Deep Learning Course (980)

Assignment Four

Assignment Goals:

- Implementing Fully Connected AutoEncoders
- Implementing Convolutional AutoEncoders
- Understand Variational Autoncoder intuition

In this assignment, you will be asked to design a Fully Connected and a CNN AutoEncoder. With a simple change in your Fully Connected AutoEncoder, you will become more familiar with Variational AutoEncoder.

DataSet: In this Assignment, you will use the MNIST handwritten digit database. You can use (x*train,*), (x*test,*) = tensorflow.keras.datasets.mnist.load_data() to load the dataset.

1. (30 points) Implement a Fully Connected AutoEncoder in TensorFlow (cf. Chapter 7). Your AutoEncoder should have a bottleneck with two neurons and Mean Squared Error (MSE) as the objective function. In an AutoEncoder, the layer with the least number of neurons is referred to as a bottleneck. Train your model on MNIST. Plot the train and test loss. Randomly select 10 images from the test set, encode them and visualize the decoded images.

2. (35 points) Implement a convolutional AutoEncoder (CAE) that uses only the following types of layers: convolution, pooling, upsampling and transpose. You are limited to use MSE. The encoder and decoder should include one or more layers, with the size and number of filters chosen by you. Start with a bottleneck of size 2, train your model on MNIST and plot the train and test loss. Randomly select 10 images from the test set, encode them and visualize the decoded images. Are the reconstructed images readable for humans? If not, try to find a CAE architecture, including a larger bottleneck, that is powerful enough to generate readable images. The bottleneck should be as small as possible for readability, this is part of the grading criteria.

3. (35 points) This question is about using an AutoEncoder to generate similar but not identical hand digits. We use a naive approach: Try to see if a trained decoder can map randomly generated inputs (random numbers) to a recognizable hand-written digit.
   A. Start with your Fully Connected and trained AutoEncoder from part 1. Try to generate new images by inputting some random numbers to the decoder (i.e. the bottleneck layer) and report your results. Hint: This is not easy. You probably want to input at least 10 random numbers.
   B. Now restrict the AutoEncoder hidden bottleneck layer(s) to have a standard multi-variate normal distribution with mean zeroes and the identity matrix as variance (i.e. no correlations). Retrain the Fully Connected AutoEncoder with the normalized bottleneck. Now randomly generate inputs to the bottleneck layer that are drawn from the multi-variate standard normal distribution, and use the random inputs to generate new images. Report your result.
   C. Are the output images different between 1) and 2)? If so, why do you think this difference occurs?

4. (20 points) Optional: change the AutoEncoder which you developed in the last part of section 3 so that it becomes a Variational AutoEncoder (Introduced by Kingma 2014; see Chapter 7.1). Does the VAE produce a different quality of output image?

Submission Notes:

Please use Jupyter Notebook. The notebook should include the final code, results, and answers. You should submit your Notebook in .pdf and .ipynb format. (penalty 10 points). Your AutoEncoders should have only one bottleneck.

Instructions:

The university policy on academic dishonesty and plagiarism (cheating) will be taken very seriously in this course. Everything submitted should be your writing or coding. You must not let other students copy your work.

```python
In [2]: import tensorflow as tf
        from tensorflow import keras
        from keras.models import Model, Sequential, load_model
        from keras.layers import Dense, Input, Conv2D, Conv2DTranspose, MaxPooling2D,
        UpSampling2D, Lambda
        from keras import backend as K
        from keras.callbacks import ModelCheckpoint
        import numpy as np
        from tensorflow.python.client import device_lib
        from matplotlib import pyplot as plt

        (x_train, _), (x_test, _) = keras.datasets.mnist.load_data()

        # normalize the data
        x_train = x_train.astype('float32') / 255.
        x_test = x_test.astype('float32') / 255.
```

In [72]:
```python
''' Part 1: Implement a Fully Connected AutoEncoder in TensorFlow
'''

# flatten the input image data
input_size = 784
x_train_flatten = x_train.reshape(-1, input_size)
x_test_flatten = x_test.reshape(-1, input_size)

# build the network with a bottleneck of two neurons
autoencoder_fc = Sequential()
autoencoder_fc.add(Dense(512, input_shape=(input_size,), activation='relu'))
autoencoder_fc.add(Dense(2, activation='relu'))
autoencoder_fc.add(Dense(512, input_shape=(input_size,), activation='relu'))
autoencoder_fc.add(Dense(input_size, activation='sigmoid'))

autoencoder_fc.summary()
```

```
_____
Layer (type)                 Output Shape              Param #
=================================================================
dense_22 (Dense)             (None, 512)               401920
_____
dense_23 (Dense)             (None, 2)                 1026
_____
dense_24 (Dense)             (None, 512)               1536
_____
dense_25 (Dense)             (None, 784)               402192
=================================================================
Total params: 806,674
Trainable params: 806,674
Non-trainable params: 0
_____
```

In [73]:
```python
# train model

print(device_lib.list_local_devices())

autoencoder_fc.compile(optimizer='adam', loss='mean_squared_error', metrics=[
'accuracy'])
best_model_checkpoint = ModelCheckpoint(
    './best_model_fc.pth',
    monitor="val_acc",
    save_best_only=True,
    save_weights_only=False
)
autoencoder_fc_history = autoencoder_fc.fit(
    x_train_flatten,
    x_train_flatten,
    epochs=100,
    batch_size=128,
    shuffle=True,
    validation_data=(x_test_flatten, x_test_flatten),
    callbacks=[best_model_checkpoint]
)
```

```
[name: "/device:CPU:0"
device_type: "CPU"
memory_limit: 268435456
locality {
}
incarnation: 12452917752891272547
, name: "/device:GPU:0"
device_type: "GPU"
memory_limit: 6700198133
locality {
  bus_id: 1
  links {
  }
}
incarnation: 14822082337168630399
physical_device_desc: "device: 0, name: GeForce GTX 1070, pci bus id: 0000:0
1:00.0, compute capability: 6.1"
]
Train on 60000 samples, validate on 10000 samples
Epoch 1/100
60000/60000 [==============================] - 3s 42us/step - loss: 0.0661 -
acc: 0.0107 - val_loss: 0.0539 - val_acc: 0.0112
Epoch 2/100
60000/60000 [==============================] - 2s 28us/step - loss: 0.0523 -
acc: 0.0124 - val_loss: 0.0501 - val_acc: 0.0103
Epoch 3/100
60000/60000 [==============================] - 2s 27us/step - loss: 0.0492 -
acc: 0.0120 - val_loss: 0.0477 - val_acc: 0.0107
Epoch 4/100
60000/60000 [==============================] - 2s 28us/step - loss: 0.0473 -
acc: 0.0101 - val_loss: 0.0465 - val_acc: 0.0096
Epoch 5/100
60000/60000 [==============================] - 2s 28us/step - loss: 0.0461 -
acc: 0.0103 - val_loss: 0.0455 - val_acc: 0.0092
Epoch 6/100
60000/60000 [==============================] - 2s 27us/step - loss: 0.0450 -
acc: 0.0122 - val_loss: 0.0446 - val_acc: 0.0105
Epoch 7/100
60000/60000 [==============================] - 2s 27us/step - loss: 0.0441 -
acc: 0.0125 - val_loss: 0.0438 - val_acc: 0.0096
Epoch 8/100
60000/60000 [==============================] - 2s 26us/step - loss: 0.0435 -
acc: 0.0119 - val_loss: 0.0433 - val_acc: 0.0110
Epoch 9/100
60000/60000 [==============================] - 2s 27us/step - loss: 0.0429 -
acc: 0.0111 - val_loss: 0.0429 - val_acc: 0.0062
Epoch 10/100
60000/60000 [==============================] - 2s 26us/step - loss: 0.0425 -
acc: 0.0106 - val_loss: 0.0427 - val_acc: 0.0080
Epoch 11/100
60000/60000 [==============================] - 2s 27us/step - loss: 0.0421 -
acc: 0.0097 - val_loss: 0.0425 - val_acc: 0.0081
Epoch 12/100
60000/60000 [==============================] - 2s 26us/step - loss: 0.0418 -
acc: 0.0096 - val_loss: 0.0421 - val_acc: 0.0081
Epoch 13/100
60000/60000 [==============================] - 2s 27us/step - loss: 0.0416 -
```

```
                     acc: 0.0097 - val_loss: 0.0418 - val_acc: 0.0106
                     Epoch 14/100
                     60000/60000 [==============================] - 2s 26us/step - loss: 0.0413 -
                     acc: 0.0101 - val_loss: 0.0418 - val_acc: 0.0085
                     Epoch 15/100
                     60000/60000 [==============================] - 2s 27us/step - loss: 0.0411 -
                     acc: 0.0094 - val_loss: 0.0416 - val_acc: 0.0105
                     Epoch 16/100
                     60000/60000 [==============================] - 2s 27us/step - loss: 0.0409 -
                     acc: 0.0091 - val_loss: 0.0415 - val_acc: 0.0101
                     Epoch 17/100
                     60000/60000 [==============================] - 2s 27us/step - loss: 0.0407 -
                     acc: 0.0101 - val_loss: 0.0415 - val_acc: 0.0072
                     Epoch 18/100
                     60000/60000 [==============================] - 2s 27us/step - loss: 0.0405 -
                     acc: 0.0101 - val_loss: 0.0412 - val_acc: 0.0105
                     Epoch 19/100
                     60000/60000 [==============================] - 2s 27us/step - loss: 0.0404 -
                     acc: 0.0105 - val_loss: 0.0411 - val_acc: 0.0093
                     Epoch 20/100
                     60000/60000 [==============================] - 2s 27us/step - loss: 0.0402 -
                     acc: 0.0100 - val_loss: 0.0411 - val_acc: 0.0126
                     Epoch 21/100
                     60000/60000 [==============================] - 2s 26us/step - loss: 0.0400 -
                     acc: 0.0110 - val_loss: 0.0409 - val_acc: 0.0109
                     Epoch 22/100
                     60000/60000 [==============================] - 2s 26us/step - loss: 0.0399 -
                     acc: 0.0099 - val_loss: 0.0409 - val_acc: 0.0095
                     Epoch 23/100
                     60000/60000 [==============================] - 2s 28us/step - loss: 0.0397 -
                     acc: 0.0106 - val_loss: 0.0407 - val_acc: 0.0104
                     Epoch 24/100
                     60000/60000 [==============================] - 2s 27us/step - loss: 0.0396 -
                     acc: 0.0106 - val_loss: 0.0407 - val_acc: 0.0096
                     Epoch 25/100
                     60000/60000 [==============================] - 2s 26us/step - loss: 0.0395 -
                     acc: 0.0108 - val_loss: 0.0406 - val_acc: 0.0115
                     Epoch 26/100
                     60000/60000 [==============================] - 2s 27us/step - loss: 0.0394 -
                     acc: 0.0108 - val_loss: 0.0405 - val_acc: 0.0093
                     Epoch 27/100
                     60000/60000 [==============================] - 2s 26us/step - loss: 0.0393 -
                     acc: 0.0104 - val_loss: 0.0406 - val_acc: 0.0102
                     Epoch 28/100
                     60000/60000 [==============================] - 2s 26us/step - loss: 0.0391 -
                     acc: 0.0110 - val_loss: 0.0404 - val_acc: 0.0110
                     Epoch 29/100
                     60000/60000 [==============================] - 2s 26us/step - loss: 0.0390 -
                     acc: 0.0111 - val_loss: 0.0403 - val_acc: 0.0110
                     Epoch 30/100
                     60000/60000 [==============================] - 2s 27us/step - loss: 0.0389 -
                     acc: 0.0114 - val_loss: 0.0403 - val_acc: 0.0119
                     Epoch 31/100
                     60000/60000 [==============================] - 2s 26us/step - loss: 0.0388 -
                     acc: 0.0109 - val_loss: 0.0403 - val_acc: 0.0123
                     Epoch 32/100
                     60000/60000 [==============================] - 2s 28us/step - loss: 0.0387 -
```

```
                     acc: 0.0108 - val_loss: 0.0403 - val_acc: 0.0101
                     Epoch 33/100
                     60000/60000 [==============================] - 2s 28us/step - loss: 0.0387 -
                     acc: 0.0107 - val_loss: 0.0402 - val_acc: 0.0108
                     Epoch 34/100
                     60000/60000 [==============================] - 2s 28us/step - loss: 0.0386 -
                     acc: 0.0118 - val_loss: 0.0401 - val_acc: 0.0128
                     Epoch 35/100
                     60000/60000 [==============================] - 2s 28us/step - loss: 0.0385 -
                     acc: 0.0109 - val_loss: 0.0401 - val_acc: 0.0131
                     Epoch 36/100
                     60000/60000 [==============================] - 2s 28us/step - loss: 0.0384 -
                     acc: 0.0109 - val_loss: 0.0401 - val_acc: 0.0105
                     Epoch 37/100
                     60000/60000 [==============================] - 2s 27us/step - loss: 0.0383 -
                     acc: 0.0111 - val_loss: 0.0400 - val_acc: 0.0085
                     Epoch 38/100
                     60000/60000 [==============================] - 2s 29us/step - loss: 0.0383 -
                     acc: 0.0111 - val_loss: 0.0400 - val_acc: 0.0118
                     Epoch 39/100
                     60000/60000 [==============================] - 2s 28us/step - loss: 0.0381 -
                     acc: 0.0112 - val_loss: 0.0399 - val_acc: 0.0107
                     Epoch 40/100
                     60000/60000 [==============================] - 2s 28us/step - loss: 0.0381 -
                     acc: 0.0109 - val_loss: 0.0399 - val_acc: 0.0108
                     Epoch 41/100
                     60000/60000 [==============================] - 2s 27us/step - loss: 0.0380 -
                     acc: 0.0113 - val_loss: 0.0399 - val_acc: 0.0134
                     Epoch 42/100
                     60000/60000 [==============================] - 2s 29us/step - loss: 0.0379 -
                     acc: 0.0114 - val_loss: 0.0398 - val_acc: 0.0132
                     Epoch 43/100
                     60000/60000 [==============================] - 2s 28us/step - loss: 0.0379 -
                     acc: 0.0113 - val_loss: 0.0399 - val_acc: 0.0135
                     Epoch 44/100
                     60000/60000 [==============================] - 2s 28us/step - loss: 0.0378 -
                     acc: 0.0114 - val_loss: 0.0398 - val_acc: 0.0106
                     Epoch 45/100
                     60000/60000 [==============================] - 2s 29us/step - loss: 0.0378 -
                     acc: 0.0109 - val_loss: 0.0397 - val_acc: 0.0101
                     Epoch 46/100
                     60000/60000 [==============================] - 2s 29us/step - loss: 0.0377 -
                     acc: 0.0115 - val_loss: 0.0398 - val_acc: 0.0113
                     Epoch 47/100
                     60000/60000 [==============================] - 2s 28us/step - loss: 0.0376 -
                     acc: 0.0114 - val_loss: 0.0398 - val_acc: 0.0115
                     Epoch 48/100
                     60000/60000 [==============================] - 2s 29us/step - loss: 0.0376 -
                     acc: 0.0116 - val_loss: 0.0397 - val_acc: 0.0100
                     Epoch 49/100
                     60000/60000 [==============================] - 2s 28us/step - loss: 0.0375 -
                     acc: 0.0114 - val_loss: 0.0398 - val_acc: 0.0107
                     Epoch 50/100
                     60000/60000 [==============================] - 2s 28us/step - loss: 0.0375 -
                     acc: 0.0112 - val_loss: 0.0398 - val_acc: 0.0114
                     Epoch 51/100
                     60000/60000 [==============================] - 2s 28us/step - loss: 0.0374 -
```

```
                      acc: 0.0114 - val_loss: 0.0397 - val_acc: 0.0122
                      Epoch 52/100
                      60000/60000 [==============================] - 2s 27us/step - loss: 0.0374 -
                      acc: 0.0118 - val_loss: 0.0399 - val_acc: 0.0109
                      Epoch 53/100
                      60000/60000 [==============================] - 2s 28us/step - loss: 0.0373 -
                      acc: 0.0117 - val_loss: 0.0397 - val_acc: 0.0093
                      Epoch 54/100
                      60000/60000 [==============================] - 2s 27us/step - loss: 0.0373 -
                      acc: 0.0109 - val_loss: 0.0398 - val_acc: 0.0138
                      Epoch 55/100
                      60000/60000 [==============================] - 2s 27us/step - loss: 0.0372 -
                      acc: 0.0119 - val_loss: 0.0396 - val_acc: 0.0105
                      Epoch 56/100
                      60000/60000 [==============================] - 2s 27us/step - loss: 0.0372 -
                      acc: 0.0112 - val_loss: 0.0397 - val_acc: 0.0121
                      Epoch 57/100
                      60000/60000 [==============================] - 2s 26us/step - loss: 0.0371 -
                      acc: 0.0126 - val_loss: 0.0395 - val_acc: 0.0126
                      Epoch 58/100
                      60000/60000 [==============================] - 2s 26us/step - loss: 0.0371 -
                      acc: 0.0115 - val_loss: 0.0397 - val_acc: 0.0115
                      Epoch 59/100
                      60000/60000 [==============================] - 2s 26us/step - loss: 0.0370 -
                      acc: 0.0116 - val_loss: 0.0396 - val_acc: 0.0103
                      Epoch 60/100
                      60000/60000 [==============================] - 2s 26us/step - loss: 0.0370 -
                      acc: 0.0114 - val_loss: 0.0396 - val_acc: 0.0110
                      Epoch 61/100
                      60000/60000 [==============================] - 2s 26us/step - loss: 0.0370 -
                      acc: 0.0117 - val_loss: 0.0396 - val_acc: 0.0110
                      Epoch 62/100
                      60000/60000 [==============================] - 2s 26us/step - loss: 0.0369 -
                      acc: 0.0118 - val_loss: 0.0396 - val_acc: 0.0117
                      Epoch 63/100
                      60000/60000 [==============================] - 2s 26us/step - loss: 0.0369 -
                      acc: 0.0116 - val_loss: 0.0395 - val_acc: 0.0103
                      Epoch 64/100
                      60000/60000 [==============================] - 2s 27us/step - loss: 0.0369 -
                      acc: 0.0116 - val_loss: 0.0396 - val_acc: 0.0127
                      Epoch 65/100
                      60000/60000 [==============================] - 2s 26us/step - loss: 0.0368 -
                      acc: 0.0118 - val_loss: 0.0396 - val_acc: 0.0130
                      Epoch 66/100
                      60000/60000 [==============================] - 2s 27us/step - loss: 0.0368 -
                      acc: 0.0115 - val_loss: 0.0396 - val_acc: 0.0109
                      Epoch 67/100
                      60000/60000 [==============================] - 2s 26us/step - loss: 0.0368 -
                      acc: 0.0117 - val_loss: 0.0396 - val_acc: 0.0123
                      Epoch 68/100
                      60000/60000 [==============================] - 2s 27us/step - loss: 0.0367 -
                      acc: 0.0116 - val_loss: 0.0398 - val_acc: 0.0105
                      Epoch 69/100
                      60000/60000 [==============================] - 2s 26us/step - loss: 0.0367 -
                      acc: 0.0119 - val_loss: 0.0396 - val_acc: 0.0154
                      Epoch 70/100
                      60000/60000 [==============================] - 2s 26us/step - loss: 0.0367 -
```

```
                    acc: 0.0121 - val_loss: 0.0395 - val_acc: 0.0138
                    Epoch 71/100
                    60000/60000 [==============================] - 2s 26us/step - loss: 0.0366 -
                    acc: 0.0119 - val_loss: 0.0395 - val_acc: 0.0105
                    Epoch 72/100
                    60000/60000 [==============================] - 2s 27us/step - loss: 0.0366 -
                    acc: 0.0117 - val_loss: 0.0396 - val_acc: 0.0137
                    Epoch 73/100
                    60000/60000 [==============================] - 2s 26us/step - loss: 0.0366 -
                    acc: 0.0117 - val_loss: 0.0397 - val_acc: 0.0127
                    Epoch 74/100
                    60000/60000 [==============================] - 2s 27us/step - loss: 0.0365 -
                    acc: 0.0121 - val_loss: 0.0397 - val_acc: 0.0141
                    Epoch 75/100
                    60000/60000 [==============================] - 2s 26us/step - loss: 0.0365 -
                    acc: 0.0122 - val_loss: 0.0397 - val_acc: 0.0100
                    Epoch 76/100
                    60000/60000 [==============================] - 2s 27us/step - loss: 0.0365 -
                    acc: 0.0119 - val_loss: 0.0395 - val_acc: 0.0129
                    Epoch 77/100
                    60000/60000 [==============================] - 2s 26us/step - loss: 0.0364 -
                    acc: 0.0121 - val_loss: 0.0398 - val_acc: 0.0111
                    Epoch 78/100
                    60000/60000 [==============================] - 2s 28us/step - loss: 0.0364 -
                    acc: 0.0121 - val_loss: 0.0395 - val_acc: 0.0119
                    Epoch 79/100
                    60000/60000 [==============================] - 2s 28us/step - loss: 0.0364 -
                    acc: 0.0118 - val_loss: 0.0395 - val_acc: 0.0104
                    Epoch 80/100
                    60000/60000 [==============================] - 2s 28us/step - loss: 0.0363 -
                    acc: 0.0118 - val_loss: 0.0396 - val_acc: 0.0117
                    Epoch 81/100
                    60000/60000 [==============================] - 2s 27us/step - loss: 0.0363 -
                    acc: 0.0123 - val_loss: 0.0395 - val_acc: 0.0101
                    Epoch 82/100
                    60000/60000 [==============================] - 2s 27us/step - loss: 0.0363 -
                    acc: 0.0125 - val_loss: 0.0396 - val_acc: 0.0119
                    Epoch 83/100
                    60000/60000 [==============================] - 2s 27us/step - loss: 0.0363 -
                    acc: 0.0124 - val_loss: 0.0397 - val_acc: 0.0111
                    Epoch 84/100
                    60000/60000 [==============================] - 2s 27us/step - loss: 0.0363 -
                    acc: 0.0113 - val_loss: 0.0395 - val_acc: 0.0101
                    Epoch 85/100
                    60000/60000 [==============================] - 2s 28us/step - loss: 0.0362 -
                    acc: 0.0123 - val_loss: 0.0397 - val_acc: 0.0118
                    Epoch 86/100
                    60000/60000 [==============================] - 2s 28us/step - loss: 0.0362 -
                    acc: 0.0124 - val_loss: 0.0397 - val_acc: 0.0105
                    Epoch 87/100
                    60000/60000 [==============================] - 2s 28us/step - loss: 0.0362 -
                    acc: 0.0120 - val_loss: 0.0395 - val_acc: 0.0124
                    Epoch 88/100
                    60000/60000 [==============================] - 2s 31us/step - loss: 0.0361 -
                    acc: 0.0116 - val_loss: 0.0396 - val_acc: 0.0136
                    Epoch 89/100
                    60000/60000 [==============================] - 2s 32us/step - loss: 0.0361 -
```

```
            acc: 0.0120 - val_loss: 0.0395 - val_acc: 0.0110
            Epoch 90/100
            60000/60000 [==============================] - 2s 29us/step - loss: 0.0361 -
            acc: 0.0123 - val_loss: 0.0397 - val_acc: 0.0110
            Epoch 91/100
            60000/60000 [==============================] - 2s 27us/step - loss: 0.0361 -
            acc: 0.0115 - val_loss: 0.0397 - val_acc: 0.0107
            Epoch 92/100
            60000/60000 [==============================] - 2s 27us/step - loss: 0.0361 -
            acc: 0.0122 - val_loss: 0.0395 - val_acc: 0.0130
            Epoch 93/100
            60000/60000 [==============================] - 2s 27us/step - loss: 0.0360 -
            acc: 0.0121 - val_loss: 0.0396 - val_acc: 0.0108
            Epoch 94/100
            60000/60000 [==============================] - 2s 29us/step - loss: 0.0360 -
            acc: 0.0121 - val_loss: 0.0396 - val_acc: 0.0118
            Epoch 95/100
            60000/60000 [==============================] - 2s 27us/step - loss: 0.0360 -
            acc: 0.0128 - val_loss: 0.0395 - val_acc: 0.0114
            Epoch 96/100
            60000/60000 [==============================] - 2s 29us/step - loss: 0.0360 -
            acc: 0.0120 - val_loss: 0.0396 - val_acc: 0.0122
            Epoch 97/100
            60000/60000 [==============================] - 2s 29us/step - loss: 0.0359 -
            acc: 0.0124 - val_loss: 0.0396 - val_acc: 0.0129
            Epoch 98/100
            60000/60000 [==============================] - 2s 27us/step - loss: 0.0359 -
            acc: 0.0124 - val_loss: 0.0395 - val_acc: 0.0101
            Epoch 99/100
            60000/60000 [==============================] - 2s 27us/step - loss: 0.0359 -
            acc: 0.0122 - val_loss: 0.0396 - val_acc: 0.0115
            Epoch 100/100
            60000/60000 [==============================] - 2s 27us/step - loss: 0.0359 -
            acc: 0.0116 - val_loss: 0.0396 - val_acc: 0.0100
```

In [74]:
```python
# plot the train and test loss
plt.plot(autoencoder_fc_history.history['loss'])
plt.plot(autoencoder_fc_history.history['val_loss'])
plt.title('Train and test loss for fully connected AE')
plt.ylabel('Loss')
plt.xlabel('Epoch')
plt.legend(['Train', 'Test'], loc='upper left')
plt.show()
```
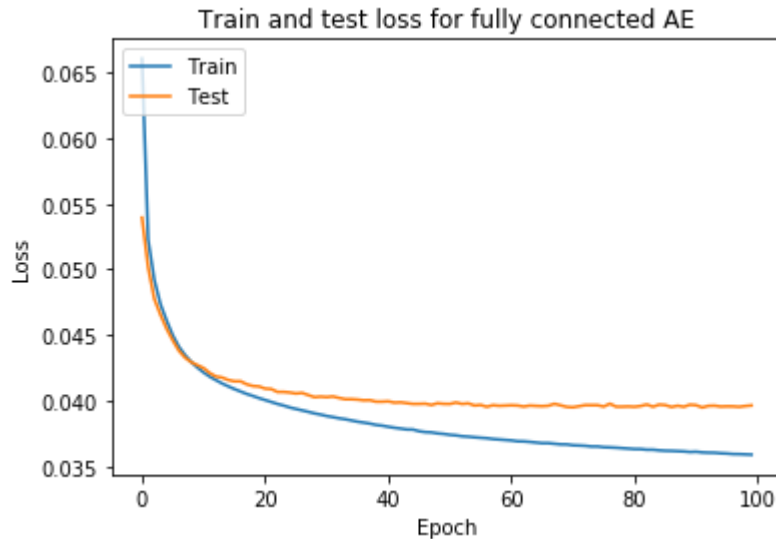

In [75]:
```python
# visualization

img_num = 10
autoencoder_fc_best = load_model('./best_model_fc.pth')
plt.figure(figsize=(18, 4))
plt.gray()
for i in range(img_num):
    # randomly pick an image from test dataset
    chosen_test_img = x_test_flatten[np.random.randint(x_test_flatten.shape[0
])]
    img_decoded = autoencoder_fc_best.predict(chosen_test_img.reshape(1, 784))
    ax = plt.subplot(2, img_num, i+1)
    plt.imshow(chosen_test_img.reshape(28, 28))
    ax.axis('off')

    ax = plt.subplot(2, img_num, img_num+i+1)
    plt.imshow(img_decoded.reshape(28, 28))
    ax.axis('off')
plt.show()
```

