CS230

Realistic Transfer of Lighting to an Indoor Scene

Image Processing

Natasha Ong
Department of Computer Science
Stanford University
natashao@stanford.edu

Kevin Su
Department of Computer Science
Stanford University
ksu20841@stanford.edu

GitHub: https://github.com/kevin20841/CS-230-Final-Project

Problem Statement

The quality and appearance of an image is largely dependent on lighting. The complex interplay of objects and light (type of light, directions, etc) is central to the appearance of objects, thereby impacting the quality and realistic appearance of an image. Thus, when photoshop artists or VFX artists want to embed an object from one image into another, lighting effects (including reflectivity, shading, etc) often becomes an issue. As such, just cropping the object itself is not enough to make this new image appear realistic. This tool attemps to remove the additional work that many visual effect artists and photo editors have to put in to combine images with different lighting. The intended outcome is a generative model that properly applies the lighting in a surrounding image to our object. To capture a wide range of lighting, we discretize the lighting to 25 different directions.

Related Work

The problem of changing lighting effects has been explored by many others in recent years, though there are not many that have tackled our specific problem of indoor lighting effects of various images. Conceptually, in 2003, Hara, Noshino, and Ikeuchi researched various techniques to recover reflectance properties of real surfaces under unknown illumination conditions [2]. Though they did not work specifically on style transfer, they proposed methods to estimate the surface reflectance property of an object, as well as the position of a light source from a single image.

More recently, many researchers have tackled variations of the lighting problem like flash effects. They have created models that take images with flash and output an identical image without the flash effects [1][8]. However, these models do not cover a wide range of indoor lighting by only focusing on flashed lighting.

Neural Style Transfer

Hold-Geoffroy et al. have attempted to transfer various outdoor lighting onto a content image [6]. Specifically, they used a CNN-based technique to estimate high dynamic range outdoor illumination, training their model with outdoor panoramas and sky images and parameters (including sun position, atmospheric conditions, and camera parameters). This research varies from ours due to the differnece in scenes (indoor vs outdoor) and the different aspects of lighting effects that need to be accounted for in the different scenes.

CS230: Deep Learning, Spring 2020, Stanford University, CA. (LateX template borrowed from NIPS 2017.)

Generative Adversarial Networks (GANs)

Generative Adversarial Networks (GANs) have been used widely in producing realistic images, where image generation occurs as a zero-sum game between a generator neural network and discriminator nerual network [5]. Specifically, StyleGan has been used to produce and customize some of the most realistic faces from a generative model to date. Unlike regular GANs, StyleGAN starts at a learned constant and continuously adds in "style" at multiple points during the generation process [6]. Repeatedly injecting the style vector into the network results in the style vector having a greater impact on the final image. On the other hand, CycleGAN uses cycle-consistency loss to enable training without paired images, ultimately trying to have a model minimize reconstruction loss [10].

Dataset

We used a dataset developed by researchers at MIT: "A Dataset of Multi-Illumination Images in the Wild" [7]. This dataset consists of 1016 real interior scenes, each captured under 25 lighting conditions using indirect illumination and an electronic flash mounted on servo motors to control direction. Each image is 1500x1000 px. We scaled each image down during training to 256x256 px.

We are mainly focused on the transfer of lighting effects of images of objects rather than people (where aspects like skin tone become important), so this dataset matches our needs as all images are those of objects/indoor scenes. In particular, the dataset spans 95 different rooms throughout 12 residential and office buildings, thereby capturing a variety of materials and room shapes. As such, it is representative of many real world scenes.

This dataset also includes segmentation for analysis of materials, though we disregard these sections since we mainly focus on the differences in lighting directions and effects (ex. flash, no flash).

Methods

We try a modified LSGAN and a modified CNN UNet architecture so we can learn lighting effects that result from each of the 25 lighting directions and can also generate new images with the lighting direction desired.

The GAN model architecture is comprised of a generator and a discriminator model:

1. G for generating images for the first domain. It is conditioned on the input image and the target lighting.
2. D: Discriminates G_{gen} from real data. It is conditioned in an input image and a matching lighting.

The generator utilizes average pooling for downsampling, and pixel shuffling for upsampling. In addition, we implement skip connections as in UNET.

