# Image Recoloring with Wasserstein Conditional Adversarial Networks

Lorenzo Valentini†, Gianmarco Nagaro Quiroz†

***Abstract*—Image colorization plays a crucial role in various computer vision tasks and has garnered significant attention in recent years. This paper attacks the problem of hallucinating a plausible color version of an input grayscale image. We present a novel approach for image colorization using a conditional Wasserstein Generative Adversarial Network with UNET architecture and patch loss discriminator.**

**We first implement the original work of Isola. It consists of a cGAN with a generator having a UNET structure and a discriminator using a combination of L1 and patch losses. One advantage of this model is that it is application-agnostic, and does not require hand-engineered loss functions.**

**In order to improve stability and convergence during the training process, and address mode collapse issues, we introduce a gradient penalty, transforming the GAN into a WGAN, mantaining at the same time the advantages of Isola's model. We study the impact of the new loss and of different sets of hyperparameters on the performance of the network.**

***Index Terms*—Wasserstein, Generative Adversarial Neural Network, Image Recoloring, UNET, Conditional Adversarial Neural Network, Deep Learning.**

## I. Introduction

Image recoloring takes care of revitalizing monochromatic visuals by infusing them with vivid colors. This has broad implications, ranging from restoring historical photographs to enhancing artistic creations. Image and video colorization can also assist other computer vision applications such as visual understanding [1] and object tracking [2]. The growing demand for efficient and reliable image recoloring methodologies makes this topic a compelling area of research with promising applications.

Previous attempts at image recoloring have often relied on traditional computer vision techniques or simple interpolation methods [3], [4], which have shown limited success in producing realistic and visually appealing colorization. More recent approaches make use of the ability of Convolutional Neural Networks (CNN) to encode image features at increasing level of abstraction along the network depth [5]. While the learning process is automatic, CNNs need a lot of manual effort for the design of effective loss functions.

Recent studies have explored the potential of GANs in this context, demonstrating their capability to generate high-quality and plausible colorized images [6]. GANs learn a loss that tries to classify if the output image is real or fake, while simultaneously training a generative model to minimize this loss.

†Department of Physics and Astronomy, University of Padova,
lorenzo.valentini.2@studenti.unipd.it,
gianmrco.nagaroquiroz@studenti.unipd.it

In this paper, we explore GANs in the conditional setting (as done in Isola et al. [6]) and with the addition of a Wasserstein Loss. Just as GANs learn a generative model of data, conditional GANs (cGANs) learn a conditional generative model. This makes cGANs suitable for image-to-image translation tasks, where we condition on an input image and generate a corresponding output image.

Wasserstein Generative Adversarial Networks (WGANs) improve upon traditional GANs by using the Wasserstein distance as the training objective, which offers a more reliable and continuous measure of similarity between generated and real data distributions. In GP-WGANs, this is achieved by using gradient penalties, an additional term in the discriminator's loss function. This term penalizes the magnitude of the gradients of the discriminator with respect to the input data. By constraining the gradients, the discriminator is encouraged to be smooth and prevents sharp changes in its output. This regularization helps stabilize the training process and prevents the generator and discriminator from diverging.

In synthesis, we present the results of our work focused on:

- **Implementation of the base model of Isola:** The base model of the original paper is implemented, and the various hyperparameters are tested.
- **Upgrade to gradient penalty WGAN:** The base model is extended to a conditional GP-WGAN, improving stability during training and plausibility of generated images.

The structure of this report is as follows. In Sec. II we describe in more detail the literature concerning the problem of style transfer. We describe the model of Isola and its implementation in Sec. III and the upgrade to WGAN in Sec. IV. Sec. V regards the data preprocessing. The results, with comparisons between the two networks and comparisons between choices of hyperparameters, are in Sec. VI. We conclude the report with some final remarks in Sec. VII.

## II. Related Work

**Previous work on image colorization.** Colorization algorithms differ primarily in how they acquire and process data to establish a correspondence between grayscale and color. Non-parametric methods typically begin with an input grayscale image and define one or more color reference images. These references can be provided by a user or retrieved automatically and serve as the source data. Subsequently, employing the Image Analogies framework [7], color is transferred from analogous regions of the reference images [8]–[10] onto the input image. On the other hand, parametric methods learn

prediction functions using extensive datasets of color images during training. They approach the problem by either regression onto a continuous color space [11], [12] or classification of quantized color values [13].