In [47]:
```python
''' Part 2: Implement a convolutional AutoEncoder (CAE)
'''

x_train = x_train.reshape((len(x_train), 28, 28, 1))
x_test = x_test.reshape((len(x_test), 28, 28, 1))

# build the network with a bottleneck of two neurons
autoencoder_conv = Sequential()
# encoder network
autoencoder_conv.add(Conv2D(32, (3, 3), activation='relu', padding='same', input_shape=x_train.shape[1:]))
autoencoder_conv.add(MaxPooling2D(pool_size=(2, 2)))
autoencoder_conv.add(Conv2D(64, (3, 3), activation='relu', padding='same'))
autoencoder_conv.add(MaxPooling2D(pool_size=(2, 2)))
autoencoder_conv.add(Conv2D(128, (3, 3), activation='relu', padding='same'))
autoencoder_conv.add(MaxPooling2D(pool_size=(2, 2)))
autoencoder_conv.add(Conv2D(2, (3, 3), activation='relu', padding='same'))
autoencoder_conv.add(MaxPooling2D(pool_size=(2, 2)))
# decoder network
autoencoder_conv.add(Conv2DTranspose(64, (7, 7), activation='relu', padding='valid'))
autoencoder_conv.add(UpSampling2D((2, 2)))
autoencoder_conv.add(Conv2DTranspose(32, (3, 3), activation='relu', padding='same'))
autoencoder_conv.add(UpSampling2D((2, 2)))
autoencoder_conv.add(Conv2D(1, (3, 3), activation='sigmoid', padding='same'))

autoencoder_conv.summary()
```

```
_____
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_1 (Conv2D)            (None, 28, 28, 32)        320
_____
max_pooling2d_1 (MaxPooling2 (None, 14, 14, 32)        0
_____
conv2d_2 (Conv2D)            (None, 14, 14, 64)        18496
_____
max_pooling2d_2 (MaxPooling2 (None, 7, 7, 64)          0
_____
conv2d_3 (Conv2D)            (None, 7, 7, 128)         73856
_____
max_pooling2d_3 (MaxPooling2 (None, 3, 3, 128)         0
_____
conv2d_4 (Conv2D)            (None, 3, 3, 2)           2306
_____
max_pooling2d_4 (MaxPooling2 (None, 1, 1, 2)           0
_____
conv2d_transpose_1 (Conv2DTr (None, 7, 7, 64)          6336
_____
up_sampling2d_1 (UpSampling2 (None, 14, 14, 64)        0
_____
conv2d_transpose_2 (Conv2DTr (None, 14, 14, 32)        18464
_____
up_sampling2d_2 (UpSampling2 (None, 28, 28, 32)        0
_____
conv2d_5 (Conv2D)            (None, 28, 28, 1)         289
=================================================================
Total params: 120,067
Trainable params: 120,067
Non-trainable params: 0
_____
```

In [48]:
```python
# train model

print(device_lib.list_local_devices())

autoencoder_conv.compile(optimizer='adam', loss='mean_squared_error', metrics=
['accuracy'])
best_model_checkpoint = ModelCheckpoint(
    './best_model_conv.pth',
    monitor="val_acc",
    save_best_only=True,
    save_weights_only=False
)
autoencoder_conv_history = autoencoder_conv.fit(
    x_train,
    x_train,
    epochs=100,
    batch_size=256,
    shuffle=True,
    validation_data=(x_test, x_test),
    callbacks=[best_model_checkpoint]
)
```

```
[name: "/device:CPU:0"
device_type: "CPU"
memory_limit: 268435456
locality {
}
incarnation: 15526130026912515061
, name: "/device:GPU:0"
device_type: "GPU"
memory_limit: 6700198133
locality {
  bus_id: 1
  links {
  }
}
incarnation: 2915467488262415717
physical_device_desc: "device: 0, name: GeForce GTX 1070, pci bus id: 0000:0
1:00.0, compute capability: 6.1"
]
Train on 60000 samples, validate on 10000 samples
Epoch 1/100
60000/60000 [==============================] - 6s 100us/step - loss: 0.0733 -
acc: 0.7987 - val_loss: 0.0582 - val_acc: 0.7959
Epoch 2/100
60000/60000 [==============================] - 4s 69us/step - loss: 0.0553 -
acc: 0.7955 - val_loss: 0.0525 - val_acc: 0.7943
Epoch 3/100
60000/60000 [==============================] - 4s 68us/step - loss: 0.0510 -
acc: 0.7951 - val_loss: 0.0499 - val_acc: 0.7942
Epoch 4/100
60000/60000 [==============================] - 4s 71us/step - loss: 0.0487 -
acc: 0.7950 - val_loss: 0.0480 - val_acc: 0.7939
Epoch 5/100
60000/60000 [==============================] - 4s 68us/step - loss: 0.0472 -
acc: 0.7954 - val_loss: 0.0465 - val_acc: 0.7977
Epoch 6/100
60000/60000 [==============================] - 4s 70us/step - loss: 0.0461 -
acc: 0.7960 - val_loss: 0.0458 - val_acc: 0.7931
Epoch 7/100
60000/60000 [==============================] - 4s 69us/step - loss: 0.0454 -
acc: 0.7964 - val_loss: 0.0451 - val_acc: 0.7935
Epoch 8/100
60000/60000 [==============================] - 4s 68us/step - loss: 0.0448 -
acc: 0.7967 - val_loss: 0.0445 - val_acc: 0.7971
Epoch 9/100
60000/60000 [==============================] - 4s 69us/step - loss: 0.0444 -
acc: 0.7970 - val_loss: 0.0441 - val_acc: 0.7939
Epoch 10/100
60000/60000 [==============================] - 4s 69us/step - loss: 0.0439 -
acc: 0.7972 - val_loss: 0.0438 - val_acc: 0.7965
Epoch 11/100
60000/60000 [==============================] - 4s 69us/step - loss: 0.0436 -
acc: 0.7974 - val_loss: 0.0434 - val_acc: 0.7955
Epoch 12/100
60000/60000 [==============================] - 4s 68us/step - loss: 0.0433 -
acc: 0.7975 - val_loss: 0.0439 - val_acc: 0.7931
Epoch 13/100
60000/60000 [==============================] - 4s 69us/step - loss: 0.0431 -
```

```
                    acc: 0.7976 - val_loss: 0.0437 - val_acc: 0.7995
                    Epoch 14/100
                    60000/60000 [==============================] - 4s 69us/step - loss: 0.0427 -
                    acc: 0.7978 - val_loss: 0.0430 - val_acc: 0.7988
                    Epoch 15/100
                    60000/60000 [==============================] - 4s 70us/step - loss: 0.0425 -
                    acc: 0.7979 - val_loss: 0.0431 - val_acc: 0.7927
                    Epoch 16/100
                    60000/60000 [==============================] - 4s 68us/step - loss: 0.0423 -
                    acc: 0.7980 - val_loss: 0.0423 - val_acc: 0.7987
                    Epoch 17/100
                    60000/60000 [==============================] - 4s 69us/step - loss: 0.0421 -
                    acc: 0.7981 - val_loss: 0.0422 - val_acc: 0.7969
                    Epoch 18/100
                    60000/60000 [==============================] - 4s 69us/step - loss: 0.0419 -
                    acc: 0.7982 - val_loss: 0.0422 - val_acc: 0.7992
                    Epoch 19/100
                    60000/60000 [==============================] - 4s 70us/step - loss: 0.0417 -
                    acc: 0.7983 - val_loss: 0.0418 - val_acc: 0.7960
                    Epoch 20/100
                    60000/60000 [==============================] - 4s 69us/step - loss: 0.0416 -
                    acc: 0.7983 - val_loss: 0.0416 - val_acc: 0.7965
                    Epoch 21/100
                    60000/60000 [==============================] - 4s 68us/step - loss: 0.0414 -
                    acc: 0.7984 - val_loss: 0.0414 - val_acc: 0.7968
                    Epoch 22/100
                    60000/60000 [==============================] - 4s 69us/step - loss: 0.0412 -
                    acc: 0.7985 - val_loss: 0.0417 - val_acc: 0.7985
                    Epoch 23/100
                    60000/60000 [==============================] - 4s 68us/step - loss: 0.0411 -
                    acc: 0.7985 - val_loss: 0.0414 - val_acc: 0.7952
                    Epoch 24/100
                    60000/60000 [==============================] - 4s 69us/step - loss: 0.0410 -
                    acc: 0.7985 - val_loss: 0.0413 - val_acc: 0.7956
                    Epoch 25/100
                    60000/60000 [==============================] - 4s 70us/step - loss: 0.0409 -
                    acc: 0.7986 - val_loss: 0.0413 - val_acc: 0.7946
                    Epoch 26/100
                    60000/60000 [==============================] - 4s 69us/step - loss: 0.0407 -
                    acc: 0.7987 - val_loss: 0.0412 - val_acc: 0.7988
                    Epoch 27/100
                    60000/60000 [==============================] - 4s 69us/step - loss: 0.0407 -
                    acc: 0.7987 - val_loss: 0.0410 - val_acc: 0.7986
                    Epoch 28/100
                    60000/60000 [==============================] - 4s 70us/step - loss: 0.0405 -
                    acc: 0.7988 - val_loss: 0.0407 - val_acc: 0.7971
                    Epoch 29/100
                    60000/60000 [==============================] - 4s 71us/step - loss: 0.0405 -
                    acc: 0.7988 - val_loss: 0.0412 - val_acc: 0.7988
                    Epoch 30/100
                    60000/60000 [==============================] - 4s 71us/step - loss: 0.0404 -
                    acc: 0.7988 - val_loss: 0.0411 - val_acc: 0.7975
                    Epoch 31/100
                    60000/60000 [==============================] - 4s 69us/step - loss: 0.0402 -
                    acc: 0.7990 - val_loss: 0.0410 - val_acc: 0.7994
                    Epoch 32/100
                    60000/60000 [==============================] - 4s 69us/step - loss: 0.0401 -
```

```
                acc: 0.7990 - val_loss: 0.0406 - val_acc: 0.7987
                Epoch 33/100
                60000/60000 [==============================] - 4s 68us/step - loss: 0.0400 -
                acc: 0.7990 - val_loss: 0.0406 - val_acc: 0.7956
                Epoch 34/100
                60000/60000 [==============================] - 4s 71us/step - loss: 0.0399 -
                acc: 0.7990 - val_loss: 0.0406 - val_acc: 0.7983
                Epoch 35/100
                60000/60000 [==============================] - 4s 69us/step - loss: 0.0398 -
                acc: 0.7991 - val_loss: 0.0407 - val_acc: 0.7986
                Epoch 36/100
                60000/60000 [==============================] - 4s 71us/step - loss: 0.0397 -
                acc: 0.7991 - val_loss: 0.0403 - val_acc: 0.7974
                Epoch 37/100
                60000/60000 [==============================] - 4s 68us/step - loss: 0.0396 -
                acc: 0.7992 - val_loss: 0.0401 - val_acc: 0.7978
                Epoch 38/100
                60000/60000 [==============================] - 4s 69us/step - loss: 0.0395 -
                acc: 0.7992 - val_loss: 0.0406 - val_acc: 0.7963
                Epoch 39/100
                60000/60000 [==============================] - 4s 72us/step - loss: 0.0395 -
                acc: 0.7993 - val_loss: 0.0401 - val_acc: 0.7958
                Epoch 40/100
                60000/60000 [==============================] - 4s 71us/step - loss: 0.0394 -
                acc: 0.7993 - val_loss: 0.0403 - val_acc: 0.7980
                Epoch 41/100
                60000/60000 [==============================] - 4s 71us/step - loss: 0.0393 -
                acc: 0.7993 - val_loss: 0.0410 - val_acc: 0.7992
                Epoch 42/100
                60000/60000 [==============================] - 4s 70us/step - loss: 0.0393 -
                acc: 0.7993 - val_loss: 0.0400 - val_acc: 0.7971
                Epoch 43/100
                60000/60000 [==============================] - 4s 71us/step - loss: 0.0392 -
                acc: 0.7994 - val_loss: 0.0403 - val_acc: 0.7961
                Epoch 44/100
                60000/60000 [==============================] - 4s 70us/step - loss: 0.0391 -
                acc: 0.7994 - val_loss: 0.0400 - val_acc: 0.7973
                Epoch 45/100
                60000/60000 [==============================] - 4s 69us/step - loss: 0.0392 -
                acc: 0.7994 - val_loss: 0.0399 - val_acc: 0.7971
                Epoch 46/100
                60000/60000 [==============================] - 4s 69us/step - loss: 0.0390 -
                acc: 0.7995 - val_loss: 0.0397 - val_acc: 0.7985
                Epoch 47/100
                60000/60000 [==============================] - 4s 69us/step - loss: 0.0389 -
                acc: 0.7996 - val_loss: 0.0399 - val_acc: 0.7961
                Epoch 48/100
                60000/60000 [==============================] - 4s 72us/step - loss: 0.0388 -
                acc: 0.7996 - val_loss: 0.0398 - val_acc: 0.7996
                Epoch 49/100
                60000/60000 [==============================] - 4s 70us/step - loss: 0.0388 -
                acc: 0.7996 - val_loss: 0.0398 - val_acc: 0.7980
                Epoch 50/100
                60000/60000 [==============================] - 4s 69us/step - loss: 0.0388 -
                acc: 0.7996 - val_loss: 0.0396 - val_acc: 0.7969
                Epoch 51/100
                60000/60000 [==============================] - 4s 69us/step - loss: 0.0387 -
```

```
                acc: 0.7996 - val_loss: 0.0397 - val_acc: 0.7986
                Epoch 52/100
                60000/60000 [==============================] - 4s 70us/step - loss: 0.0387 -
                acc: 0.7996 - val_loss: 0.0398 - val_acc: 0.7987
                Epoch 53/100
                60000/60000 [==============================] - 4s 71us/step - loss: 0.0386 -
                acc: 0.7997 - val_loss: 0.0397 - val_acc: 0.7965
                Epoch 54/100
                60000/60000 [==============================] - 4s 70us/step - loss: 0.0386 -
                acc: 0.7997 - val_loss: 0.0407 - val_acc: 0.8007
                Epoch 55/100
                60000/60000 [==============================] - 4s 69us/step - loss: 0.0385 -
                acc: 0.7998 - val_loss: 0.0396 - val_acc: 0.7983
                Epoch 56/100
                60000/60000 [==============================] - 4s 69us/step - loss: 0.0384 -
                acc: 0.7997 - val_loss: 0.0396 - val_acc: 0.7982
                Epoch 57/100
                60000/60000 [==============================] - 4s 69us/step - loss: 0.0384 -
                acc: 0.7998 - val_loss: 0.0395 - val_acc: 0.7966
                Epoch 58/100
                60000/60000 [==============================] - 4s 70us/step - loss: 0.0384 -
                acc: 0.7998 - val_loss: 0.0397 - val_acc: 0.7951
                Epoch 59/100
                60000/60000 [==============================] - 4s 69us/step - loss: 0.0383 -
                acc: 0.7998 - val_loss: 0.0395 - val_acc: 0.7969
                Epoch 60/100
                60000/60000 [==============================] - 4s 68us/step - loss: 0.0382 -
                acc: 0.7999 - val_loss: 0.0395 - val_acc: 0.7977
                Epoch 61/100
                60000/60000 [==============================] - 4s 68us/step - loss: 0.0382 -
                acc: 0.7999 - val_loss: 0.0399 - val_acc: 0.7979
                Epoch 62/100
                60000/60000 [==============================] - 4s 70us/step - loss: 0.0382 -
                acc: 0.7999 - val_loss: 0.0397 - val_acc: 0.7977
                Epoch 63/100
                60000/60000 [==============================] - 4s 71us/step - loss: 0.0382 -
                acc: 0.7999 - val_loss: 0.0397 - val_acc: 0.7980
                Epoch 64/100
                60000/60000 [==============================] - 4s 71us/step - loss: 0.0381 -
                acc: 0.8000 - val_loss: 0.0396 - val_acc: 0.7954
                Epoch 65/100
                60000/60000 [==============================] - 4s 70us/step - loss: 0.0381 -
                acc: 0.8000 - val_loss: 0.0396 - val_acc: 0.7952
                Epoch 66/100
                60000/60000 [==============================] - 4s 70us/step - loss: 0.0380 -
                acc: 0.8000 - val_loss: 0.0395 - val_acc: 0.7958
                Epoch 67/100
                60000/60000 [==============================] - 4s 70us/step - loss: 0.0380 -
                acc: 0.8000 - val_loss: 0.0397 - val_acc: 0.7977
                Epoch 68/100
                60000/60000 [==============================] - 4s 70us/step - loss: 0.0380 -
                acc: 0.8000 - val_loss: 0.0395 - val_acc: 0.7987
                Epoch 69/100
                60000/60000 [==============================] - 4s 70us/step - loss: 0.0379 -
                acc: 0.8001 - val_loss: 0.0395 - val_acc: 0.7991
                Epoch 70/100
                60000/60000 [==============================] - 4s 69us/step - loss: 0.0379 -
```

```
                    acc: 0.8001 - val_loss: 0.0397 - val_acc: 0.7984
                    Epoch 71/100
                    60000/60000 [==============================] - 4s 69us/step - loss: 0.0378 -
                    acc: 0.8001 - val_loss: 0.0396 - val_acc: 0.7976
                    Epoch 72/100
                    60000/60000 [==============================] - 4s 69us/step - loss: 0.0378 -
                    acc: 0.8001 - val_loss: 0.0396 - val_acc: 0.7960
                    Epoch 73/100
                    60000/60000 [==============================] - 4s 69us/step - loss: 0.0377 -
                    acc: 0.8001 - val_loss: 0.0397 - val_acc: 0.7965
                    Epoch 74/100
                    60000/60000 [==============================] - 4s 69us/step - loss: 0.0378 -
                    acc: 0.8001 - val_loss: 0.0395 - val_acc: 0.7962
                    Epoch 75/100
                    60000/60000 [==============================] - 4s 72us/step - loss: 0.0377 -
                    acc: 0.8001 - val_loss: 0.0394 - val_acc: 0.7975
                    Epoch 76/100
                    60000/60000 [==============================] - 4s 70us/step - loss: 0.0376 -
                    acc: 0.8002 - val_loss: 0.0393 - val_acc: 0.7980
                    Epoch 77/100
                    60000/60000 [==============================] - 4s 70us/step - loss: 0.0377 -
                    acc: 0.8002 - val_loss: 0.0394 - val_acc: 0.7976
                    Epoch 78/100
                    60000/60000 [==============================] - 4s 69us/step - loss: 0.0375 -
                    acc: 0.8003 - val_loss: 0.0394 - val_acc: 0.7975
                    Epoch 79/100
                    60000/60000 [==============================] - 4s 71us/step - loss: 0.0377 -
                    acc: 0.8001 - val_loss: 0.0399 - val_acc: 0.7998
                    Epoch 80/100
                    60000/60000 [==============================] - 4s 69us/step - loss: 0.0375 -
                    acc: 0.8003 - val_loss: 0.0392 - val_acc: 0.7981
                    Epoch 81/100
                    60000/60000 [==============================] - 4s 71us/step - loss: 0.0375 -
                    acc: 0.8003 - val_loss: 0.0395 - val_acc: 0.7986
                    Epoch 82/100
                    60000/60000 [==============================] - 4s 71us/step - loss: 0.0375 -
                    acc: 0.8003 - val_loss: 0.0399 - val_acc: 0.8001
                    Epoch 83/100
                    60000/60000 [==============================] - 4s 72us/step - loss: 0.0375 -
                    acc: 0.8002 - val_loss: 0.0393 - val_acc: 0.7984
                    Epoch 84/100
                    60000/60000 [==============================] - 4s 70us/step - loss: 0.0374 -
                    acc: 0.8003 - val_loss: 0.0397 - val_acc: 0.7990
                    Epoch 85/100
                    60000/60000 [==============================] - 4s 72us/step - loss: 0.0373 -
                    acc: 0.8004 - val_loss: 0.0391 - val_acc: 0.7980
                    Epoch 86/100
                    60000/60000 [==============================] - 4s 72us/step - loss: 0.0373 -
                    acc: 0.8004 - val_loss: 0.0397 - val_acc: 0.7959
                    Epoch 87/100
                    60000/60000 [==============================] - 4s 70us/step - loss: 0.0373 -
                    acc: 0.8004 - val_loss: 0.0394 - val_acc: 0.7993
                    Epoch 88/100
                    60000/60000 [==============================] - 4s 69us/step - loss: 0.0373 -
                    acc: 0.8004 - val_loss: 0.0396 - val_acc: 0.7990
                    Epoch 89/100
                    60000/60000 [==============================] - 4s 69us/step - loss: 0.0373 -
```

```
acc: 0.8004 - val_loss: 0.0395 - val_acc: 0.7959
Epoch 90/100
60000/60000 [==============================] - 4s 69us/step - loss: 0.0372 -
acc: 0.8004 - val_loss: 0.0395 - val_acc: 0.7966
Epoch 91/100
60000/60000 [==============================] - 4s 69us/step - loss: 0.0372 -
acc: 0.8004 - val_loss: 0.0398 - val_acc: 0.7948
Epoch 92/100
60000/60000 [==============================] - 4s 70us/step - loss: 0.0372 -
acc: 0.8005 - val_loss: 0.0395 - val_acc: 0.8001
Epoch 93/100
60000/60000 [==============================] - 4s 69us/step - loss: 0.0372 -
acc: 0.8005 - val_loss: 0.0393 - val_acc: 0.7966
Epoch 94/100
60000/60000 [==============================] - 4s 69us/step - loss: 0.0372 -
acc: 0.8005 - val_loss: 0.0392 - val_acc: 0.7978
Epoch 95/100
60000/60000 [==============================] - 4s 70us/step - loss: 0.0371 -
acc: 0.8005 - val_loss: 0.0394 - val_acc: 0.7954
Epoch 96/100
60000/60000 [==============================] - 4s 70us/step - loss: 0.0371 -
acc: 0.8005 - val_loss: 0.0393 - val_acc: 0.7983
Epoch 97/100
60000/60000 [==============================] - 4s 71us/step - loss: 0.0370 -
acc: 0.8006 - val_loss: 0.0396 - val_acc: 0.7964
Epoch 98/100
60000/60000 [==============================] - 4s 69us/step - loss: 0.0371 -
acc: 0.8005 - val_loss: 0.0394 - val_acc: 0.7966
Epoch 99/100
60000/60000 [==============================] - 4s 69us/step - loss: 0.0370 -
acc: 0.8006 - val_loss: 0.0395 - val_acc: 0.7973
Epoch 100/100
60000/60000 [==============================] - 4s 69us/step - loss: 0.0370 -
acc: 0.8006 - val_loss: 0.0395 - val_acc: 0.7962
```

