CSCI 1430 Final Project Report: Portrait Image Relighting

Adam Pikielny Ben Givertz Jack Dermer Aalia Habib
Brown University
May 2020

Abstract

Traditional methods for relighting faces require knowledge of the subject's reflectance, lighting, and structure. We sought to implement a deep learning algorithm to solve this task and relight portraits given only a single image as input. We implemented an hourglass-shaped CNN from research by Zhou et al. [5] in order to relight portrait images. The model first separates the input image into facial and lighting features, from which a specialized lighting network predicts the direction of light. Then, the facial features are combined with the desired new lighting. Using a synthesized data set of portrait images under various artificial lighting conditions for training and ground truth, we were able to achieve realistic results, outputting images at a resolution of 256×256.

1. Introduction

Image relighting is a challenging and fascinating problem in computer vision without a clear solution. Within the field of image relighting, we focused on faces (portrait relighting). This involves calculating the structure and lighting of a face, determining its reflectance, and finally determining how the image would look under new conditions. Generally, this would require many cameras in order to determine the structure of the image. We sought to implement a model that would be able to relight faces given only a single input image and target lighting condition. We believe this technology will be immensely helpful for film and photography, from video editors who need to relight a scene after it is filmed, to amateur photographers who want to edit their photos or paste someone into a new image.

Relighting images is a difficult task for multiple reasons. First, it requires knowledge of the 3D structure of the scene to be relit. Second, it requires knowledge of the reflectance of each object in the image. Thus, many current implementations require tens or hundreds of photos as input along with meticulously calculated lighting conditions. We reduced this challenge by focusing only on faces. By doing this, we were able to assume Lambertian (matte) reflectance, and our model was able to learn a general 3D structure for faces.

2. Related Work

We read several relighting papers to determine the best direction for us to proceed. We originally found a research paper by Microsoft Research on scene relighting using deep learning, which sparked our interest in the topic [1]. The researchers used a neural network to calculate the light transport matrix of a scene using hundreds of images under various lighting conditions in order to relight it. Unfortunately, we did not have the computing resources to implement their findings ourselves. We also read a paper on image relighting from optimal sparse samples by Xu et al. [4] and another on object background relighting by Toisoul et al. [3], but eventually determined the challenge of relighting arbitrary scenes was too difficult given our limited resources.

We found a paper by Sun et al. [2] to be very assuring; it was able to successfully relight faces from single images with absolutely no prior knowledge of reflectance or geometry of the face. However, the data collection process required an advanced camera rig that we couldn't recreate.

Eventually, we settled on *Deep Single Image Portrait Relighting* by Zhou et al. [5]. Similar to the one above, it focused only on faces, which we found to be a convenient division of the original task. The authors kindly released their synthesized data set of 1024×1024 high resolution portrait images under various lighting conditions, which allowed us to implement their model.

3. Method

In order to implement portrait relighting, we trained a deep neural network with the architecture provided by [5]. Both the training data and strategy had to be altered from the original paper to fit our constraints on storage space and computing power.

3.1. Dataset

Zhou et al. were able to extract the ”normal” (3D reconstruction) of the faces in their data set in order to create relit images. We used these as our ground truth examples of target