**Deep Learning general approaches for image colorization.** The most prominent work on fully automatic image colorization is deep learning based approaches that do not require any user guidance [5], [11], [14]–[16]. Cheng et al. [11] propose the first deep neural network model for fully automatic image colorization. Van den Oord et al. [17] developed Pixel Recurrent Neural Networks. Larsson et al. [14], Iizuka et al. [16], Zhang et al. [5] developed different CNN based methods. However, designing loss functions that effectively guide CNNs to produce desired outcomes, such as sharp and realistic images, remains an ongoing challenge that typically necessitates expert knowledge. The adaptability of GANs in learning loss functions instead allows them to be employed in a wide range of tasks that typically necessitate diverse types of loss functions.

**GANs for image transformations.** Several previous papers have extensively studied GANs. Most papers solely employed GANs in an unconditional manner, relying on additional terms like L2 regression to enforce conditioning based on the input. Remarkable results have been achieved in various applications such as inpainting [18], image manipulation guided by user constraints [19], style transfer [20]. Our work is instead based on the approach of Isola et al. [6], that uses GANs in an application-agnostic fashion. This key distinction significantly simplifies our setup compared to the majority of existing approaches.

**Loss functions for image colorization.** Structured losses are commonly approached by per pixel classification or regression methods [21], [22]. These approaches treat the images as unstructured, assuming conditional independence between output pixels. In contrast, cGANs employ structured losses, which consider the joint configuration of the output and penalize any discrepancies with the target. In particular in our work we make use of the a convolutional "PatchGAN" classifier [20], which only penalizes structure at the scale of image patches.

A substantial body of literature has explored many different structured losses [5], [14], [16], [23], [24]. The unique aspect of conditional GANs lies in their ability to learn the loss function by themselves, penalizing any form of structure mismatch between the output and the desired target. Traditional cGANs, like Isola et al. [6], make use of the Jehnsen Shannon divergence, for computing the loss. The JS divergence is conceptually quite clear, but it causes problems such as vanishing gradients or an unstable training process. The WGAN solves these problems by formulating a GAN loss based on the Wasserstein-1 or "earth mover's" distance, defined as the shortest average distance required to move the probability mass from one distribution to another.

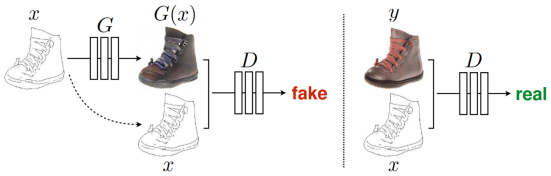


Fig. 1: Structure schema of a conditional GAN for image recoloring.

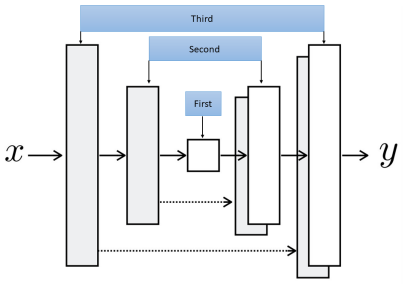


Fig. 2: Structure of the generator

## III. Implementation of Isola's architecture

**Conditional GANs.** Generative Adversarial Networks are models that have the ability to generate new data by learning a mapping from a random noise vector $z$ to an output image $y$, represented as $G : z \longrightarrow y$. On the other hand, conditional GANs take into account an observed image $x$ to generate the output image $y$, denoted as $G : x \longrightarrow y$. The primary objective in training the generator $G$ is to produce outputs that are indistinguishable from "real" images, as determined by a discriminator $D$ that has been adversarially trained. The discriminator's role is to detect and classify the generator's "fakes" as accurately as possible. A visual representation of this training process can be found in figure 1.

In this section we review the architecture of Isola et al. and define the model we use to perform the image colorization.

**Generator: U-Net**

The architecture chosen for the generator is a U-Net autoencoder. This structure allows for information to be efficiently propagated not only between adjacent layers, but also between layers located at the same depth in the network (figure 2). For example the last layer of the decoder now has the information provided not only by the previous layer of the decoder, but also by the first layer of the encoder (that is located at its same depth in the network). This information is crucial to build a more precise output.

The gray-scale image is encoded through a series of convolutional layers until it reaches a bottleneck layer, where sequential convolutions are performed. The decoder mirrors the encoder architecture and ultimately produces a 2-channel output. This U-Net architecture offers a structured approach to information propagation and has proven effective in image processing applications.

**Discriminator**

The design of our discriminator architecture (figure 3) is

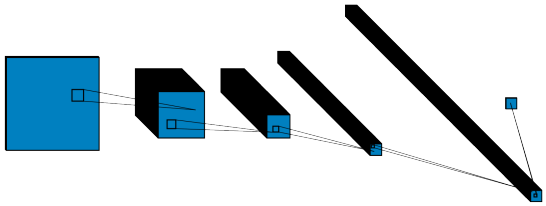

Fig. 3: Structure of the discriminator

quite straightforward. We construct the model by arranging Conv-BatchNorm-LeakyReLU. It's worth noting that the first and last blocks in the model do not employ normalization, and the activation function is absent in the last block since it is incorporated within the loss function we intend to utilize.

**Loss.** In general the loss for a conditional GAN is the following.

$$L_{GAN}(G,D) = \mathbb{E}_{x,y}[\log D(x,y)] + \\ + \mathbb{E}_x[\log(1 - D(x, G(x)))] \tag{1}$$

Prior studies have discovered the efficacy of augmenting the GAN loss by incorporating a constraint that ensures proximity to the input. In our approach, we employ an L1 loss for this purpose.

$$L_{L1}(G) = \mathbb{E}_{x,y}[\|y - G(x)\|_1] \tag{2}$$

The complete losses used in the network are therefore the following:

$$\begin{cases} L(D) = \mathbb{E}_{x,y}[\log D(x,y)] + \\ \qquad + \mathbb{E}_x[\log(1 - D(x, G(x)))] \\ L(G) = \mathbb{E}_{x,y}[\log D(x,y)] + \\ \qquad + \mathbb{E}_x[\log(1 - D(x, G(x)))] + \\ \qquad + \mathbb{E}_{x,y}[\|y - G(x)\|_1] \end{cases} \tag{3}$$

The L1 loss 2 is widely recognized for producing blurred outcomes in image generation tasks. While this loss fails to promote high-frequency sharpness, it often effectively captures low-frequency components. In scenarios where low-frequency correctness is already adequately addressed, there is no need to develop an entirely new framework. The L1 loss alone can suffice.

This realization motivates us to limit the GAN discriminator's focus to modeling high-frequency structures, while relying on an L1 term to ensure low-frequency correctness. To capture high-frequency details, it becomes adequate to concentrate on the structural characteristics present in local image patches. Hence, we design a discriminator architecture that penalizes structure solely at the patch scale. This discriminator's objective is to classify whether each $N \times N$ patch within an image is real or fake. By applying this discriminator convolutionally across the image and averaging all responses, we obtain the ultimate output of D.

## IV. WGAN IMPLEMENTATION

The conventional GAN loss involves the discriminator generating a probability indicating the authenticity of its input (whether it is real or generated). This loss is formulated as the cross entropy between the discriminator's output and the correct answer: 0 for real inputs and 1 for fakes. The primary objective is to approximate the real data distribution, $P_r$, using the generated distribution, $P_g$. By employing cross-entropy loss, we effectively optimize the Jensen-Shannon (JS) divergence between these two distributions. A smaller JS divergence indicates a higher level of similarity between the probability distributions.

The JS divergence encounters certain limitations. For instance, its gradient diminishes rapidly when dealing with highly dissimilar distributions and approaches zero when there is no overlap between the distributions. On the other hand, the WGAN (Wasserstein Generative Adversarial Network) adopts a different approach by utilizing the Wasserstein-1 distance, also known as the "earth mover's" distance, to formulate the GAN loss. This distance represents the average shortest distance required to move the probability mass from one distribution to another. Notably, the Wasserstein distance maintains a gradient even for distributions that have no overlap whatsoever. As the distributions become increasingly dissimilar, the Wasserstein distance smoothly increases. Mathematically, this distance can be defined as follows:

$$W(P_r, P_g) = \frac{1}{K} \sup_{\|f\|_L \leq K} [\mathbb{E}_y[f(y)] - \mathbb{E}_x[f(G(x))]] \tag{4}$$

The supremum in this context is computed over functions that exhibit K-Lipschitz continuity, a property indicating that the magnitude of their gradient is always less than or equal to K, to put it simply. Instead of hard clipping, however, we can penalize the gradient to impose a soft constraint on its length. This gradient penalty can be written as:

$$(|\nabla_{\hat{y}} D(\hat{y})| - 1)^2 \tag{5}$$

With the gradient is evaluated at $\hat{y}$, which represents a randomly weighted average between a real sample and a generated sample.

$$y = \epsilon y + (1 - \epsilon) G(x) \tag{6}$$

With $\epsilon$ selected randomly between 0 and 1.

To obtain the best discriminator, we minimize the Wasserstein distance while simultaneously constraining the gradients using a penalty term. Consequently, the resulting loss function for the discriminator is obtained as the sum of the standard GAN loss pluss the gradient penalty contribution.

$$L_{GP}(D) = \gamma(\|\nabla \hat{y} D(\hat{y})\| - 1)^2 \tag{7}$$

The complete losses used in the network are therefore the following:

$$\begin{cases} L(D) = \mathbb{E}_{x,y}[\log D(x,y)] + \\ \qquad + \mathbb{E}_x[\log(1 - D(x, G(x)))] \\ \qquad + \gamma(\|\nabla \hat{y} D(\hat{y})\| - 1)^2 \\ L(G) = \mathbb{E}_{x,y}[\log D(x,y)] + \\ \qquad + \mathbb{E}_x[\log(1 - D(x, G(x)))] + \\ \qquad + \mathbb{E}_{x,y}[\|y - G(x)\|_1] \end{cases} \tag{8}$$

When expressed in this manner, the concept of loss appears rather straightforward: the discriminator's objective is to produce outputs that are as small as possible for real samples and as large as possible for fake samples. Meanwhile, the gradient penalty term prevents the weights from becoming excessively large. It might seem peculiar to have a loss function without a strict lower bound, but in practice, the combination of generator and the discriminator ensures that the terms remain roughly balanced.

## V. Dataset and Preprossessing

The dataset selected for our specific task consists of a subset extracted from the Imagenet dataset. It encompasses approximately 10,000 training images and 3,000 validation ones, each representing one of 1,000 distinct classes. This ensures a diverse collection of images, facilitating a robust learning process for the network.

Fig. 4: Imagenet samples

All images are resized to the desired input size, in this case 256x256 pixels.

Afterwards, a grayscale image has to be extracted from the colored image. The best way to approach this problem is to convert the images to the Lab color scheme. The Lab color scheme enables the network to consider the L channel as the grayscale version of the image while the a and b channel represent red-green and yellow-blue respectively.

The Unet generator is fed the L channel, the grayscale version of the image, and asked to reproduce the two other channels, a and b. They are combined to the input in order to obtained the colorized image. The full 3-channel image is given to the discriminator (always in the Lab color scheme), so that it already contains gray-scale input and colored output at the same time. The advantage, with respect to the RGB color scheme, is in the fact that we reduce the dimensionality of outputs and inputs of generator and discriminator.

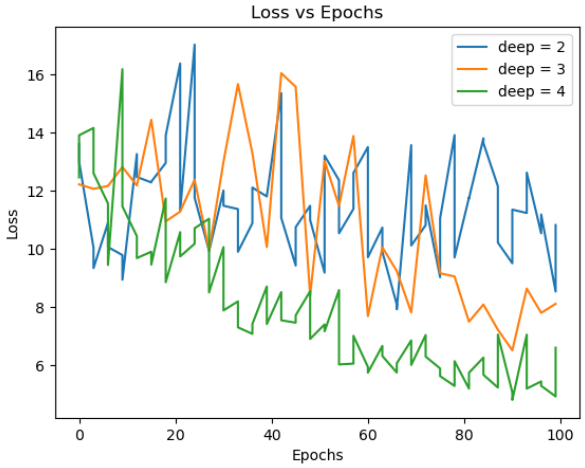


Fig. 5: Comparison between Isola's models with different number of encoding/decoding layers.

## VI. Results

In this section, we present the results of our experiments on colorizing black and white images using neural networks. We initially employed a standard U-net architecture for the task and trained models with a network depth of 2, 3, and 4 encoding/decoding layers. Subsequently, we observed the need to add extra layers around the bottleneck to improve the model's performance. We then evaluated the colorization performance of the modified U-net architectures with depths of 2, 3, and 4. Finally, we investigated the impact of different Wgan weights on the model's performance.

### A. U-net with Varying Depths

We first present the results of our re-implementation of the original model of Isola, with depths of 2, 3, and 4 layers. Figure 5 illustrates the training progress of these models over epochs. Each line in the graph represents a specific model configuration.

As shown in figure 5, the loss decreases gradually over the epochs for all three model configurations. However, we observe an erratic behaviour during training: the loss does not gradually decrease but instead jumps irregularly between a different values. But as seen in figure 5, higher depth results in higher stability, making possible to achieve stable training with bigger layers. So, the next attempt includes a triple layer before and after the bottleneck that ensures more parameters are added to our model, making it better at handling complex patters that are presented on the dataset.

Even when stable results are obtained, the generated images still have noticeable flaws. The resulting images contain some bright random coloring at the center because of a bad strategy adapted at training, as seen in figure 7. The GAN should be

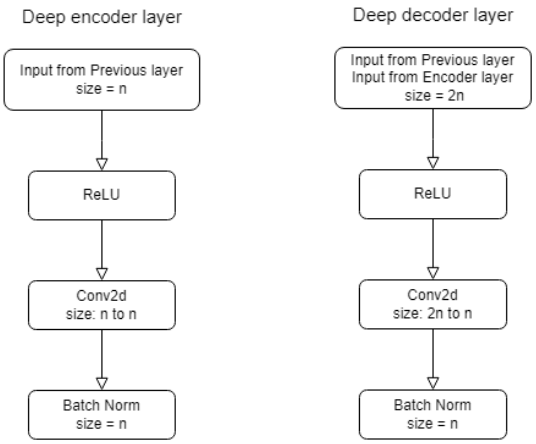


Fig. 6: Schema of the fundamental blocks used in the autoencoder.


Fig. 7: Samples of colorized images using Isola's model.

able to penalize this behavior. To fix this problem, a WGAN was implemented.

### *B. Modified U-net with Varying Depths*

Next, we present the results of the new U-Net architecture: with Wasserstein loss and with thicker encoding/decoding layers. The cases studied have depths of 2, 3, and 4 layers.

The U-net of depth=2 still exhibits some noisy behaviour when converging. This is shown apart from the other two models in figure 8.

As depicted in figure 9, the modified U-net architectures consistently outperform their standard counterparts. The loss values decrease more rapidly and reach lower levels compared to the previous set of models. Depths 3 and 4 behave in a similar way, with depth = 4 slightly outperforming its counterpart on the late epochs. For this reason we decided to use this number of layers for further testing.

These results demonstrate that modifying the U-net architecture by increasing the thickness of layers improves the

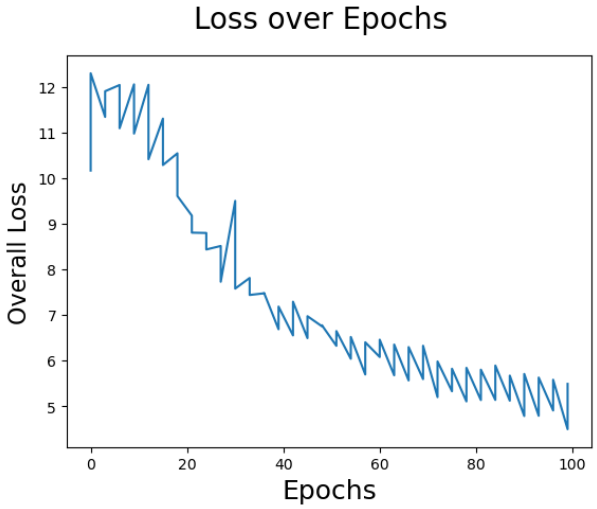


Fig. 8: Loss values for the WGAN with 2 encoding/decoding layers.

