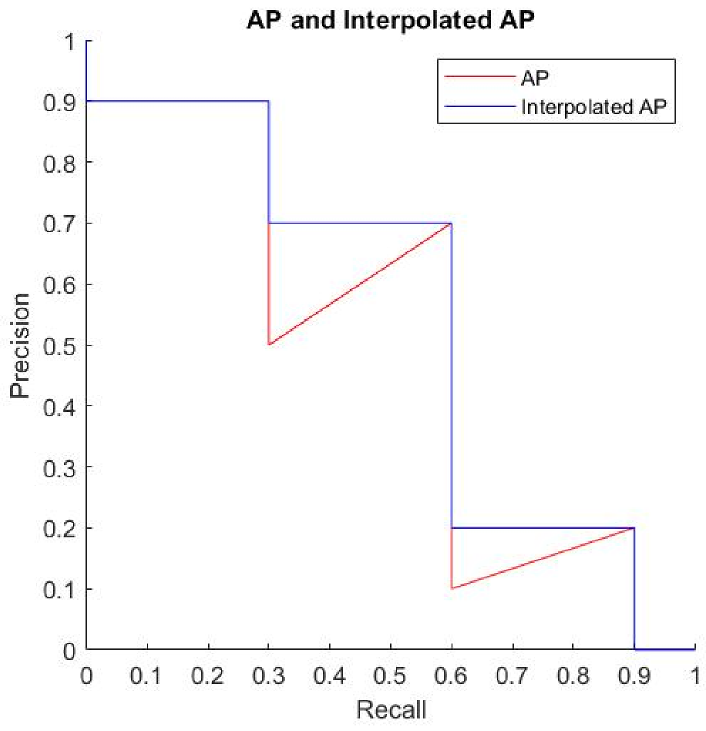


Figure 4.4 – An example plot showing the relation between AP and interpolated AP. The interpolated average precision is set to the largest precision value among all recall values greater or equal to the current value.

4.2.3 ReID Metric: AUC-ROC

As mentioned above the output of the ReID consists of a number between 0 and 1. If the output is a number above 0.5 it is interpreted as a positive prediction. The closer to 0 or 1 the output is the more confident the network is regarding its guess. This means that a prediction is a true positive if the two input images to the ReID depicts the same identity and the output has a value above 0.5, if the output is below 0.5 it is a false negative. True negatives are when the inputs are of different identities and the output has a value below 0.5, if it is above 0.5 it is a false positive. Recall is calculated using the standard formula described above.

The metric used when evaluating the ReID was the area under the receiver operating characteristic curve, the ROC-curve. The ROC-curve is obtained by plotting the recall against the false positive rate. This plot is created by changing a threshold for which output values are considered to be a positive guess by the network. By changing this threshold and getting different ratios between positive and negative guesses the recall and FPR changes as well. When the threshold is low the recall and FPR becomes higher since it is more probable that the output will be above the threshold resulting in a higher rate of positive guesses against negative ones. The first threshold is 0 and the corresponding point in the curve is the recall plotted against the FPR when every output from the ReID larger than 0 is considered a positive guess. The next point in the curve is calculated the same way but with a slightly larger threshold and so on until it reaches 1. For an example of a ROC-curve see Figure 4.5. In this project the default function in Tensorflow for creating a ROC-curve was used.

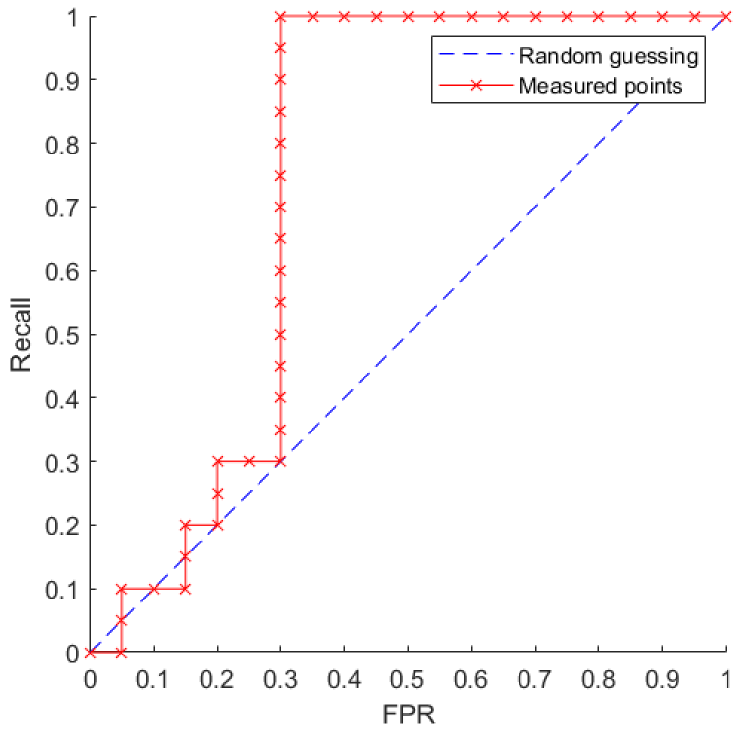


Figure 4.5 – An example of a ROC-curve where the data labels are thresholds. The blue line along the diagonal is the expected result when guessing randomly.

