Imperial College London

Department of Electrical and Electronic Engineering

Final Year Project 2019

Project Title:	**Domain transfer between images with GANs**
Student:	**Benedikt Kolbeinsson**
CID:	**01070796**
Course:	**4EM**
Project Supervisor:	**Dr Krystian Mikolajczyk**
Second Marker:	**Dr Javier A. Barria**

Abstract

Person re-identification (re-ID) is the problem of identifying the same person in multiple cameras. This is a non-trivial problem, that is confounded by non-overlapping field of view, lighting differences, occlusion, variation in poses and different camera viewpoints. Current person re-ID systems perform well on specific datasets but experience large performance drops when trained and tested on a different dataset. This report describes a new method to improve the robustness of person re-ID models. The proposed method generates new backgrounds using a generative adversarial network which allows person re-ID models to be trained on larger and more varied datasets, therefore improving robustness. Individual identities from the original dataset are recreated in new scenarios with corresponding labels, this allows person re-ID networks to utilise supervised learning on the generated data. Variations of the proposed method provide significant control over the generated images, from maintaining high similarity between the generated identities and their respective original (same pose) to generating the identity in any new pose while still maintaining significant similarities.

Acknowledgements

I would like to thank my supervisor, Dr Krystian Mikolajczyk, for his helpful advice and guidance throughout the project. I would also like to express sincere gratitude to Sara Iodice for the incredible support and practical tips.

Table of Contents

1 Introduction **5**
1.1 Aims and Objectives 5

2 Literature Review **6**
2.1 Generative Adversarial Networks 6
2.2 Pix2Pix 6
2.3 CycleGAN 8
2.4 SPGAN 9
2.5 CamStyle 10
2.6 Pose Transfer 11
2.7 Background Bias 11
2.8 Datasets 12

3 Design **14**
3.1 Overview 14
3.2 Model 1 15
3.3 Model 2 16
3.4 Model 3 17
3.5 Model 4 18

4 Implementation **20**
4.1 Data processing 20
4.2 Model 1 (Base) 21
4.3 Training the GAN 21
4.4 Model 2 (Input mask modification) 22
4.5 Model 3 (Input and output mask modification) 23
4.6 Foreground and Background Loss 23
4.7 Model 4 (Foreground and background loss) 25

5 Testing **27**

6 Results **30**
6.1 Image comparison and quality assessment 30
6.2 Nearest neighbour of generated images 32
6.3 Generating images for one camera 34
6.4 Extending a person re-ID dataset 34
6.5 Extending a large person re-ID dataset 36
6.6 Domain transfer from labelled to unlabelled datasets 37
6.7 Pose generation 38

7 Evaluation **39**

8 Conclusion and Further Work **41**

Appendix A Code **43**

Appendix B Additional results **46**

Appendix C Safety, Legal and Ethics **46**

List of Figures

1 Diagram of a generative adversarial network . . . 6
2 Example of results from Pix2Pix . . . 7
3 An illustration of CycleGAN . . . 8
4 Example images produced by CycleGAN . . . 9
5 Images generated using SPGAN . . . 10
6 Images generated using CamStyle . . . 10
7 Images generated using PoseTransfer . . . 11
8 Images with modified backgrounds . . . 12
9 Examples from well known person re-ID datasets . . . 13
10 Diagram of model 1 . . . 16
11 Image pair examples . . . 16
12 Examples of masks . . . 17
13 Diagram of model 3 . . . 17
14 Diagram of model 4 . . . 19
15 Implementation strategy . . . 20
16 Diagram of the foreground loss and the background loss . . . 25
17 Examples of generated images by each model . . . 31
18 Nearest neighbours for each model . . . 33
19 Performance of model 4 - camera pairs . . . 36
20 Performance of each model - all cameras . . . 37
21 Examples of generating images with varying pose . . . 38
22 Comparison between model 3 and state-of-the-art . . . 40

List of Tables

1 Comparison of person re-ID datasets . . . 13
2 Comparison of camera pair datasets . . . 22
3 Decomposition of selected camera pair datasets . . . 22
4 Single camera baselines . . . 28
5 Camera pair baselines . . . 28
6 All cameras baseline . . . 29
7 Performance of each model - one camera . . . 34
8 Performance of each model - camera pair . . . 35
9 Performance of model 4 - camera pairs . . . 35
10 Performance of each model - all cameras . . . 36
11 Performance of model 4 - unlabelled dataset . . . 37
12 Performance of model 4 - unlabelled dataset 2 . . . 38
13 Performance of model 1 - camera pairs . . . 46

1 Introduction

Cameras are a good way to deter crime and also help track people of interest. With the growing amount of data captured by cameras, manually tracking people has become exceptionally time consuming. Tracking one person is a difficult task, simultaneously tracking multiple persons across multiple scenes is orders of magnitude more difficult. Conventional object trackers are not effective in tackling this problem [1].

Person re-identification (re-ID) is the problem of identifying the same person across multiple cameras. For example, in a train station, a camera captures an image of a person waiting on a platform. Later, a second camera located in a train, captures an image of the same person on-board the train. Identifying that this is the same person in both images is a form of person re-ID.

Problems include non-overlapping field of view, lighting differences, occlusion, variation in poses and different camera viewpoints. Biometric markers such as facial features are not available due to low resolution. The only assumptions are that people will wear the same clothing in different sightings [2] and their body type (height and build) will remain the same.

State-of-the-art person re-ID systems, such as GLAD [3] and PDC [4], can achieve rank-1 accuracies of roughly 90% when tested on a specific dataset but experience severe performance drops when trained and tested on a different dataset [5]. This is due to the limited scale of datasets as they do not provide realistic nor sufficient data to allow for generalisation [5].

Gathering and annotating real data to create significantly large and varied datasets can be time consuming and costly. Therefore, the currently available datasets are small and homogeneous.

1.1 Aims and Objectives

The aim of this project is to generate new, hard cases to improve the robustness of current re-ID systems. Generating realistic images allows for current datasets to be extended and thus the current re-ID systems can be trained on more samples, improving the performance. One of the hardest aspects of generating re-ID images is labelling them. It is important to be able to generate realistic images of the same people as are already in the dataset and not a new label. This is to enable supervision during training of re-ID systems which leads to higher accuracies compared to unsupervised training.

The main objectives for the project are the following:

- Design a generative adversarial network to generate labelled images of existing persons with new backgrounds.
- Design a GAN to generate labelled images of existing persons with new poses.
- Expand a person re-ID dataset by adding images generated by the GAN.
- Improve on the person re-ID network baseline (ResNet [6]) by using the extended datasets.

2 Literature Review

This section is an overview of generative adversarial networks (GANs), both in general and specific re-ID GANs, current re-ID systems and information regarding re-ID datasets.

2.1 Generative Adversarial Networks

A Generative Adversarial Network (GAN) [7] is composed of two separate networks, a generator network and a discriminator network. The generator creates samples that mimic the training data while the discriminator determines whether a sample was generated (fake) or from the training data (real). Figure 1 shows how the generator and discriminator are connected.

By analogy, the generator can be viewed as a forger and the discriminator can be viewed as a detective. The forger is trying to forge paintings but at the start isn't very good. The detective can easily spot differences between the real paintings and the fake one. Eventually, the forger improves his skill and can produce better fake paintings. It is now more difficult to spot the difference between real and fake paintings and so the detective must improve his skill. This can go on until the detective becomes better than humans at detecting fakes and the forger can create paintings that are indistinguishable from the real paintings to humans.

The generator G maps a random noise vector z to match the distribution of the training data. The discriminator D distinguishes between real and fake data. The objective function can be written as:

$$\mathcal{L}_{GAN}(G, D) = \mathbb{E}_y[log(D(y)] + \mathbb{E}_z[log(1 - D(G(z))] \tag{1}$$

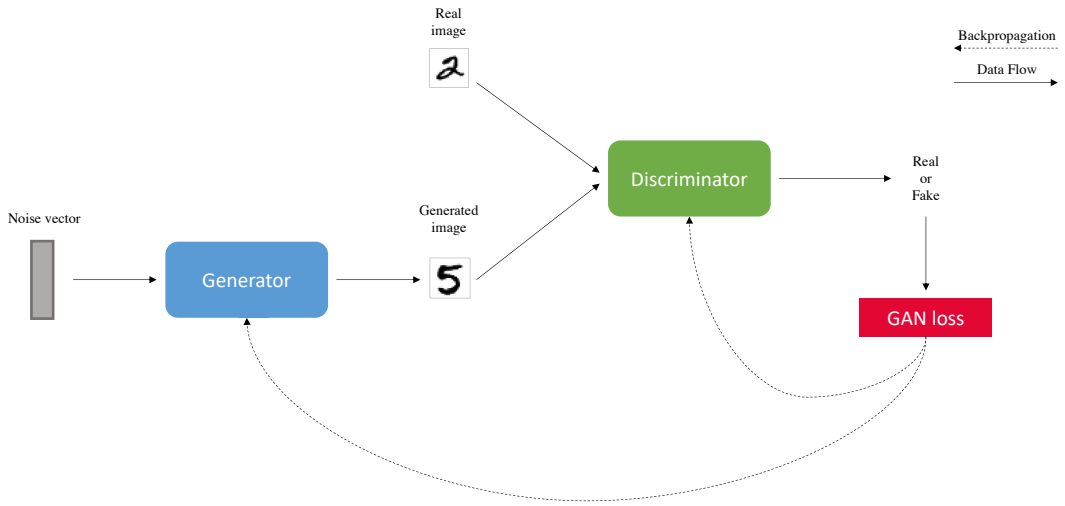


Figure 1: A diagram of a generative adversarial network (GAN) training to generate images of handwritten digits.

2.2 Pix2Pix

Pix2Pix [8] showed that conditional GANs [9] could generate realistic images. Conventional GANs learn to map an output image from a random noise vector whereas

conditional GANs learn to map an output image from an input image and a random noise vector. Pix2Pix was not tuned to be application specific and was shown to perform well on diverse sets of data.

Pix2Pix's generators utilise a U-Net [10] architecture. Generally speaking, this implies multiple layers that perform down sampling followed by multiple upsampling layers. This creates a bottleneck layer that has a compressed representation of the data in the middle. But importantly these layers contain skip connections, which allow more information to surpass the bottleneck.

What makes this a conditional GAN (cGAN) is that the input x for the generator is also used as one of the inputs for the discriminator. Thus: the objective of the cGAN is:

$$\mathcal{L}_{cGAN}(G, D) = \mathbb{E}_{x,y}[log(D(x, y)] + \mathbb{E}_{x,z}[log(1 - D(x, G(x, z))] \tag{2}$$

Where G and D are the generator and discriminator, respectively, and x, y and z are the input, output and random noise vector, respectively.

The noise input for the generator was created using dropout. The authors note that this is an area that could be further improved as the output was only slightly stochastic and is therefore not capturing the full entropy of the conditional distribution.

For the discriminator they developed PatchGAN, which restricts the discriminator to only model high frequency components then in addition use an L1 loss for the low frequencies.

With Pix2Pix being non-application specific the results can be described as acceptable. Figure 2 shows that Pix2Pix can generate sharp images. It is still far from perfect, especially generating the background/sky.

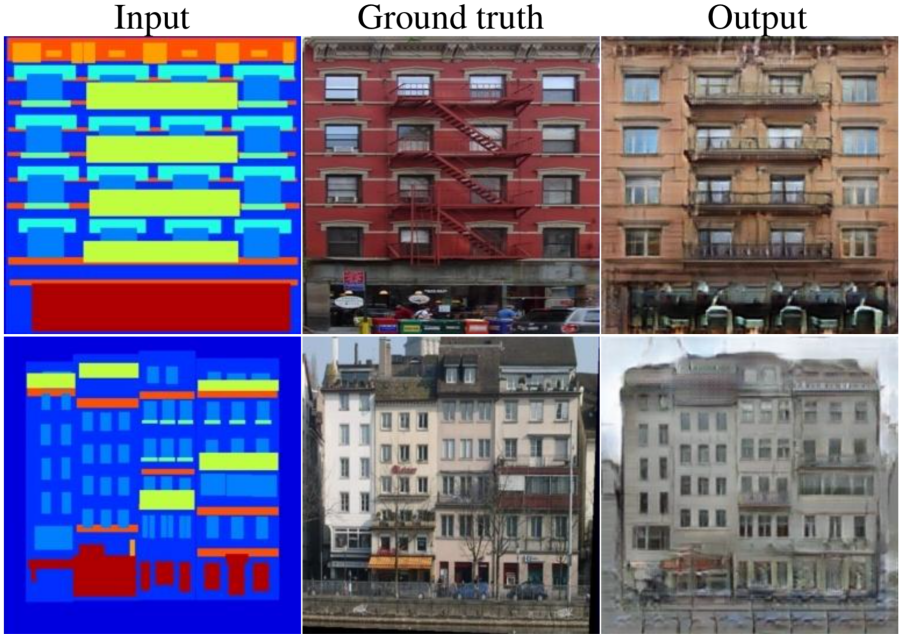


Figure 2: Example of the results from Pix2Pix when generating facades. Reproduced from the original paper [8].

