Automatic Image Colorization with Adversarial and Classification Losses

Lorenzo Saccaro
lorenzo.saccaro@studenti.unipd.it

Abstract

The generation of colored images from grayscale ones is a challenging task that have seen numerous attempts to solve it in the last few years. In this work, I implement the pix2pix model introduced by Isola et al. [1] that consists of a conditional Generative Adversarial Network with a U-Net-based generator and a convolutional PatchGAN discriminator. Following the idea proposed by Vitoria et al. [2], I add a classification branch to the generator in order to extract semantic information from the image. By jointly training the model with an adversarial (traditional GAN and Wasserstein GAN with gradient penalty are compared in this work) and a classification loss term, there is an improvement in the generated images, both qualitative and quantitative as measured by different metrics. The results on the Places205 dataset are quite promising, and to improve some shortcomings of the model, a data augmentation technique based on the use of an object detector is proposed as a future work.

1. Introduction

Image colorization is a well known problem in computer vision that remains an open challenge, despite the progress that has been made in the last few years. The goal of this task is to automatically colorize a grayscale image: in particular, the ability to restore old black and white images is important in many fields, such as photography, cinema, advertising, etc. The problem of colorization is not trivial, since the color of a pixel is not uniquely determined by its grayscale value: in fact, it is necessary to reconstruct 2 or 3 channels (depending on the color space used) from a single one. Moreover, the colorization process is multimodal, since there are many *plausible* colorizations for a single grayscale image, that depends on the subject and the context of the image. To produce realistic and high quality images, the use of Generative Adversarial Networks (GANs), firstly introduced in [3], has become a key ingredient in many attempts to solve the problem of colorization.

In this work, I start by implementing the pix2pix model designed by Philip Isola et al. [1], which leverages a conditional GAN (cGAN) to perform different types of image-to-image translation tasks, among which colorization. I then show, both quantitatively and qualitatively, that by jointly training the model to extract semantic information from the image, there is an improvement in the generated images. In this case, I adopt a similar approach to the one proposed by Patricia Vitoria et al. [2].

The rest of the paper is organized as follows: in Section 2 I briefly review the related work on colorization, in Section 3 describe the dataset used and the preprocessing steps, in Section 4 I dive into the details of the implemented architecture and the training objective, in Section 5 the experiments conducted and the results obtained are showcased and discussed and finally in Section 6 I summarize the achievements of this work and propose further possible improvements. The code is available at https://github.com/lorenzo-saccaro/image-colorization-classification.

2. Related Work

There are many approaches to the problem of colorization: to cite just a few, early architectures were based on simple convolutional neural networks (CNNs) like in [4], while user-guided networks such as Scribbler [5] require user inputs (points, strike, scribbles) to guide the colorization process. More recently, more complex architectures have been proposed such as multi-path networks, that learn features at different levels, like [6], and an instance-aware colorization model [7] that leverages a fusion module that combines the features of a full image with an instance colorization network, which is trained on the detected object instances using an off-the-shelf object detector. A complete survey of the methods used for colorization can be found in [8].

As stated before, the work of [1] is the starting point of this paper: the authors propose a general-purpose solution to the problem of image-to-image translation, which is based on a cGAN that uses a U-Net-based generator and a convolutional PatchGAN discriminator. In the specific task of colorization, the grayscale image is fed to both the generator and the discriminator as side information. A clarifying schema of the pix2pix framework is shown in Figure 1.

In the work of [2], the authors propose a novel archi-

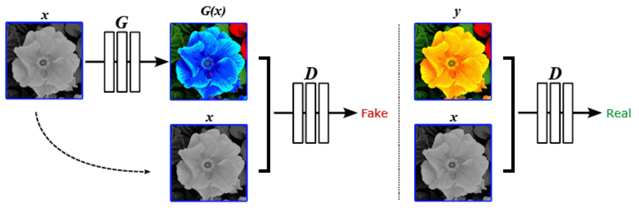

Figure 1. Schema of the pix2pix framework.

tecture called ChromaGAN, which leverages a pre-trained VGG-16 [9] on ImageNet [10] as the backbone of their generator. The model is jointly trained to perform realistic colorization through an adversarial loss and to extract a semantic interpretation of the scene by minimizing the Kullback-Leibler (KL) divergence between the class distributions of the generated and ground truth images. They show that the semantic information extracted by the generator can be used to improve the quality of the generated images.

3. Dataset

The chosen dataset for this work is the Places205 dataset [11]: it consists of 2.5 million images belonging to 205 scene or places categories, such as bedrooms, kitchens, forests, beaches and more. Due to constraint on time and computational resources, only a 10% subset of the dataset is used, which is available at [12]. Starting from a total of 238136 images, 162 are excluded for being grayscale, resulting in a total of 237974 images, which are then split as follows:

- Training set: 202278 images
- Validation set: 11899 images
- Test set: 23797 images

The validation set is used to monitor some metrics (see 4.3) during training, while also visually inspecting the generated images. The results presented in Section 5 are obtained on the test set.

Two main reasons motivate the choice of this dataset: it is fairly complex due to the great variety of scenes and the large number of object instances in many samples. Secondly, the dataset is labeled, which is a crucial requirement for the second part of this work (this is technically not mandatory since it could be possible to use a pre-trained model to extract class distributions, but for the Places205 dataset even state-of-the-art models have moderate accuracy [13]). There is some class imbalance, with some classes having three times more samples than others, but this is not a relevant problem in this case, since the classification is only a secondary task to improve the colorization process.

As far as the preprocessing is concerned, the images are resized to 256×256 and then converted to the CIE L*a*b* color space, where the L channel is the lightness (grayscale input) and the a* and b* channels contain the chrominance information (green-red and blue-yellow axis). This is a crucial step, adopted in the vast majority of the works on colorization, since it drastically reduces the complexity of the problem: starting from a single channel, it is necessary to reconstruct only two channels instead of three in the case of RGB (in case of an 8 bit image, this corresponds to reducing the complexity from 256^3 to 256^2). Moreover, the L*a*b* color space is perceptually uniform, which means that a given numerical change corresponds to a similar visual change, and it is based on the way human eyes perceive colors.

Finally, inputs are normalized to the $[-1, 1]$ range and horizontal flips are applied for data augmentation with a probability of 0.5 during training.

4. Method

4.1. Architecture

The initially implemented architecture consists of a PatchGAN discriminator and a U-Net autoencoder as the generator.

The U-Net autoencoder, originally introduced in [14] for biomedical image segmentation, has two main parts: the encoder and the decoder both composed of sequences of similar blocks. The encoder compresses the input grayscale image into a lower-dimensional representation, while the decoder expands this representation back into the color image. The U-Net architecture is chosen for its ability to preserve the details and structure of the input image during the encoding-decoding process. Going into more details, inside the encoder, each block contains a convolutional layer with a stride of 2, a batch normalization [15] layer, and a Leaky ReLU activation function [16]. For the decoder instead, the structure consists of a transposed convolutional layer, a batch normalization layer, and ReLU activation function [17]. The different choice of the activation function is due to the fact that Leaky ReLU allows for better managing the vanishing gradient problem in the encoder part. Both encoder and decoder are composed of 8 blocks, that progressively halve the size of the image from 256×256 down to a 1×1 feature vector and then up again to the original input size. This architecture is different from the original one, which makes the input pass to three convolutional blocks of the same dimension before applying a max pool or a transposed convolution to respectively reduce or increase the input size. The peculiarity of the U-Net is that it contains skip connections between layers of the same size. This is obtained by concatenating at each decoder's step the result of the whole process with the vector of features encoded up to that size. In the decoder branch, a dropout of 50% is applied for the first three blocks. The very last layer of the U-Net architecture uses $\tanh$ as an activation function to get a prediction in $(-1, 1)$, that emulates the input values. The whole architecture, shown in Figure 2, has around 54 million parameters.

The structure of the discriminator is similar to the encoder of the generator, and it is made of four blocks each composed of a convolutional layer with a stride of 2, a batch normalization layer and a Leaky ReLU activation function. The last layer is a simple convolution that returns a matrix of shape 30×30. Since the discriminator is quite shallow, each pixel's value of the final output is based only on its receptive field, which covers only a small portion of the image (i.e. a *patch*), and represents the classification of the corresponding patch. This means that instead of having a simple binary classifier, 900 values "vote" to determine if the sample is real or fake. The discriminator has about 2.5 million parameters: a schema of its architecture is shown in Figure 3.