5 Results

The results in terms of the AP score for the detector and the AUC-ROC for the ReID, when trained on the two different data sets separately without the filter, can be seen in Table 5.1. Results for the different filters can be seen in the Table 5.2. In the table the filter performance can also be seen, it is the euclidean distance between the ideal performance and the result of a filter. An ideal filter performance is considered to be a score of 0.5, since that is no better than random guessing, for the ReID and the same score as the detector trained without a filter for the detector. The results for the different filters can also be seen in the graph 5.1 for the simple data set and 5.2 for the real data set. The closer a filter performance is to the ideal performances, the points (0.5,1) and (0.5,0.938), for the simple and real data set respectively, the better the filter is.

Data Set	**Detector performance (AP)**	**ReID performance (AUC-ROC)**
Simple	1.00	0.993
Real S2	0.938	0.696

Table 5.1 – The scores for the detector and the ReID when trained without the filter.

Name	**Detector performance (AP)**	**ReID performance (AUC-ROC)**	**Filter performance**
Filter S1	0.216	0.500	0.784
Filter S2	0.821	0.372	0.220
Filter R1	0.864	0.540	0.084
Filter R2	0.286	0.573	0.656
Filter R3	0.347	0.500	0.591
Filter R4	0.000	0.530	0.938
Filter R5	0.000	0.500	0.938
Filter R6	0.482	0.504	0.456

Table 5.2 – The scores for the detector, ReID and filter for the different filters.

When evaluating the method using the simple data set the first successful filter was ”Filter S2”, which had a score of 0.220. The filter had 5 layers with the number of kernels being 5,7,7,5 and 3, the ReID trained two times in succession each training loop and the learning rate of the filter was 0.0005. As one can see in 5.2 and the graph, the best performing filter on real data is ”Filter R1” with a score of 0.084. It had 5 layers with the number of kernels being 5,7,7,5 and 3, the ReID trained only once each training loop and the learning rate of filter was 0.0005. Examples of images produced by these filters can bee seen below, 5.3 and 5.4. The prediction by the ReID on these examples were 0.537 for the image pair from the simple data set and 0.510 for the image pair from the real data set. Both of these numbers are close to 0.5, these predictions implies that the ReID was unsure whether the images depict the same person or not in these particular pairs. For examples of images produced by the remaining filters, see Appendix A.

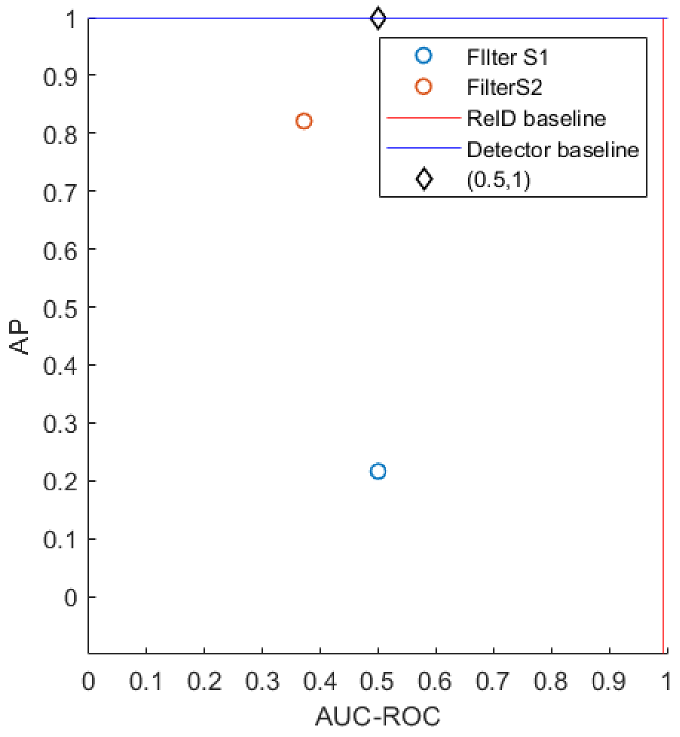


Figure 5.1 – The performance of the different filters trained on the simple data set. The baseline values are how well the detector and the ReID performed when trained on their own without the filter. The point (0.5,1), represented by a rhomboid, is the ideal result.

