Image Colorization using GANs

Sara Buttau
sara.buttau@studenti.unipd.it

Salvatore Gatto
salvatore.gatto@studenti.unipd.it

Abstract

Image colorization is the process of adding plausible color information to grayscale pictures or videos. Colorization is a highly undetermined problem that does not have a unique solution since several combinations of colors can form a plausible output. Our work consists in the implementation of one of the most prominent strategies arisen in the last years which is based on the adoption of Generative Adversarial Networks (GANs).

Our experiments focus on transfer learning using GANs, by fine-tuning our model on different datasets to obtain several levels of colorization on specific subjects, such as flowers or faces. In addition, we also experimented with the pretraining of the generator. Lastly, we analyzed the results and stressed the differences in the performances.

1. Introduction

Image colorization is the process of estimating RGB colors from grayscale images or video frames to improve their aesthetic and perceptual quality [1]. In the last decades, this task was considered highly challenging due to the lack of information about the distribution of colors inside a grayscale image. More recently, the most encouraging results to colorize grayscale images were obtained by making the system learn the scene semantics of different scenarios, in order to recognize what are the typical colors related to objects, according to the changes in illumination, variations in viewpoints and occlusion.

The system we present in this paper is a Generative Adversarial Network (GAN). A GAN learns a loss to classify whether the output image is real or fake. In the meanwhile, it trains a generative model to minimize such a loss [2]. Because GANs learn a loss that adapts to data, they can be applied to a multitude of tasks that traditionally would require very different kinds of loss functions [2].

For this project, we experimented with transfer learning, by first training a GAN on a specific dataset (source domain), and then fine-tuning it on other three datasets (target domains), in order to improve the performance of our network. As a source domain, we trained our model on a small subset of *COCO* (Common Objects in Context)[16] dataset. As target domains, we chose three subsets of *CelebA*[18] , *ImageNet*[19] and *Oxford 102 Flowers*[17] datasets.
The training of the model on specific datasets such as CelebA and Oxford 102 Flowers was aimed at improving the accuracy of the network at colorizing pictures having one main semantic scene.

Figure 1. Example of a colorized image (left) as a result of our experiments. Image taken from CelebA dataset (original on the right).

Our GAN network was built following *pix2pix*' approach [2], using two losses, implementing a Patch Discriminator and using a U-Net-shaped generator. However, we further extended our experiments by pretraining the generator on the dataset before the actual GAN training.

After presenting the state of the art for image colorization in chapter 2, in chapter 3 we give a more in-depth explanation of the datasets we used and how we pre-processed the images. Chapter 4 presents our general methods, as well as our GAN architecture and our choices for loss functions and training parameters. Finally, we present the results of our experiments in chapter 5, both in a qualitative and quantitative way, and our conclusions in chapter 6.

2. Related Work

This section aims at introducing and briefly describing the state of the art around the image colorization task and transfer learning with Generative Adversarial Networks.

2.1. User-driven vs. data-driven approach

On the topic of image colorization, a first distinction must be made between user-guided edit propagation and data-driven automatic colorization [3]. The first one was brought to popularity by the work of Levin et al. [4], and it consists in having the user's input scribbles as a reference to generate matching colors. This technique can achieve impressive results but often requires intensive user interaction, as each differently colored image region must be explicitly indicated by the user, and this holds even for regions with little color uncertainty, such as green vegetation.
Data-driven colorization methods, instead, take a grayscale photo as input and colorize it in two different ways [3], one being non-parametric and the other being parametric. The non-parametric method defines one or more color reference images to be used as source data and to transfers color onto analogous regions of the input image [5]. Parametric methods, on the other hand, learn prediction functions from large datasets of color images at training time, posing the problem as either regression onto continuous color space or classification of quantized color values [5].

2.2. Transfer Learning with GANs

GANs have been vigorously studied in the last few years. Impressive results have been achieved with GANs for inpainting, future state prediction, image manipulation guided by user constraints, style transfer and super resolution [2]. An important work is the one by Isola et al. [2], who proposed a framework using Conditional Adversarial Networks as a general-purpose solution to image-to-image translation problems. The generator is a "U-Net" based architecture whose job is to generate plausible data. Instead, the discriminator is a "PatchGAN" that tries to classify each NxN image patch as real of fake.