In order to extract semantic information from the image, the architecture of the generator is modified as follows: starting from the third encoder block, a new branch consisting of another three encoder blocks is added. The features extracted by this branch are flattened and then follow two different paths: the first one is the classification head, which consists of two dense layers with 2048 and 1024 units respectively with ReLU activations and 50% dropout, and a final dense layer with 205 units corresponding to the number of classes in the dataset. The second path has two dense layers with 1024 and 512 units respectively with ReLU activations and its output is fused back into the main path before the first layer of the decoder by concatenating it to the encoder features. The modified architecture has about 97 million parameters.

4.2. Objective

In the adversarial setting, the objective is to make the generator G produce images that the discriminator D cannot distinguish from real ones. For a cGAN this can be formalized as follows:

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

where the generator and the discriminator play a min-max game described by:

$$G^* = \arg\min_G \max_D \mathcal{L}_{cGAN}(G, D) \tag{2}$$

The authors of [1] also add a L1 term to enforce the generated images to be similar to the ground truth ones: they argue that the PatchGAN discriminator models high-

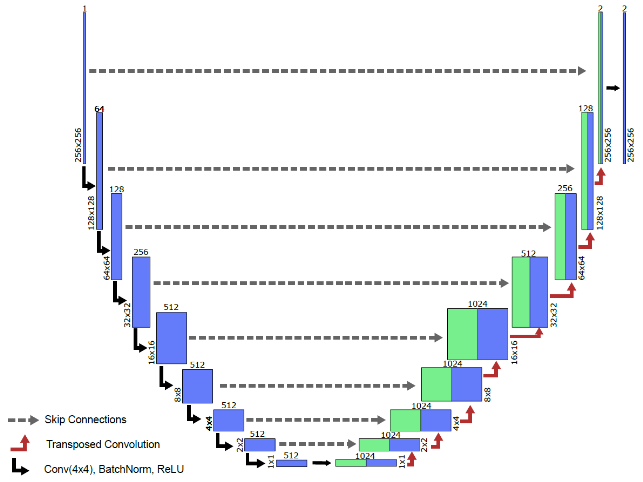

Figure 2. Schema of the implemented U-Net architecture for the generator.

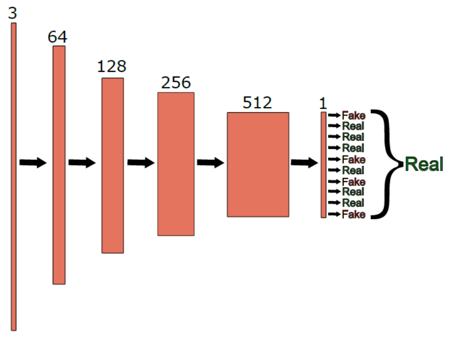

Figure 3. Schema of the implemented PatchGAN architecture for the discriminator.

frequency structure while the L1 loss encourages low-frequency correctness. The final objective is then:

$$G^* = \arg\min_G \max_D \mathcal{L}_{cGAN}(G, D) + \lambda \mathcal{L}_{L1}(G) \tag{3}$$

In this work, λ is set to 100, as suggested in the original paper.

The Wasserstein GAN (WGAN), introduced in [18], proposes a different objective function. Instead of minimizing the *Jensen-Shannon* (JS) divergence

$$JS(\mathbb{P}_r, \mathbb{P}_g) = KL(\mathbb{P}_r || \mathbb{P}_m) + KL(\mathbb{P}_g || \mathbb{P}_m) \tag{4}$$

where $\mathbb{P}_m$ is the mixture $(\mathbb{P}_r + \mathbb{P}_g)/2$, the authors suggest using the Wasserstein-1 or *Earth-Mover* (EM) distance

$$W(\mathbb{P}_r, \mathbb{P}_g) = \inf_{\gamma \in \Pi(\mathbb{P}_r, \mathbb{P}_g)} \mathbb{E}_{(x,y)\sim\gamma}\Big[||x, y||\Big] \tag{5}$$

where $\Pi(\mathbb{P}_r, \mathbb{P}_g)$ denotes the set of all joint distribution $\gamma(x, y)$ whose marginals are $\mathbb{P}_r$ and $\mathbb{P}_g$ respectively. Informally, this represents the minimum cost of transporting mass in order to transform the distribution $\mathbb{P}_r$ into the distribution $\mathbb{P}_g$. From this distance, the WGAN value function can be constructed using the Kantorovich-Rubinstein duality [19]

