Assignment 12

DSC650

Taniya Adhikari

VAE encoder network

In [1]:

```
import keras
from keras import layers
#from keras import backend as K
from keras.models import Model
import numpy as np

import tensorflow.compat.v1.keras.backend as K
import tensorflow as tf
tf.compat.v1.disable_eager_execution()
```

```
Using TensorFlow backend.
```

In [3]:

```
import os
from pathlib import Path

current_dir = Path(os.getcwd()).absolute()
results_dir = current_dir.joinpath('results')
results_dir.mkdir(parents=True, exist_ok=True)
results_dir
```

Out[3]:

```
WindowsPath('C:/Users/bibek/Documents/GitHub/dsc650/dsc650/assignments/assign
ment12/results')
```

In [4]:

```
img_shape = (28, 28, 1)
batch_size = 16
latent_dim = 2

input_img = keras.Input(shape=img_shape)

x = layers.Conv2D(32, 3,
                  padding='same', activation='relu')(input_img)
x = layers.Conv2D(64, 3,
                  padding='same', activation='relu',
                  strides=(2, 2))(x)
x = layers.Conv2D(64, 3,
                  padding='same', activation='relu')(x)
x = layers.Conv2D(64, 3,
                  padding='same', activation='relu')(x)
shape_before_flattening = K.int_shape(x)

x = layers.Flatten()(x)
x = layers.Dense(32, activation='relu')(x)

z_mean = layers.Dense(latent_dim)(x)
z_log_var = layers.Dense(latent_dim)(x)
```

```
WARNING:tensorflow:From C:\Users\bibek\anaconda3\envs\dsc650\lib\site-package
s\tensorflow_core\python\ops\resource_variable_ops.py:1635: calling BaseResou
rceVariable.__init__ (from tensorflow.python.ops.resource_variable_ops) with
constraint is deprecated and will be removed in a future version.
Instructions for updating:
If using Keras pass *_constraint arguments to layers.
```

Latent-space-sampling function

In [5]:

```
def sampling(args):
    z_mean, z_log_var = args
    epsilon = K.random_normal(shape=(K.shape(z_mean)[0], latent_dim),
                              mean=0., stddev=1.)
    return z_mean + K.exp(z_log_var) * epsilon

z = layers.Lambda(sampling)([z_mean, z_log_var])
```

VAE decoder network, mapping latent space points to images

In [6]:

```
decoder_input = layers.Input(K.int_shape(z)[1:])

x = layers.Dense(np.prod(shape_before_flattening[1:]),
                 activation='relu')(decoder_input)

x = layers.Reshape(shape_before_flattening[1:])(x)

x = layers.Conv2DTranspose(32, 3,
                           padding='same',
                           activation='relu',
                           strides=(2, 2))(x)

x = layers.Conv2D(1, 3,
                  padding='same',
                  activation='sigmoid')(x)

decoder = Model(decoder_input, x)

z_decoded = decoder(z)
```

Custom layer used to compute the VAE loss

In [7]:

```
class CustomVariationalLayer(keras.layers.Layer):

    def vae_loss(self, x, z_decoded):
        x = K.flatten(x)
        z_decoded = K.flatten(z_decoded)
        xent_loss = keras.metrics.binary_crossentropy(x, z_decoded)
        kl_loss = -5e-4 * K.mean(
            1 + z_log_var - K.square(z_mean) - K.exp(z_log_var), axis=-1)
        return K.mean(xent_loss + kl_loss)

    def call(self, inputs):
        x = inputs[0]
        z_decoded = inputs[1]
        loss = self.vae_loss(x, z_decoded)
        self.add_loss(loss, inputs=inputs)
        return x

y = CustomVariationalLayer()([input_img, z_decoded])
```

Training the VAE

In [8]:

```
from keras.datasets import mnist

vae = Model(input_img, y)
vae.compile(optimizer='rmsprop', loss=None)
vae.summary()

(x_train, _), (x_test, y_test) = mnist.load_data()

x_train = x_train.astype('float32') / 255.
x_train = x_train.reshape(x_train.shape + (1,))
x_test = x_test.astype('float32') / 255.
x_test = x_test.reshape(x_test.shape + (1,))

vae.fit(x=x_train, y=None,
        shuffle=True,
        epochs=10,
        batch_size=batch_size,
        validation_data=(x_test, None))
```

```
C:\Users\bibek\anaconda3\envs\dsc650\lib\site-packages\keras\engine\training_
utils.py:819: UserWarning: Output custom_variational_layer_1 missing from los
s dictionary. We assume this was done on purpose. The fit and evaluate APIs w
ill not be expecting any data to be passed to custom_variational_layer_1.
  'be expecting any data to be passed to {0}.'.format(name))
```

```
Model: "model_2"
_____________________________________________________________________________
_____________________
Layer (type)                    Output Shape         Param #     Connected to
=============================================================================
=====================
input_1 (InputLayer)            (None, 28, 28, 1)    0

_____________________________________________________________________________
_____________________
conv2d_1 (Conv2D)               (None, 28, 28, 32)   320         input_1[0]
[0]

_____________________________________________________________________________
_____________________
conv2d_2 (Conv2D)               (None, 14, 14, 64)   18496       conv2d_1[0]
[0]

_____________________________________________________________________________
_____________________
conv2d_3 (Conv2D)               (None, 14, 14, 64)   36928       conv2d_2[0]
[0]

_____________________________________________________________________________
_____________________
conv2d_4 (Conv2D)               (None, 14, 14, 64)   36928       conv2d_3[0]
[0]

_____________________________________________________________________________
_____________________
flatten_1 (Flatten)             (None, 12544)        0           conv2d_4[0]
[0]

_____________________________________________________________________________
_____________________
dense_1 (Dense)                 (None, 32)           401440      flatten_1[0]
[0]

_____________________________________________________________________________
_____________________
dense_2 (Dense)                 (None, 2)            66          dense_1[0]
[0]

_____________________________________________________________________________
_____________________
dense_3 (Dense)                 (None, 2)            66          dense_1[0]
[0]

_____________________________________________________________________________
_____________________
lambda_1 (Lambda)               (None, 2)            0           dense_2[0]
[0]
                                                                 dense_3[0]
[0]

_____________________________________________________________________________
_____________________
model_1 (Model)                 (None, 28, 28, 1)    56385       lambda_1[0]
[0]

_____________________________________________________________________________
_____________________
custom_variational_layer_1 (Cus [(None, 28, 28, 1),  0           input_1[0]
[0]
                                                                 model_1[1]
[0]
=============================================================================
=====================
Total params: 550,629
Trainable params: 550,629
Non-trainable params: 0
_____________________________________________________________________________
_____________________
Train on 60000 samples, validate on 10000 samples
Epoch 1/10
60000/60000 [==============================] - 167s 3ms/step - loss: 0.2127 -
val_loss: 0.1968
Epoch 2/10
60000/60000 [==============================] - 170s 3ms/step - loss: 0.1937 -
val_loss: 0.1943
Epoch 3/10
60000/60000 [==============================] - 176s 3ms/step - loss: 0.1895 -
val_loss: 0.1876
Epoch 4/10
60000/60000 [==============================] - 170s 3ms/step - loss: 0.1872 -
val_loss: 0.1852
Epoch 5/10
60000/60000 [==============================] - 165s 3ms/step - loss: 0.1857 -
val_loss: 0.1854
Epoch 6/10
60000/60000 [==============================] - 160s 3ms/step - loss: 0.1844 -
val_loss: 0.1842
Epoch 7/10
60000/60000 [==============================] - 162s 3ms/step - loss: 0.1834 -
val_loss: 0.1826
Epoch 8/10
60000/60000 [==============================] - 169s 3ms/step - loss: 0.1825 -
val_loss: 0.1828
Epoch 9/10
60000/60000 [==============================] - 178s 3ms/step - loss: 0.1818 -
val_loss: 0.1811
Epoch 10/10
60000/60000 [==============================] - 185s 3ms/step - loss: 0.1812 -
val_loss: 0.1822
```

Out[8]:

```
<keras.callbacks.callbacks.History at 0x2f07abfbe80>
```

Sampling a grid of points from the 2D latent space and decoding them to images

In [9]:

```python
vae = results_dir.joinpath('vae')
vae.mkdir(parents=True, exist_ok=True)
```

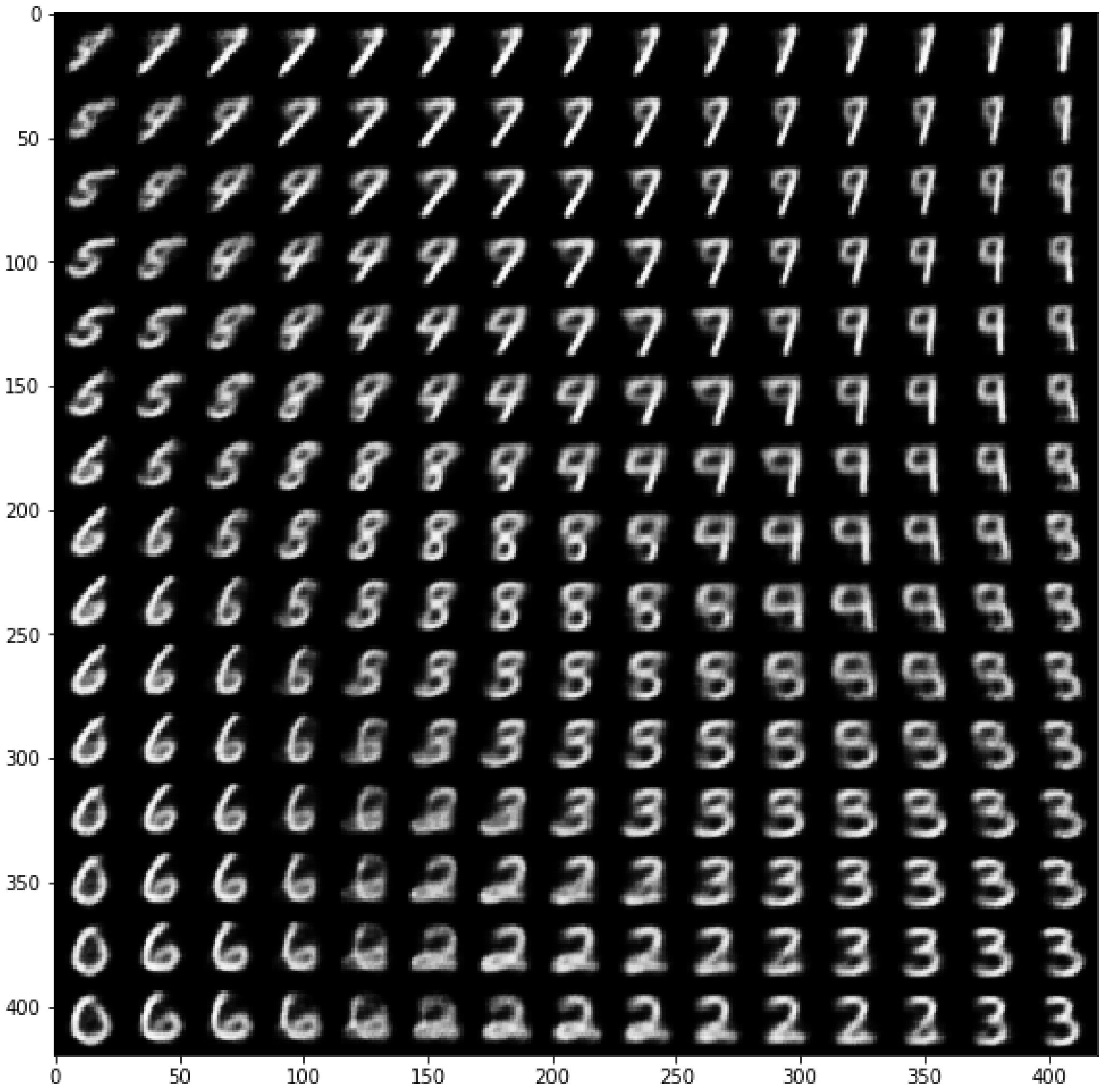
In [10]:

```
import matplotlib.pyplot as plt
from scipy.stats import norm

n = 15
digit_size = 28
figure = np.zeros((digit_size * n, digit_size * n))
grid_x = norm.ppf(np.linspace(0.05, 0.95, n))
grid_y = norm.ppf(np.linspace(0.05, 0.95, n))

for i, yi in enumerate(grid_x):
    for j, xi in enumerate(grid_y):
        z_sample = np.array([[xi, yi]])
        z_sample = np.tile(z_sample, batch_size).reshape(batch_size, 2)
        x_decoded = decoder.predict(z_sample, batch_size=batch_size)
        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=(10, 10))
plt.imshow(figure, cmap='Greys_r')
img = vae.joinpath('grid.png')
plt.savefig(img)
plt.savefig(img)
plt.show()
```



In [ ]: