**1.a.i Define what generalization means in this specific task. Are there several kinds?**

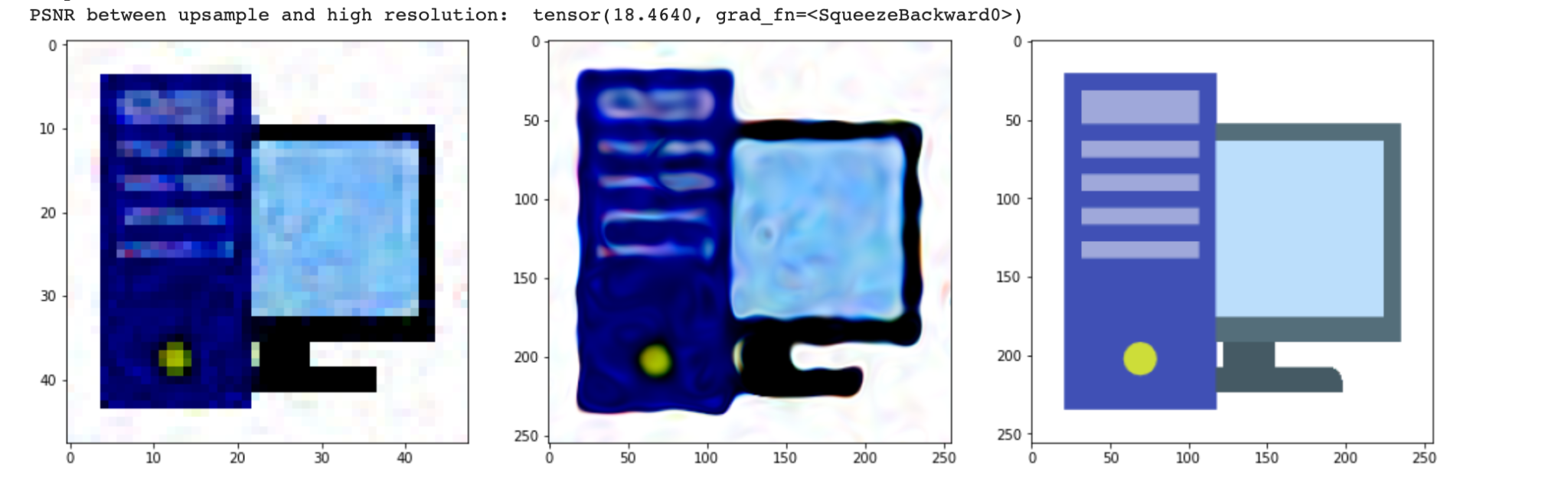
The generalization can be defined within a specific image, how well can the network generalize to an unseen patch of an image, while still respecting this unique image’s structure. This type of generalization can be tested with an “In-painting” setup, in which the network is only trained on a part of an image pixels and needs to generate the missing patch.

The generalization can be in the level of the images themselves - how well the network figures out the typical image in the set. This can be tested by letting the network generate new typical image and qualitatively examine the generated image’s “typicality” to the set.

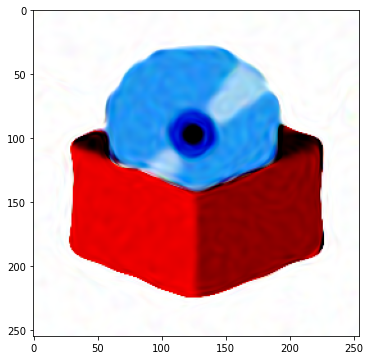
The generalization can be in the level of the signal-type - the network ability to generalize while respecting the structure of pixels of images, this can be tested with success on upsampling.

**1.a.ii Track the network’s generalization during training**

I decided to track the network’s generalization by probing its ability to upsample an image it was trained on. I used PSNR (common Super Resolution metric) with the ground truth to gauge its generalization performance.



2.a.i



2.b.i

**Based on my network architecture, measuring the similarity between different images** representations amounts to measuring the similarity between the average network activations when the inputs are the requested pixels of the appropriate images whose representations we would like to compare. After rudimentary research into the field of neural network representations comparisons, I tend to like the method described here: <https://arxiv.org/pdf/1905.00414.pdf>, it is invariant to linear transformations, can handle cases where there are more parameters than data points, and seems to be a robust statistical measure of similarity between representations.

I suggest measuring the CKA measure between pixel activations from different images. Highly correlated activations will indicate similar representations of images.



**3. Improved image interpolation - If your above result does not provide satisfactory interpolation results, design an improved solution.**

**a. Design an improved solution to the image interpolation problem.**

I’m actually not sure what exactly is the interpolation problem you mentioned. Better interpolation should probably involve a latent encoding of an image, which in turn will require a somewhat different model that encodes the entire image into a lower dimensional vector. Moving on the manifold of such lower dimensional space should give interpolation results that are better than averaging on the output as I did. I’m not sure how to incorporate such encoding in the signal representation setup here.

I did have another idea that might help with that…

**b. Describe all the details of your solution, however no need to implement it.**

I thought of using Hyper Networks, basically training a big network to generate the weights of small networks to encode the pixels of the different images. The hyper network gets the image id at the input and outputs the weights of small network that represents that image. I didn’t have enough time to finish training it during this weekend, but maybe i’ll try later on…

**c. Explain why you hypothesize that this solution will provide a refined result.**

My thinking is that having this hyper network generates a vector of weights for the small image-encoded network, maybe I could find a way to interpolate between those networks (maybe even averaging them? Or combining the activations of earlier layers… not sure yet) in order to get a better interpolation between images.