$$\min_{G} \max_{D \in \mathcal{D}} \mathbb{E}_{\boldsymbol{x} \sim \mathbb{P}_r}[D(\boldsymbol{x})] - \mathbb{E}_{\tilde{\boldsymbol{x}} \sim \mathbb{P}_g}[D(\tilde{\boldsymbol{x}})] \tag{6}$$

where $\mathcal{D}$ is the set of 1-Lipschitz functions. In this framework, the discriminator is also called *critic* since it no longer classifies samples as real or fake but outputs a scalar value that measures the realism of a sample, with the goal of producing low values for real samples and high values for generated samples. To enforce the Lipschitz constraint on the critic, the authors suggest clipping the weights of the critic to lie within a compact space [-c, c]. The WGAN value function results in a critic function whose gradient with respect to its input is better behaved than its GAN counterpart, making optimization of the generator easier (although longer) and more stable. Moreover, the authors observe that the WGAN value function appears to correlate with sample quality, which is not the case for GANs.

An additional improvement to the WGAN is proposed by [20]. The authors suggest adding a new term to the objective to overcome some of the problems of WGANs caused by the weight clipping, such as capacity underuse and exploding or vanishing gradients. Leveraging the fact that a differentiable function is 1-Lipschitz if and only if it has gradients with norm at most 1 everywhere, a gradient penalty (GP) term is added to the critic loss. The objective of the WGAN-GP is as follows:

$$\begin{aligned} &\mathbb{E}_{\tilde{\boldsymbol{x}} \sim \mathbb{P}_g}[D(\tilde{\boldsymbol{x}})] - \mathbb{E}_{\boldsymbol{x} \sim \mathbb{P}_r}[D(\boldsymbol{x})] \\ &\qquad + \lambda \mathbb{E}_{\hat{\boldsymbol{x}} \sim \mathbb{P}_{\hat{\boldsymbol{x}}}}\left[(\|\nabla_{\hat{\boldsymbol{x}}} D(\hat{\boldsymbol{x}})\|_2 - 1)^2\right] \end{aligned} \tag{7}$$

This solves the issues of weight clipping while conserving the good properties of WGAN.

The WGAN-GP is the adversarial loss ($\mathcal{L}_{adv}$) used in [2]. The authors combine it with a L2 loss (they call it color loss, $\mathcal{L}_{rec}$), that measures the Euclidean distance between the generated and ground truth *a and *b channels, and a KL divergence loss ($\mathcal{L}_{class}$), that measures the distance between the class distributions of the generated and ground truth images. So, the objective is:

$$\lambda_{adv}\mathcal{L}_{adv} + \lambda_{rec}\mathcal{L}_{rec} + \lambda_{class}\mathcal{L}_{class} \tag{8}$$

where λ_{adv}, λ_{rec} and λ_{class} are hyperparameters that control the relative importance of the three losses. In the original paper, the authors set $\lambda_{adv} = 0.1$, $\lambda_{rec} = 1$ and $\lambda_{class} = 0.003$.

Since, in this work, the labels are directly used for the classification, the KL divergence is replaced with the cross-entropy loss. The same λ values are used. The behavior of the modified architecture is also studied with the standard GAN loss with the L1 term, to compare the results with the WGAN-GP.

4.3. Evaluation metrics

A crucial aspect of the image colorization task, and in general of any image generation task, is to indentify a way to quantitatively evaluate the quality of the generated images. This, by itself, is an open problem, and lots of different metrics have been proposed in the literature: a good survey is available in [21]. The following are common choices that are adopted in this work:

- Frechet Inception Distance (FID): A metric used to assess the quality of images created by GANs. The FID compares the distribution of generated images with the distribution of a set of real images (ground truth). The lower the value of FID is, the better the similarities between the images are. The metric was originally proposed in [22].
- Structural Similarity Index Measure (SSIM): A method for predicting the perceived quality of digital television and cinematic pictures, as well as other kinds of digital images and videos. The resultant SSIM index is a decimal value between -1 and 1, where 1 indicates perfect similarity, 0 indicates no similarity, and -1 indicates perfect anti-correlation [23].
- Peak Signal Noise Ratio (PSNR): The ratio between the maximum possible power of a signal and the power of corrupting noise that affects the fidelity of its representation. Higher values of PSNR imply smaller colorizing errors [24].
- Learned Perceptual Image Patch Similarity (LPIPS): Used to judge the perceptual similarity between two images. LPIPS essentially computes the similarity between the activations of two image patches for some pre-defined network. This measure has been shown to match human perception well. A low LPIPS score means that image patches are perceptual similar [25].

A visual inspection of the generated images remains a key aspect of the evaluation process: ideally, one should test the ability of the model to generate images that are plausible and realistic to the point of fooling a human observer (services like Amazon Mechanical Turk [26] can be used for this purpose).

5. Experiments

The baseline model (GAN + L1 loss) is trained following the recipe described in [1]: for each iteration, the discriminator is trained once on both real and fake samples, then the generator is trained once on the discriminator output of the generated samples. As suggested in [27], one-sided label smoothing is applied to the real labels in the GAN loss, which are set to 0.9 instead of 1. This is done to prevent the discriminator from becoming too confident which could lead to training instability, mode collapse, and degraded image quality. The Adam optimizer [28] is used with an initial learning rate of 2×10^{-4} and $\beta_1 = 0.5$ which should help reach an equilibrium between generator and discriminator. The model is trained for 30 epochs during which the learning rate is progressively lowered to 2×10^{-6} using a cosine annealing schedule [29]. The batch size is set to 8, and the training takes about 1 hour per epoch on an NVIDIA RTX 2080Ti GPU. The exact same approach is followed for the modified architecture, with the classification branch, when using the GAN loss with the L1 and cross-entropy terms.

Just a few modifications are needed when using the WGAN-GP loss: the batch normalization layers in the critic are disabled as it is done in [20] and the critic is trained more times that the generator (5 times is the amount suggested in both [18] and [20]). In this case, the learning rate of the Adam optimizer is set to 10^{-5} and kept constant for the whole training that lasts for 30 epochs. Given the fact that, while being more stable, the WGAN-GP requires more iterations it is trained for an additional 20 epochs to see if there is any additional improvement.

In Table 1 the metrics evaluated on the test set are reported, while in Figure 4 some sample images for each configuration are shown along the input grayscale image and the ground truth color image. There is a clear improvement in all metrics with respect to the baseline pix2pix model when adding the classification branch, both when using the GAN loss with the L1 term and the WGAN-GP loss with the L2 term. The only exception is the FID, which is considerably worse when using the WGAN-GP loss, while there is not much difference in the rest of the metrics between the modified architecture with the GAN loss and the one with the WGAN-GP loss. Moreover, the additional 20 epochs of training for the WGAN-GP loss model do not seem to have any effect.

	FID	SSIM	PSNR	LPIPS
GAN + L1	1.506	0.893	26.360	0.098
GAN + L1 + Class	**1.217**	0.911	27.324	0.076
WGAN-GP + L2 + Class (30 epochs)	2.593	**0.934**	**27.734**	**0.073**
WGAN-GP + L2 + Class (50 epochs)	2.357	**0.934**	27.666	**0.073**

Table 1. Metrics evaluated on the test set. The best value for each metric is highlighted in bold.

Let's now verify if the quantitative analysis is reflected in the visual inspection of the generated images. All tested configurations are able to produce plausible and realistic images, almost indistinguishable from the ground truth ones, when dealing with outdoor scenes, such as landscapes, mountains, beaches, fields, etc. Examples of this behavior are shown in rows 1, 4, 5, 6 and 7 of Figure 4. There are some instances where the WGAN variant seems to have less saturated colors, which might explain the worse FID score. The images in row 3 are a clarifying example of the improvements brought by the added classification branch: the baseline model fails to colorize the field track, using a green shade, while the modified architecture, thanks to the extracted semantic information, recognizes the context and correctly colors the track. On the other hand, there are two major occurrences where all models do not produce satisfactory results. The first one is when dealing with indoor scenes where, due to their complexity and richness in details, the models are not precise at all in the colorization of all objects and architectural elements: this is the case with the images in row 2. The second failure case is when people are present in the image: all the proposed configurations fail to correctly colorize them, in particular leaving parts of faces and hair with grayish and bluish tones. The final row of Figure 4 illustrates this behavior.

6. Conclusion

In this work, I have successfully implemented the pix2pix model for the task of image colorization. I have then shown that the addition of a classification branch, that extracts semantic information from the image, improves the quality of the generated images, both quantitatively and qualitatively. In particular, both the GAN loss with the L1 term and the WGAN-GP loss with the L2 term have been tested, and the results obtained are comparable and more tuning and testing is needed to determine if one approach is better than the other. For examples, modification to the architecture and tuning of hyperparameters, like the relative importance of the different losses, should be investigated. Overall, the results obtained are satisfactory, but there is still room for improvement, especially for the two highlighted failure cases. It would be interesting, given more time and computational resources, to see if and how the results change when using the entire Places205 dataset, and then to test the model performance on other datasets.

An idea to tackle the problem of people and complex interior scene colorization can be formulated by looking at the work of Su et al. [7]. Without implementing their complex architecture that requires three phases of training (one for the full-image colorization network, one for the instance colorization network and one for the fusion module) it may be possible to keep the same architecture and use an off-the-shelf object detector to extract the object instances from the image. During the training phase, with a given probability, instead of training on the full image, the model could be trained on the extracted object instances, which would allow it to focus on how to properly colorize people and objects. This would allow to keep the same fairly simple architecture and training procedure, moving the required extra steps to the preprocessing phase: this approach can be considered as a data augmentation technique.

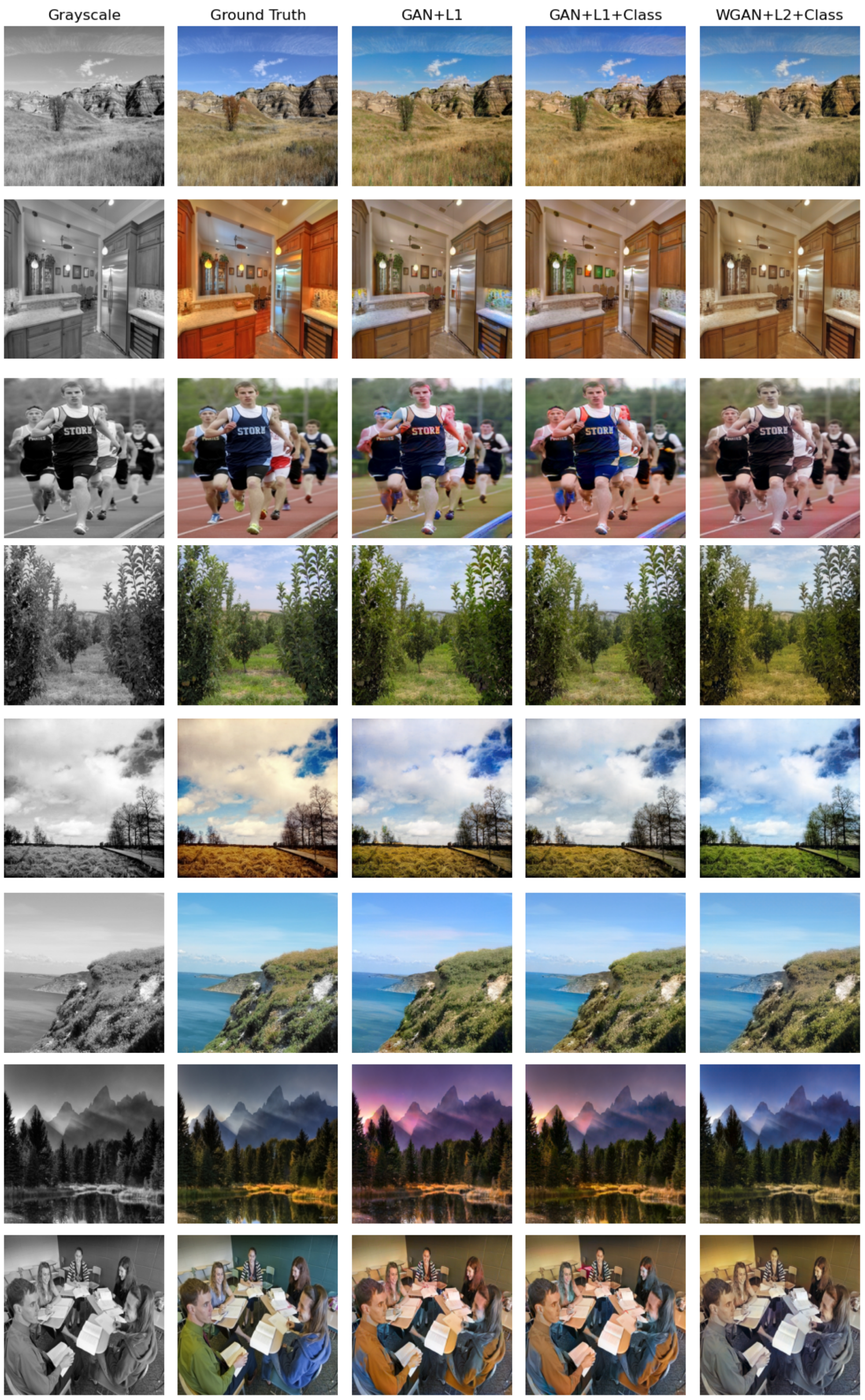

Figure 4. Sample images from the test set. The first column shows the grayscale input, the second column shows the ground truth, the third the generated image with the baseline model, the fourth the one of the added classifier brach (with GAN and L1 losses), and the fifth the one of, again the modified architecture, with the WGAN-GP and L2 losses.

References

[1] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1125–1134, 2017.

[2] Patricia Vitoria, Lara Raad, and Coloma Ballester. Chromagan: Adversarial picture colorization with semantic class distribution. In *The IEEE Winter Conference on Applications of Computer Vision*, pages 2445–2454, 2020.

[3] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2014.

[4] Richard Zhang, Phillip Isola, and Alexei A Efros. Colorful image colorization. In *ECCV*, 2016.

[5] Patsorn Sangkloy, Jingwan Lu, Chen Fang, Fisher Yu, and James Hays. Scribbler: Controlling deep image synthesis with sketch and color, 2016.

[6] Satoshi Iizuka, Edgar Simo-Serra, and Hiroshi 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.*, 35(4), jul 2016.

[7] Jheng-Wei Su, Hung-Kuo Chu, and Jia-Bin Huang. Instance-aware image colorization, 2020.

[8] Saeed Anwar, Muhammad Tahir, Chongyi Li, Ajmal Mian, Fahad Shahbaz Khan, and Abdul Wahab Muzaffar. Image colorization: A survey and dataset. *arXiv preprint arXiv:2008.10774*, 2020.

[9] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition, 2015.

[10] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.

[11] Bolei Zhou, Agata Lapedriza, Jianxiong Xiao, Antonio Torralba, and Aude Oliva. Learning deep features for scene recognition using places database. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014.

[12] Places205 10% subset. `https://www.kaggle.com/datasets/mittalshubham/images256`. [Online; accessed 15-September-2023].

[13] Wenhai Wang, Jifeng Dai, Zhe Chen, Zhenhang Huang, Zhiqi Li, Xizhou Zhu, Xiaowei Hu, Tong Lu, Lewei Lu, Hongsheng Li, Xiaogang Wang, and Yu Qiao. Internimage: Exploring large-scale vision foundation models with deformable convolutions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14408–14419, June 2023.

[14] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*, pages 234–241. Springer, 2015.

[15] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. pmlr, 2015.

[16] Andrew L Maas, Awni Y Hannun, Andrew Y Ng, et al. Rectifier nonlinearities improve neural network acoustic models. In *Proc. icml*, volume 30, page 3. Atlanta, Georgia, USA, 2013.

[17] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 807–814, 2010.

[18] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan. 2017.

[19] C. Villani. *Optimal Transport: Old and New*. Grundlehren der mathematischen Wissenschaften. Springer Berlin Heidelberg, 2008.

[20] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron Courville. Improved training of wasserstein gans. 2017.

[21] Alaa Abu-Srhan, Mohammad A.M. Abushariah, and Omar S. Al-Kadi. The effect of loss function on conditional generative adversarial networks. *Journal of King Saud University - Computer and Information Sciences*, 34(9):6977–6988, 2022.

[22] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.

[23] SSIM documentation. `https://torchmetrics.readthedocs.io/en/stable/image/structural_similarity.html`. [Online; accessed 15-September-2023].

[24] PSNR documentation. `https://torchmetrics.readthedocs.io/en/v0.8.2/image/peak_signal_noise_ratio.html`. [Online; accessed 15-September-2023].

[25] LPIPS documentation. `https://torchmetrics.readthedocs.io/en/stable/image/learned_perceptual_image_patch_similarity.html`. [Online; accessed 15-September-2023].

[26] Amazon Mechanical Turk. `https://www.mturk.com/`. [Online; accessed 15-September-2023].

[27] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans, 2016.

[28] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2014.

[29] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts, 2016.