As for what concerns domain adaptation by means of fine tuning a pretrained network, Wang et al. [6] were among the first to apply it to image generation with GANs. They evaluated several aspects such as impact of target domain size, relative distance between source and target domain and the initialization of conditional GANs. Most importantly, they proved that using knowledge from pretrained networks can shorten the convergence time and can significantly improve the quality of the generated images, especially when target data is limited [6].

Fregier et al. [7] proposed a transfer learning technique for GANs that freezes the low-level layers of both the critic and the generator of the original model. Given an autoencoder trained on a source dataset, they pass the target dataset through the encoder and use the encoded features to train a GAN, called a MindGAN, which composed with the decoder provides the transferred GAN [7]. They also formally proved a theorem ensuring the convergence of learning of the transferred GAN and demonstrated that their method enables a much faster training for GANs than other state of the art methods.

3. Dataset

For our datasets, we were inspired by Wang et al.[6]'s work. To cite them, "GANs typically require hundreds of thousands of training images to obtain convincing results", and "a GAN adapted from the pre-trained model requires roughly between two and five times fewer images to obtain a similar score than a GAN trained from scratch" [6]. Therefore, transfer learning allows us to perform our experiments using a limited amount of data, more appropriate for our limited computational resources.

In transfer learning, the choice of a source and a target domain is crucial. Our idea was to choose a source domain that contained a variety of objects and scenes, so that the model could be fine-tuned on more specific target domains. That is why we chose COCO (Common Objects in Context)[16] for our source domain. In particular, we trained our GAN on 8000 images from COCO dataset, and tested it on 2000 images. As for our target domains, we picked CelebA [18](CelebFaces Attributes), Oxford 102 Flowers [17] and ImageNet [19]. The choice for the first two was dictated by the fact that both faces and flowers were present in the source dataset, and therefore the network would find it easier to specialize on those semantic objects. The choice for the last one, instead, was to test fine-tuning on a more general dataset. In fact, like COCO, ImageNet contains a variety of objects and scenes, outdoor and indoor. Therefore, we were curious as to whether the behavior of our GAN on ImageNet would be positively influenced by the previous knowledge of similar scenes from COCO.
For every fine-tuning phase, we randomly picked 4000 images for training and 2000 images for testing. Table 1 summarizes our choices for datasets and their composition.

Dataset	Training set	Test set
COCO	8000	2000
CelebA	4000	2000
Oxford 102 Flowers	4000	2000
ImageNet	4000	2000

Table 1. Datasets used in our experiments.

All images underwent some preprocessing:

- we converted the color space from RGB to LAB
- we resized them to 224x224
- we applied some random augmentation: random vertical flip, random horizontal flip and random rotation from 0 to 180 degrees with probability 0.6.

We used the library fast.ai [9] to download our datasets.

4. Methods

Our aim is to build a model that is able to color images involving a fair number of heterogeneous scenarios (outdoor/indoor scenes, scenes with people, animals, landscapes, etc.) and more importantly, achieving realistic results.

The main reason behind our choice for GANs is their easy adaptability to data and tasks. In fact, GANs learn a loss that tries to classify the output image as real or fake while simultaneously training a generative model to minimize this loss. In other words, GANs learn a loss that adapts to data [2].

One of the drawbacks of GANs is their computational expenses [7]. Because there are two networks to train, they involve many parameters and can be extremely time-consuming. This is the main reason why we thought a transfer learning approach would be more suitable for this task. Our experiments branch over two parallel directions (see Figure 2). The background pipeline is the following:

- we train a GAN on a subset of the COCO dataset
- we fine-tune that model on
  - CelebA
  - Oxford 102 Flowers
  - ImageNet
- we test our networks respectively on each dataset.

In parallel, we compute the same experiments using another GAN model, with a generator that we first pretrained on COCO. We were inspired by the article by Shariatnia [11], which proposed a method using a pretrained generator, inspired by the paper on super resolution by Ledig et al.[12].

Hereon, we refer to the GAN model from scratch as GAN a), and to the GAN with a pretrained generator as GAN b).

4.1. Generator

Our generator's architecture is inspired by Isola et al. [2]. We built a U-Net generator from a pretrained Resnet18 body. We used fast.ai's handy functions to set the base architecture from the Resnet and shape it as a "Dynamic U-Net"[1]. Our generator receives one input, which is the L channel of the input image, and returns two outputs, i.e. the predictions for the a and b color channels. The input size is the one accepted by a Resnet model, 224x224.