In [57]:
```python
# plot the train and test loss
plt.plot(autoencoder_conv_history.history['loss'])
plt.plot(autoencoder_conv_history.history['val_loss'])
plt.title('Train and test loss for CAE')
plt.ylabel('Loss')
plt.xlabel('Epoch')
plt.legend(['Train', 'Test'], loc='upper left')
plt.show()
```
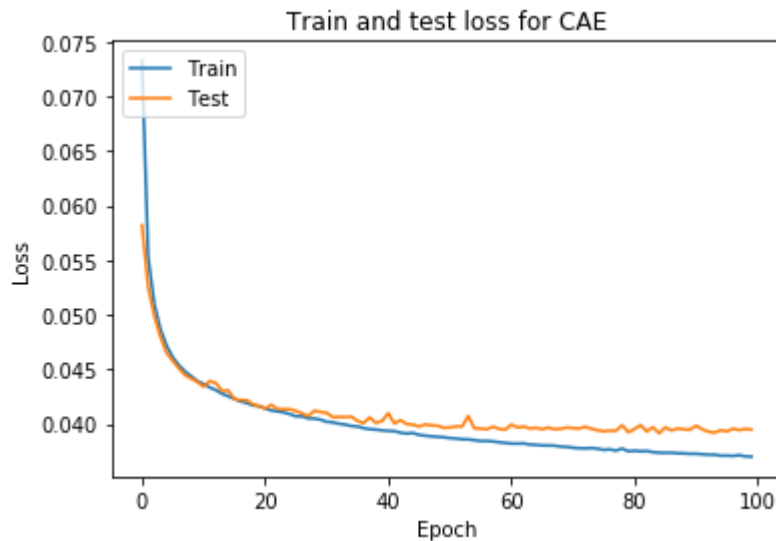

In [56]:
```python
# visualization

img_num = 10
autoencoder_conv_best = load_model('./best_model_conv.pth')
plt.figure(figsize=(18, 4))
plt.gray()
for i in range(img_num):
    # randomly pick an image from test dataset
    chosen_test_img = x_test[np.random.randint(x_test.shape[0])]
    img_decoded = autoencoder_conv_best.predict(chosen_test_img.reshape(1, 28,
28, 1))
    ax = plt.subplot(2, img_num, i+1)
    plt.imshow(chosen_test_img.reshape(28, 28))
    ax.axis('off')

    ax = plt.subplot(2, img_num, img_num+i+1)
    plt.imshow(img_decoded.reshape(28, 28))
    ax.axis('off')
plt.show()
```
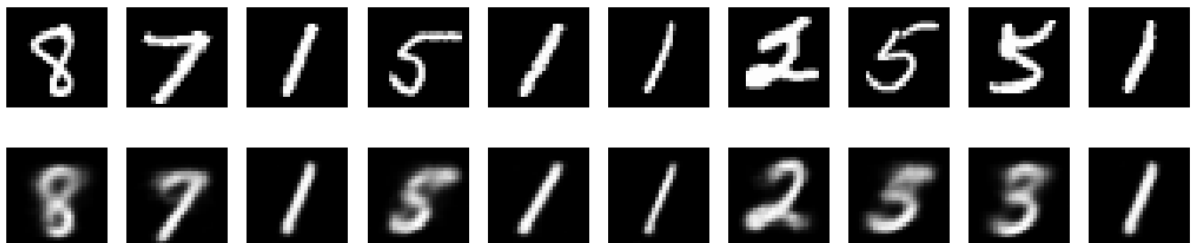
The some of the reconstructed images are not readable for human. To solve the problem, I build another model and enlarged the bottleneck to 12.

In [51]:
```python
# find a CAE architecture that is powerful enough to generate readable images

# build the network with larger bottleneck
autoencoder_conv_refine = Sequential()
# encoder network
autoencoder_conv_refine.add(Conv2D(32, (3, 3), activation='relu', padding='sam
e', input_shape=x_train.shape[1:]))
autoencoder_conv_refine.add(MaxPooling2D(pool_size=(2, 2)))
autoencoder_conv_refine.add(Conv2D(64, (3, 3), activation='relu', padding='sam
e'))
autoencoder_conv_refine.add(MaxPooling2D(pool_size=(2, 2)))
autoencoder_conv_refine.add(Conv2D(128, (3, 3), activation='relu', padding='sa
me'))
autoencoder_conv_refine.add(MaxPooling2D(pool_size=(2, 2)))
autoencoder_conv_refine.add(Conv2D(12, (3, 3), activation='relu', padding='sam
e'))
autoencoder_conv_refine.add(MaxPooling2D(pool_size=(2, 2)))
# decoder network
autoencoder_conv_refine.add(Conv2DTranspose(128, (7, 7), activation='relu', pa
dding='valid'))
autoencoder_conv_refine.add(UpSampling2D((2, 2)))
autoencoder_conv_refine.add(Conv2DTranspose(64, (3, 3), activation='relu', pad
ding='same'))
autoencoder_conv_refine.add(UpSampling2D((2, 2)))
autoencoder_conv_refine.add(Conv2D(1, (3, 3), activation='sigmoid', padding='s
ame'))

autoencoder_conv_refine.summary()
```

```
_____
Layer (type)                 Output Shape           Param #
================================================================
conv2d_6 (Conv2D)            (None, 28, 28, 32)       320
_____
max_pooling2d_5 (MaxPooling2 (None, 14, 14, 32)       0
_____
conv2d_7 (Conv2D)            (None, 14, 14, 64)       18496
_____
max_pooling2d_6 (MaxPooling2 (None, 7, 7, 64)         0
_____
conv2d_8 (Conv2D)            (None, 7, 7, 128)        73856
_____
max_pooling2d_7 (MaxPooling2 (None, 3, 3, 128)        0
_____
conv2d_9 (Conv2D)            (None, 3, 3, 12)         13836
_____
max_pooling2d_8 (MaxPooling2 (None, 1, 1, 12)         0
_____
conv2d_transpose_3 (Conv2DTr (None, 7, 7, 128)        75392
_____
up_sampling2d_3 (UpSampling2 (None, 14, 14, 128)      0
_____
conv2d_transpose_4 (Conv2DTr (None, 14, 14, 64)       73792
_____
up_sampling2d_4 (UpSampling2 (None, 28, 28, 64)       0
_____
conv2d_10 (Conv2D)           (None, 28, 28, 1)        577
================================================================
Total params: 256,269
Trainable params: 256,269
Non-trainable params: 0
_____
```

