Reproducibility report for “Deep Decoder: Concise Image Representations from Untrained Non-convolutional Networks”

Kamil Mazur, Wiktor Adamski

1. Introduction

For our final project we attended reproducibility challenge with paper [Deep Decoder: Concise Image Representations from Untrained Non-convolutional Networks]. To accomplish that, we tested three untrained methods for solving standard inverse problems:

- [Deep Decoder] ([example code])
- [Deep Image Prior] ([example code])
- [BM3D] ([code]).

As we started working on the test we quickly realized that information about experiment provided in Deep Decoder paper is insufficient to fully reproduce it. Described models need many parameters to work (mostly DIP). We couldn't find clues about majority of the values used by authors of paper so we decided to use values provided with example code. Same goes for the test images, which were mentioned in paper only by single word name (like *saturn*, *baboon*, or *zebra*).

2. Precise parameters and dataset specification

In order to reproduce our results one need to use following parameters for configuring models:

- Deep Decoder:
  - Denoising: k = 128, 2000 iterations, adam optimiser with learning rate 0.0025
  - Inpainting: k = 320, 20000 iterations, adam optimiser with learning rate 0.0025
  - Superresolution: k = 128, 5000 iterations, adam optimiser with learning rate 0.0025

- Deep Image Prior:
  - Denoising: INPUT = 'noise', pad = 'reflection',OPT_OVER = 'net',LR = 0.01,OPTIMIZER='adam',exp_weight=0.99,   num_iter = 3000, input_depth = 32, net = get_net(input_depth, 'skip', pad, skip_n33d=128, skip_n33u=128, skip_n11=4, num_scales=5, upsample_mode='bilinear')
  - Inpainting: INPUT = 'noise',pad = 'reflection',OPT_OVER = 'net',OPTIMIZER = 'adam',input_depth = 32,LR = 0.01 ,num_iter = 6001,reg_noise_std = 0.03,net = skip(input_depth, img_np.shape[0], num_channels_down = [128] * 5, num_channels_up =   [128] * 5, num_channels_skip =   [128] * 5, filter_size_up = 3, filter_size_down = 3, upsample_mode='nearest', filter_skip_size=1, need_sigmoid=True, need_bias=True, pad=pad,act_fun='LeakyReLU')
  - Superresolution: factor = 4,enforse_div32 = 'CROP',INPUT = 'noise',pad = 'reflection',OPT_OVER = 'net',OPTIMIZER = 'adam'KERNEL_TYPE='lanczos2',,input_depth = 32,LR = 0.01,tv_weight = 0.0,num_iter = 2000,reg_noise_std = 0.03,get_net(input_depth, 'skip', pad, skip_n33d=128, skip_n33u=128, skip_n11=4, num_scales=5, upsample_mode='bilinear')
- BM3D (denoising only): 2DtransformStep1 = bior,useSD1 = 0,2DtransformStep2 = dct, useSD2 = 1, ColorSpace = rgb

Link to our dataset repository

3. Comparison

Here we present results obtained by us in comparison to results from paper. All values are peak signal to noise ratio (PSNR), signal being original picture, with difference between picture and given method output being noise. Our results being in red. Bold values signify highest results in given task on given picture.

	astronaut	baboon	barbara	f16	mri	zebra
Denoising						
identity	19.1 20.6	18.8 20.3	18.8 20.3	18.9 20.3	20.6 22.1	19.0 21.3
DD128	**34.3** **29.8**	**23.8** 21.4	28.0 **26.8**	28.7 **29.1**	**33.3** 26.9	**29.0** 22.5
DIP	29.9 26.1	22.9 **22.8**	28.1 24.4	29.8 25.0	25.7 26.6	26.4 **24.8**
BM3D	30.4 26.2	**23.8** 22.6	**29.3** 24.7	**30.2** 25.2	25.1 **28.0**	27.1 22.8
Inpainting						
identity	19.3 14.0	10.8 14.0	21.0 14.9	17.0 11.7	18.5 18.3	13.8 13.0
DD320	**44.6** **36.1**	**33.3** 23.9	37.8 30.8	33.1 34.7	**42.8** 31.1	**37.8** 23.6
DIP	40.4 35.3	30.4 **26.2**	**41.6** **35.6**	**37.9** **34.7**	34.1 **32.1**	26.0 **24.2**
Superresolution						
Bicubic	29.3 29.3	21.6 **20.7**	26.0 26.3	26.4 26.4	24.5 24.5	23.0 18.2
DD128	**36.7** **30.2**	**25.4** 20.6	**28.8** **26.4**	**29.8** 26.6	**35.7** 26.4	**30.4** 19.0
DIP	29.6 29.6	21.9 20.6	26.2 26.4	27.1 **27.4**	25.9 **25.6**	24.5 **19.2**

4. Why we couldn't reproduce results presented in original paper

- Original paper didn't state the origin of dataset. We could assume, that images in question come from Imagenet, but this is only assumption based on statement about different experiment at the beginning of the paper. Our dataset composes of pictures found in demonstration code for both Deep Decoder and Deep Image Prior that matched names in results table, and similar results in bicubic baseline for superresolution task has further strengthened our confidence about right choice of pictures.
- In denoising task, authors didn't mention method by which noisy observations were generated, other that it was by adding random white noise to the picture. Lack of noise generation parameters ie. random distribution parameters and whether noise was the same in all channels.
- In inpainting task, to fully replicate experiment, we'd need original masks that pictures were inpainted with. Only example provided by authors was for picture that wasn't mentioned in results table, but shown as an empirical illustration of networks ability to restore parts of an image.
- In all tasks, authors didn't give enough parameters to train the networks. For Deep Decoder we were given parameter k - number of channels per layer, and we could assume that number of layers would be equal to 6 in each task, as it was stated that it was most commonly used. Last missing parameter was number of iterations that the net would have to train for, that would change between tasks. As for Deep Image Prior, no parameters of used net were given, and seeing how many parameters this network has, it's most probable that we tested on different network than original authors.

5. Summary

Our results were somewhat similar to ones reported in original paper, but if we take note of best values, our experiment favors Deep Decoder as a best model for three presented tasks. Most notably, Deep Decoder had bests results in superresolution challenge for every picture. Most probable reason for our results being overall higher than original ones, might be different test data origin, as we had to partially guess what the pictures mentioned in paper are. Empirical results are satisfactory for us, but both Deep Decoder and Deep Image Prior are rather slow for use in for ex. graphics suite (about 5 minutes for denoising 512x512 image, and up to 2 hours for inpainting on computer equipped with 2.20GHz 2-core Intel Xenon CPU and 12GB Nvidia Tesla K80 GPU, 13GB of RAM).