Turning a Smartphone Selfie into a Studio Portrait

Nicola Capece, Francesco Banterle, Paolo Cignoni,
Fabio Ganovelli, and Ugo Erra

January 20, 2020

Abstract

We introduce a novel algorithm that turns a flash selfie taken with a smartphone into a studio-like photograph with uniform lighting. Our method uses a convolutional neural network trained on a set of pairs of photographs acquired in a controlled environment. For each pair, we have one photograph of a subject's face taken with the camera flash and another one of the same subject in the same pose illuminated using a photographic studio-lighting setup. We show how our method can amend lighting artifacts introduced by a close-up camera flash, such as specular highlights, shadows and skin shine.

1 Introduction

Photographs taken with mobile devices are nowadays predominant on the Internet, including the web-based services dedicated to professional photography such as Flickr, 500px, etc. This is due to the steady improvement of built-in digital cameras in smartphones, which has made them a default choice of many for taking pictures. Under favorable lighting conditions, smartphone picture quality has reached that of digital reflex cameras, but smartphones are not able to capture artifact-free images in low-light conditions. This is due to their sensors' size, a constraint that is not straightforward to solve because of the little room available in modern phones. Therefore, taking pictures in low light often triggers the camera flash, which is typically a low-power LED flash mounted side by side with the camera lens that produces several artifacts.

Selfies are one of the most common forms of photographs taken with a smartphone. This practice consists of taking a picture of one's face by holding the phone in one hand or by using a so-called "selfie stick". Selfies are also often low-light photographs, an unfavorable combination that produces images with specular highlights, sharp shadows, and flat and unnatural skin tones. In this paper, we explore the possibility of turning flash selfies into studio portraits by employing a convolutional neural network (CNN). Doing so is a challenge for three reasons. Firstly, it involves handling both global and local discriminant features e.g. skin tone and highlight, respectively. Secondly, it needs to match how humans expect an image to look when a flash is not used. Finally, these two requirements have to be met in the domain of human faces, where people

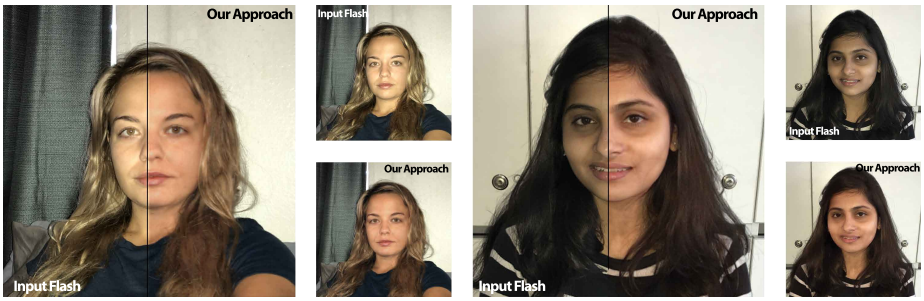

Figure 1: Two examples from our results. The split images show a comparison between the input and the output of our algorithm.

are very good at detecting any type of inconsistencies.

Smartphone flash selfies are a well-defined subdomain of photos with several common traits that our proposal is able to address: they are three-quarter or front single-face portraits, taken at close range, with a single flash co-located with the camera lens. Our approach is based on CNN training with a set of pairs of portraits (see Figure 1): one image with smartphone flash and one with photographic studio lighting (a "ground truth" image). Each pair is taken as simultaneously as possible, to keep the pose of the subject similar. The flash correction problem applies to wider application domains than just selfies, but generating the collection of images needed for this broader purpose would be an arduous undertaking, with hundreds or thousands of images required. After training our CNN with these image pairs, our model can be used to give a studio-lighting appearance to a broad range of real-world smartphone flash selfie images.

2 Related Work

Flash photography has been previously used to add details to photographs in low-light conditions, which typically suffer from high noise. In two concurrent works, Petschnigg et al. [1] and Eisemann and Durand [2] proposed transferring the ambient lighting from flash photographs with low ISO, which implies low noise, into non-flash photographs of the same subjects/scene, with reduced noise. Other works [3] have developed this idea further by removing over/under-illumination at a given flash intensity, reflections, highlights, and attenuation over depth. Removing or reducing unwanted reflections in pictures can be also obtained by the approach proposed by Zhang et al. [4], an end-to-end learning technique for single-image reflection separation with perceptual losses and a customized exclusion loss.

Eilertsen et al. [5] proposed an approach to obtain high dynamic range (HDR) images from low dynamic range images based on the U-Net architecture [1] originally developed as a CNN for biomedical image segmentation. Similarly, Chen et al. [6] showed that U-Nets can be used successfully to de-Bayer images

[1] https://en.wikipedia.org/wiki/U-Net

captured at low-light conditions and high ISO, which typically exhibit considerable noise. They extensively studied different approaches to processing such real-world noisy low-light images. For example, they tested a variety of architectures, loss functions (e.g., L1 (least absolute deviations), L2 (least square errors), and the structural similarity index (SSIM[2])), and different color inputs.

Aksoy et al. [7] presented a large-scale collection of pairs of images with ambient light and flash light of the same scene. These images were obtained by casual photographers using their smartphone cameras, and consequently, the dataset covers a wide variety of scenes. The dataset was provided for future work on high-level tasks such as semantic segmentation or depth estimation. Unlike their dataset, whose objective is to provide matching between two images under uncontrolled lighting conditions, our dataset aims to change the lighting scheme by converting from flash lighting to a controlled photograph studio light.

3 Deep Flash

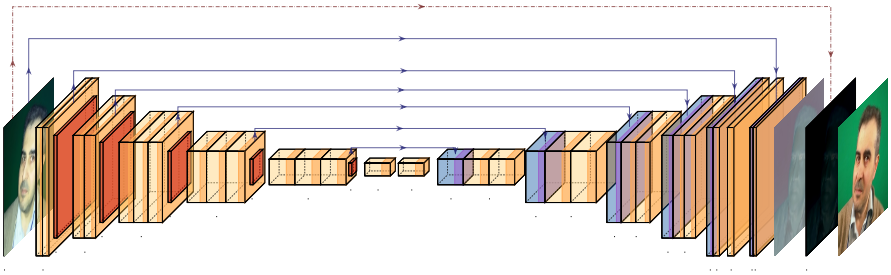

Figure 2: The architecture of our encoder-decoder, with its typical U-shape. The first 13 blocks represent the VGG-16's convolutional layers, which perform the image encoding. The second part reconstructs the output image and has several convolutional and deconvolutional layers. Arrows show the shortcut connections to the blue blocks of the decoder from their counterparts in the encoder.

We developed an encoder-decoder CNN based on two sub-networks: the first network performs the encoding of the input flash image to create a deep feature map representation; the second network recreates the image starting from the encoder's output while removing the flash defects. We use Visual Geometry Group's VGG-16 [8] network to perform the image encoding that consists of sixteen layers: the first thirteen are convolutional layers and the last three are fully-connected layers. We used only the convolutional layers of VGG-16, which are structured into five groups: the first two groups consist of two layers and the last three groups consist of three layers as shown in Figure 2. There are three operations performed by each CNN layer: several parallel convolutions, non-linear activation using ReLU (Rectified Linear Unit) functions, and max pooling operations [9].

The decoding task was performed using a decoder component that is based on Eilertsen et al.'s U-Net based approach [5]. The output produced from VGG-

[2] https://en.wikipedia.org/wiki/Structural_similarity

16 represents an input for the decoder after a further convolution operation. We use batch normalization after each convolution to normalize the output distribution of each layer in order to provide valid input for the next layer. To this end, batch normalization constrains the activation function input to have unit variance and mean zero. After each batch normalization, the output tensor crosses through the next activation function, which introduces a non-zero gradient for the next inputs. Decoder layers consist of operations such as convolutions, batch normalization, deconvolutions, and concatenations.

To reduce the "vanishing gradients" problem that affects very deep neural networks, we employed a residual learning network-based approach[3]. The vanishing gradients problem concerns the backpropagation phase, where an inverse crossing of the network is performed to update the weights through the gradient of the error function. When a network is composed of many layers, the weight updating can be reduced so much from the last to the first layers that the updates in the first network layers become inefficient, thereby stopping the training. A way to solve this problem is to concatenate each blocks output of the VGG-16 with its counterpart in the decoder using concatenation layers (see Figure 2). Another reason for our use of residual learning is that it recovers information lost through the convolutions of the encoding phase, helping the decoder in reconstructing the output image. Our encoder-decoder structured neural network is also able to recreate similar input images of faces taken in a different RGB lighting mode. Finally, we use deconvolutional layers to reconstruct the output image, starting from the VGG-16 output tensor.

