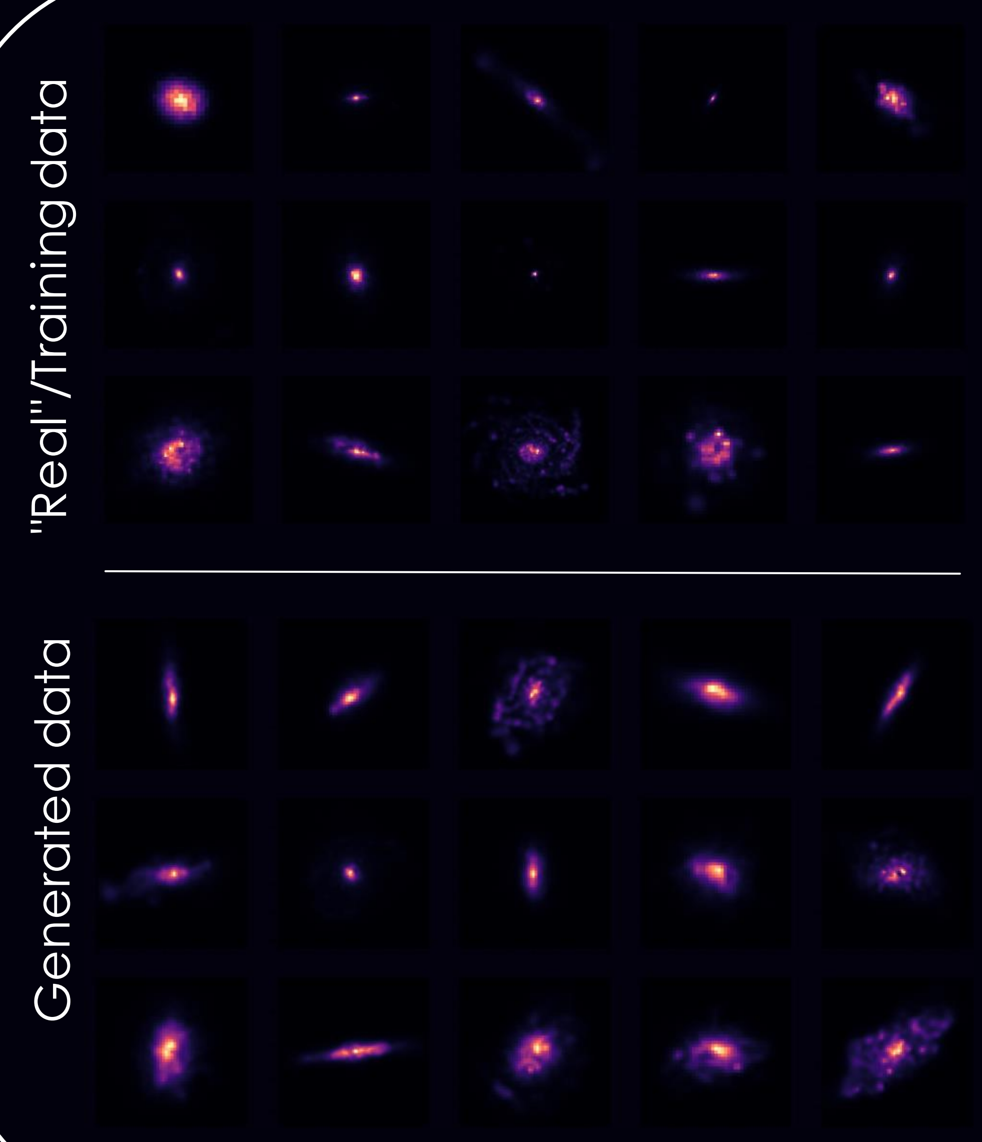
Improving Generalization using Generative Models for Galaxies

2021 IAP colloquium

October 18-22 2021, IAP, Paris

Benjamin Holzschuh



Data Generation:

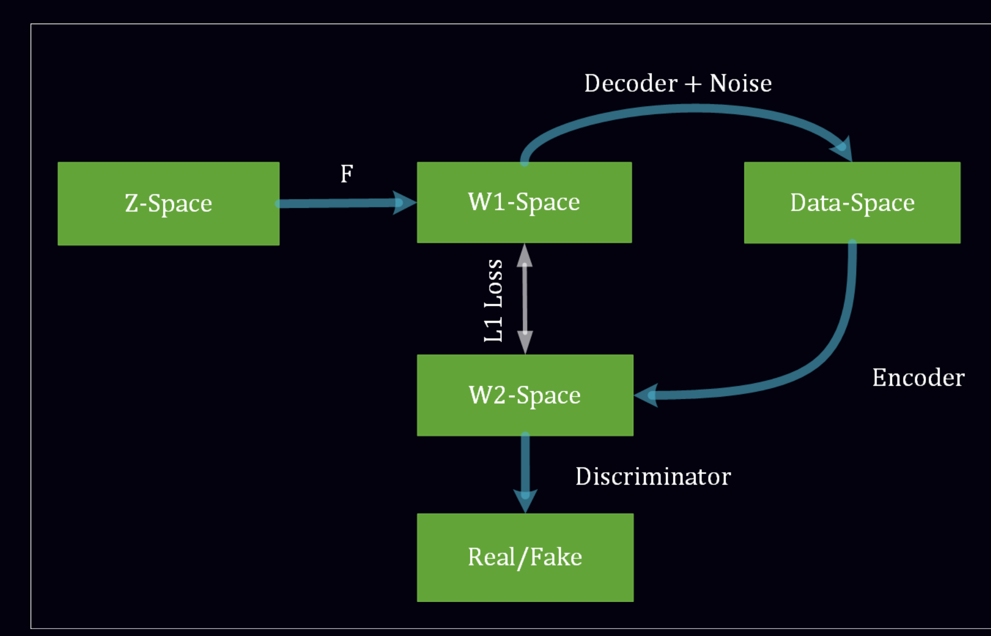
We use Adversarial Latent Autoencoders in this paper to learn a generative model for a training dataset:

- **Z-space**; space from which samples are drawn, fixed prior distribution
- **W1-space**: latent representation of the data, F is a learned dense network
- **Adversarial loss**: decoder and noise input act as a synthesis network (generator). Encoder and discriminator classify whether data is fake or real
- **Reconstruction loss**: for a drawn sample x, we optimize the network so that Encoder(Decoder(F(x))) = F(x)

Training Dataset:

A dataset of synthetic galaxy observations from the TNG-100 simulation:

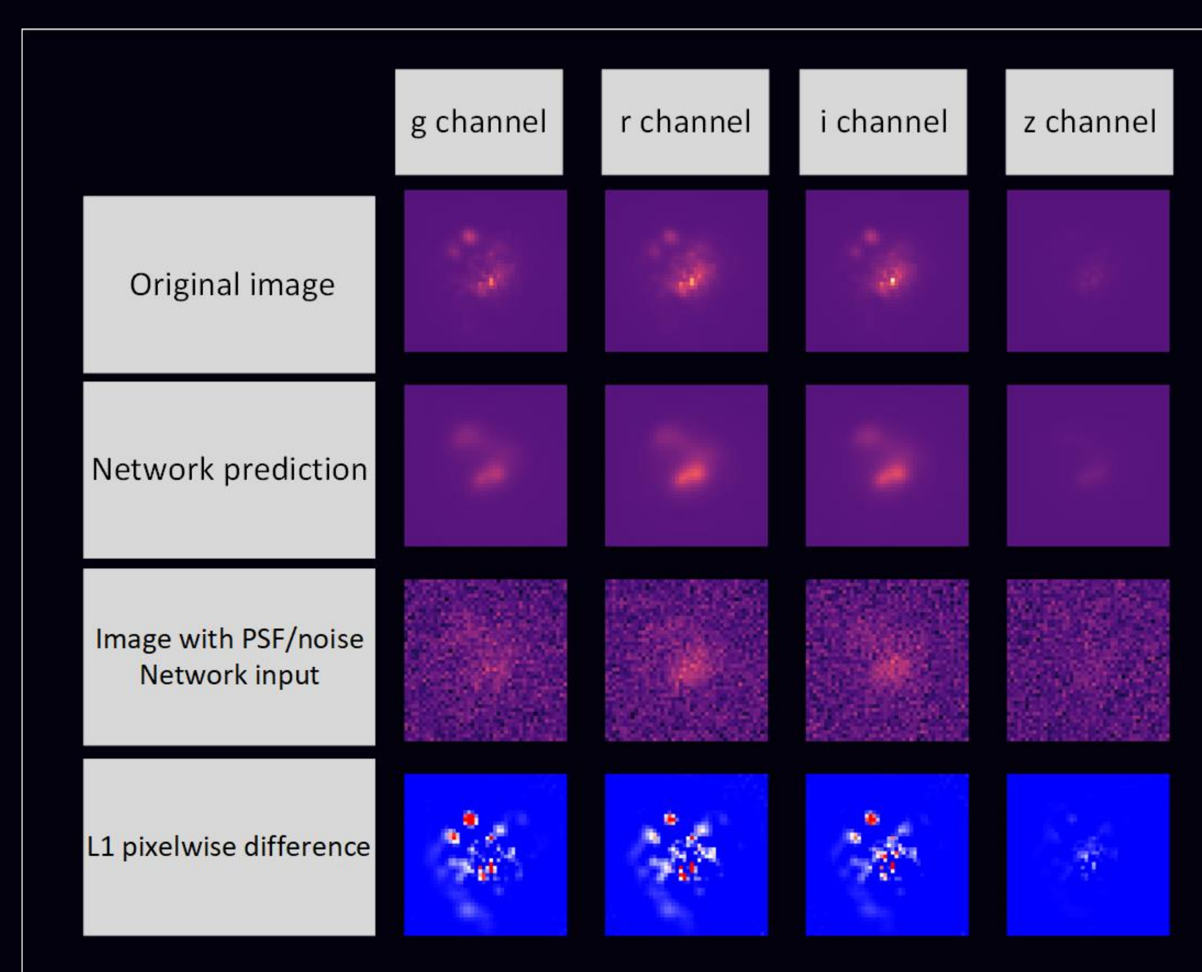
- located at redshift z=0, ~12.000 galaxies
- physical pixel size of 0.276 ckpc/h
- SKIRT radiative transfer as post-processing
- four filters that match the SDSS g,r,i,z filters



Benefits of Training with Additional Generated Data: Image Denoising

To demonstrate the advantge that additional generated data can have, we train a denoising model on four datasets with added PSF and noise:

- **100% train**: the full TNG-100 training dataset
- **10% train**: 10% of the training dataset
- **100% gen**: a dataset generated by the generative model five times the size of the training dataset
- **100% gen + train**: both datasets combined

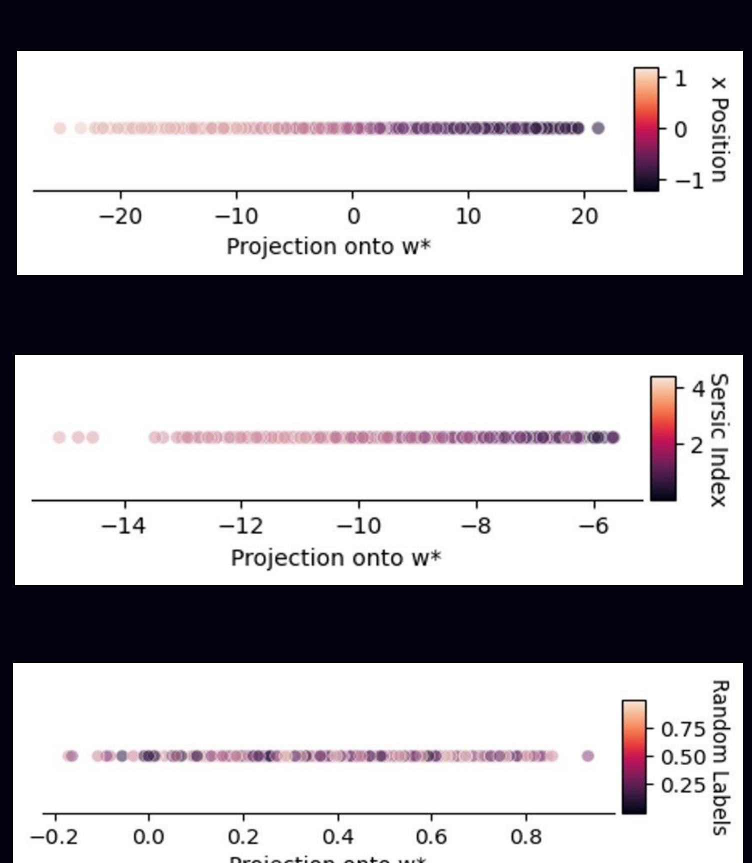


Results:

We validate our results and test our model on an independent validation set from TNG-100

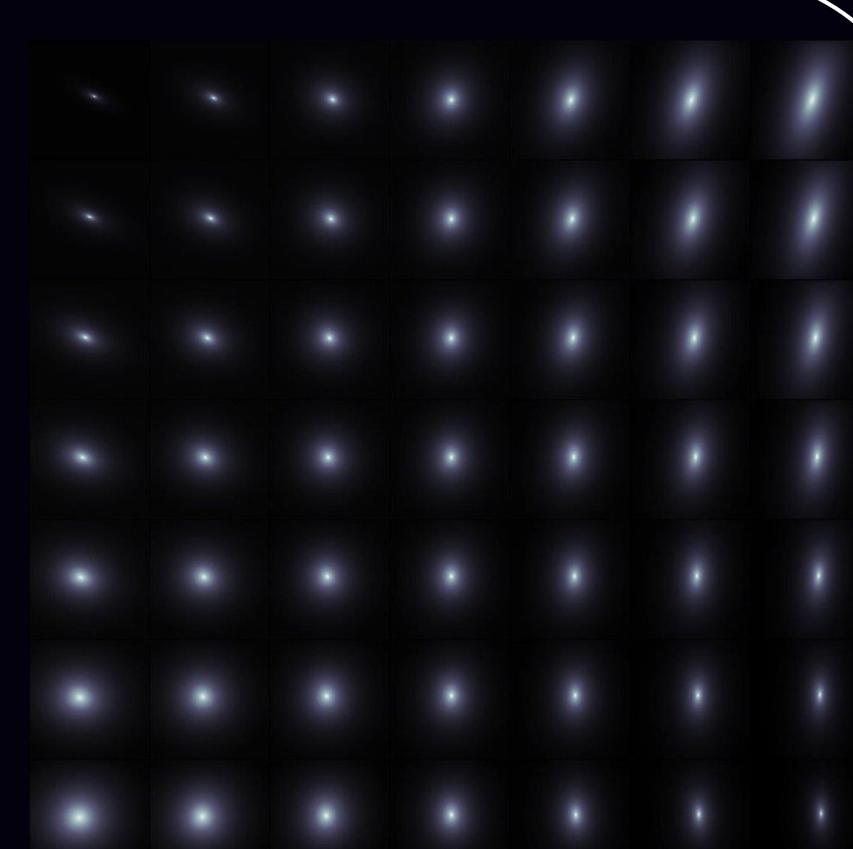
- Using only generated data gives the worst results, which shows shortcomings of the generative model
- Combining training and generated datasets yields clear improvements

Linear Structure of Latent Representation:



For this part, we train our generative model on Sersic profiles, which are fully defined by a set of seven parameters:

- Generate a set of Sersic Profiles using trained model and randomly drawn latent representation
- Fit the seven parameters to the generated images (BFGS algorithm)
- For a specific parameter, assign class 0 or 1 to the latent representation that generates the galaxy depending on whether a chosen fitted parameter is low or high
- Find a separating hyperplane between class 0 and 1 and project the latent representations onto a vector w* orthogonal to the hyperplane
- The latent representations are almost perfectly ordered according to the chosen fitted parameter **(left)**



Fixing four randomly drawn latent representation at the corners and interpolating between them shows a smooth transition