Week-7: consider the network and dataset from week 6 visualize the hidden layers features. Compute the confusion matrix.

Code:

```
import numpy as np
import tensorflow as tf
from tensorflow.keras import layers, models
from sklearn.model_selection import KFold
from tensorflow.keras.datasets import mnist
from sklearn.metrics import accuracy_score, confusion_matrix
import matplotlib.pyplot as plt


# Load the MNIST dataset
(x_train, y_train), (x_test, y_test) = mnist.load_data()

# Normalize pixel values to be between 0 and 1
x_train, x_test = x_train / 255.0, x_test / 255.0

# Expand dimensions to add a channel dimension
x_train = np.expand_dims(x_train, axis=-1)
x_test = np.expand_dims(x_test, axis=-1)

# Define the CNN model with 4 layers
def create_model():
    model = models.Sequential()
```

```
    model.add(layers.Conv2D(32, (3, 3), activation='relu', input_shape=(28,
28, 1)))
    model.add(layers.MaxPooling2D((2, 2)))
    model.add(layers.Conv2D(64, (3, 3), activation='relu'))
    model.add(layers.Flatten())
    model.add(layers.Dense(64, activation='relu'))
    model.add(layers.Dense(10, activation='softmax'))
    model.compile(optimizer='sgd',
                  loss='sparse_categorical_crossentropy',
                  metrics=['accuracy'])
    return model

# Print model summary
model = create_model()
model.summary()

# Perform K-fold cross-validation
k_folds = 10
kf = KFold(n_splits=k_folds, shuffle=True, random_state=42)

accuracies = []
conf_matrices = []

for train_index, val_index in kf.split(x_train):
    x_train_fold, x_val_fold = x_train[train_index], x_train[val_index]
    y_train_fold, y_val_fold = y_train[train_index], y_train[val_index]
```

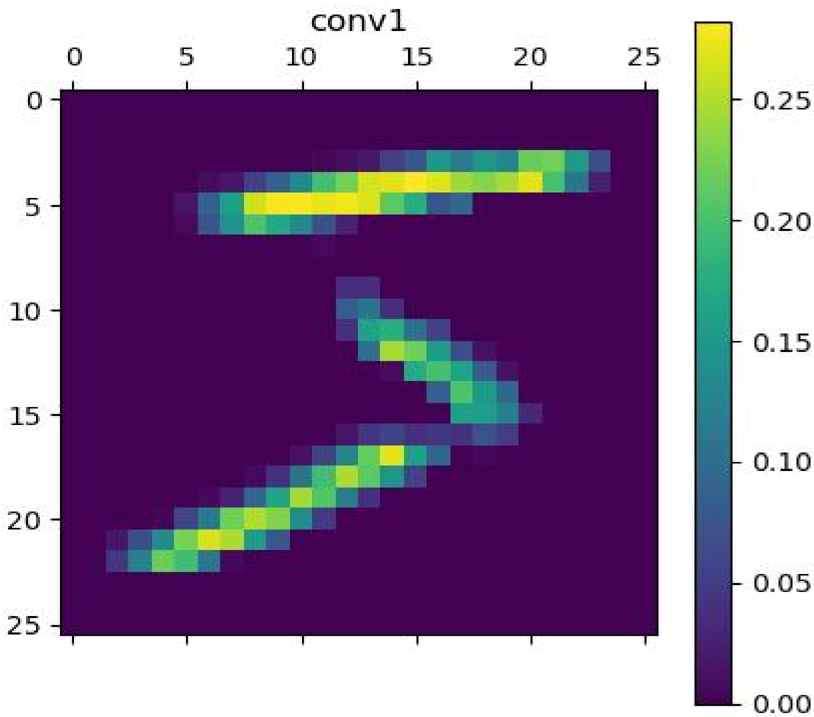
```
    model = create_model()

    # Train the model
    model.fit(x_train_fold, y_train_fold, epochs=5, batch_size=64, verbose=0)

    # Evaluate on the validation set
    val_predictions = np.argmax(model.predict(x_val_fold), axis=-1)
    accuracy = accuracy_score(y_val_fold, val_predictions)
    accuracies.append(accuracy)

    # Compute confusion matrix
    conf_matrix = confusion_matrix(y_val_fold, val_predictions)
    conf_matrices.append(conf_matrix)

    # Print the accuracy over the K fold
    print('Accuracy is:', accuracy)

# Print the confusion matrix
for fold, conf_matrix in enumerate(conf_matrices):
    print(f'Confusion Matrix for Fold {fold + 1}:')
    print(conf_matrix)
```

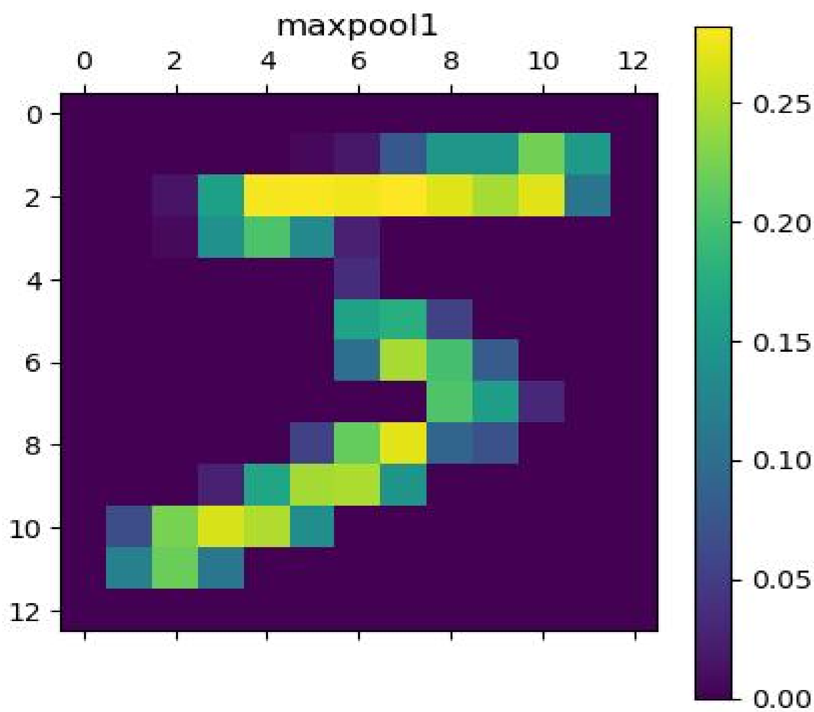
```python
# Print the average accuracy over the K folds
print(f'Average Accuracy: {np.mean(accuracies)}')


# Visualize hidden layer features
layer_outputs = [layer.output for layer in model.layers[:4]]  # Extracting the outputs of the top 4 layers
activation_model = tf.keras.models.Model(inputs=model.input, outputs=layer_outputs)  # Creates a model that will return these outputs, given the model input


sample_img = x_train[0].reshape(1, 28, 28, 1)  # Reshape an input sample
activations = activation_model.predict(sample_img)  # Predictions of the activations


# Plot the activations of each layer
for layer_activation, layer_name in zip(activations, ['conv1', 'maxpool1', 'conv2', 'flatten']):
    plt.figure()
    plt.matshow(layer_activation[0, :, :, 0], cmap='viridis')
    plt.title(layer_name)
    plt.colorbar()


plt.show()
```

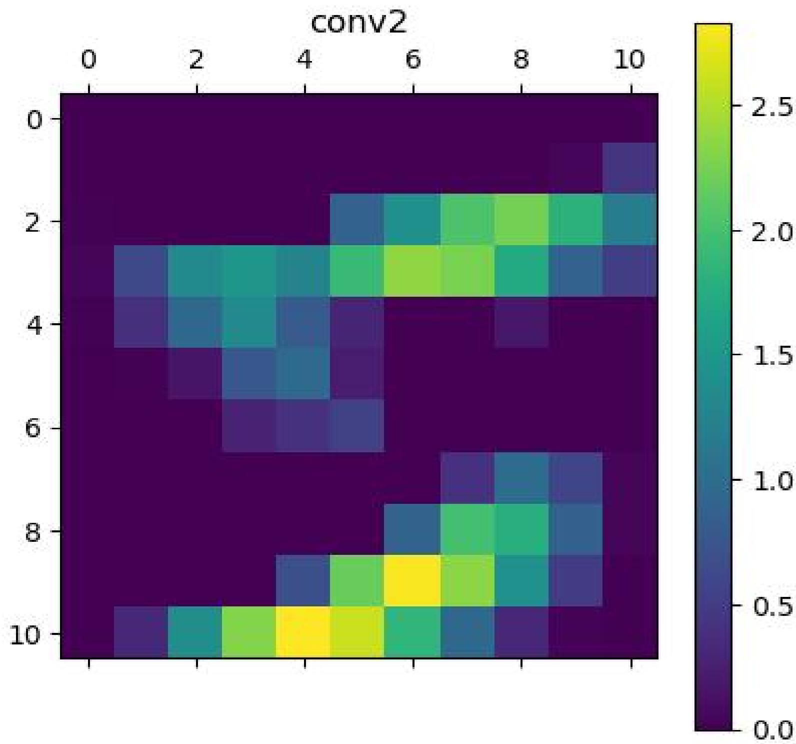
Output:

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 26, 26, 32)        320

 max_pooling2d (MaxPooling2  (None, 13, 13, 32)        0
 D)

 conv2d_1 (Conv2D)           (None, 11, 11, 64)        18496

 flatten (Flatten)           (None, 7744)              0

 dense (Dense)               (None, 64)                495680

 dense_1 (Dense)             (None, 10)                650

=================================================================
Total params: 515146 (1.97 MB)
Trainable params: 515146 (1.97 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________


188/188 [==============================] - 4s 17ms/step
Accuracy is: 0.9588333333333333
188/188 [==============================] - 4s 16ms/step
Accuracy is: 0.9598333333333333
188/188 [==============================] - 4s 16ms/step
Accuracy is: 0.9616666666666667
188/188 [==============================] - 3s 14ms/step
Accuracy is: 0.9663333333333334
188/188 [==============================] - 3s 13ms/step
Accuracy is: 0.9595
188/188 [==============================] - 4s 16ms/step
Accuracy is: 0.972
188/188 [==============================] - 3s 15ms/step
Accuracy is: 0.96
188/188 [==============================] - 3s 13ms/step
Accuracy is: 0.9571666666666667
188/188 [==============================] - 3s 13ms/step
Accuracy is: 0.9611666666666666
188/188 [==============================] - 5s 19ms/step
Accuracy is: 0.9651666666666666
```

```
Confusion Matrix for Fold 1:
[[604   0   3   2   0   2   3   1   8   1]
 [  0 645   2   0   0   0   0   0   6   1]
 [  0   2 559   2   2   0   0   2   4   1]
 [  2   1   7 557   0  10   0   1  11   0]
 [  1   3   3   2 562   0   1   1   4   3]
 [  2   2   2   9   3 520   2   0  11   0]
 [  2   0   1   0   2   4 568   0   3   0]
 [  1   6  12   3   3   3   0 589  10   6]
 [  1   4   1   3   1   1   4   0 570   0]
 [  1   2   1   9  19   3   0   3  15 579]]
Confusion Matrix for Fold 2:
[[545   0   1   0   2   1   0   0   0   2]
 [  0 659   4   2   1   0   0   0   1   1]
 [  0   5 584   2   1   0   0   6   3   1]
 [  1   1   5 602   0   9   0   3   4   5]
 [  0   0   0   0 583   0   1   0   0  12]
 [  5   1   1  12   1 524   3   0   1   5]
 [  0   3   2   0   5   8 576   0   3   0]
 [  1   2   4   4   6   1   0 634   0  14]
 [  2   6   4  23   4   8   2   3 512  11]
 [  2   2   0   1   8   3   0   6   0 540]]
-
-
-
-
-
-
-
Confusion Matrix for Fold 10:
[[578   0   1   0   1   2   5   1   1   0]
 [  0 663   3   0   0   1   1   2   4   0]
 [  1   1 573   0   5   0   0   4   4   1]
 [  0   3   9 539   0  15   1   4   7   0]
 [  1   0   1   0 566   0   4   2   0  10]
 [  0   1   0   3   2 518   3   0   1   2]
 [  7   4   1   1   0   7 586   0   3   0]
 [  0   1   3   0   1   1   1 609   1   6]
 [  1   3   4   1   1   7   4   1 599   4]
 [  0   1   2   4   5   9   0  13   5 560]]

Average Accuracy: 0.9621666666666666
1/1 [==============================] - 1s 772ms/step
```


conv1
0
5
10
15
20
25
0.25
0.20
0.15
0.10
0.05
0.00


maxpool1
0
2
4
6
8
10
12
0.25
0.20
0.15
0.10
0.05
0.00


conv2
0
2
4
6
8
10
0
2
4
6
8
10
2.5
2.0
1.5
1.0
0.5
0.0