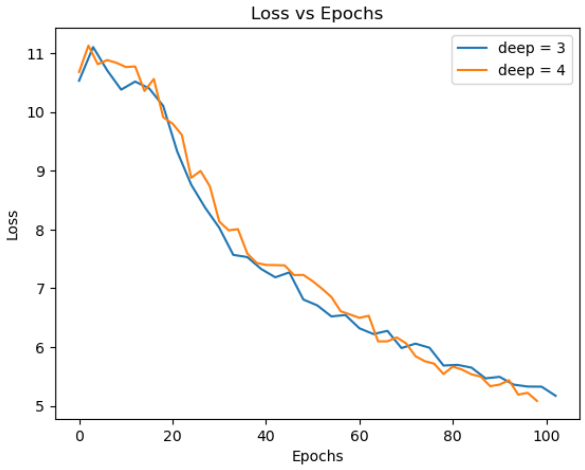


Fig. 9: Loss values for the WGAN with 3 and 4 encoding/decoding layers.

colorization performance, and deeper modified U-net architectures yield better results. As shown in figure 9, the fake images are now more prone to be recognized as true RBG images, as seen in figure 10.

### *C. Impact of WGAN Weight*

In addition to modifying the U-net architecture, we investigated the impact of different WGAN weights on the colorization performance. We applied Wgan weights of 0.1 and 0.3 to the new U-net architecture. Figure 11 displays the training progress for these models.

As shown in figure 11, these studies don't show noticeable differences. Drawing relevant results from is not possible at the moment but further testing with higher weight values would be enriching.

While we are not able to extract conclusions from the loss graphs, we are still able see differences in the images generated by the three different levels of GP loss contributions,


Fig. 10: Samples of colorized images using the WGAN model.

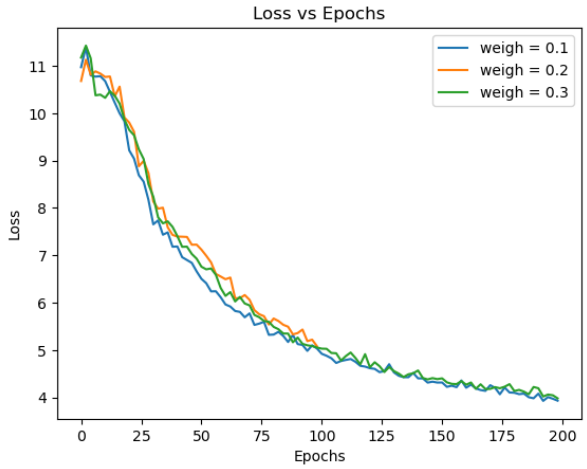


Fig. 11: Loss timeseries for the WGAN model, testing different coefficient for the GP loss contribution to the discriminator loss.

with apparently higher quality in the case of the strongest contribution.

## VII. Concluding Remarks

A Generative Adversarial Network was implemented to address the task of colorizing gray-scale images. The GAN utilizes a U-Net architecture comprising an encoder and decoder. Given a gray-scale image as input, the network generates the missing channels (a and b) of the Lab-format image. Initially, the model was developed as a re-implementation of the work by Isola et al. [6]. Subsequently, it was enhanced with the incorporation of a gradient penalty Wasserstein loss. Training and testing of the network were conducted on a subset of the Imagenet dataset.

The outcomes presented in this research demonstrate the significant potential of Conditional Adversarial Networks for diverse image-to-image translation tasks. Notably, the advantage of these networks lies in their versatile loss function, which can be applied to a wide range of image transformation tasks.

Possible future developments may consist in repeating for the WGAN the tests done by Isola in the original model: exploring different patch sizes for the Patch-loss, employing alternative generator losses instead of L1, and leveraging the GAN framework for tasks beyond image recoloring. Additionally, conducting an extensive grid search to identify the optimal coefficient for the contribution of the gradient penalty loss to the discriminator loss would be of interest.