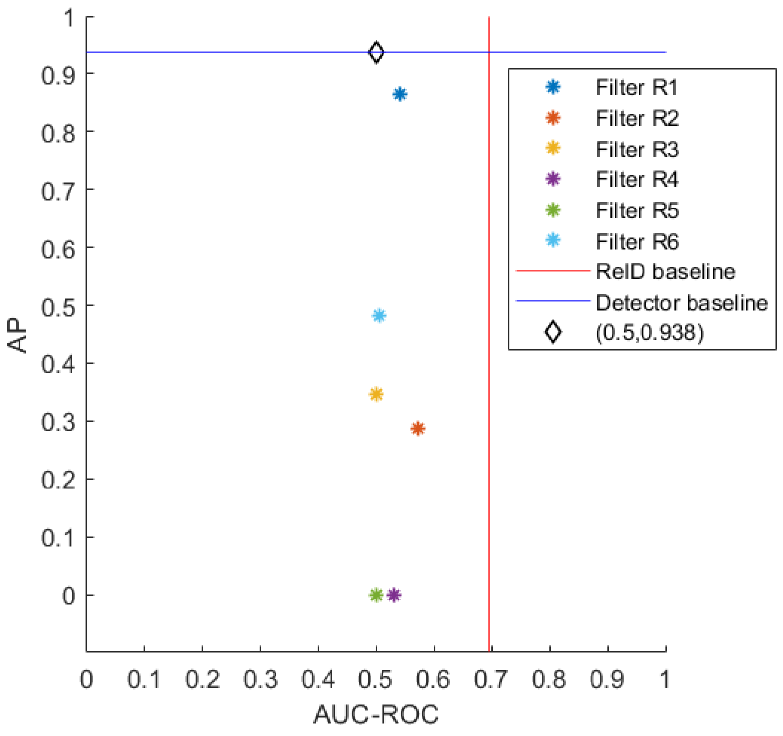
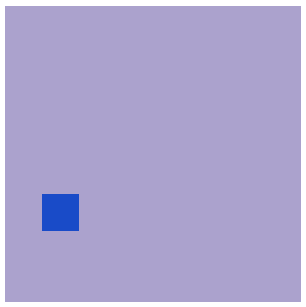
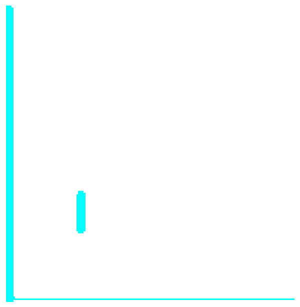
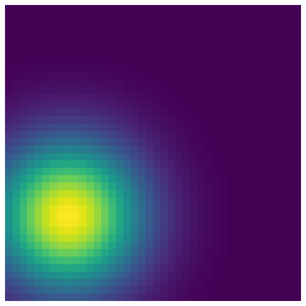
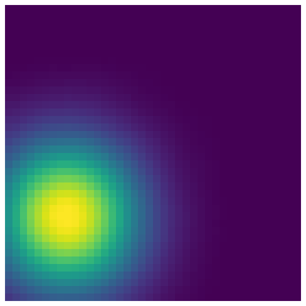


Figure 5.2 – The performance of the different filters trained on the real data set. The baseline values are how well the detector and the ReID performed when trained on their own without the filter. The point (0.5,0.938), represented by a rhomboid, is the ideal result.


(a) Input to detector


(b) Filtered detector image


(c) Heatmap


(d) Generated heatmap

(e) ReID input1

(f) Filtered ReID image1

(g) ReID input2

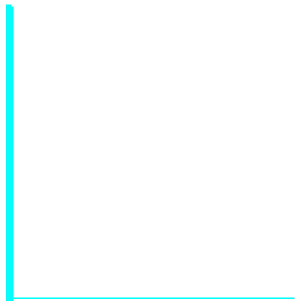
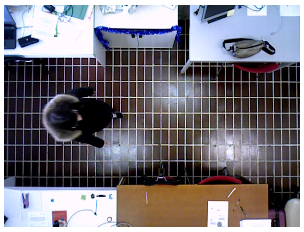
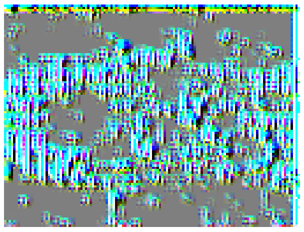
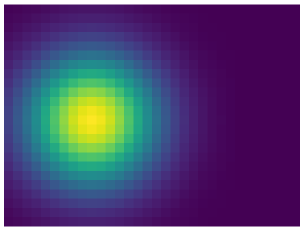
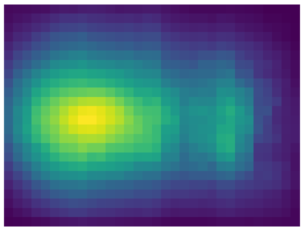

(h) Filtered ReID image2

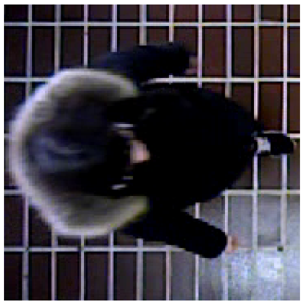
Figure 5.3 – The results from filtering a detector and ReID image from the simple data set as well as the heatmap used as ground truth and the generated heatmap. The images to the left are original images and to the right are the corresponding images after they have been anonymised or in the case of the heatmap been generated by the detector. The prediction of the ReID regarding images f and h was 0.537.