In [52]:
```python
# train model

print(device_lib.list_local_devices())

autoencoder_conv_refine.compile(optimizer='adam', loss='mean_squared_error', metrics=['accuracy'])
best_model_checkpoint = ModelCheckpoint(
    './best_model_conv_refine.pth',
    monitor="val_acc",
    save_best_only=True,
    save_weights_only=False
)
autoencoder_conv_refine_history = autoencoder_conv_refine.fit(
    x_train,
    x_train,
    epochs=20,
    batch_size=256,
    shuffle=True,
    validation_data=(x_test, x_test),
    callbacks=[best_model_checkpoint]
)
```

```
[name: "/device:CPU:0"
device_type: "CPU"
memory_limit: 268435456
locality {
}
incarnation: 14877197587185339757
, name: "/device:GPU:0"
device_type: "GPU"
memory_limit: 6700198133
locality {
  bus_id: 1
  links {
  }
}
incarnation: 13845019592877475944
physical_device_desc: "device: 0, name: GeForce GTX 1070, pci bus id: 0000:0
1:00.0, compute capability: 6.1"
]
Train on 60000 samples, validate on 10000 samples
Epoch 1/20
60000/60000 [==============================] - 7s 109us/step - loss: 0.0562 -
acc: 0.7976 - val_loss: 0.0296 - val_acc: 0.8050
Epoch 2/20
60000/60000 [==============================] - 6s 98us/step - loss: 0.0251 -
acc: 0.8061 - val_loss: 0.0214 - val_acc: 0.8072
Epoch 3/20
60000/60000 [==============================] - 6s 101us/step - loss: 0.0208 -
acc: 0.8085 - val_loss: 0.0195 - val_acc: 0.8072
Epoch 4/20
60000/60000 [==============================] - 6s 99us/step - loss: 0.0190 -
acc: 0.8095 - val_loss: 0.0180 - val_acc: 0.8084
Epoch 5/20
60000/60000 [==============================] - 6s 97us/step - loss: 0.0179 -
acc: 0.8100 - val_loss: 0.0174 - val_acc: 0.8089
Epoch 6/20
60000/60000 [==============================] - 6s 98us/step - loss: 0.0171 -
acc: 0.8104 - val_loss: 0.0165 - val_acc: 0.8094
Epoch 7/20
60000/60000 [==============================] - 6s 96us/step - loss: 0.0165 -
acc: 0.8107 - val_loss: 0.0163 - val_acc: 0.8090
Epoch 8/20
60000/60000 [==============================] - 6s 97us/step - loss: 0.0159 -
acc: 0.8110 - val_loss: 0.0157 - val_acc: 0.8095
Epoch 9/20
60000/60000 [==============================] - 6s 98us/step - loss: 0.0154 -
acc: 0.8112 - val_loss: 0.0151 - val_acc: 0.8104
Epoch 10/20
60000/60000 [==============================] - 6s 98us/step - loss: 0.0151 -
acc: 0.8114 - val_loss: 0.0148 - val_acc: 0.8102
Epoch 11/20
60000/60000 [==============================] - 6s 97us/step - loss: 0.0147 -
acc: 0.8116 - val_loss: 0.0146 - val_acc: 0.8112
Epoch 12/20
60000/60000 [==============================] - 6s 98us/step - loss: 0.0145 -
acc: 0.8117 - val_loss: 0.0143 - val_acc: 0.8110
Epoch 13/20
60000/60000 [==============================] - 6s 97us/step - loss: 0.0142 -
```

```
acc: 0.8118 - val_loss: 0.0141 - val_acc: 0.8109
Epoch 14/20
60000/60000 [==============================] - 6s 97us/step - loss: 0.0140 -
acc: 0.8119 - val_loss: 0.0140 - val_acc: 0.8113
Epoch 15/20
60000/60000 [==============================] - 6s 97us/step - loss: 0.0138 -
acc: 0.8120 - val_loss: 0.0137 - val_acc: 0.8111
Epoch 16/20
60000/60000 [==============================] - 6s 97us/step - loss: 0.0136 -
acc: 0.8121 - val_loss: 0.0140 - val_acc: 0.8116
Epoch 17/20
60000/60000 [==============================] - 6s 97us/step - loss: 0.0134 -
acc: 0.8122 - val_loss: 0.0135 - val_acc: 0.8114
Epoch 18/20
60000/60000 [==============================] - 6s 97us/step - loss: 0.0133 -
acc: 0.8123 - val_loss: 0.0133 - val_acc: 0.8113
Epoch 19/20
60000/60000 [==============================] - 6s 98us/step - loss: 0.0131 -
acc: 0.8124 - val_loss: 0.0132 - val_acc: 0.8111
Epoch 20/20
60000/60000 [==============================] - 6s 97us/step - loss: 0.0130 -
acc: 0.8124 - val_loss: 0.0130 - val_acc: 0.8112
```

In [54]:
```python
# visualization

img_num = 10
autoencoder_conv_refine_best = load_model('./best_model_conv_refine.pth')
plt.figure(figsize=(18, 4))
plt.gray()
for i in range(img_num):
    # randomly pick an image from test dataset
    chosen_test_img = x_test[np.random.randint(x_test.shape[0])]
    img_decoded = autoencoder_conv_refine_best.predict(chosen_test_img.reshape
(1, 28, 28, 1))
    ax = plt.subplot(2, img_num, i+1)
    plt.imshow(chosen_test_img.reshape(28, 28))
    ax.axis('off')

    ax = plt.subplot(2, img_num, img_num+i+1)
    plt.imshow(img_decoded.reshape(28, 28))
    ax.axis('off')
plt.show()
```

try to find a CAE architecture, including a larger bottleneck, that is powerful enough to generate readable images.

After enlarge the bottleneck from 2 to 12, the CAE is able to generate readable images as shown above

In [76]:
```
''' Part 3: using an AutoEncoder to generate similar but not identical hand di
gits
'''

# Part 3.1:
# load the model trained in part 1
autoencoder_fc_best = load_model('./best_model_fc.pth')
autoencoder_fc_best.summary()
```

```
Layer (type)                    Output Shape              Param #
=================================================================
dense_22 (Dense)                (None, 512)               401920
_____
dense_23 (Dense)                (None, 2)                 1026
_____
dense_24 (Dense)                (None, 512)               1536
_____
dense_25 (Dense)                (None, 784)               402192
=================================================================
Total params: 806,674
Trainable params: 806,674
Non-trainable params: 0
_____
```

In [77]:
```
# extract the decoder network from the original model
decoder_fc_layer1 = autoencoder_fc_best.layers[2]
decoder_fc_layer2 = autoencoder_fc_best.layers[3]
decoder_input = Input(shape=(2,))
decoder_fc = Model(decoder_input, decoder_fc_layer2(decoder_fc_layer1(decoder_
input)))
decoder_fc.summary()
```

```
Layer (type)                    Output Shape              Param #
=================================================================
input_5 (InputLayer)            (None, 2)                 0
_____
dense_24 (Dense)                (None, 512)               1536
_____
dense_25 (Dense)                (None, 784)               402192
=================================================================
Total params: 403,728
Trainable params: 403,728
Non-trainable params: 0
_____
```

In [101]:
```python
img_num = 10
plt.figure(figsize=(18, 4))
plt.gray()

# randomly generate some numbers as the input of decoder
generated_input = (np.random.rand(img_num, 2, )) * 20
print(generated_input)
generated_imgs = decoder_fc.predict(generated_input)

for i in range(img_num):
    img_decoded = generated_imgs[i]
    ax = plt.subplot(1, img_num, i+1)
    plt.imshow(img_decoded.reshape(28, 28))
    ax.axis('off')
plt.show()
```

```
[[ 0.7087463  13.67673812]
 [14.86996982 16.08176403]
 [ 3.53354096  1.45952269]
 [ 6.35220493 13.33000247]
 [15.78123443 19.65897981]
 [17.11051112 10.38312941]
 [18.26681465 17.23007229]
 [ 0.07462502 15.54017014]
 [ 1.59308199 18.56136937]
 [19.18960768  2.40478631]]
```
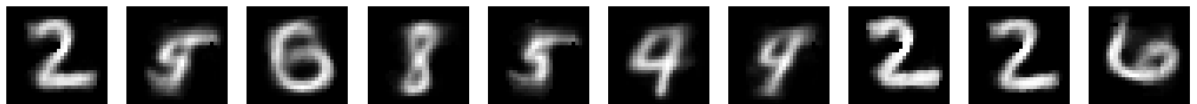
In [200]:
```python
# Part 3.2:
# restrict the AutoEncoder hidden bottleneck layer(s)
# to have a standard multi-variate normal distribution
# with mean zeroes and the identity matrix as variance
def normalize_bottleneck(args):
    norm_mean, norm_var = args
    eps = K.random_normal(shape=(K.shape(norm_mean)[0], K.int_shape(norm_mean)
[1]))
    norm = norm_mean + K.exp(0.5 * norm_var) * eps
    return norm

# flatten the input image data
input_size = 784
x_train_flatten = x_train.reshape(-1, input_size)
x_test_flatten = x_test.reshape(-1, input_size)

# construct the encoder part
autoencoder_fc_norm_input = Input(shape=(input_size, ))
intermediate_output1 = Dense(512, activation='relu')(autoencoder_fc_norm_input
)
norm_mean = Dense(2)(intermediate_output1)
norm_var = Dense(2)(intermediate_output1)
encoder_output = Lambda(normalize_bottleneck, output_shape=(2, ))([norm_mean,
norm_var])
# construct the decoder part
intermediate_output2 = Dense(512, activation='relu')(encoder_output)
decoder_output = Dense(input_size, activation='sigmoid')(intermediate_output2)

autoencoder_fc_norm = Model(autoencoder_fc_norm_input, decoder_output)
autoencoder_fc_norm.summary()
```

Layer (type)	Output Shape	Param #	Connected to
input_19 (InputLayer)	(None, 784)	0	
dense_43 (Dense) [0]	(None, 512)	401920	input_19[0]
dense_44 (Dense) [0]	(None, 2)	1026	dense_43[0]
dense_45 (Dense) [0]	(None, 2)	1026	dense_43[0]
lambda_8 (Lambda) [0] [0]	(None, 2)	0	dense_44[0] dense_45[0]
dense_46 (Dense) [0]	(None, 512)	1536	lambda_8[0]
dense_47 (Dense) [0]	(None, 784)	402192	dense_46[0]