The presence of bright color spots on images generated with our re-implementation of Isola's model was not detectable from the values of the loss, that were similar to the WGAN one. It would be useful to find a measure that reflects that kind of wrong behavior, in order to measure it objectively.

During the first stages of testing we encountered heavy problems when the network started converging to a solution that did not involve coloring the image correctly but minimized the loss by turning all the image into a sepia tone. We corrected this behavior by choosing deeper layers in the autoencoder.

A relevant thing that this project made us learn was to make the neural network more modular. We didn't expect that we would need to change the architecture of the network a lot of times in order to find an effective structure. A modular model facilitates the process.

We also wanted to have different tests and do a kind of grid-search with the new parameters that we were introducing to the model. This was the network depth that spawned new layers in the network and the WGAN weight. One difficulty was the training times, that were strongly tied to the lack of computing power and that is why we were careful on the choices of tests we would run.

## References

[1] G. Larsson, M. Maire, and G. Shakhnarovich, "Colorization as a proxy task for visual understanding," *CVPR*, 2017.
[2] C. Vondrick, A. Shrivastava, A. Fathi, S. Guadarrama, and K. Murphy, "Tracking emerges by colorizing videos," *ECCV*, 2018.
[3] A. Levin, D. Lischinski, and Y. Weiss, "Colorization using optimization," *ACM Trans. Graph.*, 2004.
[4] Y. Qu, T. Wong, , and P. Heng, "Manga colorization," *ACM Trans. Graph*, 2006.
[5] R. Zhang, P. Isola, and A. A. Efros, "Colorful Image Colorization," *ECCV*, Mar. 2016.
[6] P. Isola, J. Z. T. Zhou, and A. A. Efros, "Image-to-Image Translation with Conditional Adversarial Networks," *CVPR*, Nov. 2016.
[7] *mage analogies*, 2001.
[8] T. W. et al., "mage analogies," *ACM TOG*, 2002.
[9] A. C. et al., "Semantic colorization with internet images," *ACM TOG*, 2011.
[10] R. G. et al., "mage colorization using similar images," *ACM*, 2012.
[11] Z. Cheng, Q. Yang, and B. Sheng, "Deep colorization," *ICCV*, 2015.
[12] A. Deshpande, J. Rock., and D. Forsyth, "Learning large-scale automatic image colorization," *ICCV*, 2015.
[13] G. Charpiat, M. Hofmann, and B. S. Ì̂olkopf., "Automatic image colorization via multimodal predictions," *ECCV*, 2008.
[14] G. Larsson, M. Maire, and G. Shakhnarovich, "Learning representations for automatic colorization," *ECCV*, 2016.
[15] A. Deshpande, J. Lu, M. Yeh, M. J. Chong, and D. A. Forsyth, "Learning diverse image colorization," *CVPR*, 2017.
[16] S. Iizuka, E. Simo-Serra, and H. Ishikawa, "Let there be Color!: Joint End-to-end Learning of Global and Local Image Priors for Automatic Image Colorization with Simultaneous Classification," *ACM Trans. Graph*, 2016.
[17] A. van den Oord, N. Kalchbrenner, and K. Kavukcuoglu, "Pixel Recurrent Neural Networks," *ICML*, 2016.
[18] J. D. T. D. D. Pathak, P. Krahenbuhl and A. A. Efros., "Context encoders: Feature learning by inpainting," *CVPR*, 2016.
[19] E. S. J.-Y. Zhu, P. Kr Ì̂ahenb Ì̂uhl and A. A. Efros, "Generative visual manipulation on the natural image manifold," *ECCV*, 2016.

[20] C. Li and M. Wand., "Precomputed real-time texture synthesis with markovian generative adversarial networks," *ECCV*, 2016.
[21] C. Li and M. Wand, "Combining markov random fields and convolutional neural networks for image synthesis," *CVPR*, 2016.
[22] I. K. K. M. L.-C. Chen, G. Papandreou and A. L. Yuille, "Semantic image segmentation with deep convolutional nets and fully connected crfs," *ICLR*, 2015.
[23] S. Xie and Z. Tu, "Holistically-nested edge detection," *ICCV*, 2015.
[24] E. S. . Long and T. Darrell, "Fully convolutional networks for semantic segmentation," *CVPR*, 2015.