2.3 CycleGAN

CycleGAN [11] became famous for its impressive results with diverse sets of images. CycleGAN learns to translate between domains without input-output pairs. This is a powerful tool as it does not require paired input-output examples to learn this translation but utilises supervision based on sets (set of images from different domains). It also aims to produce cycle consistent translation meaning after translating image x to image y, a reverse translation of image y will yield image x. This is visualised in figure 3.

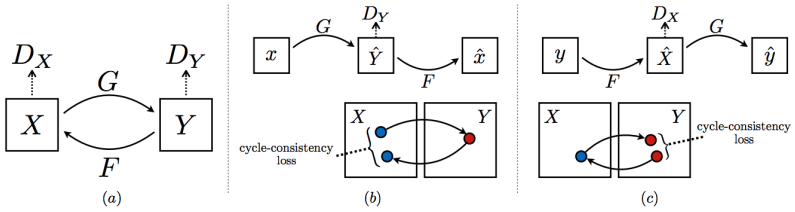


Figure 3: An illustration of CycleGAN reproduced from the original paper [11]. (a) shows the relationship between the two generators G and F, the two domains X and Y and the two discriminators D_X and D_Y. (b) shows the forward cycle-consistency loss and (c) shows the backward cycle-consistency loss.

At its core, CycleGAN utilises two generators, G: X $\rightarrow$ Y and F: Y $\rightarrow$ X, and two adversarial discriminators, D_X and D_Y. D_X discriminates between real images and generated images in set X while D_Y discriminates between real images and generated images in set Y. To incentivise the aforementioned cycle consistency, a cycle consistency loss is introduced.

Some examples of what CycleGAN can do are shown in figure 4. In the column on the right, CycleGAN was trained to change a horse to a zebra and vice versa. The output clearly captures the style of each set and generates fairly realistic images. The left column shows the style transfer from a Monet painting to a real photo. This also shows that CycleGAN can create a mapping between two styles and translate images between them.

The authors note that they were unsuccessful in generalising the transformation between styles of different geometric sizes or extreme geometric changes.

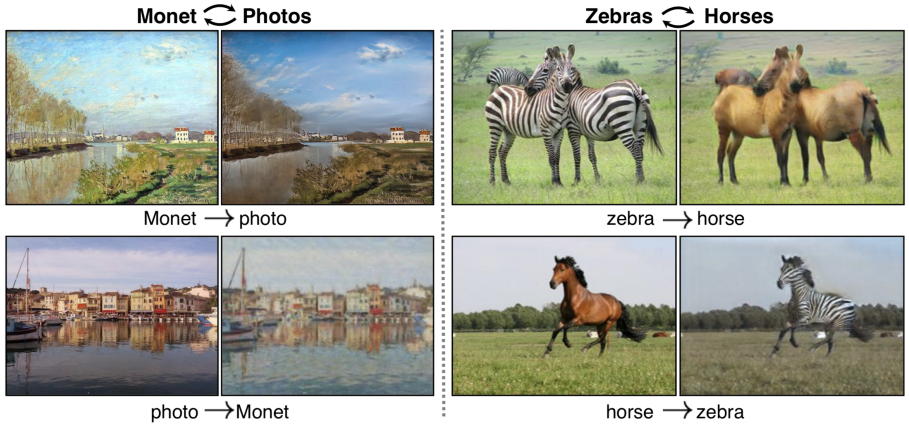


Figure 4: Example images of CycleGAN reproduced from the original paper [11]. The left column shows the transformation from a Monet painting to a realistic photo and vice versa. The right column shows the transformation from zebras to horses and vice versa.

2.4 SPGAN

Similarity Preserving cycle-consistent GAN (SPGAN) [12], aim to generate new images of people from a labelled dataset in the style of a different unlabelled dataset. There are two requirements for this to work, the ID in the new generated image should accurately represent the same ID as before the transformation and the ID being transferred should be dissimilar to other IDs in the unlabelled dataset. This transformation is unsupervised but permits re-ID models to be trained using supervised feature learning methods on the generated data.

SPGAN integrates a Siamese network [13], which constrains the learning of the mapping function, and CycleGAN, which learns the mapping between the domains. When training, there are two objectives for the two CycleGAN generators, represented by the functions G and F. First, given a source image x_S and target image x_T, the network is encouraged to pull $G(x_S)$ and $F(x_T)$ close to x_S and x_T respectively, as these image pairs contain the same identity. The second objective for the network is to push $G(x_S)$ and $F(x_T)$ away from x_T and x_S respectively, as these image pairs contain different identities.

Figure 5 shows a few examples of SPGAN in action. It is difficult to see a large difference in appearance but as the two domains are fairly similar this would be expected. To the human eye it is clear that the transformation does not change the underlying ID. Under closer inspection, the transformed images are slightly blurry. SPGAN did show a performance improvement of person re-ID compared to a direct transfer approach. Where a direct transfer approach uses a network trained on one dataset and not changed before being applied to the target dataset. As expected, a supervised approach still performs significantly better.

Figure 5: An example of SPGAN changing the style of images between two datasets, reproduced from the original paper [12]. Row (a) shows the original images. Row (b) shows the output from SPGAN after the transformation.

2.5 CamStyle

Camera style (CamStyle) [14] adaptation tries to minimise the disparities between different camera images. Two versions are proposed, the vanilla version which works well on small camera networks (few cameras) and the full version which can be used on all camera networks. The vanilla version tackles the problem of overfitting which is more common in smaller networks due to the limited data. The full version tries to mediate the transfer noise caused by non-perfect modelling due to occlusion and detection in the data.

First, CycleGAN is used to train a mapping between camera styles of each camera pair in a system. This results in models that can generate training images with the same ID but in a different camera style. The vanilla version of CamStyle merges the generated images with the real images to train a re-ID CNN model based on ResNet-50 [6]. The full CamStyle version applies label smoothing regularisation (LSR) [15] to the generated images from CycleGAN. By assigning less confidence to the labels of the generated images the transfer noise decreases.

CamStyle produces realistic images with only slight blurring as can be seen in figure 6. The full version of CamStyle showed a significant and consistent increase in person re-ID accuracy. The vanilla version proved to increase accuracy for small camera networks (few cameras).

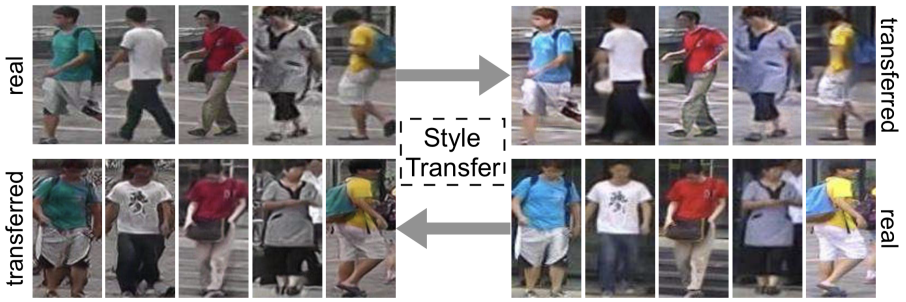


Figure 6: Example images of camera style transformation using CamStyle. Reproduced from the original paper [14].

2.6 Pose Transfer

Pose transfer [16] is the process of generating a new image of a person with a changed pose. In other words, a person's pose can be modified to match the pose of a given skeleton [17]. Generating images with multiple poses allows re-ID datasets to be extended and could result with the ability to learn more robust re-ID models.

A conditional GAN is used with inputs of an image of a person and a skeleton image. With the ground truth being the image of a person with the corresponding pose of the skeleton image. This, by itself, is not very useful for re-ID purposes as the person in the generated image does not necessarily resemble the original person. Thus, the authors proposed a "guider module" to ensure the ID characteristics remain the same. The guider module is pre-trained on the dataset utilising supervised learning to distinguish between classes (or based on triplet loss). Then, when training the GAN, the guider module is fixed and works alongside the discriminator.

As can be seen in figure 7, the guider module helps to produce clearer and overall more realistic images. It is nonetheless far from perfect. One of the most noticeable things wrong with the generated images are the missing feet. Less noticeable but important transformation errors include the head shape, colours of clothing, patterns in clothing, limbs deformed or missing and general blurring. Despite these errors, pose transfer can provide a significant improvement over a baseline approach and is even comparable to state-of-the-art methods.

Figure 7: Example images of pose transfer, reproduced from the original paper [16]. The leftmost column contains the original images and the two column pairs, "No Guider" and "With Guider", contain the output transformation given two slightly different poses and the original image as input.

2.7 Background Bias

For ideal re-ID systems, the background of an image should not provide any features for classifying individuals. However, Tian et al. [18] have shown that deep learning

re-ID models rely too much on background information. A model with a bias to backgrounds is a severe problem and solving this could lead to more robust systems.

Tian et al. propose two different solutions to eliminate background bias. Both of which require separating the foreground from the background. They introduce a deep neural network for this task.

The first solution simply replaces the background with the mean background of the dataset. Training a deep learning re-ID model on these images forces the network to focus on the foreground as there is no information in the background. This, however, does not produce realistic images for the network to train on. The second solution tries to alleviate this problem by replacing the background with a random image. This provides more realistic images to train a deep re-ID model while still removing the biased backgrounds.

Figure 8 shows examples of changed backgrounds. Examples in (b) are clearly unrealistic but re-ID models trained on them perform only slightly worse than models trained on the original images but neither perform well on images with a random background. When models are trained the random background images they perform well when tested individually on the original, the mean background and random background images. There is only a slight drop in performance (2%) but the model is more robust. The examples in (c) are more realistic than those with a mean background but are is still a bit "off".

Figure 8: Examples of changing the background, reproduced from the original paper [18]. (a) are the original images. Images in (b) have a mean background and in (c) have a random background.

2.8 Datasets

Re-ID models require large amount of data during training. Re-ID datasets should also preferably contain sufficiently diverse images to allow models to generalise over the whole re-ID problem. One of the main difficulties for re-ID is the cost of creating these datasets [5]. Table 1 compares commonly used datasets and figure 9 shows some example images from different datasets.

CUHK01 [19] has the fewest bounding boxes and the fewest identities of the highlighted datasets in table 1. Due to the low number of samples, the highest performing re-ID models does not generalise well [5].

CUHK03 [20] is a fairly simple dataset compared to others because it only includes images from two cameras. This dataset has been used and tested extensively and is a popular choice for model performance comparisons.

Market1501 [21] is similar in size as CUHK03 and DukeMTMC-reID. Importantly, the bounding boxes are detected using the Deformable Part Model (DPM) [22] which provides a more realist setting compared to hand-drawn bounding boxes. This is a commonly used dataset for re-ID training.

DukeMTMC-reID [23] is slightly larger than Market1501 and CUHK03. It has hand drawn bounding boxes. DukeMTMC-reID, Market1501 and CUHK03 are frequently used for comparing re-ID model perfomances.

MSMT17 (Multi-Scene Multi-Time) [5] is significantly larger than the other datasets in table 1. It is newer than the other datasets and thus has not been used as much. MSMT17 uses Faster RCNN [24] to create the bounding boxes. It includes cameras located both outside and inside while the other datasets contain cameras either outside or inside.

Dataset	CUHK01	CUHK03	Market1501	DukeMTMC-reID	MSMT17
Bounding boxes	3,884	28,192	32,668	36,411	126,441
Identities	971	1,467	1,501	1,812	4,101
Cameras	10	2	6	8	15

Table 1: Comparison of well-known person re-ID datasets

Figure 9: Examples of images from well known person re-ID datasets

3 Design

3.1 Overview

The aim of this project is to develop a method to improve the robustness of person re-ID networks. At the start of this project, several methods of how to accomplish this goal were proposed and discussed, such as:

- Generating new images using inpainting for masked regions [25].
- Generating new images by predicting a slight variation in pose and background based on next frame predictions [26].
- Generating new images by generating a new background.
- Generating new images by merging the upper and lower part of people's bodies.

These methods have varying difficulty as high emphasis is placed on generating an image of a person who is already in the dataset. Otherwise, labelling this new image might not be possible, but is required for supervised learning.

Developing a method to generate a new background was chosen, as current person re-ID models are biased on backgrounds [18] and no similar method had been developed.

The first model proposed, model 1, is designed to generate a realistic image with the same person but with a new background, from a source image. A simple solution is to perform segmentation of the foreground and background and paste the foreground over a new background image. This, however, creates a few problems. If the segmentation is not perfect, e.g. part of the background is labelled as foreground or vice versa, then the new image will be corrupted, as shown in figure 8. An additional problem is the lighting and perspective differences that will result in the new image being unrealistic. So, the idea behind model 1 is to take two sets of images, with the same identities but different backgrounds, and train a generative adversarial network (GAN) to learn a mapping between these two sets. Thus, model 1 is able to generate an image, with the same person as in the source image but in the same style and with the same background as other images in the target domain.

Model 2, a modified version of model 1, utilises a person mask of the input image as an addition to the source image. Examples of masks can be seen in figure 12. This modification addresses the problem of blurry, and sometimes missing limbs, in generated images from model 1.

Further modifications, resulted in model 3, which uses an output mask (mask of the target image) in addition to the source image and input mask. This allows full control over the pose of the person in the generated image.

Model 4 does not have any additional inputs, similar to model 1, but utilises a novel foreground and background loss. This loss allows the model to generate greater foreground detail and overall more realistic images.

The images generated can be added to the original person re-ID dataset, making an extended dataset. All four models are tested and compared based on the performance of a baseline re-ID network when trained on the respective extended dataset.

3.2 Model 1

Model 1 is designed to be an end-to-end solution that can generate an image of the same person as in a source image, but with a new background and style from a different domain. The two domains can be images from two different cameras from a large person re-ID dataset, such as Market1501.

Model 1 is a GAN, which is composed of a generator network and discriminator network. Figure 10 shows an overview of the GAN.

Given a simple person re-ID dataset with only two cameras, the sub-datasets *cameraA* and *cameraB* can be created, each with images from one camera only. From theses sub-datasets, image pairs can be created, with each image pair consisting of an image from *cameraA* and an image from *cameraB* with the same person, i.e. the same identity. Note, a one-to-one mapping does not exist so a single image in *cameraA* can be paired with multiple image from *cameraB* as long as all images have the same identity. Figure 11 shows how multiple images with the same identity can make many image pairs. This means the underlying function of the mapping between the domains is a multivalued function, i.e. there are multiple solutions given a single input. Model 1 is trained on the paired images, but when the GAN is trained, the only images from *cameraA* are necessary. Model 1 learns a mapping between these two domains (*cameraA* and *cameraB*) and when trained, can generate images in the style of images from *cameraB* with the same identity as a given source image from *cameraA*. Note, the style refers to all differences between the domains, for example, exposure, saturation and background. The model only learns the mapping from *cameraA* to *cameraB*. This means, a separate model (same design) needs to be trained to generate images for the other camera. Similarly, if there are more than two cameras in the original person re-ID dataset, the same process applies, only more models are needed to be trained.

When trained, model 1 can generate labelled images in the style of *cameraB* of persons that do not appear in any images in *cameraB* but only in *cameraA*. Model 1 can also simply generate more image even if the same identity exists in *cameraB*.

Adding the generated images to the original dataset, allows person re-ID networks to be trained on larger and more varied datasets and thus improve their performance.

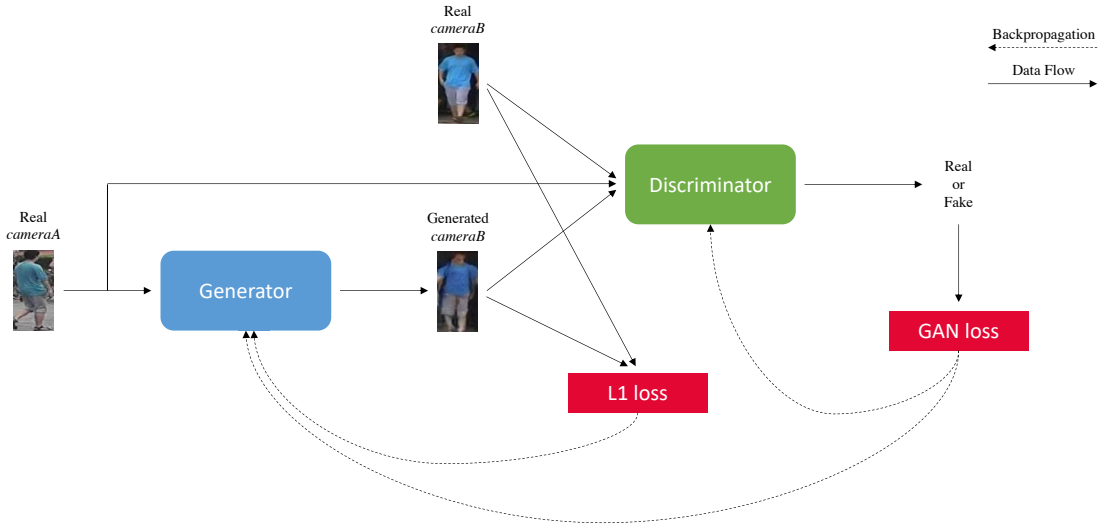


Figure 10: A diagram of model 1. This diagram shows the interaction between the generator and discriminator during training. Notice this is a conditional GAN which means the generator uses an image as input, instead of a noise vector, and the discriminator also uses the input image as an additional input.

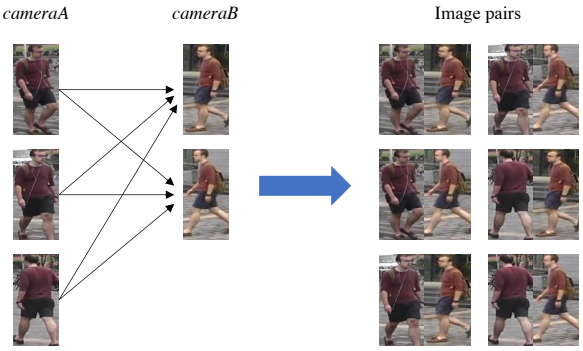


Figure 11: An example of how images with the same identity are paired together.

3.3 Model 2

Model 2 is based on model 1 but has a slight modification to improve the quality of the generated images. The modification involves using a mask of the input image, alongside the input images itself as input. The mask [27] is based on the specific input image and provides information regarding the position of the person within the image. Figure 12 shows examples of masks of Market1501 images picked at random. A large percentage of masks are of low quality but model 2 can still use them as a general guide.

The motivation of adding the input mask as an additional input is twofold. First, this allows the model to more easily locate the person, especially their limbs, to then generate a more accurate image. Secondly, as this project has limited resources, which means no optimisation can be performed on individual models, this allows the model to train more quickly as the location of the person is already given by the mask.

As discussed in detail in chapter 6, the generated images from model 2 occasionally have more refined limbs but the overall quality is slightly worse.

Figure 12: Examples of masks [27] of images from Market1501.

3.4 Model 3

Model 3 is designed to generate images with the same identity as the input image but with a specific pose. Similarly to model 1 and model 2, the generated image should have a generated background and overall the same style as the target dataset. This is accomplished by using an input image, along side an input mask and an output mask. The input mask indicates where the person is in the input image, while the output mask shows where the person is in the target image. A diagram of model 3 is shown in figure 13.

By using the output mask as part of the input to the GAN, the GAN will learn the correlation between the output mask and the position of the person in the target image. This leads the generator to generate the person in that specific pose to minimise the L1 loss. After the GAN has been trained, the pose of the person being generated can be controlled by adding the desired pose as the output mask. This is further discussed in section 6.7.

As discussed in in chapter 6, the generated persons are more realistic than those from model 1 and model 2. The generated background is also realistic, similar to the other models.

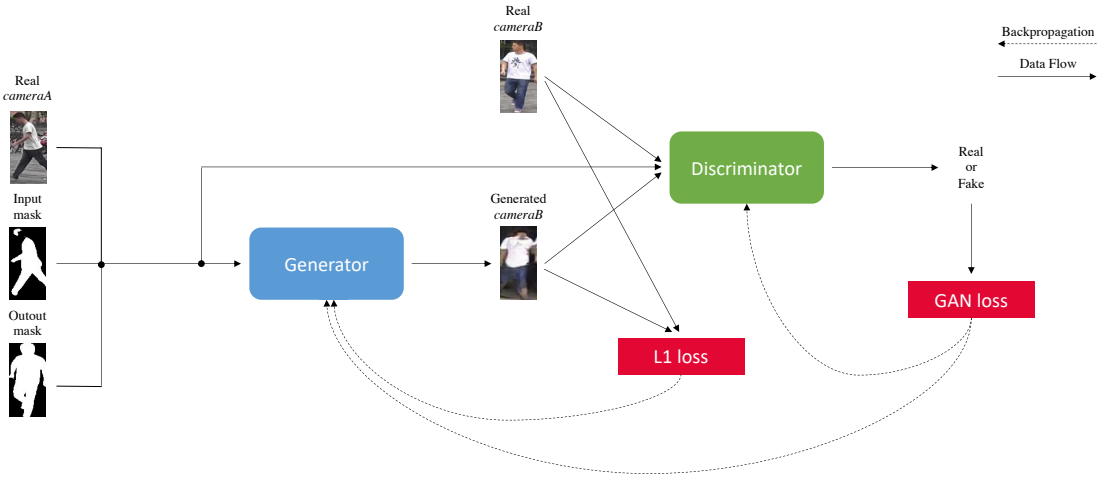


Figure 13: A diagram of model 3 which uses an input of an image concatinated with an input mask and an output mask.

3.5 Model 4

Model 4 only uses the original image as the input, the same as model 1, but employs a new loss function specifically designed for this problem. A diagram of model 4 is shown in figure 14. The proposed loss function is composed of two losses, the foreground loss and the background loss. Note that the proposed loss has two variations, the *static* version and the *variable* version, both described in detail in section 4.6. Model 4 uses the *static* version, which generates the person in the same pose as in the original image. The foreground loss calculates the absolute error between the foreground of the original image and the foreground of the generated image. This causes the person in the original image to be effectively "copied" in the generated image but not the background. The background loss calculates the absolute error between the background of the target image and the background of the generated image. This forces the generator to generate similar background as are found in the target dataset. The GAN loss is still present and affects the whole image. As the GAN loss is feedback from the discriminator, it helps make the overall image more realistic. For example, the exposure and saturation of images in the target dataset might be different to original images. This means that the foreground is not "copied" directly but changed to fit the target dataset. The GAN loss also helps the generator merge the foreground with the background and to not create an obvious boarder, known as a halo.

This design allows model 4 to be trained on unlabelled *cameraB* images, as the identity of the target image is not used.

Note that the discriminator does not receive the full input image but only the background of the image. This is to prevent the discriminator from easily determining the generated image as fake, as the input image and the generated image contain the same identity.

Model 4 generates very realistic persons with a high level of detail. Results are discussed in chapter 6.

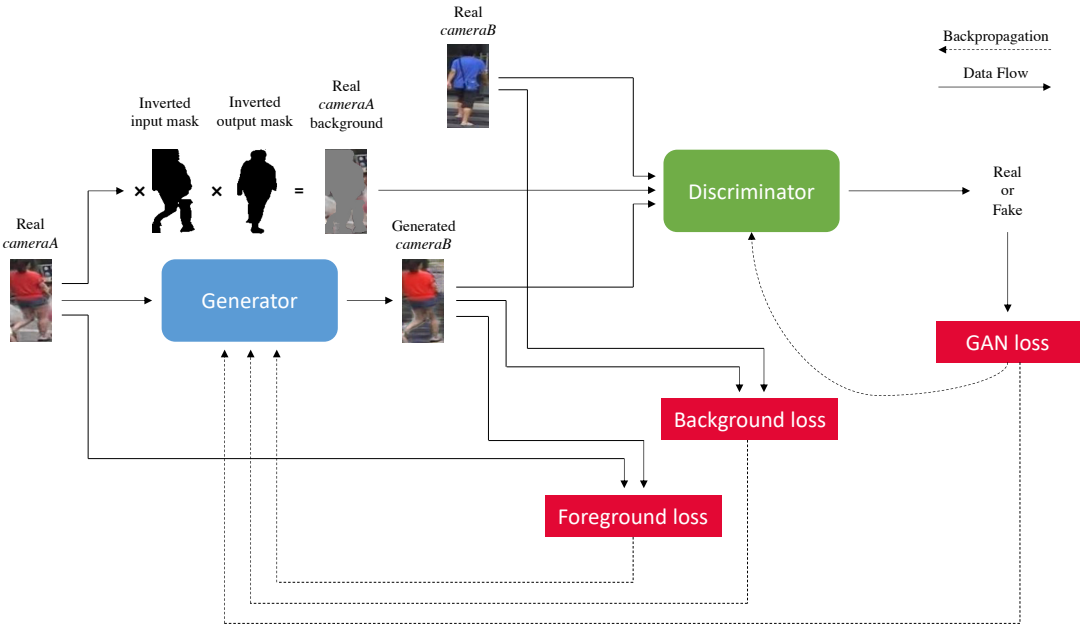


Figure 14: A diagram of model 4 which uses the proposed foreground and background loss, specifically the *static* variation. Note, the foreground loss utilises the input mask and the background loss utilises both the input mask and output mask.

4 Implementation

The initial strategy to improve the robustness of person re-ID networks is to generate new images to be used, in addition to the original images, to train the person re-ID networks. This, theoretically, allows the person re-ID network to generalise more effectively and thus be more resilient to artefacts such as occlusion, noise, view point change, background variations, etc. Figure 15 shows an overview of this implementation process.

