

Image Generation using Generative Adversarial Networks for Training Data Augmentation in Person Re-Identification Models.

Laura Álvarez González

Advisors:

PhD Víctor Uc Cetina

PhD Anabel Martín González

Autonomous University of Yucatan

Faculty of Mathematics

Master's Degree in Computer Science

Abstract This work proposes the use of generative adversarial networks (GANs) for data augmentation in order to train and improve the performance of convolutional neural networks in person re-identification tasks. The methodology is based on the StyleGAN model, which is capable of generating synthetic images of people with specific desired features, called styles. The main motivation is to study the capacity of GANs to generate synthetic images that can increase the number of images available to train a convolutional neural network from a limited number of them. A convolutional neural network for person re-identification in images obtained from multiple cameras located in different places will be used as a case study. The problem of person re-identification is currently of interest for the development of more precise video surveillance systems.




Keywords: Generative Adversarial Networks · Convolutional Neural Networks · Person Re-identification.

Contents

Table of Contents	i
1 Introduction	1
1.1 Objectives	2
1.1.1 Specific objectives	3
1.2 Thesis Organization	3
2 State of the art	5
2.1 Style transfer between different domains	10
2.2 Pose modification	16
2.3 Random Artificial Images	20
2.4 State-of-the-art Reviews	23
3 Theoretical Framework	25
3.1 Artificial Neural Networks	26
3.2 Generative Adversarial Networks	30
3.2.1 StyleGAN	32
3.3 Re-identification model	41
4 Methodology	43
4.1 Generation of Artificial Images	45
4.2 Re-identification model	54
5 Experimental Results	57
5.1 Image Generation	57
5.2 Re-identification	72
6 Conclusions and Future Work	83

List of Figures

1.1	Example of images obtained by security cameras. The green boxes indicate images corresponding to the same person. . . .	2
2.1	Style transfer. New images with different styles are generated through an input image.	7
2.2	Posture modification. A new image of a person with the given posture is generated through an input image and the heat map of the posture.	8
2.3	On the left, real images from the Market-1501 database (1). On the right, artificial images generated with Stylegan3 (2). .	8
2.4	Timeline of the state of the art. ■ State of the art reviews. ■ Style transfer between different domains. ■ Pose modification. ■ Random artificial images.	9
2.5	Style transfer. Given an input image, its style, tonality, contrast, and illumination are modified based on the styles of other cameras without changing the structure.	10
2.6	Applying CamStyle (3) to 2 domains. Images from domain A are converted to domain B and vice versa.	11
2.7	Examples of images from different databases. Market-1501 (1), DukeMTMC-ReID (4), CUHK03 (5), and VIPeR (6).	12
2.8	Example of DG-GAN (7). Style transfer, transfers the appearance of the left image to all the images on the right, combining appearance and structure.	13
2.9	Types of pose extraction. By joints or heatmaps.	16
2.10	Posture transformation using a template.	17

2.11	Different types of labeling. Top left: labeling of a real image corresponding to a person. LSRO: proportional labeling, k = number of classes. MpRL: labeling based on similarity. SLSR: proportional labeling based on the group to which the person belongs, p_c = distribution in the classes of the group to which they belong.	21
3.1	Example of a neural network consisting of an input layer of three neurons, a hidden layer, and an output layer of two neurons.	27
3.2	Example of a neuron composed of three input elements x and their respective weights w	27
3.3	Generic GAN architecture.	31
3.4	Graphical representation of a Generator. Multidimensional space where each position represents an image.	33
3.5	Example of a 2-dimensional Generator, where each coordinate belongs to an image. When inputting the latent vector $[0,0]$ as output, the upper left image is obtained.	33
3.6	The problem of entanglement. From image 1, an interpolation is made to image 2. a) interpolation without the problem of entanglement. b) interpolation with the problem of entanglement. As can be seen, interpolation a) is much smoother and more coherent.	34
3.7	The latent vector is modified in the last layers of AdaIN  . Changes can be observed in fine features such as hair color, eye color, and skin tone.. . . .	36
3.8	The latent vector is modified in the middle and last layers of AdaIN  . Changes in both medium and fine features can be observed. The face structure is modified while the posture remains the same.. . . .	36
3.9	The latent vector is modified in the early layers of AdaIN  . Changes in coarse characteristics, such as the image structure, can be observed, while the hair or skin color remains the same..	37
3.10	Left - real images. Right - obtained through the encoder. . . .	38
3.11	Izquierda - imágenes reales. Derecha - obtenidas a través del codificador.	39
4.1	Architecture - Generation of artificial images.	44

4.2	Architecture - Re-identification.	44
4.3	StyleGAN. Architecture of the Generator and Discriminator (8)	46
4.4	Diagram - Transfer Learning.	46
4.5	Artificially generated people using StyleGAN3. None of these people exist in reality.. . . .	47
4.6	Images from the Market-1501 dataset.	48
4.7	Starting from the initial latent vector, new images are gener- ated. To mix styles in the blue layers, the latent vector from other images is introduced, while the white layers receive the latent vector from the original image.	50
4.8	Example of interpolation from an input image of a girl to her adulthood.	50
4.9	Training of the StyleGAN3 encoder.	51
5.1	Evolution of the model's performance through the FID metric at different epochs during the StyleGAN3 training.	59
5.2	Artificial images generated randomly after training.	60
5.3	Application of YoloV4 tiny on images from the Market-1501 database using different thresholds. The error percentage rep- resents the images that were not classified as pedestrians. . . .	62
5.4	Histogram of the SSIM (9) metric on the Market-1501 dataset. Number of images that obtained the same SSIM value. One image per person was selected and compared to the rest of the images of that same person. As can be seen, most of the images are around the threshold of 0.75 and above.	63
5.5	A) Real images from the Market-1501 dataset. B) Artificial images.	64
5.6	Seed - It is the image generated randomly to which its latent vectors will be modified to change its mean characteristics. Generated - They are the images that have been generated by modifying the latent vectors of the seed image.	65
5.7	Example of some discarded images using the Yolo V4 tiny model for pedestrian detection.	66
5.8	Example of some images discarded using the SSIM (9) metric. Starting from one of the images of a person (original image), it is compared with the rest of the generated images of that same person.	66
5.9	LPIPS	68

5.10	L2	68
5.11	MOCO	68
5.12	LPIPS	69
5.13	L2	69
5.14	MOCO	69
5.15	Pairs of images. The real image is on the right, and its counterpart obtained through the encoder in the latent space of StyleGAN3 is on the left. It can be observed that the model performs better when showing the full body, although it does not reach the quality of randomly generated images.	70
5.16	Seed - real image. Generated - are the images generated by modifying the latent vectors of the seed image.	71
5.17	In the left plot, we can see the loss function during the training and validation phases of the base model without any additional generated people. In the right plot, we can see the Rank1 error percentage during the training and validation phases. . .	73
5.18	Adding 100 persons. Left , loss function during training and validation. Right , Rank1 error percentage during training and validation.	73
5.19	Adding 200 persons. Left , loss function during training and validation. Right , Rank1 error percentage during training and validation.	74
5.20	Adding 320 persons. Left , loss function during training and validation. Right , Rank1 error percentage during training and validation.	74
5.21	Performance of some models trained with different numbers of artificially generated persons (see Table 5.10). Adding 0, 40, 80, 120, 160, 200, 240, 280, and 320 persons.	76
5.22	Results of re-identification model. Comparison between the results of the base model (■) and the model trained after adding 280 artificial persons (■). The query is the image of the person to be searched for, and the following images represent the model's output, with green indicating a correct match and red indicating an error.	77
5.23	Base, without adding any images. Left , loss function during training and validation. Right , Rank1 error percentage during training and validation.	79

5.24	Adding 35 images to each person. Left , loss function during training and validation. Right , Rank1 error percentage during training and validation.	79
5.25	Adding 100 images to each person. Left , loss function during training and validation. Right , Rank1 error percentage during training and validation.	80
5.26	Performance of some models trained with different number of artificial images per person (see Table 5.11). Adding 0, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 110 and 120 images per person.	81

Chapter 1

Introduction

Person re-identification is a technique used in the field of artificial intelligence and machine learning to recognize a person in different images or videos, even if they have different angles, lighting, or clothing. This technique is used in various applications, such as security surveillance, identification of people in digital images, and analysis of behaviors in videos. It is based on the use of machine learning models that learn to recognize the characteristics that identify a person in different images or videos. These models are trained with a dataset that contains images or videos of people, along with information about the characteristics that identify each person. Person re-identification can be challenging due to the variability of the characteristics that identify a person, such as clothing, hairstyle, and other aspects that can change their appearance. Additionally, it can be a challenge if there is a limited amount of training data due to the privacy of people appearing in images or videos. To overcome these challenges, image and video pre-processing techniques, as well as deep learning techniques, can be used to allow the model to adapt to variations in a person's appearance and recognize relevant features in low-quality images or videos.

Currently, the most prominent training datasets for person re-identification are very limited because they do not contain a large number of images. For example, Market1501 includes only 1501 people recorded with 6 different cameras, while DukeMTMC-reID has 702 people in 8 different cameras. As shown in Fig. 1.1, various challenges are present for person re-identification in images, such as low image resolution, variations in lighting and contrast, as well as other factors that complicate the task, such as changes in clothing, the presence of objects like backpacks or sweaters, and the presence of

obstacles or people in the background that limit visibility of the person of interest in an open space.



Figure 1.1: Example of images obtained by security cameras. The green boxes indicate images corresponding to the same person.

This study investigates the use of a generative adversarial network, along with data augmentation techniques, to train a person re-identification model. The generative adversarial network is a type of machine learning model that is used to generate synthetic images or videos that can be used as additional training data. Various techniques for expanding training data are analyzed, such as image generation and feature expansion, and their effectiveness in training a person re-identification model using a generative adversarial network is evaluated. Additionally, the results obtained with a model trained with non-expanded data are compared.

1.1 Objectives

The general objective of this research is to generate artificial person images from a reduced set of training images to improve the performance in person re-identification models.

1.1.1 Specific objectives

The specific objectives are as follows:

1. Investigate data augmentation techniques for training a person re-identification model.
2. Analyze the use of a generative adversarial network to generate synthetic images that can be used as additional training data for a person re-identification model.
3. Evaluate the effectiveness of data augmentation techniques and the generative adversarial network in training a person re-identification model.
4. Contribute to the development of person re-identification techniques and provide a basis for future research in this area.

1.2 Thesis Organization

- Chapter 1 - Introduction

The introduction chapter of this thesis aims to present the context and general objective of the research. First, the context in which the thesis is developed is presented, including a brief description of the field of person re-identification and its importance in applications such as security surveillance and analysis of behaviors in videos.

- Chapter 2 - State of the Art

The objective of this chapter is to present a review of the relevant literature in the field of person re-identification and the use of generative adversarial networks to expand training data.

- Chapter 3 - Theoretical Framework

This chapter presents the theoretical framework in which the research is developed. First, a review of the basic concepts of generative adversarial networks is presented, including a description of how these networks are used to generate synthetic images that can be used as additional training data. Second, a review of the basic concepts in the field of person re-identification is presented, including a description of

the techniques used to recognize a person in different images or videos, as well as the challenges faced in this task.

- Chapter 4 - Methodology

The data analysis methodology used is described, including the data augmentation techniques used, as well as the methodology for evaluating the performance of a person re-identification model.

- Chapter 5 - Experimental Results

First, the results of the application of data augmentation techniques in the dataset used in the research, including the number of generated augmented data, are presented. Second, the results of the evaluation of the performance of a person re-identification model trained with augmented data using a generative adversarial network are presented. Third, the results obtained with a model trained with non-expanded data are compared, and the differences in performance between both models are analyzed.

- Chapter 6 - Conclusions and Future Work

The objective of this chapter is to present the conclusions obtained in the research and propose future work. The general conclusions of the research are presented, including a summary of the results obtained and a discussion of their significance and relevance in the field of person re-identification and the use of generative adversarial networks to expand training data, as well as the limitations of the research. Future lines of work are proposed to overcome these limitations and continue the development of research in this area.

Chapter 2

State of the art

The 2014 article "Generative Adversarial Networks" by Ian Goodfellow *et al.* (10) presents a new class of neural networks called generative adversarial networks (GANs). These networks are a type of machine learning model used to generate synthetic images or videos from a data set. GANs are made up of two neural networks trained simultaneously, a generator and a discriminator. The generator network is responsible for generating synthetic images or videos, while the discriminator network evaluates the quality of the images or videos generated by the generator network. The purpose of generative adversarial networks is for the generator network to produce images or videos with a high degree of similarity to real objects and scenarios, such that the discriminator network cannot differentiate between authentic images or videos and those generated synthetically by the generator network. The article presents experiments demonstrating the ability of GANs to generate high-quality synthetic images or videos, and discusses the potential of these networks in applications such as 3D image generation, improving image or video quality, and analyzing behaviors in videos.

Since this initial article, new architectures have been developed that improve the quality of generated images, such as the architecture proposed in 2017, CycleGan (11), which is capable of improving the quality of generated images by transferring the style or domain of one group of images to another group using two generative adversarial networks. Due to the improved performance of generative adversarial networks, they are now being used to increase the training data that enable the improvement of performance in machine learning models where databases are limited. In this case, as can be seen in Fig. 2.4, there has been a notable increase in the study of the use of

generative adversarial networks to increase training data in re-identification models since 2018.

The articles have been grouped into four categories corresponding to different methods used for the generation of new artificial images.

- Style transfer between different domains

Using a real input image, artificial images can be generated using different styles or domains with which the model has been trained. This enables the transfer of the style of one data set to another, such as the transfer of the style of a painting to a real photograph. In the generated images, changes can be observed with respect to the original image, such as colors, tones, lighting, among others, but there is no change in the structure of the image (see Fig. 2.1).

- Posture modification

The goal of these networks is to enable the generator network to produce images of people in various postures that are indistinguishable from real images in the data set used for training. A real image of a person and a heat or joint map corresponding to a different posture skeleton are used as input. This way, it is possible to expand the training data set of a person re-identification model by adding images with varied postures of the same person (see Fig. 2.2).

- Random artificial images

Artificial images are generated randomly and labeling techniques are applied to them (see Fig. 2.3).

- State of the art review

State of the art writings are used by researchers and professionals to obtain a general overview of a field of study and to identify the main trends and challenges in that field. They are also used as a basis for designing new research and projects. In this case, we focus on articles on the state of the art that focus on the specific topic of the use of generative adversarial networks to increase training data in person re-identification models.

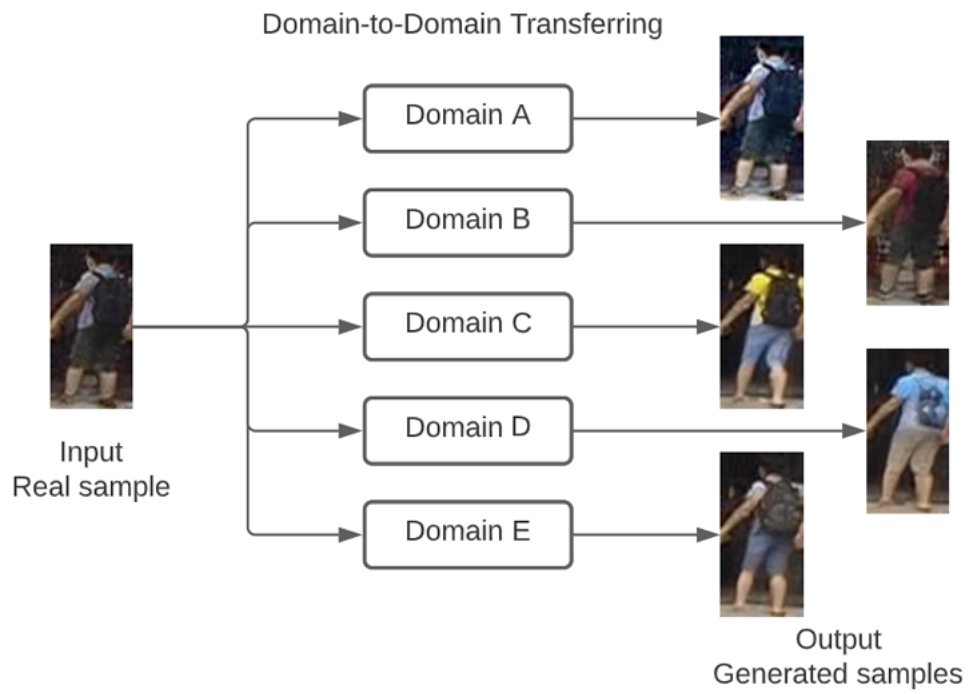


Figure 2.1: Style transfer. New images with different styles are generated through an input image.

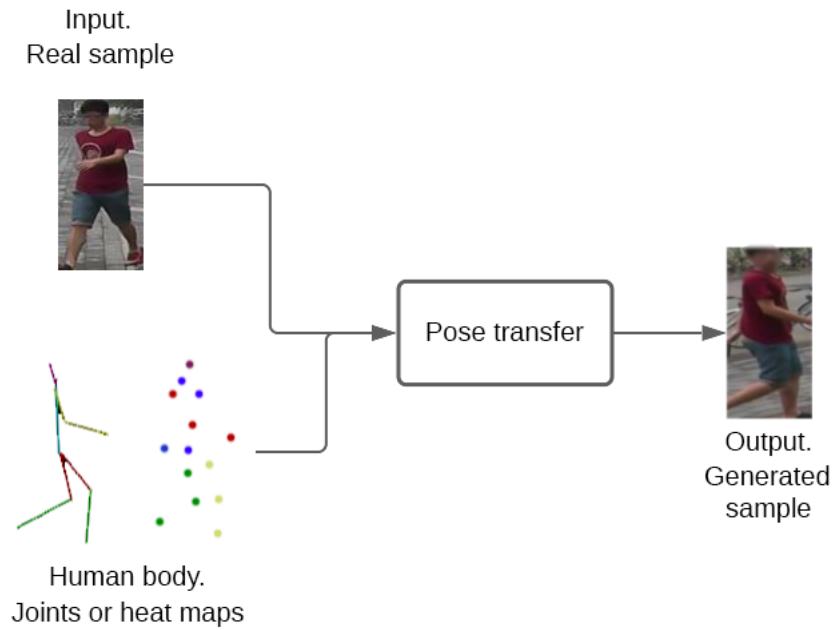


Figure 2.2: Posture modification. A new image of a person with the given posture is generated through an input image and the heat map of the posture.



Figure 2.3: On the left, real images from the Market-1501 database (1). On the right, artificial images generated with Stylegan3 (2).

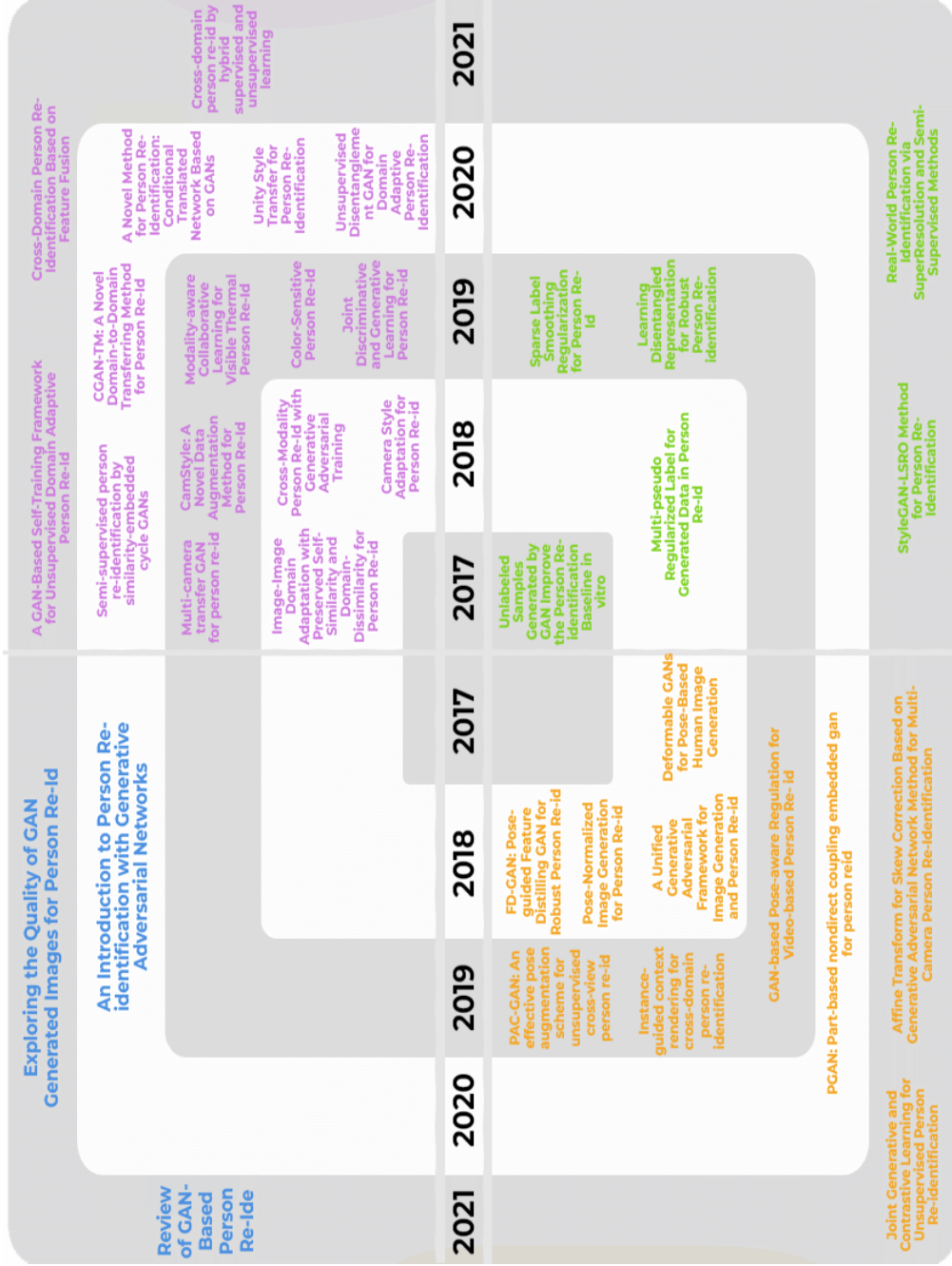


Figure 2.4: Timeline of the state of the art.

■ Style transfer between different domains. ■ Pose modification. ■ Random artificial images.

2.1 Style transfer between different domains

Images obtained from cameras often have different resolutions or positions. This can cause changes in lighting and tonality among other aspects. One way to generate new data is through domain transfer or domain adaptation, which is based on the idea of transferring the style of one or more images to other images (Fig. 2.5) without modifying the image structure or background. This means that there is no change in the pixel positions, leaving the images exactly the same.



Figure 2.5: Style transfer. Given an input image, its style, tonality, contrast, and illumination are modified based on the styles of other cameras without changing the structure.

In the literature on style transfer, several ways of tackling the problem can be found. In 2017, the architecture of a generative adversarial network called CycleGAN (11) appeared, which can learn the style of one or more images and transfer it to other different images, i.e., transfer the style from one domain to another. This was a milestone within generative adversarial networks, and from 2018 onwards, many works using this architecture began to appear, such as the work of Zhun Zhong et al.(3), where they propose CamStyle, a method that uses the CycleGAN generative adversarial network architecture(11) to transfer the style from one security camera to another. This can only transfer the style between two domains, limiting the architecture in such a way that it is necessary to generate a model for each pair of security cameras.



Figure 2.6: Applying CamStyle (3) to 2 domains. Images from domain A are converted to domain B and vice versa.

Following the same philosophy of transferring the style from one to another, in 2018 Pingyang Dai et al.(12) proposed the cmGAN model focused on style transfer to convert RGB and infrared camera images. This was the first article to use the RGB-Infrared Cross-Modality Re-ID Dataset(13), which includes images from four infrared and two RGB cameras. In this case, the discriminator of the generative adversarial network is part of the feature extractor of the re-identification model. The input to the re-identification model will be an infrared image, and it should look for that person within the RGB images. In the previous articles, we encounter the limitation of transferring the style from one domain A to another domain B, requiring the duplication of the project for each different style. This, added to one of the biggest challenges faced by re-identification models, which is the low performance obtained when using images from another database in the tests of the re-identification model. Based on these problems, some works propose domain transfer between different databases and/or multiple domains (Fig. 2.7).

In 2018, as an improvement to the Camstyle architecture and in search of better performance when using the model on different databases, the M2M-GAN architecture was proposed (14). This architecture classifies the images of each database into subdomains, i.e., for each camera. It can transfer the sub-domain of domain A to a sub-domain of domain B, with supervised training and requiring all data from different databases to be manually labeled.



Figure 2.7: Examples of images from different databases. Market-1501 (1), DukeMTMC-ReID (4), CUHK03 (5), and VIPeR (6).

On the other hand, in the work of Weijian Deng et al.(15), they attempted to globalize the model by developing the SPGAN architecture, which transfers images from the style of one database to another. It is trained in an unsupervised manner and consists of a siamese network(16) (SiaNet) and a CycleGAN (11). Similarly, Shuren Zhou et al.(17) proposed the CTGAN architecture, which reduces complexity by transferring styles from one domain to multiple domains in another database using only one generator and discriminator model. In this case, they used the architecture of a StarGAN(18) generative adversarial network.

Following this line of work, in 2020, Yingzhi Tang et al. proposed the CGAN-TM architecture (19), which converts images from one database to another using a CycleGAN (11) as a generative adversarial network. The innovation in this work is the use of Self-Labeled Triplet Net, which labels the generated artificial images to train the re-identification model in an unsupervised manner.

In the same year, Yacine Khraimeche et al. (20) presented the UD-GAN technique, which aims to improve the performance of the re-identification model through training with a database that contains labeled data and subsequently evaluating it on a different database that lacks labeled data.

Starting in 2019, more sophisticated architectures began to appear, such as the one proposed by Zhedong Zheng *et al.*(7). DG-NET uses two encoders that can extract the colors, appearance, and transfer those colors to another image where the person’s structure has been extracted (Fig.2.8). This model also uses the generative adversarial network architecture as a re-identification model.



Figure 2.8: Example of DG-GAN (7). Style transfer, transfers the appearance of the left image to all the images on the right, combining appearance and structure.

At the beginning of 2020, Rui Sun *et al.*(21) proposed the cTransNet architecture, based on the StarGAN(18) generative adversarial network. The goal of this architecture is to develop a single generator capable of generating multiple images from an input image, with the different styles of each camera. On the other hand, Yang Yang *et al.* (22) proposed the Color Translation GAN (CTGAN) architecture in their work “Color-Sensitive Person Re-Identification”, which focuses on distinguishing between different clothing colors while maintaining person identity coherence with the color of their clothes. CTGAN is capable of identifying and modifying the colors of upper clothing (such as hoodies or shirts) and lower clothing (such as pants).

Chong Liu *et al.*(23) proposed UnityGAN, which generates artificial images in a style that is a combination of all the other styles, without the need to learn to transfer by pairs. It eliminates differences between styles, leaving a unique style. The architecture is based on DiscoGAN(24) and CycleGAN (11).

In recent years, semi-supervised and unsupervised models have appeared

due to the difficulty of obtaining a large number of labeled images for re-identification models. In 2020, Xinyu Zhang *et al.*(25) proposed SECGAN, similarity-embedded CycleGANs(11). Due to the limitation of labeled data, this semi-supervised method trains with labeled and unlabeled data alternately and uses both encoders from each of the cameras, A and B, as discriminatory feature extractors.

The following works make use of labeled images to transfer their style to an unlabeled domain. In 2021, Zhiqi Pang *et al.*(26) presented a hybrid method that combines supervised and unsupervised techniques. This method uses a TC-GAN architecture to generate labeled artificial images and transfer the person from the input image to the background of the desired style image. Additionally, the authors propose the DFE-Net re-identification model, which employs a modified version of the ResNet-50(27) network pre-trained on the ImageNet (28) dataset, with both real and artificially generated images as inputs. The network is used as a feature extractor for image comparison. Also in 2021, Yuanyuan Li *et al.*(29) proposed the use of a CycleGAN(11) and a siamese neural network (16). In this case, labeled data is used as the input domain and the styles of images from an unlabeled domain are transferred. Finally, Xianjun Luo *et al.*(30) propose the FFGAN architecture. The CycleGAN(11) architecture is used to generate artificial images. The innovation of the work is found in the re-identification model, which is capable of extracting the local, global, and semantic characteristics of each image to improve model performance.

Year	Name	GAN Model	Transfer	Database	Github
2018	CamStyle (3)	CycleGAN	1 to 1	Same/Different	Yes
2018	cmGAN (12)	New	1 to 1	Same	No
2018	M2M-GAN (14)	New	Many to Many	Different	No
2018	SPGAN (15)	SiaNet CycleGAN	1 to 1	Different	No
2019	CTGAN (17)	StarGAN	1 to 1	Different	No
2019	DG-NET (7)	New	Combinations	Same	No
2019	CTGAN (22)	New	Combinations	Same	No
2020	CGAN-TM (19)	CycleGAN	1 to 1	Different	No
2020	UD-GAN (20)	New	1 to 1	Different	No
2020	cTransNet (21)	StarGAN	1 to Many	Same	No
2020	UnityGAN (23)	DiscoGAN/CycleGAN	1 to Generic	Same	No
2020	SECGAN (25)	CycleGAN	1 to 1	Same	No
2021	TC-GAN (26)	New	1 to 1	Same	No
2021	STrans (29)	New	1 to Generic	Same	No
2021	FFGAN (30)	CycleGAN	1 to 1	Same	No

Table 2.1: Table with all proposed network architectures. (Year) publication year, (Name) name of the proposed architecture, (GAN Model) whether they use or are based on an existing generative adversarial network, (Transfer) the domains between which styles are transferred, and (Database) whether the model has been globalized, i.e., whether training was done with one database and testing with another.

2.2 Pose modification

Person re-identification presents one of the biggest challenges due to the significant variation in a person's pose across different cameras. To address this issue, generating new data of the same person by modifying their pose using various architectures is proposed. The generation of new data is based on extracting the person from the original image, which can be done by obtaining the joints or heatmaps (Fig. 2.9), and adapting it to the desired pose to increase the quantity and variability of available data.

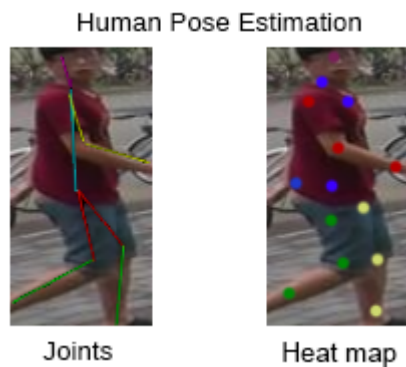


Figure 2.9: Types of pose extraction. By joints or heatmaps.

In 2018, Xuelin Qian et al.(31) presented the PN-GAN architecture, capable of generating artificial images of a person in eight different poses. The eight canonical poses were obtained using the K-Means algorithm on the distribution of all images in the database. The Open Pose tool(32) was used to generate the template, which detects 18 human body joints and their connections, and by the joint map of both images, it is possible to transfer the pose of each of the eight canonical poses to the input images, as shown in Fig. 2.10.

In a similar work, Aliaksandr Siarohin et al.(33) proposed the Deformable GAN, with the aim of generalizing the previous model. To do this, the model had to be trained in a supervised manner with pairs of images of the same person in different poses. They also used the Open Pose(32) method to obtain the human body joint map, decomposing it into 18 joints and a total of 10 subdivisions, head, torso, both arms, and legs. This model allowed transferring the pose of a person in an image A to another person in an

image B.

In another work, Yaoyu Li et al.(34) also used Open Pose(32) as a joint and color map extractor with 19 locations in different body parts, which provided robustness to the model. The training was done with pairs of images of the same person in different poses, similar to the previous work. The architecture allowed transferring the pose of one image to another.

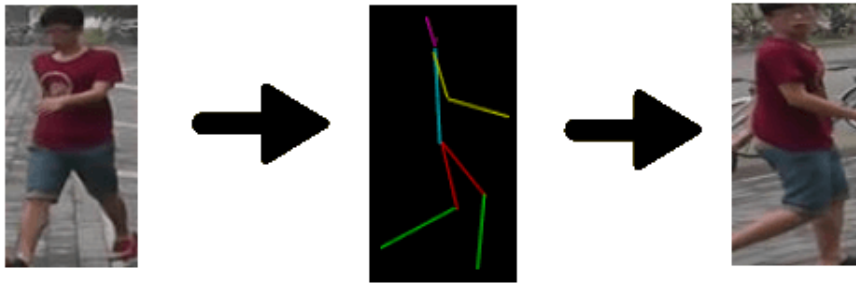


Figure 2.10: Posture transformation using a template.

In the 2018 article by Yixiao Ge et al.(35), the FD-GAN architecture was proposed for transferring the posture of one image to another. The network consisted of a generator and two discriminators, one for the person's identity and the other for the posture, where they used the PatchGAN(36) architecture. Using a ResNet-50, they extracted a feature vector from the input image. For the target posture, they used an 18-channel map, where each represents the location of a posture reference point and converted it to a Gaussian heatmap. Subsequently, it was converted to a feature vector of size 128, and with these two data, the model generated a new image of the same person with the specified posture in the heatmap.

In 2019, Alessandro Borgia *et al.*(37), following the same line as the previous architectures, extracted the joints using the Open Pose(32) method, proposed an architecture that, instead of evaluating a specific image, evaluates the video sequences of a person's movement. Eight canonical postures are predefined, three facing forward, three facing backward, one looking sideways to the right, and one to the left. First, the video sequences of a person are obtained, and the corresponding images with the canonical postures are searched by Euclidean distance. If one of those eight postures does not exist, the artificial image is generated. The same occurs with all the people in the video, their sequences are obtained, and if there is no canonical position,

it generates one. By cosine distance, they compare the eight corresponding images with the eight canonical postures of the input person with all other people's image sequences. A classification is made, and the one with the lowest cosine distance is assigned as if it were the same person in the re-identification model.

In 2020, Chengyuan Zhang and other authors (38) presented the PAC-GAN model, composed of two parts. In the first part, CPG-Net, a conditional GAN is used to generate images of a person from camera A and then convert them to the viewpoint of camera B. New images with postures from different viewpoints of different cameras are also generated, thus increasing the amount of data. The model is trained with the joints extracted by Open Pose and the image itself. In the second part, they use Cross-GAN (39), developed by the same team, as a re-identification model. In Y. Zhang *et al.*(40) work, they propose PGAN. In this case, the posture is obtained with a heatmap, and by using two input images, they transfer the posture from one to another. This architecture improves the performance obtained with the 2018 FD-GAN(35) architecture.

In 2021, Ni Ziyang *et al.*(41) proposed a new architecture capable of correcting images so that the people appearing in them are centered and straight. To do so, the database was trained with images that indicated the correct position of the people. Additionally, in that same year, Hao Chen *et al.*(42) proposed the GLD architecture of the generative adversarial network, which uses a three-dimensional mesh that represents different postures and is capable of generating a new image of the same input person with the corresponding posture.

Year	Name	Pose extractor	Num. poses	Database	Github
2018	PN-GAN (31)(3)	Open Pose	8	Same	Yes
2018	Deformable GAN(33)	Open Pose	Other image	Same	Yes
2018	Yaoyu Li <i>et al.</i> (34)	Open Pose	Other image	Same	No
2018	FD-GAN(35)	Open Pose	Other image	Same	Yes
2019	Pose-aware Regulation (37)	Open Pose	8	Same	No
2020	PAC-GAN (38)	Open Pose	Other images	Same	No
2020	PGAN (40)	Open Pose	Other images	Same	Yes
2021	Ni Ziyang <i>et al.</i> (41)	None	None	Same	No
2021	GLD(42)	3D mesh	Not specified	Same	Yes

Table 2.2: Summary of all proposed architectures. (Year) presentation date of the article, (Name) name of the proposed architecture, (Pose extractor) method used to extract pose, (Num. poses) number of poses generated, (Database) whether the model has been globalized, i.e., if training was done with one dataset and testing with another.

2.3 Random Artificial Images

This section includes articles that generate random images of people with different poses, lighting, colors, and backgrounds (see Fig. 2.3). These images are labeled with different methods for the re-identification model training. Firstly, we will detail the articles where the 1980 LSR (*Label Smooth Regularization*) algorithm is used as a basis for labeling artificial images, which was first applied to a classification problem by Christian Szegedy *et al.* in 2015.

In 2017, the first work was presented where the re-identification model was trained with randomly generated images. Zhedong Zheng *et al.*(43) used the generative adversarial network proposed in 2016, DCGAN(44), to generate random data and labeled it with a technique they call LSRO (see Fig.2.11), *label smoothing regularization for outliers*, which is a modification of the work by Christian Szegedy *et al.*(45). This technique assigns the same value to the generated artificial image in all classes, i.e., it is uniformly distributed across all classes.

In 2019, Yan Huang *et al.*(46) presented MpRL, *Multi-pseudo Regularized Label* (see Fig.2.11). Unlike the previous work, this method generates a label based on the probability of similarity with each of the training classes. They use DCGAN (44) as the generative adversarial network. In the same year, Jean-Paul Aïme *et al.*(47) proposed to cluster images from the database using the K-Means classification algorithm for the training of the DCGAN generative adversarial network. They also propose a new labeling technique, SLSR *Sparse Label Smoothing Regularization* (see Fig.2.11), which labels artificial images with a partial distribution based on the group from which they were generated using the K-Means algorithm. The labeling technique is used to classify images in the database into different categories, such as gender, age, or clothing type. The LSRO (Label Smoothing Regularization Optimization) labeling technique has been widely used in the field of person re-identification but may have some problems, such as the creation of ambiguous or incorrect labels. The SLSR technique proposed in this work aims to solve these problems. Instead of creating precise labels for each image, SLSR labels images with a partial distribution. This means that the label for each image is based on the group to which that image belongs, rather than a precise label. This technique reduces ambiguity and the creation of incorrect labels. Regarding the LSR labeling process, Saleh Hussin *et al.*, 2021 (48), propose the use of the StyleGAN generative network (8) for creating new

data. To do so, they train with one of the most commonly used datasets in the field of person re-identification, and once the new images are generated, they apply the LSRO method (see Fig.2.11), proposed by Zhedong Zheng *et al.*, 2017(43), as explained earlier.



Figure 2.11: Different types of labeling. Top left: labeling of a real image corresponding to a person. LSRO: proportional labeling, k = number of classes. MpRL: labeling based on similarity. SLSR: proportional labeling based on the group to which the person belongs, p_c = distribution in the classes of the group to which they belong.

In 2019, Chanhom Eom *et al.*, 2021 (49) proposed a new architecture of generative adversarial network, IS-GAN, *identity shuffle GAN*, where artificial images are generated through interpolation between two real images, distinguishing between the upper and lower parts. The labels for the artificial image are the images that have generated the interpolation.

Lastly, in 2021, Limin Xia *et al.*, 2021 (50) proposed a new architecture of generative adversarial network. Firstly, they propose the MSSR model

(*Mixed-Space Super-Resolution model*) which improves the resolution of input images. They use the PGCN architecture (*Part-based Graph Convolutional Network*) to generate artificial images, and with the same network, they generate soft multi-labels for the artificial images.

Year	Paper	GAN Model	Labeling	Github	Github
2017	Zhedong Zheng <i>et al.</i> (43)	DCGAN	LSRO	Yes	Yes
2019	Yan Huang <i>et al.</i> (46)	DCGAN	MpRL	No	No
2019	Jean-Paul Aïme <i>et al.</i> (47)	DCGAN	SLSR	Yes	Yes
2019	Chanho Eom <i>et al.</i> (49)	IS-GAN	Interpolation	Yes	Yes
2021	Saleh Hussin <i>et al.</i> (48)	StyleGAN	LSRO	No	No
2021	Limin Xia <i>et al.</i> , 2021(50)	PGCN	<i>Soft Multi-Labels</i>	No	No

Table 2.3: Table with all proposed network architectures. (Year) year of paper publication, (Paper) name of proposed architecture, (GAN Model) whether or not they use or are based on an existing generative adversarial network, (Labeling) labeling technique used, (Github) whether or not the code is available on Github.

2.4 State-of-the-art Reviews

In recent years, there has been a great interest in the use of generative adversarial networks in person re-identification models. In 2019, Hamed Alqahtani *et al.*(51) provided a detailed introduction to the state-of-the-art in this field, describing different types of generative adversarial networks and 11 different architectures used for style transfer, LSRO labeling, and model globalization. Additionally, Zhiyuan Luo *et al.*(52) focused exclusively on architectures that generate artificial images by changing styles between different cameras or databases. Lastly, Yiqi Jiang *et al.*(53) conducted a detailed study on the quality of images generated by different generative adversarial network architectures in re-identification models, analyzing the details of these artificial images that influence the performance of the re-identification models. They concluded that not all artificially generated images are useful for improving the performance of re-identification models.

Chapter 3

Theoretical Framework

The theoretical framework of this research is based on theories and concepts related to data augmentation for training and person re-identification. Data augmentation for training refers to the technique used to increase the training dataset of a machine learning model. This technique can improve the performance of a model by adding more training data and allowing the model to learn more effectively. In this context, StyleGAN is a generative adversarial network used to generate synthetic images or videos. On the other hand, in the field of person re-identification, different models and techniques have been developed to recognize people in images and videos, which often require a large training dataset to function effectively. Both disciplines, generative adversarial networks and re-identification models, are part of a branch of artificial intelligence called machine learning, which provides a computer with the ability to learn.

Machine learning is a discipline that is based on the idea that a machine can acquire skills to perform complex tasks without being specifically programmed to do so. It focuses on the development of algorithms and techniques that allow a machine to learn automatically from data. It is possible to program a computer to learn in various ways, such as exploring the web, reading books, playing games, or interacting with people. A machine learning program can learn any task that can be mathematically coded. This discipline is used in a wide variety of applications, such as pattern recognition, result prediction, fraud detection, and natural language processing. Machine learning algorithms and techniques are used in applications such as recommendation systems, personal assistants, medical diagnosis systems, and email spam detection systems.

Within machine learning, there is a branch called deep learning. Deep learning is a machine learning technique that focuses on the development of deep neural network models. Deep neural networks are computing networks that are inspired by the functioning of the human brain and are used to perform complex tasks such as pattern recognition and image classification. It differs from other machine learning techniques in that it uses deep neural networks with many layers of processing. These processing layers allow deep neural networks to learn complex and abstract features in input data and use them to perform complex tasks. It is used in a wide variety of applications, such as pattern recognition, image classification, natural language processing, and synthetic content generation. Deep neural networks are used in applications such as voice recognition systems, recommendation systems, and medical diagnosis systems.

3.1 Artificial Neural Networks

To understand the importance of artificial neural networks (ANNs), it is necessary to understand what these networks are. An artificial neural network is a simplified model of the functioning of the brain, which is composed of basic processing units called neurons. These neurons are grouped and organized into layers: the input layer where each artificial neuron represents an input data, hidden layers, and the output layer that extracts the output or target data. ANNs are important because they allow solving complex problems in different fields such as medicine, robotics, computer science, and industry, among others, thanks to their ability to learn autonomously and process large amounts of information.

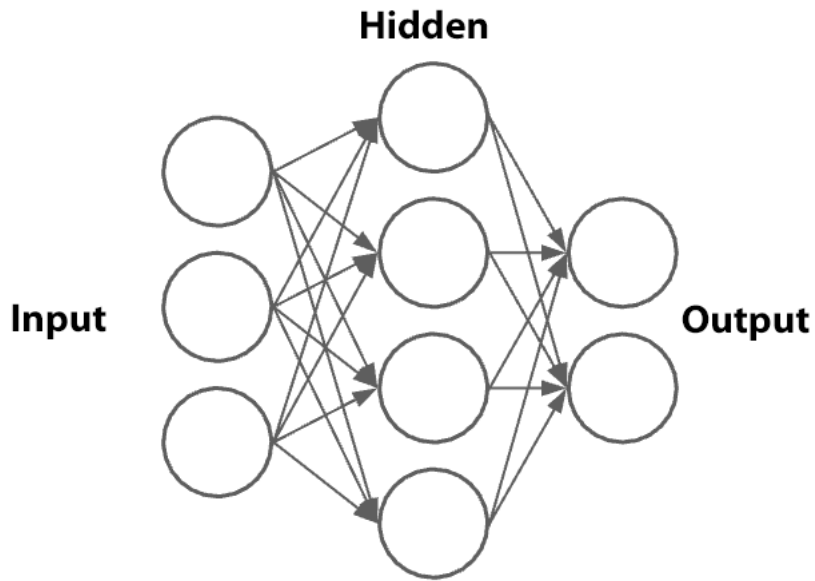


Figure 3.1: Example of a neural network consisting of an input layer of three neurons, a hidden layer, and an output layer of two neurons.

Neurons in artificial neural networks are similar to biological neurons, with input connections through which they receive stimuli, perform internal computations, and generate an output value.

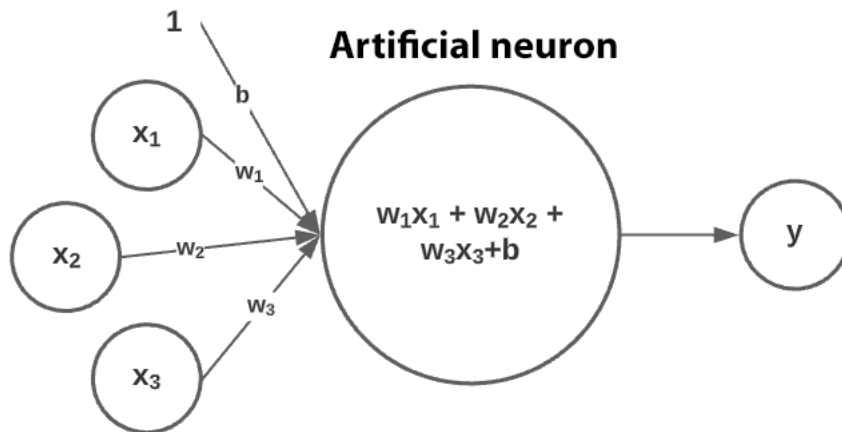


Figure 3.2: Example of a neuron composed of three input elements x and their respective weights w .

The artificial neuron works as a function, which performs the weighted sum of the input values by a value called weight. By multiplying them, it generates a value that indicates the intensity with which that input value affects the neuron. The output of the artificial neural network is given by the following equation.

$$\mathbf{z} = \sum_i^n x_i w_i + b \quad (3.1)$$

The output signal of an artificial neuron is given by the following equation, where y is the output signal, f is the activation function of the neuron, w_i are the weights of the neuron's links, x_i are the input signals of the neuron, and b is the bias term, which gives us greater control over the function. Essentially, it is another connection to the neuron where the input variable has a value of 1 and can be controlled by the assigned weight.

The activation function of a neuron is used to determine the neuron's output signal based on its input signal and the weights of its links. The most commonly used activation functions in artificial neural networks are the sigmoid function, the hyperbolic tangent function, and the ReLU function.

The sigmoid function is a non-linear function that takes an input value and transforms it into an output value in the range of 0 to 1. This function is used in artificial neural networks to model binary classification problems, such as the classification of images into two categories. The sigmoid function is mathematically represented as:

$$f(x) = \frac{1}{1 + e^{-x}}$$

The hyperbolic tangent function is a nonlinear function that takes an input value and transforms it into an output value in the range of -1 to 1. This function is used in artificial neural networks to model multiclass classification problems, such as image classification into multiple categories. The hyperbolic tangent function is mathematically represented as:

$$f(x) = \tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

The ReLU function is a non-linear function that takes an input value and transforms it into an output value in the range of 0 to ∞ . This function is used in artificial neural networks to model regression problems, such as

predicting a numerical value from a set of features. The ReLU function is mathematically represented as:

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

The purpose of an artificial neural network (ANN) is to find the weights values in such a way that it minimizes the cost or error function, which reflects if the model is approaching the desired result.

The cost functions used in an artificial neural network can be mathematically represented by the following equations:

Cross-entropy loss:

$$J = -\frac{1}{N} \sum_{i=1}^N [y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i)]$$

Where J is the cost function, N is the number of examples in the training dataset, y_i is the expected output for example i , and \hat{y}_i is the output obtained by the model for example i .

Mean Squared Error (MSE):

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

Where J is the cost function, N is the number of examples in the training dataset, y_i is the expected output for example i , and \hat{y}_i is the output obtained by the model for example i .

The binary distortion function can be mathematically represented as:

$$J = \frac{1}{N} \sum_{i=1}^N (y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i))$$

Where J is the cost or loss function, N is the number of examples in the training data set, y_i is the expected output for example i , and \hat{y}_i is the output generated by the model for example i .

KL divergence function:

$$J = \frac{1}{N} \sum_{i=1}^N y_i \log\left(\frac{y_i}{\hat{y}_i}\right) - y_i + \hat{y}_i$$

Where J is the cost function, N is the number of examples in the training dataset, y_i is the expected output for example i , and \hat{y}_i is the output obtained by the model for example i .

The combination of multiple neurons in layers and the creation of multiple layers makes it possible to create more complex artificial neural network models. The number of layers determines the depth of the model, giving rise to the name deep learning.

3.2 Generative Adversarial Networks

Within adversarial neural networks, a model proposed in the work of Ian Goodfellow *et al.* (10) emerged in 2014, presenting a revolutionary idea called generative adversarial network (GAN).

A generative adversarial network (GAN) is a type of machine learning model used to generate high-quality synthetic content. A GAN consists of two components: a generator network and a discriminator network. The generator network is responsible for generating synthetic content, while the discriminator network is responsible for evaluating the quality of the content generated by the generator network.

The generator network is an artificial neural network that receives a random vector as input and returns a synthetic image as output. It consists of several layers of nodes or neurons connected by links or weights and uses nonlinear activation functions to process input signals and generate output signals. The generator network is trained using a machine learning algorithm that allows it to improve its performance in the task of generating synthetic content.

The discriminator network is an artificial neural network that receives an image as input and returns a real value as output. It consists of several layers of nodes or neurons connected by links or weights and uses nonlinear activation functions to process input signals and generate output signals. The discriminator network is trained using a machine learning algorithm that allows it to improve its performance in the task of evaluating the quality of images.

The two networks are trained simultaneously and in an iterative competition. The generator network tries to generate as realistic images as possible to deceive the discriminator network, while the discriminator network tries to detect synthetic images generated by the generator network. This competition between the two networks allows both to improve their performance

in their respective tasks.

The global cost function of the generative adversarial network can be defined as the sum of the individual cost functions of the generator and discriminator networks and can be mathematically represented as:

$$V(D, G) = E_{x \sim p_{data}} [\log D(x)] + E_{z \sim p_z} [\log(1 - D(G(z)))]$$

Where p_{data} is the distribution of the real images of the training dataset, p_z is the distribution of the random vectors used as input to the generator network, D is the discriminator network, and G is the generator network.

The global cost function $V(D, G)$ can be interpreted as the sum of two terms: the first is the classification error of the discriminator network when presented with real images from the training dataset, and the second is the classification error of the discriminator network when presented with synthetic images generated by the generator network. The minimization is achieved through an optimization algorithm that allows the generative network to improve in its task of generating synthetic content, while the discriminative network improves in its task of evaluating the quality of the images.

In Fig. 3.3, the scheme of this network is shown, and it can be seen that the Generator never has access to the training database; it only relies on the data obtained through the Discriminator.

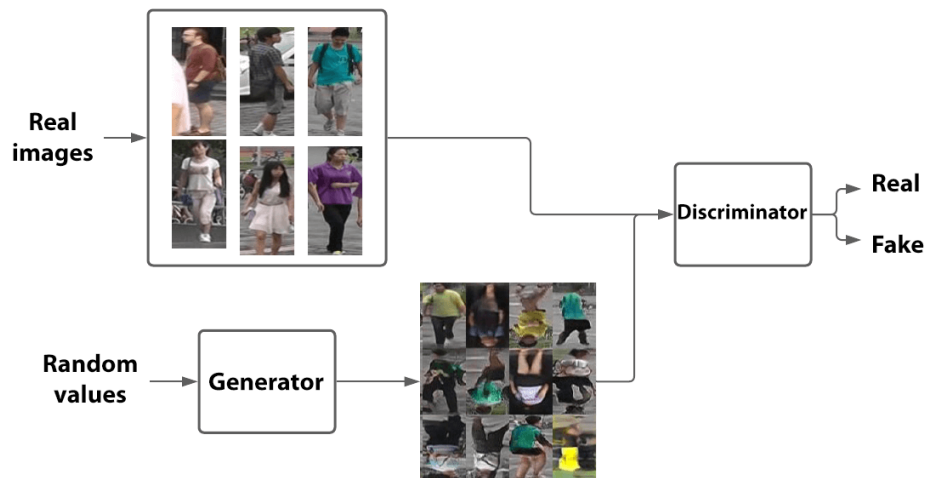


Figure 3.3: Generic GAN architecture.

This architecture is predominantly used for generating images, although it

can also be used for other types of data, such as creating audio or text, among others. Currently, it is also being used to improve the realistic graphics of some video games or the generation of real-time videos.

3.2.1 StyleGAN

StyleGAN is a generative adversarial network (GAN) architecture developed by NVIDIA in 2018 (54). This architecture has been trained to generate high-quality images of non-existent people's faces. In this case, it was trained with the FFHQ database, which consists of images of people's faces from the social network Flickr. It uses a generative network structure based on style layers that allow independent control of different aspects of the generated image, such as pose, facial expression, gender, etc.

The generative network consists of an encoder module and a generator module. The encoder module converts the input image into a style tensor, which is a fixed-dimension vector representing various aspects of the image. The style tensor is used as input for the generator module, which is a generative network that uses a structure of style layers to generate a synthetic image that approximates the input image. Once trained, the generative network can be used to generate high-quality synthetic images that approximate the images in the training dataset. Moreover, the structure of style layers allows independent control of different aspects of the generated images, such as pose, facial expression, gender, etc. This control capability in the generated images enables the use of StyleGAN in applications such as data augmentation for re-identification models. Once trained, the generator network can be used to create high-quality synthetic images that approximate the images from the training dataset. The Generator is a multidimensional latent space and is shown as a mathematical space in which the input and output vectors of an artificial neural network are represented. This space is called "*latent*" because it is not directly observed, but inferred from input and output observations. In this case, it is composed of 512 dimensions, and each of the positions corresponds to an image (see Fig.3.4). When a random numeric vector of size 512 is entered as input in the Generator, it generates an artificial face image. This means that each of the images that can be generated is composed of a 512-position latent vector. Fig.3.5 shows an example of the latent space of a two-dimensional Generator.



Figure 3.4: Graphical representation of a Generator. Multidimensional space where each position represents an image.

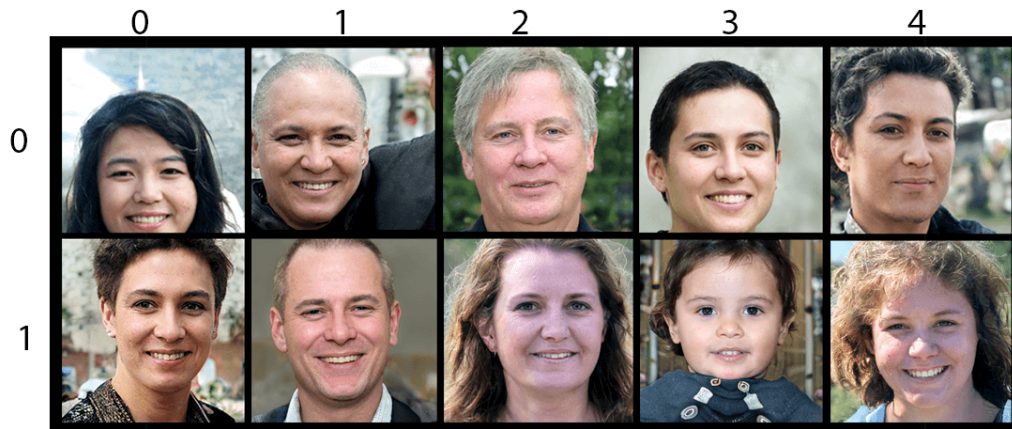


Figure 3.5: Example of a 2-dimensional Generator, where each coordinate belongs to an image. When inputting the latent vector $[0,0]$ as output, the upper left image is obtained.

Another significant property of StyleGAN was the ability to fix an issue called entanglement or *"disentangle"* that generative adversarial networks have. This occurs when the images generated by the generator network get entangled with each other, meaning that different aspects of the image, such as pose, gender, facial expression, etc., become mixed or confused. Imagine having the latent vectors of two face images, the first being a girl's face and the second an adult woman's face. If we interpolate between those two images in a model with the entanglement issue, we could find completely random images in between, whereas, with the StyleGAN model, the interpolation is coherent and smooth. Fig. 3.6 graphically exemplifies this problem.



Figure 3.6: The problem of entanglement. From image 1, an interpolation is made to image 2. a) interpolation without the problem of entanglement. b) interpolation with the problem of entanglement. As can be seen, interpolation a) is much smoother and more coherent.

Its architecture is based on progressive generative networks or *"progressive GANs,"* which is a model of adversarial generative networks that allows for the generation of higher resolution images. During the image generation process, it starts with very small images, for example, 4x4 pixels, and scales them up to the desired image size. This method is very effective in obtaining high-quality images. In the architecture image of StyleGAN, it can be seen how the image starts with a size of 4x4 pixels and gradually increases to, in this case, a size of 1024x1024.

Every time the size of the image is increased, it passes through an AdaIN layer, which stands for *"Adaptive Instance Normalization."*

AdaIN is based on the idea that the style of an image can be represented by the distribution of its visual features. Therefore, when transferring the

style of one image to another, the distribution of its features is modified to resemble the distribution of the source image.

Mathematically, this can be represented as a normalization and scaling operation of the visual features of the target image. Consider a target image x and a source image s . The AdaIN operation can be written as:

$$y = \frac{x - \mu_x}{\sigma_x} \odot \sigma_s + \mu_s$$

In this equation, μ_x and σ_x represent the mean and standard deviation of the visual features of the target image x , respectively. On the other hand, μ_s and σ_s represent the mean and standard deviation of the visual features of the source image s .

The \odot operation represents the element-wise product between two vectors and is used to apply the standard deviation of the source image to the normalized target image.

The AdaIN operation can be interpreted as a normalization of the target image, followed by a scaling of its visual features using the standard deviation of the source image. In this way, the target image is given the same style as the source image while preserving its original content.

During experiments, the StyleGAN team tried modifying the latent vector at each AdaIN layer, and as a result, the image showed different changes from the original. For example, starting from a latent vector of a person's face, if the latent vectors of the first AdaIN layers were modified, they noticed changes in what they called the "coarse" features of the image, the following represented the "middle" features and lastly the "fine" features. In Figure 3.73.83.9, several real examples can be seen by modifying the latent vector at different AdaIN layers.

1. Coarse features. These are where the structure of the image, in this case the face, is modified, generating completely different people.
2. Middle features. The structure of the image is the same as the original, except for modifications to the posture.
3. Fine features. Both the structure and posture are the same, but things like skin color, hair, eyes, etc. are modified.

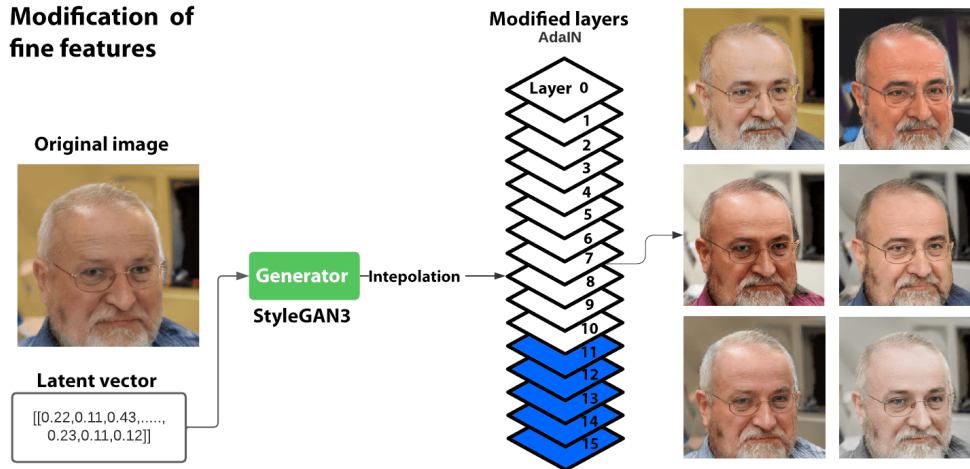


Figure 3.7: The latent vector is modified in the last layers of AdaIN. Changes can be observed in fine features such as hair color, eye color, and skin tone..

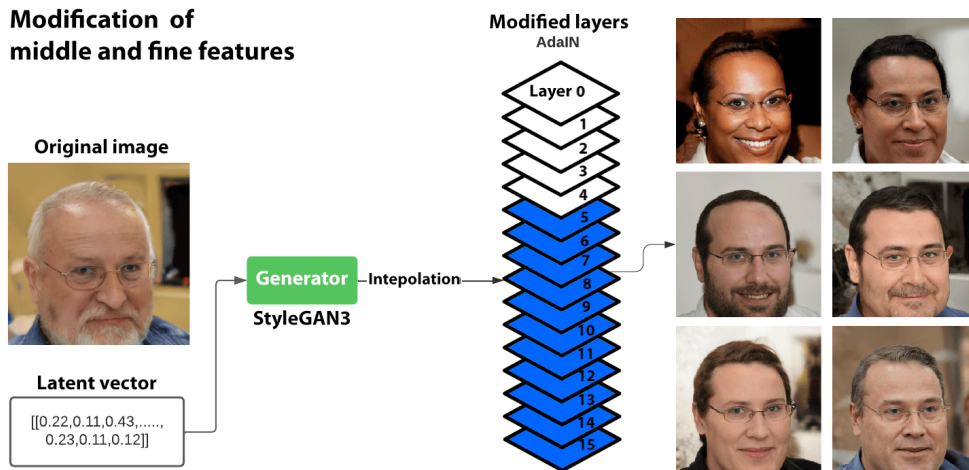


Figure 3.8: The latent vector is modified in the middle and last layers of AdaIN. Changes in both medium and fine features can be observed. The face structure is modified while the posture remains the same..

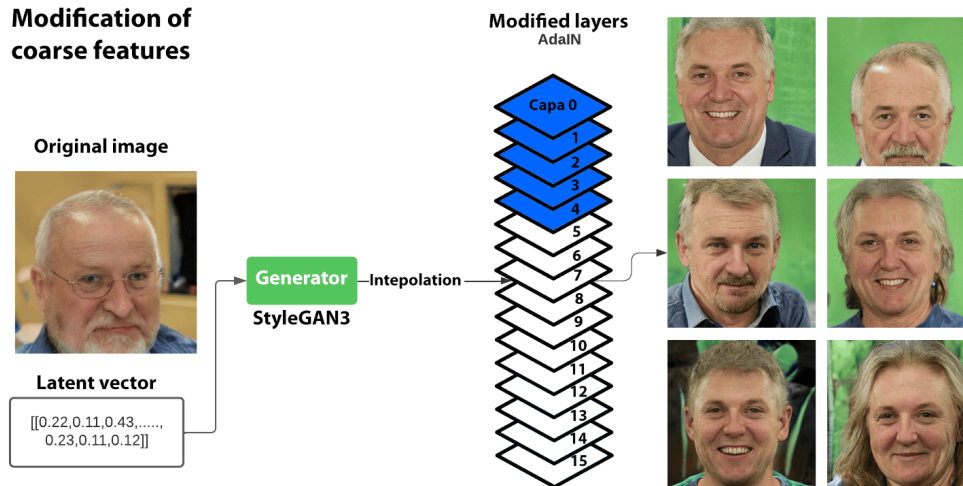


Figure 3.9: The latent vector is modified in the early layers of AdaIN. Changes in coarse characteristics, such as the image structure, can be observed, while the hair or skin color remains the same..

The flexibility of StyleGAN makes it a highly attractive architecture; however, it presents an important limitation. As can be seen in its architecture, images are always generated from a latent vector, while other generative adversarial network (GAN) architectures have the ability to create variations in pre-existing images. To achieve the goal of altering a real image with StyleGAN, encoders are used. An encoder is a component of a GAN that transforms an input image into a latent feature vector. This latent feature vector represents the input image in a latent space, which is used as input for the generative part of the GAN.

StyleGAN Encoder

Within the scientific literature, there are several StyleGAN encoders. Its operation is quite simple: a neural network takes an image as input and produces the latent vectors as output, which are then fed into the StyleGAN generator to generate an initial real image with the help of StyleGAN. Regardless of whether or not these images were used in the training of the generator, each input image will always find a corresponding image within

the generator's latent space (Fig. 3.11).



Figure 3.10: Left - real images. Right - obtained through the encoder.



Figure 3.11: Izquierda - imágenes reales. Derecha - obtenidas a través del codificador.

Generic explanation of how the training of a StyleGAN encoder works.

1. An image from the same database used to train the StyleGAN generator is introduced.
2. The model being trained generates its latent vectors.
3. The latent vectors are introduced into the pre-trained StyleGAN generator and an output image belonging to the latent space is obtained.
4. The input image and the output image are compared with a metric for comparing images, such as LPIPS, and the weights of the encoder are updated.
5. The pre-trained StyleGAN generator never modifies its weights.

Once the encoder is trained, it works as follows:

1. The real image is introduced as input into the encoder.
2. The encoder generates the corresponding latent vectors for the image.
3. The latent vectors are introduced into the StyleGAN generator to obtain the representation of the real image within the latent space.
4. Interpolations are made in the StyleGAN generator to obtain variations of the input image.

3.3 Re-identification model

In the literature, we can find solutions from as early as 1996, such as the work by Q. Cai *et al.* (55), which attempts to solve the re-identification problem. Currently, the most widely used method is the use of a neural network as a feature extractor.

The functioning of the re-identification model is straightforward.

1. A database with labeled images of people obtained from different security cameras is needed.
2. The model is supervisedly trained to classify a certain finite number of people from the training set. For example, using the Market 1501 database, it is trained to classify 751 people, and we can say that we have a model that can detect 751 people.

Once the model is trained and to use it with the test set, the following steps are followed:

1. Each of the images is input into the model as input, and each one obtains its vector of size 751, the number of people with which it was trained. Representing the proportion of each of the 751 people that the input person represents. In Fig. ?? it can be observed more easily.
2. We obtain the vector that represents the image of the person we want to search for and measure the distance with each of the other vectors that represent the other images of people. The use of cosine distance or Euclidean distance is recurrent in the literature, both are measures of similarity between two vectors in a vector space.
3. A distance-based classification of all the images where we are looking for that person is made, and the one with the shortest distance means that the image is more similar to the original, meaning that it is likely the same person.

There is no model that returns whether or not it is the same person. The classification model is used as a feature extractor, and based on these features, the distance between the vector of the image of the person being searched for and the rest of the images is calculated.

Chapter 4

Methodology

In this chapter, we provide a detailed description of the approach used to conduct the study. We explain the methods and techniques used for data collection and analysis, as well as the tools and platforms utilized. Additionally, technical details are provided on image generation and re-identification model training. The methodology proposed for the development of the study is divided into two sections. The first section focuses on training the generative adversarial network and generating artificial images. The second section, on the other hand, focuses on training and operating the re-identification model. The methodology can be summarized with the following points:

Generation of artificial images

- Training the StyleGAN3 generative adversarial network.
- Generating multiple artificial images of people in different postures.
- Using a real image of a person from the database as a base to generate artificial images of that same person in different postures.
- Filtering images by automatically removing generated images that contain noise or were generated incorrectly.

Re-identification model

- Designing the architecture and training the re-identification model.
- Running tests on the re-identification model.

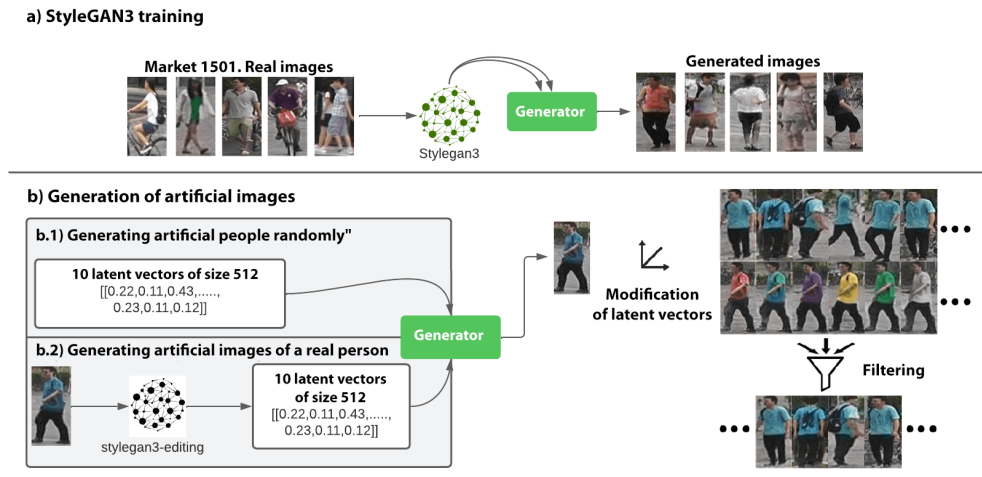


Figure 4.1: Architecture - Generation of artificial images.

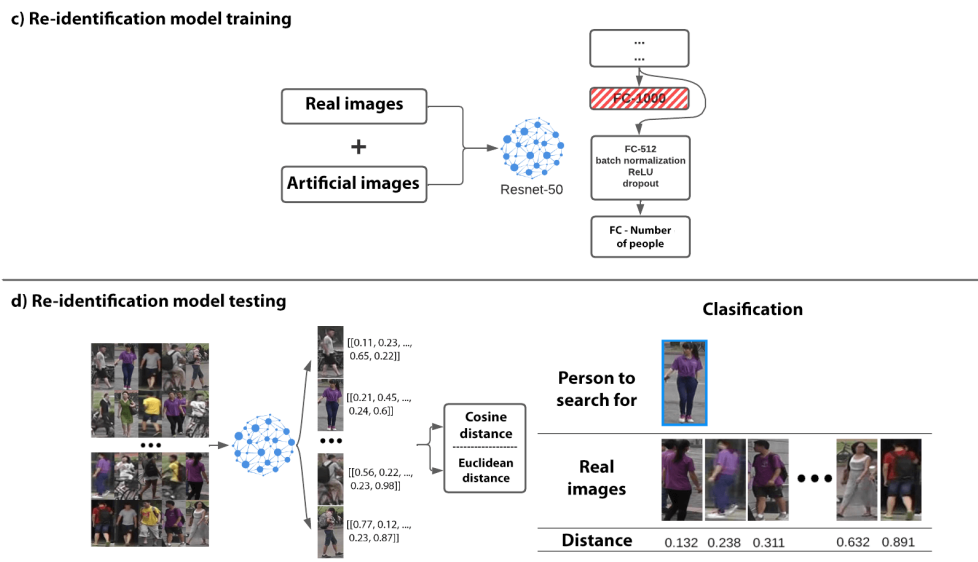


Figure 4.2: Architecture - Re-identification.

4.1 Generation of Artificial Images

To generate artificial images, we will use the StyleGAN3 generative adversarial network architecture (2). It is an image generative model developed by the Nvidia research team in 2021. It is an improved version of the StyleGAN2 model that is characterized by its ability to generate high-quality and realistic images in a wide range of content categories.

StyleGAN3 uses a deep learning approach based on generators and discriminators. The generator is a neural network that is trained to generate images that are as realistic as possible. To do this, a set of real images is shown to the generator, and it is asked to generate images that resemble them. As it is trained, the generator learns to extract relevant features from real images and use them to generate images that are as realistic as possible.

The discriminator is a neural network that is trained to distinguish between real and generated images. It is shown both real and generated images and asked to determine which ones are real and which ones are generated. As it is trained, the discriminator learns to identify the characteristics that differentiate real images from generated images, and it is used to guide the training of the generator toward generating more realistic images.

StyleGAN is pre-trained with 25 million faces images, of which 70,000 are high-quality 1024×1024 pixel real faces from the FFHQ database, and the rest were generated by the Discriminator, as shown in its detailed architecture in Fig.4.3. Currently, StyleGAN3 functions as a generator of high-quality faces, as shown in Fig.4.6. We will apply the transfer learning process to retrain StyleGAN3, as shown in Fig. 4.4.

Transfer learning is a technique in which a machine learning model that has been trained to perform a specific task is used as a starting point for training another model to perform a different task. Instead of training the new model from scratch, the knowledge and skills acquired by the original model are leveraged to initiate the training of the new model in a more advanced state. In this way, the time and effort required to train the new model is reduced and its performance is improved.

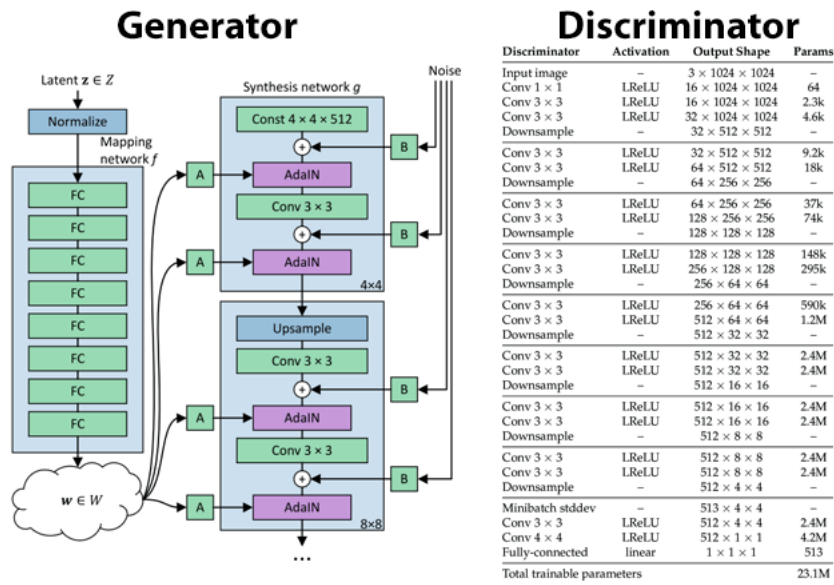


Figure 4.3: StyleGAN. Architecture of the Generator and Discriminator (8)

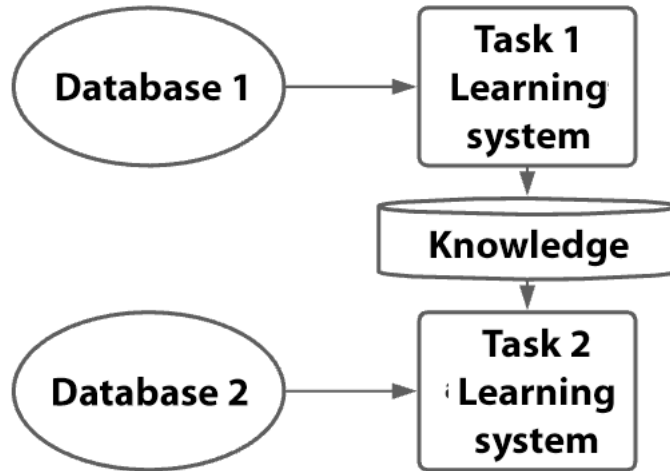


Figure 4.4: Diagram - Transfer Learning.



Figure 4.5: Artificially generated people using StyleGAN3. None of these people exist in reality..

For the retraining of StyleGAN3, the Market-1501 dataset was used, which consists of 51,247 images of 1,501 different people captured by six different cameras. To evaluate the performance of StyleGAN, the "Fréchet inception distance" (FID) metric was used, proposed by P. Dimitrakopoulos *et al.*, 2017 (56). This distance metric is used to measure the similarity between two distributions of images. The FID metric is based on the idea that the distance between two distributions of images is the same as the distance between the features of the images extracted from a deep neural network. To calculate the FID distance between two distributions of images, first the features of each distribution are extracted using a deep neural network, and then the distance between those features is calculated using the Fréchet distance. Mathematically, the FID distance between two distributions of images can be calculated as follows:

$$\text{FID} = |\mu - \mu_w|^2 + \text{tr}(\sigma + \sigma_w - 2(\sigma\sigma_w)^{1/2}) \quad (4.1)$$

It compares the mean and covariance of the real and fake images by obtaining the data from one of the deeper layers of the neural network, which is closer to the output data. It aims to mimic human perception to identify the

similarity between two images using the Discriminator as a feature extractor. If the obtained value is zero, it indicates that the generated and real data are identical, which means that the lower the obtained value, the greater the similarity between the generated images and the real images.

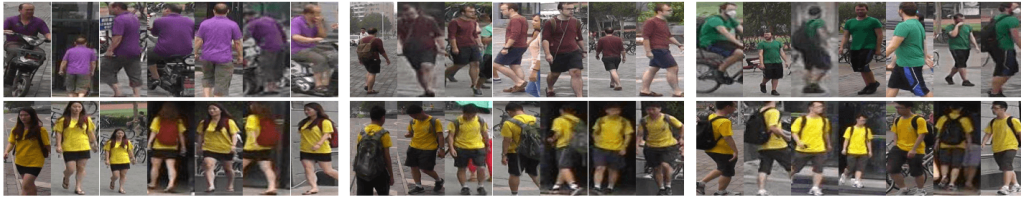


Figure 4.6: Images from the Market-1501 dataset.

Two approaches were implemented for generating images: the first is a completely random generation of artificial people, and the second is using a real person's image to generate variations of it, as shown in Fig. 4.1.

- Generation of artificial people images.

Using a random number, also known as a seed, the Generator assigns a latent vector that corresponds to an image. To obtain variations of the original image, another random latent vector can be used, through another seed or through interpolations. In the different AdaIN layers of the model, also known as style mixing, the latent vector is modified, obtaining different variations of the original image. It is possible to achieve everything from a total change in the structure of the image to more subtle changes, such as changes in tone, lighting, colors, saturation, among others. Another way to generate variations of the original image is by modifying the original latent vector through interpolations, as shown in Fig. 4.8.

Fig. 4.7 exemplifies how latent vectors are mixed to obtain variations of the original image.

- Generation of artificial images of real people.

As explained earlier, to do this, a model, an encoder, must be trained that is capable of obtaining the latent vector of a real image. In this case, the StyleGAN3-editing model (57) has been used and trained on the PsP encoder developed by Richardson *et al* (58). The encoder is part of a neural network that processes the input information and converts it into an internal representation that can be used by StyleGAN3 to generate the version of the real image within the latent space.

The LPIPS metric, Learned Perceptual Image Patch Similarity, is used for the loss function, which compares the real image with the one obtained in the generator. It is a measure of the distance or difference between two images in terms of their perceived similarity by a human observer. This metric is based on a pre-trained neural network called VGG-16, which is designed to recognize patterns in images. The idea behind LPIPS is that if two images have a small LPIPS distance, then they are perceived as similar by a human observer.

Mathematically, the LPIPS metric is calculated as follows:

First, the VGG-16 network is used to extract a representation of each of the two images in question. This representation is called a "feature map" and is a three-dimensional tensor that contains information about the visual features present in each image. We denote these feature maps as f_1 and f_2 .

Next, the Euclidean distance between the two feature maps is calculated. This distance is interpreted as the perceived similarity between the two images. The smaller this distance is, the more similar the images will be. The Euclidean distance between f_1 and f_2 is defined as:

$$\text{dist}(f_1, f_2) = |f_1 - f_2|_2$$

The process of training the encoder is shown in Fig. 4.9.

Once the latent vector of the real image is obtained within the latent space of StyleGAN3, new images are generated in the same way as in the previous case.

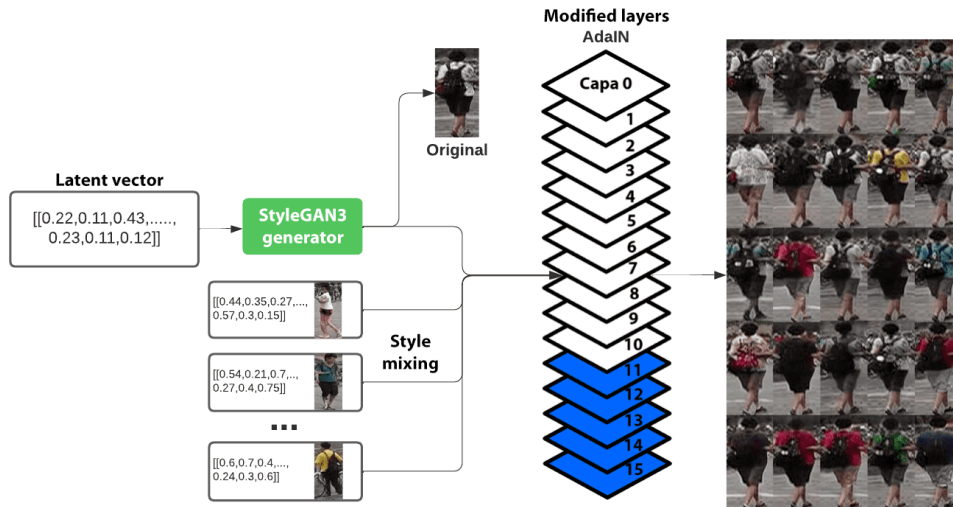


Figure 4.7: Starting from the initial latent vector, new images are generated. To mix styles in the blue layers, the latent vector from other images is introduced, while the white layers receive the latent vector from the original image.



Figure 4.8: Example of interpolation from an input image of a girl to her adulthood.

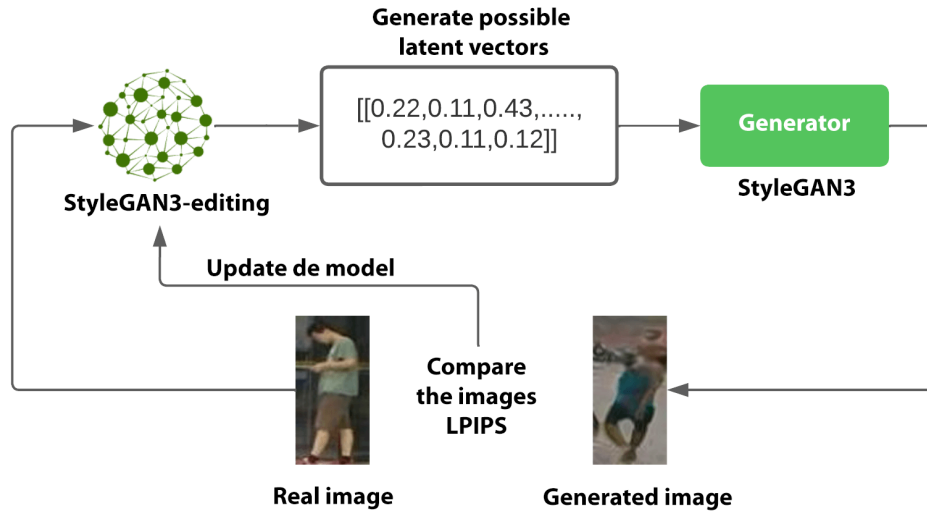


Figure 4.9: Training of the StyleGAN3 encoder.

In order to automate the filtering of the generated images, it is necessary to implement a filtering process. It is possible that some images may contain noise or distortions.

To measure and discard images generated by the generative adversarial network, metrics based on the quality of the generated data have been used. The first filter applied is a pedestrian detection model, and to measure the similarity of the generated images, the *structural similarity index measure* (SSIM) metric has been used.

- YOLOv4 Tiny Filtering

YOLOv4 Tiny, a work proposed by Z. Jiang *et al* (59), is a reduced version of the YOLOv4 model, designed to detect objects in images and videos. YOLO (You Only Look Once) is an object detection technique that stands out for its speed and accuracy. The “*tiny*” version of YOLOv4 is particularly useful for low-power devices, as it is less demanding in terms of resources and can be efficiently executed on mobile devices and low-performance computers. In general terms, YOLOv4 Tiny uses a convolutional neural network to extract features from an image and then employs a combination of machine learning techniques

to perform object detection. The Tiny version of YOLOv4 has been optimized to detect pedestrians with a precision and speed comparable to larger models, but with lower resource requirements. This makes it an excellent option for real-time applications on devices with limited capabilities.

- SSIM Filtering

The Structural SIMilarity (SSIM) metric is a measure of structural similarity between two images. The SSIM metric is often used to evaluate the quality of a processed image compared to an original image, and is calculated by comparing the structural features of both images. SSIM is based on the fact that human perception of image quality is based on its structural content, and not just the pixel difference between two images. Therefore, SSIM is used to measure the structural similarity between two images and give a score that reflects the perceived quality by a human observer.

To calculate the SSIM metric, three structural features of two images are compared: their mean intensity, intensity variance, and intensity covariance. SSIM is obtained from the product of these three features, and two images are considered to have high SSIM if they have similar mean intensity, intensity variance, and intensity covariance. The result is obtained from the product of these three features and is denoted as:

$$SSIM(x, y) = l(x, y) \cdot c(x, y) \cdot s(x, y) \quad (4.2)$$

Where x and y are the two images being compared, $l(x, y)$ is the similarity in mean intensity, $c(x, y)$ is the similarity in intensity covariance, and $s(x, y)$ is the similarity in intensity variance.

The similarity in mean intensity is calculated as:

$$l(x, y) = \frac{2 \cdot \mu_x \cdot \mu_y + C_1}{\mu_x^2 + \mu_y^2 + C_1} \quad (4.3)$$

The formula reads as follows: "The luminance l between two images x and y is equal to the quotient of the product of twice the mean μ of image x and the mean μ of image y , plus the constant value C_1 ,

divided by the square of the mean μ of image x plus the square of the mean μ of image y plus the constant value $C1$."

Here, μx and μy are the mean intensities of images x and y , respectively, and $C1$ is a constant used to avoid division by zero.

The similarity in intensity covariance is calculated as follows:

$$c(x, y) = \frac{2 \cdot \sigma xy + C_2}{\sigma x^2 + \sigma y^2 + C_2} \quad (4.4)$$

The formula reads as follows: "The contrast c between two images x and y is equal to the quotient of the product of two times the correlated standard deviation σ of images x and y plus the constant value $C2$, and the sum of the square of the standard deviation σ of image x and the square of the standard deviation σ of image y plus the constant value $C2$."

Here, σxy is the intensity covariance between images x and y , σx and σy are the intensity variances of images x and y , respectively, and $C2$ is a constant used to avoid division by zero.

The similarity in intensity variance is calculated as:

$$s(x, y) = \frac{2 \cdot \sigma x \cdot \sigma y + C_3}{\sigma x^2 + \sigma y^2 + C_3} \quad (4.5)$$

The formula reads as follows: "The edge similarity s between two images x and y is equal to the quotient of the product of two times the standard deviation σ of image x and the standard deviation σ of image y , plus the constant value $C3$, divided by the square of the standard deviation σ of image x plus the square of the standard deviation σ of image y , plus the constant value $C3$ ".

Here, σ_{xy} is the intensity covariance between images x and y , σ_x and σ_y are the intensity variances of images x and y , respectively, and $C3$ is a constant used to avoid division by zero.

In summary, the SSIM metric is calculated by comparing the mean intensities, intensity covariances, and intensity variances of two images. The SSIM is obtained as the product of the similarity in each of these features and is used to evaluate the quality of a processed image compared to an original image.

The SSIM metric is used to evaluate the quality of a processed image compared to an original image, and it is based on the comparison of the structural characteristics of both images. The higher the result, the greater the variation in the generated images.

$$\text{SSIM}(x, y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)} \quad (4.6)$$

4.2 Re-identification model

A re-identification model is an algorithm used in image processing and artificial intelligence to identify and track objects or people in a sequence of images. These models are based on comparing visual features across different images to determine whether they depict the same object or person. Mathematically, a re-identification model uses a similarity function to calculate the similarity between two images. This function takes two vectors of visual features (one from the reference image and one from the image being compared) and returns a value indicating the similarity between the two images. If the value returned by the similarity function exceeds a certain threshold, it is determined that the images correspond to the same object or person. To calculate the vectors of visual features, the model uses a neural network that has been previously trained on a dataset of labeled images. The neural network extracts relevant features from the images and groups them into a feature vector. These vectors are then used in the similarity function to determine the similarity between images.

The architecture proposed in this work is shown in Fig. 4.2. It uses a convolutional neural network Resnet50, in which the last layer is modified to adapt the output to the number of people with whom the model will be trained. During training, cross-entropy loss is employed as the loss function. This function is defined as:

$$L(y, \hat{y}) = -\frac{1}{N} \sum_{i=1}^N y_i \log \hat{y}_i$$

In this equation, the following terms are used:

- y : represents the real label or desired value of the output.

- \hat{y} : represents the output predicted by the model.
- N : is the number of examples in the dataset.

Cross-entropy loss is a commonly used loss function in classification problems, where the output of the model is interpreted as the probability that an example belongs to each class. The idea behind cross-entropy is that if the model's output is a good approximation of the true probability distribution, then the cross-entropy loss function will have a low value. Conversely, if the model's output is very different from the true probability distribution, then the cross-entropy loss function will have a high value. It is used to measure how well the model is making predictions about the real probability distribution of the classes. During model training, the loss function is optimized to improve its accuracy in predictions. Once training is complete, the model operates as a feature extractor, and image classification is performed. In this work, a Resnet50 convolutional neural network was used, in which the last layer was modified to output based on the number of people with whom the model was trained. During training, cross-entropy loss was used to optimize the model's accuracy.

Each image to be evaluated is introduced into the model one by one to obtain its corresponding feature vectors. To classify which images are of the same person, each of the image vectors is compared to the vector of the original image using cosine distance. It is a measure of similarity between two vectors in a vector space. This measure is calculated using the cosine of the angle between the two vectors and can be interpreted as the projection of the shorter vector onto the longer vector.

Mathematically, the cosine distance between two vectors \mathbf{a} and \mathbf{b} can be calculated as follows:

$$d_c(\mathbf{a}, \mathbf{b}) = 1 - \frac{\mathbf{a} \cdot \mathbf{b}}{|\mathbf{a}||\mathbf{b}|}$$

In this equation, $\mathbf{a} \cdot \mathbf{b}$ is the dot product of vectors \mathbf{a} and \mathbf{b} , and $|\mathbf{a}|$ and $|\mathbf{b}|$ are the norms of vectors \mathbf{a} and \mathbf{b} , respectively.

The cosine distance has a value between 0 and 1, where a value closer to 1 indicates greater similarity between vectors \mathbf{a} and \mathbf{b} , and a value closer to 0 indicates lower similarity between them. After obtaining the cosine distance of all images, they are sorted, and the ones with a smaller cosine distance are those that most closely match the original image, that is, those that have been detected as images of the same person.

In the classification section of Fig. 4.2, it is shown how images are classified using cosine distance.

Chapter 5

Experimental Results

This chapter presents the results obtained during the experimentation. Due to the complexity of the architecture, it has been divided into two sections.

- Image Generation

Results of the training, generation of artificial images, and filtering.

- Tests with the Re-identification Model

The training has been divided into two parts. First, the re-identification algorithm was trained with different numbers of artificial people, and secondly, with different numbers of artificial images of real people from the dataset.

5.1 Image Generation

The StyleGAN3 generative adversarial network (2) was used for the generation of artificial images. The model is pre-trained with 25 million face images, of which 70,000 are real high-quality, 512x512-pixel resolution images from the Flickr-Faces-HQ dataset (FFHQ), and the rest were generated by the discriminator.

In Table 5.3, the characteristics of the StyleGAN3 training are presented, which consisted of transfer learning and subsequent retraining with 51,247 images from the Market-1501 database. Table 5.1 details the hyperparameters used. To measure the performance of the training, the Fréchet Inception Distance (60) (FID) metric was employed, which was applied to both the generated and real images. The closer the value of both, the better the image

cfg	gpus	batch	gamma	king	snap	metrics
stylegan3-r	1	16	2	5000	20	fid50k_full

Table 5.1: Hyperparameters used during the training of StyleGAN3. The parameter “**cfg - stylegan3-r**” determines the type of training, “**config R**” or rotation equivalent, which makes small modifications to the network to allow for rotation and translation of the generated images. This ensures that the FID metric is not adversely affected if the generated images are rotated or moved. The training was done on a GPU, with “**batch - 16**” images introduced into the network at each iteration. The regularization weight “**gamma - 2**” indicates how quickly the weights are updated. The total duration of the training was “**king - 5000**” images, with the model being saved every “**snap- 20**” times (in this case, every 80,000 images). The “**metrics - fid50k_full**” metric was used to measure the performance of the model during training.

Modelo	Img. entrenamiento	Img. pruebas	Entrenamiento	Hardware
StyleGAN3	51247	No aplica	2d 08h 24m	Titan RTX

Table 5.2: Technical details of the StyleGAN3 training for the generation of artificial images, including the model, the number of images used for training, the duration of the training, and the graphics card used.

generation. As shown in Table 5.4, the performance of StyleGAN3 is significantly superior to other generative adversarial networks trained on the same database. As an example of the generated images, Figure 5.2 showcases the capacity and quality of the StyleGAN3 model for generating artificial images compared to real ones.

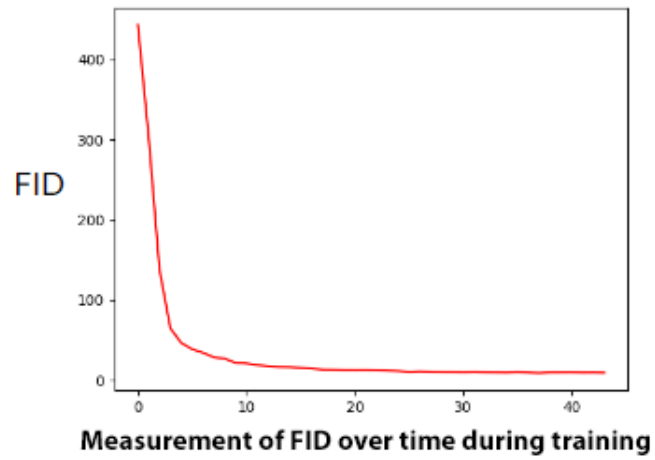


Figure 5.1: Evolution of the model's performance through the FID metric at different epochs during the StyleGAN3 training.

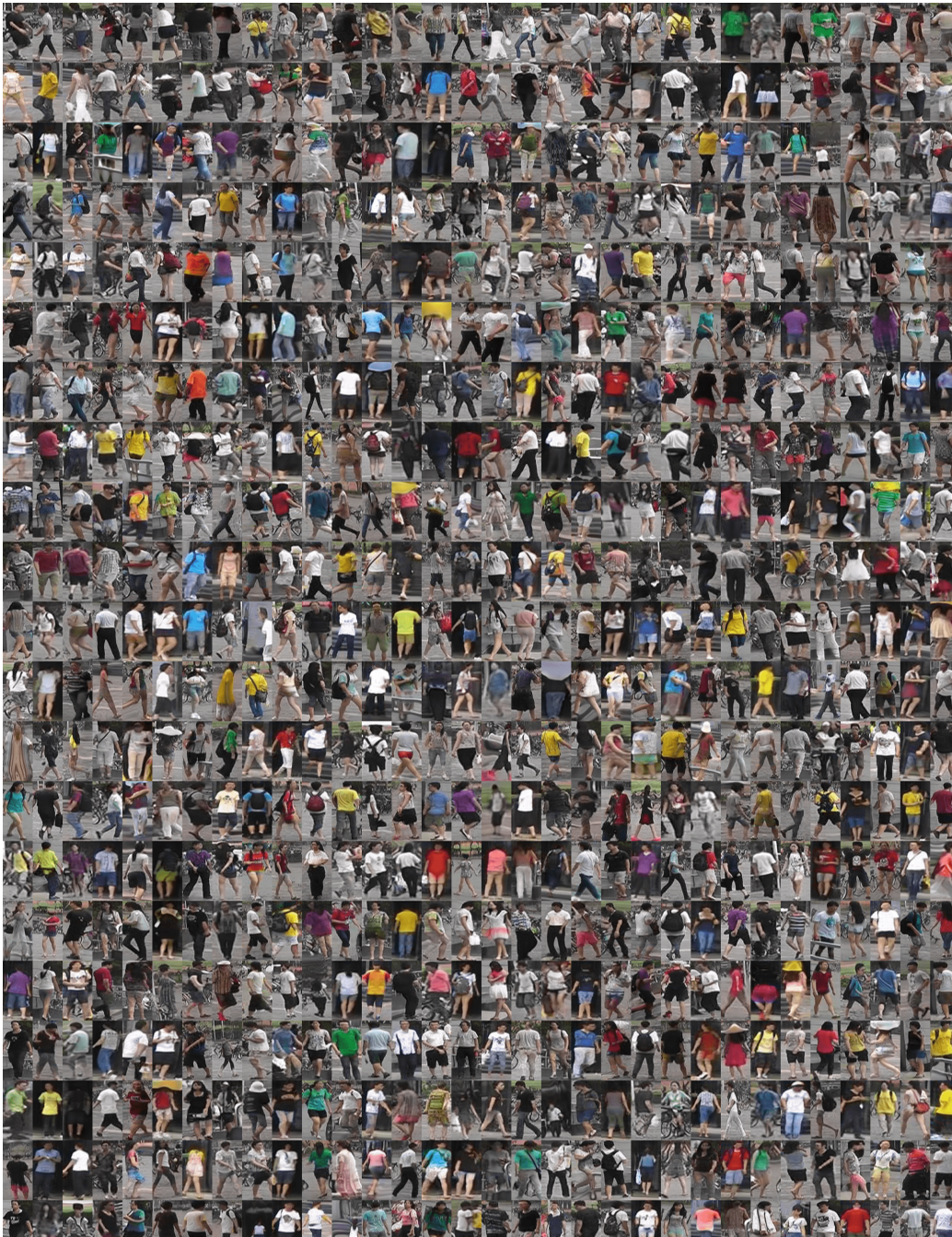


Figure 5.2: Artificial images generated randomly after training.

To generate variations of images of the same person, style mixing was used.

In the case of the model trained with Market-1501.



Features	AdaIN layers	samples
Fine	(12,13,14,15)	
Medium	(5,6,7,8,9,10,11)	
Coarse	(0,1,2,3,4,5)	

Table 5.3: Layers used to generate new images of the same person based on modification of their fine, middle, or coarse features.

Once the artificial images were generated, two filters were applied to discard images that may have been generated incorrectly or contain noise.

- YoloV4 Tiny filtering

YoloV4 tiny trained model (59) was used for pedestrian detection in the generated images. All images whose classification was below the threshold of 0.6 were discarded, which was determined by analyzing Fig. 5.3, which shows the different percentages of images classified as non-pedestrians using different threshold values on the real images from the Market-1501 database. Analyzing these data, it can be observed that when a threshold of 0.6 is used, the percentage of incorrectly classified images as non-pedestrians is only 6.45%, which is considered a conservative value for use in filtering artificially generated images.

- SSIM filtering

The structural similarity index (SSIM) (9) was used to evaluate the similarity between two images. The methodology used to apply this metric consisted of selecting an image of a person and comparing it with the rest of the images of that same person in different poses. If the similarity value was equal to one, it was considered to be the same image. This metric was applied to the real images of the Market-1501 database, and a histogram was obtained (see Fig. 5.4). Through this histogram, it was determined that images whose SSIM value was less than 0.75 would be discarded.

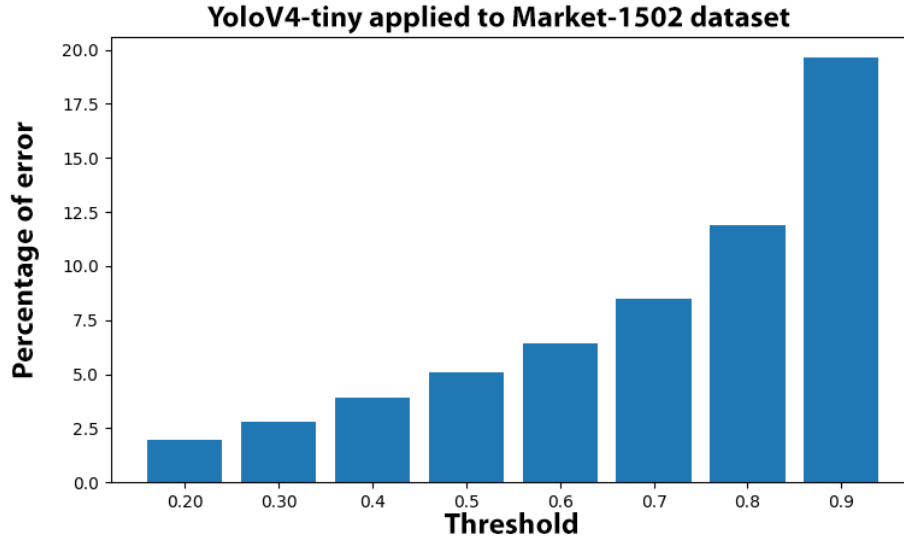


Figure 5.3: Application of YoloV4 tiny on images from the Market-1501 database using different thresholds. The error percentage represents the images that were not classified as pedestrians.

Método	Market-1501 FID	Referencia
Real	7.22	Hao Chen <i>et al.</i> (42)
IS-GAN	281.63	Hao Chen <i>et al.</i> (42)
FD-GAN	257.00	Saleh Hussin <i>et al.</i> (48)
PG-GAN	151.16	Zhedong Zheng <i>et al.</i> (7)
DCGAN	136.26	Saleh Hussin <i>et al.</i> (48)
LSGAN	136.26	Zhedong Zheng <i>et al.</i> (7)
PN-GAN	54.23	Zhedong Zheng <i>et al.</i> (7)
GCL	53.07	Hao Chen <i>et al.</i> (42)
DG-Net	18.24	Hao Chen <i>et al.</i> (42)
DG-GAN	18.24	Saleh Hussin <i>et al.</i> (48)
StyleGAN3	9.29	

Table 5.4: Table comparing different generative adversarial networks (GANs) using Fréchet Inception Distance (FID)(60) as the performance metric. All models were trained on the Market-1501 dataset(1). The first row represents the FID value obtained by applying the metric to real images from the dataset.

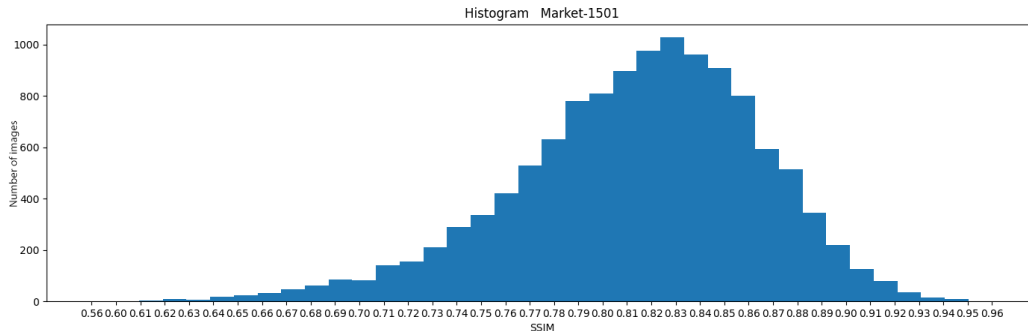


Figure 5.4: Histogram of the SSIM (9) metric on the Market-1501 dataset. Number of images that obtained the same SSIM value. One image per person was selected and compared to the rest of the images of that same person. As can be seen, most of the images are around the threshold of 0.75 and above.



Figure 5.5: A) Real images from the Market-1501 dataset. B) Artificial images.

a) Generation of artificial persons

During the experimentation, images of 401 artificial persons were generated in a completely random manner, and by modifying their latent vectors, 51 images per person were generated in different poses, resulting in a total of 20,451 images (see Fig. 5.6).

The Yolo V4 filter was applied to the generated images for pedestrian detection, eliminating 3,419 images that represent 16.7% of the total. Different examples of filtered images are shown in Fig. 5.7.

Then the SSIM filter was applied, and 386 images, or 2.3% of the total, were discarded. Some examples of images discarded by this method can be seen in Fig. 5.8.

After applying the filters, a total of 3,815 images were discarded (see Table 5.5).

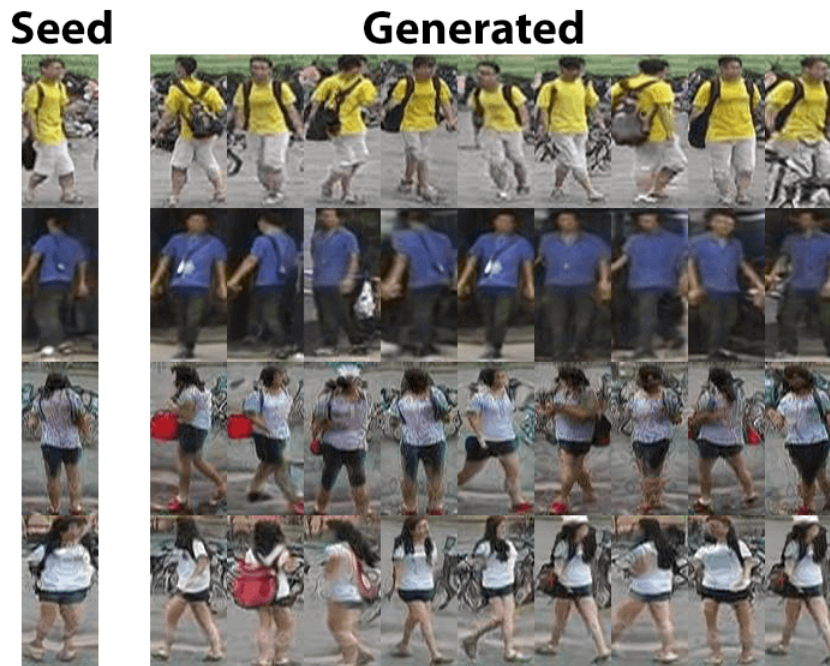


Figure 5.6: Seed - It is the image generated randomly to which its latent vectors will be modified to change its mean characteristics. Generated - They are the images that have been generated by modifying the latent vectors of the seed image.

Method	Images Discarded	%
Yolov4-tiny pedestrian detection (59)	3.419	16.7
SSIM (9)	396	2.3
TOTAL	3.815	18.6

Table 5.5: Number of discarded images during the application of different filters.



Figure 5.7: Example of some discarded images using the Yolo V4 tiny model for pedestrian detection.

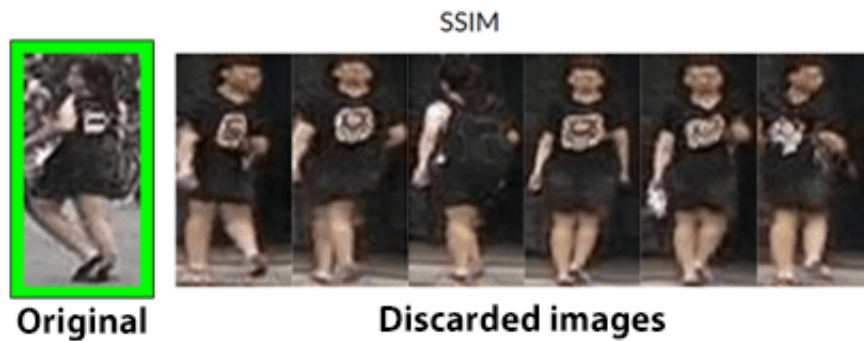


Figure 5.8: Example of some images discarded using the SSIM (9) metric. Starting from one of the images of a person (original image), it is compared with the rest of the generated images of that same person.

b) Real people images generation

In the context of generating images of real people, an encoder was used that has been retrained with the pre-trained StyleGAN3 model on the Market-1501 database. This encoder is the same one used in the article presented by Yuval Alaluf *et al.* (57), which has proven to be highly effective in generating images with visual quality very similar to those of real human faces. The objective of using this encoder is to be able to encode images of real people and find their corresponding latent vectors within the StyleGAN3 generator, which will allow for generating variations of the original image through manipulations in the latent vectors.

To generate images of a real person, the encoder needed to be trained for 240,000 epochs (see Table 5.6).

Model	Training Images	Validation Images	Training Method	Epochs	Hardware
stylegan3-editing	39466	732	3d 03h 16m	240,000	Titan RTX

Table 5.6: Technical data of the training of the model for artificial image generation.

To evaluate the performance of the model during training, three different loss functions were used, which measure different aspects of the quality of the generated images. The first one is the Perceptual Similarity Metric (61) (LPIPS), which measures the perceptual similarity between two images. The second is L2 (62), which measures the Euclidean difference between two images. Finally, the Momentum Contrast (63) (MOCO) loss function was used, which takes into account the correlation between the features of different images.

In Table 5.7, the fluctuation of the three loss functions can be observed in different epochs of the training. It is important to note that the performance of the model can vary depending on the complexity of the input data, and that the architecture of StyleGAN3 is originally designed to work with simpler datasets, such as faces. That is why the performance of the model may be affected when using more complex images, such as the full body of a person in different poses.

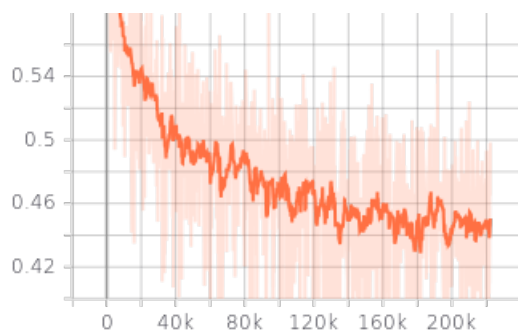


Figure 5.9: LPIPS

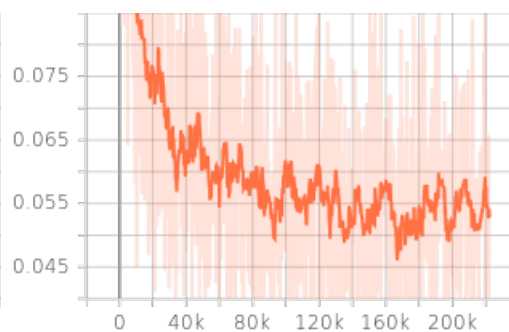


Figure 5.10: L2

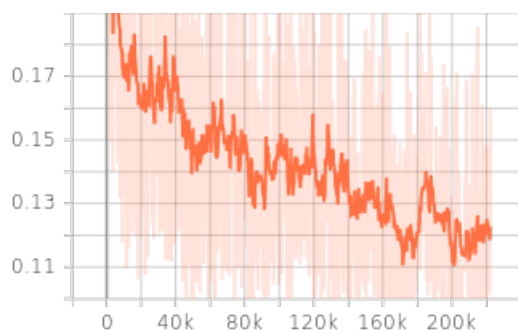


Figure 5.11: MOCO

Table 5.7: Evolution of the loss functions during training until epoch 220,000.

Next, it can be observed in Table 5.8 that the learning is smoother and it can be seen that the LPIPS metric is the one that improves more steadily.

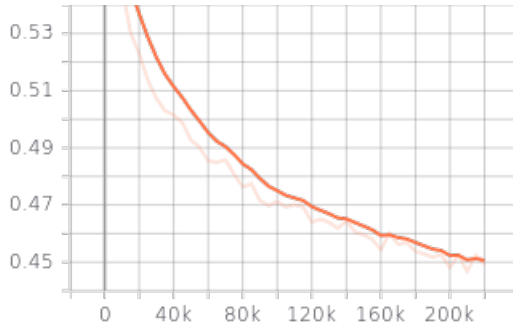


Figure 5.12: LPIPS

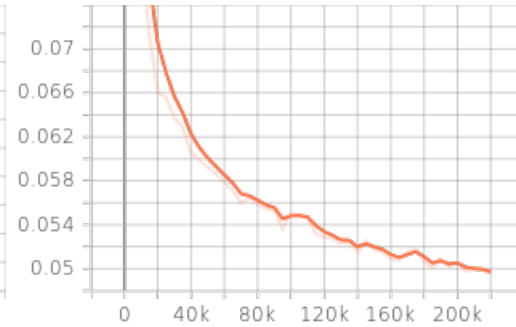


Figure 5.13: L2

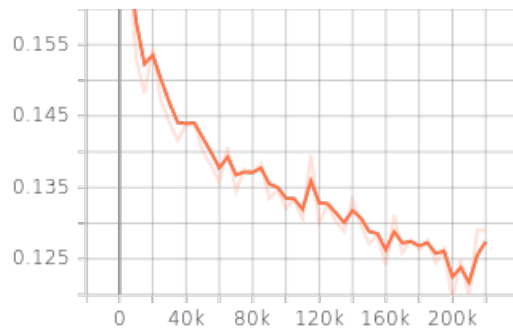


Figure 5.14: MOCO

Table 5.8: Table 5.8 shows the set of loss functions during validation at different epochs, up to epoch 220,000.

Using the model generated at epoch 220,000, the latent vectors representing the input images were obtained. As can be seen in Fig. 5.16, the obtained images resemble the originals, although they do not reach the quality of the images generated in the previous point.

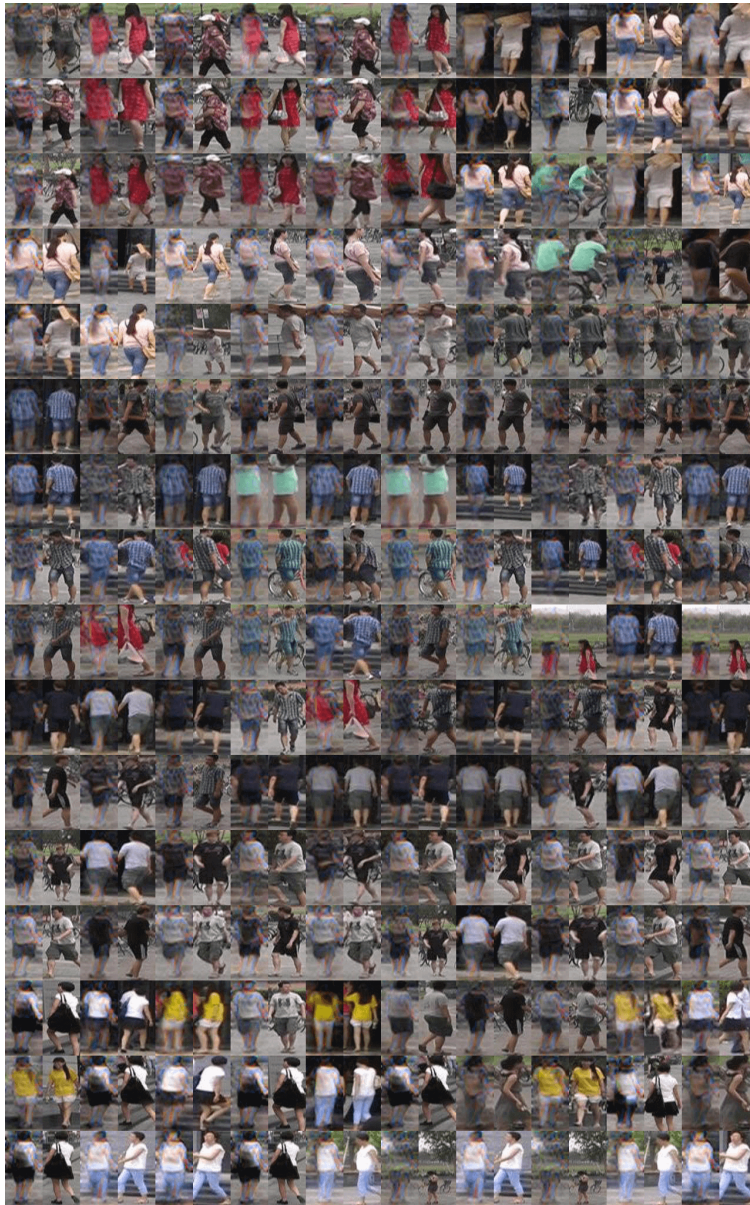


Figure 5.15: Pairs of images. The real image is on the right, and its counterpart obtained through the encoder in the latent space of StyleGAN3 is on the left. It can be observed that the model performs better when showing the full body, although it does not reach the quality of randomly generated images.

Method	Discarded images	%
Yolov4-tiny pedestrian detection (59)	12,129	16.0
SSIM (9)	1,108	1.46
TOTAL	13,237	17.46

Table 5.9: Number of images discarded during the application of different filters.

After generating the latent vector, variations of that person in different postures were generated by modifying the latent vectors. A total of 75,788 images of 751 different people were generated. After applying different filters, 63,659 images were left, as shown in Table 5.9.

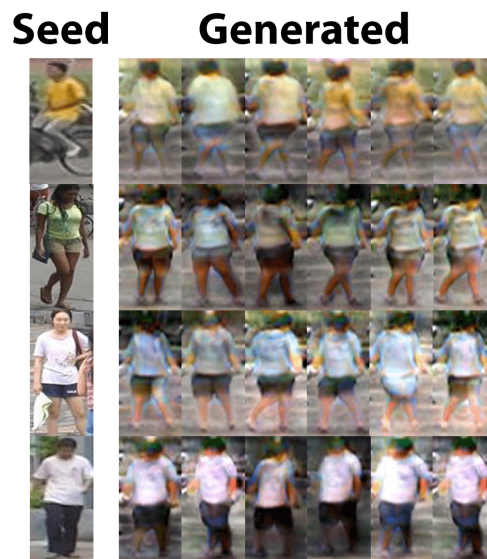


Figure 5.16: Seed - real image. Generated - are the images generated by modifying the latent vectors of the seed image.

5.2 Re-identification

To train the model, a batch size of 16 images was used with the hyperparameter `-batchsize`. Only one GPU was utilized, and the model was trained for 60 epochs in all experiments. Two types of experiments were conducted during the study based on the type of generated artificial images: firstly, analyzing the model's performance when adding artificial person images and their variations in different poses, and secondly, using artificial images generated from real person images.

a) Using 320 artificially generated people.

During the experimentation, the model was tested with a different number of added persons, adding ten persons at a time until reaching three hundred and twenty. In Fig. 5.21, it can be seen that the performance of the base re-identification model remains stable or slightly decreases and then starts to increase. This may be due to the fact that increasing the number of persons also increases the number of classes, and some classes may not be as relevant as others due to different numbers of images. Adding more persons only generates small noise. The performance improved by 1% when adding 280 persons during training.

During training, it can be observed that the model stabilizes from epoch 40 onwards in all experiments, and there is no significant change in performance based on the number of added persons (see Figs. 5.23, 5.24, 5.25, 5.20)

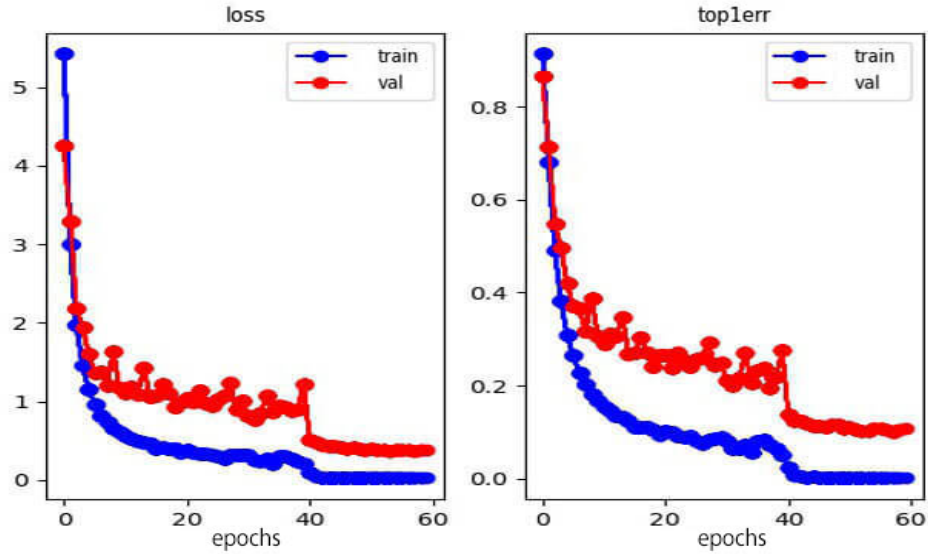


Figure 5.17: In the **left** plot, we can see the loss function during the training and validation phases of the base model without any additional generated people. In the **right** plot, we can see the Rank1 error percentage during the training and validation phases.

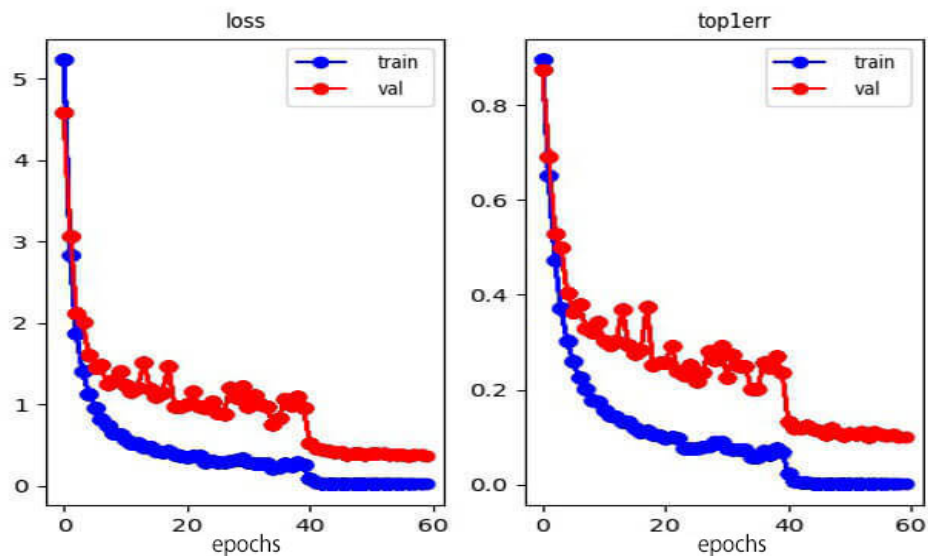


Figure 5.18: Adding 100 persons. **Left**, loss function during training and validation. **Right**, Rank1 error percentage during training and validation.

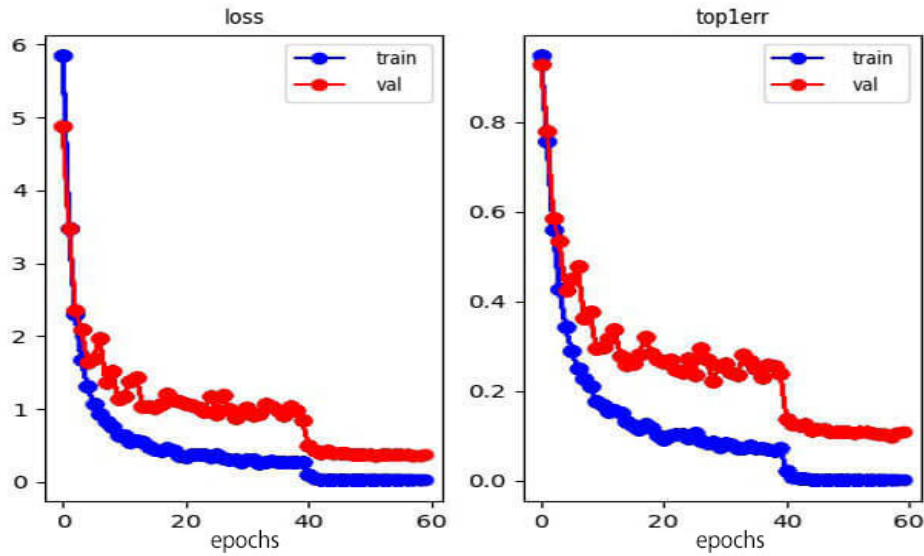


Figure 5.19: Adding 200 persons. **Left**, loss function during training and validation. **Right**, Rank1 error percentage during training and validation.

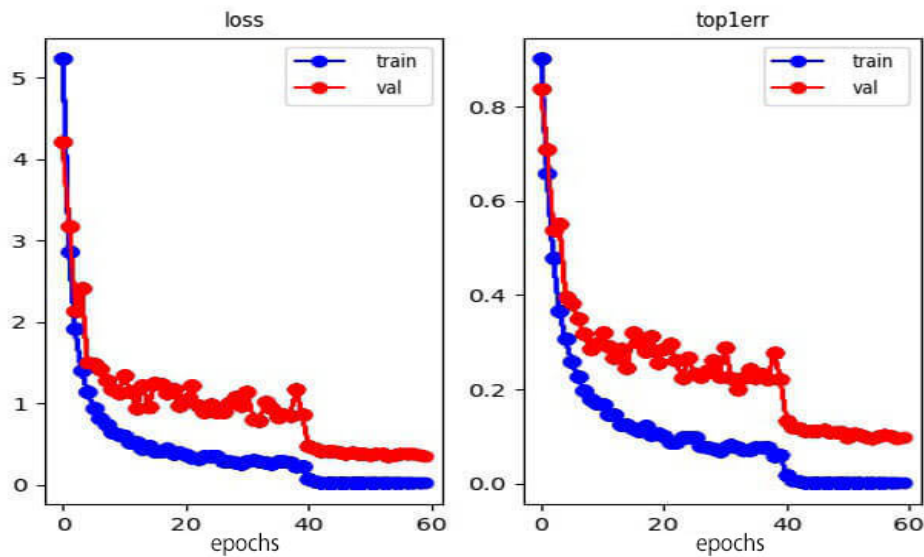


Figure 5.20: Adding 320 persons. **Left**, loss function during training and validation. **Right**, Rank1 error percentage during training and validation.

Pers.	Img.	Rank
0	0	Rank@1:0.893112 Rank@5:0.964964 Rank@10:0.977732 mAP:0.742632
10	473	Rank@1:0.888955 Rank@5:0.964074 Rank@10:0.980404 mAP:0.743795
20	874	Rank@1:0.892221 Rank@5:0.961105 Rank@10:0.974169 mAP:0.745116
30	1252	Rank@1:0.891627 Rank@5:0.965558 Rank@10:0.979513 mAP:0.741414
40	1.682	Rank@1:0.890143 Rank@5:0.964371 Rank@10:0.979513 mAP:0.748799
50	2.046	Rank@1:0.894299 Rank@5:0.962589 Rank@10:0.978028 mAP:0.746853
60	2.427	Rank@1:0.901128 Rank@5:0.967637 Rank@10:0.980404 mAP:0.751229
70	2.859	Rank@1:0.892815 Rank@5:0.965261 Rank@10:0.978325 mAP:0.748843
80	3.214	Rank@1:0.892518 Rank@5:0.965261 Rank@10:0.977732 mAP:0.748257
90	3647	Rank@1:0.899347 Rank@5:0.964964 Rank@10:0.979216 mAP:0.758927
100	4058	Rank@1:0.898159 Rank@5:0.964667 Rank@10:0.980998 mAP:0.755366
150	6.167	Rank@1:0.898753 Rank@5:0.965261 Rank@10:0.979513 mAP:0.761433
200	8.223	Rank@1:0.896378 Rank@5:0.967340 Rank@10:0.980701 mAP:0.763768
250	10.307	Rank@1:0.893705 Rank@5:0.964964 Rank@10:0.980107 mAP:0.762484
280	11.244	Rank@1:0.903504 Rank@5:0.966746 Rank@10:0.982185 mAP:0.767955
300	12.320	Rank@1:0.896081 Rank@5:0.963777 Rank@10:0.978919 mAP:0.768727
320	14.371	Rank@1:0.896675 Rank@5:0.965855 Rank@10:0.978622 mAP:0.769509

Table 5.10: Results of training with a different number of added people. The first row is the base, without adding any images.

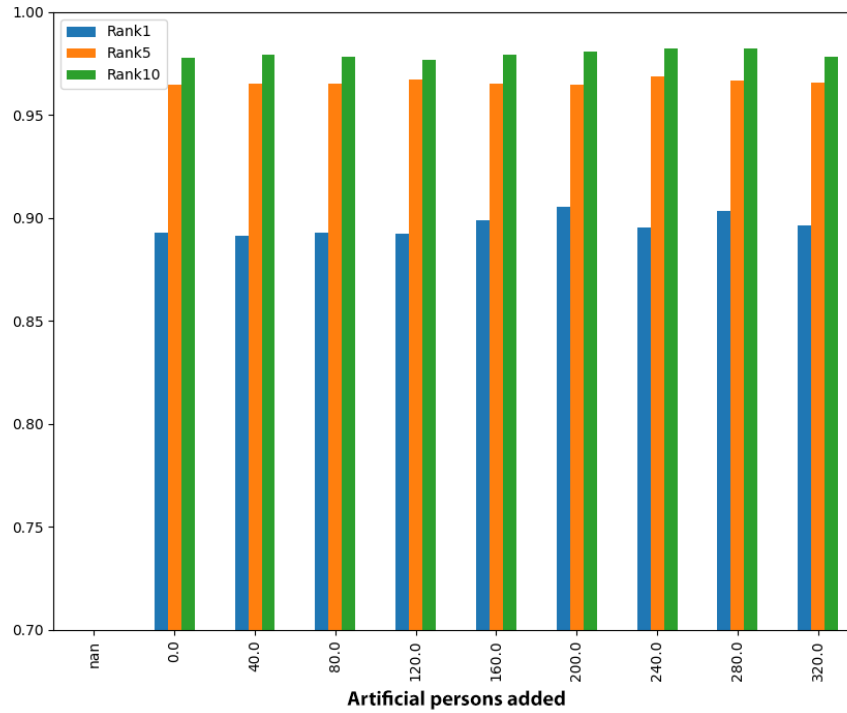


Figure 5.21: Performance of some models trained with different numbers of artificially generated persons (see Table 5.10). Adding 0, 40, 80, 120, 160, 200, 240, 280, and 320 persons.

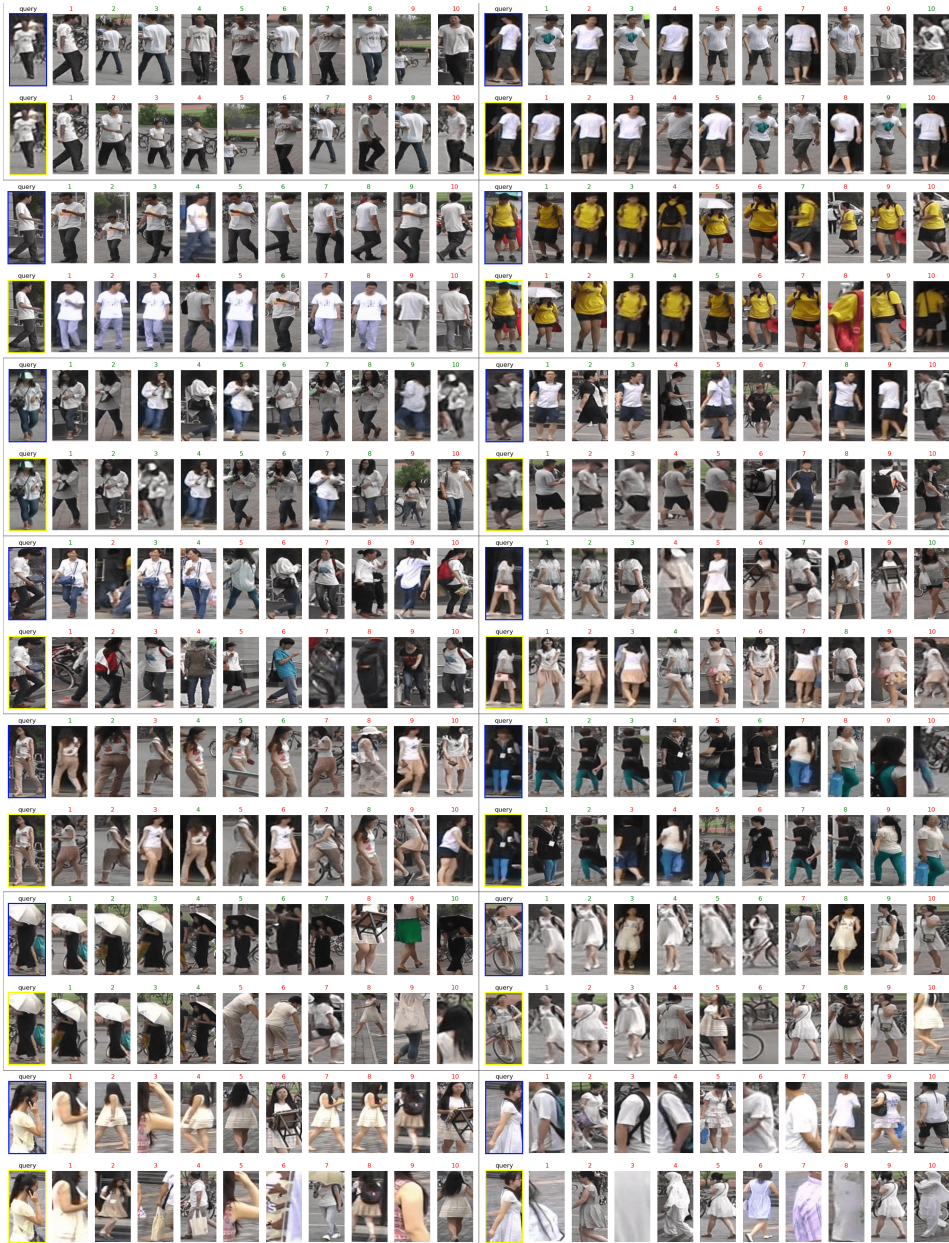


Figure 5.22: Results of re-identification model. Comparison between the results of the base model (■) and the model trained after adding 280 artificial persons (■). The query is the image of the person to be searched for, and the following images represent the model's output, with green indicating a correct match and red indicating an error.

b) Adding artificial images in different poses to each real person.

During the experimentation, we tested adding more images to each person in the training batch, adding them in increments of five. In Fig. 5.21, it can be observed that the performance of the base re-identification model worsens as we add more images, with performance decreasing by up to 8% when adding 100 images per person. This is due to the poor quality of the generated images, which appear very diffused compared to the real training images.

Similarly to the previous section, during training, it can be observed that the model stabilizes after epoch 40 in all experiments, with no significant change in performance based on the number of images added per person (see Figure) 5.23, 5.24, 5.25, 5.20).

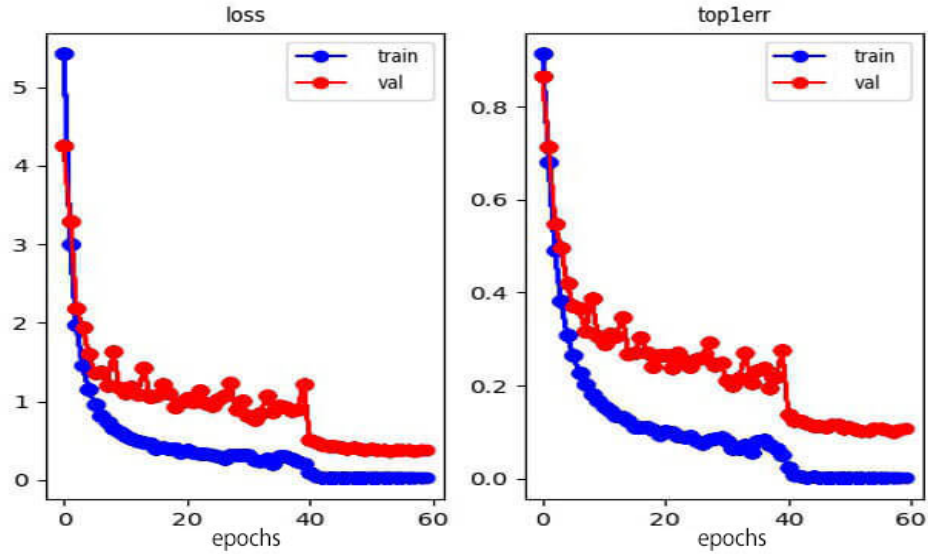


Figure 5.23: Base, without adding any images. **Left**, loss function during training and validation. **Right**, Rank1 error percentage during training and validation.

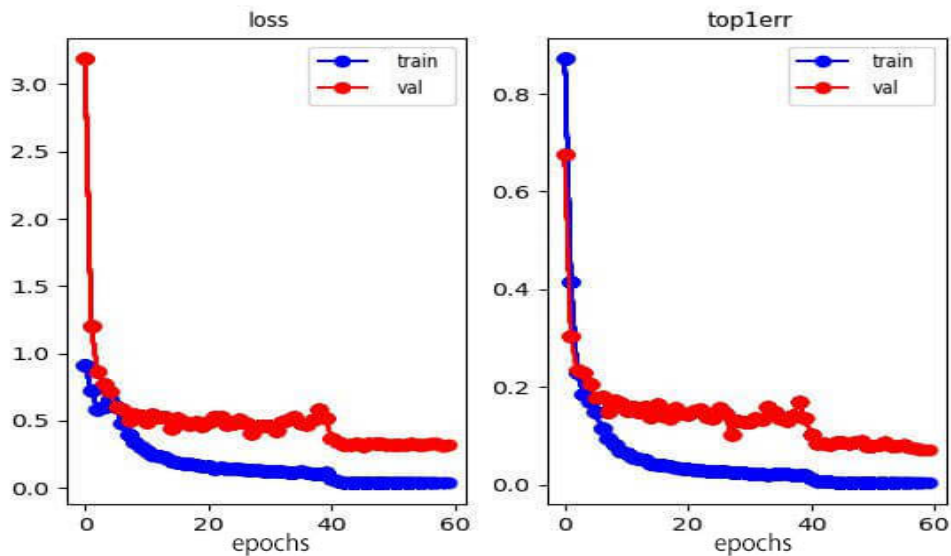


Figure 5.24: Adding 35 images to each person. **Left**, loss function during training and validation. **Right**, Rank1 error percentage during training and validation.

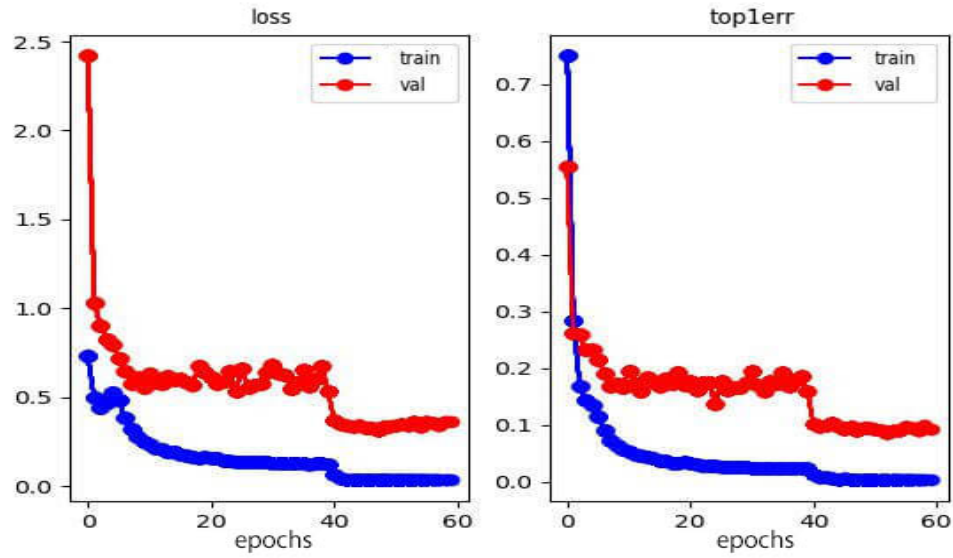


Figure 5.25: Adding 100 images to each person. **Left**, loss function during training and validation. **Right**, Rank1 error percentage during training and validation.

Pers.	Img.	Rank			
		Rank@1	Rank@5	Rank@10	mAP
0	0	0.893112	0.964964	0.977732	0.742632
751	5	0.886876	0.959620	0.975950	0.719692
751	10	0.878860	0.958432	0.977138	0.704901
751	15	0.870546	0.958729	0.969715	0.686172
751	20	0.865796	0.950119	0.972981	0.674047
751	25	0.861342	0.955166	0.971793	0.663773
751	30	0.850950	0.951010	0.969121	0.654492
751	35	0.845606	0.940915	0.964074	0.635790
751	40	0.841449	0.944477	0.966746	0.640035
751	45	0.826306	0.935273	0.961105	0.630469
751	60	0.829869	0.938836	0.960214	0.624382
751	70	0.832245	0.942102	0.965855	0.623470
751	85	0.825713	0.935570	0.959620	0.614014
751	105	0.817399	0.932304	0.958729	0.605653
751	135	0.813539	0.931710	0.958135	0.601084

Table 5.11: Results of training with different numbers of artificially generated people. The first row is the base, without adding any images.

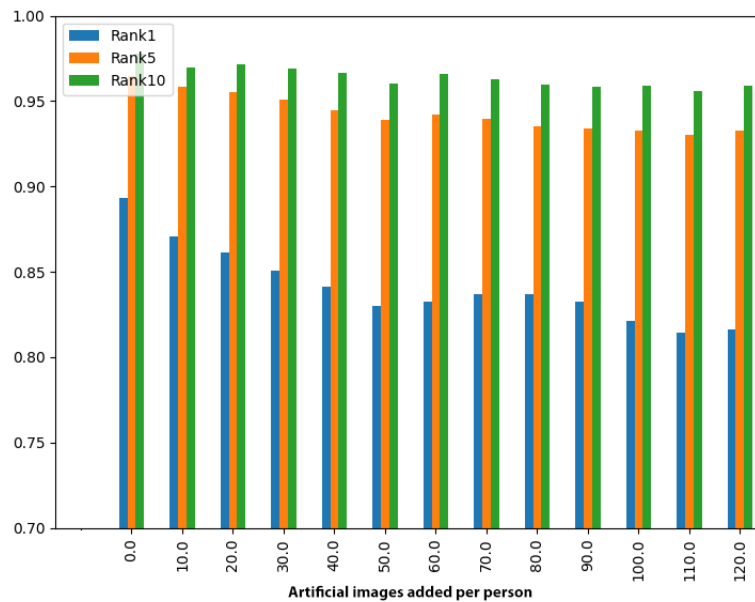


Figure 5.26: Performance of some models trained with different number of artificial images per person (see Table 5.11). Adding 0, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 110 and 120 images per person.

Chapter 6

Conclusions and Future Work

The use of generative adversarial networks to augment data is a promising technique to improve performance in re-identification models. The results obtained in our experiment demonstrate that this technique is effective in generating high-quality data and the versatility to generate modifications of them.

As a result of the experimentation, it was observed that by adding totally artificial people, the re-identification model could improve its performance by 1%. The same could not be achieved with the augmentation of images of existing people because the encoder failed to obtain latent vectors similar to the real images. This may be because these tools are prepared to work with faces and not with images as complex as a whole body.

The use of these generative adversarial networks allows for the use of fewer original data in training, which can reduce resource and time requirements in the model training process. In summary, the use of generative adversarial networks in the field of re-identification is a valuable technique that can provide a significant improvement in the performance of re-identification models. They allow adapting to different datasets and situations, making it a valuable tool not only for the field of re-identification but also for other fields where data generation and improvement are required.

As future work, fine-tuning the encoder could improve its performance in generating latent vectors. Also, performance could be improved by applying other filters that eliminate images of poorer quality. Due to the large number of images, this process should be automated. Another option would be to generate data using pose templates, that is, training StyleGAN3 in a supervised manner with images of people in different poses. Other generative adversarial networks could be used, but for this case, I consider that StyleGAN has generated high-quality and diverse artificial images. Another possibility would be to change the paradigm of re-identification models. Currently, it is based on the Resnet50 neural network with slight modifications used as a feature extractor. An interesting proposal would be to use the latent vectors in StyleGAN3, that is, to measure the cosine distance between images and their latent vectors within StyleGAN3. But for this, the encoder must be functioning correctly.

Currently, we are experiencing a boom in artificially generated content, such as diffusion models. A diffusion model is a type of mathematical model that can be used to generate artificial data. This model is based on the diffusion equation, which is a differential equation that describes how a quantity is dispersed in a continuous medium. The idea is that if the diffusion of a quantity in a continuous medium can be modeled, then artificial data that is similar to real data can be generated.

In the context of re-identification, a diffusion model could be used to generate artificial data of people in different scenes. This artificial data could be used to train a re-identification model, which could increase its performance. For example, the open-source Stable Diffusion model has been a watershed in the generation of artificial images and could be very interesting to see how it can behave for the generation of images of people.

As can be seen, there are several ways to approach and solve the problem, and thanks to advances in generative model architectures, it is possible to propose a wide variety of solutions to improve the performance of any model that requires artificially increasing its training data.

Bibliography

- [1] L. Zheng, L. Shen, L. Tian, S. Wang, J. Wang, and Q. Tian, “Scalable Person Re-identification : A Benchmark University of Texas at San Antonio,” *Iccv*, pp. 1116–1124, 2015. [Online]. Available: <http://www.liangzheng.com.cn>.
- [2] T. Karras, M. Aittala, S. Laine, E. Härkönen, J. Hellsten, J. Lehtinen, and T. Aila, “Alias-Free Generative Adversarial Networks,” no. NeurIPS, 2021. [Online]. Available: <http://arxiv.org/abs/2106.12423>
- [3] Z. Zhong, L. Zheng, Z. Zheng, S. Li, and Y. Yang, “CamStyle: A Novel Data Augmentation Method for Person Re-Identification,” *IEEE Transactions on Image Processing*, vol. 28, no. 3, pp. 1176–1190, 2019.
- [4] E. Ristani, F. Solera, R. Zou, R. Cucchiara, and C. Tomasi, “Performance measures and a data set for multi-target, multi-camera tracking,” *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 9914 LNCS, no. c, pp. 17–35, 2016.
- [5] W. Li, R. Zhao, T. Xiao, and X. Wang, “DeepReID: Deep filter pairing neural network for person re-identification,” *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 152–159, 2014.
- [6] D. Gray, S. Brennan, and H. Tao, “Evaluating appearance models for recognition, reacquisition, and tracking,” *10th International Workshop on Performance Evaluation for Tracking and Surveillance (PETS)*, vol. 3, no. March, pp. 41–47, 2007.
- [7] Z. Zheng, X. Yang, Z. Yu, L. Zheng, Y. Yang, and J. Kautz, “Joint discriminative and generative learning for person re-identification,” *Pro-*

- ceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2019-June, pp. 2133–2142, 2019.
- [8] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila, “Analyzing and improving the image quality of stylegan,” *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 8107–8116, 2020.
- [9] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, “Image quality assessment: From error visibility to structural similarity,” *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004.
- [10] C. Li, K. Xu, J. Zhu, and B. Zhang, “Triple generative adversarial nets,” *Advances in Neural Information Processing Systems*, vol. 2014-Decem, pp. 4089–4099, 2014.
- [11] J. Y. Zhu, T. Park, P. Isola, and A. A. Efros, “Unpaired Image-to-Image Translation Using Cycle-Consistent Adversarial Networks,” *Proceedings of the IEEE International Conference on Computer Vision*, vol. 2017-October, pp. 2242–2251, 2017.
- [12] P. Dai, R. Ji, H. Wang, Q. Wu, and Y. Huang, “Cross-modality person re-identification with generative adversarial training,” *IJCAI International Joint Conference on Artificial Intelligence*, vol. 2018-July, pp. 677–683, 2018.
- [13] A. Wu, W.-s. Zheng, H.-x. Yu, S. Gong, and J. Lai, “RGB-Infrared Cross-Modality Person Re-Identification,” pp. 5380–5389.
- [14] W. Liang, G. Wang, J. Lai, and J. Zhu, “M2M-GAN: Many-to-Many Generative Adversarial Transfer Learning for Person Re-Identification,” 2018. [Online]. Available: <http://arxiv.org/abs/1811.03768>
- [15] W. Deng, L. Zheng, Q. Ye, G. Kang, Y. Yang, and J. Jiao, “Image-Image Domain Adaptation with Preserved Self-Similarity and Domain-Dissimilarity for Person Re-identification,” *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 994–1003, 2018.
- [16] A. H. Re-identification, “arXiv : 1607 . 08378v2 [cs . CV] 26 Sep 2016,” pp. 1–18.

- [17] S. Zhou, M. Ke, and P. Luo, “Multi-camera transfer GAN for person re-identification,” *Journal of Visual Communication and Image Representation*, vol. 59, pp. 393–400, 2019. [Online]. Available: <https://doi.org/10.1016/j.jvcir.2019.01.029>
- [18] Y. Choi, M. Choi, M. Kim, J. W. Ha, S. Kim, and J. Choo, “StarGAN: Unified Generative Adversarial Networks for Multi-domain Image-to-Image Translation,” *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 8789–8797, 2018.
- [19] Y. Tang, Y. Xi, N. Wang, B. Song, and X. Gao, “CGAN-TM: A Novel Domain-to-Domain Transferring Method for Person Re-Identification,” *IEEE Transactions on Image Processing*, vol. 29, pp. 5641–5651, 2020.
- [20] Y. Khraimeche, G.-A. Bilodeau, D. Steele, and H. Mahadik, “Unsupervised Disentanglement GAN for Domain Adaptive Person Re-Identification,” 2020. [Online]. Available: <http://arxiv.org/abs/2007.15560>
- [21] R. Sun, W. Lu, Y. Zhao, J. Zhang, and C. Kai, “A Novel Method for Person Re-Identification: Conditional Translated Network Based on GANs,” *IEEE Access*, vol. 8, pp. 3677–3686, 2020.
- [22] G. Wang, Y. Y. Yang, J. Cheng, J. Wang, and Z. Hou, “Color-sensitive person re-identification,” *IJCAI International Joint Conference on Artificial Intelligence*, vol. 2019-Augus, no. May 2020, pp. 933–939, 2019.
- [23] C. Liu, X. Chang, and Y. D. Shen, “Unity style transfer for person re-identification,” *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 6886–6895, 2020.
- [24] T. Kim, M. Cha, H. Kim, J. Kwon, and L. Jiwon, “Learning to Discover Cross-Domain Relations with Generative Adversarial Networks.”
- [25] X. Zhang, X. Y. Jing, X. Zhu, and F. Ma, “Semi-supervised person re-identification by similarity-embedded cycle GANs,” *Neural Computing and Applications*, vol. 32, no. 17, pp. 14 143–14 152, 2020. [Online]. Available: <https://doi.org/10.1007/s00521-020-04809-7>

- [26] Z. Pang, J. Guo, W. Sun, Y. Xiao, and M. Yu, “Cross-domain person re-identification by hybrid supervised and unsupervised learning,” *Applied Intelligence*, 2021.
- [27] K. He, “Deep Residual Learning for Image Recognition.”
- [28] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, C. V. Jan, J. Krause, and S. Ma, “ImageNet Large Scale Visual Recognition Challenge.”
- [29] Y. Li, S. Chen, G. Qi, Z. Zhu, M. Haner, and R. Cai, “Imaging A GAN-Based Self-Training Framework for Unsupervised Domain Adaptive Person Re-Identification,” pp. 1–16, 2021. [Online]. Available: <https://www.mdpi.com/2313-433X/7/4/62>
- [30] X. Luo, Z. Ouyang, N. Du, J. Song, and Q. Wei, “Cross-Domain Person Re-Identification Based on Feature Fusion,” *IEEE Access*, vol. 9, pp. 98 327–98 336, 2021.
- [31] P. Re-identification, “Pose-Normalized Image Generation for,” *The European Conference on Computer Vision (ECCV)*, pp. 650–667, 2018. [Online]. Available: http://openaccess.thecvf.com/content_{-}ECCV_{-}2018/html/Xuelin_{-}Qian_{-}Pose-Normalized_{-}Image_{-}Generation_{-}ECCV_{-}2018_{-}paper.html
- [32] Z. Cao, T. Simon, S. E. Wei, and Y. Sheikh, “Realtime multi-person 2D pose estimation using part affinity fields,” *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, vol. 2017-Janua, pp. 1302–1310, 2017.
- [33] A. Siarohin, E. Sangineto, S. Lathuiliere, and N. Sebe, “Deformable GANs for Pose-Based Human Image Generation,” *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 3408–3416, 2018.
- [34] Y. Li, T. Zhang, L. Duan, and C. Xu, “A unified generative adversarial framework for image generation and person re-identification,” *MM 2018 - Proceedings of the 2018 ACM Multimedia Conference*, pp. 163–172, 2018.

- [35] Y. Ge, Z. Li, H. Zhao, G. Yin, S. Yi, X. Wang, and H. Li, “FD-GAN: Pose-guided Feature Distilling GAN for Robust Person Re-identification,” *Advances in Neural Information Processing Systems*, vol. 2018-Decem, no. NeurIPS, pp. 1222–1233, 2018.
- [36] P. Isola, J. Y. Zhu, T. Zhou, and A. A. Efros, “Image-to-image translation with conditional adversarial networks,” *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, vol. 2017-Janua, pp. 5967–5976, 2017.
- [37] N. M. G.-b. P.-a. R. Video-based, A. Borgia, Y. Hua, E. Kodirov, and N. M. Robertson, “GAN-based Pose-aware Regulation for Video-based Person Re-identification GAN-based Pose-aware Regulation for Video-based Person Re-identification,” 2019.
- [38] C. Zhang, L. Zhu, S. C. Zhang, and W. Yu, “PAC-GAN: An effective pose augmentation scheme for unsupervised cross-view person re-identification,” *Neurocomputing*, vol. 387, pp. 22–39, 2020. [Online]. Available: <https://doi.org/10.1016/j.neucom.2019.12.094>
- [39] C. Zhang, L. Wu, and Y. Wang, “Crossing generative adversarial networks for cross-view person re-identification,” *Neurocomputing*, vol. 340, pp. 259–269, 2019.
- [40] Y. Zhang, Y. Jin, J. Chen, S. Kan, Y. Cen, and Q. Cao, “PGAN: Part-based nondirect coupling embedded gan for person reidentification,” *IEEE Multimedia*, vol. 27, no. 3, pp. 23–33, 2020.
- [41] Z. Ni, J. Pei, and Y. Zhao, “Affine transform for skew correction based on generative adversarial network method for multi-camera person re-identification,” *ACM International Conference Proceeding Series*, vol. PartF16898, pp. 89–95, 2021.
- [42] H. Chen, Y. Wang, B. Lagadec, A. Dantcheva, and F. Bremond, “Joint Generative and Contrastive Learning for Unsupervised Person Re-identification,” pp. 2004–2013, 2020. [Online]. Available: <http://arxiv.org/abs/2012.09071>
- [43] Z. Zheng, L. Zheng, and Y. Yang, “Unlabeled Samples Generated by GAN Improve the Person Re-identification Baseline in Vitro,” *Proceed-*

- ings of the IEEE International Conference on Computer Vision*, vol. 2017-Octob, pp. 3774–3782, 2017.
- [44] A. Radford, L. Metz, and S. Chintala, “Unsupervised representation learning with deep convolutional generative adversarial networks,” *4th International Conference on Learning Representations, ICLR 2016 - Conference Track Proceedings*, pp. 1–16, 2016.
- [45] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, “Rethinking the Inception Architecture for Computer Vision,” *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2016-Decem, pp. 2818–2826, 2016.
- [46] Y. Huang, J. Xu, Q. Wu, Z. Zheng, Z. Zhang, and J. Zhang, “Multi-pseudo regularized label for generated data in person re-identification,” *IEEE Transactions on Image Processing*, vol. 28, no. 3, pp. 1391–1403, 2019.
- [47] J. P. Ainam, K. Qin, G. Liu, and G. Luo, “Sparse Label Smoothing Regularization for Person Re-Identification,” *IEEE Access*, vol. 7, pp. 27 899–27 910, 2019.
- [48] S. H. S. Hussin and R. Yildirim, “StyleGAN-LSRO Method for Person Re-identification,” *IEEE Access*, pp. 13 857–13 869, 2021.
- [49] C. Eom and B. Ham, “Learning disentangled representation for robust person re-identification,” *Advances in Neural Information Processing Systems*, vol. 32, 2019.
- [50] L. Xia, J. Zhu, and Z. Yu, “Real-World Person Re-Identification via Super-Resolution and Semi-Supervised Methods,” *IEEE Access*, vol. 9, pp. 35 834–35 845, 2021.
- [51] H. Alqahtani, M. Kavakli-Thorne, and C. Z. Liu, “An introduction to person re-identification with generative adversarial networks,” *arXiv*, pp. 1–15, 2019.
- [52] Z. Luo, “Review of GAN-Based Person Re-Identification,” 2021.
- [53] Y. Jiang, W. Chen, X. Sun, X. Shi, F. Wang, and H. Li, *Exploring the Quality of GAN Generated Images for Person Re-Identification*. Association for Computing Machinery, 2021, vol. 1, no. 1.

- [54] T. Karras, S. Laine, and T. Aila, “A style-based generator architecture for generative adversarial networks,” *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2019-June, pp. 4396–4405, 2019.
- [55] Q. Cai and J. K. Aggarwal, “Tracking Human Motion Using Multiple Cameras,” 1996.
- [56] P. Dimitrakopoulos, G. Sfikas, and C. Nikou, “Wind: Wasserstein Inception Distance for Evaluating Generative Adversarial Network Performance,” *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, vol. 2020-May, pp. 3182–3186, 2020.
- [57] Y. Alaluf, O. Patashnik, Z. Wu, A. Zamir, E. Shechtman, D. Lischinski, and D. Cohen-Or, “Third Time’s the Charm? Image and Video Editing with StyleGAN3,” 2022. [Online]. Available: <http://arxiv.org/abs/2201.13433>
- [58] E. Richardson, Y. Alaluf, O. Patashnik, Y. Nitzan, Y. Azar, S. Shapiro, and D. Cohen-Or, “Encoding in Style: A StyleGAN Encoder for Image-to-Image Translation,” *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 2287–2296, 2021.
- [59] Z. Jiang, L. Zhao, L. I. Shuaiyang, and J. I. Yanfei, “Real-time object detection method for embedded devices,” *arXiv*, vol. 3, no. October, pp. 1–11, 2020.
- [60] Y. Yu, D. Zhang Weibin, and Yun, “Frechet Inception Distance (FID) for Evaluating GANs,” no. September, pp. 0–7, 2021.
- [61] M. Kettunen, E. Härkönen, and J. Lehtinen, “E-LPIPS: Robust Perceptual Image Similarity via Random Transformation Ensembles,” 2019. [Online]. Available: <http://arxiv.org/abs/1906.03973>
- [62] H. Zhao, O. Gallo, I. Frosio, and J. Kautz, “Loss Functions for Neural Networks for Image Processing,” no. November, 2015. [Online]. Available: <http://arxiv.org/abs/1511.08861>
- [63] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, “Momentum Contrast for Unsupervised Visual Representation Learning,” *Proceedings of the*

IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 9726–9735, 2020.