3.1 Training

To minimize the loss function, we train our neural network using an algorithm called an Adam Optimizer [4], which is a stochastic gradient descent technique with a special way of managing its learning rate. As the initial configuration of the Adam Optimizer, we set the learning rate to 10^{-5}, set the ϵ value (useful for avoiding divisions by zero) to 10^{-8}, and set the mini-batch size to 4. The choice of mini-batch size value is due to the GPU capability and the input image dimension. To compensate for the limited amount of available training data and to increase the generalization level, we use transfer learning [5], a technique which extends learning achieved in one domain to related problems.

In particular, the weights of the VGG-16 were initialized through a pre-trained model originally used for face recognition [10]. This model was trained through a dataset of 2.6 million faces belonging to 2600 identities, using an NVIDIA Titan Black GPU. Our decoder weights were initialized using a truncated normal distribution, which ensures that the weights initialization has unit standard deviation and mean zero. This approach avoids dissolution or increasing of the gradient, decreasing the probability of introducing critical errors during training. For the initialization of the last decoder layer, we use Xavier Initialization [11], which ensures that the signal passing through the neural network is propagated accurately and that the weights are neither too small nor too large.

[3] https://en.wikipedia.org/wiki/Residual_neural_network
[4] https://en.wikipedia.org/wiki/Stochastic_gradient_descent
[5] https://en.wikipedia.org/wiki/Transfer_learning

3.2 Encoding

To implement our solution, we chose an encoding that decouples the high-frequency details such as hair or facial features from low-frequency details such as global skin tone. To this end, we employed the well-known bilateral filter [6], which is a nonlinear filter that is frequently used to smooth images while preserving edges. Such filtering was used on both the flash and uniformly-lit (ground truth) images of our dataset before training our neural network. Once the filter was applied, the flash filtered image was used as the input to our neural network and the target was the difference between the filtered flash image and filtered ground truth image normalized to $[0, 1]$. The use of this type of encoding reduces artifacts such as blur due to the small misalignment of facial expressions between the flash and ground truth images, closed/open eye, lips position and other facial landmarks.

3.3 Loss Function

The method described in the previous section allows us to preserve the low frequencies from the original non-filtered image for use in subsequent steps. We minimize the distance between the low frequencies of the input and ground truth as follows:

$$L(y_d, t_d) = \frac{1}{3N} \sum_i \Big((y_{di} - \mathbb{E}[y_{di}]) - (t_{di} - \mathbb{E}[t_{di}]) \Big)^2, \tag{1}$$

where

$$\begin{aligned} y_{di} &= BL(x_i, \sigma_s, \sigma_r) - 2y_i + 1 \\ t_{di} &= BL(x_i, \sigma_s, \sigma_r) - 2t_i + 1 \end{aligned}. \tag{2}$$

Equation 1 is our objective (loss) function to be minimized, in which N represents the number of pixels, $BL(x_i, \sigma_s, \sigma_r)$ is the CNN input, x_i is the flash image, y_i is the predicted difference of the CNN, and $t_i = BL(x_i, \sigma_s, \sigma_r) - BL(o_i, \sigma_s, \sigma_r)$, where o_i, where o_i is the ground truth.

We normalized our target difference images into the range $[0 \dots 1]$ in order to avoid negative values affecting the CNN convergence given its activation functions. Then values are remapped into the range $[-1, 1]$ and subtracted from the input values. We performed the mean subtraction for each color channel of each image pixel by pixel, but only in the evaluation phase of the objective function. This operation was performed to distribute in a balanced manner the weights of each image across the training. In this way, each image gives the same contribution to the training, none more or less important than the others. In contrast to the classical method of subtracting from each image the mean computed across the whole training dataset, we subtracted the mean for each image to remove the average brightness from each pixel. This operation can be performed because our image domain consists of stationary data for which the lighting parameters are well-defined and always the same both for the input and for the output. For further details, see [9].

[6] https://en.wikipedia.org/wiki/Bilateral_filter

4 Dataset Creation

In order to produce a training set for our network, we acquired pairs of photographs of the same subject using the camera of a Google Nexus 6 smartphone at full resolution (i.e., 13-Megapixel). We captured one photograph of the pair using only the flash of the smartphone, and the other using a studio-like set of lamps that provides uniform illumination. In post-processing, we aligned each pair using the MATLAB Image Processing Toolbox^{TM}to minimize misalignments. These are caused by a delay of about 400ms between the two shots due to switching on and off the studio lamps. We set the non-flash image as the misaligned one and the flash image as our reference, then ran a tool for affine alignment, i.e., translation, rotation, scale, and shear. Since photographs in each pair have different lighting conditions, we had to use a multi-modal optimizer to align two images using intensity-based registration. In a few cases, misalignments persist when one image has the subject with open eyes and the other with closed eyes or vice-versa. In such cases, the worst images were removed from the dataset.

