Effect of Hyper-Parameters on Image Colorization

Min Hyeok Lee
University of Toronto
minhyeok.lee@mail.utoronto.ca

Anton Liu
University of Toronto
anton.liu@mail.utoronto.ca

Vladislav Trukhin
University of Toronto
vladislav.trukhin@mail.utoronto.ca

Abstract

This paper investigates image colorization, a task of converting a gray-scale image into a colored one. Many approaches have been formulated, from manual and supervised techniques to unsupervised machine learning and neural networks models. We analyze the performance of two leading neural network architectures for this task, CNN and U-Net, over a COCO image data set relative to the tuning of their respective hyper-parameters. We found that weight initialization is the most influential hyper-parameter on the performance of the model followed by batch size and number of layers.

1 Introduction

The task of image colorization is an easy task for humans who can easily infer the colors of a gray-scale image based on intuition and experience. However, such an approach to colorization requires manpower, skill, and time. In recent years, numerous colorization techniques have been introduced to automatize this task. Researchers attempting to solve this problem have proposed combinations of convolution layers and hyper-parameters for their network architecture, with varying degrees of success. In this paper, we compare the performance of two network architectures, a CNN generator from Zhang et al. [2016], and a U-Net generator from Isola et al. [2016]. Specifically, we modify several of their hyper-parameters: learning rate, batch size, weight, and number of layers, and see how these changes affect performance, our metric being generator loss. Through our experiments, we have found:

- Weight initialization has the most influence on the model performance
- Batch size has influence up to a certain size
- Number of layers may be influential depending on the model
- Learning rate has the least influence relative to the other hyper-parameters

Therefore, to improve further performance of these networks, the results suggest that the focus on hyper-parameter tuning should be on the weight initialization and number of layers.

2 Related Work

Most colorization methods can be grouped into three categories, scribble, transfer and automatic.

Scribble techniques, introduced by Levin et al. [2004], propagate a manually chosen color across a region of a target image. These techniques can be applied multiple times to bring refined results.

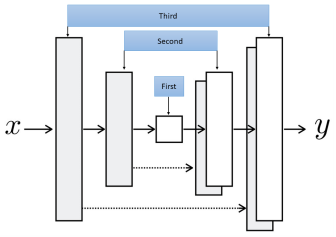
35th Conference on Neural Information Processing Systems (NeurIPS 2021).

The technique has been extended to also be aware of texture Luan et al. [2007], Qu et al. [2006], and edges Huang et al. [2005].

Transfer techniques use a collection of related reference images to match its color to the target image. Matching is achieved via local descriptors Welsh et al. [2002].

Automatic prediction utilizes machine learning and neural networks to predict the color of the target image. Works in this category include Deshpande et al. [2015] linear system approach, Cheng et al. [2016] and Iizuka et al. [2016] neural network approach with handcrafted features, and Zhang et al. [2016] and Larsson et al. [2016] neural network approach with automatic color histograms prediction for the pixels.

3 Methods



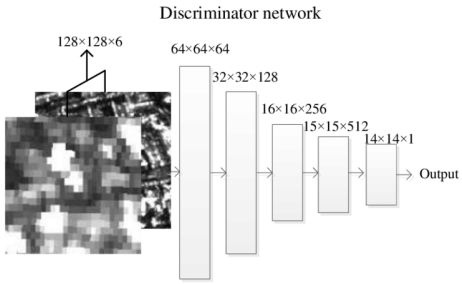


Figure 1: Architecture of the Generator network in Image-to-Image Translation with Conditional Adversarial Networks from Isola et al. [2016] (LEFT)

Figure 2: Architecture of the discriminator network in Image-to-Image Translation with Conditional Adversarial Networks from Ao et al. [2018] (RIGHT)

Isola et al. [2016] propose a conditional GAN model to perform image based tasks. The generator is a U-Net, which is structurally the same as encoder-decoder networks, with the difference of having skip connections between encoder and decoder layers. The model uses a CNN that outputs a matrix of "real" or "fake" predictions as the discriminator.

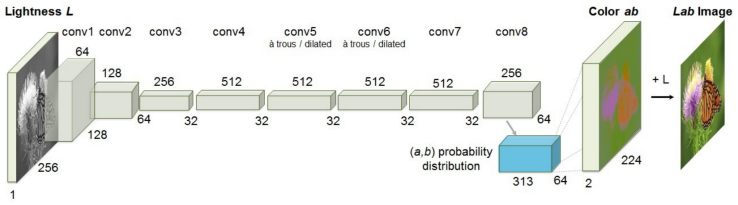
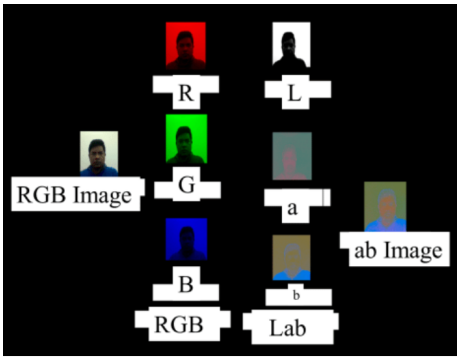


Figure 3: Architecture of Colorful Image Colorization from Zhang et al. [2016]

Zhang et al. [2016] aims to generate vibrant images by generating a probability distribution of the colors for each pixel within a discretized color space. The model's backbone is VGG-16, with more convolution layers added on the head. We replace the U-Net generator in Isola et al. [2016] GAN with Zhang et al. [2016] CNN generator.



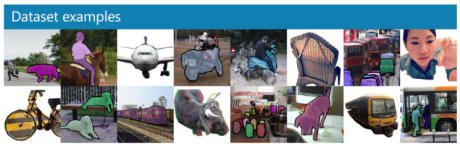


Figure 4: Difference between RGB and Lab color space Rahman et al. [2019] (LEFT)

Figure 5: Example pictures from the COCO dataset Lin et al. [2014] (RIGHT)

Instead of predicting colors in the RGB color space, both papers instead use L*a*b* color space. The L channel, which corresponds with lightness, is an additional input to the gray-scale. Therefore, only two channels needs to be predicted, a and b, both corresponding to four unique colors, red, green, blue, and yellow. This reduces the complexity of the model and improves prediction accuracy.

The COCO data set from Lin et al. [2014] is chosen to train these two models for its large and diverse repository of images and objects. The images are cropped to 256 x 256 pixels and converted to L*a*b* color space.

4 Experiments

Four hyper-parameters have been chosen to examine their effects on both models: learning rate, batch size, weight initialization, and the number of layers. The baseline hyper-parameter values for the models are 5e-4 for learning rate, 16 batch, normal weight initialization and normal number of layers depending on the model.

8 tests are done for both models, where a single parameter is changed while others are kept as baseline (e.g. learning rate is set to 1e-3 while the other three stays default). Each test is done over 100 epochs, with each epoch taking 3-5 minutes to train on a Nvidia 1070 GPU.

We tested two forms of weight initialization proposed by Glorot and Bengio [2010] and He et al. [2015]. The former initializes the weights such that the variance of the activation are the same across every layer. The later expands on the former by taking into account the non-linearity of activation functions.

As for changing the number of layers, the baseline for Zhang et al. [2016] model is without conv5, conv6, and conv7 blocks. The two experiments conducted for it is with these blocks added back in, and further removal of layers from conv3, conv4, and conv8 blocks (the blocks are seen in Figure). The baseline for Isola et al. [2016] model has 7 layers, then experimented with adding and removing two layers (one from encoding side, and another from decoding side).

5 Results

Table 1 and 2 above shows the generator loss for Zhang et al. [2016] CNN and Isola et al. [2016] Isola et al. [2016] respectively. Image results are available in the appendix.

We used the generator loss as the metric to evaluate a model's performance and default hyper-parameters model's performance as the baseline. We ignored the discriminator loss as it did not change significantly with changes to hyper-parameters.