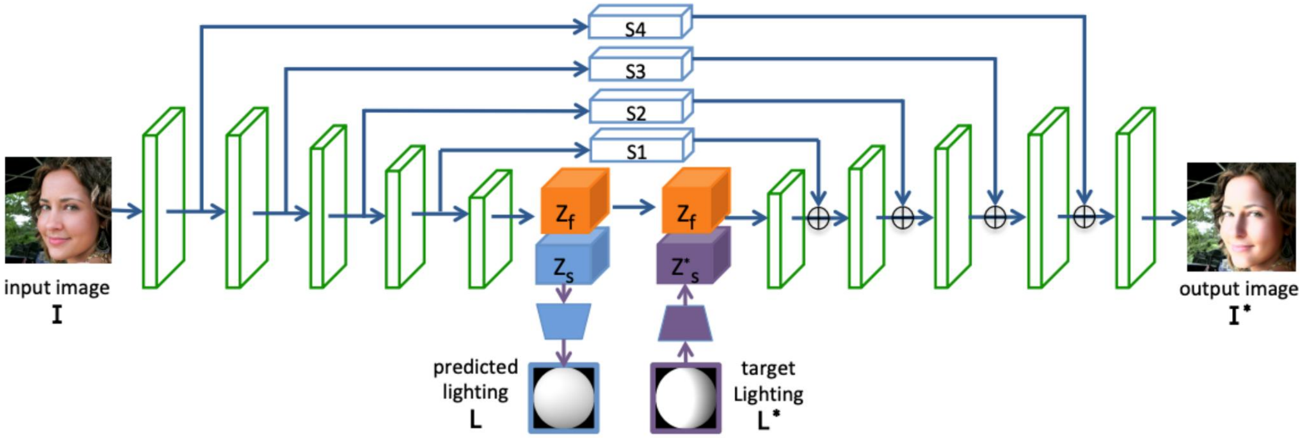

Figure 1. CNN architecture model from Zhou et al.

and predicted lighting applied to input images.

The data set linked to in their paper contained 30,000 images and their corresponding lighting matrices each under 5 lighting conditions, totaling upwards of 250 GB of data. After attempting various techniques to move the data into our Google Cloud instance, we determined it was too large to for us to manipulate or use. Thus, we decided to scale down the data set to reduce the size. Concerned about how this might affect our results, we emailed the authors of the paper, and they promptly responded, agreeing that scaling down the data set was a good idea, and further suggesting that we not implement their GAN loss for ease of training. We wrote a script to scale each image down to 128×128. This decreased our data size to around 500 MB, which was exceedingly easier for us to work with. It was seamless to upload and run this data on Google Cloud Platform (GCP). After achieving successful results in our model, we decided to create a 256×256 data set, which immensely increased the quality of our results.

In our original training sessions, the entire data set was loaded into memory before training began, a process which took around 30 minutes each time. We then transitioned to using an index based data loader provided by PyTorch which handled multi-threaded CPU data loading as well as random batching. Once implemented, we were able to efficiently train our model with batches as inputs instead of single images. Implementing this removed the 30 minute data loading period and decreased epoch training time from 12 minutes per epoch all the way down to 2 minutes.

3.2. Computation

We decided to use GCP to run our model, creating a PyTorch environment instance with a CUDA GPU. Data was saved on GCP, so it was quick to load. We also used OpenCV for image manipulation, and Kornia to calculate images gradients.

3.3. Model Architecture

We followed the hourglass model architecture of Zhou et al., as shown in figure 1. The image, I, is first passed through a series of convolutions with max pooling, batch normalization and ReLU activation. Eventually, it is split up into Z_f and Z_s, corresponding to an encoding of the facial features and lighting respectively. Next, Z_s is passed through a lighting network, and the predicted lighting, L is output. The target lighting, L^*, is passed in, encoded to Z_s^* and joined with Z_f for the second half of the network. The blue connections running across the top of the figure are skip connections. These connections pass the output of a layer to the size-corresponding layer input on the other side of the model. In the original paper skip connections were used in specific epochs to improve results; however, upon debugging we realized they worsened our results, so we decided to remove them for our final model. After more convolutions, the network returns the output image, I^*, which along with our predicted source lighting L, is used to calculate loss.

3.4. Loss

Our loss function was a slightly modified version of the function found in the source paper. As seen in equation (1), the loss is combination of the L_1 loss taken from the source image and its gradient as well as an averaged version of the L_2 loss of the predicted lighting, where N_p is the number of pixels per image, and N_l is the number of lighting channels:

$$L_1 = \frac{1}{N_p}(||I_s - I_s^*||_1 + ||\nabla I_s - \nabla I_s^*||_1) + \frac{1}{N_l}(L_s - L_s^*)^2 \tag{1}$$

In contrast to the original paper, our loss function includes the N_l term, which averages the lighting loss across its channels. Before adding this term, the L_2 lighting loss was too large in proportion to the image loss and thus the model

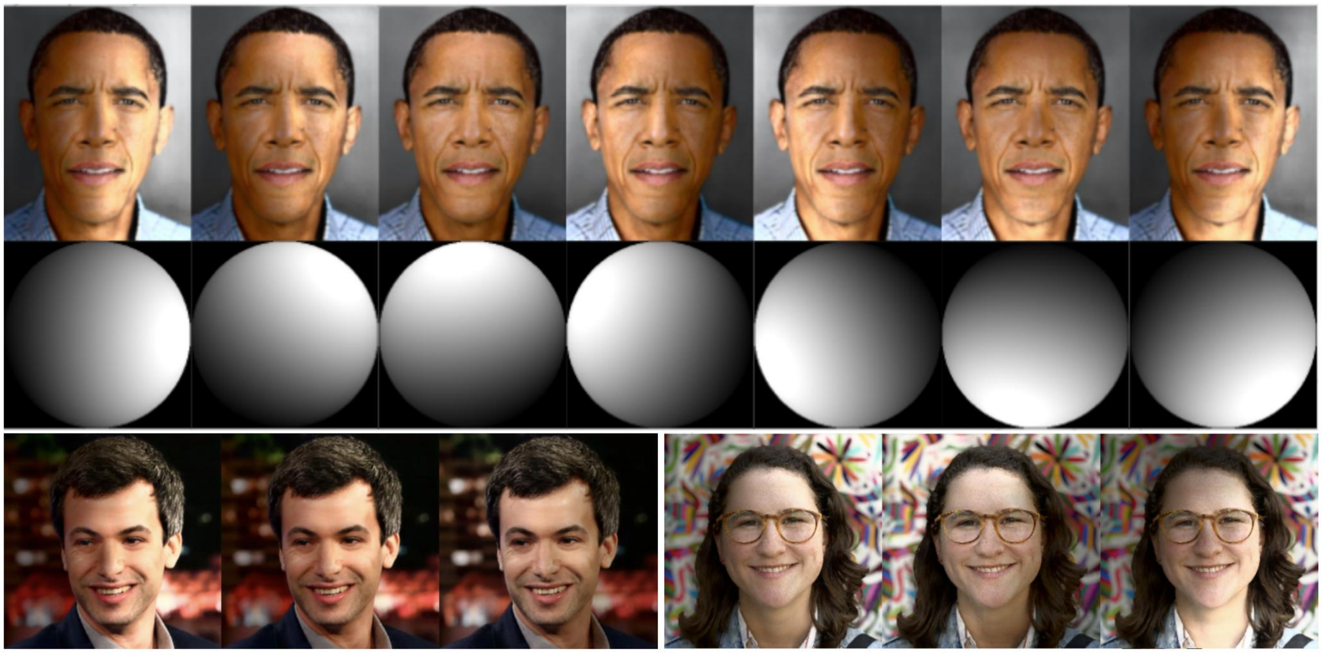

Figure 2. 256 × 256 relit portrait images under our model and corresponding lighting conditions.

was heavily over-training to predict lighting instead produce novel lighting conditions.

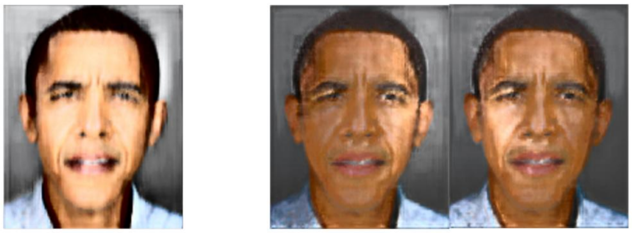

Figure 3. *Left:* Output with unscaled loss, *Right:* Output without corrected loss.

4. Results

Our final model, trained on 256 × 256 images for 20 epochs gave us the results above. We evaluated our success by taking the average mean squared error between our outputs and the outputs of the model in the paper on 50 portrait images. We chose to use the output of the model as our ”ground truth” because true relit images that matched the target relighting exactly would have been very hard to produce. We found that our output varied, on average, by 2.88%.

