Capstone Project

Probabilistic generative models

Instructions

In this notebook, you will practice working with generative models, using both normalising flow networks and the variational autoencoder algorithm. You will create a synthetic dataset with a normalising flow with randomised parameters. This dataset will then be used to train a variational autoencoder, and you will used the trained model to interpolate between the generated images. You will use concepts from throughout this course, including Distribution objects, probabilistic layers, bijectors, ELBO optimisation and KL divergence regularisers.

This project is peer-assessed. Within this notebook you will find instructions in each section for how to complete the project. Pay close attention to the instructions as the peer review will be carried out according to a grading rubric that checks key parts of the project instructions. Feel free to add extra cells into the notebook as required.

How to submit

When you have completed the Capstone project notebook, you will submit a pdf of the notebook for peer review. First ensure that the notebook has been fully executed from beginning to end, and all of the cell outputs are visible. This is important, as the grading rubric depends on the reviewer being able to view the outputs of your notebook. Save the notebook as a pdf (File -> Download as -> PDF via LaTeX). You should then submit this pdf for review.

Let's get started!

We'll start by running some imports below. For this project you are free to make further imports throughout the notebook as you wish.

```
import tensorflow as tf
import tensorflow_probability as tfp
tfd = tfp.distributions
tfb = tfp.bijectors
tfpl = tfp.layers

from tensorflow.keras import layers
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
```

Flags overview image

For the capstone project, you will create your own image dataset from contour plots of a transformed distribution using a random normalising flow network. You will then use the variational autoencoder algorithm to train generative and inference networks, and synthesise new images by interpolating in the latent space.

The normalising flow

- To construct the image dataset, you will build a normalising flow to transform the 2-D Gaussian random variable $z=(z_1, z_2)$, which has mean 0 and covariance matrix $\Sigma=\sigma^2 I_2$, with $\sigma=0.3$.
- This normalising flow uses bijectors that are parameterised by the following random variables:
  - $\theta \sim U¿$
  - $a \sim N(3,1)$

The complete normalising flow is given by the following chain of transformations:

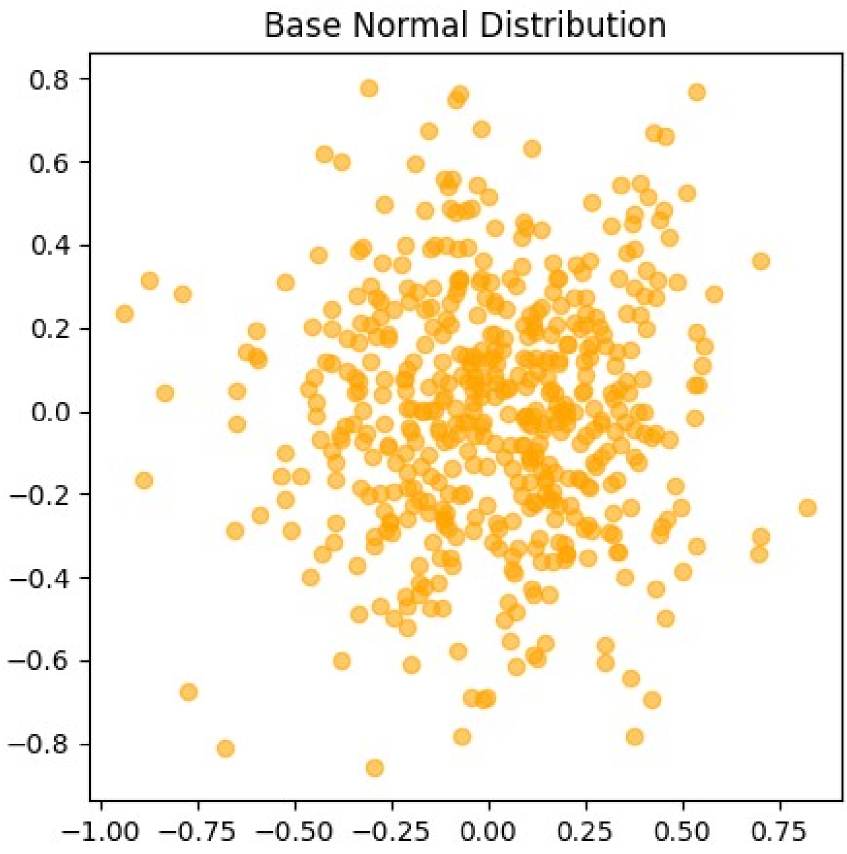
- $f_1(z)=(z_1, z_2-2)$,
- $f_2(z)=\left(z_1, \frac{z_2}{2}\right)$,
- $f_3(z)=(z_1, z_2+a z_1^2)$,
- $f_4(z)=R z$, where R is a rotation matrix with angle θ,
- $f_5(z)=\tanh(z)$, where the $\tanh$ function is applied elementwise.

The transformed random variable x is given by $x=f_5\left(f_4\left(f_3\left(f_2\left(f_1(z)\right)\right)\right)\right)$.

- You should use or construct bijectors for each of the transformations f_i, $i=1,\ldots,5$, and use `tfb.Chain` and `tfb.TransformedDistribution` to construct the final transformed distribution.
- Ensure to implement the `log_det_jacobian` methods for any subclassed bijectors that you write.
- Display a scatter plot of samples from the base distribution.
- Display 4 scatter plot images of the transformed distribution from your random normalising flow, using samples of θ and a. Fix the axes of these 4 plots to the range $[-1,1]$.

```
# Base Distribution

theta_dist = tfd.Uniform(low = 0, high = 2*np.pi)
a_dist = tfd.Normal(loc = 3, scale = 1)

mu, sigma = 0, 0.3
base_dist = tfd.MultivariateNormalDiag(loc = [mu, mu],  scale_diag = 
[sigma, sigma])

# Underlying Base Distribution

n = 500
```

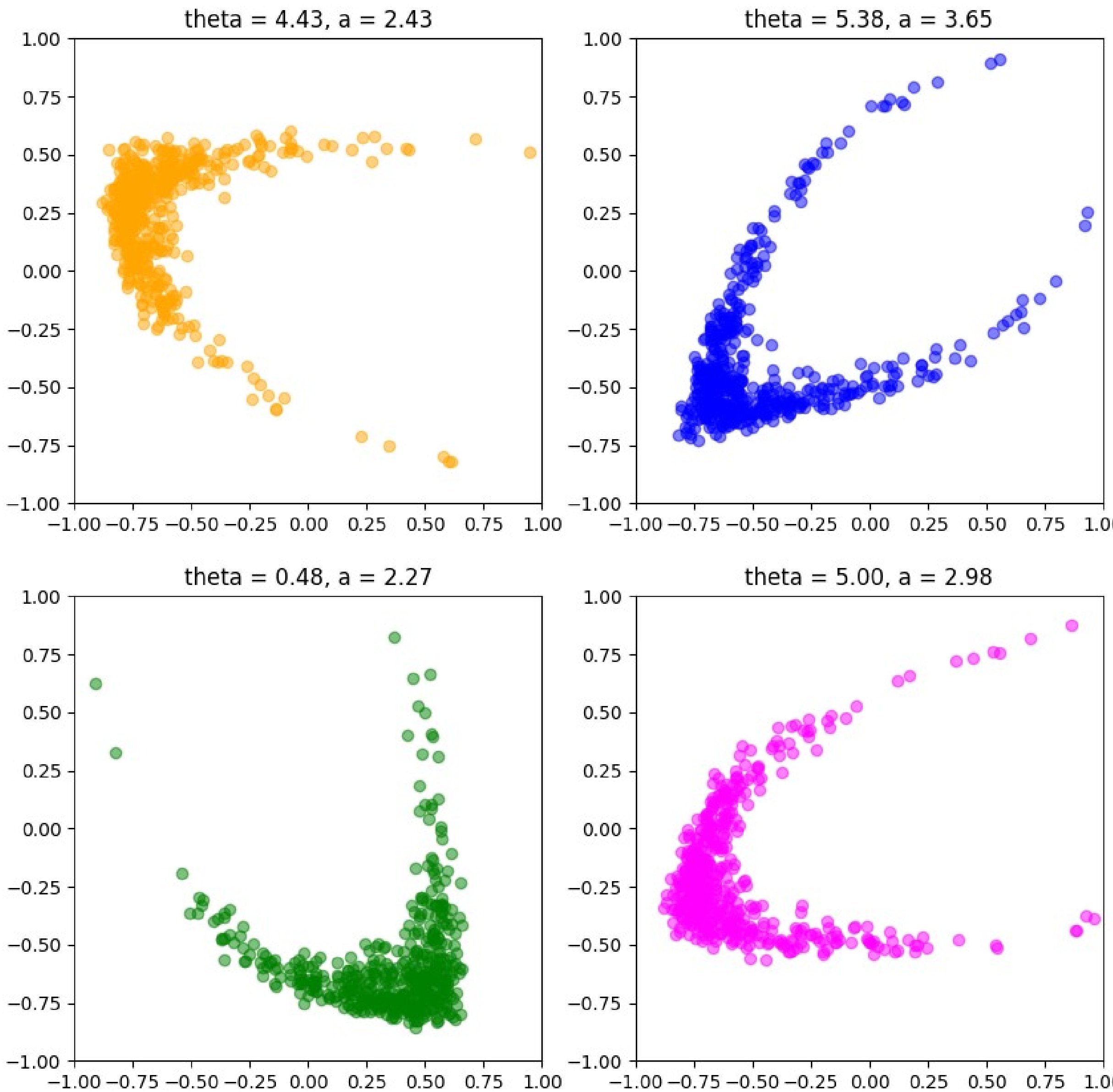
```
z = base_dist.sample(n).numpy().squeeze()
plt.figure(figsize=(5,5))
plt.scatter(z[:, 0], z[:, 1], color="orange", alpha=0.6)
plt.title("Base Normal Distribution")
plt.show()
```




```
class Polynomial(tfb.Bijector):
    def __init__(self, a, name="Polynomial", **kwargs):
        super(Polynomial, self).__init__(forward_min_event_ndims=1,
                                         name=name,
                                         is_constant_jacobian=True,
                                         validate_args=False,
                                         **kwargs)

        self.a = tf.cast(a, dtype=tf.float32)

    def _forward(self, x):
        x = tf.cast(x, dtype=tf.float32)
        return tf.concat([x[..., 0:1],
                          x[..., 1:] + self.a * tf.square(x[...,
0:1])], axis=-1)

    def _inverse(self, y):
```

```
        y = tf.cast(y, dtype=tf.float32)
        return tf.concat([y[..., 0:1],
                          y[..., 1:] - self.a * tf.square(y[...,
0:1])], axis=-1)

    def _forward_log_det_jacobian(self, x):
        return tf.constant(0., dtype=x.dtype)

class Rotation(tfb.Bijector):
    def __init__(self, theta, name="Rotation", **kwargs):
        super(Rotation, self).__init__(name=name,
                                       forward_min_event_ndims=1,
                                       validate_args=False,
                                       **kwargs)

        self.rot_matrix = tf.convert_to_tensor([[tf.cos(theta), -
tf.sin(theta)],
                                                [tf.sin(theta),
tf.cos(theta)]], dtype=tf.float32)

    def _forward(self, x):
        x = tf.cast(x, dtype=tf.float32)
        return tf.linalg.matvec(self.rot_matrix, x)

    def _inverse(self, y):
        y = tf.cast(y, dtype=tf.float32)
        return tf.linalg.matvec(tf.transpose(self.rot_matrix), y)

    def _forward_log_det_jacobian(self, x):
        return tf.constant(0., dtype=x.dtype)

# Bijectors

def GetFlow(theta, a):
    f1 = tfb.Shift([0, -2])
    f2 = tfb.Scale([1, 0.5])
    f3 = Polynomial(a)
    f4 = Rotation(theta)
    f5 = tfb.Tanh()

    return tfb.Chain([f5, f4, f3, f2, f1])

# Transformed Distribution of Bijector

GetFlowDist = lambda theta, a, base_dist:
tfd.TransformedDistribution(distribution=base_dist,

bijector=GetFlow(theta, a))

def PlotFlow(theta, a, flow, n_samples, color="blue"):
    samples = flow.sample(n_samples).numpy().squeeze()
```

```
    plt.scatter(samples[:,0], samples[:, 1], color=color, alpha=0.5)
    plt.title("theta = {:.2f}, a = {:.2f}".format(theta, a))
    plt.xlim([-1,1])
    plt.ylim([-1,1])

# Flow's Density Plots

n = 500

plt.figure(figsize = (10, 10))
for i, col in enumerate(["orange", "blue", "green", "magenta"]):
    # Parameter Sampling
    theta = theta_dist.sample(1).numpy()[0]
    a = a_dist.sample(1).numpy()[0]
    # Building a Normalizing Flow Distrubtion
    flow_dist = GetFlowDist(theta, a, base_dist)
    # Plotting the Samples.
    plt.subplot(2, 2, i+1)
    PlotFlow(theta, a, flow_dist, n, col)
plt.show()
```