All the models are implemented in python using PyTorch [28]. The processing is done through Amazon Web Services using an NVIDIA Tesla K80 GPU.

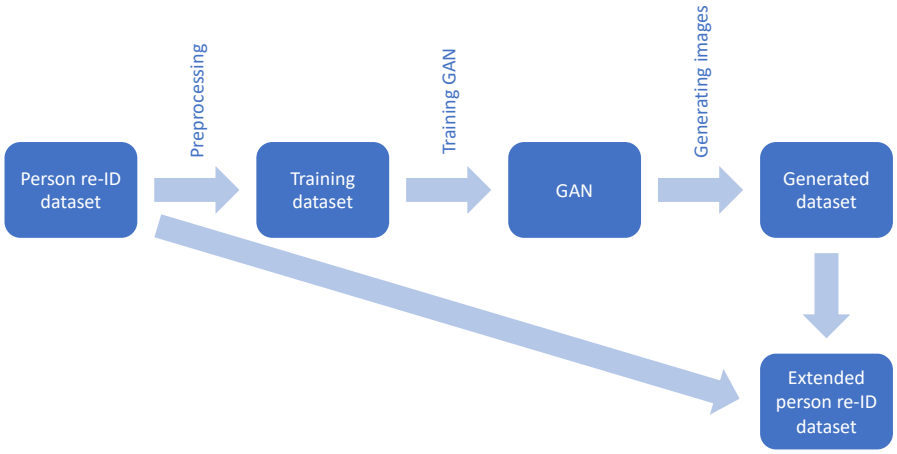


Figure 15: Diagram of the Implementation strategy.

4.1 Data processing

First, the dataset must be split into a train dataset and test dataset. The Market1501 was split 60:40, resulting with the test dataset containing 19,732 images while the train dataset contained 12,936 images. In addition, a query dataset is also provided in Market1501 containing 3,368 images, where there is only one image per identity for each camera.

The training dataset is then split into multiple sub-datasets, each containing a pair of cameras (represented as *cameraA* and *cameraB*). This process is made easier with the DataSplit script, described in Appendix A.1. This produces a total of 15 sub-datasets. The dataset with camera 1 and camera 2 is denoted as c1c2, the dataset with camera 1 and camera 3 is denoted as c1c3 and so on. Each of these sub-datasets contains images with overlapping IDs and non-overlapping IDs. In other words, this is when the same ID (person) exists in images from both cameras in the dataset. For example, camera 1 has two images of "John" and camera 2 has three images of "John", then these images have overlapping IDs. These images are very important as they allow supervision during GAN training, to maintain the same ID.

DataSplit uses the training dataset as an input and outputs one main folder containing four subfolders; "A", "B", "train" and "test". Folder "A" includes all images from *cameraA* and similarly folder "B" includes all images from *cameraB*. The "train" folder is composed of images constructed with image pairs from *cameraA* and *cameraB* with the same ID. This simplifies the input used for the GAN as one

file (PNG) is made of an input or source image (*cameraA*) and a target or ground truth image (*cameraB*). To ensure a large GAN training dataset, all combinations of image pairs between the two cameras, with the same ID, are created. This is visualised in figure 11. To guarantee high similarity between the same person in both images, only the images in the same sequence are paired together. This lowers the likelihood of significant changes to people's appearance, as different sequences signify time discrepancies. Table 2 shows the number of training images available to train the person re-ID network and the total number of test images for each training set.

4.2 Model 1 (Base)

The first model proposed is an end-to-end solution based on the Pix2Pix architecture. The goal is to transform an input image from a source dataset (*cameraA*) to a target dataset (*cameraB*). Pix2Pix was chosen as the base GAN as it performs well with diverse data, trains quickly compared to CycleGAN for example, and is well suited for this objective. The Pix2Pix implementation by the original authors [29] was used with the default hyperparameters: an Adam [30] optimiser with a learning rate of 2e-4 and a beta_1 of 0.5 and batch size of 1. The GAN was trained for 10 epochs. Optimising these parameters would likely result in better performance but that would require resources beyond those allocated for this project.

4.3 Training the GAN

The GAN is designed to generate images that mimic a specific camera using images from another camera. This means a total of six GANs are to be trained, for cameras 1 to 6. To increase the chances of generating realistic images, the largest training datasets should be picked. Table 2 shows the size of the training datasets available. The dataset c5c6 is the largest and thus camera 5 images are used to generates images for camera 6. Similarly, camera 6 images are used to train the GAN that generates images for camera 5.

Table 3 shows which camera pair was picked for each camera to be generated and further information such as the total number of real images to be trained on. These are the images where there are overlapping IDs and are also in the same sequence. The squence refers to the time period the images were taken. The total number of generated images comes from the "test" dataset, shown in table 2, where the images only include IDs found in *cameraA* and not found in *cameraB* with the same sequence, in other word non-overlapping IDs in the same sequence. Lastly, the table shows the number of generated images with IDs that do not appear in the target camera at all, even in a difference sequence.

The motivation behind separating the generated images into two, "all" and "new IDs", is twofold. Using all the generated images, "all", allows for a larger training dataset to be used by the person re-ID network and should lead to higher performance. Acknowledging that the generated images are not perfect, means the generated images could corrupt the learning process especially when a real image exists for a given input. Thus, using only the generated images with non-overlapping IDs, "new IDs", limits the possibility of corrupting real identities and rather only adds value as it is introducing new IDs for a specific camera. Note that these are not completely

new identities being created but rather transferred from *cameraA*, where it exists, to *cameraB*, where this identity was not captured.

	c6	c5	c4	c3	c2	c1
c1	(3151, 1715)	(2448, 1686)	(2408, 1308)	(1561, 1678)	(1008, 1736)	
c2	(6175, 567)	(4921, 428)	(305, 1584)	(6280, 75)		
c3	(13906, 730)	(10368, 407)	(492, **2531**)			
c4	(809, 826)	(875, 785)				
c5	(**14133**, 695)					
c6						

Table 2: The size of different datasets available when pairing two cameras together. Where c1 denotes Camera 1, c2 denotes Camera 2, etc. Each camera pair cell denotes the number of training samples and testing samples available (train, test). c5c6 produces the larges training dataset but the testing dataset is on the smaller side.

Camera pair	**Camera generated**	**Real images**	**Generated images**	**Generated images (new ID)**
c1c6	1	5262	2464	288
c2c3	2	4416	820	769
c3c6	3	5952	261	27
c1c4	4	2937	1308	1250
c5c6	5	5583	606	355
c3c5	5*	5045	407	364
c5c6	6	5583	695	517

Table 3: Decomposition of the datasets available to train the Deep-Person-ReID.

4.4 Model 2 (Input mask modification)

Model 1 produces decent results colour-wise but the images tend to be blurry and often have missing limbs. With the aim of improving the shape of the people in the generated images, an additional mask of the input image was added to the input. Examples of masks are shown in figure 12. The masks used [27] have already been created for all images in Market1501. Each mask was added as an additional channel with the input image. Instead of just the three colour channels (red, green and blue), the input consisted of four channels: red, green, blue and the mask. To implement this change the GAN architecture had to be slightly modified. Note that both the generator and the discriminator have to be modified. Importantly, the output was not changed, meaning that a mask is not generated for the output but only the red, green and blue colour channels.

The implementation involved concatenating the mask (1 x 256 x 256) with the input image (3 x 256 x 256) to make the new input (4 x 256 x 256). Code snippet 1 in Appendix A shows how the mask is found using the name of the input image and how it is concatenated.

4.5 Model 3 (Input and output mask modification)

To achieve a clearer (less blurry) and more varied output, two additional input channels are added to the input image, similar to the input mask modification for model 2. The two input channels consist of two different masks. The first is the mask of the input image, just like model 2, and the second is the mask of the output image. Figure 13 shows all the input channels. By including the output mask with the input image accomplishes two things, mainly the output becomes clearer as the generator knows where the person will be and no longer needs to guess. The output will also be more varied as the output mask determines the pose of the person in the generated image.

Similarly to section 4.4, the two masks are concatenated to the input as shown in code snippet 2 in Appendix A.

4.6 Foreground and Background Loss

The previous methods have all lacked precise control over the similarity between the input image and the generated image. To introduce this control, a modification of the loss function is proposed and tested. Pix2pix consists of two losses, referred to as the cGAN loss ($\mathcal{L}_{cGAN}$) and the L1 loss ($\mathcal{L}_{L1}$). The cGAN loss is feedback from the discriminator, while the L1 loss is the loss between the generated output and the real output. More specifically:

$$\mathcal{L}_{cGAN}(G, D) = \mathbb{E}_{x,y}[log(D(x, y))] + \mathbb{E}_{x,z}[log(1 - D(x, G(x, z)))] \tag{3}$$

$$\mathcal{L}_{L1}(G) = \mathbb{E}_{x,y,z}[||y - G(x, z)||_1] \tag{4}$$

Making the final objective:

$$G^* \arg\min_G \max_D = \mathcal{L}_{cGAN}(G, D) + \lambda \mathcal{L}_{L1}(G) \tag{5}$$

The proposed method introduces a restriction on the cGAN loss and replaces the L1 loss with two similar losses, a foreground loss ($\mathcal{L}_{foreground}(G)$) and a background loss ($\mathcal{L}_{background}(G)$). There are two variations of the proposed method, *static* if the generated person should have same pose as the person in the input image and *variable* if the generated person should have a different pose to that of the person in the output image.

The *static* version does not require any masks to be concatenated with the input but utilises the mask of the input image (m_{in}) and the mask of the target image (m_{out}) when calculating the loss during training. Crucially, the *static* variation does not require the training data pairs to have the same identities since the identities of the target images are not used. This allows the target data to be unlabelled. Thus, removes a hefty requirement previously limiting this method to be used only for a labelled target dataset. This is accomplished by removing (via masking out) the person in the target image (*cameraB*) and replacing them with the person from the input image (*cameraA*), creating a hybrid image. The generated image can be compared to the hybrid image, with the foreground from the input image and the background from the target image. More precisely, the foreground loss, using L1 loss,

only compares the generated foreground with the input foreground by multiplying both by the input mask. Similarly, the background loss, also L1 loss, only compares the generated background with the target background (where they overlap), by multiplying by the inverse of the input mask and the inverse of the target mask. To prevent the discriminator from "cheating", by detecting a fake image as having the same foreground as the input image, the foreground is removed from the input to the discriminator leaving only the background. This process for the *static* version is visualised in figure 16.

The *static* version expressed mathematically:

$$\mathcal{L}_{cGAN}^{static}(G) = \mathbb{E}_{x,y}[log(D(x * m_{in} * m_{out}, y))] + \mathbb{E}_{x,z}[log(1 - D(x * m_{in} * m_{out}, G(x, z)))] \tag{6}$$

$$\mathcal{L}_{foreground}^{static}(G) = \mathbb{E}_{x,z}[||G(x, z) - x||_1 * m_{in}] \tag{7}$$

$$\mathcal{L}_{background}^{static}(G) = \mathbb{E}_{x,y,z}[||G(x, z) - y||_1 * m_{in}^{-1} * m_{out}^{-1}] \tag{8}$$

Where m_{in} is the mask of the person in the input image and m_{out} is the mask of the person in the output image (target image). m_{in}^{-1} and m_{out}^{-1} are the inverted input mask and inverted output mask, respectively, highlighting the background. The final objective:

$$G^* \arg\min_G \max_D = \mathcal{L}_{cGAN}^{static}(G, D) + \lambda_1 \mathcal{L}_{foreground}^{static}(G) + \lambda_2 \mathcal{L}_{background}^{static}(G) \tag{9}$$

The *variable* background version does not require any restriction to the cGAN loss but does require both datasets to be labelled. The foreground loss compares the generated foreground to the target foreground, as opposed to the input foreground in the *static* version. Note the images are multiplied by the output mask instead of the input mask. The background loss compares the generated background to the target background by multiplying both by the inverted output mask.

The *variable* version expressed mathematically:

$$\mathcal{L}_{cGAN}^{variable}(G) = \mathbb{E}_{x,y}[log(D(x, y))] + \mathbb{E}_{x,z}[log(1 - D(x, G(x, z)))] \tag{10}$$

$$\mathcal{L}_{foreground}^{variable}(G) = \mathbb{E}_{x,z}[||G(x, z) - y||_1 * m_{out}] \tag{11}$$

$$\mathcal{L}_{background}^{variable}(G) = \mathbb{E}_{x,y,z}[||G(x, z) - y||_1 * m_{out}^{-1}] \tag{12}$$

Making the final objective:

$$G^* \arg\min_G \max_D = \mathcal{L}_{cGAN}^{variable}(G, D) + \lambda_1 \mathcal{L}_{foreground}^{variable}(G) + \lambda_2 \mathcal{L}_{background}^{variable}(G) \tag{13}$$

$\mathcal{L}_{foreground}$ and $\mathcal{L}_{background}$ have separate weights associated with them, which in turn allows for significant control over the importance of the similarity of the

generated identity and the original identity versus the realism of the background transformation. For example, a large weight for the foreground loss will result in the generator to focus on producing a visually similar person as the person in the input image while the background has little importance. A large weight for the background loss will have the opposite effect, with little importance on the similarity of the identity and the main focus on the background transformation.

