CS 230
Deblurring and Coloring Images with Deep Learning – CS 230 – Spring 2020

Omar El Safty, Karsu Ipek Kilic
osafty@stanford.edu, karsu@stanford.edu
Stanford University

Abstract
Image deblurring and image colorization are two active research fields in computer vision that both aim at restoring and adding style to images. In this project we combine two deep neural networks models for deblurring and image colorization to restore motion blurred grayscale images as colorized and deblurred. We explore different ways of adapting and combining a GAN model for deblurring with a pretrained CNN model for image colorization. Our implementation results in comparable image quality to the state-of-the-art models.

1. Introduction

One of the artifacts of photography is when images turn to be blurry due to camera vibration or the motion of the object being photographed, leading to poor quality blurry images. The degree of blurriness can vary depending on several factors such as depth variation, the degree to which the camera is shaken and the blockage of the boundaries of the motion [1]. Deblurring such poor quality images to restore them as realistic sharp images is an active research area. Image colorization, on the other hand, is another popular field in computer vision generally aiming to add color to old black-white pictures realistically. Image colorization has wide range of practical implementations that involves the restoration of old grayscale videos and enhancing the coloring of images. Our project combines these two different fields with the aim of restoring old pictures that suffer from gray scaling and that are blurry. The goal is to build a deep neural network that will take blurry grayscale images as input, first produce deblurred grayscale images and then colorize them as realistic as possible, depicted in Figure 1. Ultimately the proper junction of these two neural network implementations, deblurring and colorizing respectively, to build a properly functioning pipeline is the main task of this project.

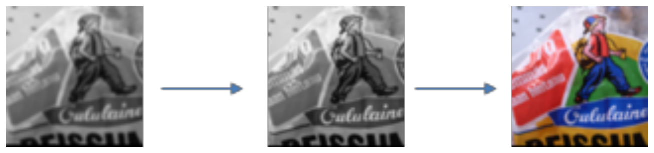
Figure 1: A blurred gray scale image will be first deblurred and then colorized. [2]

2. Related Work

Since there are many previous implementations of image colorization and deblurring we benefited from readily available algorithms with the goal of improving upon them. For both applications convolutional neural networks (CNN) are shown to be highly effective, in addition to implementations featuring generative adversial networks (GAN) [3-6]. In fact, GAN has been reported to result in improved accuracy for deblurring task [7,1,8]. One of the main challenges for colorization is to make sure that the end result has a plausible color scheme and minimize a final brownish color scale, which heavily depends on how extensive the dataset is [3]. One relatively recent study [9] achieved much more realistic colorful schemes mainly through modifying the loss function in a way to adjust the weighting of more rare colors to prevent yellow/brownish coloring of generated images. As for the deblurring of images, the adjustment of pixel coloring in the higher resolution versions has been challenging. Another challenge is that most models developed perform the best for a certain level or type of blurriness. Furthermore, since it is not possible to have ground truth versions of blurry images, sharp images are artificially blurred with blur kernels to be used for training deblurring models, thereby yielding these models' performance limited to certain types of blurs [1]. These limitations lead to difficulties for generalizing the use of deblurring models implementations good performance. While the deblurring part of our model is specific to gray scale blurry images, it performs comparable to the previously implemented GAN based deblurring models; indeed we were able to improve the quality of the final generated images although very slightly.

3. Dataset and Features

Our data set was taken from [10] prepared for a deblurring CNN model, containing 1050 sharp, slightly blurred and motion blurred images (2048x1536). For the colorization part of the pipeline we grayscaled these blurry images through data analysis and used those as the main input for our model. As for the colorization model an important modification on the images is embedded inside such that colorful RGB images taken as input are converted to "Lab" where L represents the grayscale version determining image brightness and a, b representing green-red and blue-yellow color spectrum respectively [11]. The grayscale images are used as the main input and the RGB versions represent the ground truth. For all of our training/test implementations we split the data such that 10% of it was in test and the rest was in training.