After the alignment step, we identified the subject face by running a simple face recognition API [7]. This outputs a bounding box for a photograph that we used to crop each image, which is finally downsampled to 512×512. During our acquisition process, we managed to collect 495 pairs of photographs. These pairs represent 101 people (both females and males) in different poses. In order to have a larger dataset, we augmented this set using three common techniques:

- 5 rotations from -20 to $+20$ degrees around the center of the face bounding box, using a 10 degree step;
- cropping the image to the face bounding box and rescaling to original image size;
- flipping images horizontally.

These operations increased the original dataset by a factor of 20, obtaining a training set of 9.900 images at a 3120×4160 resolution (13-Megapixel).

5 RESULTS AND DISCUSSION

We trained our CNN on pairs or 512×512 images for about 5 days on an NVIDIA Titan Xp GPU, performing 62 epochs and about 458,000 backpropagation iterations. We interrupted the training when the value of the loss function computed on 1,500 images reached a low level of approximately 0.0042.

We calculated the accuracy of the result as

$$acc = 100 - \left(\frac{100}{3w(I)h(I)} \sum_{i} \sum_{c} \mid I_c - \tilde{I}_c \mid \right), \tag{3}$$

where $I = t_d, \tilde{I} = y_d, w(I) = width(I)$, and $h(I) = height(I)$. After the training step, we obtained an accuracy value of 96.2%. In the test phase, we evaluated our approach using 740 test images, obtaining a loss-test value of 0.0045 and an accuracy value of 96.5%. The evaluation was done on the dataset provided from Y. Aksoy et al. [7], on images such as those shown in Figure 1, 3 and 4.

[7] https://github.com/ageitgey/face_recognition

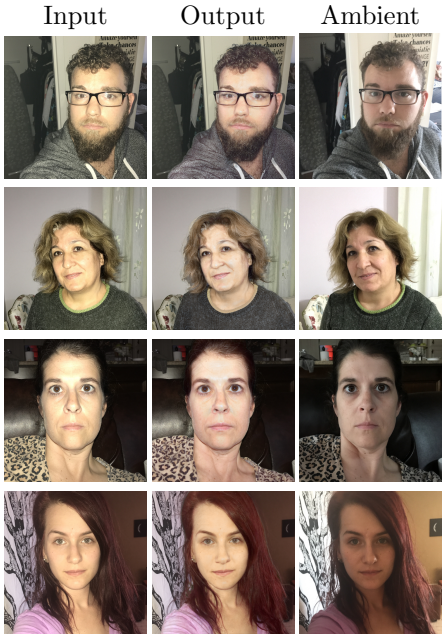

Figure 3: Results of our approach on real selfie images of the dataset provided by Y.Aksoy et al. [7]. The first column represents the input of our neural network, the column center represents our result and the last column represents the no flash ambient image.

5.1 Comparison with Reconstructed Ground Truth Images

One key idea of our technique is to train the CNN to learn the difference between the bilateral filtered target and input images. The output of the pipeline then subtracts the CNN prediction from the original input image. This allows us to preserve the high frequency detail, even though the exact ground truth can't be reconstructed exactly, even with a 0 loss function.

But of greater concern are the misalignments due to pose changes between the flash and non-flash photos, which would otherwise dominate when computing the input and output image differences. For these reasons, we introduce a preconditioning operator on the ground truth:

$$\overline{o}_i = x_i - 2t_i + 1 \tag{4}$$

where t_i is the target difference. This operator represents the reconstructed ground truth in which some of the high frequencies lost through the bilateral filter and not recoverable were not considered.

We show an excerpt of the validation data in Figure 5. Note how hairs, beard and skin color are lost in the flash photo and restored in our results. Also, shadows and highlights due to the flash are mitigated. We evaluated the data by employing the Structural Similarity Index (SSIM) and saw results ranging from 78 to 91%, indicative of relatively good agreement.

Figure 4: Our approach can be used on people with different features and ethnicities. Although the flash highlights remain evident in the lenses of people with glasses, they do not affect our approach to the rest of the image.

