Generative Models
Autoencoder: Encoder

THE UNIVERSITY OF
SOUTHERN MISSISSIPPI

Generative Models

- Is a type of machine learning algorithm for data creation.
  - Data generated is similar to the data used to train a model
- Applications:
  - Image Synthesis
  - Natural language processing
  - Density estimation

- Variations of Generative Models:
  - Autoencoders
    - Variational Autoencoder
  - Generative Adversarial Networks
    - Generator vs. discriminator
  - Autoregressive models
    - LSTM (RNN)
  - Transformer
    - GPT
  - Diffusion Models
    - Works on noise generation

Image

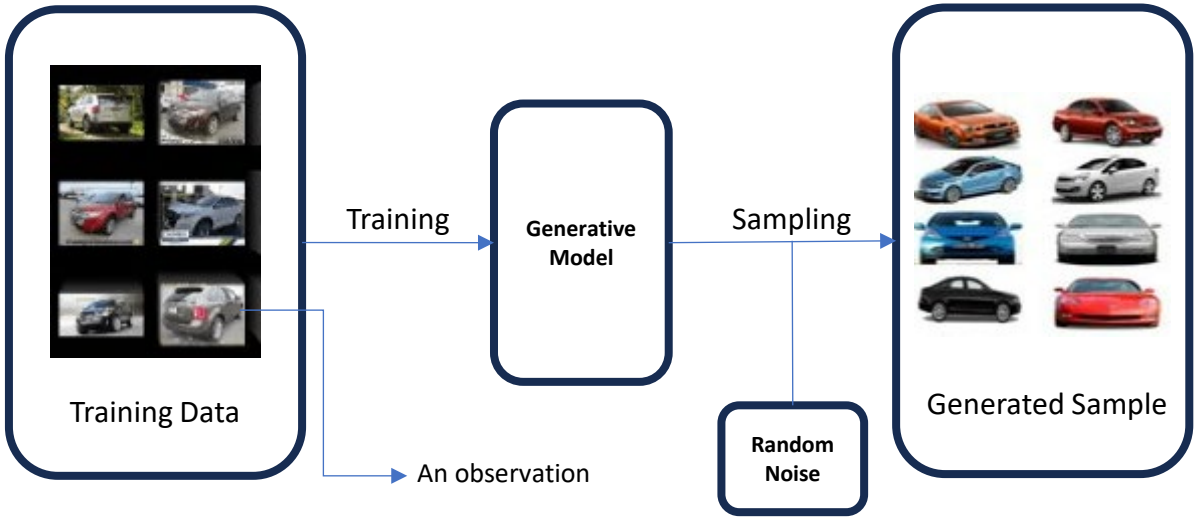
Text

Generative Models (GM)

- Train a generative model to capture rules that govern the relationship between pixels in an image of, for example, horses, cars, human faces, etc.
- Then, sample from this model to create realistic images that did not exist in the original dataset.
- Two main approaches for GM applications:
  - Image
  - Text
- Each observation consists of many features:
  - For image problems: individual pixel values
  - For text problems: individual words or groups of letters

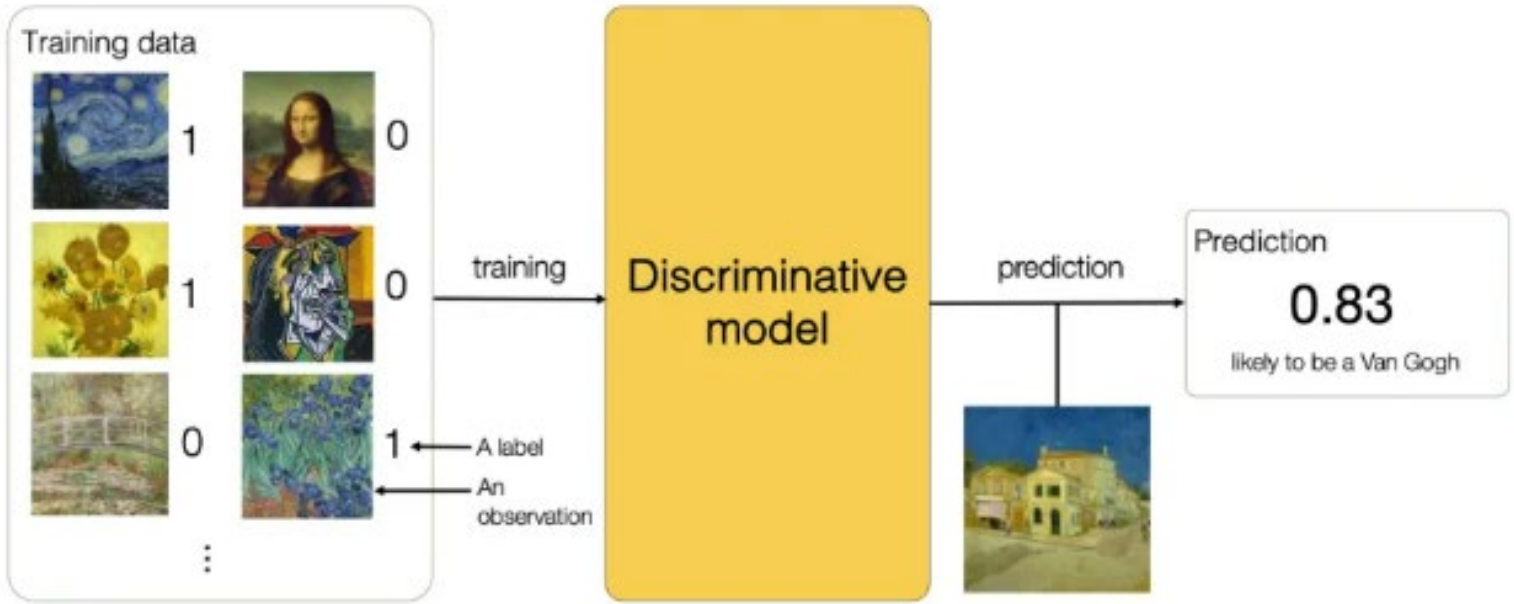
Generative Models (GM)

- GMs are probabilistic rather than deterministic
  - Reason: *sample* many variations of the output
    - If probabilistic, a random component influences the individual samples generated by the model.
  - Instead of: get the same (or approximated) output every time



Generative vs. Discriminative Modeling

- Discriminative:
  - Each observation in the training data has a *label*
  - For a binary classification, 1 – for Van Gogh, 0 – for non Van Gogh



Generative vs. Discriminative Modeling

- Formally:
  - Discriminative modeling estimates $p(y|x)$
    - Model the probability of a *label* y given some observation x
  - Generative modeling estimates $p(x)$
    - Model the probability of observing an observation x. Sampling from this distribution allows us to generate new observations.
  - Conditional Generative Models: $p(x|y)$
    - The probability of seeing an observation x with a specific label y.
    - For example, the dataset is about different types of vehicles, tell the model to generate an image of an 18-wheeler.

Generative Models (GM): Framework

- We have a dataset of observations X
- We assume that the observations have been generated according to some unknown distribution, P_{data}
- We want to build a GM P_{model} that mimics P_{data}. If we achieve this goal, we can sample from P_{model} to generate observations that appear to have been drawn from P_{data}
- The ideal for P_{model} is:

Generative Models (GM): Framework

- The ideal for P_{model} is:
  - Accuracy:
    - If the accuracy of P_{model} is high for a generated observation, it should look like it has been drawn from P_{data}
  - Generation
    - It should be possible to easily sample a new observation from P_{model}
  - Representation
    - It should be possible to understand how different high-level features in the data are represented by P_{model}

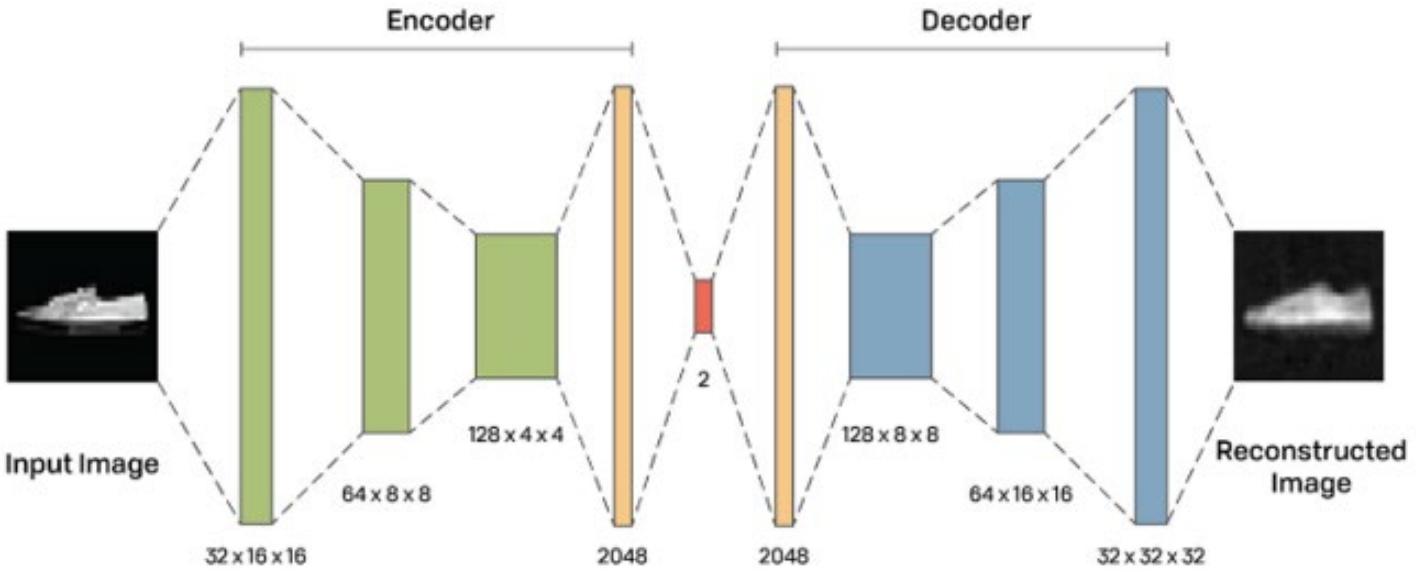
Generative Models (GM): Representation

- Representation of high-dimensional data (representation learning):
  - Instead of trying to the model high-dimensional data sample space directly, we describe each observation in the training set using lower-dimensional **latent space** and map it to a point in the original domain.

  - "Short definition": each point in the **latent space** is a representation of some high-dimensional observation

Generative Models (GM): Autoencoder

- Autoencoder:
  - Shor definition: A neural network that is trained to perform the task of encoding and decoding an item
    - The output should be as close to the original item as possible

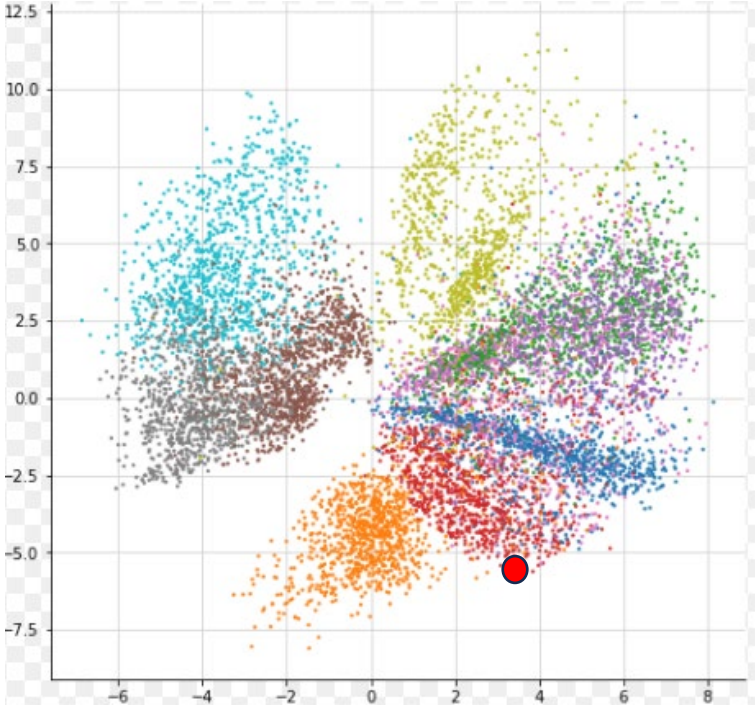


Generative Models (GM): Autoencoder

- Why reconstruct images that are already available?
  - Sampling from the latent space -> creation (generation) of new unseen image
- Embedding:
  - Compression of the original image into a lower-dimensional latent space
  - Done by the **Encoder**
- **Encoder**
  - Main job: to take an input image and map it to an embedding vector in the latent space

Generative Models (GM): Autoencoder: **Encoder**

- Encoder architecture:
  - Define the Input layer of the encoder (the image)
  - Stack Conv2D layers sequentially on top of each other
    - Remember the convolutional process for images:
      - Some pixel data will be lost due to filtering!
  - Flatten the last convolutional layer to a vector
  - **Connect this vector to the <u>2D embedding</u> with a Dense Layer**
    - This allows to plot the latent space



For example: a sample can be: [-2.5,-6.0]

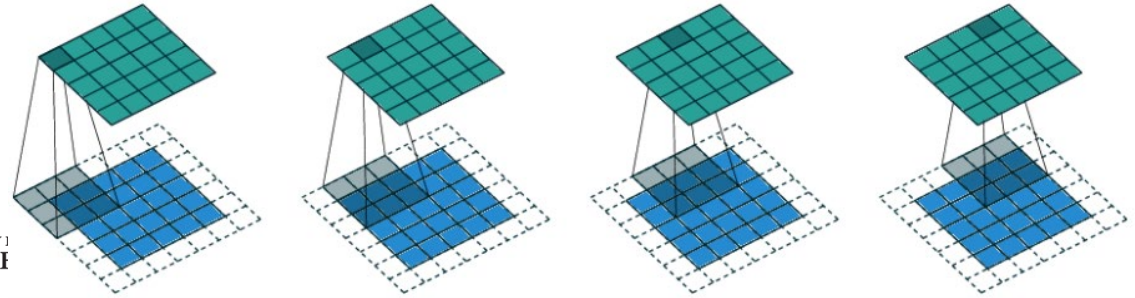
Generated image mapped to latent space

Generative Models
Autoencoder: Decoder

THE UNIVERSITY OF
SOUTHERN MISSISSIPPI

Generative Models (GM): Autoencoder: Decoder

- The decoder is a mirror image of the encoder:
  - Instead of convolutional layers, the decoder uses **convolutional transpose layers**.
  - Convolutional transpose layers:
    - In **regular convolutional layers**, the resulting output is reduced due to the filtering processes, remember key terms:
      - For example, an input image of 32x32x1 with stride = 2 will produce a 16x16x1 output
      - *Stride: step size used by the layer to move the filters across the input. By Increasing the stride, the size of the output (tensor) is reduced.
      - *Padding: “pads” the input data with zeros (zero padding) so that the output size from the layer is exactly the same as the input size when strides = 1

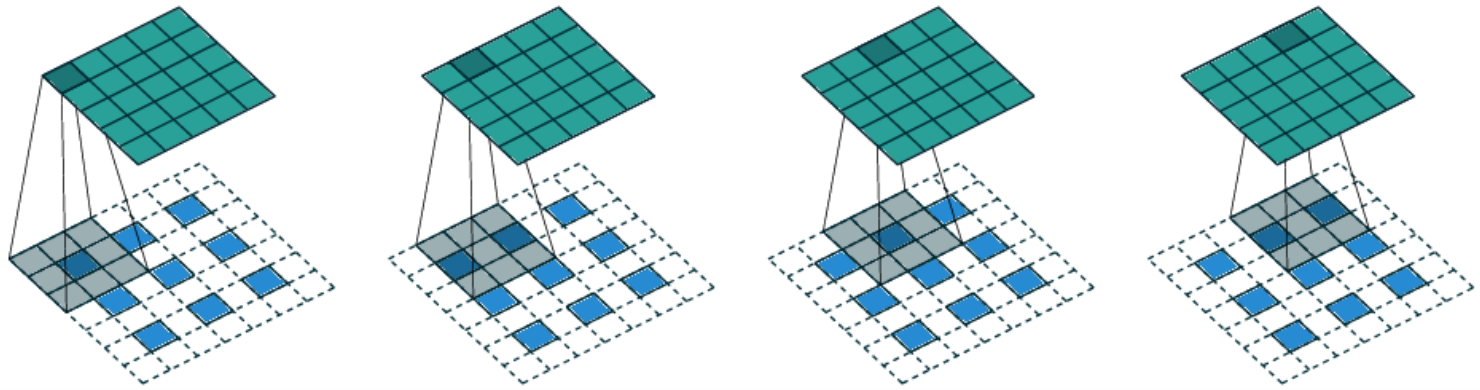


A 3x3x1 kernel (gray) being passed over a 5x5x1 input image (blue), with padding = ‘same’ and strides = 1 to generate the 5x5x1 output (green).
Padding = ‘same’ allows to easily keep track of the size of the tensor.