An additional benefit of having a separate foreground and background loss is the ability to compare the generated image to a target foreground and background not in the same image. In other words, the generated foreground can be compared to a foreground from a target image whereas the generated background can be compared to a background from another target image.

Due to resource limitations only the *static* variation has been implemented. Part of the implementation of the *static* method can be seen in code snippet 3.

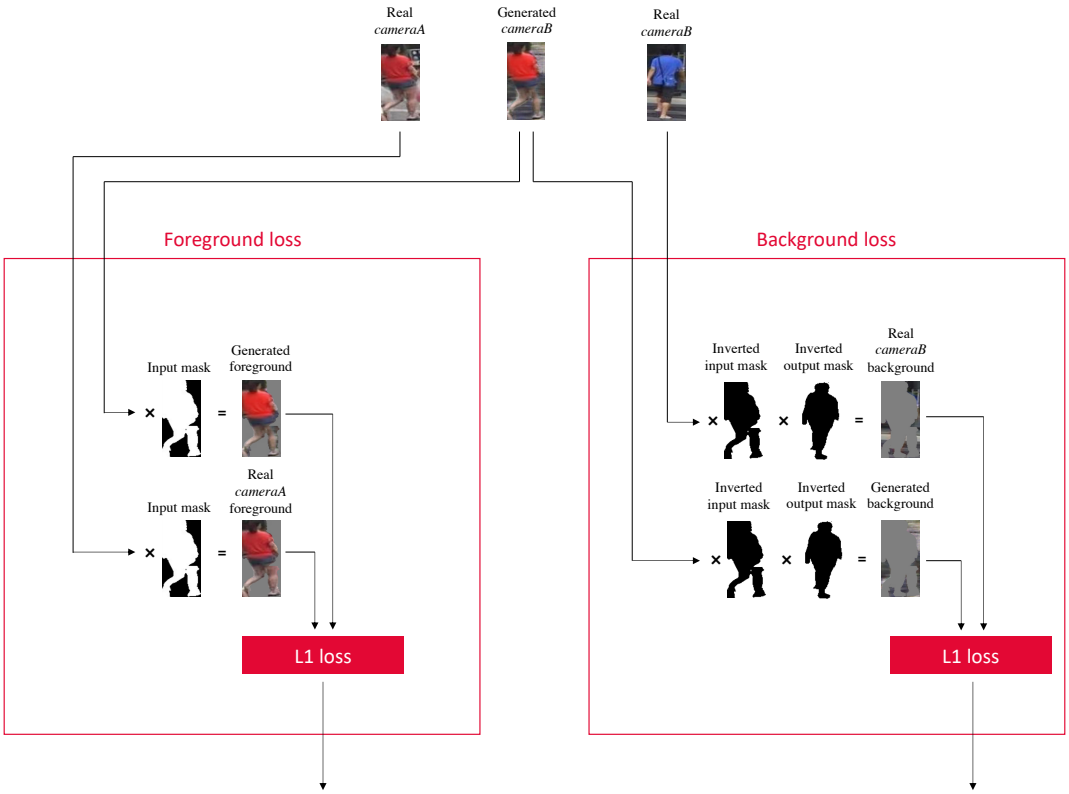


Figure 16: Diagram of the foreground loss and the background loss, specifically the *static* variation.

4.7 Model 4 (Foreground and background loss)

Model 4 implements the static version of the proposed loss function. This means the foreground loss compares the foreground of the generated image with the foreground of the input image. The generated person will not however be identical to the input person as the style is changed to the target dataset due to the GAN loss. The architecture of model 4 can be seen in figure 14.

Unlike the other models, model 4 does not require the target dataset to be labelled which means, the image pairs the GAN trains on are not of the same

identity. In the extreme, one source image can be individually paired with each image in the target dataset. The number of image pairs to be made for each source image can be treated as a hyperparameter that would need to be optimised.

For model 4, 10 image pairs are created for each source image. Model 4 is trained with the same hyperparameters as the previous models except the batch size is 32. The batch size is increased as there are more training samples for model 4. The weights of the foreground loss and background loss are set to 70 and 30, respectively.

5 Testing

The aim of this project is to generate new hard cases to improve the robustness of current re-ID systems. Specifically, generate new backgrounds for images to extend current person re-ID datasets to improve the robustness of current re-ID systems. The generated background should be realistic. There is no consistent way to quantify how realistic an image (or background) is, as it depends on people's opinion. Evaluating the accuracy and robustness of person re-ID models trained on these images can, however, be quantified.

The testing procedure for person re-ID models consists of multiple comparisons between a query image and gallery images. Both query and gallery images have been cropped to only include the full body of one person each. For each query image, a ranked list of the gallery images is created based on the likelihood of them containing the same identity.

A rank-1 accuracy indicates how often the first image in the model's ranked lists contained the same identity as the query image. Similarly, rank-5 and rank-10 accuracies indicate how often an image of the correct identity was placed in the first 5 or 10 places in the ranked lists, respectively.

The main evaluation of the proposed background GAN consists of comparing the results of a person re-ID network trained on several untouched datasets to the results of the same person re-ID network trained on the same datasets with additional generated images from the GAN. This can be split into three experiments, each with its own baseline. Note, the baselines only differ by the training data.

The person re-ID network architecture and hyperparameters are kept the same for all the experiments. The person re-ID network used is implemented using the deep-person-reID [31] library with the following parameters:

- Loss: softmax
- Architecture: resnet50 [6]
- Optimiser: Adam [30]
- Learning rate: 0.0003
- label-smooth
- Step-size: 20, 40
- Batch-size: 32
- Epochs: 60
- Random seed: 0

Baseline - one camera, the simplest baseline, is trained only on images from a single camera. This camera is sometimes referred to as *cameraA* or the source camera. The baseline itself is referred to as $\text{baseline}_{\text{c1}}$ when trained on images from camera 1, $\text{baseline}_{\text{c2}}$ when trained on images from camera 2, and so on. Table 4 shows the performance of the baselines of each camera.

Baseline - two cameras, is trained on images from two cameras, the source camera (*cameraA*) and the target camera (*cameraB*). The baseline is referred to as $\text{baseline}_{\text{c1c2}}$ when trained on images from camera 1 and camera 2, and so on. Multiple baselines exist but table 5 shows the baselines for the camera pairs that produce the largest training dataset for each camera. The size of the training dataset of each

camera pair can be seen in table 2. These baselines provide the best experiment as each camera pair is a fully functioning person re-ID dataset, but each dataset contains only two cameras out of six. This is important because each person re-ID model is tested on the test dataset which contains all six cameras. Thus, the performance greatly depends on the robustness of each person re-ID model. In other words, comparing the $\text{baseline}_{\text{c1c2}}$ with the same person re-ID model, but trained on c1c2 plus generated images, will show if the generates images help improve the overall accuracy and robustness.

Baseline - all cameras, the most advanced baseline, is trained on the whole Market1501 dataset (all six cameras). Referred to as $\text{baseline}_{\text{all-cameras}}$. The performance of this baseline is shown in table 6.

The testing procedure for the person re-ID network is kept the same for each experiment. This is the default test which, calculates the mean average precision (mAP) and rank accuracies for the Market1501 test dataset. The mAP equation:

$$mAP = \frac{\sum_{q=1}^{Q} AveP(q)}{Q} \tag{14}$$

Where Q is the number of queries.

Baseline:	c1	c2	c3	c4	c5	c6
mAP	14.2%	12.6%	15.4%	13.8%	16.9%	13.7%
Rank 1	31.2%	30.1%	33.5%	33.1%	37.1%	31.9%
Rank 5	47.0%	45.6%	48.7%	49.5%	53.5%	48.5%
Rank 10	54.8%	53.4%	54.9%	57.7%	60.8%	56.5%
Rank 20	61.6%	60.7%	61.2%	64.7%	67.3%	64.5%

Table 4: The baseline person re-ID network trained on only one camera. Each column represents the baseline for an individual camera, where c1 refers to camera 1 and so on. The baselines are tested on the same testing dataset which includes all cameras.

Baseline:	c1c6	c2c3	c3c6	c1c4	c3c5	c5c6
mAP	34.8%	29.7%	36.4%	19.2%	23.4%	35.4%
Rank 1	57.6%	55.1%	57.5%	40.9%	46.0%	56.1%
Rank 5	74.9%	68.3%	72.4%	57.7%	59.7%	73.3%
Rank 10	80.6%	73.8%	79.1%	64.1%	66.1%	80.1%
Rank 20	86.2%	78.8%	84.7%	69.2%	71.7%	85.9%

Table 5: The baseline person re-ID network trained on selective camera pairs. Each column represents the baseline for a camera pair, where c1c6 refers to the camera 1 and camera 6 pair, and so on. The baselines are tested on the same testing dataset which includes all cameras.

Baseline:	c_all
mAP	68.6%
Rank 1	85.1%
Rank 5	94.2%
Rank 10	96.2%
Rank 20	97.8%

Table 6: The baseline person re-ID network trained on the whole training dataset, which includes all cameras. The baseline is tested on the testing dataset which includes all cameras.

6 Results

All experiments were conducted on the Market1501 dataset or sub-datasets unless otherwise specified. Note, the hyperparameters used and the testing procedures are discussed in chapter 5.

6.1 Image comparison and quality assessment

Each model is trained using a camera pair, *cameraA* and *cameraB*, to generate images in the style of *cameraB*. In Market1501, there are a total of 6 cameras. To generate images in the style of each camera the largest camera pair dataset, shown in table 2, is chosen. The largest camera pair dataset is between camera 5 and camera 6 (c5c6) and should produce the best results and thus the best comparison between the proposed models. This means, images from camera 5 (*cameraA*) and camera 6 (*cameraB*) are used to train each model to generate images in the style of camera 6 (*cameraB*).

Figure 17 shows examples of generated images from each model. The generated images on the left are examples when the models struggle, while the generated images on the right are of higher quality.

Model 1 generates images with distinguishable foregrounds and backgrounds from only an input image. The person generated is quite blurry and occasionally has missing limbs. The pose and shape vary little between the generated images, most often a direct front or back view with arms to each side. Distinct features such as patterns on clothes and body shape are often lost. The colours generated are slightly different to the original but consistent with the style of the target images. For example, the images from camera 6 are more saturated than those from camera 5, and the model successfully captures this. The background generated has minimal noise and varies greatly. Both light and dark backgrounds are generated with distinctive features such as stairs or walls. These features are common in images from camera 6 but not in camera 5, which shows this model not only ignores the background from the original image but also captures and produces these features that match the background of the target camera. The stairs or lines in the background often match on both sides of the person which is very realistic.

Overall model 1 most often generates the same shape, an average person with a direct front or back view, but in the same colour scheme as the original identity. This happens because the function, which the model is learning, is a multivalued function, in contrast to a single-valued function. This means an input image from *cameraA* does not correspond to a single image in *cameraB*, but instead corresponds to an infinite amount of hypothetical *cameraB* images. However, during training the L1 loss does not take this into account but rather compares the input image (*cameraA*) to a specific target image (*cameraB*). This causes the GAN to generate the person in the most common shape and pose to minimise the loss.

Model 2 uses the mask of the input image, in addition to the input image itself, when generating a new image. This is to help the model detect the person in the input image more easily thus improve the generation of limbs. Comparing the images from model 2 and model 1, in figure 17, shows that model 2 does produce more defined limbs. However, model 2 has similar shortcomings when it comes to small details and patterns on clothes. Some fine details are produced, for example,

Figure 17: The images on the left are examples of when the models struggle when generating images while the images on the right are of higher quality. "Original" are real images from camera 5. The "model" columns show generated images in the style of camera 6, using the original as input. The images at the bottom are examples of real images from camera 6.

backpack straps that are not visible in images from model 1. The colours of the pants are slightly darker and more realistic than those from model 1 but sometimes additional noise (dark patches) appear on the shirt. Model 2 produces very similar backgrounds to model 1 in terms of the visible details and overall realism.

Model 3 improves upon the previous models especially by producing more varied output poses and defining body shapes. The generated pose depends on an additional input, the output mask, which is chosen at random. Thus, from a single input image, multiple different output images can be produced, described in more detail in section 6.7. Small details and patterns in the foreground are often lost, the same as the previous model. The background maintains the same realism and quality compared to the previous models. As the people in the generated images have specific and real poses, the overall image realism is improved.

Model 4, with a modified loss function, generates images with much greater detail than the other models. The generated foreground is very similar to the original foreground and importantly includes small details and patterns. The differences are subtle, slight changes in exposure, colour and colour saturation is expected as these settings differ between *cameraA* and *cameraB*. The generated background often includes details such as stairs or walls, but the background tends to split between the left and right side of the foreground. As in the background on the left side does not always correlate to the background on the right. For example, the backgrounds in *cameraB* can be very dark or bright. The generated images occasionally produce a bright background on one side but a dark background on the other side. This does happen in some real *cameraB* images but less frequently than in those from model 4. This abnormality can be described as a continuity split in the background. The cause of which can be attributed to a "gap" in the loss function, between the foreground loss and background loss. The part of the generated image in the "gap" is only covered by the GAN loss. The results suggest that the weights for the different losses need to be changed, most likely increasing the weighting of the GAN loss. Overall, model 4 produces very realistic people that stay true to the person in the original image. However, the generated backgrounds are slightly less realistic than those from the previous models due to the aforementioned continuity split.

6.2 Nearest neighbour of generated images

Given a real image from camera 5, each model can generate an image in the style of camera 6 but with the same identity as the original image. To show that these generated images truly capture the original identity, a k-nearest neighbour search is done on each of the images. Figure 18 shows the result of this experiment. First, the original image is tested to show that it contains the required features to correctly match with images with the same identity. Figure 18 clearly shows that the generated images from model 1, model 3 and model 4 represent the same identity as the original image. However, the image generated by model 2 does not match with the correct identity but rather a similarly dress individual. Under closer inspection, the generated person in the model 3 image has what could be misinterpreted as backpack straps over the shoulders. This detail could have caused the mismatch even though the skin tone of the generated person is more similar to the original identity than the similarly dressed individual.

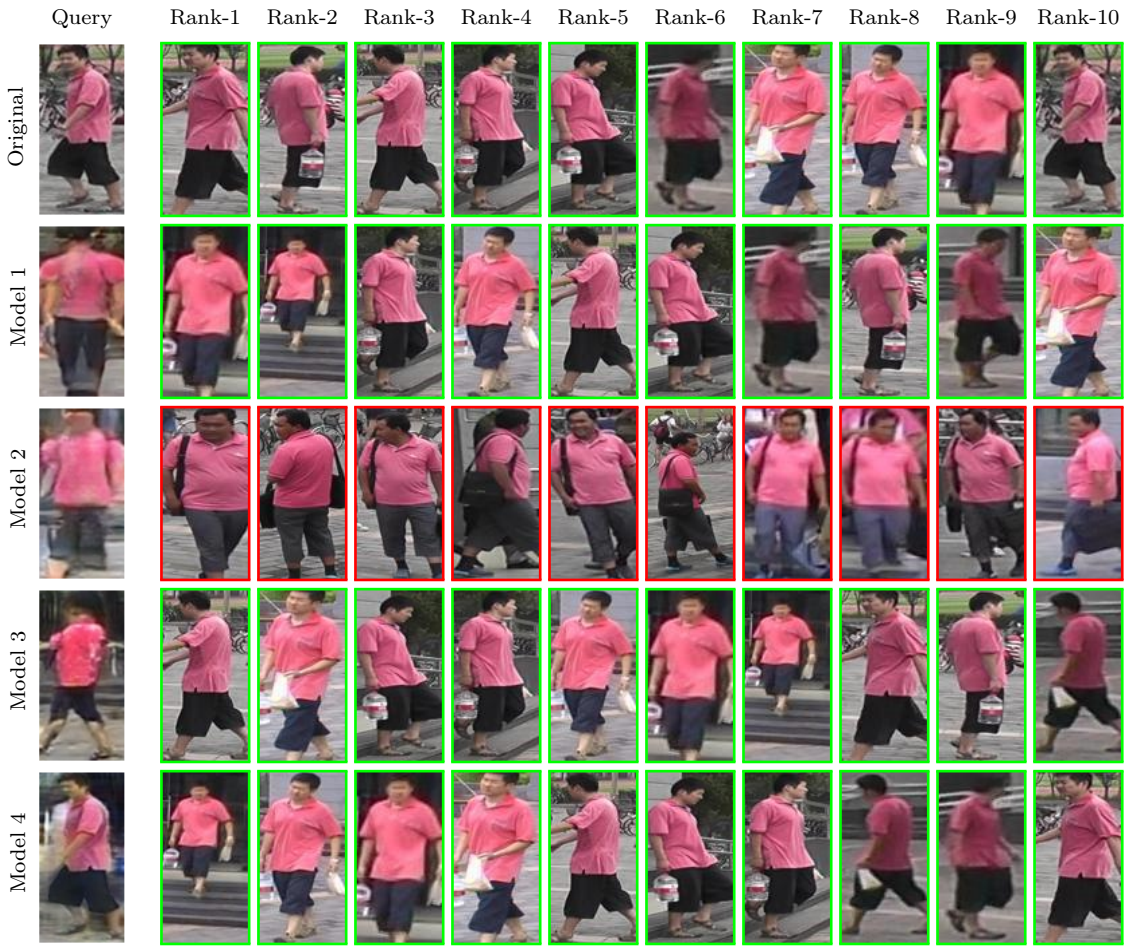


Figure 18: Nearest neighbours for the output for each model. Green outline signifies the correct identity while a red outline signifies an incorrect identity.

6.3 Generating images for one camera

Each of the proposed models is trained on the c5c6 training dataset to generate images in the style of camera 6, from images in camera 5. Using a set of images from camera 5, each model generates images in the style of camera 6, referred to as G6. The baseline person re-ID model is trained on a dataset, composed of images from camera 5 and the generated images from a single model, known as c5+G6. Table 7 shows the performance of $\text{baseline}_{\text{c5}}$ and the performance of the baseline person re-ID trained on c5+G6 from each model. Each model's generated images help improve over $\text{baseline}_{\text{c5}}$, while model 4 allows for the greatest improvement. The improvement by using the generated images shows that the models can generate images with the same identity as the original images but in the style of the target camera.

Model	Baseline	Model 1	Model 2	Model 3	Model 4
GAN training data	-	c5c6	c5c6	c5c6	c5c6 (unlabelled)
Person re-ID training data	c5	c5+G6	c5+G6	c5+G6	c5+G6
mAP	16.9%	19.7%	18.6%	18.1%	**20.7%**
Rank-1	37.1%	39.8%	39.2%	37.8%	**43.2%**
Rank-5	53.5%	57.9%	56.0%	55.2%	**62.6%**
Rank-10	60.8%	65.5%	63.6%	63.9%	**69.9%**
Rank-20	67.3%	72.6%	71.7%	71.6%	**77.2%**

Table 7: The performance of the baseline person re-ID model based on the training data. Best results are in **bold**. Model 4 is trained on labelled images from camera 5 but unlabelled images from camera 6, while the other models are trained on labelled images from both cameras.

6.4 Extending a person re-ID dataset

The same as the previous experiment, each proposed model is trained on the c5c6 train dataset. Models 1, 2 and 3 require labelled images from both camera 5 and camera 6. The c5c6 test dataset only contains images from camera 5 where the same identity, in the same sequence, does not appear in any camera 6 image. The decomposition of c5c6 can be seen in table 3. Models 1, 2 and 3 use the c5c6 testing dataset to generate the images for camera 6 to create G6. Model 4 does not require the target camera to be labelled thus is trained using the whole c5c6 dataset. Model 4 generates a new image in the style of camera 6 for each image from camera 5. This means models 1, 2 and 3 generate 695 images (G6) each, while model 4 generates 3245 images (G6). The number of generated images used to extend the original dataset can have an effect on the quality of the dataset and thus the performance. Adding a just few images might not have an effect on the performance but too many images can negatively effect the performance by corrupting the dataset. The number of generated images added to the original dataset can be considered a hyperparameter that can be optimesed, but no optimisation has been done. Table 8 shows the performance of person re-ID models when trained on datasets extended using generated images. Model 3 performs the best, with the same mAP as $\text{baseline}_{\text{c5c6}}$

but a higher rank-1, rank-5 and rank-10 accuracy. Model 4 has the best rank-5 accuracy, with rank-1 and rank-10 also better than $\text{baseline}_{\text{c5c6}}$. Model 1 and model 2 do not improve on the baseline. Model 2 has the worst performance.

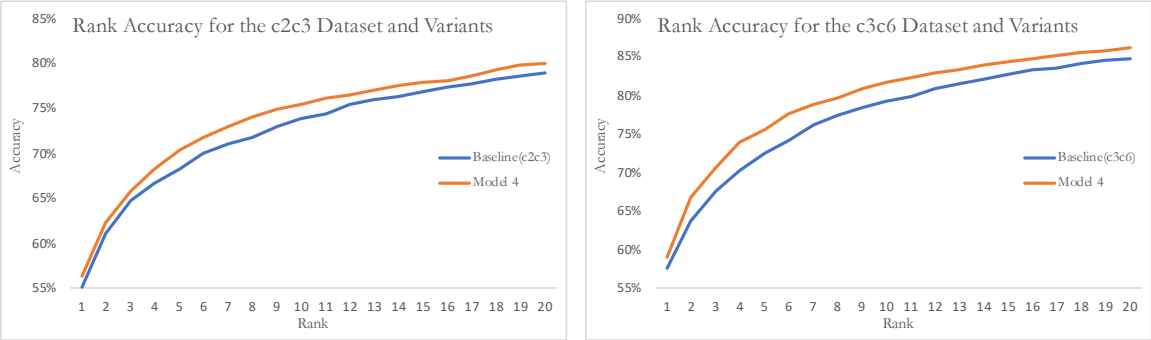
Similar experiments are conducted for the datasets c2c3 and c3c6, both extended by model 4. Table 9 shows how the extended dataset can significantly improve the performance of person re-ID networks. This is also visualised in figure 19. Note, further performance improvements are most likely possible as none of the proposed models have been optimised.

Model	Baseline	Model 1	Model 2	Model 3	Model 4
GAN training data	-	c5c6	c5c6	c5c6	c5c6 (unlabelled)
Person re-ID training data	c5c6	c5c6+G6	c5c6+G6	c5c6+G6	c5c6+G6
mAP	**35.4%**	34.3%	33.9%	**35.4%**	34.9%
Rank-1	56.1%	55.3%	54.8%	**56.8%**	56.4%
Rank-5	73.3%	73.5%	71.4%	73.6%	**74.5%**
Rank-10	80.1%	79.6%	79.5%	**80.8%**	80.3%
Rank-20	85.9%	84.9%	85.1%	**86.5%**	85.3%

Table 8: The performance of the baseline person re-ID model using different training data. Best results are in **bold**. Model 3 and model 4 have a higher rank-1, rank-5 and rank-10 accuracy than the baseline. Model 4 is trained on labelled images from camera 5 but unlabelled images from camera 6, while the other models are trained on labelled images from both cameras.

Model	$\text{Baseline}_{\text{c2c3}}$	Model 4	$\text{Baseline}_{\text{c3c6}}$	Model 4
GAN training data	-	c2c3 (unlabelled)	-	c3c6 (unlabelled)
Person re-ID training data	c2c3	c2c3+G2	c3c6	c3c6+G3
mAP	29.7%	**31.7%**	36.4%	**38.1%**
Rank-1	55.1%	**56.3%**	57.5%	**58.9%**
Rank-5	68.3%	**70.3%**	72.4%	**75.6%**
Rank-10	73.8%	**75.4%**	79.1%	**81.7%**
Rank-20	78.8%	**80.0%**	84.7%	**86.1%**

Table 9: The performance of the baseline person re-ID model using different training data. Best results are in **bold**. Model 4 does not require the target dataset to be labelled.



(a) (b)

Figure 19: Performance of a person re-ID network trained on different datasets. (a) trained on c2c3 and c2c3+G2. (b) trained on c3c6 and c3c6+G3.

6.5 Extending a large person re-ID dataset

The previous experiment consisted of extending a person re-ID dataset by adding generated images in the style of just one camera. For larger datasets, generated images for each camera, individually, could extend the original dataset and thus improve the performance of person re-ID networks. Model 1 is trained on the largest camera pair training dataset to generate images in the style of each camera individually. The training sets used are listed in table 3. With each set of generated images, G1 to G6, only the ones which improve on their respective camera pair baseline are used to extend the dataset. This results in only G3 and G5 from model 1 to be used to extend the dataset. The same is done for model 3 which results in only G6 to be used. Again, for Model 4, which results in using G2 and G3. Note that for model 4, G4 and G1 were not generated due to resource limitations. Model 2 was omitted entirely due to resource limitations. Table 10 shows the performance of the baseline person re-ID model when trained on the original dataset and the different extended versions, visualised in figure 20. None of the models improve on the $\text{baseline}_{\text{all-cameras}}$, with only model 3 achieving the same rank-10 accuracy as the baseline. This is expected as $\text{baseline}_{\text{all-cameras}}$ already performs very well and the proposed models have not been optimised. Given the results from the previous experiment, the performance of the person re-ID network trained on the extended datasets would most likely improve if the proposed models are optimised.

Model	Baseline	Model 1	Model 3	Model 4
Person re-ID training data	c_all	c_all+G3+G5	c_all+G6	c_all+G2+G3
mAP	**68.6%**	67.6%	67.8%	65.5%
Rank-1	**85.1%**	85.0%	84.9%	83.3%
Rank-5	**94.2%**	93.6%	93.9%	93.0%
Rank-10	**96.2%**	95.5%	**96.2%**	95.1%
Rank-20	**97.8%**	97.1%	97.6%	97.0%

Table 10: The performance of the baseline person re-ID model using different training data. Best results are in **bold**. Only the generated images which improved on their respective camera pair baseline are used to extend the dataset.

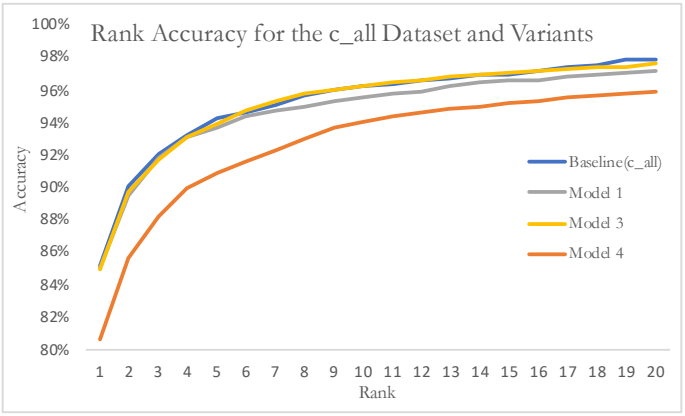


Figure 20: Performance of a person re-ID network trained on all cameras and extended variations.

6.6 Domain transfer from labelled to unlabelled datasets

Model 4 utilises the proposed foreground and background loss, specifically the *static* version, which does not require the target dataset to be labelled. This allows model 4 to extend a dataset using an unlabelled dataset. For example, given a labelled dataset only with images from camera 6, a person re-ID network can be trained using this data resulting with baseline_{c6}. Now, camera 3 is installed and images recorded but the images are not labelled as that requires significant manual labour. If the images from camera 3 would be labelled, baseline_{c3c6} could be created. Training model 4 on the labelled images from camera 6 and the unlabelled images from camera 3, allows model 4 to generate labelled images, using the identities from camera 6, in the style of camera 3. These images are referred to as G3. Table 11 shows the performance of baseline_{c3c6}, baseline_{c6} and the person re-ID network trained on c6+G3. Table 12 shows a similar situation but with different cameras. Both examples show a significant performance improvement when the person re-ID network trained on model 4's extended dataset over the single camera baseline. Note, the performance re-ID performance when using the extended dataset could most likely be improved as model 4 has not been optimised.

Model	baseline_{c3c6}	baseline_{c6}	Model 4
GAN training data	-	-	c3c6 (unlabelled)
Person re-ID training data	c3c6	c6	c6+G3
mAP	36.4%	13.7%	20.1%
Rank-1	57.5%	31.9%	40.7%
Rank-5	72.4%	48.5%	59.0%
Rank-10	79.1%	56.5%	67.0%
Rank-20	84.7%	64.5%	74.6%

Table 11: Baseline_{c3c6} shows the performance of the person re-ID network trained on real images from camera 3 and real images from camera 6. Baseline_{c6} shows the re-ID performance when only real images from camera 6 are used. Model 4 is trained on labelled images from camera 6 but unlabelled images from camera 3.

Model	$\text{baseline}_{\text{c2c3}}$	$\text{baseline}_{\text{c3}}$	Model 4
GAN training data	-	-	c2c3 (unlabelled)
Person re-ID training data	c2c3	c3	c3+G2
mAP	29.7%	15.4%	24.2%
Rank-1	55.1%	33.5%	47.3%
Rank-5	68.3%	48.7%	63.3%
Rank-10	73.8%	54.9%	69.4%
Rank-20	78.8%	61.2%	75.6%

Table 12: $\text{Baseline}_{\text{c2c3}}$ shows the performance of the person re-ID network trained on real images from camera 2 and real images from camera 3. $\text{Baseline}_{\text{c3}}$ shows the re-ID performance when only real images from camera 3 are used. Model 4 is trained on labelled images from camera 3 but unlabelled images from camera 2.

6.7 Pose generation

The architecture of model 3 provides the ability to control the pose of the person in the generated image. Figure 21 shows the effect of keeping the input image constant but changing the target mask. The foreground is successfully generated in the same shape as the target mask, but small details are lost as the generated person is quite blurry. In the second example, the red shoes are not generated correctly, with only the right shoe in two images being slightly red. This might be caused by a lack of training samples with coloured shoes. The generated backgrounds vary greatly, and often include details such as stairs or walls. The ability to control the pose has many benefits, most importantly, the distribution of pose variation can be adjusted, and multiple different images can be generated from a single input image.

Figure 21: Examples of the varying output from model 3 depending on the target mask. "Gen" is the generated image, in the style of images from camera 6, given the "original", from camera 5, and the corresponding "pose" as input.

7 Evaluation

At the start of this project several objectives were set out to accomplish the main aim of this project, which is to generate new, hard cases to improve the robustness of current re-ID systems. Four models are proposed in this project, tackling different aspects of the objectives.

Model 1 can generate labelled images of existing persons with new backgrounds. However, as discussed in section 6.1, while the generated backgrounds are decent, the generated persons are blurry and lack details and patterns on clothes that the original identity has. Due to the lack of detail on the generated person, an argument can be made that model 1 does not achieve the desired objective of generating labelled images of existing persons. But despite the perceived lack of detail, results discussed in section 6.2 indicate that the generated images from model 1 include sufficient details to be classified as the same identity as the original image. Thus model 1 can be classified as achieving the objective to generate labelled images of existing persons with new backgrounds. As the generated images are correctly labelled, as it matches the original identity, and also capture the style of the target dataset as shown in 6.3, the images can be used to extend the original dataset, thus achieving its second objective. Training the person re-ID network on the extended dataset gives mixed results. Importantly, for some datasets the extended version allows the re-ID network to perform better than the baseline, as shown in the appendix in table 13. This is despite the model not being optimised. With optimisation, the performance would most likely improve. Even without optimisation, object of to improve on the person re-ID network baseline by using the extended dataset is achieved.

Model 2 was designed to be an improvement over model 1. As discussed in section 6.1, model 2 generates images very similar to that of model 1. The generated images have more defined limbs than those from model 1 but with small additional artefacts. However, the results discussed in section 6.2 do not indicate that the generated images from model 2 include sufficient details to be classified as the same identity as the original image. This means that model 2 does not achieve its objective.

Model 3 was designed to generate labelled images of existing persons with new poses. The quality of the generated images is better than images from the previous models, as discussed in section 6.1 and the images include sufficient details to be classified as the same identity as the original image, as shown in section 6.2. The pose of the person can be controlled as shown in section 6.7. Importantly, the generated images capture the style of the target dataset as shown in section 6.3. This was not part of the objective regarding the pose changes. Due to the clever design of model 3 it achieves both the objective for generating labelled images of existing persons with new poses, and the objective to generate labelled images of existing persons with new backgrounds. The generated images can be used to extend the original dataset. A person re-ID network trained on the extended dataset does not improve on the baseline, as their performance are very similar. As with the other models, model 3 has not been optimised and the results would likely improve with optimisation.

The quality of the generated images is directly comparable to the state-of-the-art methods such as Pose Transfer [16]. Examples of Pose Transfer and model 3 images are shown in figure 22. Pose Transfer generates the person in the same style, but the generated backgrounds contain obvious artefacts. Model 3 however, does change

the style of the image and generates realistic backgrounds. The style change can even be compared to state-of-the-art methods such as CamStyle [14].

Model 4 utilises the novel foreground and background loss, specifically the *static* version, to generate images with the same identity as the original image but with a new background. The generated persons are of very high quality which successfully captures sufficient details to be classified as the same identity as the original image, as shown in 6.2. The generated images also capture the style of the target camera, as shown in section 6.3. This means the original dataset can be extended using the generated images. A person re-ID network trained on the extended can improve significantly on the baseline as shown in section 6.4. Thus, three objectives are achieved, labelled images of existing persons with new backgrounds are generated, a person re-ID dataset is extended, and the person re-ID network trained on the extended dataset improves on the baseline.

Model 4 can generate images that capture the overall style of the target dataset and also generates the background in the style of the target dataset. Importantly, model 4 does not need the target dataset to be labelled, thus can be compared to SPGAN [12]. SPGAN does not change the background of the images but the overall style. As model 4 changes the style and the background, model 4 could theoretically improve on SPGAN.

Overall, all the objectives have been met and the proposed solutions are comparable to state-of-the-art methods, and even have additional benefits.

Figure 22: Examples of the generated output from model 3 and Pose Transfer [16]. The "Original" is used to generated the two "Gen" images. Model 3 also changes the style of the image so an example of the same identity in the target dataset is shown for comparison as "Target".

8 Conclusion and Further Work

Person re-ID is a difficult problem which involves identifying the same person across multiple disjoint cameras. The difficulty of achieving high person re-ID accuracy is caused by multiple factors such as non-overlapping field of view, lighting differences, occlusion, variation in poses and different camera viewpoints. The images are often of low quality and low resolution so biometric markers such as facial features cannot be used.

There are multiple use cases for person re-ID models, for example, surveillance and tracking by police, or general data gathering by businesses such as supermarkets or malls. A very high accuracy and robust person re-ID model could be used in places such as the London Underground to remove the need to "tap out" to indicate where person exited the system. The current state-of-the-art person re-ID methods can achieve high accuracies on some specific datasets but due to the lack of robustness, experience a significant drop in performance when tested on a different dataset.

This project proposes and analyses several different methods to improve the robustness of person re-ID networks. Each of the proposed methods are designed to generate new, hard cases which can be used to extend a person re-ID dataset. Training existing person re-ID networks using the dataset extended with the generated images, allows for higher accuracy and increased robustness. This is a non-trivial problem as the generated images are required to be labelled for supervised learning to be possible.

Model 1 is designed to generate labelled images of existing persons with new backgrounds in the style of a target dataset. It requires both the source dataset and the target dataset to be labelled and contain the same identities. Model 1 proved to be effective at improving the robustness of person re-ID networks.

Model 2, an attempt to improve on model 1, proved to be unsuccessful as the quality of the generated images was below what is required.

Model 3 was also proposed as an improvement on model 1. It is designed to generate labelled images of existing persons with new poses, in the style of the target dataset. It successfully generates labelled images with the person in the desired pose. It performs better than model 1 and also includes additional features which makes it superior.

A novel loss function was proposed and implemented in model 4. It allows model 4 to generate much more realistic persons compared to the other methods. With the new loss function, model 4 does not require both the source dataset and the target dataset to be labelled and contain the same identities. Only the source dataset needs to be labelled. Removing the requirement of a labelled target dataset allows model 4 to be much more versatile and useful. It can create labelled dataset in the style of an unlabelled dataset using any existing labelled dataset.

Overall, the project presents two viable methods to improve person re-ID networks. The first is model 3 which, based on the quality of the generated images can be compared to state-of-the-art methods regarding pose transfer for person re-ID images. In addition, the generated images form model 3 also changes the style of the image and generates realistic backgrounds to match the target dataset. The other viable method is model 4, which is directly comparable to state-of-the-art methods regarding image style changes for person re-ID images. Model 4 can theoretically improve on some of these state-of-the-art methods, as model 4 also changes the

background and not just the style.

The models presented have not been optimised in any significant way. Future work requires optimising the models, for example by performing a hyperparameter search. With the optimised models, a qualitative comparison with state-of-the-art methods can be made. In addition, implementing the *variable* version of the proposed loss would be very interesting as it has similar control over the foreground and background generation as model 4 but can also control the pose of the generated person like model 3.

Appendix A Code

```
# get input mask
if self.input_nc == 4: # If 4 input channels then find and concatenate the mask
    # path to the input mask
    M_path = "./datasets/mask/bounding_box_train_seg/"+AB_path.rpartition('train/')[2]
    M_path = M_path.rpartition('+')[0]+".jpg"
    # retrieving the input mask
    M = Image.open(M_path).convert('RGB')

    # Append the mask to the input image
    A = torch.cat((A, M[0][None, :, :]), 0)
```

Code Snippet 1: Simplified code of how the input mask is found and concatenated.

```
# get input and output masks
if self.input_nc == 5: # If 5 input channels then find and concatenate the input and output masks
    # path to the input mask (Mi)
    Mi_path = "./datasets/mask/bounding_box_train_seg/"+AB_path.rpartition('test/')[2]
    Mi_path = Mi_path.rpartition('+')[0]+".jpg"
    # path to the output mask (Mo)
    Mo_path = "./datasets/mask/bounding_box_train_seg/"+AB_path.rpartition('test/')[2]
    Mo_path = "./datasets/mask/bounding_box_train_seg/"+Mo_path.rpartition('+')[2]

    # retrieving the input mask and output mask
    Mi = Image.open(Mi_path).convert('RGB')
    Mo = Image.open(Mo_path).convert('RGB')

    # Append the input mask to the input image
    A = torch.cat((A, Mi[0][None, :, :]), 0)
    # Append the output mask to the input image
    A = torch.cat((A, Mo[0][None, :, :]), 0)
```

Code Snippet 2: Simplified code of how the input mask and output mask are found and concatenated.

```
    def get_foreground_loss(self): # matching the generated foreground with the target foreground
        # get the input mask (Mi)
        Mi = self.Mi.clone()
        # adjust the mask so that it is 0 for the background and 1 for the foreground
        Mi = Mi.add_(1).mul_(0.5)

        # multiply the generated image by the input mask to remove the background
        self.generatedForeground = self.fake_B[:,0:3] * Mi # generated foreground
        # multiply the target image by the input mask to remove the background
        self.targetForground = self.real_A[:,0:3] * Mi # target foreground

        # get the L1 loss between the generated foreground and the target foreground
        # and multiply by a weight
        self.loss_G_foreground = self.criterionL1(self.generatedForeground,
                                                  self.targetForground) * self.opt.foreground_L1
        return self.loss_G_foreground

    def get_background_loss(self): # matching the generated background with the target background
        # get the input mask (Mi)
        Mi = self.Mi.clone()
        # invert and adjust the mask so that it is 1 for the background and 0 for the foreground
        Mi_inv = Mi.mul_(-0.5).add_(0.5)
        # get the output mask (Mo)
        Mo = self.Mo.clone()
        # invert and adjust the mask so that it is 1 for the background and 0 for the foreground
        Mo_inv = Mo.mul_(-0.5).add_(0.5)

        # multiply the generated image by the inverted input mask and inverted output mask to
        # remove the foreground which only leaves the background that overlaps with the
        # target background
        self.generatedBackground = self.fake_B[:,0:3] * Mi_inv * Mo_inv # generated background
        # multiply the target image by the inverted input mask and inverted output mask to
        # remove the foreground which only leaves the background that overlaps with the
        # generated background
        self.targetBackground = self.real_B[:,0:3] * Mi_inv * Mo_inv # target background

        # get the L1 loss between the generated background and the target background
        # and multiply by a weight
        self.loss_G_background = self.criterionL1(self.generatedBackground,
                                                  self.targetBackground) * self.opt.background_L1
        return self.loss_G_background
```

Code Snippet 3: Simplified code of how the foreground and background losses are calculated (*static* variation).

A.1 DataSplit script

The DataSplit.py script automatically creates a dataset composed of a camera pair in the required format for the GAN. To run DataSplit.py the following command should be called.

```
python DataSplit.py -A 5 -B 6 -G 6
```

Code Snippet 4: Creating a dataset containing camera 5 and 6 to use when generating images for camera 6.

Note the options "-A" and "-B" represent *cameraA* and *cameraB* respectively. Option "-G" choses which camera the GAN will generate images for. Other options are described in the user guide in section /ref.

The operation of DataSplit.py is twofold. First it creates a folder "A/test" by copying *cameraA* images that do not share the same identity and sequence as any

image from *cameraB*. These images are later used when testing the GAN. Similarly, folder "B/test" is created with the same process. This is shown in code snippet 5.

```
## Creating the test folder for cameraA ##
maskNumber = 0
for i in AllCameraA: # loops through all images from cameraA
    match = False
    for j in AllCameraB: # loops through all images from cameraB
        if i[-23:-18]+i[-16:-14] == j[-23:-18]+j[-16:-14]: #same identity and sequence
            match = True
    if match == False: # Non-overlapping identities
        maskNumber += 1
        if maskNumber >= len(AllCameraB): # loops the mask assignment around
            maskNumber = 0

        # Copy the image and append to the original name a "mask name" to be
        # used as the output mask if needed.
        os.system("cp "+str(i)+" "+str(args.output)+folderName+"A/test/"+str(i[52:-4])
                  +"+"+AllCameraB[maskNumber][52:-4]+".jpg")
        testANames.append(i[-23:-14]) # stores which images have been used for testing
```

Code Snippet 5: Simplified code of how the training images are selected. This code snippet shows the process for *cameraA* and is very similar for *cameraB*.

Note the naming scheme, where the training images are now labelled by their original name and a name for the output mask (also referred to as the target mask). The output mask represents the desired pose of the person in the generated image. The input mask represents the pose of the person in the input image. This can be ignored if an output mask is not wanted during training of the GAN but if required helps match a specific mask to an image. Also note that no output masks exist for the identities in the testing images as they never appear in the target camera. Instead, masks of other identities from the target camera are picked at random. This creates a few problems as the output mask might not be a good match for a specific person. For example, transforming a person without a backpack into a pose of a person with a backpack can be problematic.

The second process involves pairing together the training images. Importantly, testing images are not used for training as they would corrupt the test results. The *cameraA* images are paired together with *cameraB* images if they have the same identity and the same sequence. To produce a larger training dataset, each *cameraA* image is paired with each *cameraB* image with the same identity. This is visualised in figure 11. Code snippet 6 shows how these images are paired.

```
## Creating trainA and trainB ##
for i in AllCameraA: # loops through all cameraA images
    used_for_testing = False
    for k in testANames: # loops through all cameraA testing images
        if i[-23:-14] == k: # check if this image is used for testing
            used_for_testing = True
    if used_for_testing == False: # if not used for testing (thus overlapping identities)
        for j in AllCameraB: # loops through all cameraB images
            if i[-23:-18]+i[-16:-14] == j[-23:-18]+j[-16:-14]: #same identity and sequence

                # Copy the cameraA image to "A/train" and append the cameraB name
                # resulting in the name being: cameraA_name+cameraB_name.jpg
                os.system("cp "+str(i)+" "+str(args.output)+folderName+"A/train/"+str(i[52:-4])
                          +"+"+str(j[52:-4])+".jpg")

                # Copy the cameraB image to "B/train" and the name must be the same
                # as the cameraA copy
                os.system("cp "+str(j)+" "+str(args.output)+folderName+"B/train/"+str(i[52:-4])
                          +"+"+str(j[52:-4])+".jpg")
```

Code Snippet 6: Simplified code of how the training images are paired together depending on their ID labels and sequence.

After creating the "A/train", "B/train", "A/test" and "B/test" folders, two final folders "train" and "test" are created. The "train" folder is composed of either "A/train" or "B/train" images depending whether *cameraA* or *cameraB* images are being generated. For the "train" folder, images from "A/test" and "B/test" are merged into a single image file, as this is the required format to train the GAN.

Appendix B Additional results

Model	$\text{Baseline}_{\text{c5c6}}$	Model 1	$\text{Baseline}_{\text{c3c6}}$	Model 1
GAN training data	-	c5c6	-	c3c6
Person re-ID training data	c5c6	c5c6+G2	c3c6	c3c6+G3
mAP	**35.4%**	**35.4%**	36.4%	**37.6%**
Rank-1	56.1%	**57.3%**	57.5%	**58.7%**
Rank-5	73.3%	**74.4%**	72.4%	**74.9%**
Rank-10	80.1%	**80.8%**	79.1%	**81.5%**
Rank-20	85.9%	**85.9%**	84.7%	**85.5%**

Table 13: The performance of the baseline person re-ID model using different training data. Best results are in **bold**.

Appendix C Safety, Legal and Ethics

C.1 Safety

This project revolves around programming and thus safety hazards are at a minimum. However, precautions need to be made to decrease the chance of compromising or damaging my own or Imperial's IT systems. The largest safety concern is

downloading malicious data from the internet. This means extra care must be taken when downloading the required datasets. Running code found on the internet can also be dangerous and must be done with due diligence.

C.2 Legal

The EU General Data Protection Regulation (GDPR) [32] is a regulation protecting personal data. Using people's personal data without consent can result in heavy fines and must be avoided at all costs. The data required for this project includes personally identifiable images and thus great care must be taken to ensure this data is acquired legally and conforms to the regulation. The datasets that will be used in this project are well-known, open and comply with the EU's GDPR.

Additionally, care must be taken to not infringe on patented or copyrighted material.

C.3 Ethics

The idea of surveillance and tracking people is an ethical dilemma. One can argue that surveillance cameras prevent crime and can assist authorities in catching criminals and bring them to justice. On the other hand, one can argue that surveillance cameras and tracking people is an invasion of privacy, especially as most people caught in these cameras are innocent. As today's society heavily relies on surveillance, this project does not raise any ethical issues.

References

[1] Itseez, "Open source computer vision library," https://github.com/itseez/opencv, 2015.

[2] M. Farenzena, L. Bazzani, A. Perina, V. Murino, and M. Cristani, "Person re-identification by symmetry-driven accumulation of local features," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*. IEEE, 2010, pp. 2360–2367.

[3] L. Wei, S. Zhang, H. Yao, W. Gao, and Q. Tian, "Glad: global-local-alignment descriptor for pedestrian retrieval," in *Proceedings of the 2017 ACM on Multimedia Conference*. ACM, 2017, pp. 420–428.

[4] C. Su, J. Li, S. Zhang, J. Xing, W. Gao, and Q. Tian, "Pose-driven deep convolutional model for person re-identification," in *2017 IEEE International Conference on Computer Vision (ICCV)*. IEEE, 2017, pp. 3980–3989.

[5] L. Wei, S. Zhang, W. Gao, and Q. Tian, "Person transfer gan to bridge domain gap for person re-identification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 79–88.

[6] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[7] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014, pp. 2672–2680.

[8] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," *arXiv preprint*, 2017.

[9] M. Mirza and S. Osindero, "Conditional generative adversarial nets," *arXiv preprint arXiv:1411.1784*, 2014.

[10] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.

[11] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," *arXiv preprint*, 2017.

[12] W. Deng, L. Zheng, G. Kang, Y. Yang, Q. Ye, and J. Jiao, "Image-image domain adaptation with preserved self-similarity and domain-dissimilarity for person reidentification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 1, no. 2, 2018, p. 6.

[13] J. Bromley, I. Guyon, Y. LeCun, E. Säckinger, and R. Shah, "Signature verification using a" siamese" time delay neural network," in *Advances in neural information processing systems*, 1994, pp. 737–744.

[14] Z. Zhong, L. Zheng, Z. Zheng, S. Li, and Y. Yang, "Camera style adaptation for person re-identification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 5157–5166.

[15] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2818–2826.

[16] J. Liu, B. Ni, Y. Yan, P. Zhou, S. Cheng, and J. Hu, "Pose transferrable person re-identification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4099–4108.

[17] Z. Cao, T. Simon, S.-E. Wei, and Y. Sheikh, "Realtime multi-person 2d pose estimation using part affinity fields," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2017, pp. 1302–1310.

[18] M. Tian, S. Yi, H. Li, S. Li, X. Zhang, J. Shi, J. Yan, and X. Wang, "Eliminating background-bias for robust person re-identification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 5794–5803.

[19] W. Li, R. Zhao, and X. Wang, "Human reidentification with transferred metric learning," in *Asian Conference on Computer Vision*. Springer, 2012, pp. 31–44.

[20] W. Li, R. Zhao, T. Xiao, and X. Wang, "Deepreid: Deep filter pairing neural network for person re-identification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 152–159.

[21] L. Zheng, L. Shen, L. Tian, S. Wang, J. Wang, and Q. Tian, "Scalable person re-identification: A benchmark," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 1116–1124.

[22] P. Felzenszwalb, D. McAllester, and D. Ramanan, "A discriminatively trained, multiscale, deformable part model," in *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*. IEEE, 2008, pp. 1–8.

[23] Z. Zheng, L. Zheng, and Y. Yang, "Unlabeled samples generated by gan improve the person re-identification baseline in vitro," *arXiv preprint arXiv:1701.07717*, vol. 3, 2017.

[24] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in neural information processing systems*, 2015, pp. 91–99.

[25] M. Bertalmio, G. Sapiro, V. Caselles, and C. Ballester, "Image inpainting," in *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*. ACM Press/Addison-Wesley Publishing Co., 2000, pp. 417–424.

[26] M. Mathieu, C. Couprie, and Y. LeCun, "Deep multi-scale video prediction beyond mean square error," *arXiv preprint arXiv:1511.05440*, 2015.

[27] C. Song, Y. Huang, W. Ouyang, and L. Wang, "Mask-guided contrastive attention model for person re-identification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 1179–1188.

[28] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," *NeurIPS workshop*, 2017.

[29] junyanz, "pytorch-cyclegan-and-pix2pix," https://github.com/junyanz/pytorch-CycleGAN-and-pix2pix, 2019.

[30] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[31] K. Zhou, "Deep person re-id," https://github.com/KaiyangZhou/deep-person-reid, 2018.

[32] Council of European Union, "Council regulation (EU) no 2016/679," https://eur-lex.europa.eu/eli/reg/2016/679/oj, 2016.