[1]A Dynamic U-Net builds a U-Net from any backbone pretrained on ImageNet, automatically inferring the intermediate sizes [9]

4.2. Discriminator

We implemented Isola et al.'s [2] Patch discriminator, which only models high-frequency structure, by classifying each NxN patch in an image as real or fake. Such a discriminator assumes independence between pixels separated by more than a patch diameter, modelling the image as a Markov random field [2].

Our discriminator is modelled by stacking blocks of Conv-BatchNorm-LeakyReLU, and a sigmoid as the last layer's activation function. We initialized weights with values drawn from the normal distribution.

4.3. Loss function

The original GAN architecture was described by Ian Goodfellow et al. [13] as a generative model that simultaneously trains two models: a generative model G that captures the data distribution, and a discriminative model D that estimates the probability that a sample came from the training data rather than G. The training procedure for G is to maximize the probability of D making a mistake, while D tries to minimize it. Mathematically, the discriminator seeks to maximize the average of the log probability for real images and the log of the inverted probabilities of fake images.

$$\text{maximize } logD(x) + log(1–D(G(z)))$$

It is more commonly implemented as the standard average binary cross-entropy cost, also called log loss, that is minimized when training a standard binary classifier with a sigmoid output [13]. Following Isola et al.'s [2] approach, we used two losses: L1 loss for the generator and an adversarial (GAN) loss (BCEWithLogitsLoss()) for the GAN model. The two losses are combined in the following:

$$G^* = arg\min_G \max_D L_{GAN}(G, D) + \lambda L_{L1}(G)$$

4.4. Training Parameters

We trained GAN a) and b) models on COCO for 180 epochs, using a mini-batch size of 16 and a learning rate of 2e-4 for both generator and discriminator. We used Adam optimizer for both networks, with beta = 0.5 for the generator and beta = 0.999 for the critic (the choice for these hyperparameters was inspired by the state of the art). We used a lambda parameter equal to 100 as a coefficient for combining the two loss functions, following [11].

For every epoch during training, we printed the network's predictions on a batch of the validation set to monitor its performance on unseen data (see figure 3).
For branch b) of our experiments we pretrained our generator for 10 epochs on COCO, before the actual GAN training.

We fine-tuned our models for 50 epochs on every target dataset, using a batch size of 8 and same other hyperparameters as in the previous step.

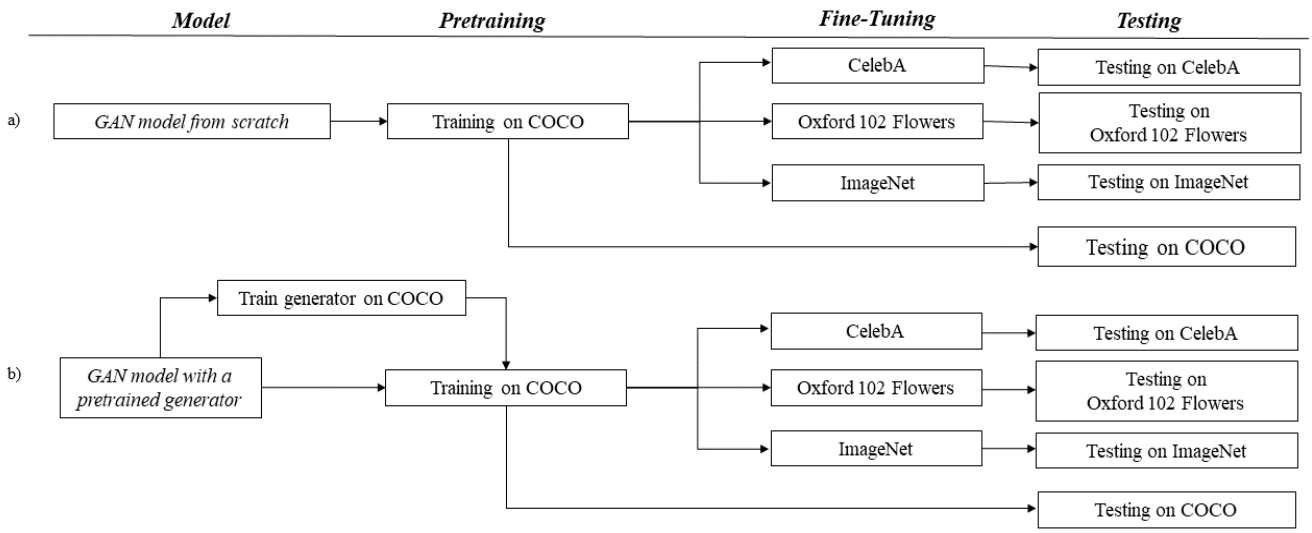

Figure 2. Our pipeline: we performed two parallel experiments, one with a GAN model trained from scratch and one with a GAN model with a pretrained generator.

Figure 3. Fine tuning of our GAN on Oxford 102 Flowers

5. Experiments and Results

As for the evaluation of our results, the state of the art still does not provide a clear agreed reference metric for GANs[6]. Different metrics can lead to different trade-offs, and different evaluations favor different models [14]. Several metrics have been used in the state of the art to evaluate the quality and the diversity in the generated data. Metrics like the Fréchet Inception Distance the Independent Wasserstein critic [6] measure the distance between two datasets.

Anwar et al. [1] give an exhaustive list of the problems related to the lack of good metrics for this task, as well as listing the state of the art's solutions in the matter. The mathematical metrics are general and may not provide an accurate performance of the algorithms. Nonetheless, we decided to validate our models with a quantitative metrics to have an idea of the quality of the images generated. The metrics typically used to assess colorization quality are commonly used mathematical metrics such as PSNR and SSIM [1]. We chose to compute the peak signal-to-noise ratio (PSNR) between the ground truth images and the images in our test set for every model we trained. For a qualitative analysis we submitted a perceptual test to some users, and measured how many of them were "fooled" by our fake images.

5.1. Quantitative analysis

The peak signal-to-noise-ratio (PSNR) is a metric used to evaluate the quality of image reconstruction. The formula is the following:

$$PSNR = 10log_{10}(\frac{(2^n-1)^2}{MSE})$$

A higher PSNR score means better image reconstruction. [15]

For every image in our test set we computed the PSNR value using the predicted image and the ground truth. For our comparisons we took the average PSNR value for each dataset.

Table 2 summarizes our results. All values range around 70, which is a good value indicating the quality of the image. For the GAN model from scratch (a), we obtained the largest PSNR value on CelebA, whereas for model (b) with

PSNR average values				
GAN	COCO	Oxford 102 Flowers	CelebA	ImageNet
(a)	72.68	70.24	**74.63**	72.37
(b)	71.17	**74.67**	69.93	72.38

Table 2. Each value is the average PSNR value for that specific dataset. Model (a) refers to the experiments with a GAN trained from scratch, whilst model (b) to the GAN with a pretrained generator. In bold: best PSNR values for each GAN model.

a pretrained generator, the largest value was obtained on Oxford 102 Flowers dataset.

5.2. Qualitative analysis

We submitted a perceptual test to a sample of 131 people, mostly friends and family, who we reached through word-of-mouth. To prevent the test from being biased, we selected both real and fake images, and asked our friends to indicate whether that image was real or fake to them, based on its colorization. We constrained our sample to a relatively small number of images (around 30) to make our test more quick and appealing to the users.

We summarize our results in tables 3 and 4.

% "real" answers to fake images (models)		
GAN from scratch	GAN pretrained	Total
53.63%	48.61%	**50.81%**

Table 3. Percentages of people that labelled our fake images as "real" in our perceptual test, based on the models. The last column is the overall percentage of both model types.

% "real" answers to fake images (datasets)			
COCO	CelebA	Oxford 102 Flowers	ImageNet
51.0%	52.43%	**56.28%**	43.53%

Table 4. Percentages of people that labelled our fake images as "real" in our perceptual test, based on the datasets. These percentages are referred to the images created by models of type a) and b).

Half the people in our sample labelled our 16 fake images as "real". There is not much difference between the network from scratch and the one with a pretrained generator, probably because our sample of images was carefully selected among the best results of the two networks. Overall, people seem to have been misled by the network from scratch more than the pretrained one (table 3) Among the datasets, Oxford 102 Flowers is the one with the best results: 56% of our sample was led to think the fake images were real. The worst results were achieved on the ImageNet dataset, as expected, since it has a wider variety of semantic scenes. In conclusion, our results are overall satisfying. We reported some images from our perceptual test in the appendix.

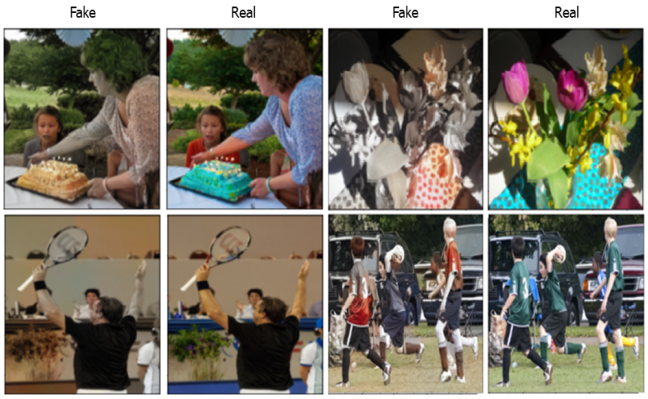

Figure 4. Failure cases: non uniform colors (ImageNet)

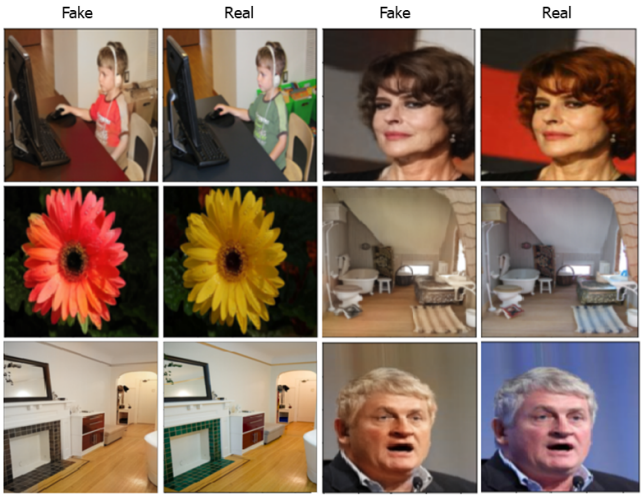

Figure 5. Different but plausible. Images from COCO, CelebA, Oxford 102 Flowers and ImageNet

5.3. Failure cases

There were also some failure cases. We reported some examples in figure 4. In particular, one main failure case was related to the non-uniformity of color, being unrepresentative of the semantics. Particularly, this was found more on COCO and ImageNet.

We also inserted in this section some examples of failure cases that we still consider valid, i.e. those cases that are differently-colored from the original but still plausible to a human eye (figure 5). This could result in a t-shirt being brown rather than blue, or in a flower being yellow rather than white. As mentioned already, our main goal was to develop a model that could create some plausible results, regardless of them being equivalent to the original ones. Overall, our model tends to have less saturated and more brownish results, probably due to the L1 loss contribution.

5.4. GAN from scratch vs. pretrained

The results of our experiments showed that the two networks produced similar results, although with some differences. As you can see in Figure 8, after the fine tuning on CelebA both networks generate promising results that are quite similar to the original images. The difference between the two GANs is instead more evident in the results obtained after the fine-tuning on ImageNet, as shown in Figure 9. In this case, the GAN (a) produced low saturated pictures with uniform colors, comparing to the pictures of the ground truth and of GAN(b).
Figure 6 shows the differences on Oxford 102 Flowers. In this case, GAN a) obtained results more similar to the ground truth, while b) still colored the images in a plausible way.

More in general, we can say that GAN b) has a better behavior on generating more natural colors, with a saturation that is more similar to the ground truth. Instead, whilst GAN a) still obtains some realistic results, its colors are in general less balanced in saturation.

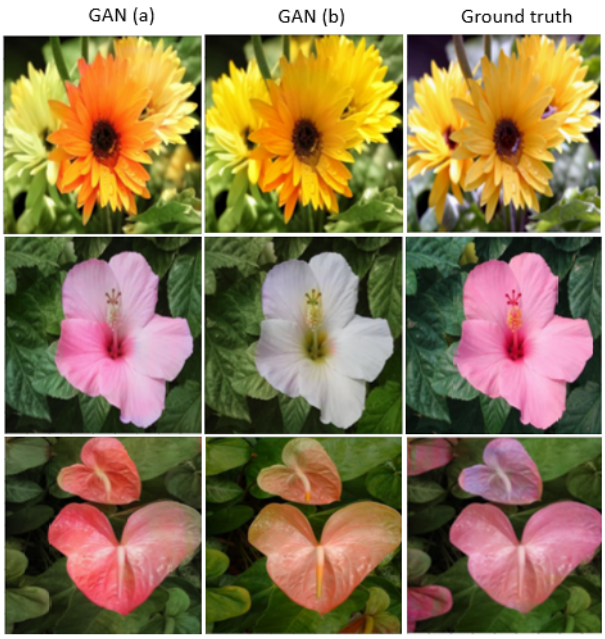

Figure 6. Comparison between GAN models a) and b) on Oxford 102 Flowers dataset with respect to the ground truth

6. Conclusions

We have used transfer learning to train and fine-tune two Generative Adversarial Networks for the task of image colorization. We implemented transfer learning in two ways: one for our generator, by pretraining it on the source dataset before training the network, and one for fine-tuning the model on the target domains.