Generative Models (GM): Autoencoder: Decoder

- In the case of **convolutional transpose layers**:
  - Use the same principle as a standard convolutional layer, but setting **strides = 2** doubles the size of the input tensor in both dimensions.

  - In convolutional transpose layers, the strides parameter determines the internal zero padding between pixels in the image



A 3x3x1 filter (gray) is being passed across a 3x3x1 image (blue) with strides = 2 to produce a 6x6x1 output sensor (green).

The **Conv2DTransponse** method in Keras allows the conv. tranpose operations until the image reaches its original dimensions. The 0-padding method in transpose "adds" extra space according to the stride parameter.

Generative Models (GM): Autoencoder: Joining the Encoder and Decoder

- To train the encoder and decoder simultaneously we need to define a model that will represent the flow of image
  - Flow: Encoder and back out through the decoder

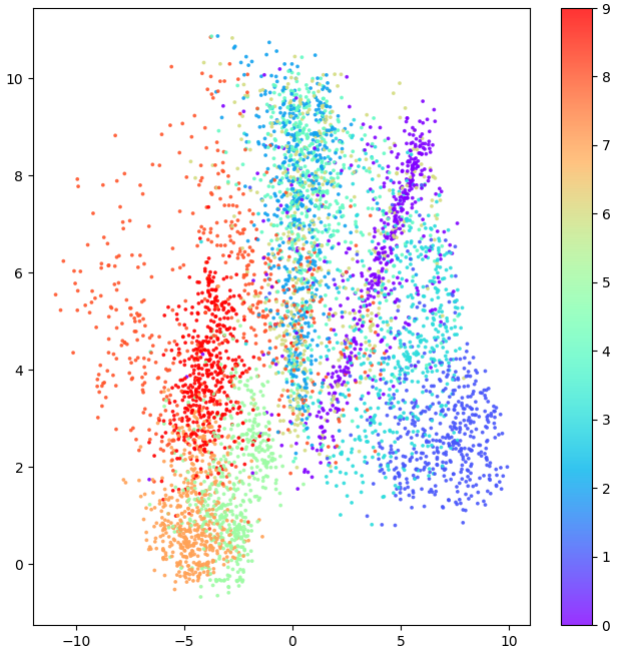
- In Keras, the flow can be specified:
  - Output from the autoencoder is:
    - The output from the encoder after it has been passed through the decoder.

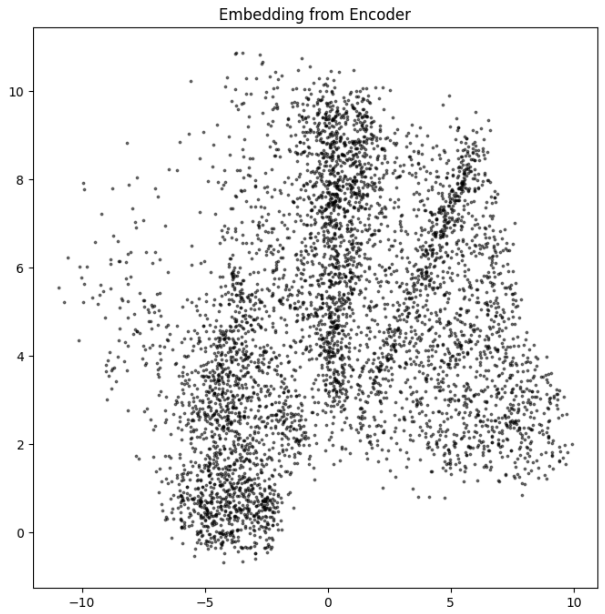
- `autoencoder = Model(encoder_input, decoder(encoder_output))`

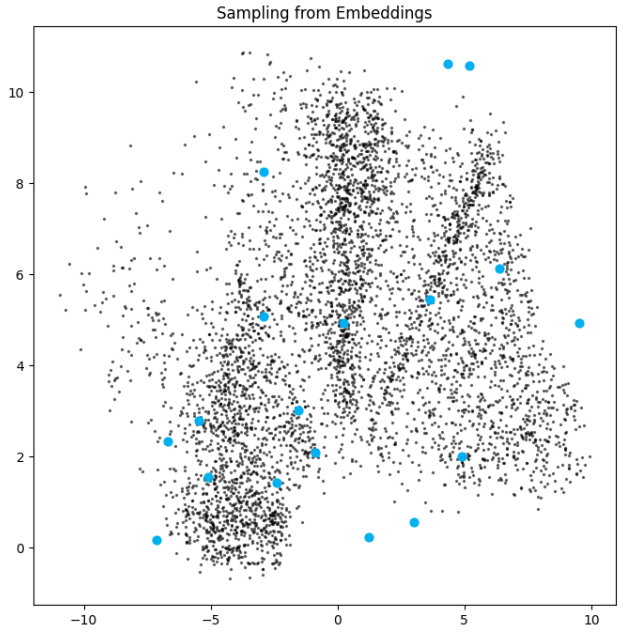
- Model takes an image and passes it through the encoder and back out through the decoder to generate a reconstruction of the original image.

Generative Models (GM): Decoder

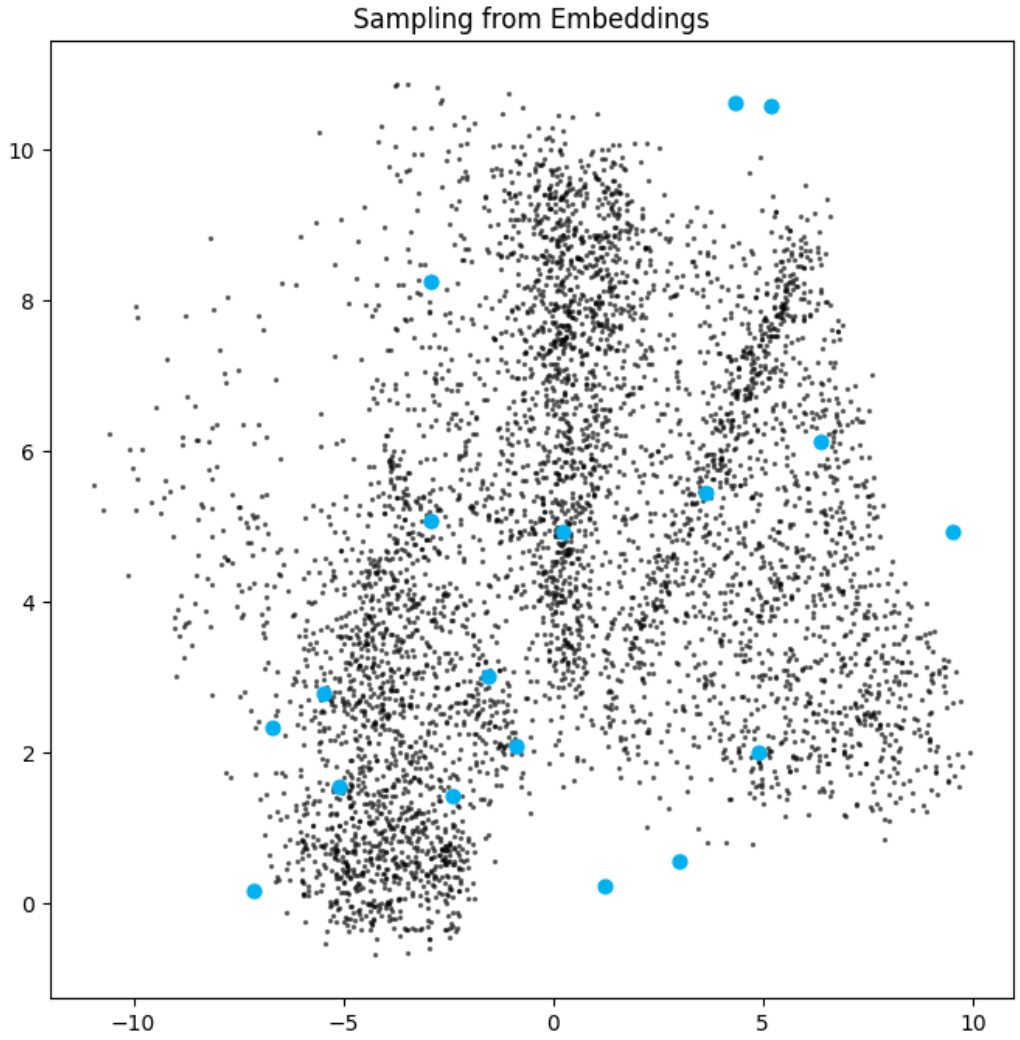
- Generating new images
  - We can generate novel images by sampling some points in the latent space and using the decoder to convert these back into pixel space.







Generative Models (GM): Decoder



Example

On Canvas: Download Autoencoder_test.py