5.2 Comparisons with Other Techniques

We compared our results against three other approaches in the literature, see Figure 6. The first approach is HDRNet by Gharbi et.al [12] and is based on the use of a CNN combined with bilateral grid processing and local affine color transforms. HDRNet is designed to learn the effect of any image operator and hence is a suitable candidate to remove flash artifacts from photographs. The second approach is Pix2Pix by Isola et al. [13], which is based on a "conditional" Generative Adversarial Network (cGAN) for which image generation is conditional on the type of image. This type of neural network was investigated as a general-purpose solution to image-to-image translation problems. Isola et al. tested their cGAN on different tasks such as photo generation and semantic segmentation. The third approach is the style transfer method proposed by Shih et al. [14] in which a multi-scale local transfer approach is applied to portraits. The last two columns of Figure 6 also shows comparisons between the ground truth images and reconstructed predictions.

We feel our technique compares favorably with these other methods in that it makes the lighting uniform, removing the flash highlights without introducing problems like altering geometries and blur effect.

Figure 5: Validation dataset samples where the SSIM was computed through central images and top right images. Such a dataset as well as the training dataset, consists of images taken to approximate real selfie images using a smartphone and smartphone flash at a similar distance and angle of a real selfie. Top left images are the flash images taken with the smartphone; bottom left images are the filtered images with bilateral filter; centered images are the results of our approach; top right images are reconstructed ground truth images and finally, the bottom right images are the original ground truth images.

6 Conclusion

In photography, glare is a common issue that causes shiny highlights, especially in portraits. In the majority of cases, glare is a defect, and the subjects seem to be greasy. Glare can be removed from the face manually, but this involves a complicated and uncertain result process that requires photo editing skills.

This paper proposes a technique that is able to dramatically increase the quality of smartphone flash selfies by turning them into portraits with studio-like lighting. The approach is able to automatically remove flash lighting artifacts such as hard shadows and highlights by using a regression model based on supervised learning. Our results confirm that learning-powered computational photography is capable of highly effective lighting control, suggesting that it might be valuable in other contexts like relighting for better presentation of objects or for advanced shading removal in photogrammetric reconstructions.

References

[1] G. Petschnigg, R. Szeliski, M. Agrawala, M. Cohen, H. Hoppe, and K. Toyama, "Digital photography with flash and no-flash image pairs," *ACM Trans. Graph.*, vol. 23, no. 3, pp. 664–672, Aug. 2004.

[2] E. Eisemann and F. Durand, "Flash photography enhancement via intrinsic relighting," *ACM Trans. Graph.*, vol. 23, no. 3, pp. 673–678, Aug. 2004.

Figure 6: An example of comparisons among our method and HDRNet, Pix2Pix, and Style Transfer. Note that our method keeps the geometric features of the input images; this makes the lighting uniform.

[3] A. Agrawal, R. Raskar, S. K. Nayar, and Y. Li, "Removing photography artifacts using gradient projection and flash-exposure sampling," *ACM Trans. Graph.*, vol. 24, no. 3, pp. 828–835, Jul. 2005.

[4] X. Zhang, R. Ng, and Q. Chen, "Single image reflection separation with perceptual losses," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, June 2018, pp. 4786–4794.

[5] E. Gabriel, K. Joel, D. Gyorgy, M. Rafał, and U. Jonas, "Hdr image reconstruction from a single exposure using deep cnns," *ACM Transactions on Graphics (TOG)*, vol. 36, no. 6, 2017.

[6] C. Chen, Q. Chen, J. Xu, and V. Koltun, "Learning to See in the Dark," 2018.

[7] Y. Aksoy, C. Kim, P. Kellnhofer, S. Paris, M. Elgharib, M. Pollefeys, and W. Matusik, "A dataset of flash and ambient illumination pairs from the crowd," in *Proc. ECCV*, 2018.

[8] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9*, 2015.

[9] N. Capece, F. Banterle, P. Cignoni, F. Ganovelli, R. Scopigno, and U. Erra, "Deepflash: Turning a flash selfie into a studio portrait," *Signal Processing: Image Communication*, vol. 77, pp. 28 – 39, 2019.

[10] O. M. Parkhi, A. Vedaldi, and A. Zisserman, "Deep face recognition," vol. 1, 01 2015, pp. 41.1–41.12.

[11] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, 2010, pp. 249–256.

[12] M. Gharbi, J. Chen, J. T. Barron, S. W. Hasinoff, and F. Durand, "Deep bilateral learning for real-time image enhancement," *ACM Transactions on Graphics (TOG)*, vol. 36, no. 4, p. 118, 2017.

[13] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," *CVPR*, 2017.

[14] Y. Shih, S. Paris, C. Barnes, W. T. Freeman, and F. Durand, "Style transfer for headshot portraits," *ACM Transactions on Graphics (TOG)*, vol. 33, no. 4, p. 148, 2014.