We have used a subset of COCO dataset as a source domain, and experimented fine-tuning on other three different datasets as target domain: CelebA, Oxford 102 Flowers and ImageNet, chosen respectively to improve colorization on faces, flowers and general scenes and objects.

We analyzed our results in a quantitative way by computing the PSNR values on our test sets, and obtaining some good values averaging around 70. Moreover, we performed a qualitative perceptual test involving some users, and concluded that our fake-colorized images managed to fool half our sample of people.

Comparing our GAN models from scratch and pretrained, we have come to the conclusion that while both models generated plausible results, pretraining the generator was a successful experiment, as it led to a more realistic and balanced coloring. Failure cases are mostly found in the ImageNet fine-tuning, probably because it is a dataset as varied in subjects as COCO.

For further improvements, more powerful computational resources could be used to train our networks on larger datasets, as well as on new source or target domains. Also, it would be interesting to test our generated images as domain for the training of a classifier CNN, for further validation. Lastly, additional experiments could be done on the architecture of our GAN, perhaps trying a different pretrained network from the Resnet18 of our generator.

References

[1] Saeed Anwar, Muhammad Tahir, Chongyi Li, Ajmal Mian, Fahad Shahbaz Khan, Abdul Wahab Muzaffar, "Image Colorization: A Survey and Dataset," Computer Vision and Pattern Recognition (cs.CV), 2016.

[2] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, Alexei A. Efros ,"Image-to-Image Translation with Conditional Adversarial Networks," Computer Vision and Pattern Recognition , 2013.

[3] Richard Zhang, Jun-Yan Zhu, Phillip Isola, Xinyang Geng, Angela S. Lin, Tianhe Yu, Alexei A. Efros, "Real-Time User-Guided Image Colorization with Learned Deep Priors," Computer Vision and Pattern Recognition (cs.CV); Graphics (cs.GR), 2017.

[4] Anat Levin, "Colorization using Optimization," ACM Transactions on Graphics, 2004.

[5] Richard Zhang, Phillip Isola, Alexei A. Efros, "Colorful Image Colorization," Computer Vision and Pattern Recognition (cs.CV), 2016.

[6] Yaxing Wang, Chenshen Wu, Luis Herranz, Joost van de Weijer, Abel Gonzalez-Garcia, Bogdan Raducanu,"Transferring GANs: generating images from limited data," Computer Vision – ECCV 2018. ECCV 2018. Lecture Notes in Computer Science, vol 11210, 2018.

[7] Yael Fregier, Jean-Baptiste Gouray, "Mind2Mind: Transfer Learning for GANs," Geometric Science of Information, pp. (pp.851-859), 2021.

[8] Moein Shariatnia, "Colorizing black white images with U-Net and conditional GAN".Towards Data Science.

[9] Sylvain Gugger,Jeremy Howard, Deep Learning for Coders with fastai and PyTorch, O'Reilly Media, 2020.

[10] Leila Kiani, Masoud Saeed, Hossein Nezamabadipour, "Image Colorization Using a Deep Transfer Learning," 8th Iranian Joint Congress on Fuzzy and intelligent Systems (CFIS), pp. pp. 027-032, 2020 .

[11] Moein Shariatnia, "Colorizing black white images with U-Net and conditional GAN — A Tutorial", Towards Data Science,2020.

[12] Christian Ledig, Lucas Theis, Ferenc Husz´ar, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, Wenzhe Shi, "Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network. 2017 IEEE Conference on Computer Vision and Pattern Recognition" (CVPR), 105-114, 2017.

[13] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, Yoshua Bengio, "Generative adversarial networks", 2014.

[14] Lucas Theis, Aäron van den Oord, Matthias Bethge, A note on the evaluation of generative models, 2015.

