Learning to See in the Dark

Paper Presentation Vidhey Oza April 22, 2021

Agenda

01. Introduction

02. Data Exploration

03. Implementation

04. Results

05. Conclusion & Future Work

Introduction

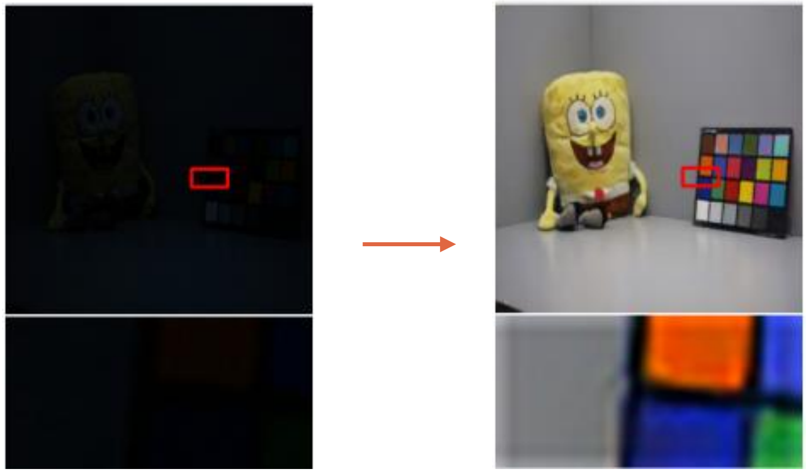
- Capturing low-light imagery difficult for daily-use devices (smartphones, GoPro, …)
- Multi-exposure techniques prone to shaking, bad processing
- AI-based techniques require high-quality datasets and light model footprint
- Fully convolutional generative models show strong potential

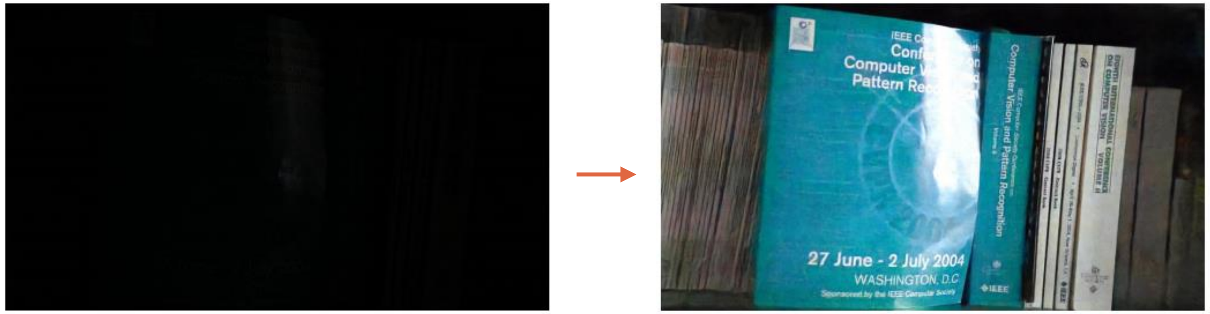
Cameraman capturing in dark forest

Low visibility on road cams at night

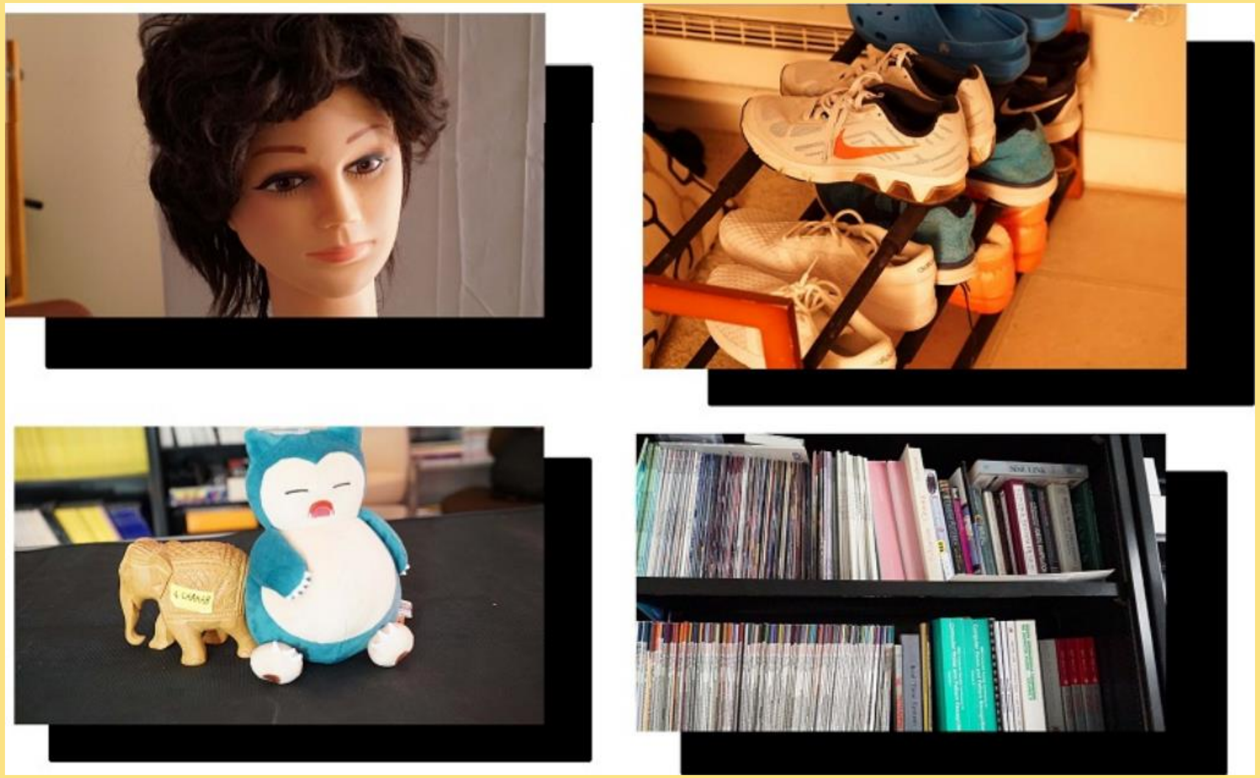
Problem Definition

- Given a low-light image, enhance it by dynamically adding light in the scene
- Supervised training implies dark-bright image pairs; difficult to find comprehensive dataset
- GAN-based training can take dark image input, produce illuminated version that can fool discriminator from “real” bright image
- Inspired by CVPR 2018 paper with same name