Total params: 807,700
Trainable params: 807,700
Non-trainable params: 0

In [201]:
```python
# train model

print(device_lib.list_local_devices())

autoencoder_fc_norm.compile(optimizer='adam', loss='mean_squared_error', metrics=['accuracy'])
best_model_checkpoint = ModelCheckpoint(
    './best_model_fc_norm.pth',
    monitor="val_acc",
    save_best_only=True,
    save_weights_only=False
)
autoencoder_fc_history = autoencoder_fc_norm.fit(
    x_train_flatten,
    x_train_flatten,
    epochs=50,
    batch_size=256,
    shuffle=True,
    validation_data=(x_test_flatten, x_test_flatten),
    callbacks=[best_model_checkpoint]
)
```

```
[name: "/device:CPU:0"
device_type: "CPU"
memory_limit: 268435456
locality {
}
incarnation: 5998046289788614467
, name: "/device:GPU:0"
device_type: "GPU"
memory_limit: 6700198133
locality {
  bus_id: 1
  links {
  }
}
incarnation: 14375085748194495944
physical_device_desc: "device: 0, name: GeForce GTX 1070, pci bus id: 0000:0
1:00.0, compute capability: 6.1"
]
Train on 60000 samples, validate on 10000 samples
Epoch 1/50
60000/60000 [==============================] - 5s 82us/step - loss: 0.0705 -
acc: 0.0129 - val_loss: 0.0562 - val_acc: 0.0099
Epoch 2/50
60000/60000 [==============================] - 2s 27us/step - loss: 0.0543 -
acc: 0.0148 - val_loss: 0.0520 - val_acc: 0.0128
Epoch 3/50
60000/60000 [==============================] - 2s 27us/step - loss: 0.0513 -
acc: 0.0146 - val_loss: 0.0501 - val_acc: 0.0126
Epoch 4/50
60000/60000 [==============================] - 2s 27us/step - loss: 0.0496 -
acc: 0.0111 - val_loss: 0.0489 - val_acc: 0.0084
Epoch 5/50
60000/60000 [==============================] - 2s 27us/step - loss: 0.0482 -
acc: 0.0101 - val_loss: 0.0476 - val_acc: 0.0075
Epoch 6/50
60000/60000 [==============================] - 2s 26us/step - loss: 0.0471 -
acc: 0.0090 - val_loss: 0.0468 - val_acc: 0.0085
Epoch 7/50
60000/60000 [==============================] - 1s 25us/step - loss: 0.0462 -
acc: 0.0090 - val_loss: 0.0459 - val_acc: 0.0093
Epoch 8/50
60000/60000 [==============================] - 2s 26us/step - loss: 0.0455 -
acc: 0.0086 - val_loss: 0.0454 - val_acc: 0.0068
Epoch 9/50
60000/60000 [==============================] - 2s 25us/step - loss: 0.0450 -
acc: 0.0084 - val_loss: 0.0448 - val_acc: 0.0057
Epoch 10/50
60000/60000 [==============================] - 1s 25us/step - loss: 0.0444 -
acc: 0.0091 - val_loss: 0.0443 - val_acc: 0.0059
Epoch 11/50
60000/60000 [==============================] - 1s 24us/step - loss: 0.0439 -
acc: 0.0087 - val_loss: 0.0442 - val_acc: 0.0067
Epoch 12/50
60000/60000 [==============================] - 1s 24us/step - loss: 0.0435 -
acc: 0.0087 - val_loss: 0.0437 - val_acc: 0.0092
Epoch 13/50
60000/60000 [==============================] - 1s 24us/step - loss: 0.0432 -
```

```
                 acc: 0.0092 - val_loss: 0.0434 - val_acc: 0.0094
                 Epoch 14/50
                 60000/60000 [==============================] - 1s 24us/step - loss: 0.0428 -
                 acc: 0.0089 - val_loss: 0.0432 - val_acc: 0.0099
                 Epoch 15/50
                 60000/60000 [==============================] - 1s 24us/step - loss: 0.0425 -
                 acc: 0.0086 - val_loss: 0.0429 - val_acc: 0.0068
                 Epoch 16/50
                 60000/60000 [==============================] - 1s 25us/step - loss: 0.0422 -
                 acc: 0.0094 - val_loss: 0.0426 - val_acc: 0.0092
                 Epoch 17/50
                 60000/60000 [==============================] - 2s 26us/step - loss: 0.0420 -
                 acc: 0.0092 - val_loss: 0.0425 - val_acc: 0.0088
                 Epoch 18/50
                 60000/60000 [==============================] - 2s 26us/step - loss: 0.0418 -
                 acc: 0.0102 - val_loss: 0.0423 - val_acc: 0.0082
                 Epoch 19/50
                 60000/60000 [==============================] - 2s 25us/step - loss: 0.0416 -
                 acc: 0.0096 - val_loss: 0.0421 - val_acc: 0.0104
                 Epoch 20/50
                 60000/60000 [==============================] - 2s 26us/step - loss: 0.0414 -
                 acc: 0.0102 - val_loss: 0.0420 - val_acc: 0.0066
                 Epoch 21/50
                 60000/60000 [==============================] - 2s 26us/step - loss: 0.0412 -
                 acc: 0.0100 - val_loss: 0.0418 - val_acc: 0.0100
                 Epoch 22/50
                 60000/60000 [==============================] - 2s 25us/step - loss: 0.0410 -
                 acc: 0.0104 - val_loss: 0.0419 - val_acc: 0.0071
                 Epoch 23/50
                 60000/60000 [==============================] - 1s 25us/step - loss: 0.0409 -
                 acc: 0.0106 - val_loss: 0.0416 - val_acc: 0.0109
                 Epoch 24/50
                 60000/60000 [==============================] - 1s 25us/step - loss: 0.0407 -
                 acc: 0.0100 - val_loss: 0.0414 - val_acc: 0.0137
                 Epoch 25/50
                 60000/60000 [==============================] - 2s 25us/step - loss: 0.0405 -
                 acc: 0.0101 - val_loss: 0.0415 - val_acc: 0.0137
                 Epoch 26/50
                 60000/60000 [==============================] - 2s 26us/step - loss: 0.0404 -
                 acc: 0.0103 - val_loss: 0.0413 - val_acc: 0.0137
                 Epoch 27/50
                 60000/60000 [==============================] - 2s 26us/step - loss: 0.0403 -
                 acc: 0.0112 - val_loss: 0.0412 - val_acc: 0.0116
                 Epoch 28/50
                 60000/60000 [==============================] - 2s 26us/step - loss: 0.0402 -
                 acc: 0.0109 - val_loss: 0.0411 - val_acc: 0.0088
                 Epoch 29/50
                 60000/60000 [==============================] - 2s 25us/step - loss: 0.0400 -
                 acc: 0.0114 - val_loss: 0.0411 - val_acc: 0.0109
                 Epoch 30/50
                 60000/60000 [==============================] - 2s 26us/step - loss: 0.0399 -
                 acc: 0.0109 - val_loss: 0.0411 - val_acc: 0.0068
                 Epoch 31/50
                 60000/60000 [==============================] - 2s 25us/step - loss: 0.0398 -
                 acc: 0.0113 - val_loss: 0.0410 - val_acc: 0.0129
                 Epoch 32/50
                 60000/60000 [==============================] - 1s 25us/step - loss: 0.0397 -
```

```
                      acc: 0.0116 - val_loss: 0.0410 - val_acc: 0.0075
                      Epoch 33/50
                      60000/60000 [==============================] - 1s 25us/step - loss: 0.0396 -
                      acc: 0.0112 - val_loss: 0.0409 - val_acc: 0.0111
                      Epoch 34/50
                      60000/60000 [==============================] - 1s 25us/step - loss: 0.0395 -
                      acc: 0.0117 - val_loss: 0.0409 - val_acc: 0.0097
                      Epoch 35/50
                      60000/60000 [==============================] - 1s 25us/step - loss: 0.0394 -
                      acc: 0.0123 - val_loss: 0.0408 - val_acc: 0.0082
                      Epoch 36/50
                      60000/60000 [==============================] - 2s 25us/step - loss: 0.0393 -
                      acc: 0.0115 - val_loss: 0.0408 - val_acc: 0.0078
                      Epoch 37/50
                      60000/60000 [==============================] - 1s 25us/step - loss: 0.0392 -
                      acc: 0.0125 - val_loss: 0.0407 - val_acc: 0.0118
                      Epoch 38/50
                      60000/60000 [==============================] - 1s 25us/step - loss: 0.0391 -
                      acc: 0.0124 - val_loss: 0.0407 - val_acc: 0.0104
                      Epoch 39/50
                      60000/60000 [==============================] - 2s 26us/step - loss: 0.0390 -
                      acc: 0.0127 - val_loss: 0.0406 - val_acc: 0.0127
                      Epoch 40/50
                      60000/60000 [==============================] - 2s 26us/step - loss: 0.0390 -
                      acc: 0.0126 - val_loss: 0.0405 - val_acc: 0.0096
                      Epoch 41/50
                      60000/60000 [==============================] - 2s 26us/step - loss: 0.0389 -
                      acc: 0.0120 - val_loss: 0.0405 - val_acc: 0.0124
                      Epoch 42/50
                      60000/60000 [==============================] - 1s 25us/step - loss: 0.0388 -
                      acc: 0.0129 - val_loss: 0.0405 - val_acc: 0.0065
                      Epoch 43/50
                      60000/60000 [==============================] - 1s 24us/step - loss: 0.0387 -
                      acc: 0.0132 - val_loss: 0.0406 - val_acc: 0.0121
                      Epoch 44/50
                      60000/60000 [==============================] - 1s 25us/step - loss: 0.0387 -
                      acc: 0.0134 - val_loss: 0.0404 - val_acc: 0.0133
                      Epoch 45/50
                      60000/60000 [==============================] - 1s 25us/step - loss: 0.0386 -
                      acc: 0.0132 - val_loss: 0.0404 - val_acc: 0.0131
                      Epoch 46/50
                      60000/60000 [==============================] - 1s 25us/step - loss: 0.0385 -
                      acc: 0.0130 - val_loss: 0.0404 - val_acc: 0.0088
                      Epoch 47/50
                      60000/60000 [==============================] - 1s 25us/step - loss: 0.0384 -
                      acc: 0.0134 - val_loss: 0.0404 - val_acc: 0.0127
                      Epoch 48/50
                      60000/60000 [==============================] - 1s 25us/step - loss: 0.0384 -
                      acc: 0.0131 - val_loss: 0.0403 - val_acc: 0.0138
                      Epoch 49/50
                      60000/60000 [==============================] - 1s 25us/step - loss: 0.0383 -
                      acc: 0.0134 - val_loss: 0.0403 - val_acc: 0.0140
                      Epoch 50/50
                      60000/60000 [==============================] - 1s 25us/step - loss: 0.0382 -
                      acc: 0.0134 - val_loss: 0.0402 - val_acc: 0.0137
```

In [3]:
```python
# load the best model with normalized bottleneck
autoencoder_fc_norm_best = load_model('./best_model_fc_norm.pth')

# extract the decoder network from the original model
decoder_fc_norm_layer1 = autoencoder_fc_norm_best.layers[5]
decoder_fc_norm_layer2 = autoencoder_fc_norm_best.layers[6]
decoder_norm_input = Input(shape=(2,))
decoder_fc_norm = Model(decoder_norm_input, decoder_fc_norm_layer2(decoder_fc__
norm_layer1(decoder_norm_input)))
decoder_fc_norm.summary()
```

```
WARNING:tensorflow:From C:\Users\songyih\AppData\Roaming\Python\Python37\site
-packages\tensorflow\python\framework\op_def_library.py:263: colocate_with (f
rom tensorflow.python.framework.ops) is deprecated and will be removed in a f
uture version.
Instructions for updating:
Colocations handled automatically by placer.
WARNING:tensorflow:From C:\Users\songyih\AppData\Roaming\Python\Python37\site
-packages\tensorflow\python\ops\math_ops.py:3066: to_int32 (from tensorflow.p
ython.ops.math_ops) is deprecated and will be removed in a future version.
Instructions for updating:
Use tf.cast instead.
```

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	(None, 2)	0
dense_46 (Dense)	(None, 512)	1536
dense_47 (Dense)	(None, 784)	402192

```
Total params: 403,728
Trainable params: 403,728
Non-trainable params: 0
```

In [102]:
```python
# generate images by inputing random number to decoder
print("Generate images by inputing different var values draw from the distribu
tion")
img_num = 10
plt.figure(figsize=(18, 18))
plt.gray()

random_vars = np.linspace(-4, 4, img_num)[::-1]

for i in range(len(random_vars)):
    z_sample = np.array([[0, random_vars[i]]])
    img_decoded = decoder_fc_norm.predict(z_sample)
    ax = plt.subplot(1, img_num, i+1)
    plt.imshow(img_decoded.reshape(28, 28))
    ax.axis('off')
plt.show()
```

Generate images by inputing different var values draw from the distribution
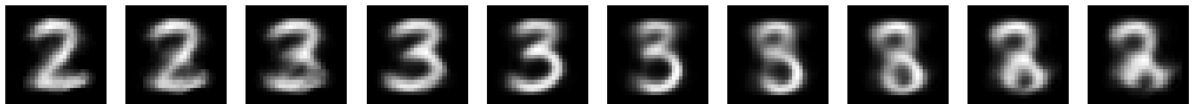
In [105]:
```python
# generate images by inputing random number to decoder
print("Generate images by inputing different mean values draw from the distrib
ution")
random_means = np.linspace(-3, 3, img_num)
plt.figure(figsize=(18, 18))
plt.gray()
for i in range(len(random_means)):
    z_sample = np.array([[random_means[i], 1]])
    img_decoded = decoder_fc_norm.predict(z_sample)
    ax = plt.subplot(1, img_num, i+1)
    plt.imshow(img_decoded.reshape(28, 28))
    ax.axis('off')
plt.show()
```

Generate images by inputing different mean values draw from the distribution

Part 3.3: Are the output images different between 1) and 2)? If so, why do you think this difference occurs?

The output images are different between 1) and 2). In the output image from part 3.2 we can clearly see how the generated digits gradually shifting from one to another, and the reconstructed images is not simplying imitating the original dataset, but creating new styles. I think this is because the encoder we build in part 3.2 is not just generating encoding information for a single image embedding, but an embedding that is shifted from orignal embedding.

In [30]:
```python
''' Part 4: Optional: change the AutoEncoder which you developed in the
last part of section 3 so that it becomes a Variational AutoEncoder
'''

# the network structure is a copy of part 3.3, the different part is at the lo
ss function

def normalize_bottleneck(args):
    norm_mean, norm_var = args
    eps = K.random_normal(shape=(K.shape(norm_mean)[0], K.int_shape(norm_mean)
[1]))
    norm = norm_mean + K.exp(0.5 * norm_var) * eps
    return norm

# flatten the input image data
input_size = 784
x_train_flatten = x_train.reshape(-1, input_size)
x_test_flatten = x_test.reshape(-1, input_size)

# the network is the same as in section 3
# construct the encoder part
vae_input = Input(shape=(input_size, ))
intermediate_output1 = Dense(512, activation='relu')(vae_input)
norm_mean = Dense(2)(intermediate_output1)
norm_var = Dense(2)(intermediate_output1)
encoder_output = Lambda(normalize_bottleneck, output_shape=(2, ))([norm_mean,
norm_var])
# construct the decoder part
intermediate_output2 = Dense(512, activation='relu')(encoder_output)
decoder_output = Dense(input_size, activation='sigmoid')(intermediate_output2)

variational_autoencoder = Model(vae_input, decoder_output)
variational_autoencoder.summary()
```

Layer (type)	Output Shape	Param #	Connected to
input_2 (InputLayer)	(None, 784)	0	
dense_1 (Dense) [0]	(None, 512)	401920	input_2[0]
dense_2 (Dense) [0]	(None, 2)	1026	dense_1[0]
dense_3 (Dense) [0]	(None, 2)	1026	dense_1[0]
lambda_1 (Lambda) [0] [0]	(None, 2)	0	dense_2[0] dense_3[0]
dense_4 (Dense) [0]	(None, 512)	1536	lambda_1[0]
dense_5 (Dense) [0]	(None, 784)	402192	dense_4[0]

```
Total params: 807,700
Trainable params: 807,700
Non-trainable params: 0
```

In [31]:
```python
# make the losses for VAE
from keras.losses import mse

# reconstruction loss
mse_loss = mse(vae_input, decoder_output) * input_size

# variational loss
variational_loss = -1 * K.sum((1 + norm_var - K.square(norm_mean) - K.exp(norm
_var)) / 2, axis=-1)

# combine the reconstruction loss and variational loss to make the total loss
variational_autoencoder.add_loss(K.mean(mse_loss + variational_loss))
```

In [33]:
```python
# train model

print(device_lib.list_local_devices())

variational_autoencoder.compile(optimizer='adam')
best_model_checkpoint = ModelCheckpoint(
    './best_model_vae_weights.pth',
    monitor="val_loss",
    save_best_only=True,
    save_weights_only=True
)
vae_history = variational_autoencoder.fit(
    x_train_flatten,
    None,
    epochs=200,
    batch_size=256,
    shuffle=True,
    validation_data=(x_test_flatten, None),
    callbacks=[best_model_checkpoint]
)
```

```
[name: "/device:CPU:0"
device_type: "CPU"
memory_limit: 268435456
locality {
}
incarnation: 601061563355615771
, name: "/device:GPU:0"
device_type: "GPU"
memory_limit: 6700198133
locality {
  bus_id: 1
  links {
  }
}
incarnation: 10085529536070215849
physical_device_desc: "device: 0, name: GeForce GTX 1070, pci bus id: 0000:0
1:00.0, compute capability: 6.1"
]
Train on 60000 samples, validate on 10000 samples
Epoch 1/200
60000/60000 [==============================] - 1s 20us/step - loss: 36.9952 -
val_loss: 37.8383
Epoch 2/200
60000/60000 [==============================] - 1s 13us/step - loss: 36.9169 -
val_loss: 37.6753
Epoch 3/200
60000/60000 [==============================] - 1s 13us/step - loss: 36.8884 -
val_loss: 37.6780
Epoch 4/200
60000/60000 [==============================] - 1s 14us/step - loss: 36.8542 -
val_loss: 37.6410
Epoch 5/200
60000/60000 [==============================] - 1s 13us/step - loss: 36.8374 -
val_loss: 37.6742
Epoch 6/200
60000/60000 [==============================] - 1s 13us/step - loss: 36.8007 -
val_loss: 37.6107
Epoch 7/200
60000/60000 [==============================] - 1s 14us/step - loss: 36.7830 -
val_loss: 37.6381
Epoch 8/200
60000/60000 [==============================] - 1s 13us/step - loss: 36.7391 -
val_loss: 37.6143
Epoch 9/200
60000/60000 [==============================] - 1s 13us/step - loss: 36.7035 -
val_loss: 37.6207
Epoch 10/200
60000/60000 [==============================] - 1s 13us/step - loss: 36.6921 -
val_loss: 37.5911
Epoch 11/200
60000/60000 [==============================] - 1s 13us/step - loss: 36.6466 -
val_loss: 37.5805
Epoch 12/200
60000/60000 [==============================] - 1s 13us/step - loss: 36.6287 -
val_loss: 37.6085
Epoch 13/200
60000/60000 [==============================] - 1s 13us/step - loss: 36.6082 -
```

```
                val_loss: 37.4984
                Epoch 14/200
                60000/60000 [==============================] - 1s 14us/step - loss: 36.5781 -
                val_loss: 37.5257
                Epoch 15/200
                60000/60000 [==============================] - 1s 13us/step - loss: 36.5691 -
                val_loss: 37.5432
                Epoch 16/200
                60000/60000 [==============================] - 1s 13us/step - loss: 36.5426 -
                val_loss: 37.6034
                Epoch 17/200
                60000/60000 [==============================] - 1s 13us/step - loss: 36.5053 -
                val_loss: 37.6021
                Epoch 18/200
                60000/60000 [==============================] - 1s 13us/step - loss: 36.4748 -
                val_loss: 37.5333
                Epoch 19/200
                60000/60000 [==============================] - 1s 13us/step - loss: 36.4801 -
                val_loss: 37.4724
                Epoch 20/200
                60000/60000 [==============================] - 1s 13us/step - loss: 36.4459 -
                val_loss: 37.5506
                Epoch 21/200
                60000/60000 [==============================] - 1s 14us/step - loss: 36.4246 -
                val_loss: 37.4551
                Epoch 22/200
                60000/60000 [==============================] - 1s 13us/step - loss: 36.3860 -
                val_loss: 37.4665
                Epoch 23/200
                60000/60000 [==============================] - 1s 13us/step - loss: 36.3644 -
                val_loss: 37.4245
                Epoch 24/200
                60000/60000 [==============================] - 1s 13us/step - loss: 36.3708 -
                val_loss: 37.4610
                Epoch 25/200
                60000/60000 [==============================] - 1s 13us/step - loss: 36.3357 -
                val_loss: 37.4702
                Epoch 26/200
                60000/60000 [==============================] - 1s 13us/step - loss: 36.3282 -
                val_loss: 37.4650
                Epoch 27/200
                60000/60000 [==============================] - 1s 13us/step - loss: 36.2967 -
                val_loss: 37.5081
                Epoch 28/200
                60000/60000 [==============================] - 1s 13us/step - loss: 36.2638 -
                val_loss: 37.5185
                Epoch 29/200
                60000/60000 [==============================] - 1s 13us/step - loss: 36.2556 -
                val_loss: 37.4402
                Epoch 30/200
                60000/60000 [==============================] - 1s 14us/step - loss: 36.2393 -
                val_loss: 37.4399
                Epoch 31/200
                60000/60000 [==============================] - 1s 14us/step - loss: 36.2189 -
                val_loss: 37.4493
                Epoch 32/200
                60000/60000 [==============================] - 1s 13us/step - loss: 36.2089 -
```

```
                    val_loss: 37.4030
                    Epoch 33/200
                    60000/60000 [==============================] - 1s 13us/step - loss: 36.2029 -
                    val_loss: 37.4180
                    Epoch 34/200
                    60000/60000 [==============================] - 1s 13us/step - loss: 36.1933 -
                    val_loss: 37.4648
                    Epoch 35/200
                    60000/60000 [==============================] - 1s 13us/step - loss: 36.1450 -
                    val_loss: 37.3682
                    Epoch 36/200
                    60000/60000 [==============================] - 1s 13us/step - loss: 36.1310 -
                    val_loss: 37.3750
                    Epoch 37/200
                    60000/60000 [==============================] - 1s 13us/step - loss: 36.1170 -
                    val_loss: 37.3665
                    Epoch 38/200
                    60000/60000 [==============================] - 1s 13us/step - loss: 36.1044 -
                    val_loss: 37.4114
                    Epoch 39/200
                    60000/60000 [==============================] - 1s 13us/step - loss: 36.0741 -
                    val_loss: 37.3864
                    Epoch 40/200
                    60000/60000 [==============================] - 1s 13us/step - loss: 36.0625 -
                    val_loss: 37.3510
                    Epoch 41/200
                    60000/60000 [==============================] - 1s 13us/step - loss: 36.0420 -
                    val_loss: 37.2938
                    Epoch 42/200
                    60000/60000 [==============================] - 1s 13us/step - loss: 36.0579 -
                    val_loss: 37.3987
                    Epoch 43/200
                    60000/60000 [==============================] - 1s 13us/step - loss: 36.0229 -
                    val_loss: 37.3851
                    Epoch 44/200
                    60000/60000 [==============================] - 1s 13us/step - loss: 36.0207 -
                    val_loss: 37.3757
                    Epoch 45/200
                    60000/60000 [==============================] - 1s 13us/step - loss: 35.9887 -
                    val_loss: 37.3486
                    Epoch 46/200
                    60000/60000 [==============================] - 1s 13us/step - loss: 35.9710 -
                    val_loss: 37.3606
                    Epoch 47/200
                    60000/60000 [==============================] - 1s 13us/step - loss: 35.9669 -
                    val_loss: 37.3607
                    Epoch 48/200
                    60000/60000 [==============================] - 1s 13us/step - loss: 35.9458 -
                    val_loss: 37.4169
                    Epoch 49/200
                    60000/60000 [==============================] - 1s 13us/step - loss: 35.9243 -
                    val_loss: 37.3572
                    Epoch 50/200
                    60000/60000 [==============================] - 1s 13us/step - loss: 35.9402 -
                    val_loss: 37.4062
                    Epoch 51/200
                    60000/60000 [==============================] - 1s 13us/step - loss: 35.9274 -
```

```
                val_loss: 37.4800
                Epoch 52/200
                60000/60000 [==============================] - 1s 13us/step - loss: 35.9028 -
                val_loss: 37.2688
                Epoch 53/200
                60000/60000 [==============================] - 1s 13us/step - loss: 35.8827 -
                val_loss: 37.3620
                Epoch 54/200
                60000/60000 [==============================] - 1s 13us/step - loss: 35.8693 -
                val_loss: 37.3269
                Epoch 55/200
                60000/60000 [==============================] - 1s 13us/step - loss: 35.8654 -
                val_loss: 37.4246
                Epoch 56/200
                60000/60000 [==============================] - 1s 13us/step - loss: 35.8555 -
                val_loss: 37.3315
                Epoch 57/200
                60000/60000 [==============================] - 1s 13us/step - loss: 35.8347 -
                val_loss: 37.3694
                Epoch 58/200
                60000/60000 [==============================] - 1s 13us/step - loss: 35.8240 -
                val_loss: 37.3218
                Epoch 59/200
                60000/60000 [==============================] - 1s 13us/step - loss: 35.8015 -
                val_loss: 37.3415
                Epoch 60/200
                60000/60000 [==============================] - 1s 13us/step - loss: 35.8064 -
                val_loss: 37.3676
                Epoch 61/200
                60000/60000 [==============================] - 1s 13us/step - loss: 35.7843 -
                val_loss: 37.3181
                Epoch 62/200
                60000/60000 [==============================] - 1s 13us/step - loss: 35.7646 -
                val_loss: 37.2809
                Epoch 63/200
                60000/60000 [==============================] - 1s 13us/step - loss: 35.7812 -
                val_loss: 37.3449
                Epoch 64/200
                60000/60000 [==============================] - 1s 13us/step - loss: 35.7490 -
                val_loss: 37.2400
                Epoch 65/200
                60000/60000 [==============================] - 1s 13us/step - loss: 35.7331 -
                val_loss: 37.3666
                Epoch 66/200
                60000/60000 [==============================] - 1s 13us/step - loss: 35.7238 -
                val_loss: 37.3437
                Epoch 67/200
                60000/60000 [==============================] - 1s 13us/step - loss: 35.7316 -
                val_loss: 37.3185
                Epoch 68/200
                60000/60000 [==============================] - 1s 14us/step - loss: 35.6913 -
                val_loss: 37.2665
                Epoch 69/200
                60000/60000 [==============================] - 1s 14us/step - loss: 35.6897 -
                val_loss: 37.3279
                Epoch 70/200
                60000/60000 [==============================] - 1s 13us/step - loss: 35.6802 -
```

```
                    val_loss: 37.3781
                    Epoch 71/200
                    60000/60000 [==============================] - 1s 13us/step - loss: 35.6535 -
                    val_loss: 37.2465
                    Epoch 72/200
                    60000/60000 [==============================] - 1s 13us/step - loss: 35.6538 -
                    val_loss: 37.1988
                    Epoch 73/200
                    60000/60000 [==============================] - 1s 13us/step - loss: 35.6569 -
                    val_loss: 37.3542
                    Epoch 74/200
                    60000/60000 [==============================] - 1s 13us/step - loss: 35.6145 -
                    val_loss: 37.1547
                    Epoch 75/200
                    60000/60000 [==============================] - 1s 13us/step - loss: 35.6286 -
                    val_loss: 37.2323
                    Epoch 76/200
                    60000/60000 [==============================] - 1s 13us/step - loss: 35.6195 -
                    val_loss: 37.2157
                    Epoch 77/200
                    60000/60000 [==============================] - 1s 13us/step - loss: 35.5989 -
                    val_loss: 37.3590
                    Epoch 78/200
                    60000/60000 [==============================] - 1s 13us/step - loss: 35.5941 -
                    val_loss: 37.2881
                    Epoch 79/200
                    60000/60000 [==============================] - 1s 13us/step - loss: 35.5663 -
                    val_loss: 37.2350
                    Epoch 80/200
                    60000/60000 [==============================] - 1s 13us/step - loss: 35.5701 -
                    val_loss: 37.1743
                    Epoch 81/200
                    60000/60000 [==============================] - 1s 13us/step - loss: 35.5463 -
                    val_loss: 37.2264
                    Epoch 82/200
                    60000/60000 [==============================] - 1s 13us/step - loss: 35.5518 -
                    val_loss: 37.3206
                    Epoch 83/200
                    60000/60000 [==============================] - 1s 14us/step - loss: 35.5258 -
                    val_loss: 37.2497
                    Epoch 84/200
                    60000/60000 [==============================] - 1s 14us/step - loss: 35.5249 -
                    val_loss: 37.3483
                    Epoch 85/200
                    60000/60000 [==============================] - 1s 13us/step - loss: 35.5361 -
                    val_loss: 37.3037
                    Epoch 86/200
                    60000/60000 [==============================] - 1s 13us/step - loss: 35.5092 -
                    val_loss: 37.3865
                    Epoch 87/200
                    60000/60000 [==============================] - 1s 14us/step - loss: 35.5036 -
                    val_loss: 37.2918
                    Epoch 88/200
                    60000/60000 [==============================] - 1s 13us/step - loss: 35.4845 -
                    val_loss: 37.1966
                    Epoch 89/200
                    60000/60000 [==============================] - 1s 14us/step - loss: 35.4678 -
```

```
val_loss: 37.2311
Epoch 90/200
60000/60000 [==============================] - 1s 14us/step - loss: 35.4398 -
val_loss: 37.2437
Epoch 91/200
60000/60000 [==============================] - 1s 13us/step - loss: 35.4643 -
val_loss: 37.1710
Epoch 92/200
60000/60000 [==============================] - 1s 13us/step - loss: 35.4633 -
val_loss: 37.2677
Epoch 93/200
60000/60000 [==============================] - 1s 13us/step - loss: 35.4379 -
val_loss: 37.1997
Epoch 94/200
60000/60000 [==============================] - 1s 14us/step - loss: 35.4168 -
val_loss: 37.1976
Epoch 95/200
60000/60000 [==============================] - 1s 13us/step - loss: 35.4283 -
val_loss: 37.2507
Epoch 96/200
60000/60000 [==============================] - 1s 13us/step - loss: 35.4167 -
val_loss: 37.1987
Epoch 97/200
60000/60000 [==============================] - 1s 13us/step - loss: 35.4113 -
val_loss: 37.3720
Epoch 98/200
60000/60000 [==============================] - 1s 13us/step - loss: 35.3899 -
val_loss: 37.2006
Epoch 99/200
60000/60000 [==============================] - 1s 13us/step - loss: 35.3808 -
val_loss: 37.2104
Epoch 100/200
60000/60000 [==============================] - 1s 13us/step - loss: 35.3844 -
val_loss: 37.2518
Epoch 101/200
60000/60000 [==============================] - 1s 14us/step - loss: 35.3649 -
val_loss: 37.2221
Epoch 102/200
60000/60000 [==============================] - 1s 14us/step - loss: 35.3489 -
val_loss: 37.2647
Epoch 103/200
60000/60000 [==============================] - 1s 14us/step - loss: 35.3582 -
val_loss: 37.1926
Epoch 104/200
60000/60000 [==============================] - 1s 14us/step - loss: 35.3356 -
val_loss: 37.3029
Epoch 105/200
60000/60000 [==============================] - 1s 13us/step - loss: 35.3275 -
val_loss: 37.1519
Epoch 106/200
60000/60000 [==============================] - 1s 14us/step - loss: 35.3371 -
val_loss: 37.2554
Epoch 107/200
60000/60000 [==============================] - 1s 14us/step - loss: 35.3193 -
val_loss: 37.2947
Epoch 108/200
60000/60000 [==============================] - 1s 14us/step - loss: 35.2957 -
```

```
val_loss: 37.1770
Epoch 109/200
60000/60000 [==============================] - 1s 14us/step - loss: 35.2929 -
val_loss: 37.1822
Epoch 110/200
60000/60000 [==============================] - 1s 13us/step - loss: 35.2917 -
val_loss: 37.2941
Epoch 111/200
60000/60000 [==============================] - 1s 13us/step - loss: 35.2995 -
val_loss: 37.2422
Epoch 112/200
60000/60000 [==============================] - 1s 14us/step - loss: 35.2918 -
val_loss: 37.3082
Epoch 113/200
60000/60000 [==============================] - 1s 14us/step - loss: 35.2819 -
val_loss: 37.2484
Epoch 114/200
60000/60000 [==============================] - 1s 13us/step - loss: 35.2685 -
val_loss: 37.3603
Epoch 115/200
60000/60000 [==============================] - 1s 13us/step - loss: 35.2827 -
val_loss: 37.2066
Epoch 116/200
60000/60000 [==============================] - 1s 13us/step - loss: 35.2381 -
val_loss: 37.1642
Epoch 117/200
60000/60000 [==============================] - 1s 13us/step - loss: 35.2449 -
val_loss: 37.1738
Epoch 118/200
60000/60000 [==============================] - 1s 13us/step - loss: 35.2319 -
val_loss: 37.2021
Epoch 119/200
60000/60000 [==============================] - 1s 13us/step - loss: 35.2098 -
val_loss: 37.2116
Epoch 120/200
60000/60000 [==============================] - 1s 13us/step - loss: 35.2094 -
val_loss: 37.1892
Epoch 121/200
60000/60000 [==============================] - 1s 13us/step - loss: 35.2295 -
val_loss: 37.2024
Epoch 122/200
60000/60000 [==============================] - 1s 13us/step - loss: 35.2040 -
val_loss: 37.2288
Epoch 123/200
60000/60000 [==============================] - 1s 13us/step - loss: 35.2009 -
val_loss: 37.2448
Epoch 124/200
60000/60000 [==============================] - 1s 14us/step - loss: 35.1904 -
val_loss: 37.2295
Epoch 125/200
60000/60000 [==============================] - 1s 14us/step - loss: 35.1797 -
val_loss: 37.1070
Epoch 126/200
60000/60000 [==============================] - 1s 13us/step - loss: 35.1598 -
val_loss: 37.2343
Epoch 127/200
60000/60000 [==============================] - 1s 14us/step - loss: 35.1530 -
```

```
val_loss: 37.2558
Epoch 128/200
60000/60000 [==============================] - 1s 14us/step - loss: 35.1626 -
val_loss: 37.1623
Epoch 129/200
60000/60000 [==============================] - 1s 13us/step - loss: 35.1526 -
val_loss: 37.1796
Epoch 130/200
60000/60000 [==============================] - 1s 13us/step - loss: 35.1585 -
val_loss: 37.1577
Epoch 131/200
60000/60000 [==============================] - 1s 13us/step - loss: 35.1409 -
val_loss: 37.1362
Epoch 132/200
60000/60000 [==============================] - 1s 13us/step - loss: 35.1416 -
val_loss: 37.2200
Epoch 133/200
60000/60000 [==============================] - 1s 13us/step - loss: 35.1342 -
val_loss: 37.1798
Epoch 134/200
60000/60000 [==============================] - 1s 13us/step - loss: 35.1424 -
val_loss: 37.1684
Epoch 135/200
60000/60000 [==============================] - 1s 13us/step - loss: 35.1064 -
val_loss: 37.1960
Epoch 136/200
60000/60000 [==============================] - 1s 13us/step - loss: 35.1127 -
val_loss: 37.1992
Epoch 137/200
60000/60000 [==============================] - 1s 13us/step - loss: 35.0893 -
val_loss: 37.1657
Epoch 138/200
60000/60000 [==============================] - 1s 13us/step - loss: 35.0891 -
val_loss: 37.1862
Epoch 139/200
60000/60000 [==============================] - 1s 13us/step - loss: 35.0820 -
val_loss: 37.1704
Epoch 140/200
60000/60000 [==============================] - 1s 13us/step - loss: 35.0716 -
val_loss: 37.1153
Epoch 141/200
60000/60000 [==============================] - 1s 13us/step - loss: 35.0770 -
val_loss: 37.2717
Epoch 142/200
60000/60000 [==============================] - 1s 13us/step - loss: 35.0770 -
val_loss: 37.1966
Epoch 143/200
60000/60000 [==============================] - 1s 13us/step - loss: 35.0731 -
val_loss: 37.1733
Epoch 144/200
60000/60000 [==============================] - 1s 13us/step - loss: 35.0455 -
val_loss: 37.1020
Epoch 145/200
60000/60000 [==============================] - 1s 13us/step - loss: 35.0365 -
val_loss: 37.1848
Epoch 146/200
60000/60000 [==============================] - 1s 14us/step - loss: 35.0548 -
```

```
                        val_loss: 37.2207
                        Epoch 147/200
                        60000/60000 [==============================] - 1s 13us/step - loss: 35.0496 -
                        val_loss: 37.1514
                        Epoch 148/200
                        60000/60000 [==============================] - 1s 13us/step - loss: 35.0224 -
                        val_loss: 37.1320
                        Epoch 149/200
                        60000/60000 [==============================] - 1s 13us/step - loss: 35.0139 -
                        val_loss: 37.2379
                        Epoch 150/200
                        60000/60000 [==============================] - 1s 13us/step - loss: 35.0092 -
                        val_loss: 37.1663
                        Epoch 151/200
                        60000/60000 [==============================] - 1s 13us/step - loss: 35.0133 -
                        val_loss: 37.1807
                        Epoch 152/200
                        60000/60000 [==============================] - 1s 13us/step - loss: 35.0004 -
                        val_loss: 37.2297
                        Epoch 153/200
                        60000/60000 [==============================] - 1s 13us/step - loss: 34.9995 -
                        val_loss: 37.1857
                        Epoch 154/200
                        60000/60000 [==============================] - 1s 13us/step - loss: 34.9926 -
                        val_loss: 37.2727
                        Epoch 155/200
                        60000/60000 [==============================] - 1s 13us/step - loss: 34.9701 -
                        val_loss: 37.1845
                        Epoch 156/200
                        60000/60000 [==============================] - 1s 13us/step - loss: 34.9944 -
                        val_loss: 37.1764
                        Epoch 157/200
                        60000/60000 [==============================] - 1s 13us/step - loss: 34.9690 -
                        val_loss: 37.1127
                        Epoch 158/200
                        60000/60000 [==============================] - 1s 13us/step - loss: 34.9538 -
                        val_loss: 37.2683
                        Epoch 159/200
                        60000/60000 [==============================] - 1s 13us/step - loss: 34.9541 -
                        val_loss: 37.1843
                        Epoch 160/200
                        60000/60000 [==============================] - 1s 13us/step - loss: 34.9441 -
                        val_loss: 37.1824
                        Epoch 161/200
                        60000/60000 [==============================] - 1s 13us/step - loss: 34.9647 -
                        val_loss: 37.1457
                        Epoch 162/200
                        60000/60000 [==============================] - 1s 13us/step - loss: 34.9670 -
                        val_loss: 37.1996
                        Epoch 163/200
                        60000/60000 [==============================] - 1s 13us/step - loss: 34.9269 -
                        val_loss: 37.2346
                        Epoch 164/200
                        60000/60000 [==============================] - 1s 13us/step - loss: 34.9673 -
                        val_loss: 37.2294
                        Epoch 165/200
                        60000/60000 [==============================] - 1s 13us/step - loss: 34.9083 -
```

```
                val_loss: 37.2393
                Epoch 166/200
                60000/60000 [==============================] - 1s 13us/step - loss: 34.9223 -
                val_loss: 37.1361
                Epoch 167/200
                60000/60000 [==============================] - 1s 13us/step - loss: 34.8989 -
                val_loss: 37.1900
                Epoch 168/200
                60000/60000 [==============================] - 1s 13us/step - loss: 34.9037 -
                val_loss: 37.0476
                Epoch 169/200
                60000/60000 [==============================] - 1s 13us/step - loss: 34.9042 -
                val_loss: 37.2811
                Epoch 170/200
                60000/60000 [==============================] - 1s 13us/step - loss: 34.9087 -
                val_loss: 37.3212
                Epoch 171/200
                60000/60000 [==============================] - 1s 13us/step - loss: 34.9001 -
                val_loss: 37.2154
                Epoch 172/200
                60000/60000 [==============================] - 1s 13us/step - loss: 34.8903 -
                val_loss: 37.1862
                Epoch 173/200
                60000/60000 [==============================] - 1s 13us/step - loss: 34.8856 -
                val_loss: 37.2628
                Epoch 174/200
                60000/60000 [==============================] - 1s 13us/step - loss: 34.8861 -
                val_loss: 37.1823
                Epoch 175/200
                60000/60000 [==============================] - 1s 13us/step - loss: 34.8772 -
                val_loss: 37.1817
                Epoch 176/200
                60000/60000 [==============================] - 1s 13us/step - loss: 34.8907 -
                val_loss: 37.1231
                Epoch 177/200
                60000/60000 [==============================] - 1s 13us/step - loss: 34.8745 -
                val_loss: 37.1837
                Epoch 178/200
                60000/60000 [==============================] - 1s 13us/step - loss: 34.8564 -
                val_loss: 37.2309
                Epoch 179/200
                60000/60000 [==============================] - 1s 14us/step - loss: 34.8661 -
                val_loss: 37.2312
                Epoch 180/200
                60000/60000 [==============================] - 1s 14us/step - loss: 34.8648 -
                val_loss: 37.2136
                Epoch 181/200
                60000/60000 [==============================] - 1s 13us/step - loss: 34.8437 -
                val_loss: 37.1749
                Epoch 182/200
                60000/60000 [==============================] - 1s 13us/step - loss: 34.8588 -
                val_loss: 37.1522
                Epoch 183/200
                60000/60000 [==============================] - 1s 13us/step - loss: 34.8223 -
                val_loss: 37.2122
                Epoch 184/200
                60000/60000 [==============================] - 1s 13us/step - loss: 34.8189 -
```

```
                val_loss: 37.1229
                Epoch 185/200
                60000/60000 [==============================] - 1s 13us/step - loss: 34.8331 -
                val_loss: 37.2309
                Epoch 186/200
                60000/60000 [==============================] - 1s 13us/step - loss: 34.8326 -
                val_loss: 37.1323
                Epoch 187/200
                60000/60000 [==============================] - 1s 13us/step - loss: 34.8076 -
                val_loss: 37.1670
                Epoch 188/200
                60000/60000 [==============================] - 1s 13us/step - loss: 34.8295 -
                val_loss: 37.2170
                Epoch 189/200
                60000/60000 [==============================] - 1s 13us/step - loss: 34.8040 -
                val_loss: 37.2571
                Epoch 190/200
                60000/60000 [==============================] - 1s 13us/step - loss: 34.7864 -
                val_loss: 37.2005
                Epoch 191/200
                60000/60000 [==============================] - 1s 13us/step - loss: 34.7864 -
                val_loss: 37.2764
                Epoch 192/200
                60000/60000 [==============================] - 1s 13us/step - loss: 34.7937 -
                val_loss: 37.1294
                Epoch 193/200
                60000/60000 [==============================] - 1s 13us/step - loss: 34.7991 -
                val_loss: 37.2520
                Epoch 194/200
                60000/60000 [==============================] - 1s 13us/step - loss: 34.7950 -
                val_loss: 37.2566
                Epoch 195/200
                60000/60000 [==============================] - 1s 13us/step - loss: 34.7870 -
                val_loss: 37.2069
                Epoch 196/200
                60000/60000 [==============================] - 1s 13us/step - loss: 34.7706 -
                val_loss: 37.1804
                Epoch 197/200
                60000/60000 [==============================] - 1s 13us/step - loss: 34.7662 -
                val_loss: 37.1843
                Epoch 198/200
                60000/60000 [==============================] - 1s 13us/step - loss: 34.7473 -
                val_loss: 37.1853
                Epoch 199/200
                60000/60000 [==============================] - 1s 13us/step - loss: 34.7520 -
                val_loss: 37.1706
                Epoch 200/200
                60000/60000 [==============================] - 1s 13us/step - loss: 34.7438 -
                val_loss: 37.2497
```

```
In [35]:  # load the best varational auto-encoder model
          variational_autoencoder.load_weights('./best_model_vae_weights.pth')

          # extract the decoder network from the original model
          decoder_vae_layer1 = variational_autoencoder.layers[5]
          decoder_vae_layer2 = variational_autoencoder.layers[6]
          vae_input = Input(shape=(2,))
          decoder_vae = Model(vae_input, decoder_vae_layer2(decoder_vae_layer1(vae_input
          )))
          decoder_vae.summary()
```

```
_____
Layer (type)                 Output Shape              Param #
=================================================================
input_4 (InputLayer)         (None, 2)                 0
_____
dense_4 (Dense)              (None, 512)               1536
_____
dense_5 (Dense)              (None, 784)               402192
=================================================================
Total params: 403,728
Trainable params: 403,728
Non-trainable params: 0
_____
```

```
In [43]:  # generate images by inputing random number to decoder
          print("Generate images by inputing different var values draw from the distribu
          tion")
          img_num = 10
          plt.figure(figsize=(18, 18))
          plt.gray()

          random_vars = np.linspace(-3, 3, img_num)[::-1]

          for i in range(len(random_vars)):
              z_sample = np.array([[2, random_vars[i]]])
              img_decoded = decoder_vae.predict(z_sample)
              ax = plt.subplot(1, img_num, i+1)
              plt.imshow(img_decoded.reshape(28, 28))
              ax.axis('off')
          plt.show()
```

Generate images by inputing different var values draw from the distribution

In [44]:
```python
# generate images by inputing random number to decoder
print("Generate images by inputing different mean values draw from the distrib
ution")
random_means = np.linspace(-2, 2, img_num)
plt.figure(figsize=(18, 18))
plt.gray()
for i in range(len(random_means)):
    z_sample = np.array([[random_means[i], 1]])
    img_decoded = decoder_vae.predict(z_sample)
    ax = plt.subplot(1, img_num, i+1)
    plt.imshow(img_decoded.reshape(28, 28))
    ax.axis('off')
plt.show()
```

Generate images by inputing different mean values draw from the distribution

Does the VAE produce a different quality of output image?

The VAE do produce different quality of output image. We can see the images produces by VAE have more randomness.

In [ ]: