Machine Learning in Applications
Face Aging with Generative Adversarial Network
Project Report

Mattia Dutto, Julian Neubert, Simone Alberto Peirone
Politecnico di Torino

CONTENTS

LIST OF FIGURES

LIST OF TABLES

Machine Learning in Applications
Face Aging with Generative Adversarial Network
Project Report

***Abstract*—The recent explosion in the popularity of image and video-centric social networks, including Instagram and Snapchat, has fostered the development of applications and algorithms that can modify existing content or create realistic new images and videos. Among these, mobile applications such as FaceApp leverage machine learning techniques to allow users to edit their face by applying a wide range of filters, from hairstyle changes and gender swapping to wrinkles and acne removal. While the mechanisms behind these popular apps are not yet publicly available, researchers have addressed this topic by exploiting autoencoders and generative adversarial networks to translate images from a source domain to a different target domain. In this work, we analyze state-of-the-art approaches based on CycleGAN and StyleGAN for creating a facial ageing filter and propose a new solution that shows convincing results.**

I. Introduction

A filter translates an image from one domain to a different one, e.g. by adding wrinkles and skin imperfections to the face of a young person to make it appear older. Some filters can be implemented using traditional computer graphics algorithms while others require more profound modifications of the original image. Deep learning has proved successful in the development of complex image and video manipulation techniques, often known as Deepfakes. Authors of [1] provide a throughout analysis of the literature on deepfake creation and detection. We concentrate on image filters, with an emphasis on ageing filters. However, all the proposed approaches are general enough that they can be applied to different filters with minimal modifications or finetuning.

A common approach for filter development exploits encoder-decoder pairs to construct an intermediate representation of the input image to be reconstructed in a possibly different output domain. The encoder is shared between the source domains while a decoder is trained specifically for each target domain. Additionally, an *adversarial loss* can improve the network outputs by ensuring that the generated samples are indistinguishable from the real samples in the target domain. Similarly, a *perceptual loss* can enhance the reconstruction capability of the network by reducing the distance between the mid-level representations of the original and reconstructed samples. CycleGAN [2] is a generative adversarial approach to domain translation that combines an adversarial component with a cycle consistency loss that minimizes the reconstruction error in the image's source domain. Recently, StyleGAN [3] showed astonishing fake image generation capability.

Recently, StyleGAN [3] showed astonishing fake image generation capability. In this work, we analyze the transferability of pretrained CycleGAN models to the face ageing task on a large image dataset (CelebA). Then, we build upon the work of [4] to adapt StyleGAN for the domain translation task.

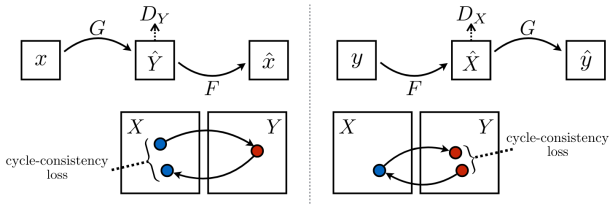

Fig. 1. **Block diagram of CycleGAN** from [2]. Generator G translates image x from domain X to Y while F performs the opposite operation. A consistency loss is computed between the original image x and its reconstructed version $\hat{x}$. Ideally, the composition of G and F should approximate the identity function. A similar operation, translates image y to the target domain and back. The adversarial losses produced by D_X and D_Y ensure that the generated samples $\tilde{x}$ and $\tilde{y}$ are indistinguishable from the real samples of X and Y. On the other side, the cycle-consistency component of the loss measures the distance between the original image and its reconstruction in the same domain.

II. Background

The initial assignment proposed to base our project on autoencoders (AEs), however, after reading the survey by Nguyen et al. [1] we discover that this approach only works well with low-resolution images. Since we work with the CelebA dataset, we decided to focus our experiments on Generative Adversarial Networks (GANs) which allow us to achieve better results with the higher resolution of the images provided. The concepts of GANs and AEs are somewhat related, however, and can be combined, as we show in section IV-C.

While there is a lot of research publicly available about creating and more importantly detecting deep fakes, this is not the case for filters. The vast availability of filters in apps and on social media networks stands in stark contrast to the low amount of research papers or open source code that can be found online. Apart from a few specialized cases, such as the removal of eyewear by Hu et al. [5] most algorithms seem to be proprietary software. Nonetheless, public projects do exist that aim to implement popular filters. Depending on the specifics of the filter, one popular approach is to use landmark detection or segmentation masks to align objects and stitch them together, such as this face swap algorithm proposed by Korshunova et al. [6]

Other popular approaches borrow from the fields of style transfer and image-to-image translation. The authors of [7] implement the CycleGAN by Zhu et al. [2] to map images from a dataset containing young people to a different dataset with

images of old people. StyleGAN is a powerful architecture introduced by Karras et al. [3] which is able to generate rich images and seamlessly mix the styles of two images. Though purely a GAN architecture this serves as the basis for the auto-encoder StyleGAN (AE-StyleGAN) by Han et al. [4] which encodes real input images as styles and allows for complex methods to mix these styles together.

Antipov et al. [8] use Conditional Generative Adversarial Networks to age or rejuvenate people. However, they exploit the granular values for the age attribute provided by the *IMDB-Wiki_cleaned* [9] dataset for training. The CelebA dataset on the other hand only defines a binary *young/old* flag for each image.

A. CelebA dataset

CelebFaces Attributes Dataset (CelebA) [10] is a large-scale dataset providing more than 200K celebrity images. Each image is annotated with multiple attributes describing its semantic content. Some of them are objective image properties such as hair color, beard and face shape, while others are more subjective, such as youth and attractiveness. To reduce the performance burden of high-resolution image processing, all the experiments were performed on center-aligned images cropped to 128 by 128 pixels.

CelebA images are labeled with the attribute *young*, which makes this dataset a suitable testbed for creating a face aging filter. However, by plotting the correlation matrix of the image annotations, we notice that the attributes tend to be highly correlated with each other, as shown in Figure 2. In addition, the composition of the dataset itself is a source of bias: the authors of [11] show that less than 20 percent of CelebA represent nonwhite faces. Another problem with the dataset is the overall quality of the images, which are sometimes blurred, have occlusions or visible artifacts. Comparison of CelebA with other similar datasets, such as FFHQ, which contain higher quality images, led us to the conclusion that CelebA is a difficult dataset to learn.

III. Materials and methods

This project focuses on the implementation and analysis of different GAN architectures for image manipulation. A discussion of the impact of the different CNN backbones used to implement all network components is beyond the scope of this project and left as future work.

A. A reference network

Generative Adversarial Networks describe a class of neural networks in which two components, a generator and a discriminator, compete in a minimax game to generate fake samples that have the same probability distribution as real data. The generator transforms random noise into fake samples while the discriminator distinguishes between the real samples and the ones produced by the generator. The generator is trained to increase the discriminator's confusion, causing it to classify fake samples as real. The latter, on the other hand, tries to reduce its classification error. Several extensions to the GAN concept have been proposed over the past few years. Regarding style transfer, one of the most common approaches is CycleGAN.

With Cycle-GAN, whose diagram is shown on fig. 1, we have a GAN network mapping between two domains, followed by a second GAN network that connects back to the first domain, closing the loop. The objective of the network is to translate a sample image $x \in X$ to a different image $\tilde{y}$, that appears to belong to a different domain Y, and then use the second network to go back to the original image. Obviously, unless the network can perfectly implement the identity function, the resulting image $\tilde{x}$ is different from the original and represents an attempt to reconstruct the original image. In other words, CycleGAN combines two objectives: image translation between different domains and reconstruction in the source domain.

More in detail:

1) $G : X \longrightarrow Y$ is a GAN generator that maps input x from domain X to Y.
2) $F : Y \longrightarrow X$ translates input y from Y to X.

G and F are implemented using a sequence of convolution and deconvolution blocks to reduce the dimensionality of the input image to an intermediate representation and reconstruct it in the target domain.

For each GAN component, the **adversarial loss** ensures that the network produces realistic images that look like instances of the target domain. The formula of the adversarial loss for $G : X \longrightarrow Y$ is:

$$L_{GAN}(G, D_Y, X, Y) = E_{y \sim pdata(y)}[logD_Y(y)] + E_{x \sim pdata(x)}[log(1 - D_Y(G(x))] \quad (1)$$

Where:

- X is the source domain,
- Y is the target domain,
- G is the network's backbone that translates the input from X to Y,
- D_y is the discriminator for domain Y,
- $x \sim pdata(x)$ is a sample from X according to distribution $pdata(x)$.

Basically, the generator and the discriminator compete against each other. Improving the performance of the discriminator worsens the performance of the generator and vice versa. The **cycle-consistency-loss** that is a loss that will help the network in the reconstruction of the original image. The cycle-consistency-loss measures the L1 (or L2) distance between the original images and their reconstructed versions, as shown by eq. 2.

$$L_{cyc}(G, F) = E_{x \sim pdata}(x)[\|F(G(x)) - x\|_1] + E_{y \sim pdata}(y)[\|G(F(y)) - y\|_1] \quad (2)$$

CycleGAN is trained using data from both domains X and Y to train the translation process in both directions. More specifically, for the implementation of the face ageing filter images are separated separated in two domains *young* and *old* according to the *young* label provided by CelebA.

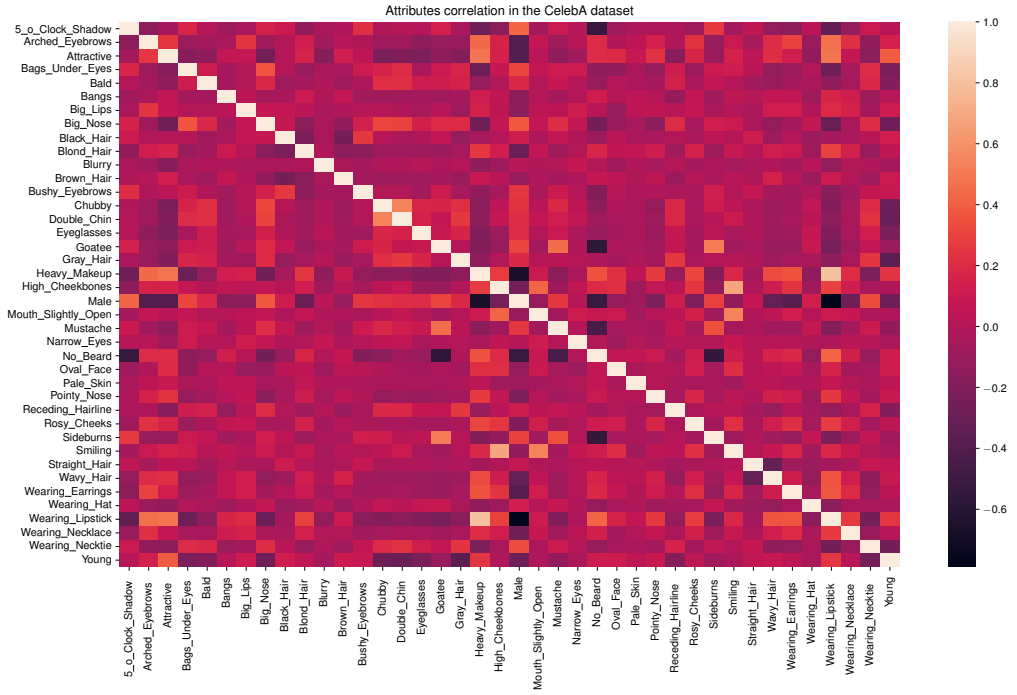

Fig. 2. **Attributes correlation in the CelebA dataset.** The *young* attribute is positively correlated to *attractive* ($\sigma = 0.39$), *wearing_lipstick* ($\sigma = 0.25$) and *heavy_makeup* ($\sigma = 0.24$) and negatively correlated to *gray_hair* ($\sigma = -0.36$), *double_chin* ($\sigma = -0.31$) and *chubby* ($\sigma = -0.29$). Overall, these correlation values hinder the network's ability to adequately discriminate between young and old and can result in artifacts, even in the presence of proper regularization terms.

B. AE-StyleGAN

StyleGAN was introduced by NVIDIA in 2019. Authors of [4] provide an extension of StyleGAN combining it with an autoencoder architecture. StyleGAN's results are quite better with respect to the ones of previous GAN architectures, to a different way to manage the latent space and an effective style control mechanism. Indeed, AE-StyleGAN allows to mix styles of different images at the latent space level, enabling the implementation of an ageing filter by simply mixing the styles information of young and old people. On 3 we can see the base architecture implemented in the original StyleGAN paper.

A central component of StyleGAN is the AdaIN normalization layer. Given an image input x_i and style information y, it computes the following expression:

$$AdaIN(x_i, y) = y_{s,i} \frac{x_i - \mu(x_i)}{\sigma(x_i)} + y_{b,i} \tag{3}$$

AdaIN normalizes each feature map x_i with respect to the style provided by y. Feature maps are normalized separately to have zero mean and unit variance, as in Instance Normalization. Then, the output is scaled and translated using the corresponding parameters from style y to match its statistics. Intuitively, AdaIN gives more or less importance to the feature maps of the image according to y, enabling style transfer to the final image.

StyleGAN requires some additional regularization terms and measures to improve the training and ensure convergence of the network weights. Among these, R1 regularization and

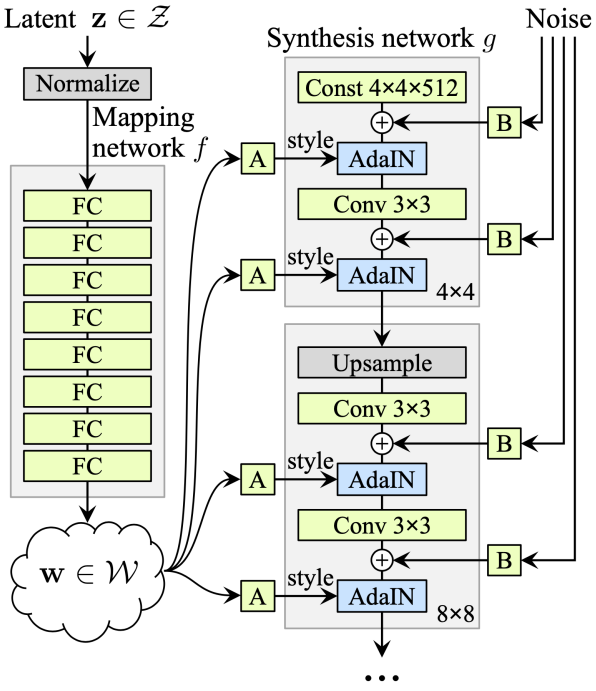

Fig. 3. **Architecture of StyleGAN.** The mapping network implements a non-linear mapping of the latent code **z** to **w**, which encodes the *style* information. The synthesis network is composed of 8 block, each doubling the resolution of the output (4x4 to 8x8 and so on...up to 512x512). The output of each convolution layer is summed with a random noise **B** and normalized using an AdaIN block and the affine coefficients from style **A**.

Exponential Moving Average are two keypoints for both AE-StyleGAN and its extension presented in the following section.

1) R1 regularization: this regularization term smooths out changes in the discriminator weights ψ by introducing a gradient penalty term on real data as shown in eq. 4. R1 regularization is a key to stabilizing training and improving the convergence of the discriminator loss.

$$R_1(\psi) = \frac{\gamma}{2} E_{p_D(x)} \left[||\nabla D_\psi(x)||^2 \right] \tag{4}$$

2) Exponential Moving Average: to improve the stability of the network, the weights of the encoder and the generator are accumulated during training using an exponential moving average (EMA) which reduces the impact of one single batch of data. Given a network G at iteration i and a coefficient λ, its EMA counterpart $\tilde{G}_i$ is computed as shown in eq. 5. [1]

$$\tilde{G}_i = \lambda \tilde{G}_i + (1 - \lambda) G_i \tag{5}$$

The AE-StyleGAN network is composed of three components. An **encoder** projects the input images in the latent space. A **generator** randomly samples the latent space to produce new fake images and reconstructs the images encoded by the encoder. A **discriminator** classifies real and back images. More in details, the training process is divided into three parts:

1) *train the discriminator*: all the other components are frozen and the discriminator is trained to identify fake and real images using a classification loss and R1 regularization.
2) *train the encoder*: input images are projected in the latent space and reconstructed in the original domain, ensuring that the reconstruction error is minimized. Only the encoder updates its weights, while the generator is kept frozen.
3) *train the generator*: the generator is trained to reconstruct images projected by the encoder in the latent space and to generate completely new images. The generator is trained using an adversarial loss to improve the quality of the fake samples and a reconstruction loss for the real samples. During this phase, only the generator receives gradient updates.

C. Cycle-StyleGAN

StyleGAN shows excellent results in image generation while AE-StyleGAN successfully expands the network with the ability to reconstruct existing images. The implementation of a filter requires a further extension. Indeed, the generator component of the network should reconstruct the input image in a different domain while retaining most of its original visual appearance.

Therefore, we combine the cycle approach of CycleGAN with the reconstruction capability of AE-StyleGAN, as shown in figure 4. A source encoder projects the input image to a latent space conditioned on its class label. Then, a generator reconstructs the image in the target domain. An adversarial loss provided by a discriminator for the target domain is used to train the two networks. We call this operation *forward pass*.

To ensure that the visual appearance of the original image is preserved, the output of the generator is fed to a second encoder and projected back to the original domain (*backward pass*). A pixel loss measures the distance between the original image and its reconstructed version. The encoders and the generators are initialized with the weights of AE-StyleGAN and finetuned using the adversarial and pixel losses to move from their reconstruction task to the filter implementation. Equation 6 shows all the components of the loss for an *old-to-young* filter. $\tilde{G}_{young2old}$ and $\tilde{E}_{young}$ denote the EMA counterparts of $G_{young2old}$ and E_{young}.

$$\begin{aligned} y &= G_{old2young}(E_{young}(x)) \\ L_{young,adv} &= \textit{Softplus}\left(D_{old}(y)\right) \\ L_{young,pix} &= \|\tilde{G}_{young2old}(\tilde{E}_{young}(x)) - y\|_2 \\ L_{young} &= \alpha_{adv} L_{young,adv} + \alpha_{pix} L_{young,pix} \end{aligned} \tag{6}$$

Only the encoder and the generator from the forward pass are updated with the adversarial and pixel losses, encouraging the two networks to learn a mapping from the source domain to the target independently of the reconstruction technique used in the backward pass. From a practical point of view, limiting the number of network subject to gradient updates also allows to reduce training time and increase the batch size.

The training process is split in three steps. First, the discriminators for the two domains are trained on real data using softplus loss and R1 regularization. Then, the first filter *young-to-old* is trained and the weights of the updated network components are accumulated using EMA. Finally, the same process is repeated for the *old-to-young* filter.

IV. Experiments and discussion

A. CycleGAN

To test CycleGAN, we exploited pretrained weights obtained on several datasets different from CelebA. The network is finetuned on the train split of CelebA for a limited number of iterations. The reasons behind this choice are the following:

- **Training time**: CycleGAN training is slow. Running 200 training epochs on the full dataset requires weeks of GPU power, making this route impractical.
- **Overfitting**: Training time could be reduced by limiting training to a subset of the CelebA dataset. However, training from scratch on a limited dataset could lead to overfitting.

We decided to use the weights pretrained on the female images of IMDB-Wiki dataset [12] and fine-tuned on male images of IMDB-Wiki. According to [7] this configuration produces the best results. Figure 5 shows the effect of different pretrained models finetuned on CelebA. Different models have varying results and some models create more artifacts than others.

We tested both a fine-tuning strategy on a 10% portion of the dataset used and with a class-balanced subset. To create a

[1] Exponential Moving Average is particularly important in the context of StyleGAN due to the big size of the network that limits the batch size during training.

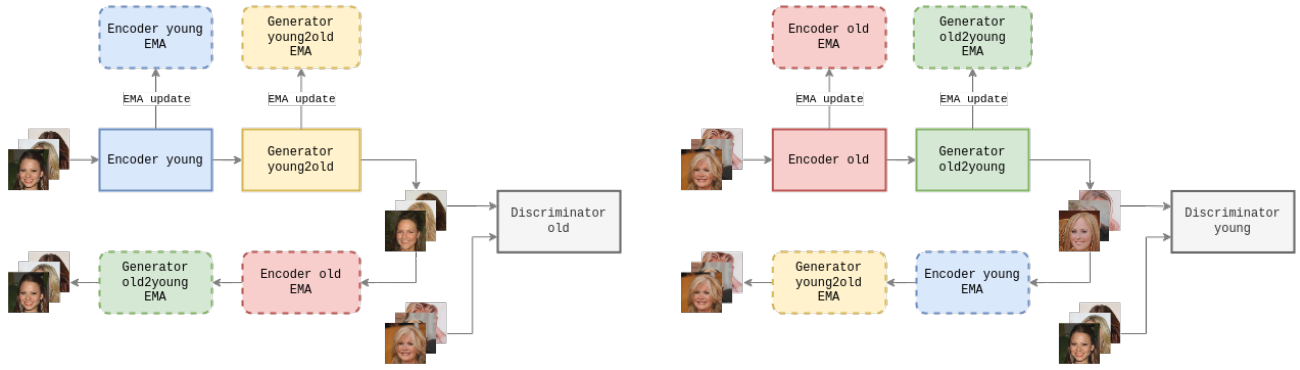

Fig. 4. **Block diagram of Cycle-StyleGAN.** The diagram on the left implements a *young-to-old* filter while the one on the right describes an *old-to-young* filter. The blocks with a solid border are updated using gradient descent while the dashed ones follow their counterparts using an exponential moving average. Overall, training a filter only updates the encoder and the generator of the forward pass and exploits the frozen encoder and generator during the backward pass. The training loss is computed as a weighted sum of the adversarial component produced by the discriminator and a pixel loss on the original and reconstructed images.

class-balanced dataset we selected the first 8000 images from both classes (*young* and *old*).

We can now start to fine-tune the network and see how different results we can produce. In particular the pretrained models used for these experiments were trained for 200 epochs [7]. Then, the selected one the number #9 was fine-tuned for other 100 epochs. We decide to fine-tune a little bit more and we compare 1, 5, 10, 25 and 50 fine-tuning epochs. We can see the different results between the 1/10 of the dataset and the balanced one on figure 6. CycleGAN shows the best results after 10-25 epochs. A deeper study on the loss may tell us that with 50 epochs and 16000 we are already over-fitting. A few considerations, training from scratch on the full dataset probably would have given us better results, but training a cycle-GAN and performing an hyper-parameters search to find the correct parameters is a time-consuming effort. Moreover, we have to consider that this is just a starting point: we look at these results with the hope of improving them.

For the next experiments we move from a subset of the database to the full dataset.

B. AE-StyleGAN

Since pretrained models for AE-StyleGAN are not available, we trained AE-StyleGAN from scratch, using only the pre-trained weights for the generator component from the original StyleGAN repository. Training is slow, requiring almost 24 hours to complete 10k iterations. The model trained for 75k iterations, compared to the original StyleGAN which was trained for 200k iteration. For doing this the training process lasted about 1 week. For training, we used the configuration suggested by [4] (1 encoder step, weight of the R1 regularization term set to 0.2048 and weight of the reconstruction loss 0.25). The maximum batch size allowed on a single V100-32GB GPU is 16. Even though the code supports distributed training, we decided to use only a single GPU for training to reduce the wait time for job scheduling on the HPC cluster. Overall, we encountered several issues in getting the code to work properly.

After AE-StyleGAN training, we performed a small hyper-parameters search of an intermediate model (11k iterations) to determine whether we could have improved the training. The hyper-parameters search was performed after the main network training for the following reasons:

- We couldn't move to the Cycle-AE-StyleGAN without a trained AE-StyleGAN this due to the fact that in the last part of the research we just fine-tuned a little bit the network. Therefore, training of AE-StyleGAN represented a major bottleneck for the project development.
- Probably, if we decided to do an ablation study at the start we could have obtained better performance of the network. However, results show that the default configuration works quite well with our dataset.

For the hyper-parameters search, we compare the effect of what we consider the most important parameter of the network:

- **Number of the encoder step**: number of times that we have to train the encoder per each iteration. See results on fig. 7.
- **Regularization factor R1**: weight of the R1 regularization factor on the regularization part, it will act on the r1_loss. See results on fig. 8.
- **Reconstruction weight** λ_{rec_d}: this is the weight of the reconstruction loss when we compute the general loss of the network. See results on 9.

The number of the encoder step is the only parameters that affects the performance of the network. Doubling the number of step means doubling the execution time. All the experiments were trained for 1 day. We measure the performance of the hyper-parameters search in terms of reconstruction loss and FID.

- **Reconstruction loss**: we want to see if the pixel loss of the network is decreasing with the training time.
- **FID**: needing to evaluate a generative model with an unbalanced dataset, we thought that the Frechet Inception Distance would be the benchmark metric for the network. The purpose of the metric is to tell us how far apart the

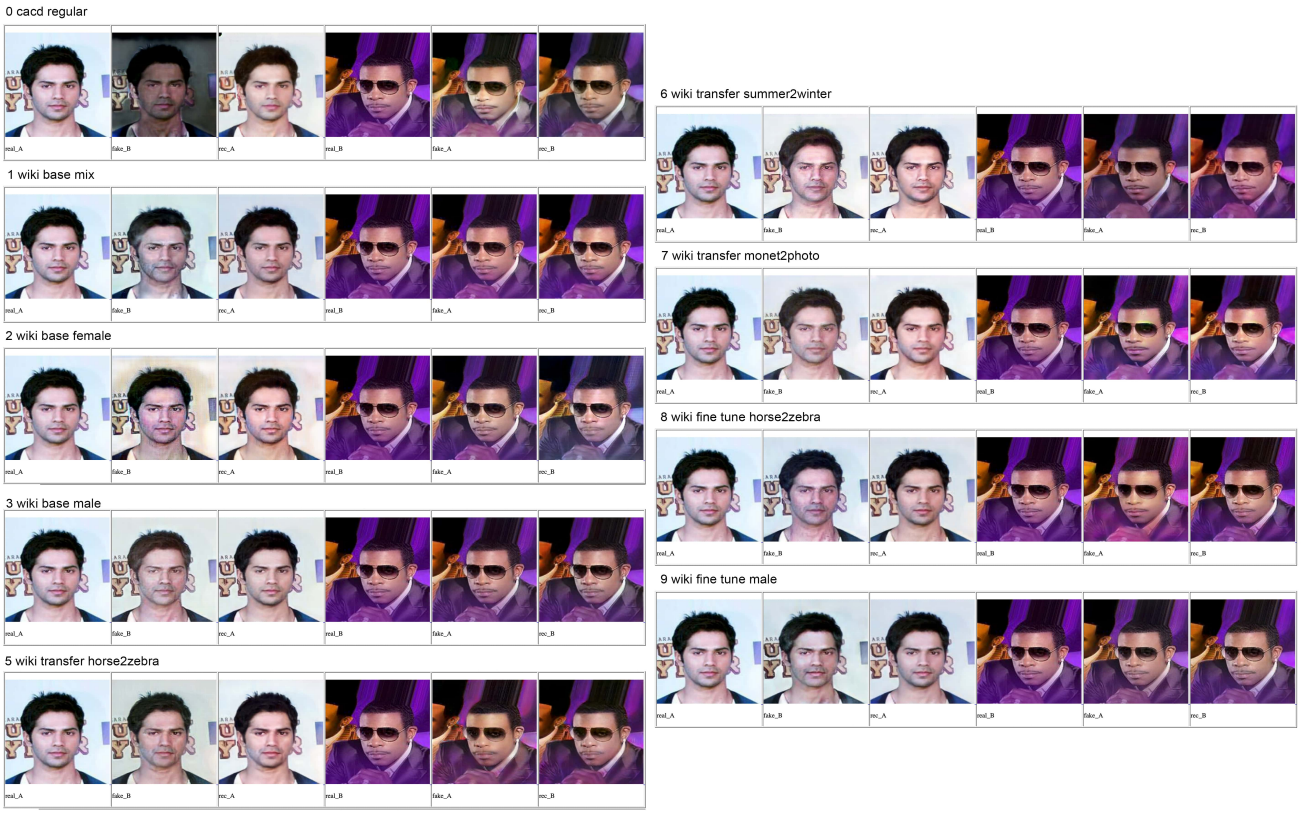

Fig. 5. In this image we can see how the base model is performing with respect to the different pretrained models on the CelebA dataset. On each row, the first three images implement a *young-to-old* filter while the remaining three show the opposite filter. For each filter, the three images represent respectively the source image, the image after the filter application and reconstructed image in the original domain. The pretrained models are pretrained or on CACD [13] dataset or on IMDB-Wiki [12] dataset (similar datasets to CelebA). Even though the filter application results in very subtle changes in the overall appearance of the faces, white faces show a slightly better behaviour while non-white faces do not seem to be affected at all.

Fig. 6. **Left: results of Cycle-GAN on the class-unbalanced dataset**. In this case we selected the pretrained model n. 9 (fine-tuning of the model 3) and we can see the difference at different epochs. Basically more fine-tuning on CelebA better results. **Right: results of Cycle-GAN on the class-balanced dataset.** The results are quite similar.

generated images are from the original ones. First, we calculate the mean and covariance matrix of the features extracted from an Inception network from the real and generated images. Then, eq. 7 determines the distance between the computed statistics.

$$fid = d^2 = ||\mu_g - \mu_r||^2 + Tr(C_r + C_g - 2 * \sqrt{C_r * C_g}) \quad (7)$$

Where:

- μ_r and μ_g: are the mean of the features of the real and generated images
- C_r and C_g: are the covariance matrix of the real and generated images

The AE-StyleGAN give us the possibility to compute different losses: in this case we will focus on the pixel loss of the and not on the vgg loss, and different FID: we will focus on the FID on the reconstruction images.

For each hyper parameter we compare the results with the base models described before. And for each hyper parameter we tried 4 different values (including the base configuration).

C. Mixing styles

In this section, we explore different approaches to applying a filter by exploiting StyleGAN's inherent ability to mix styles of different images at different levels of granularity. Different from the CycleGAN concept, we do not need to retrain our model for each filter we want to implement. Instead, we use the AE-StyleGAN architecture that was trained as explained in section IV-B. By choosing which image we mix at each level we control the appearance of the result. To implement our filter, we select a set of representative images of young and old people with different characteristics to compare the results. Figure 10 shows the images we selected and how well the AE-StyleGAN is able to reconstruct them. For the rest of this section, we refer to images by the label assigned to them in this figure.

We can see the reconstructed images as an upper bound for how recognizable the original person is after applying the filter. The StyleGAN architecture we used has 12 distinct locations or levels at which the latent representation is fed into the synthesis network (See fig. 3). When we refer to mixing a latent at a higher level we refer to taking the fine grained style of that latent, i.e. feeding it later into the synthesis network at higher resolutions. Analogously lower level mixing refers to taking the coarse styles of that latent. We observe that lower level styles encode the main features and shape of a face making us recognize that person in the output, while higher level styles influence the more detailed appearance of a picture by changing the colors and textures. Based on this observation we attempt to mix different styles to apply an aging filter to an input image.

In our initial experiments, we take the naive approach of taking the coarse lower-level styles from the input image and the finer higher-level styles of a different image. Figure 11 shows some results for taking "young 1" as our input image and applying the styles of pictures of old people at different levels. We can see that this works to some degree but is highly sensitive to the selected images and styles. It is not possible to tell the network which specific characteristic of the mixing image we want to apply to the input image. In practice, this would mean selecting a mixing image that is as similar as possible to the input image except for that single characteristic that we are trying to change. Due to the large number of factors playing a role here (age, gender, skin color, lighting, ...), we deem this approach unsuitable for real world applications.

To address these issues we experiment with several techniques. Firstly, instead of relying entirely on the mixing image for higher level styles, we also take some from the input image in an alternating pattern. The intuition behind this is that the network has to negotiate the appearance of the output image to include finer details of both the input image and the mixing image giving a better balance in skin color and other details. In figure 12 we see that this does improve the similarity to the person in the input image while keeping features from the mixing image that are important to the filter, such as wrinkly skin. We note however that some details of the input image are lost with this technique and that its usefulness may depend on the specific filter you want to implement.

Furthermore, we attempt to reduce the dependence on the quality and fitness of the mixing image. Simply extending the idea of using different styles at alternating levels with more than one mixing image does not yield great results, as can be seen in figure 13. Instead, we rely on another feature of the StyleGAN architecture, the linear relation of features in latent space. We note that this assumption may be violated due to the retraining in the AE-StyleGAN setting. Specifically, we observe that computing the mean over a large set of images in latent space does not yield any sensible results when fed directly into the synthesis network. However, figure 14 shows us that we can achieve good results when computing the mean over smaller sets of images, such as the ones presented in figure 10. We are able to achieve pretty good results when combining the mean of a set of styles as the mixing style with the mixing techniques presented at the beginning of this section, see figure 15.

Overall we achieve impressive results which are mostly limited by the network's ability to reconstruct the original input image. One should note that this technique does require some manual effort when implementing a new filter, such as selecting suitable images and verifying the results. However, once a good setting has been found, this technique is simple at practical, giving good results for a variety of input images regardless of gender or race. Its only shortcoming is that it is not possible to directly tune the filter, only via the selection of mixing images.

D. Cycle-StyleGAN

We use the weights of the AE-StyleGAN network after 75k iterations to bootstrap our Cycle-StyleGAN architecture. The network is further trained for 300 iterations using a smaller learning rate (0.0001) and $\lambda = 0.9999$ for EMA updates. The loss weights are $\alpha_{adv} = 0.1$ and $\alpha_{pix} = 1.0$, as in AE-StyleGAN. All the remaining parameters are shared with the configuration of AE-StyleGAN.

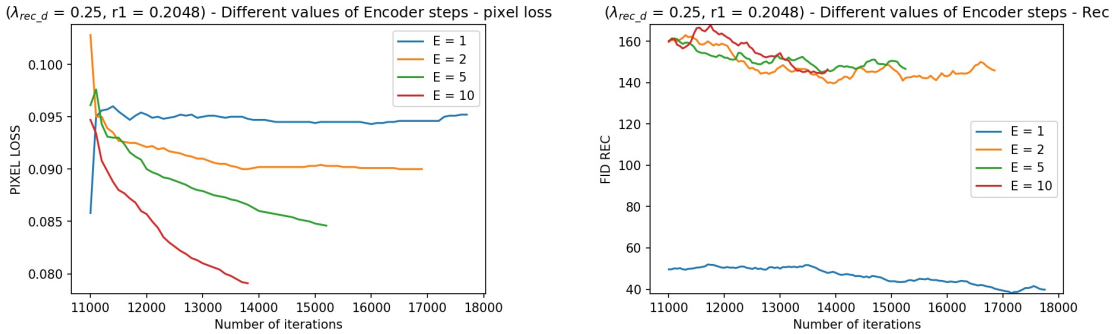

Fig. 7. **Hyper-parameters search on the number of encoder steps**. **Left**: the number of steps of the encoder affects the training time. The good point is that the loss is always decreasing after a small increment at the start. **Right**: from the FID point of view the trend is basically the same except for case with number of steps equal to 1.

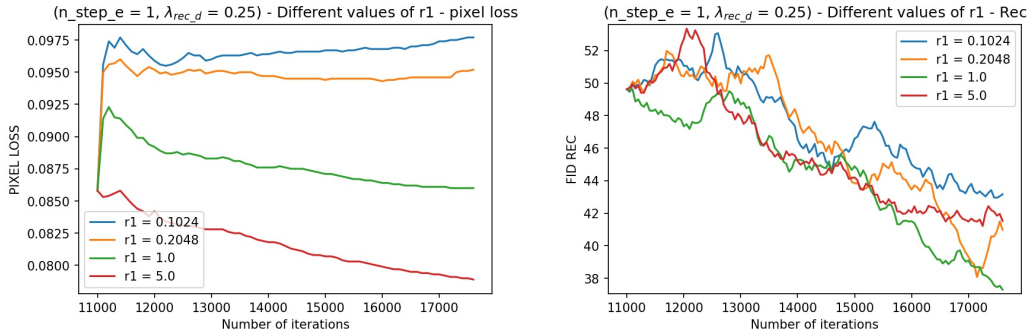

Fig. 8. **Hyper-parameters search on the R1 regularization factor**. **Left**: effect of R1 on the pixel loss. Higher values of R1 result in a more pronounced downward trend. **Right**: effect of R1 on the reconstruction FID. The FID tell us that the trend is basically the same independently the $r1$ value.

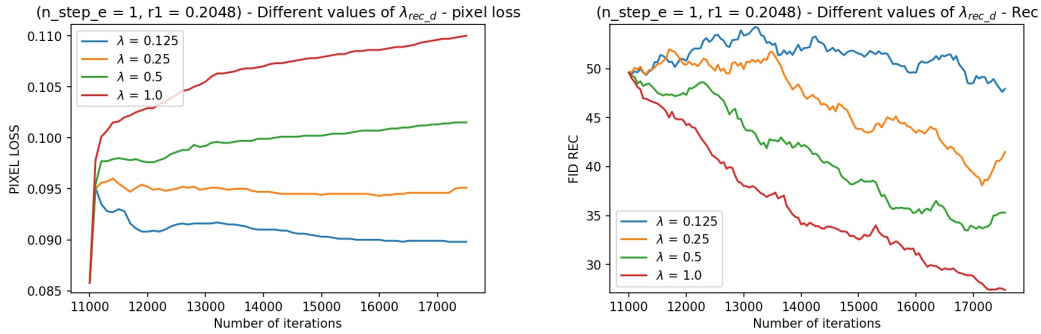

Fig. 9. **Hyper-parameters search on λ_{rec_d}**. **Left**: effect on the pixel loss. The lambda regularization factor we have selected may be a good value, we can definitely say that higher value are bad, due to an increasing value of the loss. **Right**: effect of λ_{rec_d} on the reconstruction FID.

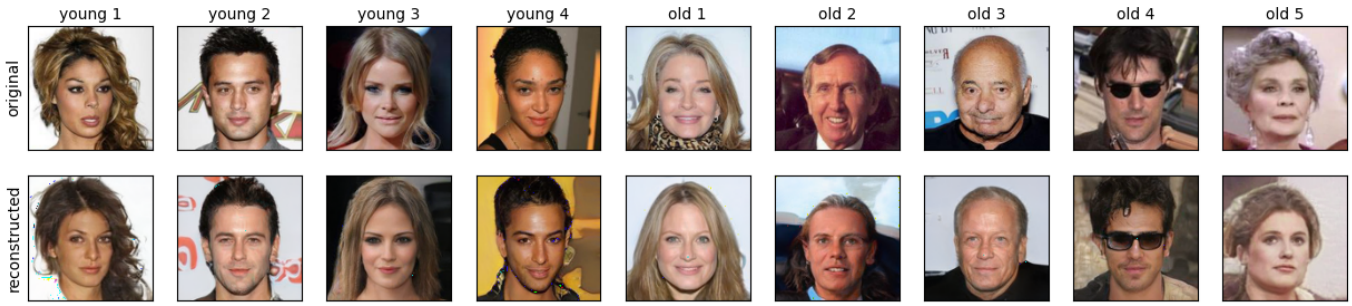

Fig. 10. **Selection of pictures for style mixing.** A selection of pictures of young and old people we use for our experiments in mixing styles. The first row shows the original image. The second row shows the image as reconstructed by the AE-StyleGAN.

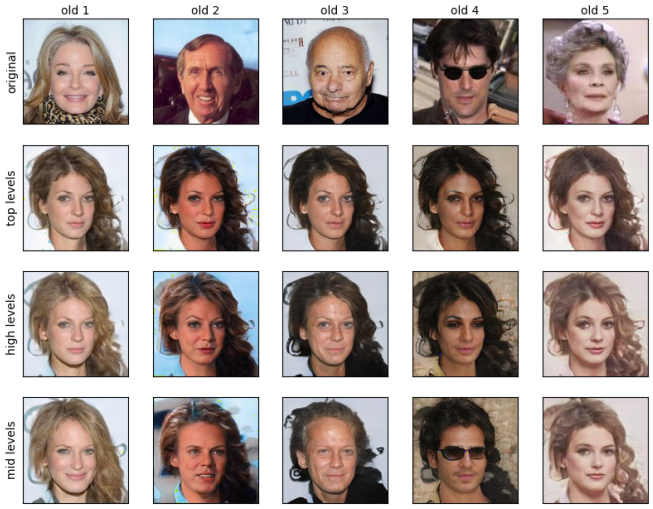

Fig. 11. **Style mixing.** Mixing the latent representation of "young 1" with each of the images of old people from figure 10 at different levels. "top levels" refers to taking the style of "young 1" at levels $1-8$ and the mixing style of each old person at levels $9-12$. For "high levels" we take the style of "young 1" at levels $1-6$ and old for $7-12$. Finally "mid levels" means "young 1" for style levels $1-4$ and old for $5-12$.

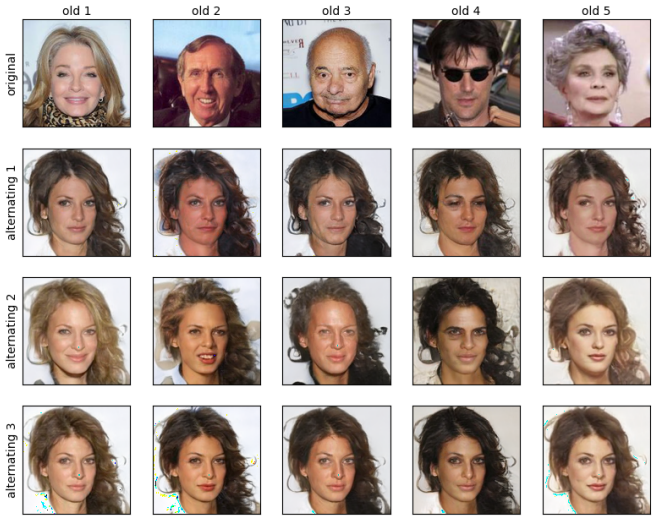

Fig. 12. **Mixing styles at alternating levels.** We mix the style of "young 1" with the style of pictures with old people at alternating levels. Coarse styles are always taken from image one. For "alternating 1" we take the style of the old person at levels $6, 8, 10, 12$ and the style of "young 1" everywhere else. "alternating 2" means taking the old person image's style at levels $5, 7, 9, 11$ and "young 1" everywhere else. Finally with "alternating 3" we take the mixing style at levels 9 and 11 and "young 1" everywhere else.

Figure 16 shows the effects of our filter after 150 iterations while figure 17 highlights how the visual appearance of the results decreases during the second half of the training process. When the AE-StyleGAN is able to properly reconstruct an image, then the filters produce convincing results. On the other side, artifacts of the reconstruction process are transferred to the result images. Often, the filter application results in unexpected changes with respect to the original image, e.g. gender swap, an issue that stems from the correlation of attributes in the CelebA dataset. Indeed, *young* and *gender*

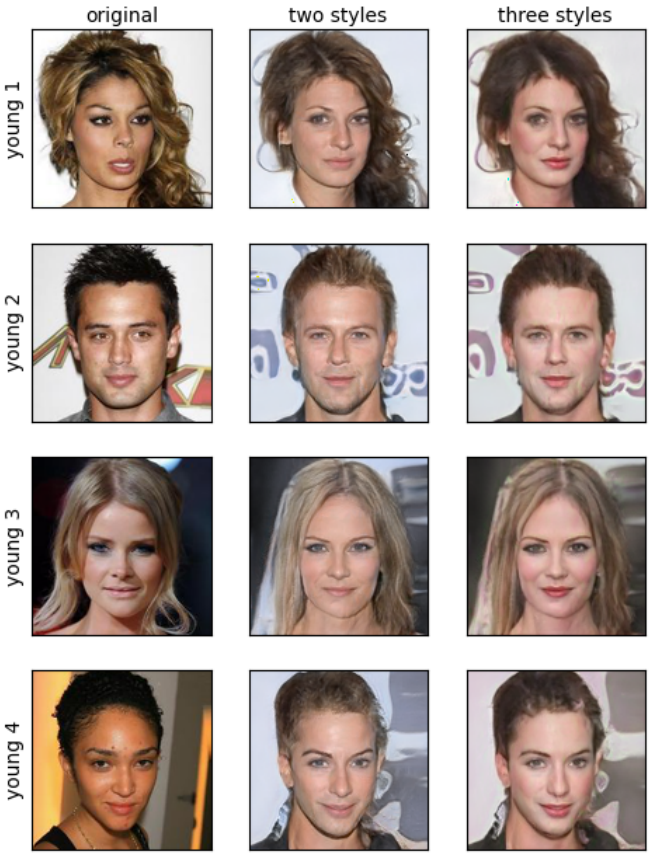

Fig. 13. **Mixing styles of several images.** Mixing the styles of 4 images from the young dataset with multiple images from the old dataset. For "two styles" we take the style of "old 1" at levels 9 and 11 while "old 2" is used at level 9 and "young 1" everywhere else. For "three styles" we use "old 1" at level 10, "old 2" at level 11 and "old 3" at levels 9 and 12.

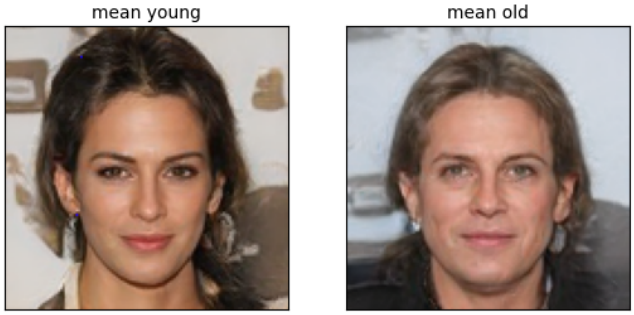

Fig. 14. **Means of young and old latents.** The left image shows what the synthesis network generates when taking the euclidean mean of the 4 young samples (left) and the 5 old samples of the pictures in figure 10.

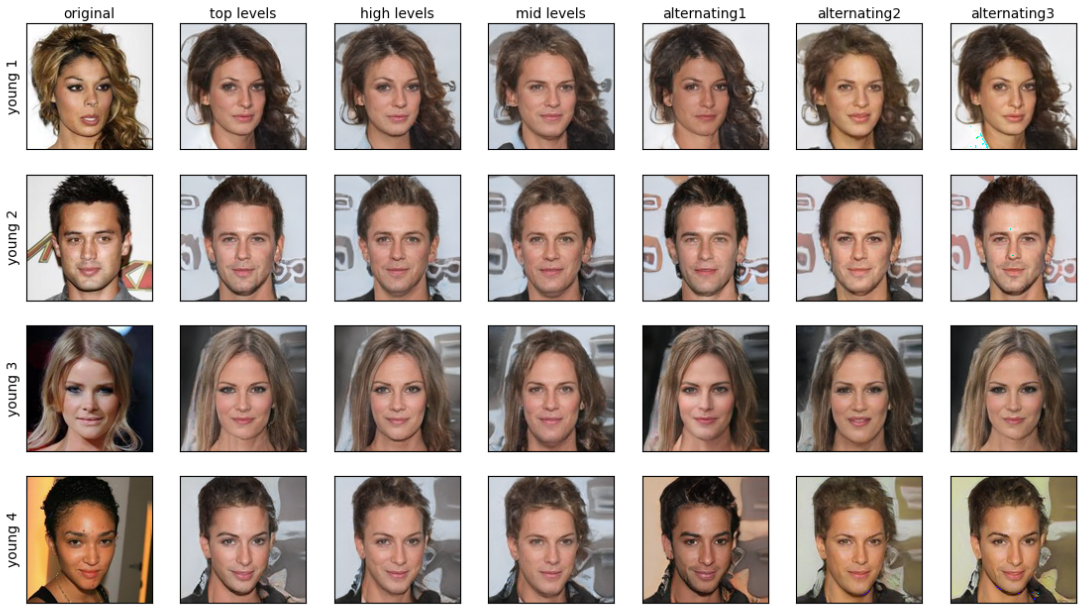

Fig. 15. **Mixing the domain's mean.** Mixing the input images of the young people from figure 10 with the mean old person (figure 14) at the different style levels presented in figures 11 and 12.

Fig. 16. **Results of Cycle-StyleGAN** after 150 tuning iterations. The grid on the left shows the result of a *young-to-old* filter while the results of an *old-to-young* filter are reported on the right. Overall, the performance are quite convincing even though the grids both contain reconstruction artifacts.

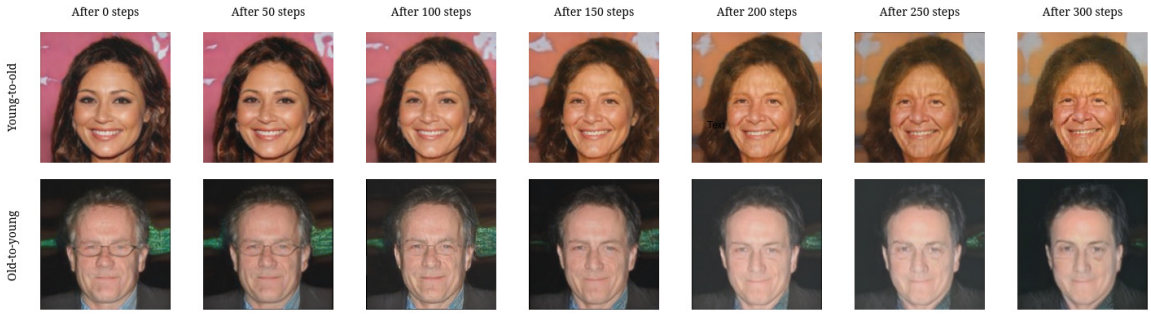

Fig. 17. **Progression of Cycle-StyleGAN results** versus increasing number of iterations. After 150 iterations, faces start to become more noisy. Wrinkles in the first row become too exaggerated while some weird marks start appearing around the eyes of the man in the second row.

	Iterations						
Target	0	50	100	150	200	250	300
Young	32.30	**27.79**	28.58	40.90	00.00	00.00	00.00
Old	53.68	48.55	39.37	**35.97**	00.00	00.00	00.00
Average	42.99	38.16	33.98	**38.44**	00.00	00.00	00.00

TABLE I
FID FOR CYCLE-STYLEGAN

are negatively correlated and the network is trained with a predominance of young and female samples over young and male samples.

Table I provides a quantitative point of view on the results showing the FID metric, separately for each target domain, versus increasing number of iterations. The FID confirms the findings provided by figure 16: the quality of the generated images shows an initially increasing trend before beginning to deteriorate rapidly. This behavior is entirely predictable. Although Cycle-StyleGAN and AE-StyleGAN share much of the architecture, they solve two different tasks. EMA updates ensure that task switching goes smoothly, without losing the knowledge learned during AE-StyleGAN training. However, as the number of iterations increases, the network begins to forget the initial task, worsening performance.

V. CONCLUSIONS AND FUTURE WORKS

The space of possible directions for future work is broad. Cycle-StyleGAN works well even though it suffers long training periods. CelebA is probably not the best dataset for a face ageing filter since it misses granular values for the age attribute. In this sense, Cross-Age Celebrity Dataset (CACD) [13] provides several images of the same people at different ages and it could be a better candidate to improve the identity preservation ability of the network. All the GAN-based methods tend to change the overall image, including the background and the elements unrelated to the specific filter. Segmentation masks could help the network focus on limited regions of the input image. Finally, all the results would have to be validated on higher resolution images.

Computational resources were provided by HPC@POLITO (http://www.hpc.polito.it).

REFERENCES

[1] T. T. Nguyen, Q. V. H. Nguyen, D. T. Nguyen, D. T. Nguyen, T. Huynh-The, S. Nahavandi, T. T. Nguyen, Q.-V. Pham, and C. M. Nguyen, "Deep learning for deepfakes creation and detection: A survey," 2019. [Online]. Available: https://arxiv.org/abs/1909.11573

[2] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Computer Vision (ICCV), 2017 IEEE International Conference on*, 2017.

[3] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," 2018. [Online]. Available: https://arxiv.org/abs/1812.04948

[4] L. Han, S. H. Musunuri, M. R. Min, R. Gao, Y. Tian, and D. Metaxas, "Ae-stylegan: Improved training of style-based auto-encoders," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2022, pp. 3134–3143.

[5] B. Hu, Z. Zheng, P. Liu, W. Yang, and M. Ren, "Unsupervised eyeglasses removal in the wild," *IEEE Transactions on Cybernetics*, 2020. [Online]. Available: https://arxiv.org/pdf/1909.06989.pdf

[6] I. Korshunova, W. Shi, J. Dambre, and L. Theis, "Fast face-swap using convolutional neural networks," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 3677–3685.

[7] Y. Z. Jie Chen, Junwen Bu. Aginggan: Age progression with cyclegan. [Online]. Available: https://github.com/jiechen2358/FaceAging-by-cycleGAN

[8] G. Antipov, M. Baccouche, and J.-L. Dugelay, "Face aging with conditional generative adversarial networks," 2017. [Online]. Available: https://arxiv.org/abs/1702.01983

[9] G. Antipov, M. Baccouche, S.-A. Berrani, and J.-L. Dugelay, "Apparent age estimation from face images combining general and children-specialized deep learning models," in *2016 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2016, pp. 801–809.

[10] Z. Liu, P. Luo, X. Wang, and X. Tang, "Deep learning face attributes in the wild," in *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

[11] K. Karkkainen and J. Joo, "Fairface: Face attribute dataset for balanced race, gender, and age for bias measurement and mitigation," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2021, pp. 1548–1558.

[12] R. Rothe, R. Timofte, and L. V. Gool, "Deep expectation of real and apparent age from a single image without facial landmarks," *International Journal of Computer Vision*, vol. 126, no. 2-4, pp. 144–157, 2018.

[13] B.-C. Chen, C.-S. Chen, and W. H. Hsu, "Cross-age reference coding for age-invariant face recognition and retrieval," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2014.