VAE_model

February 19, 2023

[ ]:
```
# Christine Orosco
# Develop a variational autoencoder using the MNIST data set
# and save a grid of 15 x 15 digits to the results/vae directory
```

0.1 Build the Model

[1]:
```
from tensorflow import keras
from tensorflow.keras import layers

latent_dim = 2

encoder_inputs = keras.Input(shape=(28, 28, 1))
x = layers.Conv2D(32, 3, activation="relu", strides=2,␣
↪padding="same")(encoder_inputs)
x = layers.Conv2D(64, 3, activation="relu", strides=2, padding="same")(x)
x = layers.Flatten()(x)
x = layers.Dense(16, activation="relu")(x)
z_mean = layers.Dense(latent_dim, name="z_mean")(x)
z_log_var = layers.Dense(latent_dim, name="z_log_var")(x)
encoder = keras.Model(encoder_inputs, [z_mean, z_log_var], name="encoder")
```

[2]:
```
encoder.summary()
```

```
Model: "encoder"

________________________________________________________________________________
__________________
Layer (type)                    Output Shape         Param #     Connected to
================================================================================
==================
input_1 (InputLayer)            [(None, 28, 28, 1)]  0

________________________________________________________________________________
__________________
conv2d (Conv2D)                 (None, 14, 14, 32)   320         input_1[0][0]

________________________________________________________________________________
__________________
conv2d_1 (Conv2D)               (None, 7, 7, 64)     18496       conv2d[0][0]

________________________________________________________________________________
```

```
------------------
flatten (Flatten)                (None, 3136)         0           conv2d_1[0][0]

--------------------------------------------------------------------------------
------------------
dense (Dense)                    (None, 16)           50192       flatten[0][0]

--------------------------------------------------------------------------------
------------------
z_mean (Dense)                   (None, 2)            34          dense[0][0]

--------------------------------------------------------------------------------
------------------
z_log_var (Dense)                (None, 2)            34          dense[0][0]
================================================================================
==================
Total params: 69,076
Trainable params: 69,076
Non-trainable params: 0

--------------------------------------------------------------------------------
------------------
```

[3]:
```python
import tensorflow as tf

class Sampler(layers.Layer):
    def call(self, z_mean, z_log_var):
        batch_size = tf.shape(z_mean)[0]
        z_size = tf.shape(z_mean)[1]
        epsilon = tf.random.normal(shape=(batch_size, z_size))
        return z_mean + tf.exp(0.5 * z_log_var) * epsilon
```

[ ]:
```python
### Define the layers and decoder
```

[4]:
```python
latent_inputs = keras.Input(shape=(latent_dim,))
x = layers.Dense(7 * 7 * 64, activation="relu")(latent_inputs)
x = layers.Reshape((7, 7, 64))(x)
x = layers.Conv2DTranspose(64, 3, activation="relu", strides=2,␣
 ↪padding="same")(x)
x = layers.Conv2DTranspose(32, 3, activation="relu", strides=2,␣
 ↪padding="same")(x)
decoder_outputs = layers.Conv2D(1, 3, activation="sigmoid", padding="same")(x)
decoder = keras.Model(latent_inputs, decoder_outputs, name="decoder")
```

[5]:
```python
decoder.summary()
```

```
Model: "decoder"

_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_2 (InputLayer)         [(None, 2)]               0
```

```
_________________________________________________________________
dense_1 (Dense)              (None, 3136)              9408

_________________________________________________________________
reshape (Reshape)            (None, 7, 7, 64)          0

_________________________________________________________________
conv2d_transpose (Conv2DTran (None, 14, 14, 64)        36928

_________________________________________________________________
conv2d_transpose_1 (Conv2DTr (None, 28, 28, 32)        18464

_________________________________________________________________
conv2d_2 (Conv2D)            (None, 28, 28, 1)         289
=================================================================
Total params: 65,089
Trainable params: 65,089
Non-trainable params: 0

_________________________________________________________________
```

[9]:
```
from keras.datasets import mnist
```

0.1.1 Train and fit the model

[6]:
```
class VAE(keras.Model):
    def __init__(self, encoder, decoder, **kwargs):
        super().__init__(**kwargs)
        self.encoder = encoder
        self.decoder = decoder
        self.sampler = Sampler()
        self.total_loss_tracker = keras.metrics.Mean(name="total_loss")
        self.reconstruction_loss_tracker = keras.metrics.Mean(
            name="reconstruction_loss")
        self.kl_loss_tracker = keras.metrics.Mean(name="kl_loss")

    @property
    def metrics(self):
        return [self.total_loss_tracker,
                self.reconstruction_loss_tracker,
                self.kl_loss_tracker]

    def train_step(self, data):
        with tf.GradientTape() as tape:
            z_mean, z_log_var = self.encoder(data)
            z = self.sampler(z_mean, z_log_var)
            reconstruction = decoder(z)
            reconstruction_loss = tf.reduce_mean(
                tf.reduce_sum(
                    keras.losses.binary_crossentropy(data, reconstruction),
                    axis=(1, 2)
                )
```

```
            )
            kl_loss = -0.5 * (1 + z_log_var - tf.square(z_mean) - tf.
↪exp(z_log_var))
            total_loss = reconstruction_loss + tf.reduce_mean(kl_loss)
        grads = tape.gradient(total_loss, self.trainable_weights)
        self.optimizer.apply_gradients(zip(grads, self.trainable_weights))
        self.total_loss_tracker.update_state(total_loss)
        self.reconstruction_loss_tracker.update_state(reconstruction_loss)
        self.kl_loss_tracker.update_state(kl_loss)
        return {
            "total_loss": self.total_loss_tracker.result(),
            "reconstruction_loss": self.reconstruction_loss_tracker.result(),
            "kl_loss": self.kl_loss_tracker.result(),
        }
```

[7]:
```
import numpy as np

(x_train, _), (x_test, _) = keras.datasets.mnist.load_data()
mnist_digits = np.concatenate([x_train, x_test], axis=0)
mnist_digits = np.expand_dims(mnist_digits, -1).astype("float32") / 255

vae = VAE(encoder, decoder)
vae.compile(optimizer=keras.optimizers.Adam(), run_eagerly=True)
vae.fit(mnist_digits, epochs=30, batch_size=128)
```

```
Epoch 1/30
547/547 [==============================] - 61s 112ms/step - total_loss: 948.2521
- reconstruction_loss: 942.6980 - kl_loss: 5.5530
Epoch 2/30
547/547 [==============================] - 60s 110ms/step - total_loss: 769.0019
- reconstruction_loss: 760.3369 - kl_loss: 8.6654
Epoch 3/30
547/547 [==============================] - 60s 110ms/step - total_loss: 730.3389
- reconstruction_loss: 723.6174 - kl_loss: 6.7220
Epoch 4/30
547/547 [==============================] - 60s 109ms/step - total_loss: 710.4087
- reconstruction_loss: 703.9483 - kl_loss: 6.4601
Epoch 5/30
547/547 [==============================] - 60s 110ms/step - total_loss: 698.8322
- reconstruction_loss: 692.5614 - kl_loss: 6.2710
Epoch 6/30
547/547 [==============================] - 60s 110ms/step - total_loss: 690.8210
- reconstruction_loss: 684.6925 - kl_loss: 6.1289
Epoch 7/30
547/547 [==============================] - 60s 110ms/step - total_loss: 685.2900
- reconstruction_loss: 679.2664 - kl_loss: 6.0235
Epoch 8/30
```

```
547/547 [==============================] - 61s 111ms/step - total_loss: 681.1357
- reconstruction_loss: 675.1987 - kl_loss: 5.9366
Epoch 9/30
547/547 [==============================] - 61s 112ms/step - total_loss: 676.6614
- reconstruction_loss: 670.8033 - kl_loss: 5.8582
Epoch 10/30
547/547 [==============================] - 61s 111ms/step - total_loss: 674.5480
- reconstruction_loss: 668.7725 - kl_loss: 5.7762
Epoch 11/30
547/547 [==============================] - 61s 111ms/step - total_loss: 671.4193
- reconstruction_loss: 665.6986 - kl_loss: 5.7199
Epoch 12/30
547/547 [==============================] - 61s 111ms/step - total_loss: 669.2781
- reconstruction_loss: 663.6031 - kl_loss: 5.6747
Epoch 13/30
547/547 [==============================] - 61s 111ms/step - total_loss: 667.9094
- reconstruction_loss: 662.3022 - kl_loss: 5.6077
Epoch 14/30
547/547 [==============================] - 61s 111ms/step - total_loss: 665.5259
- reconstruction_loss: 659.9603 - kl_loss: 5.5652
Epoch 15/30
547/547 [==============================] - 61s 111ms/step - total_loss: 663.8563
- reconstruction_loss: 658.3548 - kl_loss: 5.5015
Epoch 16/30
547/547 [==============================] - 61s 111ms/step - total_loss: 662.6117
- reconstruction_loss: 657.1343 - kl_loss: 5.4773
Epoch 17/30
547/547 [==============================] - 60s 110ms/step - total_loss: 661.1255
- reconstruction_loss: 655.6710 - kl_loss: 5.4547
Epoch 18/30
547/547 [==============================] - 61s 111ms/step - total_loss: 659.5481
- reconstruction_loss: 654.1368 - kl_loss: 5.4116
Epoch 19/30
547/547 [==============================] - 60s 110ms/step - total_loss: 659.1514
- reconstruction_loss: 653.7558 - kl_loss: 5.3962
Epoch 20/30
547/547 [==============================] - 60s 110ms/step - total_loss: 657.6087
- reconstruction_loss: 652.2276 - kl_loss: 5.3810
Epoch 21/30
547/547 [==============================] - 60s 110ms/step - total_loss: 656.5546
- reconstruction_loss: 651.2164 - kl_loss: 5.3380
Epoch 22/30
547/547 [==============================] - 61s 111ms/step - total_loss: 655.6756
- reconstruction_loss: 650.3491 - kl_loss: 5.3274
Epoch 23/30
547/547 [==============================] - 61s 111ms/step - total_loss: 654.9643
- reconstruction_loss: 649.6589 - kl_loss: 5.3060
Epoch 24/30
```

```
547/547 [==============================] - 61s 111ms/step - total_loss: 654.1458
- reconstruction_loss: 648.8718 - kl_loss: 5.2747
Epoch 25/30
547/547 [==============================] - 60s 110ms/step - total_loss: 653.3022
- reconstruction_loss: 648.0149 - kl_loss: 5.2871
Epoch 26/30
547/547 [==============================] - 60s 110ms/step - total_loss: 652.0563
- reconstruction_loss: 646.7878 - kl_loss: 5.2691
Epoch 27/30
547/547 [==============================] - 60s 110ms/step - total_loss: 651.5031
- reconstruction_loss: 646.2761 - kl_loss: 5.2269
Epoch 28/30
547/547 [==============================] - 60s 110ms/step - total_loss: 651.2033
- reconstruction_loss: 645.9885 - kl_loss: 5.2151
Epoch 29/30
547/547 [==============================] - 60s 110ms/step - total_loss: 649.8629
- reconstruction_loss: 644.6339 - kl_loss: 5.2288
Epoch 30/30
547/547 [==============================] - 60s 111ms/step - total_loss: 649.5643
- reconstruction_loss: 644.3535 - kl_loss: 5.2111
```

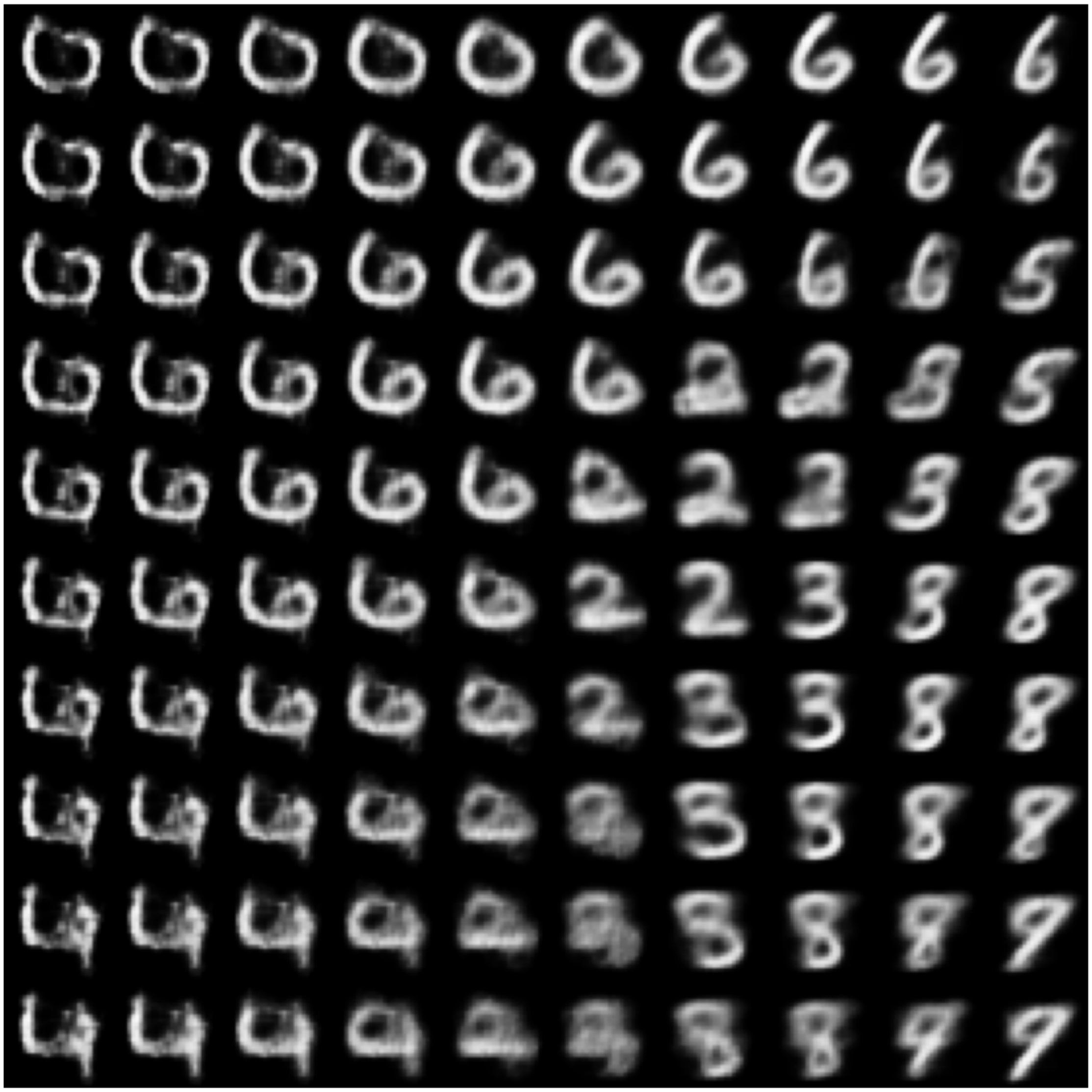
[7]:
```
<tensorflow.python.keras.callbacks.History at 0x7fc76c9fbb50>
```

0.1.2 Display the model output

[14]:
```python
import matplotlib.pyplot as plt

n = 10
digit_size = 28
figure = np.zeros((digit_size * n, digit_size * n))

grid_x = np.linspace(-1, 1, n)
grid_y = np.linspace(-1, 1, n)[::-1]

for i, yi in enumerate(grid_y):
    for j, xi in enumerate(grid_x):
        z_sample = np.array([[xi, yi]])
        x_decoded = vae.decoder.predict(z_sample)
        digit = x_decoded[0].reshape(digit_size, digit_size)
        figure[
            i * digit_size : (i + 1) * digit_size,
            j * digit_size : (j + 1) * digit_size,
        ] = digit

plt.figure(figsize=(15, 15))
start_range = digit_size // 2
end_range = n * digit_size + start_range
```

```python
pixel_range = np.arange(start_range, end_range, digit_size)
sample_range_x = np.round(grid_x, 1)
sample_range_y = np.round(grid_y, 1)
plt.xticks(pixel_range, sample_range_x)
plt.yticks(pixel_range, sample_range_y)
plt.xlabel("z[0]")
plt.ylabel("z[1]")
plt.axis("off")
plt.imshow(figure, cmap="Greys_r")
```

[14]: <matplotlib.image.AxesImage at 0x7fc7041491c0>



```python
[ ]:
```

```
[ ]:
```