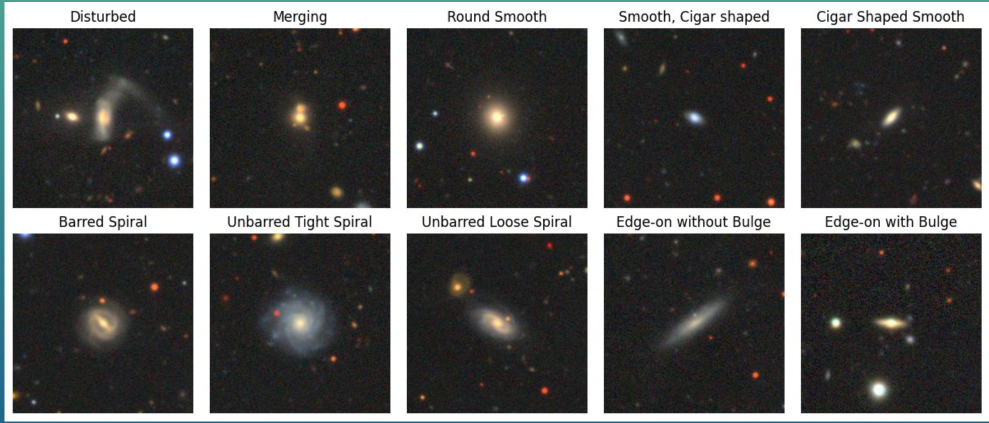
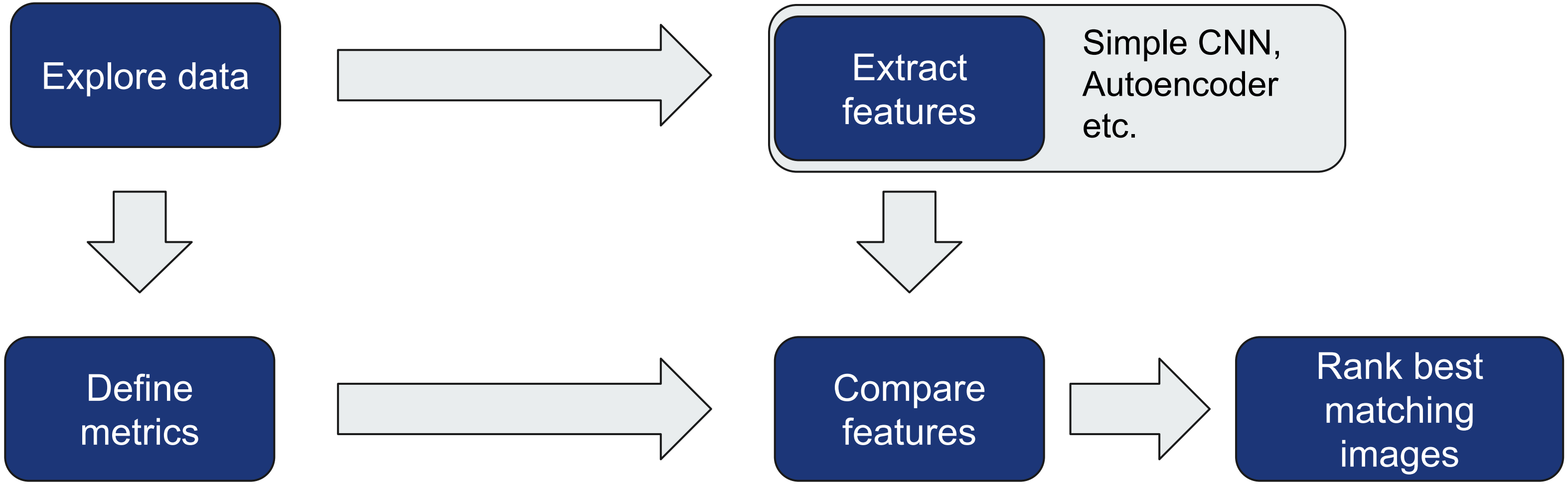
Milky Way Challenge

Team Yellow - Josef, Tobias, Nitish, Stephanie, Aaron

Dataset Exploration



How to tackle this problem?