We train the generator using Adam optimization with mean squared error, $lr = 0.0001, \beta_1 = 0.5, \beta_2 = 0.999, \varepsilon = 1e-07$). We train the discriminators using Adam optimization with mean squared error, $lr = 0.0003, \beta_1 = 0.5, \beta_2 = 0.999, \varepsilon = 1e-07$). Note the difference in learning rates, we implemented TTUR. In addition, we implemented label smoothing and noisy labels: making labels 0.1 and 0.9 instead of 0 and 1, and flipping labels fed to the discriminator with probability 0.05. Our batch size was 20, and we utilized BatchNorm.

The CNN model architecture is comprised of a modified UNET architecture. It is a similar architecture to our GAN generator, except it utilizes maxpooling instead of average pooling.

Loss Function

For the GAN, our discriminator is trained directly on real and generated images. For the discriminator, we use a L2 loss (mean-squared error).

$$\mathcal{L}_{adv}(G, D, A) = \frac{1}{m} \sum_{i=1}^{m} \left(1 - D(G(a^{(i)}))\right)^2$$

For the CNN, we use a L1 loss (mean-absolute error) since we found it to be more stable in the long run that way.

$$\mathcal{L}(Mod, In, tar) = \frac{1}{m}\sum_{i=1}^{m} \|Mod(In) - tar\|$$

Other Architectures explored

We explored using neural style transfer, and a cycle GAN model. The cycleGAN model was too cumbersome and did not fit our problem formulation well. The neural style transfer was ill suited to our problem as well.

Hyperparameter Tuning

Most of our hyper-parameter tuning had to do with our GAN and CNN. For training our GAN, we often ran into mode collapse or diminished gradient and found that our model's stability was very sensitive to minor changes in our hyperparameters. As such, we tried to balance the generator and discriminator by monitoring the losses and sample generated images. In particular, we focused on fine-tuning the learning rate of the generators/discriminators, batch size, and latent dimensions of our discriminators and generators, etc.

GAN Hyperparemeters	**Value**
Number of Epochs	400
Batch Size	20
Discriminator Learning Rate	0.0003
Generator Learning Rate	0.0001
Discriminator Latent Dimension	200
Generator Latent Dimension	300
Discriminator Optimizer	Adam
Generator Optimizer	Adam

For our CNN, we focused on stabilizing our loss function and balancing run time and accuracy.

CNN Hyperparemeters	**Value**
Number of Epochs	400
Batch Size	20
Learning Rate	0.0002
Latent Dimension	200
Optimizer	Adam

See our code for the full list of all the hyperparameters for the different training models.

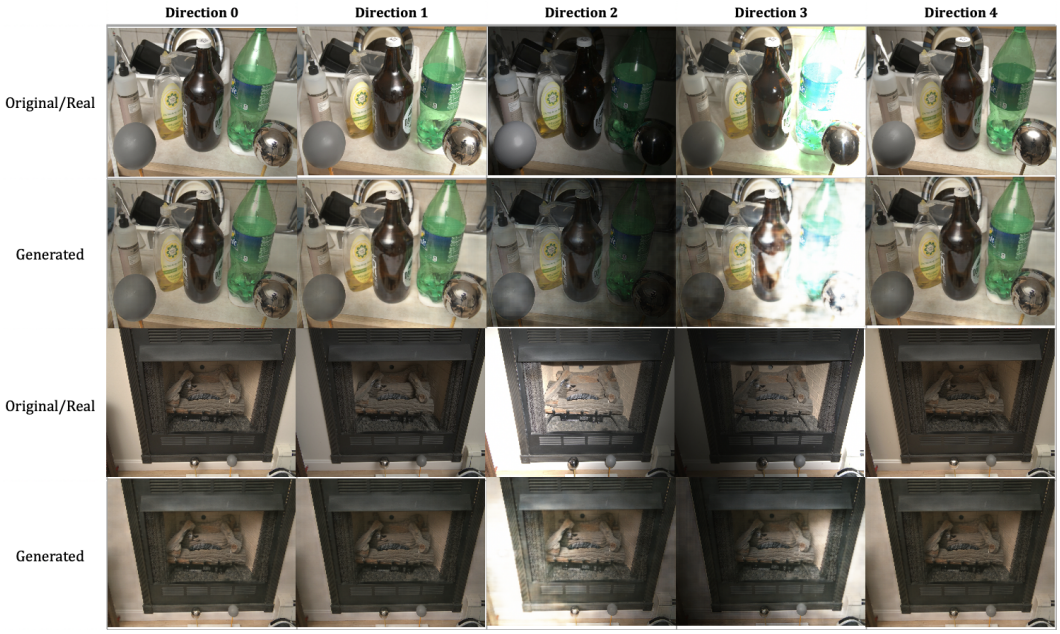

Figure 1: Comparison of real vs generated images of 5 of the 25 lighting directions for our CNN.

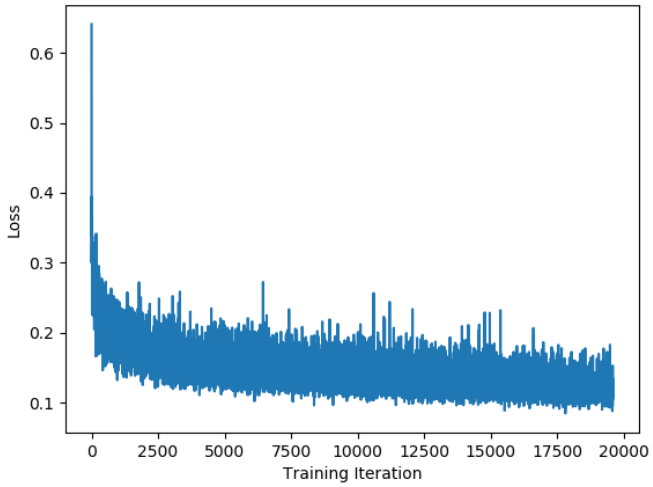

Figure 2: L1 losses for our CNN

Results

Unfortunately, we could not get our GAN training to both stabilize and produce satisfactory results. When the GAN did stabilize, it entered mode collapse and just outputted the input image. Thus, the results here are focused on the that obtained from the CNN (lighting probe model). Our model was definitely converging, as seen through the overall decreasing losses in figure 2.

In terms of the images more specifically, our model seems to work well in transferring diffuse light. We can see how the lighting directions changing effects the objects generally brightness in all of the images. It is also relatively successful in transferring flash effects as see through direction 3 in the first image and direction 2 in the second image. Moreover, it does not over-expose the image significantly more than the original.

However, we notice that the specular highlights do not transfer well, such as reflectance. For example, when we look at the bottles in the first image with lighting direction 4, we can see that while the original image's bottles do not reflect light (glass and plastic bottles) the generated images' do. For direction 1 of the first image, we also see that the reflection of light for the glass and plastic bottle exist for both the generated and real images. However, the directions do not correspond correctly. The original image evidently has light coming in from the right (and thus reflecting in that direction) while the generated image simulates lighting from the left where the specular highlights on the bottles appear on the left side.

Conclusion/Future Work

To summarize, our CNN performs well in capturing general diffuse lighting and flashed effects but fails to perform well with specular highlights. This is likely due to the lighting properties of different objects, which we did not train on or really take account of. In short, we underestimated the impact that a material's object would have on its appearance under different lighting conditions. As such, in future work, we should definitely incorporate image segmentation based on material to be able to capture more complex lighting properties like reflectance/specular highlights.

Moreover, in future works, we would spend more time to train our GAN such that it is more stable across the various iterations/epochs. In particular, we would design it such that it does not run into Mode Collapse.

References

[1] Capece, Nicola, et al. "Deepflash: Turning a flash selfie into a studio portrait." Signal Processing: Image Communication 77 (2019): 28-39.

[2] Gardner, Marc-André, et al. "Learning to predict indoor illumination from a single image." arXiv preprint arXiv:1704.00090 (2017).

[3] Hara, Kenji, Ko Nishino, and Katsushi Ikeuchi. "Determining reflectance and light position from a single image without distant illumination assumption."IEEE, 2003.

[4] Hold-Geoffroy, Yannick, et al. "Deep outdoor illumination estimation." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2017.

[5] Goodfellow, Ian, et al. "Generative adversarial nets." Advances in neural information processing systems. 2014.

[6] Karras, Tero, Samuli Laine, and Timo Aila. "A style-based generator architecture for generative adversarial networks." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2019.

[7] Murmann, Lukas, et al. "A Dataset of Multi-Illumination Images in the Wild." Proceedings of the IEEE International Conference on Computer Vision. 2019.

[8]"Neural Style Transfer: TensorFlow Core." TensorFlow,`www.tensorflow.org/tutorials/generative/style_transfer?hl=ko`.

[9] Xie, Michael, et al. "Transfer learning from deep features for remote sensing and poverty mapping." Thirtieth AAAI Conference on Artificial Intelligence. 2016.

[10] Zhu, Jun-Yan, et al. "Unpaired image-to-image translation using cycle-consistent adversarial networks." Proceedings of the IEEE international conference on computer vision. 2017.