4. Method

Our approach involves the combination of a deblurring GAN model and an image colorization CNN model. A blurry gray scale input image first enters the GAN to be deblurred and the output from the deblurring part of the model then enters the colorization model to be recovered as colorized and deblurred. The CNN model was integrated as a pretrained model, whereas GAN model was rained with grayscale blurry images as input. The quality of the output images is evaluated both visually and based on their peak to signal ratio (PSNR) values, which we use as our main accuracy metric. PSNR is defined by the formula given in Eq. 1, where *MAX* corresponds to the maximum possible value for the pixels, which is 255 for 8 bit images, and MSE is simple mean squared error calculated between ground truth images and generated images:

$$PSNR = 10log_{10}\,(MAX^2/MSE) \qquad (1)$$

Initially we experimented with combining these two model implementations in two different ways and compared their performance. The final form of the pipeline of the model is shown in Figure 2. For the image colorization CNN model, every convolution layer shown is formed by three to two convolutional layers that also include ReLU activation layer and a layer for batch norm [9]. Instead of an Euclidean loss function, multinomial cross entropy function was preferred as the main cost function, since the former has been widely reported to lead to a more grayscale coloring. For the deblurring part, a conditional GAN model using multiple loss functions was adapted from [7]. The generator receives the input image and outputs a deblurred version of it, whereas the discriminator takes both the ground truth image and the output from the generator, and decides which one is more realistic where the goal of the generator is to produce outputs that can trick the discriminator. For the discriminator a Wasserstein GAN (WGAN-GP) model was used which is known to provide a stable training of GAN architectures [12]. The loss function is the combination of the content loss and adversarial loss as shown in Eq. 2:

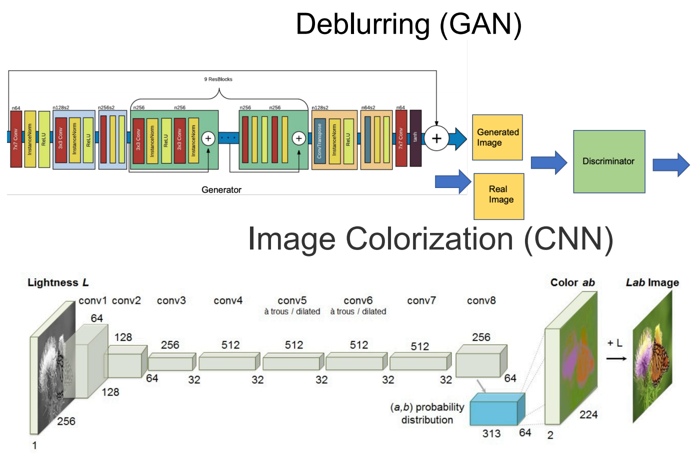

Figure 2: Schematic for the pipeline where the images are first deblurred with a GAN model and the output of the deblurring part is used as input for a CNN model for colorization. Deblurring part is adapted from [7] whereas the CNN model was adapted from [9].

$$Loss = Ladv + \lambda Lcont \qquad (2)$$

where λ is taken as 100 as suggested in [7]. Adversarial loss was chosen to be a Wasserstein loss and the content loss was chosen to be perceptual loss, evaluating the difference between the generated and real image. The GAN architecture consists of convolution layers with ½ strides, as well as nine residual blocks and convolution blocks that are transposed; whereas the ResBlock layers are formed by a convolution layer and a normalization layer. ReLU activation is used in each layer.

5. Experiments/Results/Discussion

We first experimented with two models for image colorization adapted from a study presented here [11] and both models are based on CNNs, with one of them benefiting from pretrained weights for object classification to support colorization. These weights were obtained through training around a million images obtained from image-net [13]. We split 350 images into training and test sets only, having 315 and 35 images respectively. Figure 3 shows the results we obtained with these two models using blurry images. For the first version, the final colorized images are in green/red scale whereas for the second version the results are in purple/blue color scheme. Note that these input images we used were not taken from image-net, on which the object classification was trained, where the algorithm does a much better job although still resulting in a brownish color scheme. This shows that the dataset has a significant effect on the results. Furthermore, we experimented with changing the architecture of the network and playing with the hyperparameters as shown in Table 1, however the best loss values we obtained, shown in Table 2, indicate that the model still performs very poorly. Loss Another very important reason for why we were not able to improve the results as expected is likely to be that the number of images we work with was not sufficient. For a successful image colorization task the model needs to be trained by a very large number of images which can be on the order of million. For this reason we decided to use a pretrained image colorization model to support our network as described in Fig. 2, which was adapted from another study [9]. This adaptation performs much better compared to our previous implementation in our first milestone, as shown in Figure 4 and 5. Although there is still a brownish color scale in the outputs, they are significantly more realistic than the outputs shown in Fig. 3 (b) and (d).

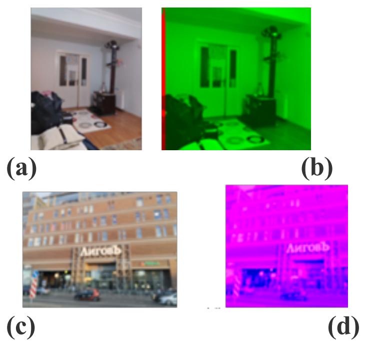

Figure 3: Test results obtained with model Version 1 with a blurry image (a,b) and with model Version 2 blurry images with another blurry image (c,d). RGB images represent ground truth.

Table 1: Some hyperparameters and information related to the runs both models performed with both blurred images.

	Number of epoch	**Number of steps per epoch**	**Learning rate**	**Optimization Algorithm**	**Batch Size**
Version 1&2	20	32	0.003	Adam	10

Table 2 compares the loss values obtained with the CNN utilizing object classification for blurred and sharp data sets.

Table 2: Loss values for the blurred and sharp data sets.

	Loss Version 1	**Loss Version 2**
Blurred Data Set	0.979	1.0091
Sharp Data Set	0.950	0.0043

We combined the deblurring GAN model with the new image colorization CNN model in two different ways. First, we used grayscale blurry images as input that enter into the colorization model; and then the colorized blurry images enter into the deblurring model to restore their sharpness. The second method is to use grayscale blurry images as input to the deblurring model to deblur them first and then input the deblurred grayscale images to the colorization model to restore the images as sharp and colorized. We compared the two models in terms of how the colorization performs depending on working with blurred or deblurred input images; in other words, whether deblurring performs prior to or after colorization. Our results indicate that depending on which model is used first the quality of the final restored image changes. Figure 4 shows that for two different pictures that when the images are first deblurred, colorization model seems to perform better in that a sharper contrast is achieved. To further support this result, we also evaluated how the model works for different levels of deblur. Figure 5 shows the same image moderately blurred and heavily blurred (Fig. 5(a) and Fig. 5 (d)), and in both cases first deblurring and then colorizing the image results in a better more realistic color scale, since first deblurred and then colorized images have less of a brownish tone overall. It can be concluded that the deblurring model should be implemented before the colorization model, therefore, the next steps involved tailoring the deblurring model to better suit the particular input of this project, which are gray-scaled blurred images as opposed to colorized blurred images.

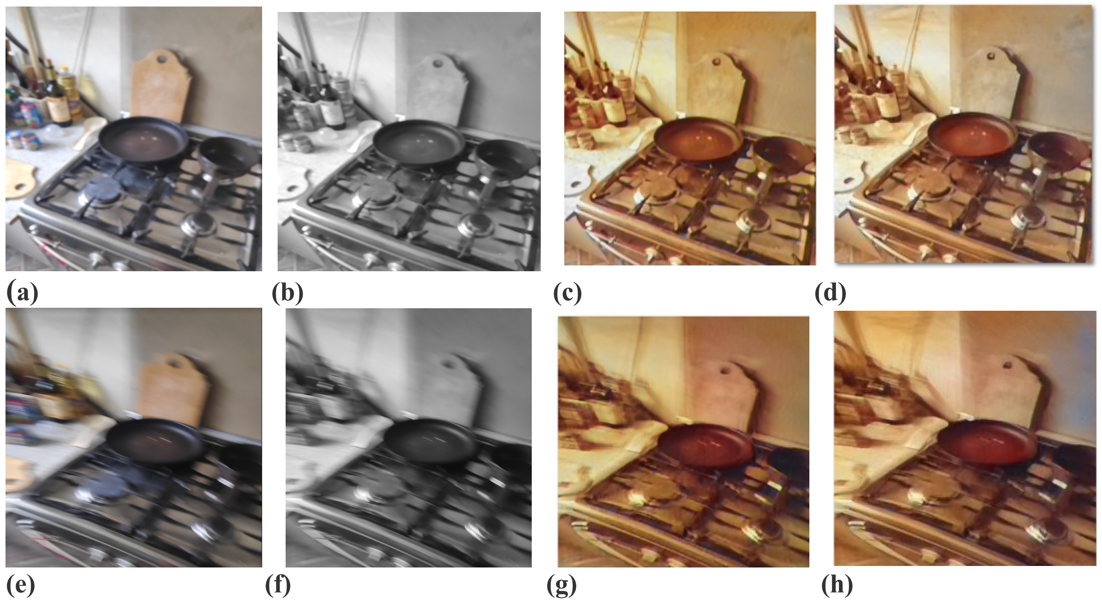

Figure 4: Test results obtained with the same picture but with two different levels of blur, where the blurry colorful versions of the image in (a) is less blurred than the one in (e). The colors of the images in (a) and (e) represent the ground truth and their gray scale versions are used as input to the pipeline, shown in (b) and (f). (c) and (g) are both first colored and deblurred; (d) and (h) are both first deblurred and then colored.

Before training on our grayscale images, we also tested the effect of multiple passes into the deblurring model on a heavily blurred image, one that is not fully deblurred after passing through the deblurring model once. The results, seen in Figure 5, indicate that multiple passes into the deblurring model does not improve the resultant image. In fact, it causes some noise in the form of blurriness in the image. As such, we settled on using only one pass through the deblurring model for our next analyses, which involved retraining the deblurring model using greyscale images.

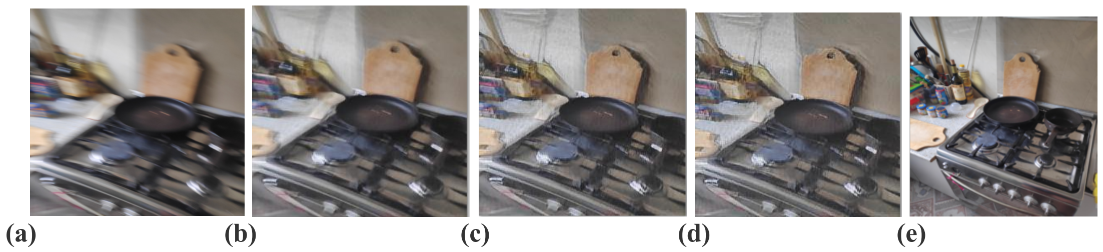

Figure 5: Test results obtained for multiple pass into the deblurring model. The original blurred image is seen in (a); followed by one, two, and three passes into the deblurring model as seen by (b), (c), and (d), respectively. The final sharp image (ground truth) is shown in (e).

We trained the model with motion blurred grayscale images using batch size 1 and 2. 4 epochs with 2000 iterations each were completed, and for both the generator and the discriminator Adam optimization was used with a learn rate equal to = 10^{-4}, which are all listed in Table 3. For batch size 2, the default loss value that was obtained with the previous implementation of the deblurring model [7] was able improve slightly, whereas for batch size 1 the loss value did not improve as can be seen in Table 4. Consequently, we were able to improve the PSNR values from 28.3 to 29 with batch size 2 as shown in Table 5. Figure 6 show the results on two different blurry images obtained with batch size 2 respectively. The PSNR values we obtained are close to the values obtained with the previous state-of-the-art deblurring models [1,14].

Table 3: Some hyperparameters and information on deblurring model training with motion blurred gray scale images.

Number of epoch	**Number of steps per epoch**	**Learning rate**	**Optimization Algorithm**	**Batch Size**
4	2000	10^{-4}	Adam	1 and 2

Table 4: Loss values for obtained during the training of the deblurring model.

	Discriminator on Generator Loss Values for batch size = 1	**Discriminator on Generator Loss Values for batch size = 2**
Epoch 1	2436.38	2384.23
Epoch 2	2437.05	2388.47
Epoch 3	2437.53	2382.05
Epoch 4	2437.50	2344.92

Table 5: PSNR values obtained with the default and new model for both colorful and grayscale test images.

	PSNR with Default Weights	**PSNR with New Weights Batch Size = 1**	**PSNR with New Weights Batch Size = 2**
Colorful	28.2	28.6	28.8
Grayscale	28.3	28.7	29.0

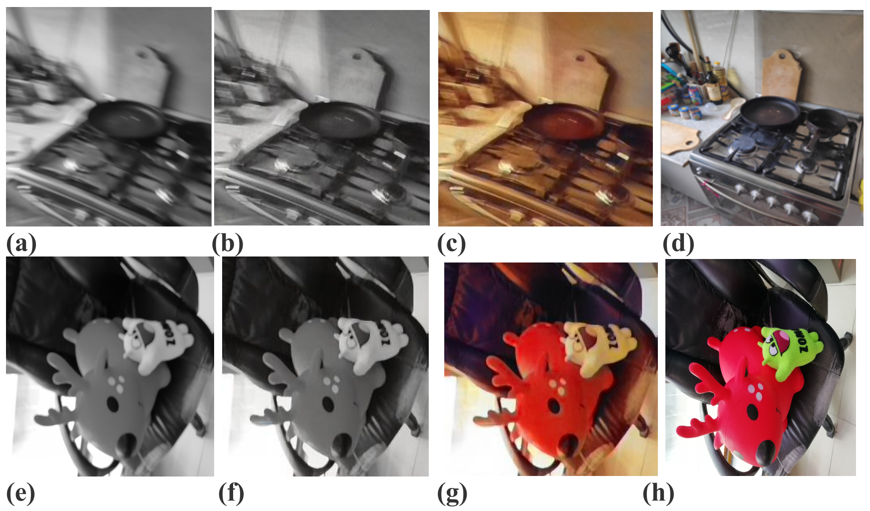

Figure 7: Test results obtained with two different blurry grayscale images (a) and (e); first deblurred with the new weights for **batch size = 2** we obtained through our model as shown in (b) and (f), and then colorized as shown in (c) and (g). (d) and (h) are the sharp, colorful (ground restored) versions of the two images. Note that (a) is motion blurred (e) is defocused.

Conclusion/Future Work

Using a pretrained image colorization CNN model and training a deblurring GAN model with motion blurred grayscale images were implemented a pipeline that takes blurry grayscale images and outputs them as colorized and deblurred images. We were able to achieve a slight increase in the PSNR values of our restored images. Due to time constraints, we were only able to train the model with 1050 images and used the weights from the previous implementation trained over a different set of images which were colorful as our initial weights. Training the model from scratch using our grayscale images and with more number of images is likely to result in a much more significant improvement in the deblurring performance. Furthermore, our model has been trained based on motion blur grayscale images. However, as mentioned previously depending on the type of blur the models can perform differently. One of the future implementations we would like to pursue is to experiment with different types of blurry/noisy grayscale images and test how the model performs when input images have different blurriness. It would be interesting to tune the model parameters for different categories of blurry images, and evaluate the performance of colorization.

7. Contributions

Research: Omar and Karsu
Preprocessing colorful motion blurred images: Karsu
First Implementation – Image Colorization Version 1&2 model training: Omar and Karsu
Second Implementation – New colorization model installation: Omar
Second Implementation – New Deblurring GAN model installation: Omar and Karsu
Second Implementation - Deblurring GAN model training: Omar and Karsu
Proposal/Milestones/FinalReport/Video: Omar and Karsu

References

[1] Nah, Seungjun, et al. “Deep Multi-Scale Convolutional Neural Network for Dynamic Scene Deblurring.” 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017.
[2] “Deblur Photos Using Generic Pix2Pix - Machine ... - Medium.” [Online]. Available: https://medium.com/machine-learning-world/deblur-photos-using-generic-pix2pix-6f8774f9701e. [Accessed: 23-Apr-2020].
[3] R. Zhang, P. Isola, and A. A. Efros, “Colorful Image Colorization,” Computer Vision – ECCV 2016 Lecture Notes in Computer Science, pp. 649–666, 2016.
[4] G. Larsson, M. Maire, and G. Shakhnarovich, “Learning Representations for Automatic Colorization,” Computer Vision – ECCV 2016 Lecture Notes in Computer Science, pp. 577–593, 2016
[5] S. Iizuka, E. Simo-Serra, and H. Ishikawa, “Let there be color!,” ACM Transactions on Graphics, vol. 35, no. 4, pp. 1–11, Nov. 2016.
[6] H. Zhao, Z. Ke, N. Chen, S. Wang, K. Li, L. Wang, X. Gong, W. Zheng, L. Song, Z. Liu, D. Liang, and C. Liu, “A new deep learning method for image deblurring in optical microscopic systems,” Journal of Biophotonics, vol. 13, no. 3, 2020.
[7] O. Kupyn, V. Budzan, M. Mykhailych, D. Mishkin, and J. Matas, “DeblurGAN: Blind Motion Deblurring Using Conditional Adversarial Networks,” 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2018.
[8] Ouyang, Yi. “Total Variation Constraint GAN for Dynamic Scene Deblurring.” Image and Vision Computing, vol. 88, 2019, pp. 113–119., doi:10.1016/j.imavis.2019.05.007.
[9] R. Zhang, P. Isola, and A. A. Efros, “Colorful Image Colorization,” Computer Vision – ECCV 2016 Lecture Notes in Computer Science, pp. 649–666, 2016.
[10] A. Alekseev, “Blur dataset,” Kaggle, 29-Jul-2019. [Online]. Available: https://www.kaggle.com/kwentar/blur-dataset. [Accessed: 09-May-2020].
[11] freeCodeCamp.org, “How to colorize black &amp; white photos with just 100 lines of neural network code,” freeCodeCamp.org, 12-Oct-2017. [Online]. Available: https://www.freecodecamp.org/news/colorize-b-w-photos-with-a-100-line-neural-network-53d9b4449f8d/. [Accessed: 07-May-2020].
[12] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville. “Improved Training of Wasserstein GANs”. ArXiv e-prints, Mar. 2017.
[13] Guirao et al,. Deep Koalarization: Image Colorization using CNNs and Inception-ResNet-v2
[14] Sun, Jian, et al. “Learning a Convolutional Neural Network for Non-Uniform Motion Blur Removal.” 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, doi:10.1109/cvpr.2015.7298677.
[15] The models ware run on Google Colab. Caffe library was used for the image colorization implementation. Keras 2.1.5 and Tensorflow 1.5 versions were used for the deblurring implementation.