The primary differences between our model and theirs include: the use of a facial-feature-loss, training on different image sizes, and the removal of skip layers. We chose not to implement the feature loss because we were unable to determine the best way to extract facial features and wanted to focus more on relighting. We trained on 256 × 256 images instead of 1024 × 1024 in order to upload them to GCP and train in a reasonable amount of time.

Our choice to remove the skip layers resulted from experimentation. In figure 4 we show the output using skip layers (every relit image output was the same) and the output once we trained without them. We hypothesize that the reduced size of our inputs to the model was incompatible with the skip layer architecture.

As evidenced in the above images, there are some artifacts that appear in the cases of extreme lighting (on the leftmost and rightmost images of Obama, they are the most evident), especially near the bridge of the nose. We believe the feature loss would have helped smooth them out in order to maintain shading differences that characterise certain facial features.

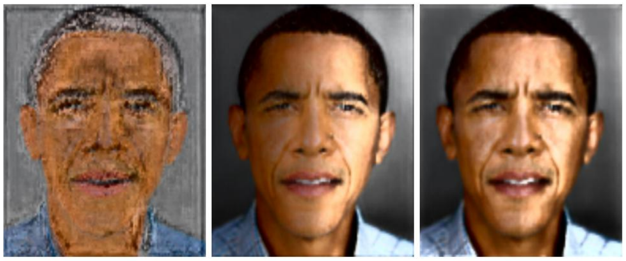

Figure 4. Progression of epochs under incremented skip layers.

4.1. Discussion

An interesting outcome of our implementation is the fact that in order to mimic the state of the art image relighting using much smaller training data, we had to remove the skip layers. Since skip layers are typically used to preserve infor-

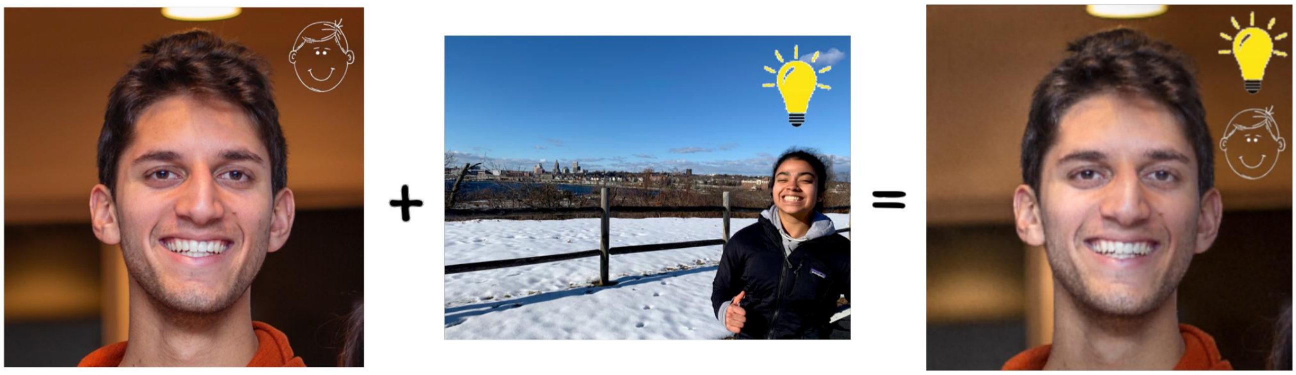

Figure 5. The lighting from the second image is transferred to the portrait.

mation that is lost during convolution or pooling and passed along to deeper layers, it is surprising that removing them improved our results. Since we did not modify the parameters of convolutional layers, we believe that the outputs of each individual convolutional layer did not contain enough information to feed into another high level layer directly. Thus, this resulted in a model that was unable to extract or insert lighting information. We hypothesize that the skip layers were actually propagating some noise and removing them allowed our model to train better on the smaller images.

We also opted not to implement the L_f feature loss, which compared extracted feature points of the images. We hypothesized that this loss would not be beneficial for the same reason that skip layers were not. The smaller resolution would not allow for proper feature identification. We can see in our data that the nose is often the best way to tell that our images are artificial. This may have been fixed with feature loss and higher resolution images.

5. Applications and Augmentations

After finishing our model, we were able to save our weights and run a testing script on our local machines to quickly relight images. This opened up a world of possibilities for possible applications.

5.1. Image Combination

Our model conveniently outputs the predicted lighting of the input image. We realized we could use this to extract the lighting of one face and add it as the target for another. We implemented this script and it worked quite well immediately. However, we realized that in the case of input images where the face was not prominent or centered in the frame, it was difficult to extract lighting. We utilized a simple bounding box face classifier implemented in OpenCV to crop our inputs for the network. Unsurprisingly, cropping the relit image did not work very well because upon uncropping, the lighting did not fit with the background and our model was unable to account for relighting other parts of the scene. However, cropping the lighting image successfully isolated the face and helped us achieve better results. In figure 5, the face in the lighting image is relatively small but our network still successfully extracts and applies its lighting.

5.2. Python GUI

We created a Python GUI that applies face relighting from one image to another. This is intended for people who do not have experience with Python and may have trouble running scripts through a command line. The GUI is meant to be used as a supplement to photo editing applications such as Adobe Photoshop. The user can upload a cropped image of a face they are trying to insert into a group picture, as well as the group picture for lighting reference. We utilized a face detection CNN to find faces in the group picture. The user selects a face from the group picture that most clearly represents the desired light. Then, the program saves the new, relit image to a location of the user's choosing.

5.3. Photoshop Plugin

We also would like to implement a Photoshop plugin to aid in relighting faces. We began to work on it but found that it required extensive knowledge of C++ and the Xcode editor. We are currently looking for a way to export our python code to a binary that can be easily integrated with the Photoshop API.

5.4. Live Webcam

We used OpenCV to write a Python script that connects to the webcam and runs the relighting network on each frame. Because the model is large and requires a lot of computational power, running this script on a 2018 Macbook Pro results in a frame rate of approximately 4 fps, and the relighting has a few artifacts between frames.

6. Conclusion

We believe that our portrait relighting implementation was very successful. The biggest challenge was working with and trying to understand the hourglass neural network architecture. Through trial and error, we were able to better understand skip layers and lighting in images as well as different loss metrics. We foresee that our extensions of the functionality, especially using the model to transfer lighting between images, could potentially be useful in image editing software. Overall, we are excited that our model performs similarly to the state of the art results of our source paper and are interested in exploring further applications.

References

[1] P. Ren, Y. Dong, S. Lin, X. Tong, and B. Guo. Image based relighting using neural networks. 2015. 1

[2] T. Sun, J. T. Barron, Y. Tsai, Z. Xu, X. Yu, G. Fyffe, C. Rhemann, J. Busch, P. E. Debevec, and R. Ramamoorthi. Single image portrait relighting. 2019. 1

[3] A. Toisoul and A. Ghosh. Image-based relighting using room lighting basis. 2016. 1

[4] Z. Xu, K. Sunkavalli, S. Hadap, and R. Ramamoorthi. Deep image-based relighting from optimal sparse samples. *ACM Transactions on Graphics (TOG)*, 37(4):126, 2018. 1

[5] H. Zhou, S. Hadap, K. Sunkavalli, and D. W. Jacobs. Deep single portrait image relighting. 2019. 1

Appendix

Team contributions

Adam Pikielny I worked on the training function and developing our modified loss function. After we successfully relit faces, I worked on the face-to-face relighting and implemented our face detection features to improve the model.

Ben Givertz I worked on batching and training efficiency, as well as data preprocessing to scale down the initial data set. I also worked on the loss function, and after we were able to train models quickly, I trained various models by tweaking the hyperparameters of the system.

Jack Dermer I worked on data preprocessing, specifically researching pytorch Datasets and Dataloaders. I also worked on batching and skip layers. Lastly, I developed a GUI using python tkinter, our network, and a face detection CNN.

Aalia Habib I worked on data preprocessing and reworking the loss function as well as implementing the live image relighting functionality. I also attempted to integrate the network with the Photoshop API but was unsuccessful.