GAN-based Inpainting

Final Project

Nik Bhatt

Datasets

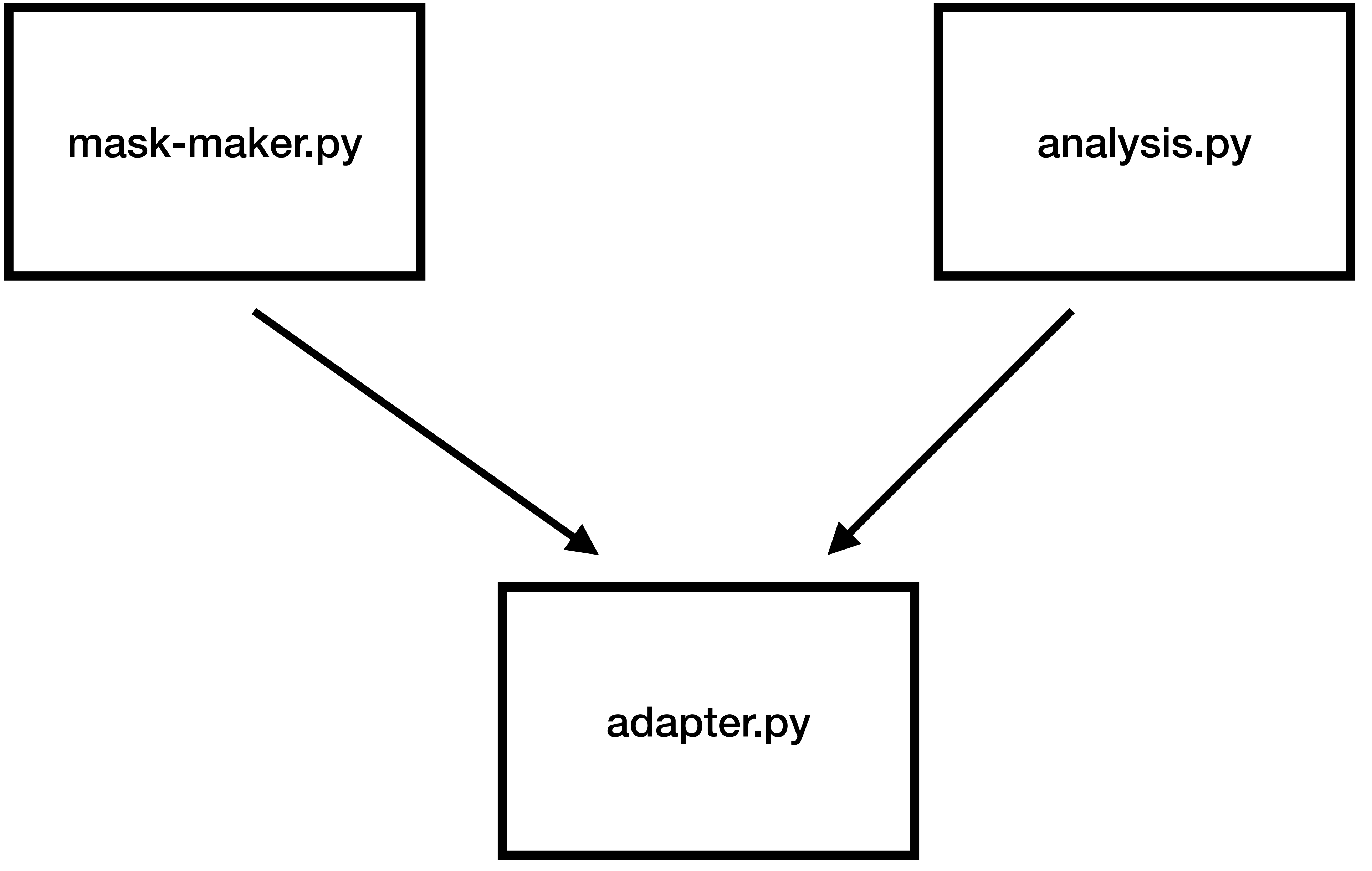
	Train	Test	Validation	Segmentation Masks
Places	1.8 M	45,000	36,500	0
Open Images	1.7 M	125,000	41,620	13,524

Researchers have to make their own (random) masks for Places

http://places2.csail.mit.edu/download-private.html

https://storage.googleapis.com/openimages/web/download_v7.html

Library Contents



Adapter.py

Bridges GAN code with analysis and mask making

- Different naming convention for photos, masks, and outputs
- Different directory hierarchy.
- Some use black-on-white masks, others use white-on-black.
- Some have limitations on image size (e,g., 256 x 256)
- Some have assumptions on file types (jpeg, png, etc.)
- For each GAN, I write an adapter subclass.
- Subclasses have 5-10 lines of code.
- So to support another paper, just write a small subclass!

Mask-Maker.py

Generate and Match Masks from Open Images

- For Open Images, match photos to their segmentation masks. If multiple objects detected, composite masks into a single mask. If photo has no masks, skip it.

- For Places, randomly assign Open Image composite masks to photos (akin to random mask generation currently done)

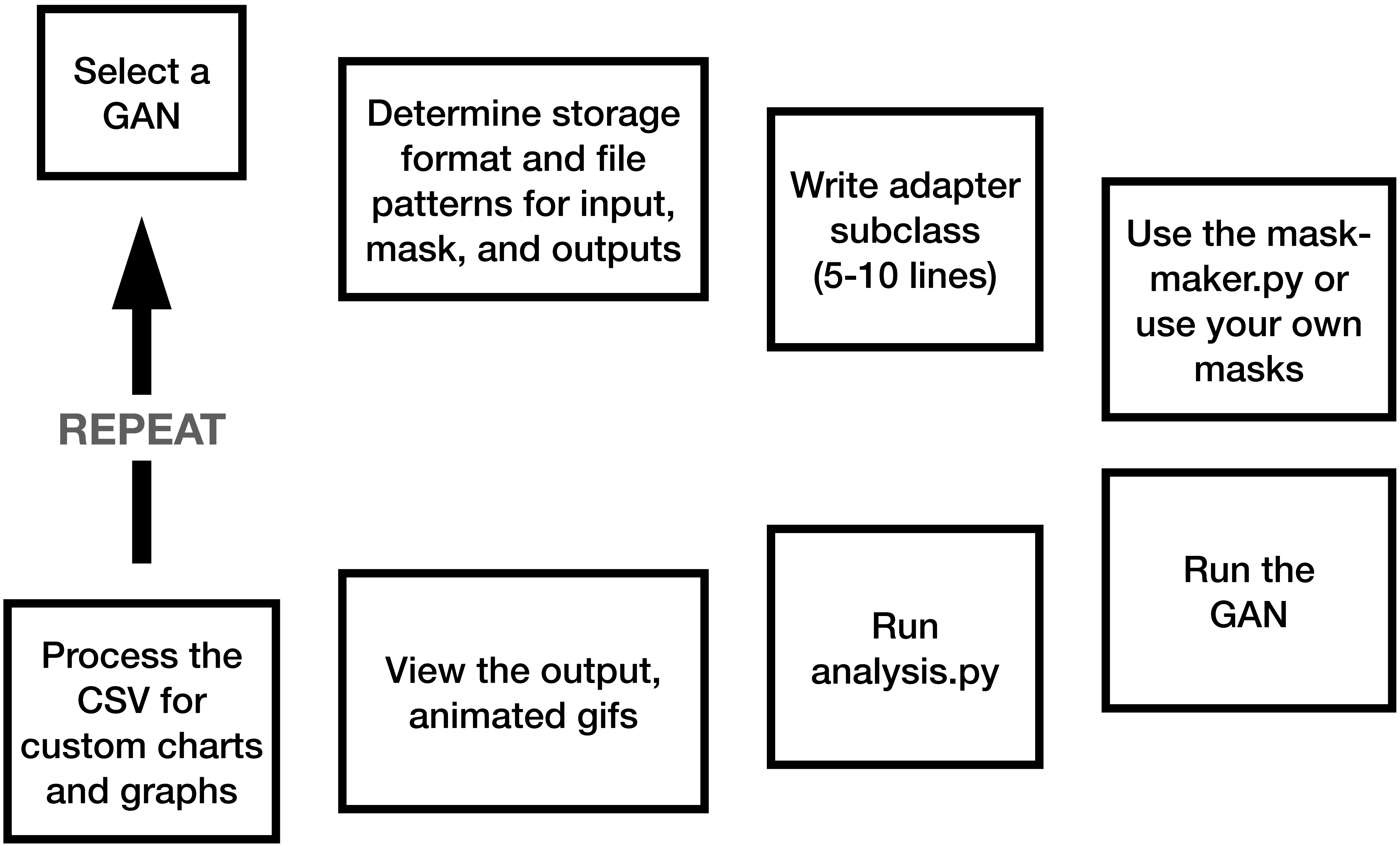
- Scales mask to source image size, inverts mask if needed.

- Final output: directory of photos to repair, directory of masks that match those, and directory of composite masks.

- Now the GAN can be run using my desired dataset and masks.

Workflow

How to integrate the library



Analysis.py

Mask Composite (Open Images)

Analysis.py

Animated GIF - some impressive results

Analysis.py

Animated GIF - Lots of terrible results

The downside of image synthesis

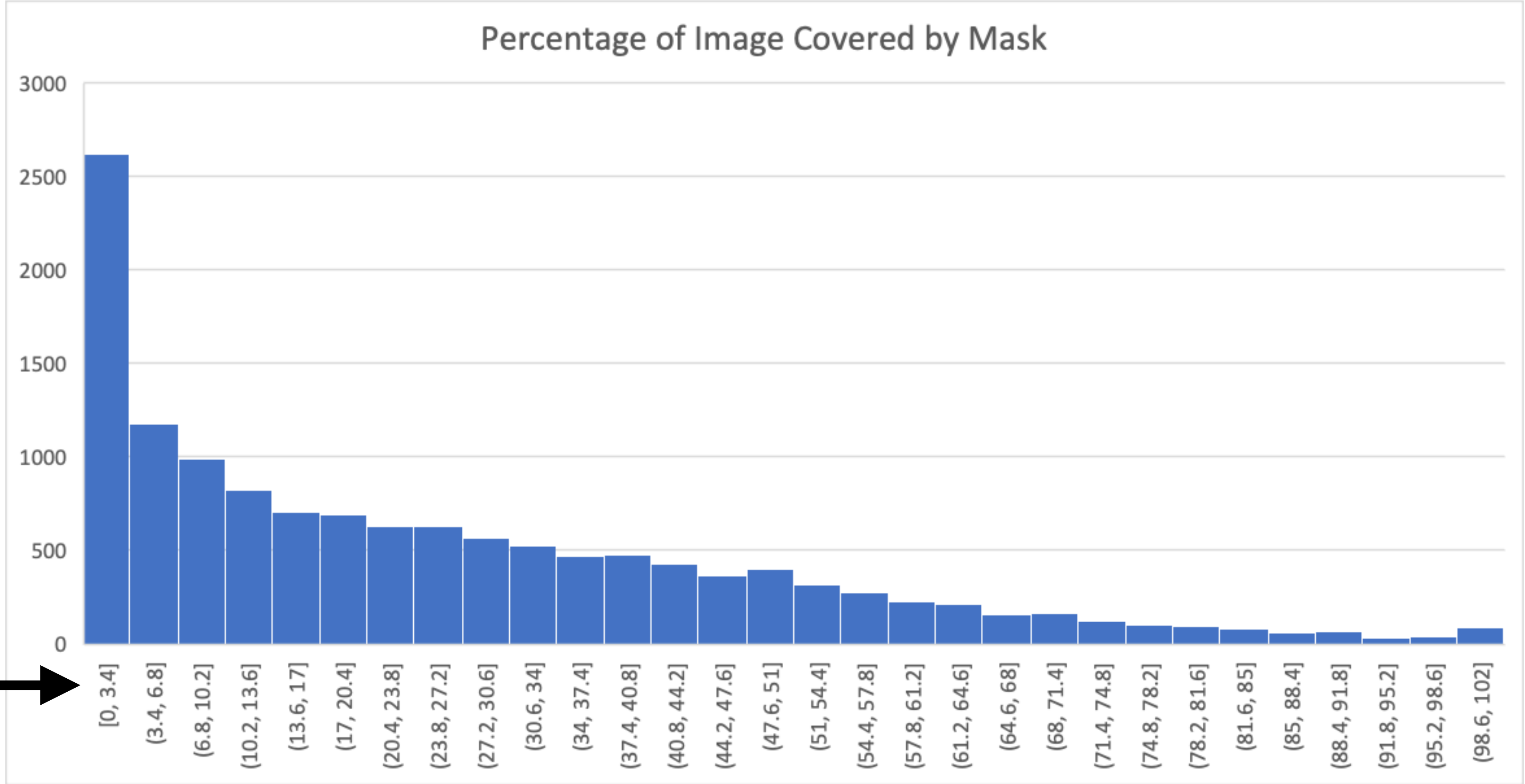
Analysis.py

Statistics

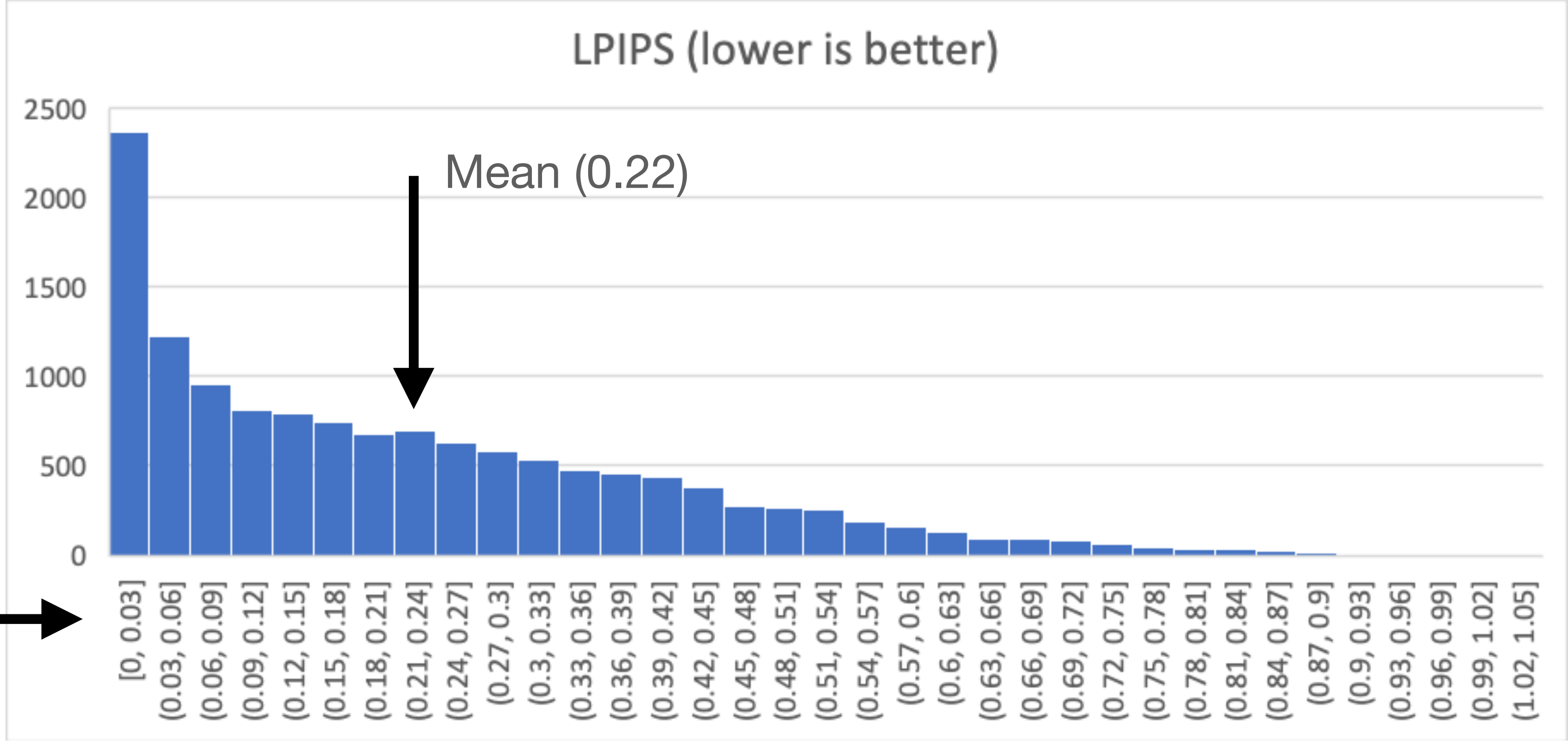
- LPIPS: Perceptual Similarity
- SSIM: Structural Similarity
- FID: How similar the generated images are to the “real images”

CSV to Graphs (Excel)

[0, 3.4%] coverage



[0, 0.03] LPIPS



Sample Results

MI-GAN	SSIM ↑	LPIPS ↓	FID ↓
Places	0.822	0.148	3.487
Open Images	0.795	0.219	36.469

CoModGAN	SSIM ↑	LPIPS ↓	FID ↓
Places	0.821	0.148	2.755
Open Images	0.792	0.220	34.410

Random masks did better on Places (the training dataset) than objects on the unseen dataset => some element of overtraining.

“That’s all, Folks!”