Data Exploration



Finding the right dataset

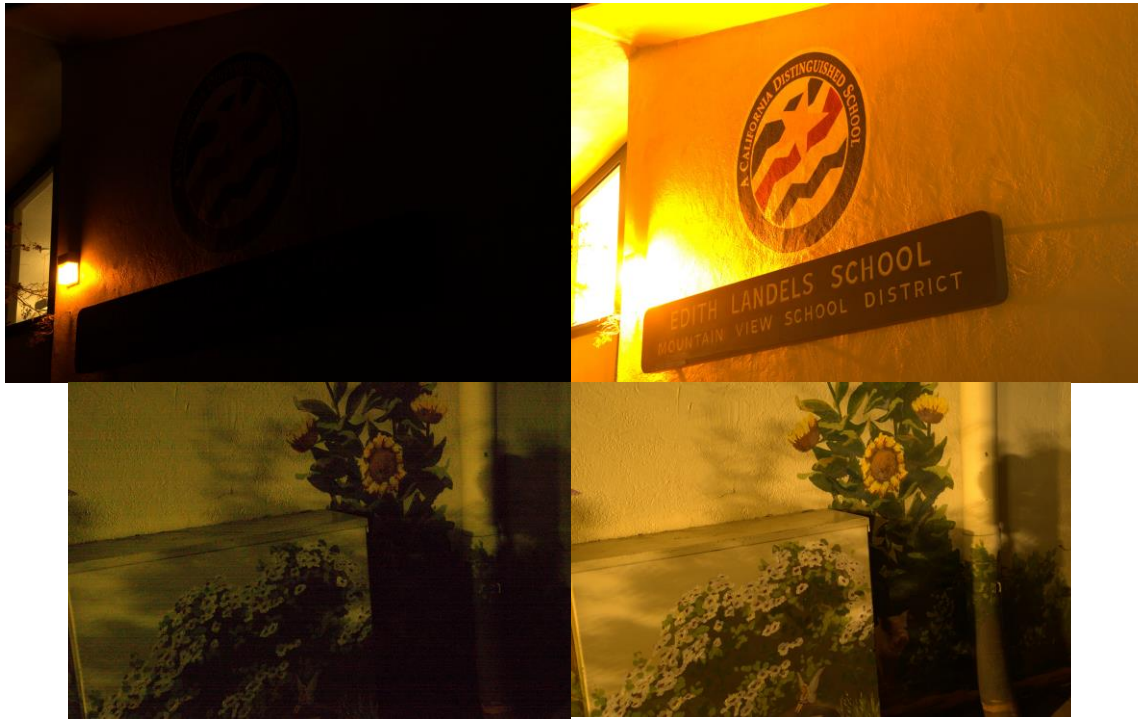
Datasets used by common methods

- Custom-made datasets, bespoke for proposed algorithms
- Singleton images generally – no bright counterpart for supervised training

Problem

- Imbalance in indoors vs outdoors samples
- Non-standardized capturing – significant variation in exposure, ISO etc.
- Variety in image types missing – resort to combined datasets (for e.g., EnlightenGAN)

Finding the right dataset



Keep in mind: RAW images converted to JPG for display, has its own rebalancing algorithm

See-in-the-Dark (SITD) Dataset

- RAW images from 2 cameras: Fujifilm and Sony
- 5094 dark-bright pairs of images
  - Dark images at 1/10s – 1/30s exposure
  - Bright at 10s – 30s exposure (~100x higher)
- Illumination for dark images: 0.03 lux indoors, 0.2 lux outdoors

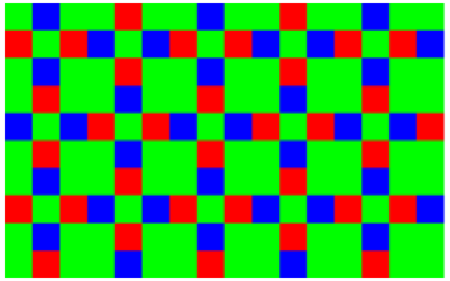
Problem of 4-channel images

RAW image formats are 4-channel

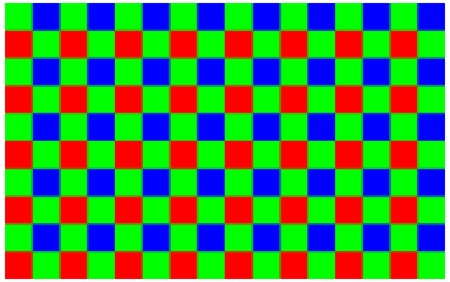
- Variants in SITD dataset use different filter arrays
  - Fujifilm – X-Trans sensor
  - Sony – Bayer sensor