[15] Sandra Treneska, Eftim Zdravevski, Ivan Miguel Pires, Petre Lameski and Sonja Gievska, GAN-Based Image Colorization for Self-Supervised Visual Feature Learning. Sensors 2022, 22, 1599

[16] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, Piotr Dollár, "Microsoft COCO: Common Objects in Context", 2014.

[17] M. Nilsback, A. Zisserman, Automated Flower Classification over a Large Number of Classes, 2008.

[18] Liu, Ziwei and Luo, Ping and Wang, Xiaogang and Tang, Xiaoou, "Deep Learning Face Attributes in the Wild.", 2015.

[19] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li; Li Fei-Fei, ImageNet: A Large-Scale Hierarchical Image Database. IEEE Computer Vision and Pattern Recognition (CVPR), 2009.

7. Appendix

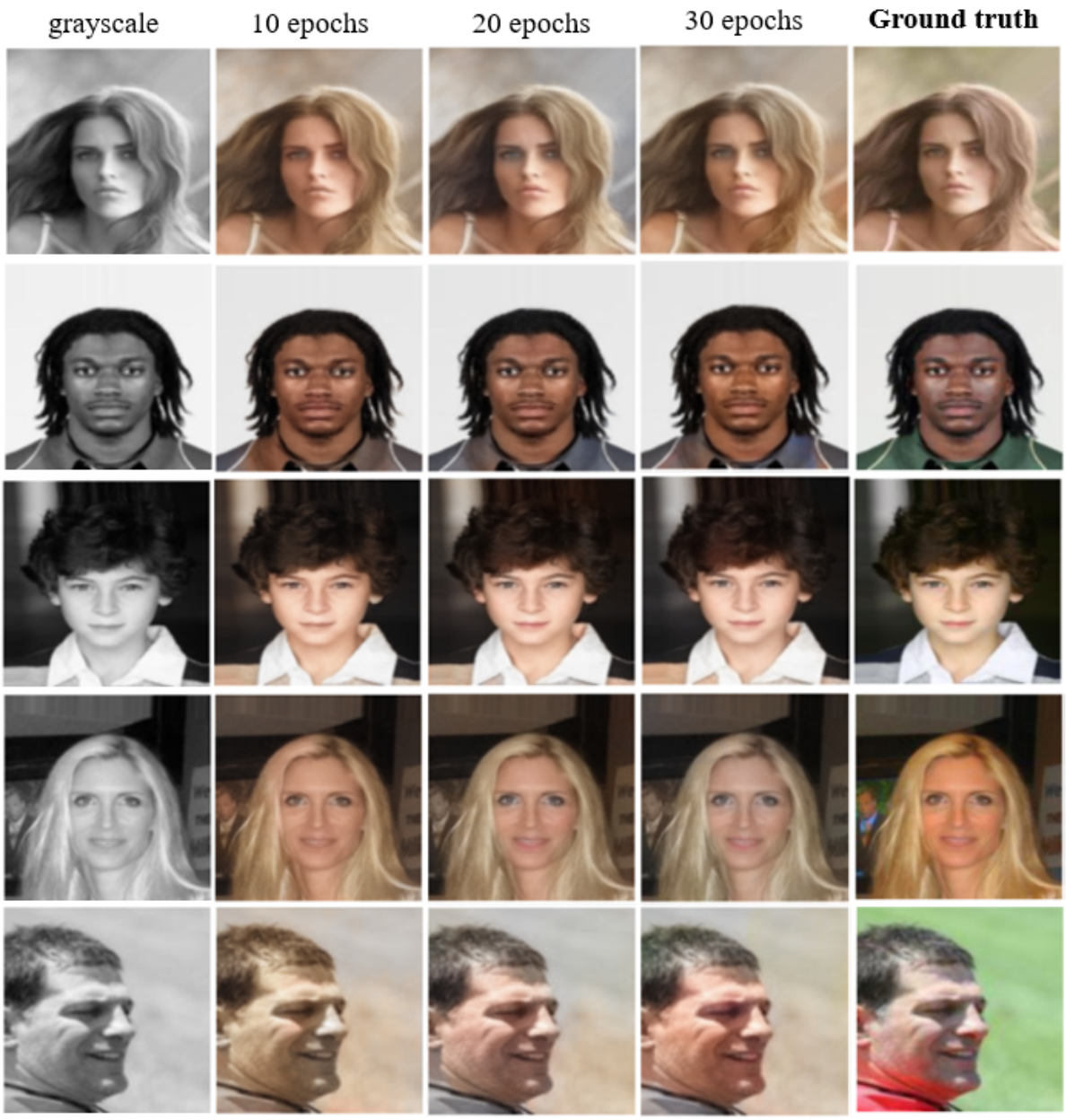

Figure 7. Fine tuning COCO-CelebA GAN from scratch 8bs

Figure 8. Recap results on CelebA dataset

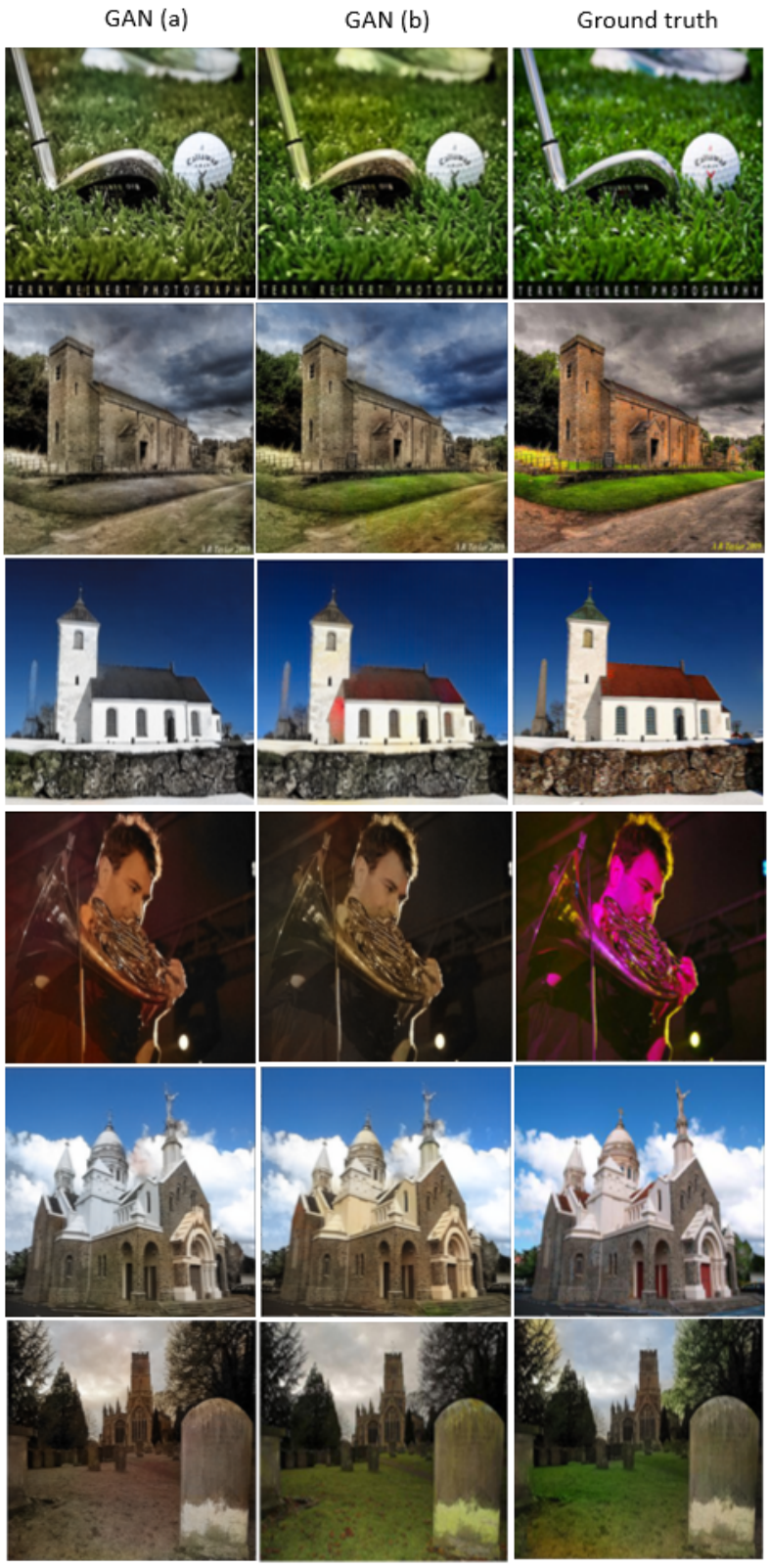

Figure 9. Recap results on ImageNET dataset