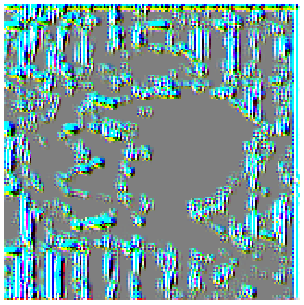

(a) Input to detector

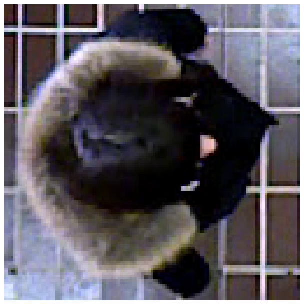

(b) Filtered detector image


(c) Heatmap


(d) Generated heatmap


(e) ReID input1


(f) Filtered ReID image1


(g) ReID input2

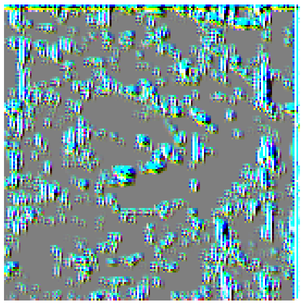

(h) Filtered ReID image2

Figure 5.4 – The results from filtering a detector and ReID image from the real data set as well as the heatmap used as ground truth and the generated heatmap. The images to the left are original images and to the right are the corresponding images after they have been anonymised or in the case of the heatmap been generated by the detector. The prediction of the ReID regarding images f and h was 0.501.

6 Discussion

As can be seen in the results, the aim of producing a filter which anonymises image data while preserving enough features for person detection using deep learning can be considered achieved. The reason being that for both the simple data set and the more realistic one, filters were produced which achieved high AP scores and AUC-ROC scores of around 0.5. Results like these suggests that the person detector performs very well on the filtered images and the performance of the ReID is no better than random guesses. Even though the results most likely still can be improved, and would have to be before the filter could be used in any practical applications, it can be assumed that the methodology used in this project can be used to create an anonymisation filter.

6.1 General discussion

There are several things worth noting about the method used to produce the filter as well as the training process and the results. One of these is that fairly small changes in the setup of the network had a very large impact on the results, as can be seen in the results graph 5.2. One interpretation of this behavior could be that the method used to create the filters in this project is quite sensitive and might as a result be hard to train well. This would not be very surprising since the very similar GAN structure is notoriously hard to train [11]. Also worth noting is that when using the realistic data set some overfitting was observed, for example when training the ReID without filtering the images, and it is therefore very likely that the results would improve by training on a larger data set. For the simple data set no overfitting was observed due to the fact that new data was created continuously throughout the training.

Another topic of discussion is that there is a possibility that a seemingly well performing filter is a result of a generous filter and a relatively fast detector compared to the ReID. All filters were trained longer than the time it took for the ReID, trained on unfiltered images, to converge. However, there is not a lot of insight to be had into what happens during the training process when the networks are trained together. Thus it is hard to tell whether the training time needed for the ReID to converge is changed compared to when it is trained on unfiltered images. The ReID did generally not produce high AUC-ROC scores at any point during the training. Therefore, regardless of the results produced, one can always question whether the ReId had a chance to learn to produce good results.

Also worth noting is that the task given to the detector when using the real data set probably is easier than when training an average person detector while the task given to the ReID most likely is harder. The reason for this is that the person detector is given images of people where the background remains exactly the same and the only aspect of the image that changes is whether or not there is a person in the image. Furthermore there is only one person in each image, no more no less. For the ReID however the task is to determine whether or not two persons are the same from a top down perspective instead of from a side angle, which is harder than what most ReIDs are trained to do. On the other hand, people are depicted in higher resolution in the ReID images than in the Detector images giving the ReID more information to base its predictions on than the detector.

Worth commenting on is also the fact that the ReID was trained to determine whether or not two filtered images depicted the same person or not. One could imagine that its task just as well could have been to judge one filtered image against an unfilterd one. This would have made a slight difference in the function of the ReID. In this project the purpose of the ReID was to make sure the filter is created in such a way that filter representations of images of the same person cannot be identified as such. If instead one of the images given to the ReID would have been unfiltered it is possible that the filter could have produced filter representations where it was possible to tell if filtered images depicted the same identity or not. The issue with such filter representations is that if the identity of a person in one filtered image is known it could be matched with other images of the same individual and thus leaving them identifiable. On the other hand it could be useful in the instances tracking of individuals are needed but their identities are irrelevant. Even so this method of training the ReID was considered a weaker form of anonymity and was not used in this project.

Finally, it is also worth mentioning that the way the filter was created makes it very task specific. It is only regulated whether or not people can be found or identified in images. As a result it is likely that information about objects or other kinds of life are lost in the filtered images, examples of this can be seen in our results, see image 5.4. Since the filter only was tasked with specifically making people anonymous it is also not unimaginable that other sensitive information could be left in the images such as registration numbers on vehicles. It could therefore not be expected to perform anonymisations on anything other than people without being retrained on the new task.

6.2 What is Anonymity?

Even considering the results and that one accepting that the network has done its job and created a anonymisation filter as intended, despite the following discussion, one might still question whether or not the images produced by the filter is in fact anonymous. According to Merriam-Webster's dictionary the definition of anonymous is "not named or identified" or alternatively "lacking individuality, distinction, or recognizability" [15]. In the recent GDPR legislations personal data, which can be considered nonanonymous data, is defined as:

> "any information relating to an identified or identifiable natural person ('data subject'); an identifiable natural person is one who can be identified, directly or indirectly, in particular by reference to an identifier such as a name, an identification number, location data, an online identifier or to one or more factors specific to the physical, physiological, genetic, mental, economic, cultural or social identity of that natural person [10]."

In both definitions the essence of anonymity seems to be that one can not be recognised and matched to a specific identity. The question is to whom or what this applies. Many of the images produced by already existing anonymisation filters which only distorts faces could be considered anonymous to most people. Erasing the face of an unfamiliar individual in an image limits the possibility of identification. However if the image depicts a known person, a distorted face will most likely not prevent the person from being identified. One could even imagine that people with close relations could identify each other by movement pattern alone, performed for example by an avatar. Is an image anonymous if even one human can recognize the person in the image? What if no one can identify a person by eye but provided the right skills and a regular computer they can? Is an image anonymous if one needs a computer with extraordinary computing power? This is a complex question that cannot be answered in this project. Perhaps the definition of anonymity depends on the situation and the level of precaution needed given the circumstances, perhaps it does not.

The filter in this project was created using deep learning and regular computers. Therefore, it can be considered a step towards creating a filter which produces images that are anonymous enough so that at least having the right skill set and a regular computer is not enough for identifying people. Hopefully this method of creating a filter could even produce filters where two different images could have the same filtered image. Such a filter would probably be considered a perfect anonymisation filter since the identification of anonymised people would be ambigous even using an extraordinarily powerful computer. But even given an image produced by a perfect anonymisation filter, if one knows the time and place the picture was taken and who usually goes there at that time, an anonymised image could potentially give information about an individual. However there is still a point in trying to anonymise personal data as much as possible. It is better to use a filter to save an image where it is difficult to recognise people than an image where just looking would be enough to recognise someone. While doing so, it is vital to be aware that using current anonymisation methods, for example pixelating faces, does not make your data anonymous to everyone.

6.3 Future Work

While the aim of producing an anonymisation filter using deep learning was achieved, there is room for further studies. It would for example be of interest to try to improve the performance of the filter. One of the most obvious ways of doing this for the filter using real data, is to acquire more data. Also, to improve the filter performance one could try to further tune parameters and hyperparameters. It would be beneficial to also unravel what parts of the network or training process it is that make the network sensitive to small changes in the setup, in order to gain more control over the training process. Furthermore, it would be relevant to examine how using other detectors and ReIDs would affect the forming of a filter. An interesting modification would be to ad yet another addition to the loss function for the combined model, which encourage similarity between the the filtered images and the originals. Such an addition might combat the loss of information in the filtered images that does not regard humans. However it could potentially also make the model even harder to train.

Before using this method further it is probably also wise to test how well the filter performs to a greater extent, for example by training a ReID on filtered images from the finalised filter in order to compare the results of this ReID to the performance of the ReID included in the combined model. Even better would be to determine if images depicting two different identities could be projected to the same filtered image and thereby determine that an identification would be ambiguous.

Another possibility for further studies is to create a filter which works for video data. In such a filter the amount of identifiable features would be greater, including all kinds of movement patterns, and thus the demands on the filter would be higher. Once such a filter was implemented it would be of interest to minimise the number of weights while maintaining the performance of the filter to determine if it could be implemented in a product with limited computing power such as a regular surveillance camera.

It is also of interest to research other applications, than anonymisation, for this sort of GAN-like network which finds a distribution for which it knows a few characteristics instead of emulating one, like a typical GAN would.

6.4 Conclusions

It is possible to use a deep learning method to anonymise image data while still preserving enough features for person detection, without using a detector to target certain parts of the image to modify. However as mentioned above, the specifics of the method needs to be further developed in order to be able to produce a filter which can be used in practise.

References

[1] Aphex34. *Typical CNN, licensed under CC BY-SA 4.0 https://creativecommons.org/licenses/by-sa/4.0/deed.en*. Dec. 2015. URL: https://commons.wikimedia.org/w/index.php?curid=45679374.

[2] Zhaowei Cai and Nuno Vasconcelos. "Cascade R-CNN: High Quality Object Detection and Instance Segmentation". In: *CoRR* abs/1906.09756 (2019). arXiv: 1906.09756. URL: http://arxiv.org/abs/1906.09756.

[3] Nikhil Chaabra. *Generative Adversarial Networksfor Image Anonymization*. May 2017. URL: http://www.morrisriedel.de/wp-content/uploads/2019/05/2017-Master-Thesis-Nikhil-Chhabra.pdf.

[4] Jimmy Ba Diederik P. Kingma. "Adam: A Method for Stochastic Optimization". In: *CoRR* abs/1412.6980 (2017). arXiv: 1412.6980. URL: https://arxiv.org/abs/1412.6980.

[5] Kaiwen Duan, Song Bai, Lingxi Xie, Honggang Qi, Qingming Huang, and Qi Tian. *CenterNet: Keypoint Triplets for Object Detection*. 2019. arXiv: 1904.08189 [cs.CV]. URL: https://arxiv.org/abs/1904.08189.

[6] Ross B. Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. "Rich feature hierarchies for accurate object detection and semantic segmentation". In: *CoRR* abs/1311.2524 (2013). arXiv: 1311.2524. URL: http://arxiv.org/abs/1311.2524.

[7] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. URL: http://www.deeplearningbook.org.

[8] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross B. Girshick. "Mask R-CNN". In: *CoRR* abs/1703.06870 (2017). arXiv: 1703.06870. URL: http://arxiv.org/abs/1703.06870.

[9] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications". In: *CoRR* abs/1704.04861 (2017). arXiv: 1704.04861. URL: http://arxiv.org/abs/1704.04861.

[10] Richie Koch. *What is considered personal data under the EU GDPR?* 2020. URL: https://gdpr.eu/eu-gdpr-personal-data/.

[11] Sebastian Nowozin Lars M. Mescheder Andreas Geiger. "Which Training Methods for GANs do actually Converge?" In: *CoRR* abs/1801.04406 (2018). arXiv: 1801.04406. URL: http://arxiv.org/abs/1801.04406.

[12] Hussam Lawen, Avi Ben-Cohen, Matan Protter, Itamar Friedman, and Lihi Zelnik-Manor. *Attention Network Robustification for Person ReID*. 2019. arXiv: 1910.07038 [cs.CV]. URL: https://arxiv.org/abs/1910.07038v2.

[13] Daniele Liciotti, Marina Paolanti, Emanuele Frontoni, Adriano Mancini, and Primo Zingaretti. "Person Re-identification Dataset with RGB-D Camera in a Top-View Configuration". In: *Video Analytics. Face and Facial Expression Recognition and Audience Measurement: Third International Workshop, VAAM 2016, and Second International Workshop, FFER 2016, Cancun, Mexico, December 4, 2016, Revised Selected Papers.* Ed. by Kamal Nasrollahi, Cosimo Distante, Gang Hua, Andrea Cavallaro, Thomas B. Moeslund, Sebastiano Battiato, and Qiang Ji. Cham: Springer International Publishing, 2017, pp. 1–11. ISBN: 978-3-319-56687-0. DOI: 10.1007/978-3-319-56687-0_1. URL: http://dx.doi.org/10.1007/978-3-319-56687-0_1.

[14] Niki Martinel, Gian Luca Foresti, and Christian Micheloni. "Aggregating Deep Pyramidal Representations for Person Re-Identification". In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops.* June 2019. URL: http://openaccess.thecvf.com/content_CVPRW_2019/html/TRMTMCT/Martinel_Aggregating_Deep_Pyramidal_Representations_for_Person_Re-Identification_CVPRW_2019_paper.html.

[15] Merriam-Webster. *anonymous.* 2020. URL: https://www.merriam-webster.com/dictionary/anonymous.

[16] European Parliament. *Europaparlamentets och Rådets Förordningar (EU) 2016/679.* 2016. URL: https://eur-lex.europa.eu/legal-content/SV/TXT/PDF/?uri=CELEX:32016R0679&rid=1.

[17] Joseph Redmon, Santosh Kumar Divvala, Ross B. Girshick, and Ali Farhadi. "You Only Look Once: Unified, Real-Time Object Detection". In: *CoRR* abs/1506.02640 (2015). arXiv: 1506.02640. URL: http://arxiv.org/abs/1506.02640.

[18] Joseph Redmon and Ali Farhadi. "YOLO9000: Better, Faster, Stronger". In: *CoRR* abs/1612.08242 (2016). arXiv: 1612.08242. URL: http://arxiv.org/abs/1612.08242.

[19] Joseph Redmon and Ali Farhadi. "YOLOv3: An Incremental Improvement". In: *arXiv* (2018). URL: https://pjreddie.com/media/files/papers/YOLOv3.pdf.

[20] Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks". In: *CoRR* abs/1506.01497 (2015). arXiv: 1506.01497. URL: http://arxiv.org/abs/1506.01497.

[21] Zhongzheng Ren, Yong Jae Lee, and Michael S. Ryoo. "Learning to Anonymize Faces for Privacy Preserving Action Detection". In: *CoRR* abs/1803.11556 (2018). arXiv: 1803.11556. URL: http://arxiv.org/abs/1803.11556.

[22] Johan Rodhe and Gustaf Oxenstierna. *Estimating Median Visiting Times using Re-identification.* eng. Student Paper. 2018. URL: https://lup.lub.lu.se/student-papers/search/publication/8945431.

[23] Natacha Ruchaud and Jean-Luc Dugelay. "Automatic face anonymization in visual data: Are we really well protected?" In: *EI 2016, ISAT International Symposium on Electronic Imaging, February 14-18, 2016, San Francisco, USA.* San Francisco, UNITED STATES, Feb. 2016. URL: http://www.eurecom.fr/publication/4914.

[24] Mark Sandler, Andrew G. Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. "Inverted Residuals and Linear Bottlenecks: Mobile Networks for Classification, Detection and Segmentation". In: *CoRR* abs/1801.04381 (2018). arXiv: 1801.04381. URL: http://arxiv.org/abs/1801.04381.

[25] Walber. *Precisionrecall.svg, licensed under CC BY-SA 4.0 `https://creativecommons.org/licenses/by-sa/4.0/deed.en`*. Nov. 2014. URL: `https://commons.wikimedia.org/wiki/File:Precisionrecall.svg`.

[26] Guangcong Wang, Jianhuang Lai, Peigen Huang, and Xiaohua Xie. "Spatial-Temporal Person Re-identification". In: *CoRR* abs/1812.03282 (2018). arXiv: `1812.03282`. URL: `http://arxiv.org/abs/1812.03282`.

Appendix A

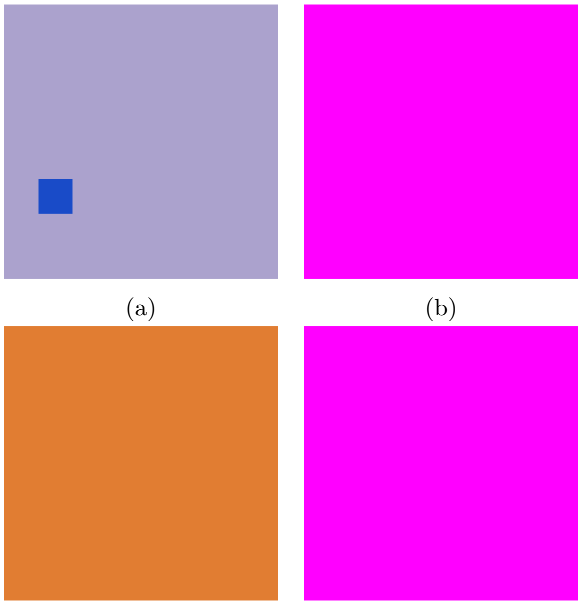


Figure A.1 – The results from filtering a detector and a ReID image from the real data set using filter F1. The left image is the input to the filter and the right image is the image after it has been anonymised.

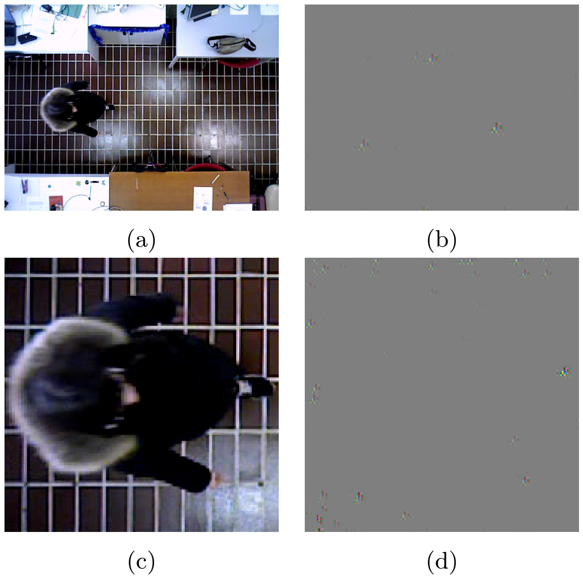


Figure A.2 – The results from filtering a detector and a ReID image from the real data set using filter R2. The left image is the input to the filter and the right image is the image after it has been anonymised.

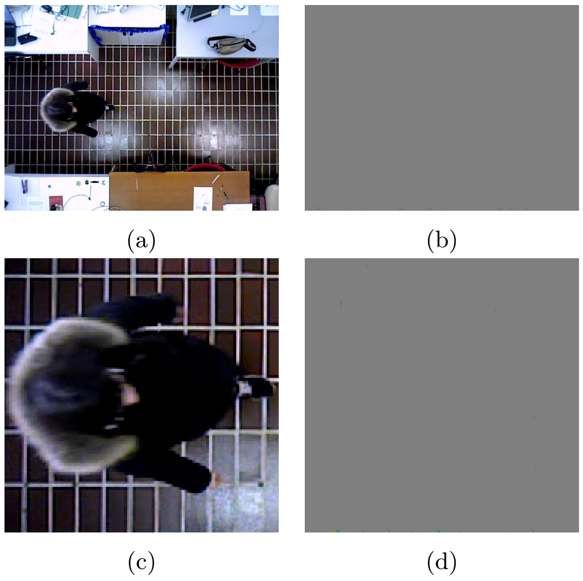
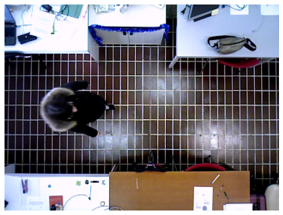
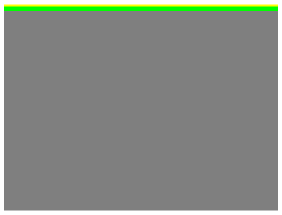
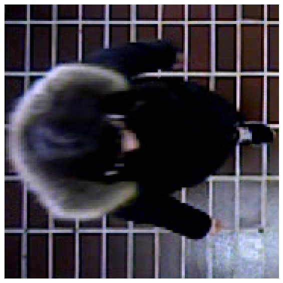
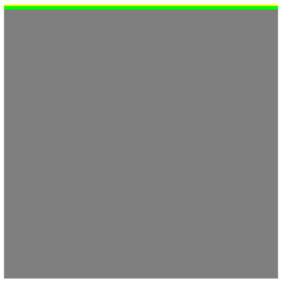


Figure A.3 – The results from filtering a detector and a ReID image from the real data set using filter R3. The left image is the input to the filter and the right image is the image after it has been anonymised.

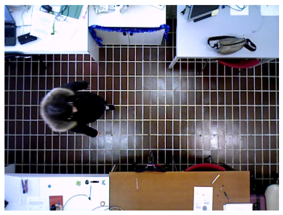
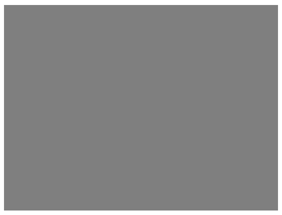



(a) (b)

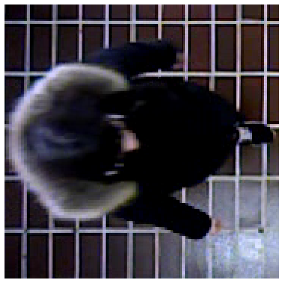
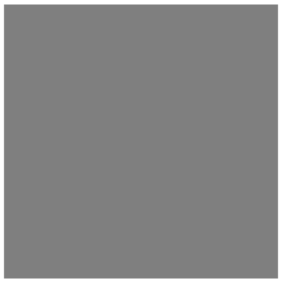



(c) (d)

Figure A.4 – The results from filtering a detector and a ReID image from the real data set using filter R4. The left image is the input to the filter and the right image is the image after it has been anonymised.

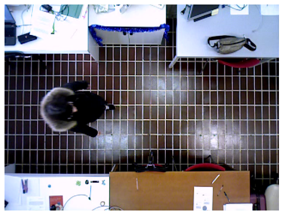



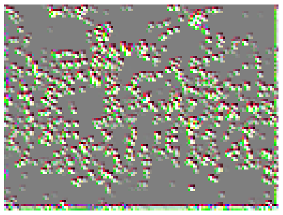
(a) (b)

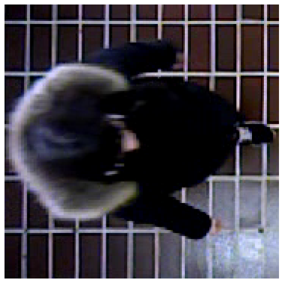



(c) (d)

Figure A.5 – The results from filtering a detector and a ReID image from the real data set using filter R5. The left image is the input to the filter and the right image is the image after it has been anonymised.


(a)


(b)


(c)

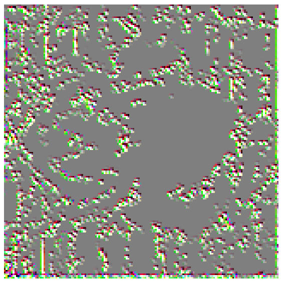

(d)

Figure A.6 – The results from filtering a detector and a ReID image from the real data set using filter R6. The left image is the input to the filter and the right image is the image after it has been anonymised.