1. Cannot train on entire dataset
2. Cannot use JPEGs

Filter arrays



X-Trans filter

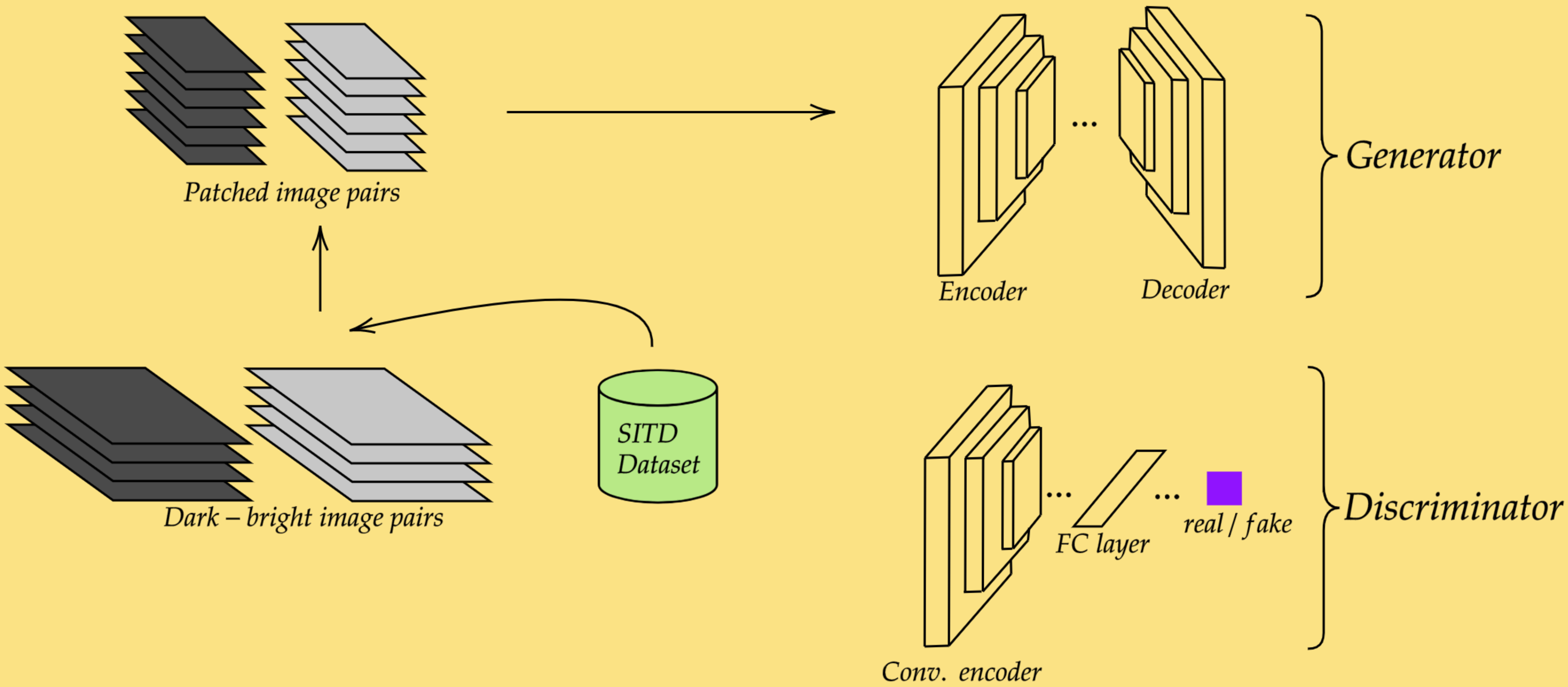


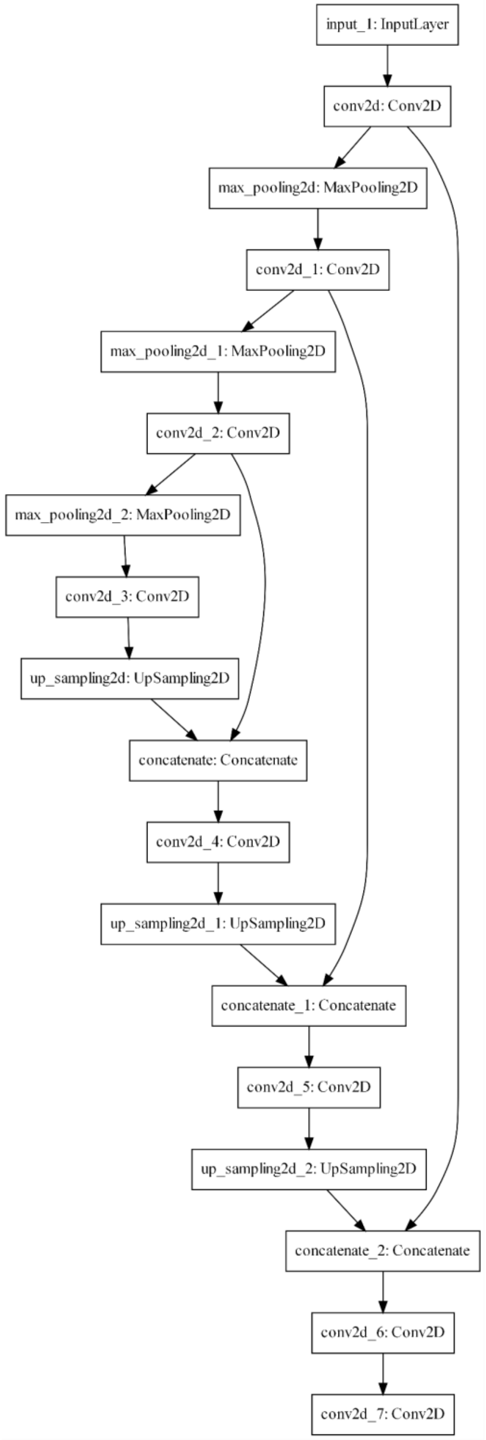
Bayer filter

Possible solutions

- Train 4-channel model – cannot be used for common JPG/PNG images
- Convert to 3-channel images – risk loss of color accuracy
- Full convolutions – risk losing model explainability

Implementation





Fully convolutional networks

- Use convolutional filters only – no fully connected layers
- Can retain image shape from input to output
- Can perform image-to-image training
- Can apply convolutions throughout the network for various layer designs
  - Residual connections
  - Inception-like modules

Basic GAN-based training

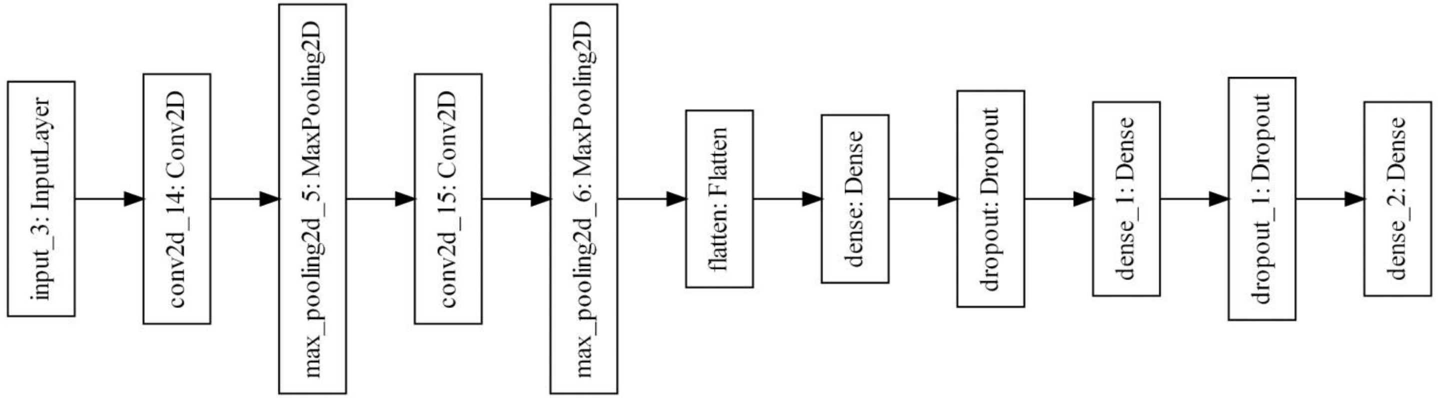
U-Net based generator

- Encoder-decoder architecture with 2 modifications for conv filters

1. Max-pooling for encoder, un-pooling for decoder
2. Skip connections before-max-pooling to after-un-pooling

Basic CNN classifier as discriminator

- 2 conv blocks, 2 fully connected blocks
- Conv blocks with max-pooling
- FC blocks with dropout



Problem of training on large images

Huge training load

- Images at ~1400x2100 leads to FCNs with >3M parameters – significantly more difficult to train
- Loading large model and large images together

Solution: Patching

- Convert large images into small patches for training
- Merge patches after output
- Caveat: unwanted noise at borders

Final model pipeline

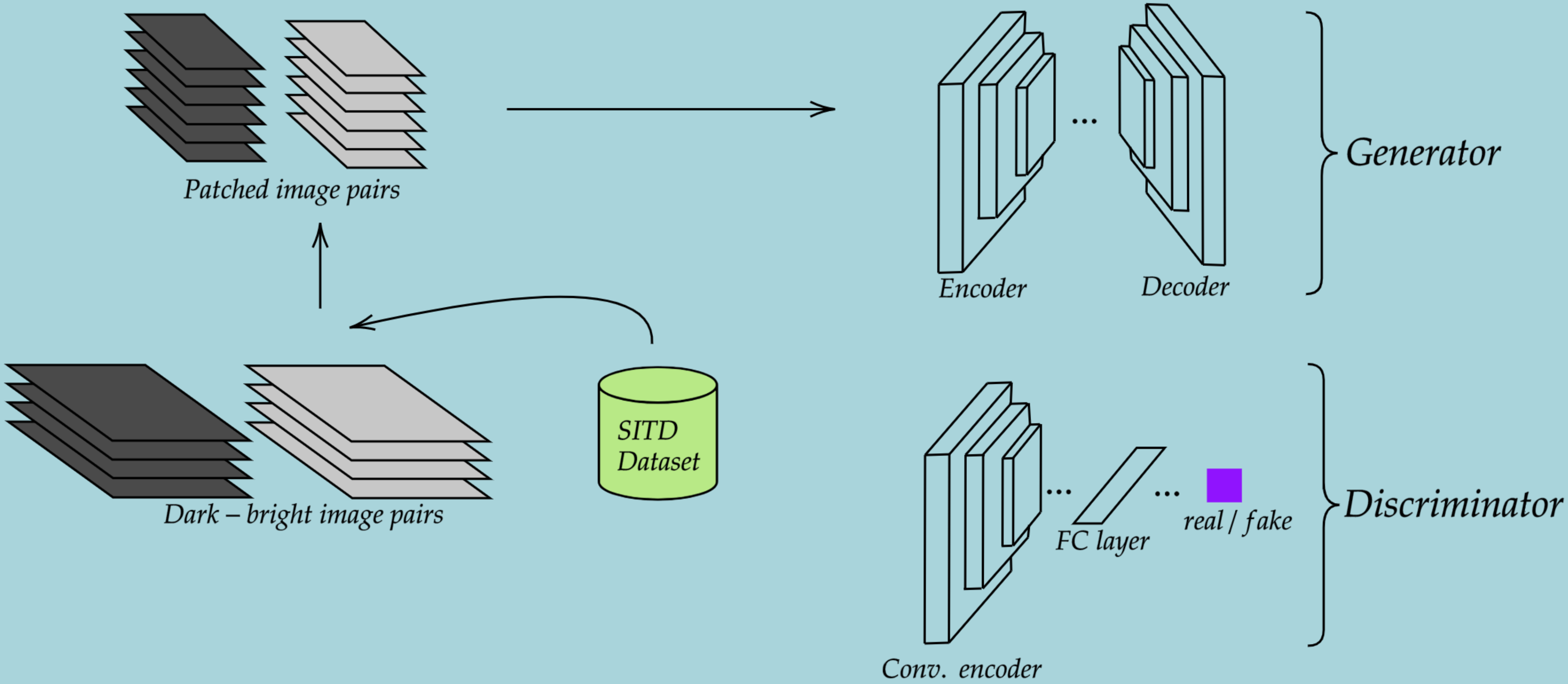
Dataset preparation

- Sony captures selected – smaller data size, easier conversion to JPG
- Each bright image has up to 10 dark counterparts – darkest selected
- Overlapping patches of size 100x100 with stride 50x50 for removing border noise

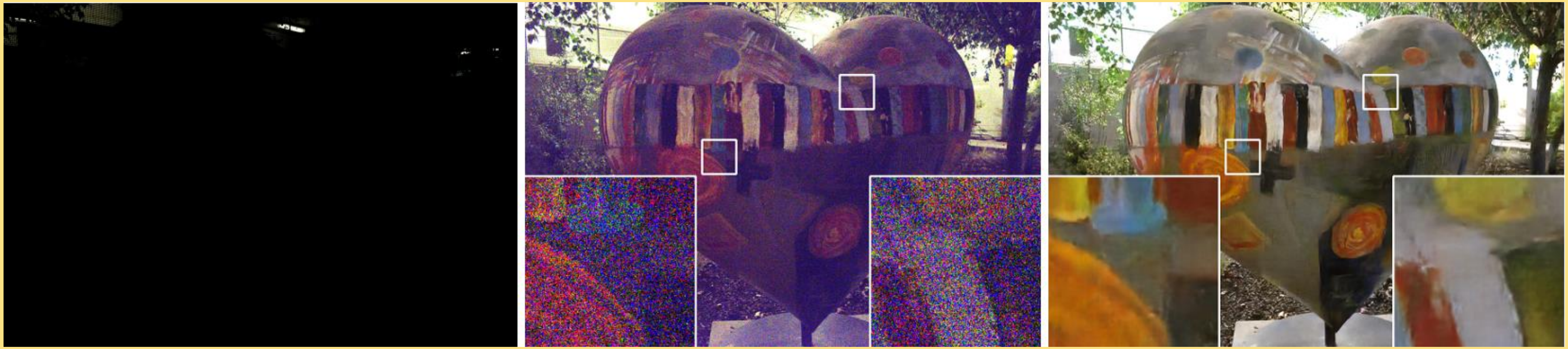
Model design

- 2-step training for simple GAN design as discussed
- Generator loss minimizes difference b/w real and fake (generated) bright image
- Discriminator loss used to maximize ability to separate real and fake