We have found that increasing and decreasing the learning rate improved and hampered a model's performance respectively, although not as significantly as other hyper-parameters. Lower batch sizes worsened the performance for both models but higher batch sizes only improved performance for Isola et al. [2016]. He et al. [2015] weight initialization improved Zhang et al. [2016] model's

Table 1: Result Table for Zhang et al. [2016] CNN Model

hyper-parameters	Loss G GAN	Loss G L1	Loss G
Default	17.24310	483.06842	500.31152
Lower Learning Rate	18.82335	475.31469	494.13804
Higher Learning Rate	20.48103	490.86598	511.34701
Lower Batch Size	26.30590	963.49554	989.80144
Higher Batch Size	20.26480	497.55519	517.82000
Weight Initialization (Glorot and Bengio [2010])	11.82945	621.77997	633.60942
Weight Initialization (He et al. [2015])	16.83687	448.28673	465.12361
Number of Layers (Full)	13.88863	454.20604	468.09468
Number of Layers (Removal 348)	16.39976	750.22294	766.62271

Table 2: Result Table for Isola et al. [2016] Isola et al. [2016]

hyper-parameters	Loss G GAN	Loss G L1	Loss G
Default	1.92645	7.01105	8.93749
Lower Learning Rate	2.43476	5.80457	8.23933
Higher Learning Rate	3.08370	6.09162	9.17532
Lower Batch Size	3.07491	6.08733	9.16224
Higher Batch Size	1.67501	5.44368	7.11869
Weight Initialization (Glorot and Bengio [2010])	1.56918	5.76778	7.33696
Weight Initialization (He et al. [2015])	6.0986	11.61434	17.71295
Number of Layers ($n_{down} = 9$)	2.72992	5.88656	8.61648
Number of Layers ($n_{down} = 7$)	2.69694	5.97365	8.67059

performance while Glorot and Bengio [2010]'s improved Isola et al. [2016]'s performance. Changes to layer counts significantly affected Zhang et al. [2016] while both improved for Isola et al. [2016].

6 Discussion

Our motivation was to see the effect of hyper-parameters tuning on different colorization models. We found that weight initialization to be the most influential, followed by batch size and layer count. Our experiment was constrained by lack of access to time and compute power for running additional colorization models, which make it difficult to generalize our results. Future work can be on testing additional hyper-parameters and techniques for weight initialization and expanding the amount of image colorization models tested.

7 Conclusion

The results of this paper suggest that weight initialization is the most influential hyper-parameter on performance of image colorization models, followed by batch size and number of layers. This has been tested on both Zhang et al. [2016] CNN and Isola et al. [2016] U-Net generative models, suggesting that our findings could be generalized on a broad range of colorization models.

8 Contribution

- Min Hyeok Lee: Merged Two model into one file and tested COCO image data set on Isola et al. [2016] U-Net generative models
- Anton Liu: Merged Two model into one file and tested COCO image data set on Zhang et al. [2016] CNN Model.
- Vladislav Trukhin: Research, Report

References

D. Ao, C. Dumitru, G. Schwarz, and M. Datcu. Dialectical gan for sar image translation: From sentinel-1 to terrasar-x. *Remote Sensing*, 10:1597, 10 2018. doi: 10.3390/rs10101597.

Z. Cheng, Q. Yang, and B. Sheng. Deep colorization. *CoRR*, abs/1605.00075, 2016. URL `http://arxiv.org/abs/1605.00075`.

A. Deshpande, J. Rock, and D. Forsyth. Learning large-scale automatic image colorization. pages 567–575, 12 2015. doi: 10.1109/ICCV.2015.72.

X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In Y. W. Teh and M. Titterington, editors, *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pages 249–256, Chia Laguna Resort, Sardinia, Italy, 13–15 May 2010. PMLR. URL `https://proceedings.mlr.press/v9/glorot10a.html`.

K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification, 2015. URL `https://arxiv.org/abs/1502.01852`.

Y.-C. Huang, Y.-S. Tung, J.-C. Chen, S.-W. Wang, and J.-L. Wu. An adaptive edge detection based colorization algorithm and its applications. pages 351–354, 01 2005. doi: 10.1145/1101149.1101223.

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 Transactions on Graphics*, 35:1–11, 07 2016. doi: 10.1145/2897824.2925974.

P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks, 2016. URL `https://arxiv.org/abs/1611.07004`.

G. Larsson, M. Maire, and G. Shakhnarovich. Learning representations for automatic colorization, 2016. URL `https://arxiv.org/abs/1603.06668`.

A. Levin, D. Lischinski, and Y. Weiss. Colorization using optimization. *ACM Transactions on Graphics*, 23, 06 2004. doi: 10.1145/1015706.1015780.

T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, and P. Dollár. Microsoft coco: Common objects in context, 2014. URL `https://arxiv.org/abs/1405.0312`.

Q. Luan, F. Wen, D. Cohen-Or, L. Liang, Y.-Q. Xu, and H.-Y. Shum. Natural image colorization. pages 309–320, 01 2007. doi: 10.2312/EGWR/EGSR07/309-320.

Y. Qu, T.-T. Wong, and P.-A. Heng. Manga colorization. *ACM Trans. Graph.*, 25(3):1214–1220, jul 2006. ISSN 0730-0301. doi: 10.1145/1141911.1142017. URL `https://doi.org/10.1145/1141911.1142017`.

H. Rahman, M. Ahmed, and S. Begum. Non-contact physiological parameters extraction using facial video considering illumination, motion, movement and vibration. *IEEE Transactions on Biomedical Engineering*, PP:1–1, 05 2019. doi: 10.1109/TBME.2019.2908349.

T. Welsh, M. Ashikhmin, and K. Mueller. Transferring color to greyscale images. *ACM Trans. Graph.*, 21:277–280, 07 2002. doi: 10.1145/566570.566576.

R. Zhang, P. Isola, and A. A. Efros. Colorful image colorization, 2016. URL `https://arxiv.org/abs/1603.08511`.

A Appendix

A.1 default: learning rate: 5e-4, batch size: 16, weight norm, skip layer 567

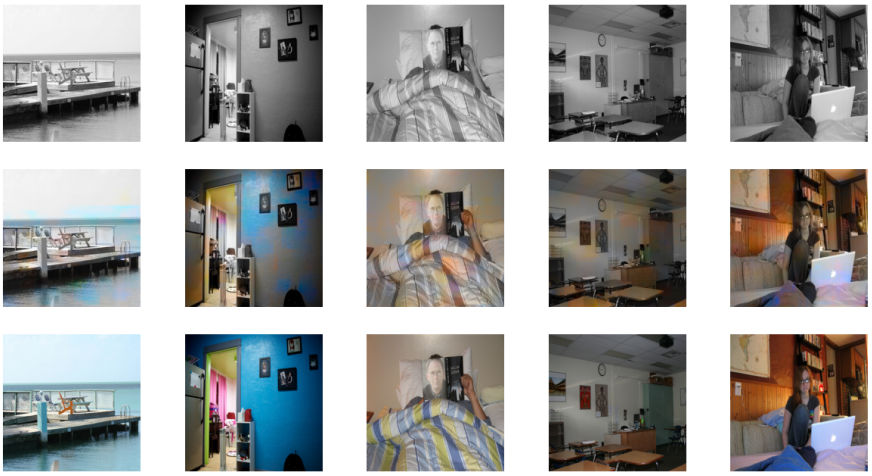


Figure 6: default result of Zhang et al. [2016]

Figure 7: default result of Isola et al. [2016]

A.2 learning rate = 2e-4

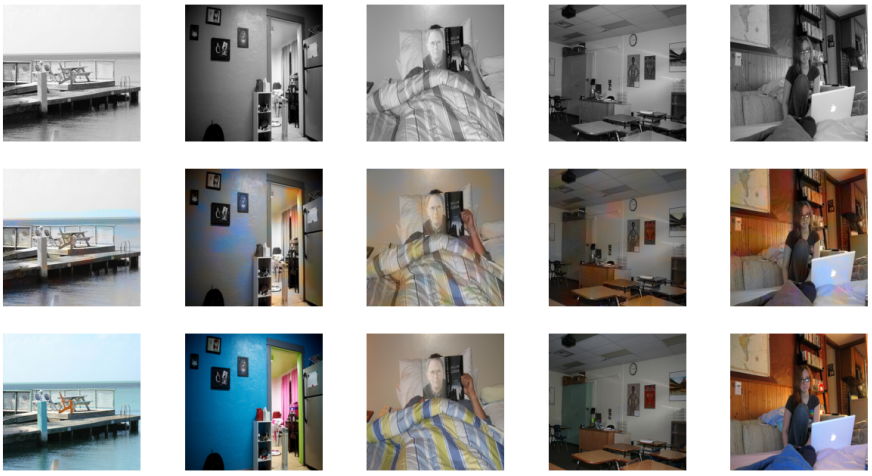


Figure 8: lower learning rate (2e-4) of Zhang et al. [2016]

Figure 9: lower learning rate (2e-4) of Isola et al. [2016]

A.3 learning rate = 1e-3

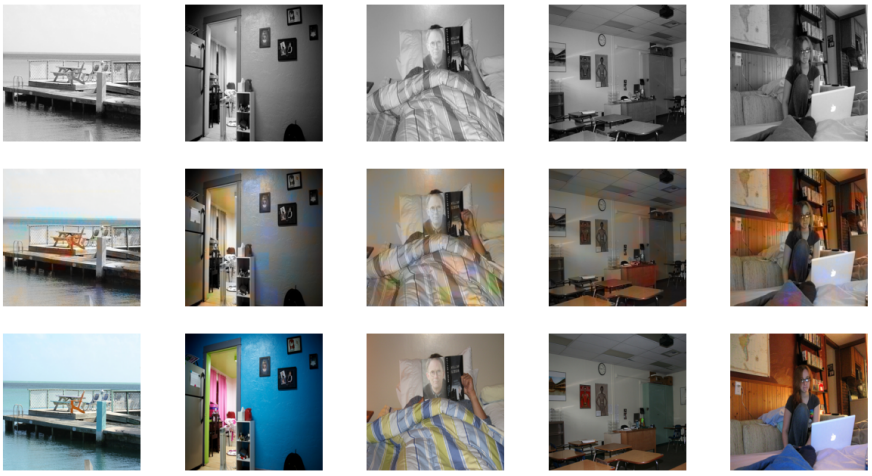


Figure 10: higher learning rate (1e-3) of Zhang et al. [2016]

Figure 11: higher learning rate (1e-3) of Isola et al. [2016]

A.4 batch size = 32

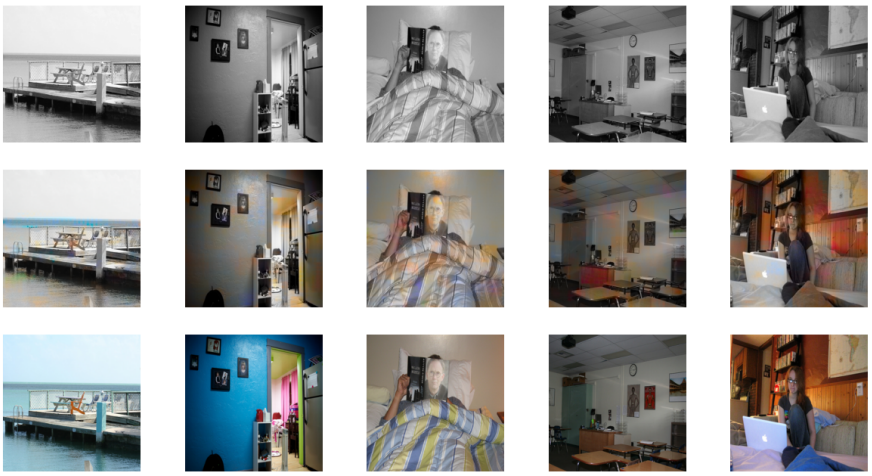


Figure 12: batch size 32 of Zhang et al. [2016]

Figure 13: batch size 32 of Isola et al. [2016]

A.5 batch size = 8

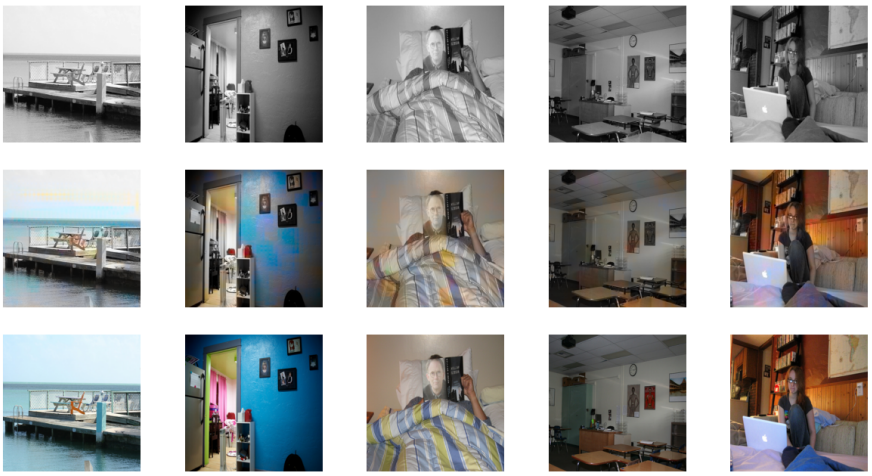


Figure 14: batch size 8 of Zhang et al. [2016]

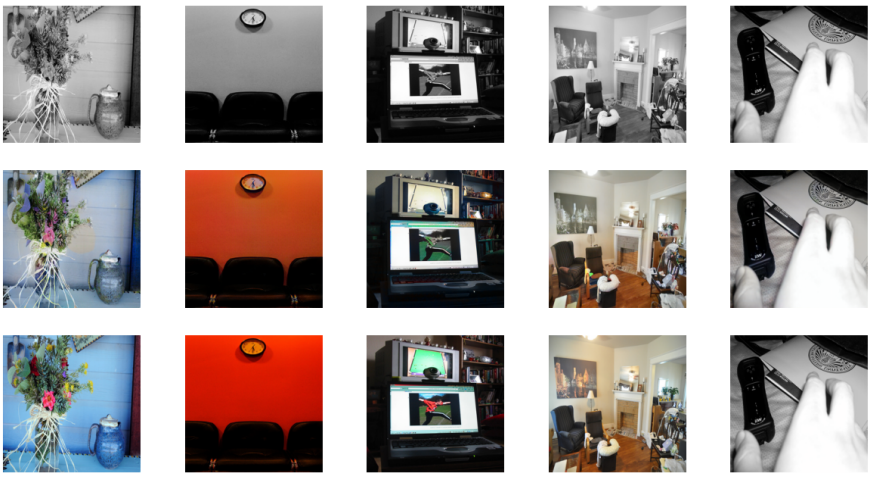


Figure 15: batch size 8 of Isola et al. [2016]

A.6 weight initialization: Glorot and Bengio [2010]

Figure 16: Glorot and Bengio [2010] weight initialization of Zhang et al. [2016]

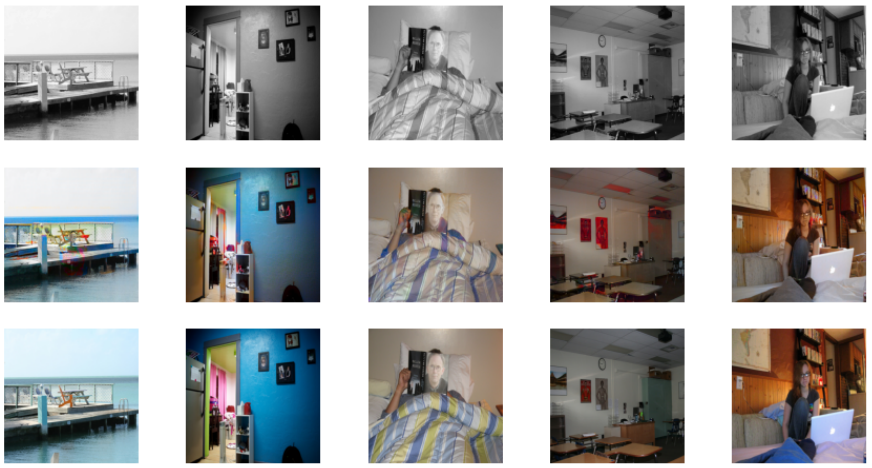


Figure 17: Glorot and Bengio [2010] weight initialization of Isola et al. [2016]

A.7 weight initialization: He et al. [2015]

Figure 18: He et al. [2015] weight initialization of Zhang et al. [2016]

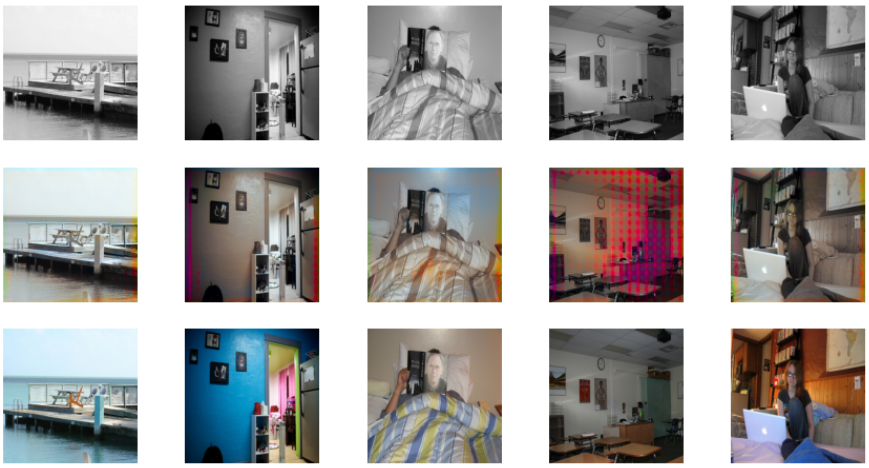


Figure 19: He et al. [2015] weight initialization of Isola et al. [2016]

A.8 full layer

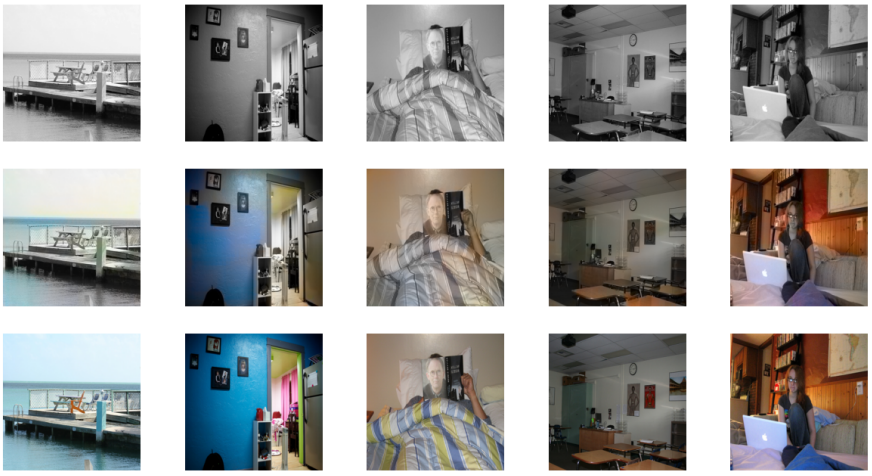


Figure 20: full layer of Zhang et al. [2016]

Figure 21: n_{down} = 9 of Isola et al. [2016]

A.9 skip layer

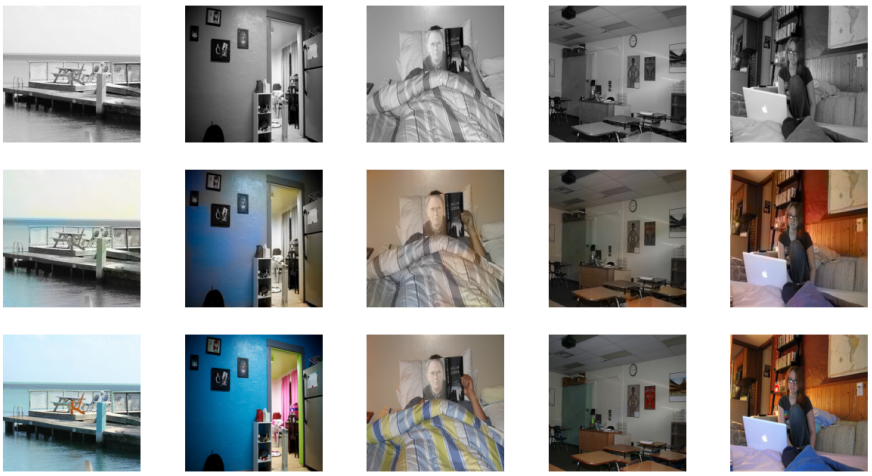


Figure 22: removal layer 348 of Zhang et al. [2016]

Figure 23: n_{down} = 7 of Isola et al. [2016]