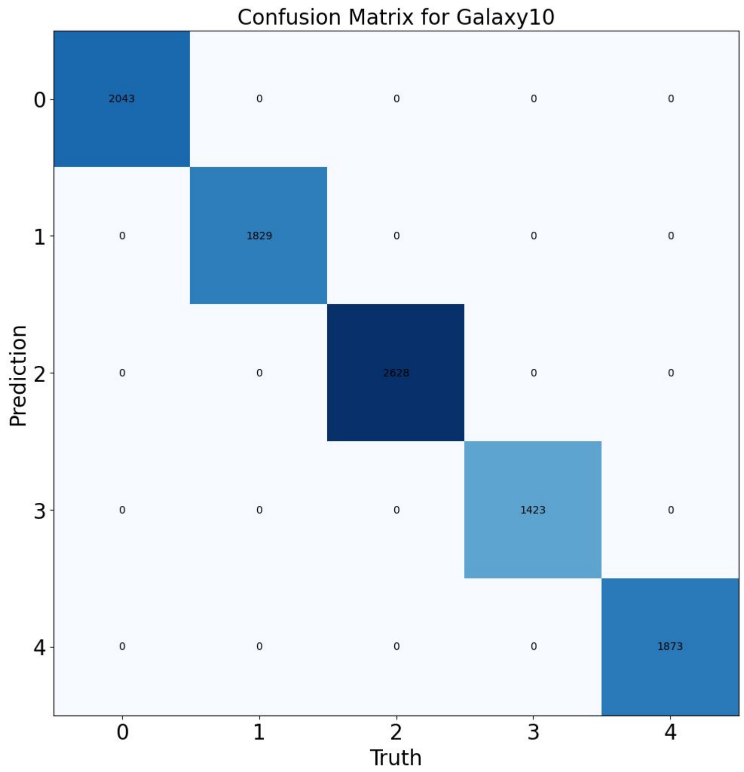
ERUM DATA HUB

Data Exploration

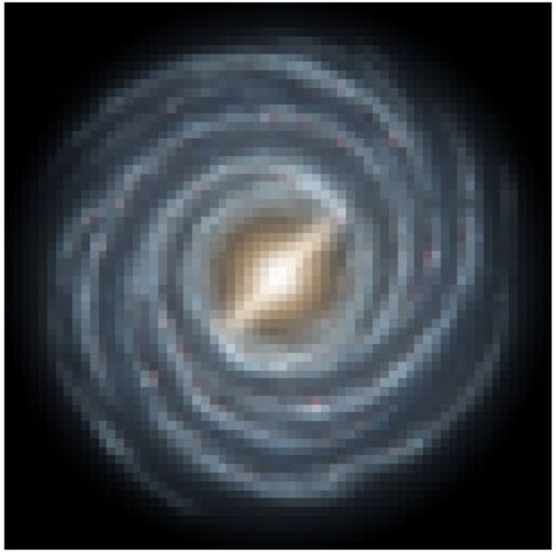
- Evenly distributed classes → good for classification
- Classes:
    - Barred Spiral
    - Unbarred Tight Spiral
    - Unbarred Loose Spiral

      similar classes
    - Edge-on without Bulge
    - Edge-on with Bulge

      similar classes
- Preprocessed all images to 64x64x3 → values [0,1] by division by 255



Target Image



- Target image is a barred Spiral
- large black edges / sparseness
- artists representation

ERUM DATA HUB

Metrics for Image Comparison

```
from sewar.full_ref import mse, rmse, psnr, uqi, ssim, ergas, scc, rase, sam, msssim, vifp
```

- find suitable metrics for image comparison
- several packages allow non-AI image comparison

	mse	rmse	psnr	uqi	ergas	scc	rase	sam	vifp
0	0.027954	0.167193	inf	0.759659	11.718323	0.012926	2850.967992	0.359854	0.411005
1	0.043954	0.209652	inf	0.675065	16.856165	-0.016002	4604.006668	0.521818	0.127013
2	0.071874	0.268093	inf	0.685547	15.651699	-0.002293	3286.736504	0.508952	0.280538
3	0.030361	0.174243	inf	0.746876	12.795824	0.006357	3087.710577	0.438481	0.390696
4	0.237948	0.487799	inf	0.479943	17.485029	0.012184	4083.393081	0.623266	0.632648
5	0.049683	0.222897	inf	0.702993	13.407740	-0.035949	3062.915620	0.457193	0.369263
6	0.021903	0.147998	inf	0.764772	12.913064	0.009743	3366.983789	0.419909	0.328860
7	0.119115	0.345131	inf	0.566635	15.642411	-0.008720	3665.240271	0.490860	0.554170
8	0.033524	0.183095	inf	0.682135	17.269837	-0.001357	4606.280992	0.559283	0.177191
9	0.047732	0.218477	inf	0.711160	12.910992	0.008070	2811.744986	0.482299	1.305406

ERUM
DATA HUB

Metrics for Image Comparison

```
from sewar.full_ref import mse, rmse, psnr, uqi, ssim, ergas, scc, rase, sam, msssim, vifp
```

MSE (Mean Squared Error):

- Measures the average squared difference between corresponding pixels of two images.
- Higher values indicate greater dissimilarity between images.

RMSE (Root Mean Squared Error):

- The square root of MSE.
- Provides a similar measure of pixel-wise dissimilarity but in the same units as the image data.

ERUM DATA HUB

Metrics for Image Comparison

```
from sewar.full_ref import mse, rmse, psnr, uqi, ssim, ergas, scc, rase, sam, msssim, vifp
```

UQI (Universal Quality Index):

- A quality metric that combines luminance, contrast, and structure information.
- Higher UQI values indicate better image quality.

ERGAS (Normalized Global Relative Error):

- Measures the relative error in the average spectral information of images.
- Often used in remote sensing applications.

Metrics for Image Comparison

```
from sewar.full_ref import mse, rmse, psnr, uqi, ssim, ergas, scc, rase, sam, msssim, vifp
```

SCC (Structural Content Correlation):

- Evaluates the structural similarity between two images.
- Higher values indicate greater structural similarity

RASE (Relative Average Spectral Error):

- Quantifies the spectral distortion between two images.
- Useful for assessing spectral image quality.

Metrics for Image Comparison

```
from sewar.full_ref import mse, rmse, psnr, uqi, ssim, ergas, scc, rase, sam, msssim, vifp
```

SAM (Spectral Angle Mapper):

- Measures the spectral similarity between pixels in two hyperspectral images.
- Computes the angle between spectral vectors.
- Lower angles indicate greater similarity.

VIFP (Visual Information Fidelity for Perception):

- A perceptual image quality metric.
- Reflects the degradation of visual information in an image.
- Higher VIFP values indicate better image quality.

Metrics for Image Comparison

Cosine Similarity Metric

used to compare the **similarity between two vectors** in a multi-dimensional space.

Cosine Similarity(A, B) = (A · B) / (||A|| * ||B||)

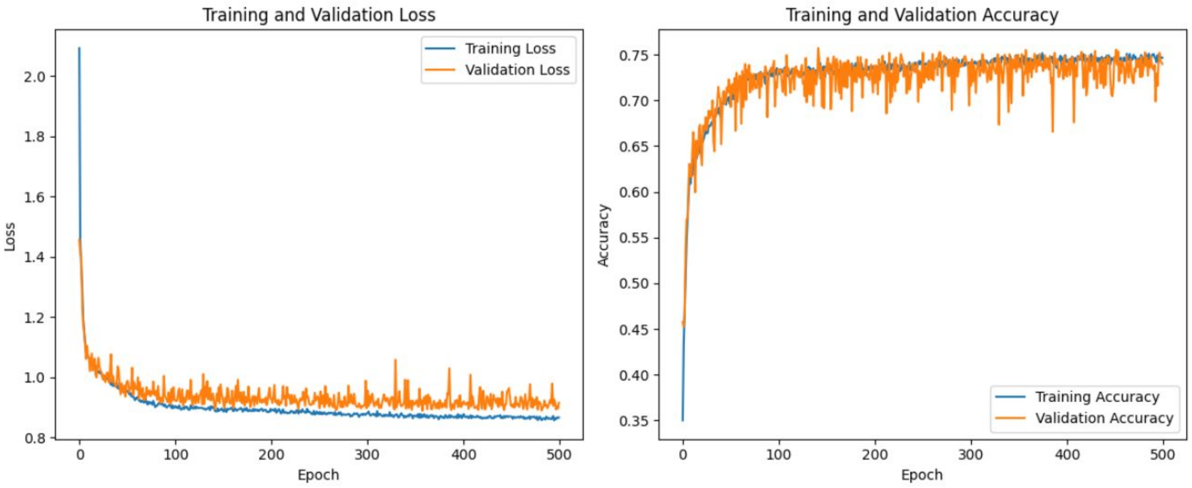
A and B are the vectors = images, (A · B) = dot product, ||A|| and ||B|| = norms of the vectors

Interpretation:

- Cosine similarity produces values between -1 and 1.
- A similarity score of 1 indicates that two vectors are identical.
- A score of -1 indicates complete dissimilarity.
- 0 suggests no similarity.

Classifier CNN Architecture + Results

Convolutional Classification Network



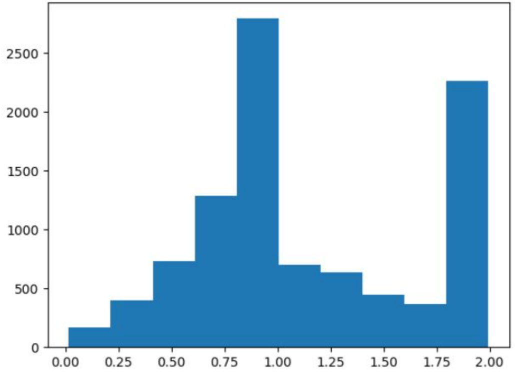
- 4 Blocks Conv 2D + MaxPooling (2,2)
  - decreasing Filter Size (256, 128, 64, 32)
  - ReLU Activation
  - L2 regularization
- 2 Dense Blocks
  - Nodes (128, 5)
  - ReLU and Softmax for Output
- Milky way classified as Barred Spiral

Convolutional Classification Network

Idea: Use output vector as latent space representation
-> Calc absolute Error between the Target Image and all others

Barred Spiral

Absolute Error
(Image/Target Image)

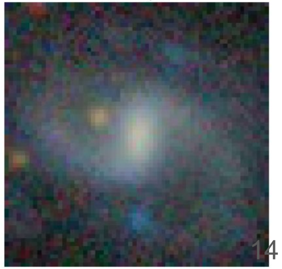


Take the 3 images with the smallest absolute error

Barred Spiral

Unbarred Loose Spiral

Barred Spiral



Feature Extraction using CNN-MLP-Classifier

ERUM DATA HUB

Feature Extraction using CNN-MLP-Classifier

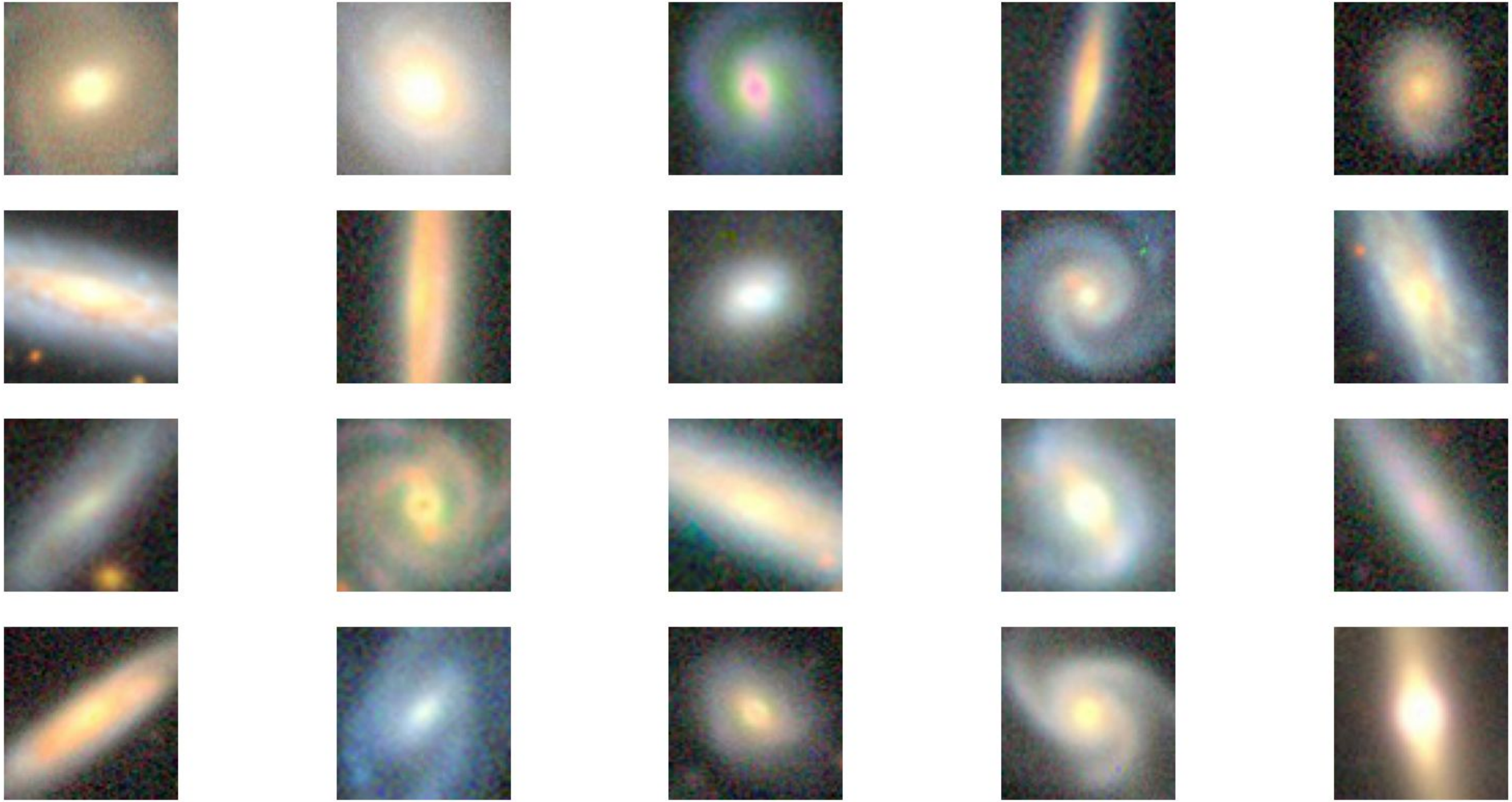
```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv1 (Conv2D)              (None, 62, 62, 32)        896

 maxpool1 (MaxPooling2D)     (None, 31, 31, 32)        0

 conv2 (Conv2D)              (None, 29, 29, 64)        18496

 max_pooling2d_3 (MaxPoolin  (None, 14, 14, 64)        0
 g2D)

 conv3 (Conv2D)              (None, 12, 12, 128)       73856

 max_pooling2d_4 (MaxPoolin  (None, 6, 6, 128)         0
 g2D)

 flatten_1 (Flatten)         (None, 4608)              0

 dense_2 (Dense)             (None, 128)               589952

 dense_3 (Dense)             (None, 5)                 645

=================================================================
Total params: 683845 (2.61 MB)
Trainable params: 683845 (2.61 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

Use simple CNN to extract image features:

- last conv. layer = feature extractor
- apply suitable metric
- rank images according to metric

Metric: Cosine Similarities

Feature Extraction using CNN-MLP-Classifier



ERUM
DATA HUB

Feature Extraction using CNN-MLP-Classifier

	mse	rmse	uqi	ergas	scc	rase	sam	vifp
0	0.004785	0.069173	0.642318	NaN	0.031262	24475.653885	NaN	NaN
1	0.011317	0.106382	0.582789	NaN	-0.003588	31489.553002	N	
2	0.010158	0.100789	0.612713	NaN	0.023427	29033.014865	N	
3	0.011948	0.109305	0.625424	NaN	0.010847	26833.516599	N	
4	0.005961	0.077209	0.624085	NaN	0.032570	26128.943268	N	
...	...	...	...	...	...	...	...	
1955	0.014452	0.120215	0.580614	NaN	-0.033621	31615.875474	N	
1956	0.007230	0.085028	0.625738	NaN	-0.001913	24503.071977	N	
1957	0.006867	0.082866	0.667316	NaN	0.050297	26485.917927	NaN	NaN
1958	0.006419	0.080116	0.629740	NaN	0.037840	25054.603753	NaN	NaN
1959	0.004079	0.063863	0.678443	NaN	0.041187	19881.260985	NaN	NaN

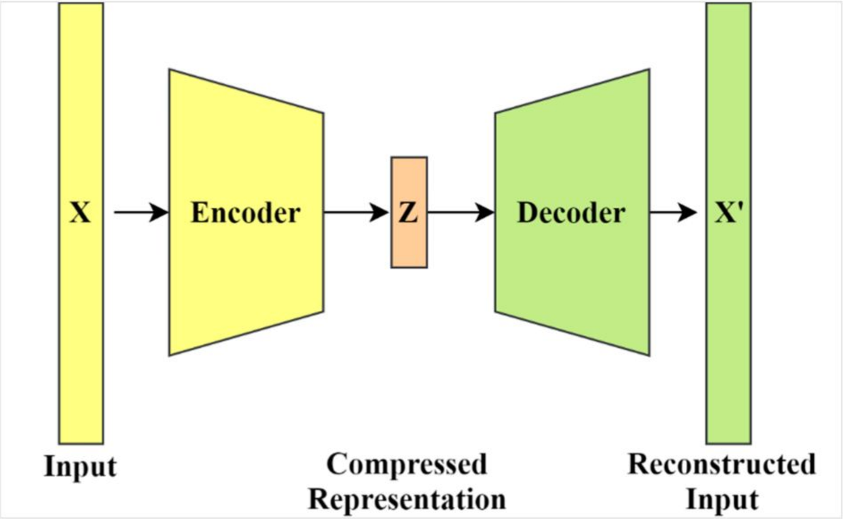
1960 rows × 8 columns

Other metrics might more efficient.

	Minimum	Maximum	Argmin	Argmax
mse	0.001612	0.023407	1410.0	768.0
rmse	0.040149	0.152994	1410.0	768.0
uqi	0.546815	0.801562	1229.0	1410.0
ergas	NaN	NaN	NaN	NaN
scc	-0.042413	0.168412	608.0	1410.0
rase	12803.164511	48516.174279	1410.0	968.0
sam	NaN	NaN	NaN	NaN
vifp	NaN	NaN	NaN	NaN

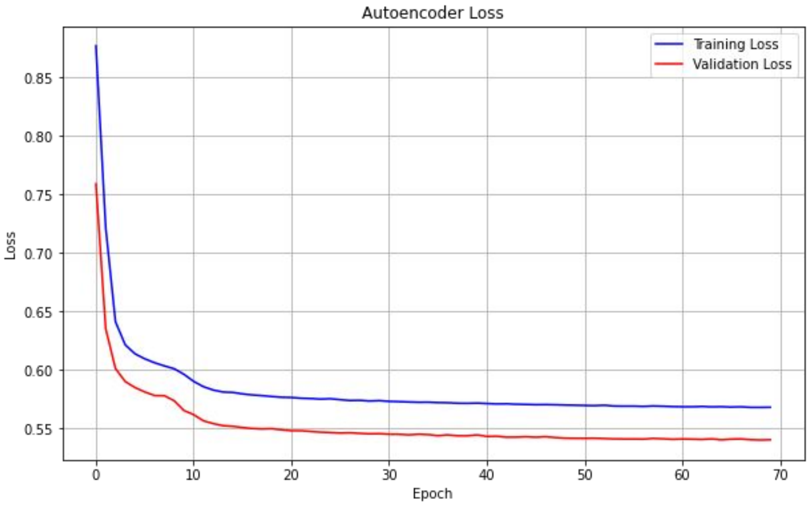
Feature Extraction using Autoencoder

Autoencoder Structure



- Encoder:
  - 4 blocks Conv2D + Max Pooling (2,2)
  - decreasing filter size (64,32,16,8)
  - ReLU
  - Latent space (4,4,8)
- Decoder:
  - 4 blocks Conv2D + Upsampling2D (2,2)
  - increasing filter size (8,16,32,64)
  - ReLU
  - Last Conv2D with 3 Filters (RBG) + Sigmoid

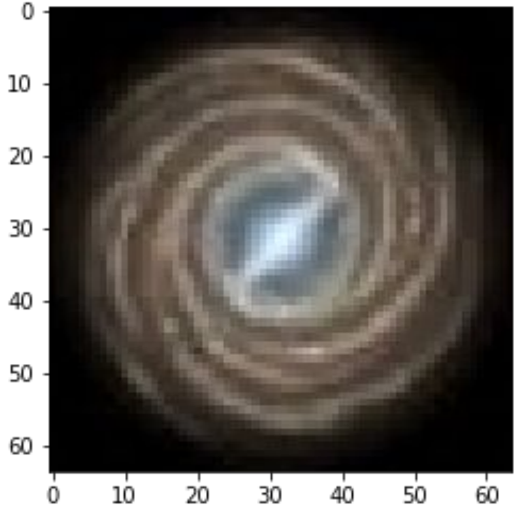
Training loss and selecting the most similar image



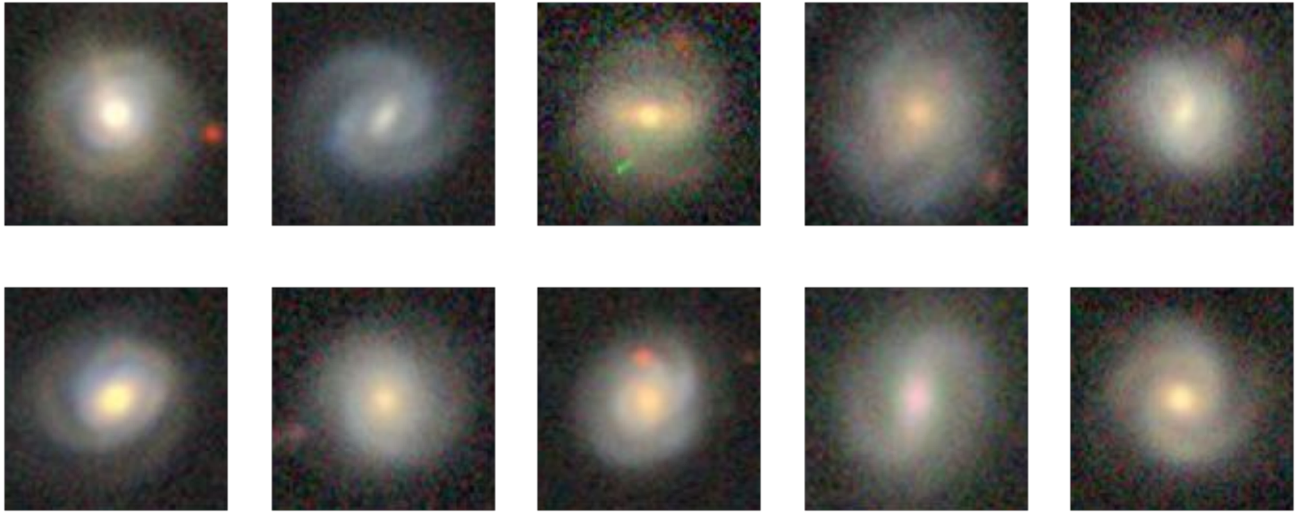
- **Loss:** MSE
- **How to get the most similar images:** check for cosine similarity between the target and the other images in the (4,4,8) latent space.
- Select the images with cosine similarity greater than 0.9875.

Most similar images

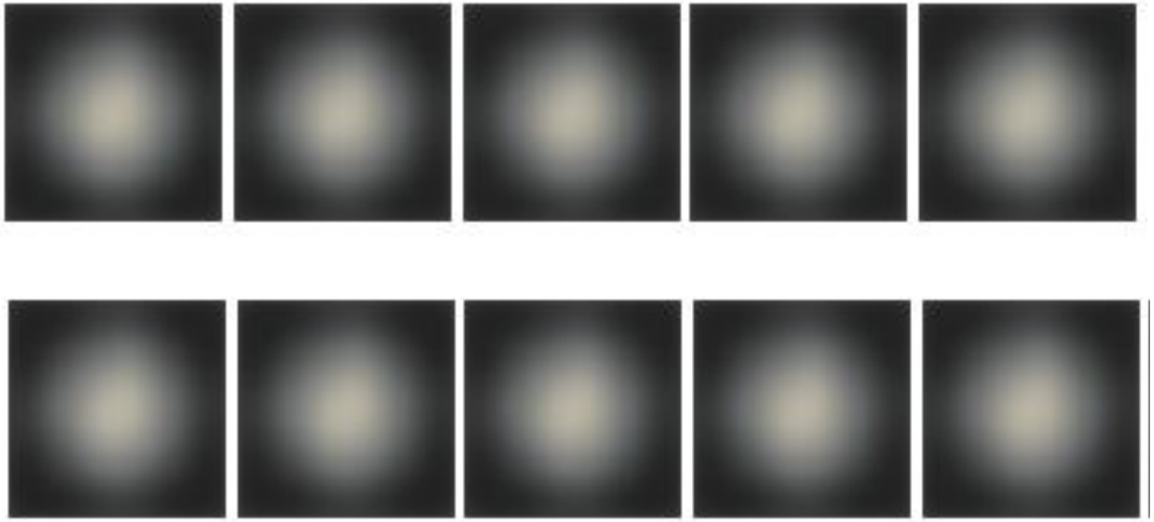
Target image



Closest Images (cosine similarity > 0.9875)



Bonus Task: Generated Images



- Encode target Image to obtain features
- add noise to the features
- decode new features

Thank you for your attention!