Final model pipeline



Results



Qualitative evaluation

Compared to traditional pipeline

- Upscaling + BM3D denoising
- Variations across creative platforms like Photoshop, Affinity, GIMP, …
- Upscaling messes with color balance, while denoising messes with sharpness

Compared to other ML techniques

- More realistic light balance
- Better understanding of reflections and light sources
- Color retention close to real-life even with “pitch-black” images

Conclusion & Future Work

Non-trivial problem

Image illumination especially useful for mobile photography

Simplified training procedure

Supervised learning means basic model designs and variations can be used

Final experiments remaining

Comprehensive quantitative evaluation

3-channel image dataset

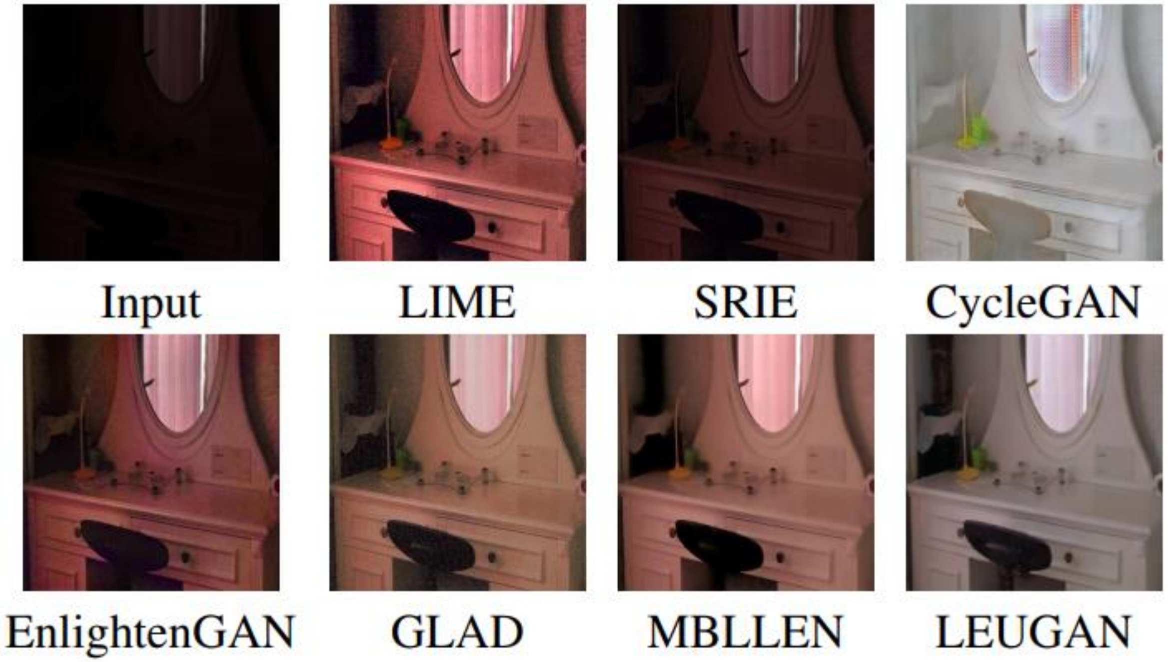
Having original 3-channel database can help in better color accuracy with

TFLite backend open mobile deployment

Converting basic Keras or TF models to TFLite can help in designing apps with natively deployed image illumination models

Real-time processing

Current model is at ~50ms processing time, too high for 30fps (~30ms) for low-light video



Thank You!

1. Chen, Chen, et al. "Learning to see in the dark." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2018.
2. Jiang, Yifan et al. “EnlightenGAN: Deep Light Enhancement Without Paired Supervision.” IEEE Transactions on Image Processing 30 (2021): 2340-2349.
3. Qu, Yangyang, and Yongsheng Ou. "LEUGAN: Low-Light Image Enhancement by Unsupervised Generative Attentional Networks." arXiv preprint arXiv:2012.13322 (2020).

Some images gathered from public domain