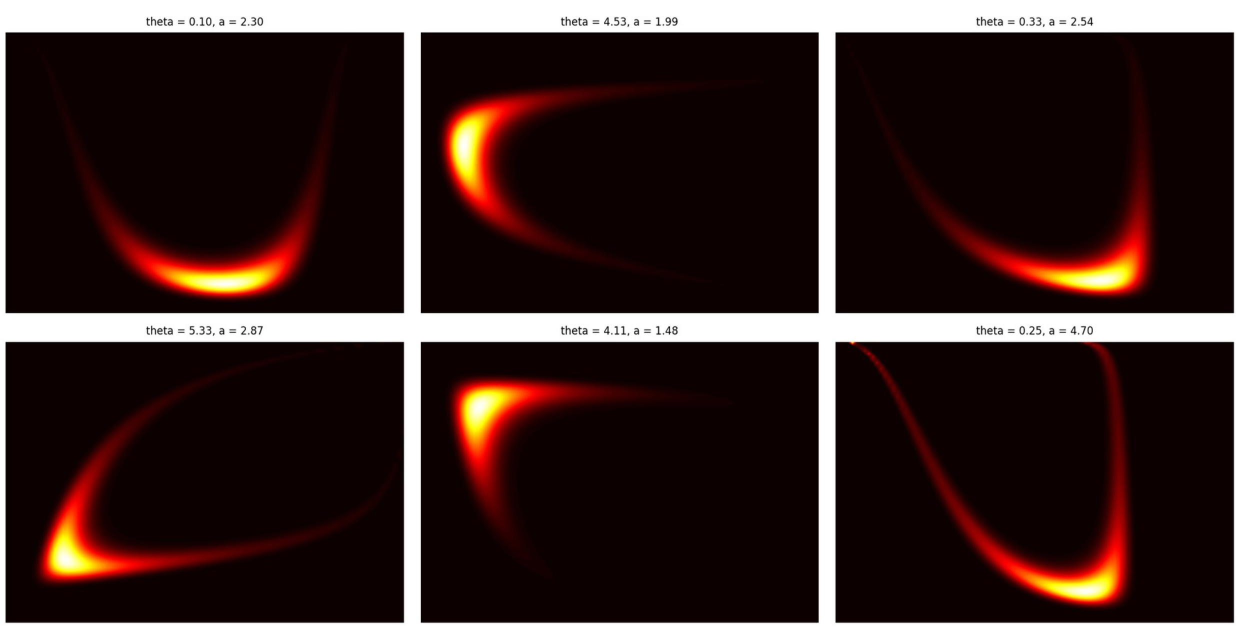
2. Create the image dataset

- You should now use your random normalising flow to generate an image dataset of contour plots from your random normalising flow network.
  - Feel free to get creative and experiment with different architectures to produce different sets of images!
- First, display a sample of 4 contour plot images from your normalising flow network using 4 independently sampled sets of parameters.
  - You may find the following `get_densities` function useful: this calculates density values for a (batched) Distribution for use in a contour plot.

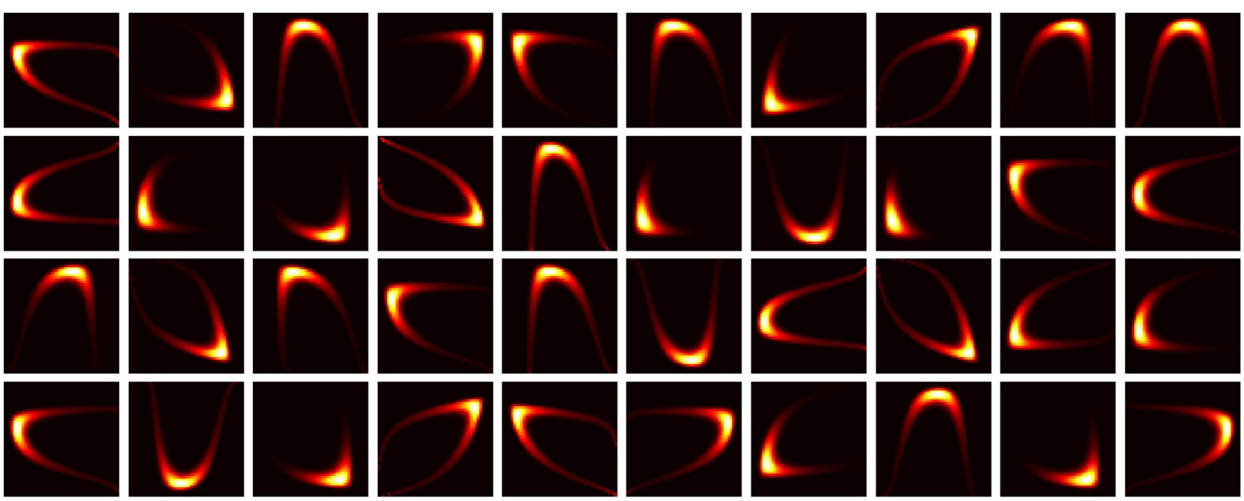
- Your dataset should consist of at least 1000 images, stored in a numpy array of shape `(N, 36, 36, 3)`. Each image in the dataset should correspond to a contour plot of a transformed distribution from a normalising flow with an independently sampled set of parameters s, T, S, b. It will take a few minutes to create the dataset.
- As well as the `get_densities` function, the `get_image_array_from_density_values` function will help you to generate the dataset.
    - This function creates a numpy array for an image of the contour plot for a given set of density values Z. Feel free to choose your own options for the contour plots.
- Display a sample of 20 images from your generated dataset in a figure.

```
# Helper function to compute transformed distribution densities

X, Y = np.meshgrid(np.linspace(-1, 1, 100), np.linspace(-1, 1, 100))
inputs = np.transpose(np.stack((X, Y)), [1, 2, 0])

def get_densities(transformed_distribution):
    """
    This function takes a (batched) Distribution object as an
argument, and returns a numpy
    array Z of shape (batch_shape, 100, 100) of density values, that
can be used to make a
    contour plot with:
    plt.contourf(X, Y, Z[b, ...], cmap='hot', levels=100)
    where b is an index into the batch shape.
    """
    batch_shape = transformed_distribution.batch_shape
    Z = transformed_distribution.prob(np.expand_dims(inputs, 2))
    Z = np.transpose(Z, list(range(2, 2+len(batch_shape))) + [0, 1])
    return Z

# Helper function to convert contour plots to numpy arrays

import numpy as np
from matplotlib.backends.backend_agg import FigureCanvasAgg as
FigureCanvas
from matplotlib.figure import Figure

def get_image_array_from_density_values(Z):
    """
    This function takes a numpy array Z of density values of shape
(100, 100)
    and returns an integer numpy array of shape (36, 36, 3) of pixel
values for an image.
    """
    assert Z.shape == (100, 100)
    fig = Figure(figsize=(0.5, 0.5))
    canvas = FigureCanvas(fig)
    ax = fig.gca()
```

```
    ax.contourf(X, Y, Z, cmap='hot', levels=100)
    ax.axis('off')
    fig.tight_layout(pad=0)

    ax.margins(0)
    fig.canvas.draw()
    image_from_plot = np.frombuffer(fig.canvas.tostring_rgb(),
dtype=np.uint8)
    image_from_plot =
image_from_plot.reshape(fig.canvas.get_width_height()[::-1] + (3,))
    return image_from_plot

# Flow's Images Plots

plt.figure(figsize = (20, 10))
for i in range(6):
    # Parameter Sampling
    theta = theta_dist.sample(1).numpy()[0]
    a = a_dist.sample(1).numpy()[0]
    # Building a Normalizing Flow Distrubtion
    flow_dist = GetFlowDist(theta, a, base_dist)
    flow_dist = tfd.BatchReshape(flow_dist, [1])
    # Contour Plot
    plt.subplot(2, 3, i+1)
    plt.contourf(X, Y, get_densities(flow_dist).squeeze(), cmap='hot',
levels=100)
    plt.title("theta = {:.2f}, a = {:.2f}".format(theta, a))
    plt.axis('off')
plt.tight_layout()
plt.show()
```



```python
# Generating Images

images = []
img_params = []
N = 1000

for _ in range(N):
    # Parameter Sampling
    theta = theta_dist.sample(1).numpy()[0]
    a = a_dist.sample(1).numpy()[0]
    # Building a Normalizing Flow Distrubtion
    flow_dist = GetFlowDist(theta, a, base_dist)
    flow_dist = tfd.BatchReshape(flow_dist, [1])
    # Getting Density
    Z = get_densities(flow_dist).squeeze()
    #Saving Images
    images.append(get_image_array_from_density_values(Z))

images = np.array(images)

# Plotting a subset of iamges from the dataset.

plt.figure(figsize=(20, 8))
for i in range(40):
    plt.subplot(4, 10, i+1)
    idx = np.random.randint(0, N)
    plt.imshow(images[idx])
    plt.axis("off")
plt.tight_layout()
plt.show()
```



3. Make `tf.data.Dataset` objects

- You should now split your dataset to create `tf.data.Dataset` objects for training and validation data.

- Using the `map` method, normalise the pixel values so that they lie between 0 and 1.
- These Datasets will be used to train a variational autoencoder (VAE). Use the `map` method to return a tuple of input and output Tensors where the image is duplicated as both input and output.
- Randomly shuffle the training Dataset.
- Batch both datasets with a batch size of 20, setting `drop_remainder=True`.
- Print the `element_spec` property for one of the Dataset objects.

```
def test_train_split(data, test_fraction):
    data = data.astype(dtype=np.float32)
    N = data.shape[0]
    test_idx = np.random.choice(np.arange(N), int(test_fraction*N),
replace=False)
    train_idx = np.setdiff1d(np.arange(N), test_idx)

    return data[train_idx], data[test_idx]

def GetDataset(data, test_fraction, batch_size=20):
    train, test = test_train_split(data, test_fraction)

    train = tf.data.Dataset.from_tensor_slices(train)
    train = train.map(lambda x: x/255.0)
    train = train.map(lambda x: (x, x))
    train = train.batch(batch_size, drop_remainder=True)

    test = tf.data.Dataset.from_tensor_slices(test)
    test = test.map(lambda x: x/255.0)
    test = test.map(lambda x: (x, x))
    test = test.batch(batch_size, drop_remainder=True)

    return train, test

# Dataset Creation

batch_size = 20
train_data, test_data = GetDataset(images, 0.2)

print(train_data.element_spec)
print(test_data.element_spec)
```

```
(TensorSpec(shape=(20, 50, 50, 3), dtype=tf.float32, name=None),
TensorSpec(shape=(20, 50, 50, 3), dtype=tf.float32, name=None))
(TensorSpec(shape=(20, 50, 50, 3), dtype=tf.float32, name=None),
TensorSpec(shape=(20, 50, 50, 3), dtype=tf.float32, name=None))
```

4. Build the encoder and decoder networks

- You should now create the encoder and decoder for the variational autoencoder algorithm.
- You should design these networks yourself, subject to the following constraints:

- The encoder and decoder networks should be built using the `Sequential` class.
- The encoder and decoder networks should use probabilistic layers where necessary to represent distributions.
- The prior distribution should be a zero-mean, isotropic Gaussian (identity covariance matrix).
- The encoder network should add the KL divergence loss to the model.

- Print the model summary for the encoder and decoder networks.

```
# Image Dimensions

image_dims = images.shape[1:]
image_dims

(50, 50, 3)

# Latent Space Dimension

latent_dim = 2

def get_prior(latent_dim):
    prior = tfd.MultivariateNormalDiag(loc =
tf.Variable(tf.zeros(latent_dim), dtype=tf.float32),
                                       scale_diag =
tfp.util.TransformedVariable(initial_value = tf.ones(latent_dim,
dtype=tf.float32),

bijector = tfb.Softplus(),

dtype = tf.float32)
                                       )

    return prior

def get_encoder(latent_dim):
    prior = get_prior(latent_dim)

    model = tf.keras.models.Sequential([

layers.InputLayer(input_shape=image_dims),

                               layers.Conv2D(filters=32,
kernel_size=(3,3)),
                               layers.BatchNormalization(),
                               layers.LeakyReLU(0.2),

                               layers.Conv2D(filters=64,
kernel_size=(3,3)),
                               layers.BatchNormalization(),
                               layers.LeakyReLU(0.2),

                               layers.Conv2D(filters=128,
```

```
kernel_size=(3,3)),
                               layers.BatchNormalization(),
                               layers.LeakyReLU(0.2),

                               layers.Conv2D(filters=256,
kernel_size=(3,3)),
                               layers.BatchNormalization(),
                               layers.LeakyReLU(0.2),

                               layers.Conv2D(filters=8,
kernel_size=(1,1)),
                               layers.BatchNormalization(),

                               layers.Flatten(),

                               layers.Dense(100),
                               layers.BatchNormalization(),
                               layers.LeakyReLU(0.2),


layers.Dense(tfpl.MultivariateNormalTriL.params_size(latent_dim)),

tfpl.MultivariateNormalTriL(latent_dim),

                               tfpl.KLDivergenceAddLoss(prior,
                                                        use_exact_kl =
False,
                                                        test_points_fn
= lambda q:q.sample(5),

test_points_reduce_axis=(0,1))
    ])

    return model

enc_model = get_encoder(latent_dim)

enc_model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 48, 48, 32)        896

 batch_normalization (BatchN  (None, 48, 48, 32)       128
 ormalization)

 leaky_re_lu (LeakyReLU)     (None, 48, 48, 32)        0

 conv2d_1 (Conv2D)           (None, 46, 46, 64)        18496
```

```
 batch_normalization_1 (Batc  (None, 46, 46, 64)       256
 hNormalization)

 leaky_re_lu_1 (LeakyReLU)   (None, 46, 46, 64)        0

 conv2d_2 (Conv2D)           (None, 44, 44, 128)       73856

 batch_normalization_2 (Batc  (None, 44, 44, 128)      512
 hNormalization)

 leaky_re_lu_2 (LeakyReLU)   (None, 44, 44, 128)       0

 conv2d_3 (Conv2D)           (None, 42, 42, 256)       295168

 batch_normalization_3 (Batc  (None, 42, 42, 256)      1024
 hNormalization)

 leaky_re_lu_3 (LeakyReLU)   (None, 42, 42, 256)       0

 conv2d_4 (Conv2D)           (None, 42, 42, 8)         2056

 batch_normalization_4 (Batc  (None, 42, 42, 8)        32
 hNormalization)

 flatten (Flatten)           (None, 14112)             0

 dense (Dense)               (None, 100)               1411300

 batch_normalization_5 (Batc  (None, 100)              400
 hNormalization)

 leaky_re_lu_4 (LeakyReLU)   (None, 100)               0

 dense_1 (Dense)             (None, 5)                 505

 multivariate_normal_tri_l (  ((None, 2),              0
 MultivariateNormalTriL)      (None, 2))

 kl_divergence_add_loss (KLD  (None, 2)                4
 ivergenceAddLoss)

=================================================================
Total params: 1,804,633
Trainable params: 1,803,457
Non-trainable params: 1,176
_________________________________________________________________

def get_decoder(latent_dim, image_dim):
    model = tf.keras.models.Sequential([
```

```
layers.InputLayer(input_shape=(latent_dim,)),

                              layers.Dense(64),
                              layers.BatchNormalization(),
                              layers.LeakyReLU(0.2),

                              layers.Dense(128),
                              layers.BatchNormalization(),
                              layers.LeakyReLU(0.2),

                              layers.Dense(256),
                              layers.BatchNormalization(),
                              layers.LeakyReLU(0.2),

                              layers.Reshape(target_shape=(8, 8, 4)),

                              layers.UpSampling2D(size=(3,3)),
                              layers.Conv2D(filters=128,
kernel_size=(3,3)),
                              layers.BatchNormalization(),
                              layers.LeakyReLU(0.2),

                              layers.UpSampling2D(size=(2, 2)),
                              layers.Conv2D(filters=64,
kernel_size=(3,3)),
                              layers.BatchNormalization(),
                              layers.LeakyReLU(0.2),

                              layers.Conv2D(filters=32,
kernel_size=(3,3)),
                              layers.BatchNormalization(),
                              layers.LeakyReLU(0.2),

                              layers.Conv2D(filters=16,
kernel_size=(3,3)),
                              layers.BatchNormalization(),
                              layers.LeakyReLU(0.2),

                              layers.Conv2D(filters=1,
kernel_size=(3, 3), strides=(2,2)),
                              layers.BatchNormalization(),
                              layers.LeakyReLU(0.2),

                              layers.Flatten(),


layers.Dense(tfpl.IndependentBernoulli.params_size(image_dim)),


tfpl.IndependentBernoulli(event_shape=image_dim)
```

```
    ])

    return model

dec_model = get_decoder(latent_dim, image_dims)
dec_model.summary()
```

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #   
=================================================================
 dense_2 (Dense)             (None, 64)                192       
                                                                 
 batch_normalization_6 (Batc  (None, 64)               256       
 hNormalization)                                                 
                                                                 
 leaky_re_lu_5 (LeakyReLU)   (None, 64)                0         
                                                                 
 dense_3 (Dense)             (None, 128)               8320      
                                                                 
 batch_normalization_7 (Batc  (None, 128)              512       
 hNormalization)                                                 
                                                                 
 leaky_re_lu_6 (LeakyReLU)   (None, 128)               0         
                                                                 
 dense_4 (Dense)             (None, 256)               33024     
                                                                 
 batch_normalization_8 (Batc  (None, 256)              1024      
 hNormalization)                                                 
                                                                 
 leaky_re_lu_7 (LeakyReLU)   (None, 256)               0         
                                                                 
 reshape (Reshape)           (None, 8, 8, 4)           0         
                                                                 
 up_sampling2d (UpSampling2D  (None, 24, 24, 4)        0         
 )                                                               
                                                                 
 conv2d_5 (Conv2D)           (None, 22, 22, 128)       4736      
                                                                 
 batch_normalization_9 (Batc  (None, 22, 22, 128)      512       
 hNormalization)                                                 
                                                                 
 leaky_re_lu_8 (LeakyReLU)   (None, 22, 22, 128)       0         
                                                                 
 up_sampling2d_1 (UpSampling  (None, 44, 44, 128)      0         
 2D)                                                             
                                                                 
 conv2d_6 (Conv2D)           (None, 42, 42, 64)        73792     
                                                                 
 batch_normalization_10 (Bat  (None, 42, 42, 64)       256       
```

```
 chNormalization)

 leaky_re_lu_9 (LeakyReLU)   (None, 42, 42, 64)        0

 conv2d_7 (Conv2D)           (None, 40, 40, 32)        18464

 batch_normalization_11 (Bat  (None, 40, 40, 32)       128
 chNormalization)

 leaky_re_lu_10 (LeakyReLU)  (None, 40, 40, 32)        0

 conv2d_8 (Conv2D)           (None, 38, 38, 16)        4624

 batch_normalization_12 (Bat  (None, 38, 38, 16)       64
 chNormalization)

 leaky_re_lu_11 (LeakyReLU)  (None, 38, 38, 16)        0

 conv2d_9 (Conv2D)           (None, 18, 18, 1)         145

 batch_normalization_13 (Bat  (None, 18, 18, 1)        4
 chNormalization)

 leaky_re_lu_12 (LeakyReLU)  (None, 18, 18, 1)         0

 flatten_1 (Flatten)         (None, 324)               0

 dense_5 (Dense)             (None, 7500)              2437500

 independent_bernoulli (Inde  ((None, 50, 50, 3),      0
 pendentBernoulli)            (None, 50, 50, 3))

=================================================================
Total params: 2,583,553
Trainable params: 2,582,175
Non-trainable params: 1,378
_________________________________________________________________
```

5. Train the variational autoencoder

- You should now train the variational autoencoder. Build the VAE using the `Model` class and the encoder and decoder models. Print the model summary.
- Compile the VAE with the negative log likelihood loss and train with the `fit` method, using the training and validation Datasets.
- Plot the learning curves for loss vs epoch for both training and validation sets.

```
# Reconstruction Loss
```

```
NLL = lambda real, estimated: -
tf.reduce_mean(estimated.log_prob(real))

# Variational Autoencoder

def get_vae(encoder, decoder):
    vae = tf.keras.models.Model(inputs = encoder.inputs,
                          outputs= decoder(encoder.outputs))

    vae.compile(optimizer="adam", loss=NLL)

    return vae

vae_model = get_vae(enc_model, dec_model)
vae_model.summary()

Model: "model"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_1 (InputLayer)        [(None, 50, 50, 3)]       0

 conv2d (Conv2D)             (None, 48, 48, 32)        896

 batch_normalization (BatchN  (None, 48, 48, 32)       128
 ormalization)

 leaky_re_lu (LeakyReLU)     (None, 48, 48, 32)        0

 conv2d_1 (Conv2D)           (None, 46, 46, 64)        18496

 batch_normalization_1 (Batc  (None, 46, 46, 64)       256
 hNormalization)

 leaky_re_lu_1 (LeakyReLU)   (None, 46, 46, 64)        0

 conv2d_2 (Conv2D)           (None, 44, 44, 128)       73856

 batch_normalization_2 (Batc  (None, 44, 44, 128)      512
 hNormalization)

 leaky_re_lu_2 (LeakyReLU)   (None, 44, 44, 128)       0

 conv2d_3 (Conv2D)           (None, 42, 42, 256)       295168

 batch_normalization_3 (Batc  (None, 42, 42, 256)      1024
 hNormalization)

 leaky_re_lu_3 (LeakyReLU)   (None, 42, 42, 256)       0
```

```
 conv2d_4 (Conv2D)           (None, 42, 42, 8)         2056

 batch_normalization_4 (Batc  (None, 42, 42, 8)        32
 hNormalization)

 flatten (Flatten)           (None, 14112)             0

 dense (Dense)               (None, 100)               1411300

 batch_normalization_5 (Batc  (None, 100)              400
 hNormalization)

 leaky_re_lu_4 (LeakyReLU)   (None, 100)               0

 dense_1 (Dense)             (None, 5)                 505

 multivariate_normal_tri_l (  ((None, 2),              0
 MultivariateNormalTriL)      (None, 2))

 kl_divergence_add_loss (KLD  (None, 2)                4
 ivergenceAddLoss)

 sequential_1 (Sequential)   (None, 50, 50, 3)         2583553

=================================================================
Total params: 4,388,186
Trainable params: 4,385,632
Non-trainable params: 2,554
_________________________________________________________________
```

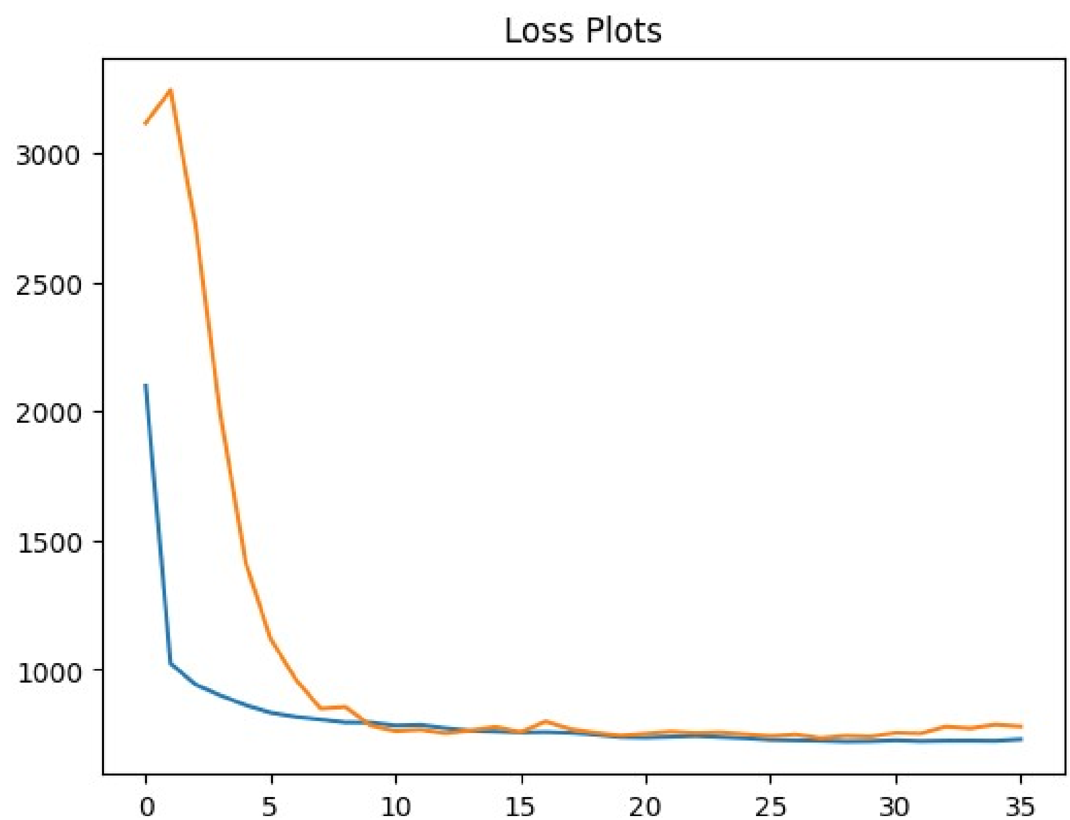
```
# Model Training

es_callback = tf.keras.callbacks.EarlyStopping(monitor="val_loss",
                                               min_delta=0.1,
                                               patience=8,
                                               restore_best_weights=True)


history = vae_model.fit(train_data,
                        validation_data=test_data,
                        epochs=500,
                        callbacks=[es_callback])
```

```
Epoch 1/500
40/40 [==============================] - 26s 61ms/step - loss:
2099.9761 - val_loss: 3117.8516
Epoch 2/500
40/40 [==============================] - 2s 42ms/step - loss:
1023.4821 - val_loss: 3246.1821
Epoch 3/500
```

```
40/40 [==============================] - 2s 41ms/step - loss: 941.8669
- val_loss: 2717.6204
Epoch 4/500
40/40 [==============================] - 2s 40ms/step - loss: 899.5288
- val_loss: 1983.3750
Epoch 5/500
40/40 [==============================] - 2s 40ms/step - loss: 862.0684
- val_loss: 1416.1327
Epoch 6/500
40/40 [==============================] - 2s 41ms/step - loss: 833.2338
- val_loss: 1118.6913
Epoch 7/500
40/40 [==============================] - 2s 41ms/step - loss: 817.0461
- val_loss: 962.8014
Epoch 8/500
40/40 [==============================] - 2s 42ms/step - loss: 806.4696
- val_loss: 848.8140
Epoch 9/500
40/40 [==============================] - 2s 42ms/step - loss: 795.1638
- val_loss: 855.0101
Epoch 10/500
40/40 [==============================] - 2s 41ms/step - loss: 794.7696
- val_loss: 783.2071
Epoch 11/500
40/40 [==============================] - 2s 41ms/step - loss: 782.9130
- val_loss: 761.9182
Epoch 12/500
40/40 [==============================] - 2s 40ms/step - loss: 785.5213
- val_loss: 767.4680
Epoch 13/500
40/40 [==============================] - 2s 42ms/step - loss: 772.7452
- val_loss: 752.7924
Epoch 14/500
40/40 [==============================] - 2s 46ms/step - loss: 763.2082
- val_loss: 763.9762
Epoch 15/500
40/40 [==============================] - 2s 43ms/step - loss: 760.4654
- val_loss: 777.8560
Epoch 16/500
40/40 [==============================] - 2s 41ms/step - loss: 756.0848
- val_loss: 757.5745
Epoch 17/500
40/40 [==============================] - 2s 47ms/step - loss: 757.5262
- val_loss: 798.4344
Epoch 18/500
40/40 [==============================] - 2s 43ms/step - loss: 754.5084
- val_loss: 769.3860
Epoch 19/500
40/40 [==============================] - 2s 40ms/step - loss: 747.3674
```

```
- val_loss: 755.1434
Epoch 20/500
40/40 [==============================] - 2s 42ms/step - loss: 737.9172
- val_loss: 744.5165
Epoch 21/500
40/40 [==============================] - 2s 42ms/step - loss: 736.3153
- val_loss: 751.6827
Epoch 22/500
40/40 [==============================] - 2s 42ms/step - loss: 739.0023
- val_loss: 759.7833
Epoch 23/500
40/40 [==============================] - 2s 41ms/step - loss: 742.3630
- val_loss: 754.0919
Epoch 24/500
40/40 [==============================] - 2s 40ms/step - loss: 738.0756
- val_loss: 756.7106
Epoch 25/500
40/40 [==============================] - 2s 40ms/step - loss: 733.8170
- val_loss: 749.6890
Epoch 26/500
40/40 [==============================] - 2s 41ms/step - loss: 727.5867
- val_loss: 742.3588
Epoch 27/500
40/40 [==============================] - 2s 41ms/step - loss: 725.8134
- val_loss: 748.8053
Epoch 28/500
40/40 [==============================] - 2s 42ms/step - loss: 724.2389
- val_loss: 733.5547
Epoch 29/500
40/40 [==============================] - 2s 41ms/step - loss: 721.1714
- val_loss: 743.1306
Epoch 30/500
40/40 [==============================] - 2s 41ms/step - loss: 722.1490
- val_loss: 740.0511
Epoch 31/500
40/40 [==============================] - 2s 41ms/step - loss: 725.7937
- val_loss: 754.8381
Epoch 32/500
40/40 [==============================] - 2s 40ms/step - loss: 722.4783
- val_loss: 752.1525
Epoch 33/500
40/40 [==============================] - 2s 40ms/step - loss: 724.0310
- val_loss: 779.0916
Epoch 34/500
40/40 [==============================] - 2s 42ms/step - loss: 724.2294
- val_loss: 772.5142
Epoch 35/500
40/40 [==============================] - 2s 40ms/step - loss: 723.5015
- val_loss: 786.8664
```

```
Epoch 36/500
40/40 [==============================] - 2s 42ms/step - loss: 729.2704
- val_loss: 779.6308

# Save Mode Weights

vae_model.save_weights('/content/my_checkpoint')

# Training Visualization

plt.plot(history.history["loss"])
plt.plot(history.history["val_loss"])
plt.title("Loss Plots")

plt.show()
```



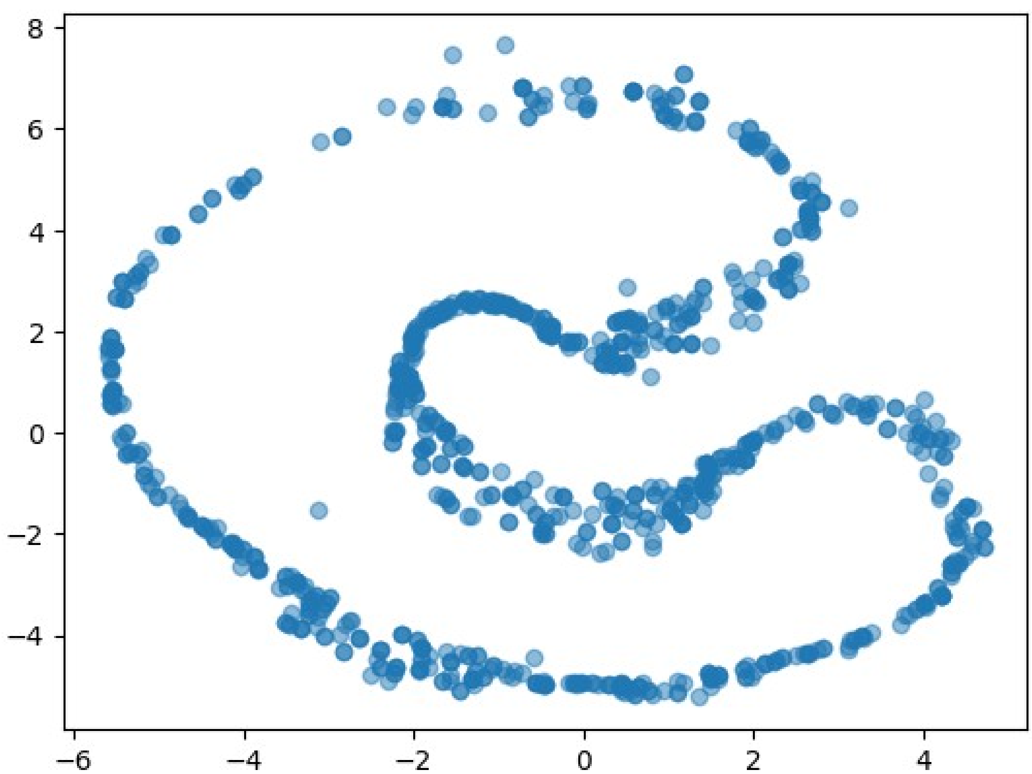
6. Use the encoder and decoder networks

- You can now put your encoder and decoder networks into practice!
- Randomly sample 1000 images from the dataset, and pass them through the encoder. Display the embeddings in a scatter plot (project to 2 dimensions if the latent space has dimension higher than two).
- Randomly sample 4 images from the dataset and for each image, display the original and reconstructed image from the VAE in a figure.

    – Use the mean of the output distribution to display the images.

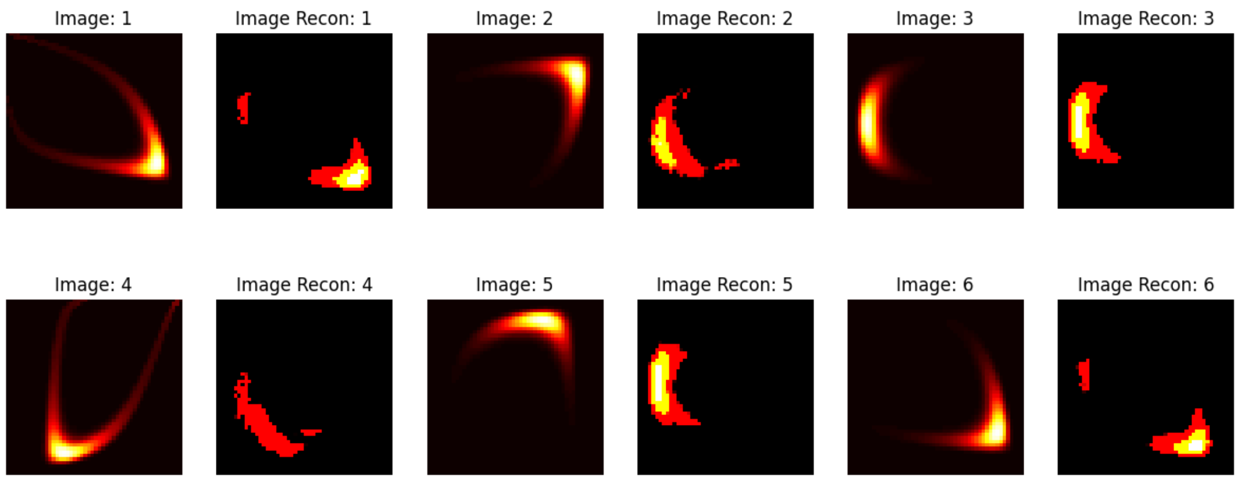
- Randomly sample 6 latent variable realisations from the prior distribution, and display the images in a figure.
    – Again use the mean of the output distribution to display the images.

```
# Sampling the Images fom Dataset

N = 1000
idx = np.random.choice(np.arange(images.shape[0]), N)

embeddings = enc_model(images[idx]/255.0).mean()

# Embedding Visualization

plt.scatter(embeddings[:,0], embeddings[:,1], alpha=0.5)
plt.show()
```



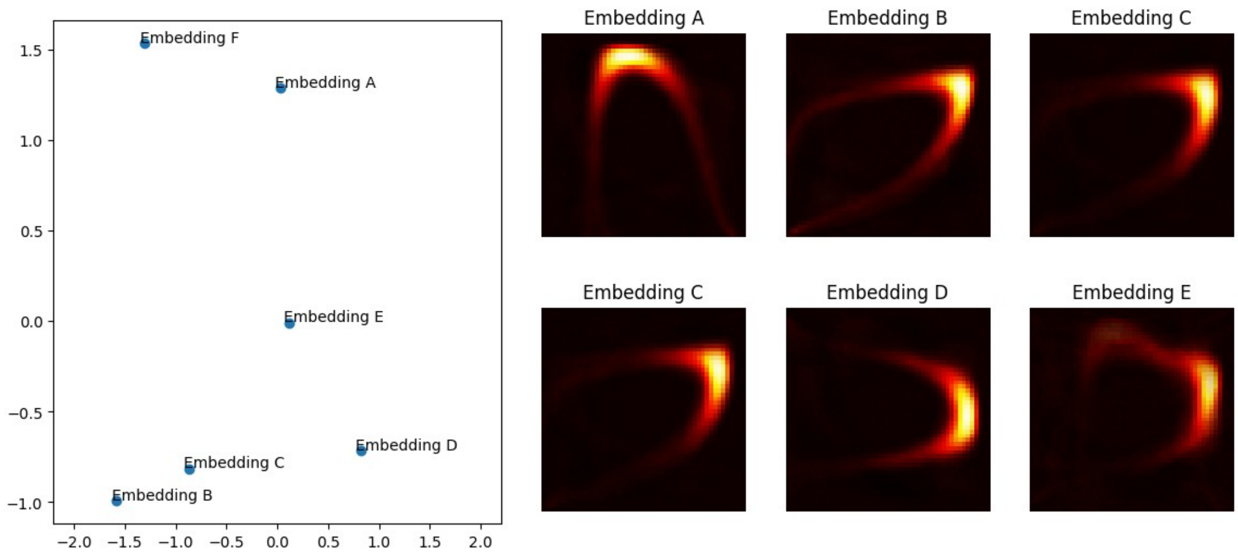

```
# Images an Reocnstructed Image Pairs

N = 6
idx = np.random.choice(np.arange(images.shape[0]), N)
rec_images = vae_model(images[idx]).mean().numpy()

plt.figure(figsize=(15, 6))
for i in range(N):
    plt.subplot(2, 6, 2*i+1)
```

```
    plt.imshow(images[idx[i]])
    plt.title("Image: {}".format(i+1))
    plt.axis("off")

    plt.subplot(2, 6, 2*i+2)
    plt.imshow(rec_images[i])
    plt.title("Image Recon: {}".format(i+1))
    plt.axis("off")
plt.show()
```




```
# Random Latent Space Embedding Visualization

from matplotlib import gridspec

N = 6
embeddings = np.random.uniform(-2, 2, (N, latent_dim))
rec_images = dec_model(embeddings).mean()


fig = plt.figure(figsize=(14, 6))
gs = gridspec.GridSpec(2, 5)

ax0 = plt.subplot(gs[:, 0:2])
ax0.scatter(embeddings[:, 0], embeddings[:, 1])
for i in range(N):
    ax0.annotate("Embedding "+chr(ord("A")+i), (embeddings[i, 0]-0.05,
embeddings[i, 1]+7e-3))
ax0.set_xlim(-2.2, 2.2)

for i in range(2):
    for j in range(3):
        ax1 = plt.subplot(gs[i, 2+j])
        ax1.imshow(rec_images[2*i + j])
        ax1.set_axis_off()
```

```
        ax1.set_title("Embedding "+chr(ord("A")+(2*i + j)))

plt.show()
```




```
!apt-get install texlive texlive-xetex texlive-latex-extra pandoc
!pip install pypandoc
```

```
Building dependency tree... Done
Reading state information... Done
pandoc is already the newest version (2.9.2.1-3ubuntu2).
texlive is already the newest version (2021.20220204-1).
texlive-latex-extra is already the newest version (2021.20220204-1).
texlive-xetex is already the newest version (2021.20220204-1).
0 upgraded, 0 newly installed, 0 to remove and 16 not upgraded.
Requirement already satisfied: pypandoc in
/usr/local/lib/python3.10/dist-packages (1.11)
```

```
!sudo apt-get install texlive-xetex texlive-fonts-recommended texlive-
plain-generic
```

```
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
texlive-fonts-recommended is already the newest version
(2021.20220204-1).
texlive-plain-generic is already the newest version (2021.20220204-1).
```

```
texlive-xetex is already the newest version (2021.20220204-1).
0 upgraded, 0 newly installed, 0 to remove and 16 not upgraded.
```

```
!jupyter nbconvert --to PDF "/content/drive/MyDrive/Colab
Notebooks/C3_Capstone_Project.ipynb"
```

```
[NbConvertApp] Converting notebook /content/drive/MyDrive/Colab
Notebooks/C3_Capstone_Project.ipynb to PDF
[NbConvertApp] Support files will be in C3_Capstone_Project_files/
[NbConvertApp] Making directory ./C3_Capstone_Project_files
[NbConvertApp] Making directory ./C3_Capstone_Project_files
[NbConvertApp] Making directory ./C3_Capstone_Project_files
[NbConvertApp] Making directory ./C3_Capstone_Project_files
[NbConvertApp] Making directory ./C3_Capstone_Project_files
[NbConvertApp] Making directory ./C3_Capstone_Project_files
[NbConvertApp] Making directory ./C3_Capstone_Project_files
[NbConvertApp] Making directory ./C3_Capstone_Project_files
[NbConvertApp] Writing 127962 bytes to notebook.tex
[NbConvertApp] Building PDF
[NbConvertApp] Running xelatex 3 times: ['xelatex', 'notebook.tex', '-
quiet']
[NbConvertApp] CRITICAL | xelatex failed: ['xelatex', 'notebook.tex',
'-quiet']
This is XeTeX, Version 3.141592653-2.6-0.999993 (TeX Live
2022/dev/Debian) (preloaded format=xelatex)
 restricted \write18 enabled.
entering extended mode
(./notebook.tex
LaTeX2e <2021-11-15> patch level 1
L3 programming layer <2022-01-21>
(/usr/share/texlive/texmf-dist/tex/latex/base/article.cls
Document Class: article 2021/10/04 v1.4n Standard LaTeX document class
(/usr/share/texlive/texmf-dist/tex/latex/base/size11.clo))
(/usr/share/texlive/texmf-dist/tex/latex/tcolorbox/tcolorbox.sty
(/usr/share/texlive/texmf-dist/tex/latex/pgf/basiclayer/pgf.sty
(/usr/share/texlive/texmf-dist/tex/latex/pgf/utilities/pgfrcs.sty
(/usr/share/texlive/texmf-dist/tex/generic/pgf/utilities/pgfutil-
common.tex
(/usr/share/texlive/texmf-dist/tex/generic/pgf/utilities/pgfutil-
common-lists.t
ex)) (/usr/share/texlive/texmf-dist/tex/generic/pgf/utilities/pgfutil-
latex.def
)
(/usr/share/texlive/texmf-dist/tex/generic/pgf/utilities/pgfrcs.code.t
ex
(/usr/share/texlive/texmf-dist/tex/generic/pgf/pgf.revision.tex)))
(/usr/share/texlive/texmf-dist/tex/latex/pgf/basiclayer/pgfcore.sty
(/usr/share/texlive/texmf-dist/tex/latex/graphics/graphicx.sty
(/usr/share/texlive/texmf-dist/tex/latex/graphics/keyval.sty)
(/usr/share/texlive/texmf-dist/tex/latex/graphics/graphics.sty
```

```
(/usr/share/texlive/texmf-dist/tex/latex/graphics/trig.sty)
(/usr/share/texlive/texmf-dist/tex/latex/graphics-cfg/graphics.cfg)
(/usr/share/texlive/texmf-dist/tex/latex/graphics-def/xetex.def)))
(/usr/share/texlive/texmf-dist/tex/latex/pgf/systemlayer/pgfsys.sty
(/usr/share/texlive/texmf-dist/tex/generic/pgf/systemlayer/pgfsys.code
.tex
(/usr/share/texlive/texmf-dist/tex/generic/pgf/utilities/pgfkeys.code.
tex
(/usr/share/texlive/texmf-dist/tex/generic/pgf/utilities/pgfkeysfilter
ed.code.t
ex))
(/usr/share/texlive/texmf-dist/tex/generic/pgf/systemlayer/pgf.cfg)
(/usr/share/texlive/texmf-dist/tex/generic/pgf/systemlayer/pgfsys-
xetex.def
(/usr/share/texlive/texmf-dist/tex/generic/pgf/systemlayer/pgfsys-
dvipdfmx.def
(/usr/share/texlive/texmf-dist/tex/generic/pgf/systemlayer/pgfsys-
common-pdf.de
f))))
(/usr/share/texlive/texmf-dist/tex/generic/pgf/systemlayer/pgfsyssoftp
ath.code.
tex)
(/usr/share/texlive/texmf-dist/tex/generic/pgf/systemlayer/pgfsysproto
col.code.
tex)) (/usr/share/texlive/texmf-dist/tex/latex/xcolor/xcolor.sty
(/usr/share/texlive/texmf-dist/tex/latex/graphics-cfg/color.cfg))
(/usr/share/texlive/texmf-dist/tex/generic/pgf/basiclayer/pgfcore.code
.tex
(/usr/share/texlive/texmf-dist/tex/generic/pgf/math/pgfmath.code.tex
(/usr/share/texlive/texmf-dist/tex/generic/pgf/math/pgfmathcalc.code.t
ex
(/usr/share/texlive/texmf-dist/tex/generic/pgf/math/pgfmathutil.code.t
ex)
(/usr/share/texlive/texmf-dist/tex/generic/pgf/math/pgfmathparser.code
.tex)
(/usr/share/texlive/texmf-dist/tex/generic/pgf/math/pgfmathfunctions.c
ode.tex
(/usr/share/texlive/texmf-dist/tex/generic/pgf/math/pgfmathfunctions.b
asic.code
.tex)
(/usr/share/texlive/texmf-dist/tex/generic/pgf/math/pgfmathfunctions.t
rigonomet
ric.code.tex)
(/usr/share/texlive/texmf-dist/tex/generic/pgf/math/pgfmathfunctions.r
andom.cod
e.tex)
(/usr/share/texlive/texmf-dist/tex/generic/pgf/math/pgfmathfunctions.c
omparison
.code.tex)
```

```
(/usr/share/texlive/texmf-dist/tex/generic/pgf/math/pgfmathfunctions.b
ase.code.
tex)
(/usr/share/texlive/texmf-dist/tex/generic/pgf/math/pgfmathfunctions.r
ound.code
.tex)
(/usr/share/texlive/texmf-dist/tex/generic/pgf/math/pgfmathfunctions.m
isc.code.
tex)
(/usr/share/texlive/texmf-dist/tex/generic/pgf/math/pgfmathfunctions.i
ntegerari
thmetics.code.tex)))
(/usr/share/texlive/texmf-dist/tex/generic/pgf/math/pgfmathfloat.code.
tex))
(/usr/share/texlive/texmf-dist/tex/generic/pgf/math/pgfint.code.tex)
(/usr/share/texlive/texmf-dist/tex/generic/pgf/basiclayer/pgfcorepoint
s.code.te
x)
(/usr/share/texlive/texmf-dist/tex/generic/pgf/basiclayer/pgfcorepathc
onstruct.
code.tex)
(/usr/share/texlive/texmf-dist/tex/generic/pgf/basiclayer/pgfcorepathu
sage.code
.tex)
(/usr/share/texlive/texmf-dist/tex/generic/pgf/basiclayer/pgfcorescope
s.code.te
x)
(/usr/share/texlive/texmf-dist/tex/generic/pgf/basiclayer/pgfcoregraph
icstate.c
ode.tex)
(/usr/share/texlive/texmf-dist/tex/generic/pgf/basiclayer/pgfcoretrans
formation
s.code.tex)
(/usr/share/texlive/texmf-dist/tex/generic/pgf/basiclayer/pgfcorequick
.code.tex
)
(/usr/share/texlive/texmf-dist/tex/generic/pgf/basiclayer/pgfcoreobjec
ts.code.t
ex)
(/usr/share/texlive/texmf-dist/tex/generic/pgf/basiclayer/pgfcorepathp
rocessing
.code.tex)
(/usr/share/texlive/texmf-dist/tex/generic/pgf/basiclayer/pgfcorearrow
s.code.te
x)
(/usr/share/texlive/texmf-dist/tex/generic/pgf/basiclayer/pgfcoreshade
.code.tex
)
(/usr/share/texlive/texmf-dist/tex/generic/pgf/basiclayer/pgfcoreimage
```

```
.code.tex

(/usr/share/texlive/texmf-dist/tex/generic/pgf/basiclayer/pgfcoreexter
nal.code.
tex))
(/usr/share/texlive/texmf-dist/tex/generic/pgf/basiclayer/pgfcorelayer
s.code.te
x)
(/usr/share/texlive/texmf-dist/tex/generic/pgf/basiclayer/pgfcoretrans
parency.c
ode.tex)
(/usr/share/texlive/texmf-dist/tex/generic/pgf/basiclayer/pgfcorepatte
rns.code.
tex)
(/usr/share/texlive/texmf-dist/tex/generic/pgf/basiclayer/pgfcorerdf.c
ode.tex))
)
(/usr/share/texlive/texmf-dist/tex/generic/pgf/modules/pgfmoduleshapes
.code.tex
)
(/usr/share/texlive/texmf-dist/tex/generic/pgf/modules/pgfmoduleplot.c
ode.tex
)
(/usr/share/texlive/texmf-dist/tex/latex/pgf/compatibility/pgfcomp-
version-0-65
.sty)
(/usr/share/texlive/texmf-dist/tex/latex/pgf/compatibility/pgfcomp-
version-1-18
.sty)) (/usr/share/texlive/texmf-dist/tex/latex/tools/verbatim.sty)
(/usr/share/texlive/texmf-dist/tex/latex/environ/environ.sty
(/usr/share/texlive/texmf-dist/tex/latex/trimspaces/trimspaces.sty))
(/usr/share/texlive/texmf-dist/tex/latex/etoolbox/etoolbox.sty)
(/usr/share/texlive/texmf-dist/tex/latex/tcolorbox/tcbbreakable.code.t
ex
Library (tcolorbox): 'tcbbreakable.code.tex' version '5.0.2'
(/usr/share/texlive/texmf-dist/tex/generic/oberdiek/pdfcol.sty
(/usr/share/texlive/texmf-dist/tex/generic/ltxcmds/ltxcmds.sty)
(/usr/share/texlive/texmf-dist/tex/generic/infwarerr/infwarerr.sty)
(/usr/share/texlive/texmf-dist/tex/generic/iftex/iftex.sty))))
(/usr/share/texlive/texmf-dist/tex/latex/parskip/parskip.sty
(/usr/share/texlive/texmf-dist/tex/latex/kvoptions/kvoptions.sty
(/usr/share/texlive/texmf-dist/tex/generic/kvsetkeys/kvsetkeys.sty)))
(/usr/share/texlive/texmf-dist/tex/latex/caption/caption.sty
(/usr/share/texlive/texmf-dist/tex/latex/caption/caption3.sty))
(/usr/share/texlive/texmf-dist/tex/latex/float/float.sty)
(/usr/share/texlive/texmf-dist/tex/latex/tools/enumerate.sty)
(/usr/share/texlive/texmf-dist/tex/latex/geometry/geometry.sty
(/usr/share/texlive/texmf-dist/tex/generic/iftex/ifvtex.sty))
(/usr/share/texlive/texmf-dist/tex/latex/amsmath/amsmath.sty
```

```
For additional information on amsmath, use the `?' option.
(/usr/share/texlive/texmf-dist/tex/latex/amsmath/amstext.sty
(/usr/share/texlive/texmf-dist/tex/latex/amsmath/amsgen.sty))
(/usr/share/texlive/texmf-dist/tex/latex/amsmath/amsbsy.sty)
(/usr/share/texlive/texmf-dist/tex/latex/amsmath/amsopn.sty))
(/usr/share/texlive/texmf-dist/tex/latex/amsfonts/amssymb.sty
(/usr/share/texlive/texmf-dist/tex/latex/amsfonts/amsfonts.sty))
(/usr/share/texlive/texmf-dist/tex/latex/base/textcomp.sty)
(/usr/share/texlive/texmf-dist/tex/latex/upquote/upquote.sty)
(/usr/share/texlive/texmf-dist/tex/latex/eurosym/eurosym.sty)
(/usr/share/texlive/texmf-dist/tex/latex/fontspec/fontspec.sty
(/usr/share/texlive/texmf-dist/tex/latex/l3packages/xparse/xparse.sty
(/usr/share/texlive/texmf-dist/tex/latex/l3kernel/expl3.sty
(/usr/share/texlive/texmf-dist/tex/latex/l3backend/l3backend-xetex.def
(|extractbb --version))))
(/usr/share/texlive/texmf-dist/tex/latex/fontspec/fontspec-xetex.sty
(/usr/share/texlive/texmf-dist/tex/latex/base/fontenc.sty)
(/usr/share/texlive/texmf-dist/tex/latex/fontspec/fontspec.cfg)))
(/usr/share/texlive/texmf-dist/tex/latex/unicode-math/unicode-math.sty
(/usr/share/texlive/texmf-dist/tex/latex/unicode-math/unicode-math-
xetex.sty
(/usr/share/texlive/texmf-dist/tex/latex/l3packages/l3keys2e/l3keys2e.
sty)
(/usr/share/texlive/texmf-dist/tex/latex/base/fix-cm.sty
(/usr/share/texlive/texmf-dist/tex/latex/base/ts1enc.def))
(/usr/share/texlive/texmf-dist/tex/latex/unicode-math/unicode-math-
table.tex)))
(/usr/share/texlive/texmf-dist/tex/latex/fancyvrb/fancyvrb.sty)
(/usr/share/texlive/texmf-dist/tex/latex/grffile/grffile.sty)
(/usr/share/texlive/texmf-dist/tex/latex/adjustbox/adjustbox.sty
(/usr/share/texlive/texmf-dist/tex/latex/xkeyval/xkeyval.sty
(/usr/share/texlive/texmf-dist/tex/generic/xkeyval/xkeyval.tex
(/usr/share/texlive/texmf-dist/tex/generic/xkeyval/xkvutils.tex)))
(/usr/share/texlive/texmf-dist/tex/latex/adjustbox/adjcalc.sty)
(/usr/share/texlive/texmf-dist/tex/latex/adjustbox/trimclip.sty
(/usr/share/texlive/texmf-dist/tex/latex/collectbox/collectbox.sty)
(/usr/share/texlive/texmf-dist/tex/latex/adjustbox/tc-xetex.def))
(/usr/share/texlive/texmf-dist/tex/latex/ifoddpage/ifoddpage.sty)
(/usr/share/texlive/texmf-dist/tex/latex/varwidth/varwidth.sty))
(/usr/share/texlive/texmf-dist/tex/latex/hyperref/hyperref.sty
(/usr/share/texlive/texmf-dist/tex/generic/pdftexcmds/pdftexcmds.sty)
(/usr/share/texlive/texmf-dist/tex/generic/kvdefinekeys/kvdefinekeys.s
ty)
(/usr/share/texlive/texmf-dist/tex/generic/pdfescape/pdfescape.sty)
(/usr/share/texlive/texmf-dist/tex/latex/hycolor/hycolor.sty)
(/usr/share/texlive/texmf-dist/tex/latex/letltxmacro/letltxmacro.sty)
(/usr/share/texlive/texmf-dist/tex/latex/auxhook/auxhook.sty)
(/usr/share/texlive/texmf-dist/tex/latex/hyperref/pd1enc.def)
(/usr/share/texlive/texmf-dist/tex/latex/hyperref/hyperref-
```

```
langpatches.def)
(/usr/share/texlive/texmf-dist/tex/generic/intcalc/intcalc.sty)
(/usr/share/texlive/texmf-dist/tex/generic/etexcmds/etexcmds.sty)
(/usr/share/texlive/texmf-dist/tex/latex/hyperref/puenc.def)
(/usr/share/texlive/texmf-dist/tex/latex/url/url.sty)
(/usr/share/texlive/texmf-dist/tex/generic/bitset/bitset.sty
(/usr/share/texlive/texmf-dist/tex/generic/bigintcalc/bigintcalc.sty))
(/usr/share/texlive/texmf-dist/tex/latex/base/atbegshi-ltx.sty))
(/usr/share/texlive/texmf-dist/tex/latex/hyperref/hxetex.def
(/usr/share/texlive/texmf-dist/tex/generic/stringenc/stringenc.sty)
(/usr/share/texlive/texmf-dist/tex/latex/rerunfilecheck/rerunfilecheck
.sty
(/usr/share/texlive/texmf-dist/tex/latex/base/atveryend-ltx.sty)
(/usr/share/texlive/texmf-dist/tex/generic/uniquecounter/uniquecounter
.sty)))
(/usr/share/texlive/texmf-dist/tex/latex/titling/titling.sty)
(/usr/share/texlive/texmf-dist/tex/latex/tools/longtable.sty)
(/usr/share/texlive/texmf-dist/tex/latex/booktabs/booktabs.sty)
(/usr/share/texlive/texmf-dist/tex/latex/tools/array.sty)
(/usr/share/texlive/texmf-dist/tex/latex/tools/calc.sty)
(/usr/share/texlive/texmf-dist/tex/latex/enumitem/enumitem.sty)
(/usr/share/texlive/texmf-dist/tex/generic/ulem/ulem.sty)
(/usr/share/texlive/texmf-dist/tex/latex/jknapltx/mathrsfs.sty)
No file notebook.aux.
(/usr/share/texlive/texmf-dist/tex/latex/caption/ltcaption.sty)
*geometry* driver: auto-detecting
*geometry* detected driver: xetex
*geometry* verbose mode - [ preamble ] result:
* driver: xetex
* paper: <default>
* layout: <same size as paper>
* layoutoffset:(h,v)=(0.0pt,0.0pt)
* modes:
* h-part:(L,W,R)=(72.26999pt, 469.75502pt, 72.26999pt)
* v-part:(T,H,B)=(72.26999pt, 650.43001pt, 72.26999pt)
* \paperwidth=614.295pt
* \paperheight=794.96999pt
* \textwidth=469.75502pt
* \textheight=650.43001pt
* \oddsidemargin=0.0pt
* \evensidemargin=0.0pt
* \topmargin=-37.0pt
* \headheight=12.0pt
* \headsep=25.0pt
* \topskip=11.0pt
* \footskip=30.0pt
* \marginparwidth=59.0pt
* \marginparsep=10.0pt
* \columnsep=10.0pt
```

```
* \skip\footins=10.0pt plus 4.0pt minus 2.0pt
* \hoffset=0.0pt
* \voffset=0.0pt
* \mag=1000
* \@twocolumnfalse
* \@twosidefalse
* \@mparswitchfalse
* \@reversemarginfalse
* (1in=72.27pt=25.4mm, 1cm=28.453pt)

(/usr/share/texlive/texmf-dist/tex/latex/hyperref/nameref.sty
(/usr/share/texlive/texmf-dist/tex/latex/refcount/refcount.sty)
(/usr/share/texlive/texmf-dist/tex/generic/gettitlestring/gettitlestri
ng.sty))

Package hyperref Warning: Rerun to get /PageLabels entry.

(/usr/share/texlive/texmf-dist/tex/latex/amsfonts/umsa.fd)
(/usr/share/texlive/texmf-dist/tex/latex/amsfonts/umsb.fd)
(/usr/share/texlive/texmf-dist/tex/latex/jknapltx/ursfs.fd)

LaTeX Warning: No \author given.

[1]

LaTeX Warning: File `data/example_images.png' not found on input line
453.

! Unable to load picture or PDF file 'data/example_images.png'.
<to be read again>
                   }
l.453 \includegraphics{data/example_images.png}

?
! Emergency stop.
<to be read again>
                   }
l.453 \includegraphics{data/example_images.png}

Output written on notebook.pdf (1 page).
Transcript written on notebook.log.

Traceback (most recent call last):
  File "/usr/local/bin/jupyter-nbconvert", line 8, in <module>
    sys.exit(main())
  File
"/usr/local/lib/python3.10/dist-packages/jupyter_core/application.py",
line 285, in launch_instance
    return super().launch_instance(argv=argv, **kwargs)
  File
```

```
"/usr/local/lib/python3.10/dist-packages/traitlets/config/application.
py", line 992, in launch_instance
    app.start()
  File
"/usr/local/lib/python3.10/dist-packages/nbconvert/nbconvertapp.py",
line 423, in start
    self.convert_notebooks()
  File
"/usr/local/lib/python3.10/dist-packages/nbconvert/nbconvertapp.py",
line 597, in convert_notebooks
    self.convert_single_notebook(notebook_filename)
  File
"/usr/local/lib/python3.10/dist-packages/nbconvert/nbconvertapp.py",
line 560, in convert_single_notebook
    output, resources = self.export_single_notebook(
  File
"/usr/local/lib/python3.10/dist-packages/nbconvert/nbconvertapp.py",
line 488, in export_single_notebook
    output, resources = self.exporter.from_filename(
  File
"/usr/local/lib/python3.10/dist-packages/nbconvert/exporters/exporter.
py", line 189, in from_filename
    return self.from_file(f, resources=resources, **kw)
  File
"/usr/local/lib/python3.10/dist-packages/nbconvert/exporters/exporter.
py", line 206, in from_file
    return self.from_notebook_node(
  File
"/usr/local/lib/python3.10/dist-packages/nbconvert/exporters/pdf.py",
line 194, in from_notebook_node
    self.run_latex(tex_file)
  File
"/usr/local/lib/python3.10/dist-packages/nbconvert/exporters/pdf.py",
line 164, in run_latex
    return self.run_command(
  File
"/usr/local/lib/python3.10/dist-packages/nbconvert/exporters/pdf.py",
line 150, in run_command
    raise raise_on_failure(
nbconvert.exporters.pdf.LatexFailed: PDF creating failed, captured
latex output:
Failed to run "['xelatex', 'notebook.tex', '-quiet']" command:
This is XeTeX, Version 3.141592653-2.6-0.999993 (TeX Live
2022/dev/Debian) (preloaded format=xelatex)
 restricted \write18 enabled.
entering extended mode
(./notebook.tex
LaTeX2e <2021-11-15> patch level 1
L3 programming layer <2022-01-21>
```

```
(/usr/share/texlive/texmf-dist/tex/latex/base/article.cls
Document Class: article 2021/10/04 v1.4n Standard LaTeX document class
(/usr/share/texlive/texmf-dist/tex/latex/base/size11.clo))
(/usr/share/texlive/texmf-dist/tex/latex/tcolorbox/tcolorbox.sty
(/usr/share/texlive/texmf-dist/tex/latex/pgf/basiclayer/pgf.sty
(/usr/share/texlive/texmf-dist/tex/latex/pgf/utilities/pgfrcs.sty
(/usr/share/texlive/texmf-dist/tex/generic/pgf/utilities/pgfutil-
common.tex
(/usr/share/texlive/texmf-dist/tex/generic/pgf/utilities/pgfutil-
common-lists.t
ex)) (/usr/share/texlive/texmf-dist/tex/generic/pgf/utilities/pgfutil-
latex.def
)
(/usr/share/texlive/texmf-dist/tex/generic/pgf/utilities/pgfrcs.code.t
ex
(/usr/share/texlive/texmf-dist/tex/generic/pgf/pgf.revision.tex)))
(/usr/share/texlive/texmf-dist/tex/latex/pgf/basiclayer/pgfcore.sty
(/usr/share/texlive/texmf-dist/tex/latex/graphics/graphicx.sty
(/usr/share/texlive/texmf-dist/tex/latex/graphics/keyval.sty)
(/usr/share/texlive/texmf-dist/tex/latex/graphics/graphics.sty
(/usr/share/texlive/texmf-dist/tex/latex/graphics/trig.sty)
(/usr/share/texlive/texmf-dist/tex/latex/graphics-cfg/graphics.cfg)
(/usr/share/texlive/texmf-dist/tex/latex/graphics-def/xetex.def)))
(/usr/share/texlive/texmf-dist/tex/latex/pgf/systemlayer/pgfsys.sty
(/usr/share/texlive/texmf-dist/tex/generic/pgf/systemlayer/pgfsys.code
.tex
(/usr/share/texlive/texmf-dist/tex/generic/pgf/utilities/pgfkeys.code.
tex
(/usr/share/texlive/texmf-dist/tex/generic/pgf/utilities/pgfkeysfilter
ed.code.t
ex))
(/usr/share/texlive/texmf-dist/tex/generic/pgf/systemlayer/pgf.cfg)
(/usr/share/texlive/texmf-dist/tex/generic/pgf/systemlayer/pgfsys-
xetex.def
(/usr/share/texlive/texmf-dist/tex/generic/pgf/systemlayer/pgfsys-
dvipdfmx.def
(/usr/share/texlive/texmf-dist/tex/generic/pgf/systemlayer/pgfsys-
common-pdf.de
f))))
(/usr/share/texlive/texmf-dist/tex/generic/pgf/systemlayer/pgfsyssoftp
ath.code.
tex)
(/usr/share/texlive/texmf-dist/tex/generic/pgf/systemlayer/pgfsysproto
col.code.
tex)) (/usr/share/texlive/texmf-dist/tex/latex/xcolor/xcolor.sty
(/usr/share/texlive/texmf-dist/tex/latex/graphics-cfg/color.cfg))
(/usr/share/texlive/texmf-dist/tex/generic/pgf/basiclayer/pgfcore.code
.tex
(/usr/share/texlive/texmf-dist/tex/generic/pgf/math/pgfmath.code.tex
```

```
(/usr/share/texlive/texmf-dist/tex/generic/pgf/math/pgfmathcalc.code.t
ex
(/usr/share/texlive/texmf-dist/tex/generic/pgf/math/pgfmathutil.code.t
ex)
(/usr/share/texlive/texmf-dist/tex/generic/pgf/math/pgfmathparser.code
.tex)
(/usr/share/texlive/texmf-dist/tex/generic/pgf/math/pgfmathfunctions.c
ode.tex
(/usr/share/texlive/texmf-dist/tex/generic/pgf/math/pgfmathfunctions.b
asic.code
.tex)
(/usr/share/texlive/texmf-dist/tex/generic/pgf/math/pgfmathfunctions.t
rigonomet
ric.code.tex)
(/usr/share/texlive/texmf-dist/tex/generic/pgf/math/pgfmathfunctions.r
andom.cod
e.tex)
(/usr/share/texlive/texmf-dist/tex/generic/pgf/math/pgfmathfunctions.c
omparison
.code.tex)
(/usr/share/texlive/texmf-dist/tex/generic/pgf/math/pgfmathfunctions.b
ase.code.
tex)
(/usr/share/texlive/texmf-dist/tex/generic/pgf/math/pgfmathfunctions.r
ound.code
.tex)
(/usr/share/texlive/texmf-dist/tex/generic/pgf/math/pgfmathfunctions.m
isc.code.
tex)
(/usr/share/texlive/texmf-dist/tex/generic/pgf/math/pgfmathfunctions.i
ntegerari
thmetics.code.tex)))
(/usr/share/texlive/texmf-dist/tex/generic/pgf/math/pgfmathfloat.code.
tex))
(/usr/share/texlive/texmf-dist/tex/generic/pgf/math/pgfint.code.tex)
(/usr/share/texlive/texmf-dist/tex/generic/pgf/basiclayer/pgfcorepoint
s.code.te
x)
(/usr/share/texlive/texmf-dist/tex/generic/pgf/basiclayer/pgfcorepathc
onstruct.
code.tex)
(/usr/share/texlive/texmf-dist/tex/generic/pgf/basiclayer/pgfcorepathu
sage.code
.tex)
(/usr/share/texlive/texmf-dist/tex/generic/pgf/basiclayer/pgfcorescope
s.code.te
x)
(/usr/share/texlive/texmf-dist/tex/generic/pgf/basiclayer/pgfcoregraph
icstate.c
```

```
ode.tex)
(/usr/share/texlive/texmf-dist/tex/generic/pgf/basiclayer/pgfcoretrans
formation
s.code.tex)
(/usr/share/texlive/texmf-dist/tex/generic/pgf/basiclayer/pgfcorequick
.code.tex
)
(/usr/share/texlive/texmf-dist/tex/generic/pgf/basiclayer/pgfcoreobjec
ts.code.t
ex)
(/usr/share/texlive/texmf-dist/tex/generic/pgf/basiclayer/pgfcorepathp
rocessing
.code.tex)
(/usr/share/texlive/texmf-dist/tex/generic/pgf/basiclayer/pgfcorearrow
s.code.te
x)
(/usr/share/texlive/texmf-dist/tex/generic/pgf/basiclayer/pgfcoreshade
.code.tex
)
(/usr/share/texlive/texmf-dist/tex/generic/pgf/basiclayer/pgfcoreimage
.code.tex

(/usr/share/texlive/texmf-dist/tex/generic/pgf/basiclayer/pgfcoreexter
nal.code.
tex))
(/usr/share/texlive/texmf-dist/tex/generic/pgf/basiclayer/pgfcorelayer
s.code.te
x)
(/usr/share/texlive/texmf-dist/tex/generic/pgf/basiclayer/pgfcoretrans
parency.c
ode.tex)
(/usr/share/texlive/texmf-dist/tex/generic/pgf/basiclayer/pgfcorepatte
rns.code.
tex)
(/usr/share/texlive/texmf-dist/tex/generic/pgf/basiclayer/pgfcorerdf.c
ode.tex))
)
(/usr/share/texlive/texmf-dist/tex/generic/pgf/modules/pgfmoduleshapes
.code.tex
)
(/usr/share/texlive/texmf-dist/tex/generic/pgf/modules/pgfmoduleplot.c
ode.tex
)
(/usr/share/texlive/texmf-dist/tex/latex/pgf/compatibility/pgfcomp-
version-0-65
.sty)
(/usr/share/texlive/texmf-dist/tex/latex/pgf/compatibility/pgfcomp-
version-1-18
.sty)) (/usr/share/texlive/texmf-dist/tex/latex/tools/verbatim.sty)
(/usr/share/texlive/texmf-dist/tex/latex/environ/environ.sty
```

```
(/usr/share/texlive/texmf-dist/tex/latex/trimspaces/trimspaces.sty))
(/usr/share/texlive/texmf-dist/tex/latex/etoolbox/etoolbox.sty)
(/usr/share/texlive/texmf-dist/tex/latex/tcolorbox/tcbbreakable.code.t
ex
Library (tcolorbox): 'tcbbreakable.code.tex' version '5.0.2'
(/usr/share/texlive/texmf-dist/tex/generic/oberdiek/pdfcol.sty
(/usr/share/texlive/texmf-dist/tex/generic/ltxcmds/ltxcmds.sty)
(/usr/share/texlive/texmf-dist/tex/generic/infwarerr/infwarerr.sty)
(/usr/share/texlive/texmf-dist/tex/generic/iftex/iftex.sty))))
(/usr/share/texlive/texmf-dist/tex/latex/parskip/parskip.sty
(/usr/share/texlive/texmf-dist/tex/latex/kvoptions/kvoptions.sty
(/usr/share/texlive/texmf-dist/tex/generic/kvsetkeys/kvsetkeys.sty)))
(/usr/share/texlive/texmf-dist/tex/latex/caption/caption.sty
(/usr/share/texlive/texmf-dist/tex/latex/caption/caption3.sty))
(/usr/share/texlive/texmf-dist/tex/latex/float/float.sty)
(/usr/share/texlive/texmf-dist/tex/latex/tools/enumerate.sty)
(/usr/share/texlive/texmf-dist/tex/latex/geometry/geometry.sty
(/usr/share/texlive/texmf-dist/tex/generic/iftex/ifvtex.sty))
(/usr/share/texlive/texmf-dist/tex/latex/amsmath/amsmath.sty
For additional information on amsmath, use the `?' option.
(/usr/share/texlive/texmf-dist/tex/latex/amsmath/amstext.sty
(/usr/share/texlive/texmf-dist/tex/latex/amsmath/amsgen.sty))
(/usr/share/texlive/texmf-dist/tex/latex/amsmath/amsbsy.sty)
(/usr/share/texlive/texmf-dist/tex/latex/amsmath/amsopn.sty))
(/usr/share/texlive/texmf-dist/tex/latex/amsfonts/amssymb.sty
(/usr/share/texlive/texmf-dist/tex/latex/amsfonts/amsfonts.sty))
(/usr/share/texlive/texmf-dist/tex/latex/base/textcomp.sty)
(/usr/share/texlive/texmf-dist/tex/latex/upquote/upquote.sty)
(/usr/share/texlive/texmf-dist/tex/latex/eurosym/eurosym.sty)
(/usr/share/texlive/texmf-dist/tex/latex/fontspec/fontspec.sty
(/usr/share/texlive/texmf-dist/tex/latex/l3packages/xparse/xparse.sty
(/usr/share/texlive/texmf-dist/tex/latex/l3kernel/expl3.sty
(/usr/share/texlive/texmf-dist/tex/latex/l3backend/l3backend-xetex.def
(|extractbb --version))))
(/usr/share/texlive/texmf-dist/tex/latex/fontspec/fontspec-xetex.sty
(/usr/share/texlive/texmf-dist/tex/latex/base/fontenc.sty)
(/usr/share/texlive/texmf-dist/tex/latex/fontspec/fontspec.cfg)))
(/usr/share/texlive/texmf-dist/tex/latex/unicode-math/unicode-math.sty
(/usr/share/texlive/texmf-dist/tex/latex/unicode-math/unicode-math-
xetex.sty
(/usr/share/texlive/texmf-dist/tex/latex/l3packages/l3keys2e/l3keys2e.
sty)
(/usr/share/texlive/texmf-dist/tex/latex/base/fix-cm.sty
(/usr/share/texlive/texmf-dist/tex/latex/base/ts1enc.def))
(/usr/share/texlive/texmf-dist/tex/latex/unicode-math/unicode-math-
table.tex)))
(/usr/share/texlive/texmf-dist/tex/latex/fancyvrb/fancyvrb.sty)
(/usr/share/texlive/texmf-dist/tex/latex/grffile/grffile.sty)
(/usr/share/texlive/texmf-dist/tex/latex/adjustbox/adjustbox.sty
```

```
(/usr/share/texlive/texmf-dist/tex/latex/xkeyval/xkeyval.sty
(/usr/share/texlive/texmf-dist/tex/generic/xkeyval/xkeyval.tex
(/usr/share/texlive/texmf-dist/tex/generic/xkeyval/xkvutils.tex)))
(/usr/share/texlive/texmf-dist/tex/latex/adjustbox/adjcalc.sty)
(/usr/share/texlive/texmf-dist/tex/latex/adjustbox/trimclip.sty
(/usr/share/texlive/texmf-dist/tex/latex/collectbox/collectbox.sty)
(/usr/share/texlive/texmf-dist/tex/latex/adjustbox/tc-xetex.def))
(/usr/share/texlive/texmf-dist/tex/latex/ifoddpage/ifoddpage.sty)
(/usr/share/texlive/texmf-dist/tex/latex/varwidth/varwidth.sty))
(/usr/share/texlive/texmf-dist/tex/latex/hyperref/hyperref.sty
(/usr/share/texlive/texmf-dist/tex/generic/pdftexcmds/pdftexcmds.sty)
(/usr/share/texlive/texmf-dist/tex/generic/kvdefinekeys/kvdefinekeys.s
ty)
(/usr/share/texlive/texmf-dist/tex/generic/pdfescape/pdfescape.sty)
(/usr/share/texlive/texmf-dist/tex/latex/hycolor/hycolor.sty)
(/usr/share/texlive/texmf-dist/tex/latex/letltxmacro/letltxmacro.sty)
(/usr/share/texlive/texmf-dist/tex/latex/auxhook/auxhook.sty)
(/usr/share/texlive/texmf-dist/tex/latex/hyperref/pd1enc.def)
(/usr/share/texlive/texmf-dist/tex/latex/hyperref/hyperref-
langpatches.def)
(/usr/share/texlive/texmf-dist/tex/generic/intcalc/intcalc.sty)
(/usr/share/texlive/texmf-dist/tex/generic/etexcmds/etexcmds.sty)
(/usr/share/texlive/texmf-dist/tex/latex/hyperref/puenc.def)
(/usr/share/texlive/texmf-dist/tex/latex/url/url.sty)
(/usr/share/texlive/texmf-dist/tex/generic/bitset/bitset.sty
(/usr/share/texlive/texmf-dist/tex/generic/bigintcalc/bigintcalc.sty))
(/usr/share/texlive/texmf-dist/tex/latex/base/atbegshi-ltx.sty))
(/usr/share/texlive/texmf-dist/tex/latex/hyperref/hxetex.def
(/usr/share/texlive/texmf-dist/tex/generic/stringenc/stringenc.sty)
(/usr/share/texlive/texmf-dist/tex/latex/rerunfilecheck/rerunfilecheck
.sty
(/usr/share/texlive/texmf-dist/tex/latex/base/atveryend-ltx.sty)
(/usr/share/texlive/texmf-dist/tex/generic/uniquecounter/uniquecounter
.sty)))
(/usr/share/texlive/texmf-dist/tex/latex/titling/titling.sty)
(/usr/share/texlive/texmf-dist/tex/latex/tools/longtable.sty)
(/usr/share/texlive/texmf-dist/tex/latex/booktabs/booktabs.sty)
(/usr/share/texlive/texmf-dist/tex/latex/tools/array.sty)
(/usr/share/texlive/texmf-dist/tex/latex/tools/calc.sty)
(/usr/share/texlive/texmf-dist/tex/latex/enumitem/enumitem.sty)
(/usr/share/texlive/texmf-dist/tex/generic/ulem/ulem.sty)
(/usr/share/texlive/texmf-dist/tex/latex/jknapltx/mathrsfs.sty)
No file notebook.aux.
(/usr/share/texlive/texmf-dist/tex/latex/caption/ltcaption.sty)
*geometry* driver: auto-detecting
*geometry* detected driver: xetex
*geometry* verbose mode - [ preamble ] result:
* driver: xetex
* paper: <default>
```

```
* layout: <same size as paper>
* layoutoffset:(h,v)=(0.0pt,0.0pt)
* modes:
* h-part:(L,W,R)=(72.26999pt, 469.75502pt, 72.26999pt)
* v-part:(T,H,B)=(72.26999pt, 650.43001pt, 72.26999pt)
* \paperwidth=614.295pt
* \paperheight=794.96999pt
* \textwidth=469.75502pt
* \textheight=650.43001pt
* \oddsidemargin=0.0pt
* \evensidemargin=0.0pt
* \topmargin=-37.0pt
* \headheight=12.0pt
* \headsep=25.0pt
* \topskip=11.0pt
* \footskip=30.0pt
* \marginparwidth=59.0pt
* \marginparsep=10.0pt
* \columnsep=10.0pt
* \skip\footins=10.0pt plus 4.0pt minus 2.0pt
* \hoffset=0.0pt
* \voffset=0.0pt
* \mag=1000
* \@twocolumnfalse
* \@twosidefalse
* \@mparswitchfalse
* \@reversemarginfalse
* (1in=72.27pt=25.4mm, 1cm=28.453pt)

(/usr/share/texlive/texmf-dist/tex/latex/hyperref/nameref.sty
(/usr/share/texlive/texmf-dist/tex/latex/refcount/refcount.sty)
(/usr/share/texlive/texmf-dist/tex/generic/gettitlestring/gettitlestri
ng.sty))

Package hyperref Warning: Rerun to get /PageLabels entry.

(/usr/share/texlive/texmf-dist/tex/latex/amsfonts/umsa.fd)
(/usr/share/texlive/texmf-dist/tex/latex/amsfonts/umsb.fd)
(/usr/share/texlive/texmf-dist/tex/latex/jknapltx/ursfs.fd)

LaTeX Warning: No \author given.

[1]

LaTeX Warning: File `data/example_images.png' not found on input line
453.

! Unable to load picture or PDF file 'data/example_images.png'.
<to be read again>
                   }
```

```
l.453 \includegraphics{data/example_images.png}

?
! Emergency stop.
<to be read again>
                   }
l.453 \includegraphics{data/example_images.png}

Output written on notebook.pdf (1 page).
Transcript written on notebook.log.
```