NAME - JATAN SAHU

ID-202218061

LAB - 04

SUBJECT - DEEP LEARNING

INSTRUCTIONS

1.Introduction

Working with images This lab aims to demonstrate the efficacy of ANNs in image processing, alongside regression analysis on the provided dataset. Additionally, it provides an opportunity to gain expertise in implementing normalization techniques and integrating skip connections within a deep learning model.

2.Working with SKIP connections and regularization

Follow the given notebook for building the models with skip connection and regularization.

3.Dataset

1.MNIST: The stepping stone or Hello world of Deep Learning this dataset contains images of handwritten digits

2.CIFAR-10: Contains colored images of various objects

3.Auto-mpg: The data is technical spec of cars. In this regression dataset we need to predict ‘mpg’ attribute from other column values.

4.Tasks:

4.1Classification

1.Load and visualize the images from the dataset.

2.Apply preprocessing and encoding to labels.

3.Define the ANN model for image classification. Include normalization and skip connections in our model.

4.Experiment with the different activation functions and loss functions while training the model

5.Analyze ANN model performance with different batch sizes (test of 3 different batch size) and learning rates (3 different learning rates)

6.Plot the accuracy for train and test data

7.You can use matplotlib plots to present your analysis for different hyper- parameters

8.Evaluate the model’s performance using different performance matrices discussed in the class

4.2Regression

1.Load and preprocess the given data.

2.Build ANN model with regularization and skip connections and train it on the given data.

3.Analyze ANN model performance with different batch sizes (test of 3 different batch size) and learning rates (3 different learning rates).

4.Plot mse, mae and rmse for different batch size and learning rates.

Classification TASK

DATASET01 - CIFAR DATA

1. Importing Libraries

```
from tensorflow.keras.models import Model
from tensorflow.keras.layers import Dense,Input, Dropout, BatchNormalization, Add, Concatenate
from keras.utils import to_categorical, plot_model
```

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.datasets import load_linnerud
from sklearn import datasets
from sklearn.model_selection import train_test_split
import numpy as np
from keras.datasets import cifar10
from keras.models import Sequential, Model
from keras.layers import Dense, BatchNormalization, Flatten, Input, Concatenate
from keras.optimizers import SGD
from keras.utils import to_categorical
import matplotlib.pyplot as plt
from sklearn.metrics import classification_report, confusion_matrix
```

2.Importing dataset using Tensorflow

```
import tensorflow as tf

# Load CIFAR-10 dataset
(x_train, y_train), (x_test, y_test) = tf.keras.datasets.cifar10.load_data()
```

```
Downloading data from https://www.cs.toronto.edu/~kriz/cifar-10-python.tar.gz
170498071/170498071 [==============================] - 11s 0us/step
```

Learning rate - 0.001

2. ML Training model PIPELINE

2.1Splitting data

2.2Preprocessing

2.3Flattening data

2.4Adding activation function

2.5Normalisation

2.6Skip Connectin

2.7 Adding Optimizer,loss function, metrics

3. Experiment with

3.1Activation functions

3.2Loss functions

3.3Batch sizes

3.4Learning rates

```
# Load the CIFAR-10 dataset and preprocess it
(x_train, y_train), (x_test, y_test) = cifar10.load_data()
x_train = x_train.astype('float32') / 255
x_test = x_test.astype('float32') / 255
y_train_encoded = to_categorical(y_train, num_classes=10)
y_test_encoded = to_categorical(y_test, num_classes=10)

# Define a function to create and train the model
def create_and_train_model(activation_func, loss_func, batch_size, learning_rate):
    input_layer = Input(shape=(32, 32, 3))
    x = Flatten()(input_layer)
    x = Dense(128, activation=activation_func)(x)
    x = BatchNormalization()(x)

    skip_connection = x  # Save a copy of the output for the skip connection

    x = Dense(64, activation=activation_func)(x)
    x = BatchNormalization()(x)

    x = Dense(32, activation=activation_func)(x)
    x = BatchNormalization()(x)

    # Concatenate the output of the skip connection with the current output
    x = Concatenate()([x, skip_connection])

    output_layer = Dense(10, activation='softmax')(x)

    model = Model(inputs=input_layer, outputs=output_layer)

    optimizer = SGD(learning_rate=learning_rate)
    model.compile(optimizer=optimizer, loss=loss_func, metrics=['accuracy'])

    history = model.fit(x_train, y_train_encoded, validation_data=(x_test, y_test_encoded),
                        batch_size=batch_size, epochs=10, verbose=0)

    return history, model

# Experiment with different activation functions, loss functions, batch sizes, and learning rates
activation_functions = ['relu', 'tanh', 'sigmoid']
loss_functions = ['categorical_crossentropy', 'mean_squared_error']
batch_sizes = [32, 64, 128]
learning_rates = [0.001]

results = []

for activation_func in activation_functions:
    for loss_func in loss_functions:
        for batch_size in batch_sizes:
            for learning_rate in learning_rates:
                history, model = create_and_train_model(activation_func, loss_func, batch_size, learni
                accuracy = history.history['accuracy'][-1]
                val_accuracy = history.history['val_accuracy'][-1]
                results.append((activation_func, loss_func, batch_size, learning_rate, accuracy, val_a
```

4.Plots

4.1Accuracy for train and test

5.Evaluation

5.1Classification Report

5.2Confusion Matrix

```
# Plot the accuracy for train and test data for each combination
for result in results:
    activation_func, loss_func, batch_size, learning_rate, accuracy, val_accuracy, history, model = re
    label = f'{activation_func}_{loss_func}_{batch_size}_{learning_rate}'

    # Create a new figure and axes for each combination
    plt.figure()
    plt.plot(history.history['accuracy'], label=f'{label}_train', linestyle='-')
    plt.plot(history.history['val_accuracy'], label=f'{label}_test', linestyle='--')
    plt.xlabel('Epochs')
    plt.ylabel('Accuracy')
    plt.title(f'Model with {activation_func} activation, {loss_func} loss, batch size {batch_size}, le
    plt.legend(loc='lower right')
    plt.show()

# Evaluate the model's performance using different performance metrics
for result in results:
    activation_func, loss_func, batch_size, learning_rate, accuracy, val_accuracy, history, model = re
    y_pred = model.predict(x_test)
    y_pred_classes = np.argmax(y_pred, axis=1)
    y_true = y_test.squeeze()

    print(f'Model with {activation_func} activation, {loss_func} loss, batch size {batch_size}, learni

    # Classification Report
    print(classification_report(y_true, y_pred_classes))

    # Confusion Matrix
    cm = confusion_matrix(y_true, y_pred_classes)
    print("Confusion Matrix:")
    print(cm)
```

DATASET 02 -MNIST DATA

```
 from tensorflow.keras.datasets import mnist
```

```
(train_images, train_labels), (test_images, test_labels) = mnist.load_data()
train_images.shape
```

```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/mnist.npz
11490434/11490434 [==============================] - 1s 0us/step
(60000, 28, 28)
```

```
train_labels.shape
```

```
(60000,)
```

```
test_images.shape
```

```
(10000, 28, 28)
```

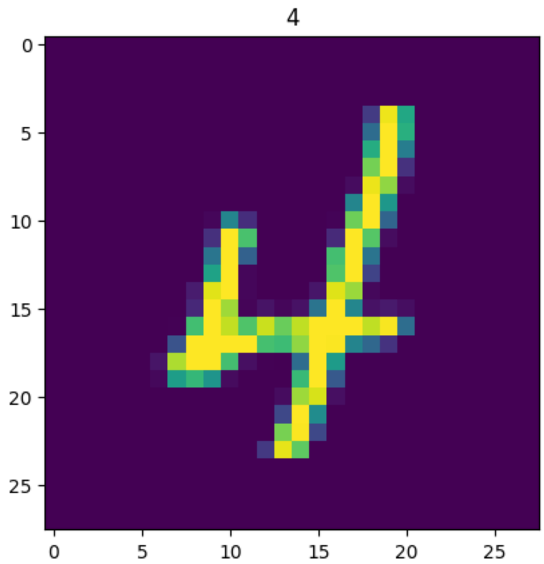
```
train_labels
```

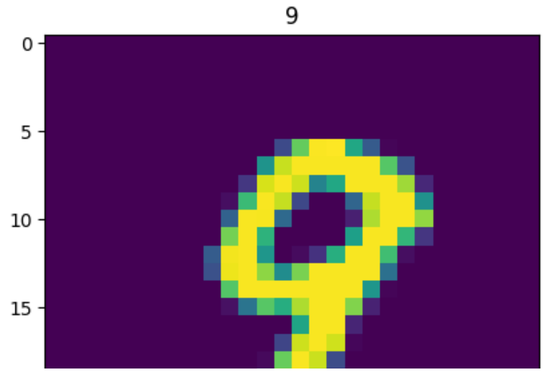
```
array([5, 0, 4, ..., 5, 6, 8], dtype=uint8)
```

```
train_images
```

```
array([[[0, 0, 0, ..., 0, 0, 0],
        [0, 0, 0, ..., 0, 0, 0],
        [0, 0, 0, ..., 0, 0, 0],
        ...,
        [0, 0, 0, ..., 0, 0, 0],
        [0, 0, 0, ..., 0, 0, 0],
        [0, 0, 0, ..., 0, 0, 0]],

       [[0, 0, 0, ..., 0, 0, 0],
        [0, 0, 0, ..., 0, 0, 0],
        [0, 0, 0, ..., 0, 0, 0],
        ...,
        [0, 0, 0, ..., 0, 0, 0],
        [0, 0, 0, ..., 0, 0, 0],
        [0, 0, 0, ..., 0, 0, 0]],

       [[0, 0, 0, ..., 0, 0, 0],
        [0, 0, 0, ..., 0, 0, 0],
        [0, 0, 0, ..., 0, 0, 0],
        ...,
        [0, 0, 0, ..., 0, 0, 0],
        [0, 0, 0, ..., 0, 0, 0],
        [0, 0, 0, ..., 0, 0, 0]],

       ...,

       [[0, 0, 0, ..., 0, 0, 0],
        [0, 0, 0, ..., 0, 0, 0],
        [0, 0, 0, ..., 0, 0, 0],
        ...,
        [0, 0, 0, ..., 0, 0, 0],
        [0, 0, 0, ..., 0, 0, 0],
        [0, 0, 0, ..., 0, 0, 0]],

       [[0, 0, 0, ..., 0, 0, 0],
        [0, 0, 0, ..., 0, 0, 0],
        [0, 0, 0, ..., 0, 0, 0],
        ...,
        [0, 0, 0, ..., 0, 0, 0],
        [0, 0, 0, ..., 0, 0, 0],
        [0, 0, 0, ..., 0, 0, 0]],

       [[0, 0, 0, ..., 0, 0, 0],
        [0, 0, 0, ..., 0, 0, 0],
        [0, 0, 0, ..., 0, 0, 0],
        ...,
        [0, 0, 0, ..., 0, 0, 0],
        [0, 0, 0, ..., 0, 0, 0],
        [0, 0, 0, ..., 0, 0, 0]]], dtype=uint8)
```

```
# Plotting images
for index in np.random.randint(0,60000,4):
  plt.imshow(train_images[index,:,:])
  plt.title(train_labels[index])
  plt.show()
```





2. Apply preprocessing and encoding to labels.

```
# Encoding Labels: Convert integer labels to one-hot encoded vectors
train_labels_encoded = to_categorical(train_labels)
test_labels_encoded = to_categorical(test_labels)
```

```
# Preprocessing: Normalize pixel values to the range [0, 1]
train_images = train_images.astype('float32') / 255
test_images = test_images.astype('float32') / 255

# Reshape the image data
train_images = train_images.reshape((train_images.shape[0], -1))
test_images = test_images.reshape((test_images.shape[0], -1))
```

```
train_labels_encoded.shape, test_labels_encoded.shape
```

```
((60000, 10), (10000, 10))
```

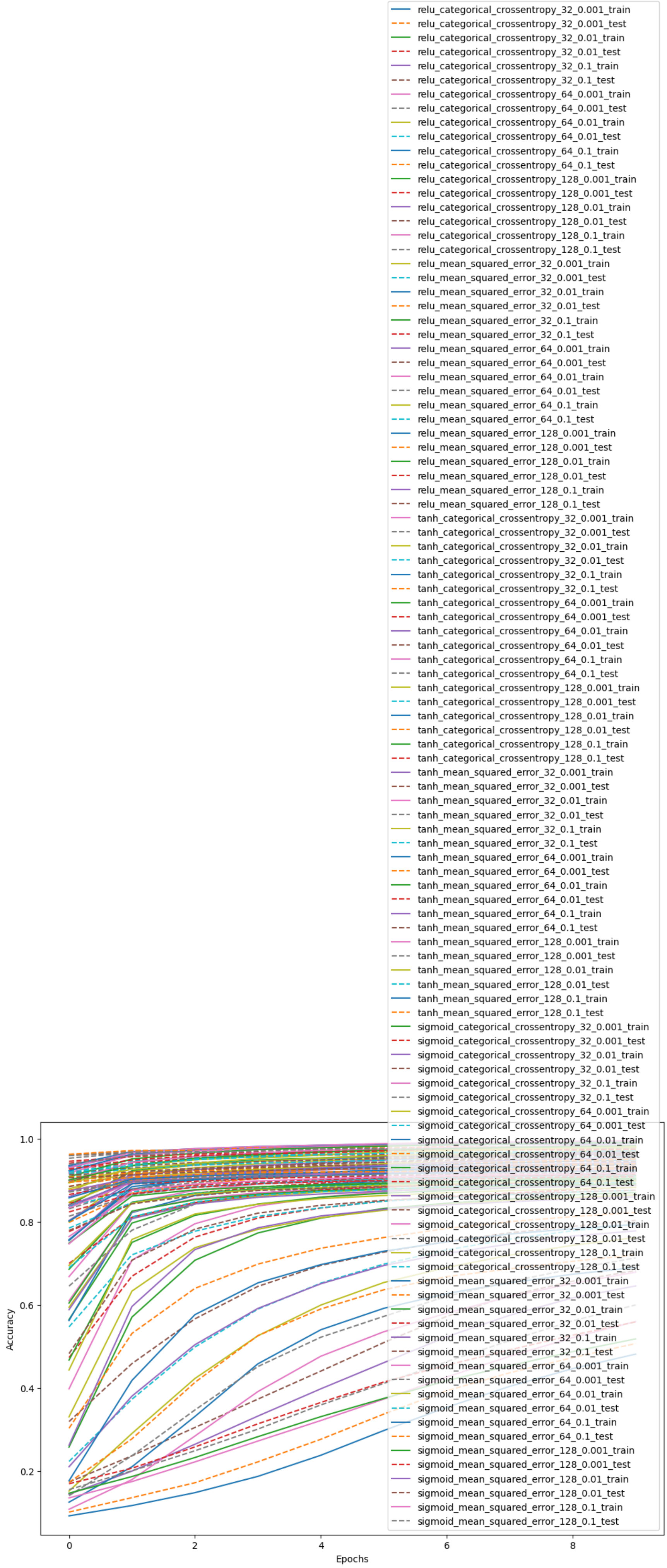
```
train_images.shape, train_images.shape
```

```
((60000, 784), (60000, 784))
```

3. Define the ANN model for image classification. Include normalization and skip connections in our model.

```
import numpy as np
from keras.datasets import mnist
from keras.models import Sequential, Model
from keras.layers import Dense, BatchNormalization, Flatten, Input, Concatenate
from keras.optimizers import SGD
from keras.utils import to_categorical
import matplotlib.pyplot as plt
from sklearn.metrics import classification_report, confusion_matrix

(train_images, train_labels), (test_images, test_labels) = mnist.load_data()
```

```
train_images = train_images.astype('float32') / 255
test_images = test_images.astype('float32') / 255
train_labels_encoded = to_categorical(train_labels)
test_labels_encoded = to_categorical(test_labels)

# Define a function to create and train the model
def create_and_train_model(activation_func, loss_func, batch_size, learning_rate):
    input_layer = Input(shape=(28, 28))
    x = Flatten()(input_layer)
    x = Dense(128, activation=activation_func)(x)
    x = BatchNormalization()(x)

    skip_connection = x  # Save a copy of the output for the skip connection

    x = Dense(64, activation=activation_func)(x)
    x = BatchNormalization()(x)

    x = Dense(32, activation=activation_func)(x)
    x = BatchNormalization()(x)

    # Concatenate the output of the skip connection with the current output
    x = Concatenate()([x, skip_connection])

    output_layer = Dense(10, activation='softmax')(x)

    model = Model(inputs=input_layer, outputs=output_layer)

    optimizer = SGD(learning_rate=learning_rate)
    model.compile(optimizer=optimizer, loss=loss_func, metrics=['accuracy'])

    history = model.fit(train_images, train_labels_encoded, validation_data=(test_images, test_labels_
                        batch_size=batch_size, epochs=10, verbose=0)

    return history, model

# Experiment with different activation functions, loss functions, batch sizes, and learning rates
activation_functions = ['relu', 'tanh', 'sigmoid']
loss_functions = ['categorical_crossentropy', 'mean_squared_error']
batch_sizes = [32, 64, 128]
learning_rates = [0.001, 0.01, 0.1]

results = []

for activation_func in activation_functions:
    for loss_func in loss_functions:
        for batch_size in batch_sizes:
            for learning_rate in learning_rates:
                history, model = create_and_train_model(activation_func, loss_func, batch_size, learni
                accuracy = history.history['accuracy'][-1]
                val_accuracy = history.history['val_accuracy'][-1]
                results.append((activation_func, loss_func, batch_size, learning_rate, accuracy, val_a

# Plot the accuracy for train and test data for each combination
plt.figure(figsize=(12, 8))
for result in results:
    activation_func, loss_func, batch_size, learning_rate, accuracy, val_accuracy, history, model = re
    label = f'{activation_func}_{loss_func}_{batch_size}_{learning_rate}'
    plt.plot(history.history['accuracy'], label=f'{label}_train', linestyle='-')
    plt.plot(history.history['val_accuracy'], label=f'{label}_test', linestyle='--')

plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend(loc='lower right')
plt.show()

# Evaluate the model's performance using different performance metrics
for result in results:
    activation_func, loss_func, batch_size, learning_rate, accuracy, val_accuracy, history, model = re
    y_pred = model.predict(test_images)
    y_pred_classes = np.argmax(y_pred, axis=1)
    y_true = np.argmax(test_labels_encoded, axis=1)

    print(f'Model with {activation_func} activation, {loss_func} loss, batch size {batch_size}, learni

    # Classification Report
    print(classification_report(y_true, y_pred_classes))

    # Confusion Matrix
    cm = confusion_matrix(y_true, y_pred_classes)
    print("Confusion Matrix:")
    print(cm)
```



```
313/313 [==============================] - 1s 3ms/step
Model with relu activation, categorical_crossentropy loss, batch size 32, learning ra
```

```
              precision    recall  f1-score   support

           0       0.96      0.98      0.97       980
           1       0.98      0.98      0.98      1135
           2       0.95      0.94      0.95      1032
           3       0.94      0.94      0.94      1010
           4       0.93      0.95      0.94       982
           5       0.94      0.94      0.94       892
           6       0.95      0.96      0.96       958
           7       0.96      0.95      0.95      1028
           8       0.94      0.93      0.93       974
           9       0.94      0.92      0.93      1009

    accuracy                           0.95     10000
   macro avg       0.95      0.95      0.95     10000
weighted avg       0.95      0.95      0.95     10000

Confusion Matrix:
[[ 963    0    1    1    0    4    6    2    3    0]
 [   0 1114    2    3    0    1    5    1    9    0]
 [   6    0  973    7   11    3    4   10   17    1]
 [   0    2   10  953    1   14    4    9   10    7]
 [   2    3    4    1  931    0    6    2    4   29]
 [   6    1    0   19    4  838   11    1    9    3]
 [  10    3    4    0    5   10  921    1    4    0]
 [   3    5   18    6    5    1    0  973    2   15]
 [   7    2    8   11    8   11    8    8  907    4]
 [   6    6    2    9   32    8    1   10    3  932]]
313/313 [==============================] - 1s 3ms/step
Model with relu activation, categorical_crossentropy loss, batch size 32, learning ra
              precision    recall  f1-score   support

           0       0.97      0.99      0.98       980
           1       0.99      0.99      0.99      1135
           2       0.98      0.97      0.97      1032
           3       0.97      0.97      0.97      1010
           4       0.97      0.98      0.98       982
           5       0.98      0.97      0.97       892
           6       0.97      0.97      0.97       958
           7       0.97      0.97      0.97      1028
           8       0.98      0.97      0.97       974
           9       0.96      0.97      0.97      1009

    accuracy                           0.97     10000
   macro avg       0.97      0.97      0.97     10000
weighted avg       0.97      0.97      0.97     10000

Confusion Matrix:
[[ 969    1    1    0    1    2    4    1    1    0]
 [   0 1122    3    1    0    0    3    1    5    0]
 [   6    2  997    4    5    0    3    9    6    0]
 [   0    0    5  981    0   10    0    5    4    5]
 [   1    0    2    1  961    1    3    1    0   12]
 [   5    0    0   11    0  864    6    1    2    3]
 [   5    3    1    1    8    3  934    1    2    0]
 [   1    4   10    3    3    0    0  994    2   11]
 [   2    0    2    5    1    6    5    4  943    6]
 [   5    3    0    3   10    0    0    4    2  982]]
313/313 [==============================] - 1s 3ms/step
Model with relu activation, categorical_crossentropy loss, batch size 32, learning ra
              precision    recall  f1-score   support

           0       0.98      0.99      0.99       980
           1       0.99      0.99      0.99      1135
           2       0.98      0.98      0.98      1032
           3       0.97      0.99      0.98      1010
           4       0.98      0.97      0.97       982
           5       0.98      0.98      0.98       892
           6       0.99      0.98      0.98       958
           7       0.98      0.98      0.98      1028
           8       0.98      0.98      0.98       974
           9       0.97      0.97      0.97      1009

    accuracy                           0.98     10000
   macro avg       0.98      0.98      0.98     10000
weighted avg       0.98      0.98      0.98     10000

Confusion Matrix:
[[ 974    1    1    0    0    1    1    1    1    0]
 [   0 1126    3    1    0    1    2    0    2    0]
 [   4    1 1010    4    1    1    0    9    2    0]
 [   0    0    0  998    0    1    0    4    5    2]
 [   2    1    2    1  954    0    3    2    2   15]
 [   2    1    0   10    0  870    4    1    2    2]
 [   4    1    1    1    6    6  938    0    1    0]
 [   1    1    7    1    1    0    0 1011    2    4]
 [   3    1    2    6    1    2    1    3  952    3]
 [   2    2    0    5   13    3    0    4    2  978]]
313/313 [==============================] - 1s 4ms/step
Model with relu activation, categorical_crossentropy loss, batch size 64, learning ra
              precision    recall  f1-score   support
              precision    recall  f1-score   support

           0       0.95      0.98      0.96       980
           1       0.97      0.98      0.97      1135
           2       0.95      0.92      0.93      1032
           3       0.92      0.92      0.92      1010
           4       0.91      0.93      0.92       982
           5       0.92      0.88      0.90       892
           6       0.93      0.96      0.94       958
           7       0.93      0.93      0.93      1028
           8       0.91      0.90      0.91       974
           9       0.92      0.91      0.91      1009

    accuracy                           0.93     10000
   macro avg       0.93      0.93      0.93     10000
weighted avg       0.93      0.93      0.93     10000

Confusion Matrix:
[[ 959    1    0    0    0    6   11    1    2    0]
 [   0 1107    2    3    1    1    5    2   14    0]
 [   9    2  946   17   13    4    8   20   13    0]
 [   4    0   12  931    1   24    4   11   13   10]
 [   1    3    3    1  918    0   14    0    7   35]
 [   8    4    6   27   10  787   13    6   24    7]
 [  11    3    1    1   10   11  915    2    4    0]
 [   2   17   20    4    9    0    0  953    2   21]
 [   6    4    6   17   10   19   12   11  877   12]
 [  12    6    0   13   33    7    0   16    4  918]]
313/313 [==============================] - 1s 3ms/step
Model with relu activation, categorical_crossentropy loss, batch size 64, learning ra
              precision    recall  f1-score   support

           0       0.98      0.98      0.98       980
           1       0.99      0.99      0.99      1135
           2       0.96      0.97      0.97      1032
           3       0.96      0.97      0.97      1010
           4       0.96      0.97      0.97       982
           5       0.97      0.97      0.97       892
           6       0.97      0.97      0.97       958
           7       0.97      0.96      0.96      1028
           8       0.95      0.97      0.96       974
           9       0.97      0.95      0.96      1009

    accuracy                           0.97     10000
   macro avg       0.97      0.97      0.97     10000
weighted avg       0.97      0.97      0.97     10000

Confusion Matrix:
[[ 964    0    1    0    0    2    8    1    3    1]
 [   0 1120    2    2    0    0    5    1    5    0]
 [   5    1 1000    5    5    0    2    5    9    0]
 [   0    0    6  978    0    9    0    4    8    5]
 [   1    0    9    0  957    0    2    2    2    9]
 [   3    0    0    6    2  863    8    2    6    2]
 [   5    4    1    0    6    8  925    2    7    0]
 [   1    9   14    7    4    0    0  982    2    9]
 [   2    0    4    8    5    5    4    2  940    4]
 [   3    3    0    9   21    2    1    7    7  956]]
313/313 [==============================] - 1s 3ms/step
Model with relu activation, categorical_crossentropy loss, batch size 64, learning ra
              precision    recall  f1-score   support

           0       0.98      0.99      0.99       980
           1       0.99      0.99      0.99      1135
           2       0.98      0.98      0.98      1032
           3       0.98      0.98      0.98      1010
           4       0.98      0.97      0.98       982
           5       0.98      0.98      0.98       892
           6       0.98      0.98      0.98       958
           7       0.98      0.98      0.98      1028
           8       0.97      0.98      0.97       974
           9       0.97      0.97      0.97      1009

    accuracy                           0.98     10000
   macro avg       0.98      0.98      0.98     10000
weighted avg       0.98      0.98      0.98     10000

Confusion Matrix:
[[ 967    0    1    0    0    1    3    2    4    2]
 [   1 1125    2    0    0    1    2    1    3    0]
 [   4    1 1011    1    1    0    2    4    8    0]
 [   0    0    4  993    0    3    0    3    1    6]
 [   1    0    0    1  957    0    5    0    1   17]
 [   2    0    0    6    1  875    3    1    3    1]
 [   3    3    0    1    6    7  936    0    2    0]
 [   0    1    8    2    2    0    0 1007    3    5]
 [   0    0    2    6    2    5    2    3  951    3]
 [   4    2    1    5    6    4    1    5    2  979]]
313/313 [==============================] - 1s 3ms/step
Model with relu activation, categorical_crossentropy loss, batch size 128, learning r
              precision    recall  f1-score   support
              precision    recall  f1-score   support

           0       0.94      0.97      0.96       980
           1       0.95      0.98      0.96      1135
           2       0.91      0.89      0.90      1032
           3       0.91      0.92      0.91      1010
           4       0.89      0.92      0.91       982
           5       0.90      0.87      0.89       892
           6       0.93      0.94      0.93       958
           7       0.93      0.90      0.91      1028
           8       0.91      0.88      0.90       974
           9       0.90      0.88      0.89      1009

    accuracy                           0.92     10000
   macro avg       0.92      0.92      0.92     10000
weighted avg       0.92      0.92      0.92     10000

Confusion Matrix:
[[ 955    1    1    2    0    9    7    2    3    0]
 [   0 1113    2    3    1    0    4    0   12    0]
 [  10    7  918   14   23    5   13   18   22    2]
 [   3    3   17  926    3   24    4   12   13    5]
 [   1    6    5    2  907    1   13    1    6   40]
 [  14    4    7   30    9  777   17    9   17    8]
 [  14    5    6    1    6   14  905    2    4    1]
 [   2   18   34    3    6    3    3  929    1   29]
 [   6    8   10   22   15   25    9   11  859    9]
 [  10    8    5   18   45    4    3   19    6  891]]
313/313 [==============================] - 1s 3ms/step
Model with relu activation, categorical_crossentropy loss, batch size 128, learning r
              precision    recall  f1-score   support

           0       0.97      0.98      0.98       980
           1       0.99      0.99      0.99      1135
           2       0.95      0.96      0.95      1032
           3       0.95      0.96      0.95      1010
           4       0.96      0.97      0.96       982
           5       0.95      0.96      0.95       892
           6       0.96      0.96      0.96       958
           7       0.97      0.96      0.96      1028
           8       0.95      0.95      0.95       974
           9       0.96      0.95      0.96      1009

    accuracy                           0.96     10000
   macro avg       0.96      0.96      0.96     10000
weighted avg       0.96      0.96      0.96     10000

Confusion Matrix:
[[ 958    0    2    1    0    3    9    2    4    1]
 [   0 1119    5    2    0    1    2    0    6    0]
 [   4    1  986    7    5    1   10    9    9    0]
 [   0    0    8  965    0   18    0    7    9    3]
 [   3    0    7    1  948    0    4    2    3   14]
 [   5    1    1   15    1  853    7    1    5    3]
 [   7    3    0    1    7   11  922    1    6    0]
 [   1    7   19    4    1    0    0  983    3   10]
 [   4    0    4   13    6    5    6    3  926    7]
 [   3    4    1    5   15    3    1    6    8  963]]
313/313 [==============================] - 1s 3ms/step
Model with relu activation, categorical_crossentropy loss, batch size 128, learning r
              precision    recall  f1-score   support

           0       0.97      0.99      0.98       980
           1       0.99      0.99      0.99      1135
           2       0.98      0.97      0.98      1032
           3       0.96      0.98      0.97      1010
           4       0.98      0.98      0.98       982
           5       0.98      0.97      0.97       892
           6       0.98      0.98      0.98       958
           7       0.98      0.98      0.98      1028
           8       0.97      0.97      0.97       974
           9       0.97      0.97      0.97      1009

    accuracy                           0.98     10000
   macro avg       0.98      0.98      0.98     10000
weighted avg       0.98      0.98      0.98     10000

Confusion Matrix:
[[ 971    0    1    0    0    2    4    1    1    0]
 [   0 1121    2    1    0    2    3    1    5    0]
 [   4    0 1006    6    3    0    3    3    7    0]
 [   2    0    2  990    0    5    0    3    3    5]
 [   4    0    1    1  963    0    3    2    0    8]
 [   3    0    0   11    1  865    4    2    3    3]
 [   4    2    0    0    2    9  936    0    5    0]
 [   2    2    8    3    1    0    0 1003    4    5]
 [   5    0    2    8    3    3    4    2  943    4]
 [   5    2    1    6   12    1    0    4    4  974]]
313/313 [==============================] - 1s 3ms/step
Model with relu activation, mean_squared_error loss, batch size 32, learning rate 0.0
              precision    recall  f1-score   support
           0       0.86      0.92      0.89       980
           1       0.84      0.98      0.90      1135
           2       0.82      0.72      0.76      1032
           3       0.78      0.82      0.80      1010
           4       0.74      0.80      0.77       982
           5       0.74      0.52      0.61       892
           6       0.80      0.87      0.83       958
           7       0.82      0.81      0.81      1028
           8       0.76      0.67      0.71       974
           9       0.74      0.79      0.77      1009

    accuracy                           0.79     10000
   macro avg       0.79      0.79      0.79     10000
weighted avg       0.79      0.79      0.79     10000

Confusion Matrix:
[[ 901    5   16   11    0   16   21    5    4    1]
 [   0 1112    4    4    1    1    3    2    8    0]
 [  18   47  740   36   34   10   42   38   52   15]
 [   8   11   27  826    9   42    8   29   27   23]
 [   8    7    4    4  785   10   51    4   11   98]
 [  37   49   22   97   64  466   34   26   64   33]
 [  31   14   25    6   25   17  829    2    9    0]
 [  10   33   16    3   33    6    7  828   20   72]
 [  16   36   46   53   35   38   33   30  653   34]
 [  15   11    5   14   72   22    9   51   13  797]]
313/313 [==============================] - 1s 3ms/step
Model with relu activation, mean_squared_error loss, batch size 32, learning rate 0.0
              precision    recall  f1-score   support

           0       0.94      0.98      0.96       980
           1       0.97      0.98      0.97      1135
           2       0.93      0.91      0.92      1032
           3       0.93      0.92      0.92      1010
           4       0.92      0.94      0.93       982
           5       0.93      0.90      0.92       892
           6       0.93      0.95      0.94       958
           7       0.94      0.92      0.93      1028
           8       0.91      0.91      0.91       974
           9       0.92      0.91      0.91      1009

    accuracy                           0.93     10000
   macro avg       0.93      0.93      0.93     10000
weighted avg       0.93      0.93      0.93     10000

Confusion Matrix:
[[ 956    1    0    0    2    5    8    2    6    0]
 [   0 1111    3    3    0    0    6    1   11    0]
 [   8    2  938   14   10    4   15   14   16   11]
 [   3    1   20  929    4   22    2    9   13    7]
 [   2    2    6    0  920    0   11    2    8   31]
 [   8    0    3   26    3  806   14    5   23    4]
 [  15    4    5    0    9    9  912    1    3    0]
 [   2   13   22    6   12    3    1  948    1   20]
 [  13    4    7   14    8   13   15    8  883    9]
 [  11    9    1   12   28    7    1   22    4  914]]
313/313 [==============================] - 1s 3ms/step
Model with relu activation, mean_squared_error loss, batch size 32, learning rate 0.1
              precision    recall  f1-score   support

           0       0.97      0.99      0.98       980
           1       0.98      0.99      0.98      1135
           2       0.96      0.97      0.96      1032
           3       0.97      0.96      0.97      1010
           4       0.96      0.97      0.97       982
           5       0.97      0.96      0.96       892
           6       0.97      0.98      0.97       958
           7       0.97      0.96      0.97      1028
           8       0.97      0.96      0.96       974
           9       0.98      0.95      0.96      1009

    accuracy                           0.97     10000
   macro avg       0.97      0.97      0.97     10000
weighted avg       0.97      0.97      0.97     10000

Confusion Matrix:
[[ 967    0    2    2    1    1    3    2    1    1]
 [   0 1120    4    2    1    1    4    0    3    0]
 [   6    1 1005    5    2    0    2    6    5    0]
 [   0    0   11  971    1   12    2    6    4    3]
 [   1    0    6    1  957    0    3    2    4    8]
 [   4    0    2    7    2  856    8    2    8    3]
 [   6    2    2    0    6    4  936    1    1    0]
 [   1   10   14    1    3    1    1  991    2    4]
 [   6    2    5    4    5    5    4    4  935    4]
 [   4    6    1    9   19    4    2    9    1  954]]
313/313 [==============================] - 1s 4ms/step
Model with relu activation, mean_squared_error loss, batch size 64, learning rate 0.0
              precision    recall  f1-score   support
           0       0.71      0.88      0.79       980
           1       0.76      0.94      0.84      1135
           2       0.73      0.65      0.69      1032
           3       0.65      0.64      0.65      1010
           4       0.62      0.64      0.63       982
           5       0.52      0.33      0.41       892
           6       0.80      0.78      0.79       958
           7       0.71      0.81      0.76      1028
           8       0.57      0.49      0.53       974
           9       0.62      0.57      0.60      1009

    accuracy                           0.68     10000
   macro avg       0.67      0.68      0.67     10000
weighted avg       0.67      0.68      0.67     10000

Confusion Matrix:
[[ 863    7   16    8    9   14   29   17   15    2]
 [   0 1067   25    8   12    1    4    2   10    6]
 [  41   60  671   63   21   20   42   55   44   15]
 [  26   33   46  650   11  102   20   40   73    9]
 [  20   23   25   20  633   16   27   10   29  179]
 [ 115   91   22   81   64  298   28   35  145   13]
 [  47   20   42    3   52   15  750    1    3   25]
 [  14   34   15   13   29   18    5  834   14   52]
 [  44   38   46  137   67   66   17   32  478   49]
 [  44   27    9   19  123   19   11  142   35  580]]
313/313 [==============================] - 1s 3ms/step
Model with relu activation, mean_squared_error loss, batch size 64, learning rate 0.
              precision    recall  f1-score   support

           0       0.92      0.97      0.94       980
           1       0.96      0.98      0.97      1135
           2       0.93      0.88      0.90      1032
           3       0.92      0.90      0.91      1010
           4       0.89      0.94      0.92       982
           5       0.90      0.87      0.89       892
           6       0.92      0.95      0.93       958
           7       0.92      0.91      0.92      1028
           8       0.88      0.87      0.87       974
           9       0.90      0.88      0.89      1009

    accuracy                           0.92     10000
   macro avg       0.91      0.92      0.91     10000
weighted avg       0.92      0.92      0.92     10000

Confusion Matrix:
[[ 950    0    0    3    0    5   12    1    7    2]
 [   0 1111    2    4    1    1    6    0   10    0]
 [  18    4  910   11   21    1   16   17   30    4]
 [   2    1   24  904    1   26    4   18   22    8]
 [   1    2    3    2  927    0   10    1    5   31]
 [  16    1    6   26   11  780   17    7   17   11]
 [  19    3    6    0    9    8  908    0    5    0]
 [   6   20   19    1   13    3    1  935    6   24]
 [   7    4   10   28    9   28   16   13  843   16]
 [  12    6    1    6   44   13    2   21   12  892]]
313/313 [==============================] - 1s 3ms/step
Model with relu activation, mean_squared_error loss, batch size 64, learning rate 0.
              precision    recall  f1-score   support

           0       0.97      0.99      0.98       980
           1       0.99      0.99      0.99      1135
           2       0.95      0.95      0.95      1032
           3       0.95      0.96      0.96      1010
           4       0.96      0.96      0.96       982
           5       0.97      0.95      0.96       892
           6       0.97      0.97      0.97       958
           7       0.96      0.96      0.96      1028
           8       0.94      0.95      0.95       974
           9       0.96      0.94      0.95      1009

    accuracy                           0.96     10000
   macro avg       0.96      0.96      0.96     10000
weighted avg       0.96      0.96      0.96     10000

Confusion Matrix:
[[ 967    0    1    2    0    1    4    1    3    1]
 [   0 1120    5    0    0    3    2    1    4    0]
 [   6    0  984    5    5    2    5    9   13    3]
 [   0    1   11  971    0    6    0   12    7    2]
 [   2    0    5    0  944    1    9    1    3   17]
 [   3    1    2   13    2  849    7    3    7    5]
 [   8    2    2    0    4    6  931    0    5    0]
 [   1    5   18    4    5    1    0  982    1   11]
 [   4    0    8   14    5    5    5    6  923    4]
 [   7    4    0    9   19    3    0    5   13  949]]
313/313 [==============================] - 1s 3ms/step
Model with relu activation, mean_squared_error loss, batch size 128, learning rate 0
              precision    recall  f1-score   support
           0       0.59      0.76      0.67       980
           1       0.70      0.84      0.76      1135
           2       0.32      0.22      0.26      1032
           3       0.46      0.41      0.43      1010
           4       0.41      0.45      0.43       982
           5       0.21      0.11      0.14       892
           6       0.60      0.71      0.65       958
           7       0.61      0.70      0.65      1028
           8       0.50      0.52      0.51       974
           9       0.31      0.29      0.30      1009

    accuracy                           0.51     10000
   macro avg       0.47      0.50      0.48     10000
weighted avg       0.48      0.51      0.49     10000

Confusion Matrix:
[[747   9  19  38  32  13  63  11  18  30]
 [  3 949  81   1  34  23   5   4  34   1]
 [127  87 226  29  58  76 134  79 100 116]
 [ 65  48  97 413  33  70  16 115  95  58]
 [ 32  48  32  51 442  40  73  16  27 221]
 [152  32  70 154 107  97  52  18 137  73]
 [ 55  24   4  34  27  54 682  15  13  50]
 [ 14  62  58  17  39  38   3 715  46  36]
 [ 39  65  75  62  79  25  37  40 504  48]
 [ 25  40  38  97 219  27  79 154  39 291]]
313/313 [==============================] - 1s 3ms/step
Model with relu activation, mean_squared_error loss, batch size 128, learning rate 0
              precision    recall  f1-score   support

           0       0.89      0.96      0.92       980
           1       0.91      0.98      0.94      1135
           2       0.90      0.84      0.87      1032
           3       0.88      0.88      0.88      1010
           4       0.88      0.87      0.87       982
           5       0.86      0.79      0.82       892
           6       0.89      0.92      0.91       958
           7       0.90      0.88      0.89      1028
           8       0.87      0.82      0.84       974
           9       0.85      0.87      0.86      1009

    accuracy                           0.88     10000
   macro avg       0.88      0.88      0.88     10000
weighted avg       0.88      0.88      0.88     10000

Confusion Matrix:
[[ 940    0    5    5    0    9   12    2    6    1]
 [   0 1110    4    5    0    2    5    0    9    0]
 [  25   16  869   24   13    3   18   21   36    7]
 [   9    9   19  886    2   30    4   11   18   22]
 [   7   11    9    1  853    4   14    7    9   67]
 [  21   15    3   43   24  705   29   22   25    5]
 [  24    8    7    1   15   13  883    2    4    1]
 [   6   22   37    6   12    2    1  908    4   30]
 [  14   17   11   29   10   44   17   11  796   25]
 [  14   12    2   10   42   11    5   26   12  875]]
313/313 [==============================] - 1s 3ms/step
Model with relu activation, mean_squared_error loss, batch size 128, learning rate 0
              precision    recall  f1-score   support

           0       0.96      0.98      0.97       980
           1       0.98      0.99      0.98      1135
           2       0.95      0.94      0.94      1032
           3       0.95      0.94      0.94      1010
           4       0.94      0.96      0.95       982
           5       0.95      0.94      0.94       892
           6       0.95      0.96      0.96       958
           7       0.96      0.95      0.95      1028
           8       0.95      0.94      0.95       974
           9       0.95      0.94      0.94      1009

    accuracy                           0.95     10000
   macro avg       0.95      0.95      0.95     10000
weighted avg       0.95      0.95      0.95     10000

Confusion Matrix:
[[ 962    0    0    1    1    4    7    3    2    0]
 [   0 1120    2    2    0    0    5    0    6    0]
 [  10    1  972   11    9    1    7    9   10    2]
 [   2    2   18  946    2   16    1   10   11    2]
 [   1    1    2    0  944    0   11    0    3   20]
 [   7    2    0   16    2  836   11    5    6    7]
 [   4    3    6    1    9   10  924    0    1    0]
 [   1    8   21    2    7    0    0  974    0   15]
 [   7    1    5    7   12    6    9    8  913    6]
 [   7    5    1   10   21    5    2    8    6  944]]
313/313 [==============================] - 1s 3ms/step
Model with tanh activation, categorical_crossentropy loss, batch size 32, learning ra
              precision    recall  f1-score   support
           0       0.95      0.98      0.96       980
           1       0.96      0.98      0.97      1135
           2       0.94      0.91      0.92      1032
           3       0.93      0.91      0.92      1010
           4       0.91      0.94      0.92       982
           5       0.91      0.88      0.90       892
           6       0.93      0.95      0.94       958
           7       0.93      0.93      0.93      1028
           8       0.90      0.88      0.89       974
           9       0.91      0.90      0.90      1009

    accuracy                           0.93     10000
   macro avg       0.93      0.93      0.93     10000
weighted avg       0.93      0.93      0.93     10000

Confusion Matrix:
[[ 956    0    0    1    0    8   11    2    2    0]
 [   0 1112    2    2    0    2    4    1   12    0]
 [   8    4  935   11   10    4   13   13   27    7]
 [   3    3   16  924    2   20    5   12   15   10]
 [   2    4    5    0  920    0   10    2    6   33]
 [  10    2    3   23   10  788   16    8   24    8]
 [   8    3    4    1   11   13  914    1    3    0]
 [   3   13   21    4   10    0    0  951    1   25]
 [   9    9    7   19   10   23   14   11  860   12]
 [  12    7    1   13   41    5    1   19    5  905]]
313/313 [==============================] - 1s 3ms/step
Model with tanh activation, categorical_crossentropy loss, batch size 32, learning ra
              precision    recall  f1-score   support

           0       0.97      0.99      0.98       980
           1       0.98      0.99      0.99      1135
           2       0.97      0.97      0.97      1032
           3       0.96      0.98      0.97      1010
           4       0.98      0.96      0.97       982
           5       0.97      0.96      0.97       892
           6       0.97      0.98      0.98       958
           7       0.97      0.96      0.97      1028
           8       0.97      0.96      0.96       974
           9       0.96      0.97      0.96      1009

    accuracy                           0.97     10000
   macro avg       0.97      0.97      0.97     10000
weighted avg       0.97      0.97      0.97     10000

Confusion Matrix:
[[ 970    0    0    1    0    3    3    1    2    0]
 [   0 1125    3    0    0    1    2    1    3    0]
 [   6    2  999    2    0    2    2    8   11    0]
 [   0    0    4  986    0    6    0    7    4    3]
 [   1    0    4    1  946    0    6    2    2   20]
 [   4    1    1   12    2  857    7    0    6    2]
 [   7    3    0    2    2    4  935    2    3    0]
 [   1    7   11    4    1    0    0  992    1   11]
 [   3    1    5    7    4    4    4    4  937    5]
 [   4    4    1    8   10    2    0    3    1  976]]
313/313 [==============================] - 1s 4ms/step
Model with tanh activation, categorical_crossentropy loss, batch size 32, learning ra
              precision    recall  f1-score   support

           0       0.98      0.99      0.98       980
           1       0.99      0.99      0.99      1135
           2       0.97      0.98      0.98      1032
           3       0.97      0.98      0.98      1010
           4       0.99      0.96      0.97       982
           5       0.99      0.97      0.98       892
           6       0.98      0.97      0.98       958
           7       0.97      0.98      0.98      1028
           8       0.97      0.98      0.98       974
           9       0.97      0.98      0.97      1009

    accuracy                           0.98     10000
   macro avg       0.98      0.98      0.98     10000
weighted avg       0.98      0.98      0.98     10000

Confusion Matrix:
[[ 966    0    1    2    1    0    5    2    2    1]
 [   1 1125    3    0    0    1    2    1    2    0]
 [   2    1 1016    3    2    0    0    4    4    0]
 [   0    1    5  989    0    4    0    5    3    3]
 [   2    1    4    1  947    0    3    3    1   20]
 [   2    0    0   12    1  863    3    1    5    5]
 [   6    3    1    1    6    3  932    0    5    1]
 [   0    3   12    1    0    0    0 1010    2    0]
 [   4    0    3    3    2    0    0    3  957    2]
 [   1    3    0    4    2    1    2   11    1  984]]
313/313 [==============================] - 1s 3ms/step
Model with tanh activation, categorical_crossentropy loss, batch size 64, learning ra
              precision    recall  f1-score   support

           0       0.93      0.97      0.95       980
           0       0.93      0.97      0.95       980
           1       0.95      0.97      0.96      1135
           2       0.92      0.89      0.91      1032
           3       0.91      0.90      0.90      1010
           4       0.90      0.93      0.92       982
           5       0.89      0.86      0.87       892
           6       0.92      0.94      0.93       958
           7       0.92      0.91      0.91      1028
           8       0.89      0.87      0.88       974
           9       0.90      0.89      0.89      1009

    accuracy                           0.91     10000
   macro avg       0.91      0.91      0.91     10000
weighted avg       0.91      0.91      0.91     10000

Confusion Matrix:
[[ 948    0    1    2    0   11   12    3    3    0]
 [    0 1106    2    6    0    2    4    0   15    0]
 [  14   10  916   13   12    3   16   15   28    5]
 [   6    2   20  913    1   25    3   16   19    5]
 [   1    3    6    1  913    1   13    2    4   38]
 [   9    5    5   35    9  767   20    9   25    8]
 [  12    4    8    0   12   14  905    1    2    0]
 [   2   15   27    4   10    0    1  932    0   37]
 [  10   11    6   22   13   33   12   10  845   12]
 [  14    6    1   12   40    8    1   27    4  896]]
313/313 [==============================] - 1s 3ms/step
Model with tanh activation, categorical_crossentropy loss, batch size 64, learning ra
              precision    recall  f1-score   support

           0       0.96      0.99      0.97       980
           1       0.98      0.99      0.99      1135
           2       0.96      0.95      0.96      1032
           3       0.94      0.96      0.95      1010
           4       0.97      0.96      0.97       982
           5       0.97      0.94      0.96       892
           6       0.97      0.97      0.97       958
           7       0.96      0.95      0.96      1028
           8       0.96      0.96      0.96       974
           9       0.96      0.95      0.95      1009

    accuracy                           0.96     10000
   macro avg       0.96      0.96      0.96     10000
weighted avg       0.96      0.96      0.96     10000

Confusion Matrix:
[[ 966    0    2    1    0    3    5    1    2    0]
 [    0 1123    2    1    0    1    4    1    3    0]
 [   6    3  985   10    4    0    7    7    8    2]
 [   0    1   12  968    1    7    1   11    3    6]
 [   1    0    5    1  946    0    5    2    5   17]
 [   7    0    1   21    3  842    7    4    5    2]
 [  10    2    0    0    4    9  927    1    5    0]
 [   2    5   17    4    2    2    0  980    2   14]
 [   4    1    5    9    2    4    2    5  939    3]
 [   6    6    1   10   11    2    1    5    6  961]]
313/313 [==============================] - 1s 3ms/step
Model with tanh activation, categorical_crossentropy loss, batch size 64, learning ra
              precision    recall  f1-score   support

           0       0.98      0.99      0.99       980
           1       0.99      0.99      0.99      1135
           2       0.97      0.98      0.97      1032
           3       0.95      0.98      0.97      1010
           4       0.98      0.97      0.98       982
           5       0.98      0.97      0.97       892
           6       0.98      0.98      0.98       958
           7       0.97      0.98      0.98      1028
           8       0.99      0.95      0.97       974
           9       0.97      0.96      0.96      1009

    accuracy                           0.98     10000
   macro avg       0.98      0.98      0.98     10000
weighted avg       0.98      0.98      0.98     10000

Confusion Matrix:
[[ 973    1    0    0    0    0    1    1    3    1]
 [    0 1127    4    0    0    1    0    1    2    0]
 [   5    2 1007    5    1    1    3    4    3    1]
 [   0    0    6  994    0    4    0    3    0    3]
 [   1    0    0    1  953    0    6    3    2   16]
 [   3    0    0   16    1  863    4    1    1    3]
 [   4    3    3    1    4    5  937    0    1    0]
 [   0    4    7    5    0    0    0 1009    1    2]
 [   3    0   10   17    2    3    3    6  928    2]
 [   4    4    1   10    9    2    0   13    1  965]]
313/313 [==============================] - 1s 3ms/step
Model with tanh activation, categorical_crossentropy loss, batch size 128, learning r
              precision    recall  f1-score   support

           0       0.93      0.97      0.95       980
           1       0.94      0.97      0.96      1135
           2       0.91      0.86      0.89      1032
           3       0.89      0.89      0.89      1010
           4       0.88      0.91      0.90       982
           5       0.88      0.82      0.85       892
           6       0.91      0.94      0.92       958
           7       0.90      0.91      0.91      1028
           8       0.87      0.83      0.85       974
           9       0.88      0.87      0.87      1009

    accuracy                           0.90     10000
   macro avg       0.90      0.90      0.90     10000
weighted avg       0.90      0.90      0.90     10000

Confusion Matrix:
[[ 949    0    3    2    1    7   12    2    4    0]
 [   0 1104    2    3    1    3    3    1   18    0]
 [  11   12  890   19   18    2   14   26   34    6]
 [   6    3   23  897    1   29    5   20   13   13]
 [   0    3    6    2  896    1   19    2    6   47]
 [  11    8    7   47   14  734   19   10   33    9]
 [  18    5    7    2   11   12  900    1    2    0]
 [   3   14   26    3    9    2    1  935    2   33]
 [   8   21   10   24   14   35   18   16  813   15]
 [  13    7    3   12   52   10    3   22    8  879]]
313/313 [==============================] - 1s 3ms/step
Model with tanh activation, categorical_crossentropy loss, batch size 128, learning r
              precision    recall  f1-score   support

           0       0.95      0.99      0.97       980
           1       0.98      0.99      0.98      1135
           2       0.96      0.94      0.95      1032
           3       0.94      0.95      0.94      1010
           4       0.94      0.95      0.95       982
           5       0.94      0.92      0.93       892
           6       0.95      0.96      0.96       958
           7       0.95      0.95      0.95      1028
           8       0.93      0.93      0.93       974
           9       0.95      0.92      0.94      1009

    accuracy                           0.95     10000
   macro avg       0.95      0.95      0.95     10000
weighted avg       0.95      0.95      0.95     10000

Confusion Matrix:
[[ 966    0    0    2    0    4    6    1    1    0]
 [   0 1121    2    2    0    2    3    2    3    0]
 [   9    2  972    8    7    1    8    7   16    2]
 [   1    1    9  955    1   12    2   13   13    3]
 [   1    1    3    2  937    0    8    1    5   24]
 [   8    3    0   22    2  822   10    5   17    3]
 [   9    3    3    0    6   12  921    1    3    0]
 [   3    8   19    5    5    1    0  972    2   13]
 [   6    4    4    9    9   15   11    7  905    4]
 [  10    6    1   10   25    5    1   12    6  933]]
313/313 [==============================] - 1s 3ms/step
Model with tanh activation, categorical_crossentropy loss, batch size 128, learning r
              precision    recall  f1-score   support

           0       0.98      0.98      0.98       980
           1       0.99      0.99      0.99      1135
           2       0.99      0.97      0.98      1032
           3       0.93      0.99      0.96      1010
           4       0.98      0.97      0.97       982
           5       0.97      0.97      0.97       892
           6       0.98      0.97      0.98       958
           7       0.97      0.97      0.97      1028
           8       0.98      0.94      0.96       974
           9       0.96      0.96      0.96      1009

    accuracy                           0.97     10000
   macro avg       0.97      0.97      0.97     10000
weighted avg       0.97      0.97      0.97     10000

Confusion Matrix:
[[ 965    0    0    3    0    2    5    2    2    1]
 [   0 1124    2    4    0    1    1    1    2    0]
 [   2    0  999   15    2    0    3    5    5    1]
 [   0    0    1  996    0    4    0    6    2    1]
 [   0    1    2    1  948    0    6    6    0   18]
 [   4    0    0   12    0  865    3    0    3    5]
 [   6    3    0    2    5    7  932    1    2    0]
 [   1    3    6    6    1    0    0 1002    1    8]
 [   3    1    1   31    1    8    2    5  919    3]
 [   2    3    1    5   10    7    0    7    2  972]]
313/313 [==============================] - 1s 3ms/step
Model with tanh activation, mean_squared_error loss, batch size 32, learning rate 0.0
              precision    recall  f1-score   support

           0       0.85      0.95      0.89       980
           1       0.81      0.96      0.88      1135
           2       0.82      0.73      0.77      1032
           3       0.76      0.80      0.78      1010
           4       0.78      0.84      0.81       982
           5       0.79      0.58      0.67       892
           6       0.81      0.89      0.85       958
           7       0.81      0.87      0.84      1028
           8       0.81      0.63      0.71       974
           9       0.79      0.73      0.76      1009

    accuracy                           0.80     10000
   macro avg       0.80      0.80      0.80     10000
weighted avg       0.80      0.80      0.80     10000

Confusion Matrix:
[[ 927    2    5    8    2    6   24    5    1    0]
 [   0 1085   17    4    3    9    4    2   11    0]
 [  19   52  758   50   17    3   37   37   53    6]
 [  10   12   32  812    1   52   11   42   24   14]
 [   2   11   11    9  829    4   26    6    6   78]
 [  69   54   28   80   29  520   49   23   21   19]
 [  19   14   12    2   32   18  856    5    0    0]
 [   6   35   21    4   12    0    4  893    8   45]
 [  24   54   31   87   30   39   43   18  613   35]
 [  19   18   13   17  106   11    1   67   18  739]]
313/313 [==============================] - 1s 4ms/step
Model with tanh activation, mean_squared_error loss, batch size 32, learning rate 0.0
              precision    recall  f1-score   support

           0       0.93      0.98      0.95       980
           1       0.96      0.98      0.97      1135
           2       0.92      0.88      0.90      1032
           3       0.91      0.91      0.91      1010
           4       0.89      0.94      0.91       982
           5       0.91      0.83      0.87       892
           6       0.92      0.95      0.93       958
           7       0.90      0.91      0.90      1028
           8       0.90      0.86      0.88       974
           9       0.89      0.88      0.88      1009

    accuracy                           0.91     10000
   macro avg       0.91      0.91      0.91     10000
weighted avg       0.91      0.91      0.91     10000

Confusion Matrix:
[[ 961    0    2    1    0    3    6    3    4    0]
 [   0 1107    2    4    1    2    4    1   14    0]
 [  16    3  909   11   16    3   15   20   30    9]
 [   3    2   26  917    2   18    3   15   12   12]
 [   1    1    4    0  920    0   14    2    3   37]
 [  15    2    5   35   20  744   20   16   27    8]
 [  17    3    3    1   10    9  910    3    2    0]
 [   3   18   19    3   12    0    1  936    1   35]
 [   7   11   11   23   14   26   17   16  838   11]
 [  14    7    2   11   42   10    0   30    5  888]]
313/313 [==============================] - 1s 4ms/step
Model with tanh activation, mean_squared_error loss, batch size 32, learning rate 0.1
              precision    recall  f1-score   support

           0       0.96      0.98      0.97       980
           1       0.98      0.99      0.98      1135
           2       0.95      0.95      0.95      1032
           3       0.96      0.95      0.95      1010
           4       0.97      0.96      0.96       982
           5       0.96      0.94      0.95       892
           6       0.96      0.97      0.96       958
           7       0.94      0.95      0.95      1028
           8       0.95      0.95      0.95       974
           9       0.96      0.94      0.95      1009

    accuracy                           0.96     10000
   macro avg       0.96      0.96      0.96     10000
weighted avg       0.96      0.96      0.96     10000

Confusion Matrix:
[[ 964    0    1    1    0    3    6    4    1    0]
 [   0 1120    4    2    0    1    4    1    3    0]
 [   6    4  977    6    7    1    5   11   12    3]
 [   0    0   14  957    2   14    1   11    8    3]
 [   1    0    4    0  941    1   11    3    3   18]
 [   8    1    1    8    3  838    9    5   13    6]
 [   6    2    1    1    4    6  931    2    5    0]
 [   2   11   18    5    1    1    0  977    2   11]
 [   6    0    2   10    5    5    6    9  928    3]
 [   6    5    2   12    9    4    1   11    7  952]]
313/313 [==============================] - 1s 3ms/step
Model with tanh activation, mean_squared_error loss, batch size 64, learning rate 0.0
              precision    recall  f1-score   support

           0       0.78      0.84      0.81       980
           1       0.77      0.86      0.85      1135
           1       0.77      0.96      0.85      1135
           2       0.78      0.70      0.74      1032
           3       0.65      0.76      0.70      1010
           4       0.70      0.79      0.74       982
           5       0.55      0.10      0.17       892
           6       0.71      0.89      0.79       958
           7       0.74      0.78      0.76      1028
           8       0.66      0.66      0.66       974
           9       0.73      0.62      0.67      1009

    accuracy                           0.72     10000
   macro avg       0.71      0.71      0.69     10000
weighted avg       0.71      0.72      0.70     10000

Confusion Matrix:
[[ 825    3    5   12    6   14   84   15   12    4]
 [   0 1088    9   11    2    0    7    2   16    0]
 [  20   63  727   25   28    7   54   43   51   14]
 [  21   35   52  767   13   20   23   18   35   26]
 [   5   20   10    5  774    3   65   14   16   70]
 [ 109   78   20  255   78   89   54   46  129   34]
 [  25   11   16   12   15    3  857    2   16    1]
 [  13   50   30    9   25    6   10  799   28   58]
 [  23   51   56   68   29   17   38   23  645   24]
 [  20   18   12   23  143    4    9  125   25  630]]
313/313 [==============================] - 1s 3ms/step
Model with tanh activation, mean_squared_error loss, batch size 64, learning rate 0.0
              precision    recall  f1-score   support

           0       0.93      0.97      0.95       980
           1       0.95      0.97      0.96      1135
           2       0.89      0.86      0.87      1032
           3       0.90      0.90      0.90      1010
           4       0.86      0.92      0.89       982
           5       0.88      0.81      0.84       892
           6       0.91      0.94      0.92       958
           7       0.91      0.91      0.91      1028
           8       0.87      0.84      0.85       974
           9       0.89      0.87      0.88      1009

    accuracy                           0.90     10000
   macro avg       0.90      0.90      0.90     10000
weighted avg       0.90      0.90      0.90     10000

Confusion Matrix:
[[ 952    0    6    1    1    7    9    1    3    0]
 [   0 1098    1    5    2    1    6    2   20    0]
 [  15    8  887   21   20    2   15   21   36    7]
 [   5    1   20  907    2   32    4   12   16   11]
 [   1    5    5    0  906    1   17    1    5   41]
 [  15    4   13   42   19  722   18   15   35    9]
 [  12    4    9    1   10   19  898    2    3    0]
 [   5   14   31    4   12    1    0  933    1   27]
 [  11   15   17   20   21   24   16   20  817   13]
 [  13    7    7   12   55   13    2   20    3  877]]
313/313 [==============================] - 1s 3ms/step
Model with tanh activation, mean_squared_error loss, batch size 64, learning rate 0.1
              precision    recall  f1-score   support

           0       0.95      0.98      0.97       980
           1       0.97      0.99      0.98      1135
           2       0.95      0.93      0.94      1032
           3       0.93      0.93      0.93      1010
           4       0.93      0.96      0.95       982
           5       0.95      0.90      0.92       892
           6       0.94      0.96      0.95       958
           7       0.94      0.94      0.94      1028
           8       0.93      0.92      0.93       974
           9       0.94      0.93      0.93      1009

    accuracy                           0.94     10000
   macro avg       0.94      0.94      0.94     10000
weighted avg       0.94      0.94      0.94     10000

Confusion Matrix:
[[ 965    0    1    1    0    3    7    1    2    0]
 [   0 1120    3    3    0    1    4    2    2    0]
 [   8    7  957    8    9    1   11   11   18    2]
 [   1    0   16  941    2   16    1   12   14    7]
 [   1    1    2    2  940    0   12    2    2   20]
 [   8    2    2   22    7  803   16    5   18    9]
 [   8    3    4    3    8    8  921    1    2    0]
 [   2    7   21    2    7    2    0  970    0   17]
 [   7    5    4   15   11   10    7   13  897    5]
 [  12    5    0   11   23    4    2   10    7  935]]
313/313 [==============================] - 1s 3ms/step
Model with tanh activation, mean_squared_error loss, batch size 128, learning rate 0
              precision    recall  f1-score   support

           0       0.68      0.63      0.65       980
           1       0.64      0.96      0.77      1135
           2       0.68      0.69      0.69      1032
           3       0.56      0.58      0.57      1010
           4       0.57      0.51      0.54       982
           5       0.38      0.26      0.31       892
           6       0.63      0.54      0.58       958
           7       0.63      0.72      0.67      1028
           8       0.62      0.52      0.57       974
           9       0.48      0.48      0.48      1009

    accuracy                           0.60     10000
   macro avg       0.59      0.59      0.58     10000
weighted avg       0.59      0.60      0.59     10000

Confusion Matrix:
[[ 616   37   22   55    8   93   39   50   23   37]
 [   0 1084   25    2   12    0    4    1    6    1]
 [  23   95  714   46   23   27   46   24   33    1]
 [  34  100   50  589   24   89   23   38   19   44]
 [   3   33    4   15  505   53   57   60   20  232]
 [ 100   52   24  190   52  235   42   41   86   70]
 [  35  104  117   12   50   37  519   23   59    2]
 [  10   82   41    3   26   22    9  741   10   84]
 [  58   82   44  114   36   25   50   12  504   49]
 [  30   20    6   27  146   32   29  181   50  488]]
313/313 [==============================] - 1s 3ms/step
Model with tanh activation, mean_squared_error loss, batch size 128, learning rate 0
              precision    recall  f1-score   support

           0       0.89      0.96      0.92       980
           1       0.91      0.96      0.94      1135
           2       0.89      0.83      0.86      1032
           3       0.86      0.87      0.86      1010
           4       0.84      0.91      0.87       982
           5       0.87      0.77      0.82       892
           6       0.87      0.91      0.89       958
           7       0.89      0.89      0.89      1028
           8       0.85      0.79      0.82       974
           9       0.87      0.84      0.86      1009

    accuracy                           0.88     10000
   macro avg       0.87      0.87      0.87     10000
weighted avg       0.88      0.88      0.87     10000

Confusion Matrix:
[[ 937    1    6    5    1   10   14    1    5    0]
 [   0 1092    4    5    1    3    6    2   22    0]
 [  22   22  859   21   18    2   21   26   35    6]
 [   7    5   22  876    2   39   14   15   17   13]
 [   3    7    5    2  893    1   12    6    8   45]
 [  23   12   10   43   29  689   34   16   25   11]
 [  27    3    4    2   20   16  876    1    9    0]
 [   3   19   28   12   15    0    4  913    1   33]
 [  19   25   15   44   18   26   22   16  771   18]
 [  12    8    9   13   69    9    7   25    9  848]]
313/313 [==============================] - 1s 3ms/step
Model with tanh activation, mean_squared_error loss, batch size 128, learning rate 0
              precision    recall  f1-score   support

           0       0.95      0.98      0.96       980
           1       0.97      0.98      0.97      1135
           2       0.94      0.91      0.92      1032
           3       0.92      0.91      0.91      1010
           4       0.92      0.94      0.93       982
           5       0.90      0.89      0.90       892
           6       0.93      0.96      0.94       958
           7       0.94      0.92      0.93      1028
           8       0.91      0.89      0.90       974
           9       0.91      0.91      0.91      1009

    accuracy                           0.93     10000
   macro avg       0.93      0.93      0.93     10000
weighted avg       0.93      0.93      0.93     10000

Confusion Matrix:
[[ 961    0    1    3    0    2    9    2    2    0]
 [   0 1110    3    3    0    1    4    2   12    0]
 [  10    6  935   12   12    3   11   12   25    6]
 [   4    1   14  917    3   31    3    9   15   13]
 [   1    3    4    0  927    0   11    2    4   30]
 [   7    2    3   21    9  795   17    7   22    9]
 [  10    4    5    1    7   13  915    2    1    0]
 [   2   10   22    8   11    1    1  947    1   25]
 [   9    4    6   19   10   23   14   11  870    8]
 [   9    7    1   12   32   12    1   18    2  915]]
313/313 [==============================] - 1s 3ms/step
Model with sigmoid activation, categorical_crossentropy loss, batch size 32, learning
              precision    recall  f1-score   support

           0       0.95      0.97      0.96       980
           1       0.95      0.97      0.96      1135
           2       0.93      0.89      0.91      1032
           3       0.92      0.90      0.91      1010
           4       0.90      0.93      0.91       982
           5       0.88      0.87      0.88       892
           6       0.93      0.95      0.94       958
           7       0.92      0.92      0.92      1028
           8       0.89      0.86      0.88       974
           9       0.89      0.89      0.89      1009

    accuracy                           0.92     10000
   macro avg       0.92      0.92      0.92     10000
weighted avg       0.92      0.92      0.92     10000

Confusion Matrix:
[[ 955    0    3    1    0    8   10    1    2    0]
 [   0 1106    3    2    1    3    4    1   15    0]
 [   8   10  921   10   12    6   11   15   31    8]
 [   4    2   21  910    2   29    3   15   14   10]
 [   1    5    4    0  913    1   12    1    4   41]
 [   9    4    2   33    9  775   19    8   24    9]
 [   9    5    6    0   11   15  907    2    3    0]
 [   2   14   20    6    7    1    0  943    0   35]
 [   9   14    5   21   14   32   13   15  840   11]
 [  13    6    2   10   45    7    1   23    7  895]]
313/313 [==============================] - 1s 4ms/step
Model with sigmoid activation, categorical_crossentropy loss, batch size 32, learning
              precision    recall  f1-score   support

           0       0.96      0.98      0.97       980
           1       0.98      0.99      0.98      1135
           2       0.96      0.96      0.96      1032
           3       0.94      0.96      0.95      1010
           4       0.95      0.96      0.96       982
           5       0.96      0.94      0.95       892
           6       0.96      0.97      0.97       958
           7       0.96      0.95      0.96      1028
           8       0.95      0.95      0.95       974
           9       0.96      0.93      0.94      1009

    accuracy                           0.96     10000
   macro avg       0.96      0.96      0.96     10000
weighted avg       0.96      0.96      0.96     10000

Confusion Matrix:
[[ 964    0    0    1    0    3    9    2    1    0]
 [   0 1121    2    2    0    1    4    1    4    0]
 [   7    3  993    7    3    1    3    9    5    1]
 [   0    1    9  973    1    6    0    9    9    2]
 [   1    0    6    0  944    0    7    1    3   20]
 [   6    2    0   14    4  841   11    1   11    2]
 [   8    3    0    1    3    9  931    0    3    0]
 [   2   11   14    6    4    1    0  978    2   10]
 [   4    1    6   16    6    7    4    8  921    1]
 [   7    5    2   13   24    5    1    5   11  936]]
313/313 [==============================] - 2s 4ms/step
Model with sigmoid activation, categorical_crossentropy loss, batch size 32, learning
              precision    recall  f1-score   support

           0       0.99      0.98      0.98       980
           1       0.99      0.99      0.99      1135
           2       0.98      0.99      0.98      1032
           3       0.96      0.99      0.97      1010
           4       0.97      0.98      0.98       982
           5       0.98      0.97      0.97       892
           6       0.98      0.99      0.98       958
           7       0.97      0.98      0.98      1028
           8       0.99      0.97      0.98       974
           9       0.98      0.96      0.97      1009

    accuracy                           0.98     10000
   macro avg       0.98      0.98      0.98     10000
weighted avg       0.98      0.98      0.98     10000

Confusion Matrix:
[[ 962    0    1    2    2    1    4    3    2    3]
 [   0 1127    2    1    0    1    2    1    1    0]
 [   2    2 1017    3    2    0    1    4    1    0]
 [   1    0    4  995    0    2    0    5    3    0]
 [   0    0    1    1  965    0    5    3    0    7]
 [   3    1    0   14    2  862    3    2    2    3]
 [   3    2    1    0    3    4  945    0    0    0]
 [   2    5    7    5    0    0    0 1006    2    1]
 [   1    1    3   10    0    4    3    5  946    1]
 [   0    2    0   10   16    4    2    5    1  969]]
313/313 [==============================] - 1s 3ms/step
Model with sigmoid activation, categorical_crossentropy loss, batch size 64, learning
              precision    recall  f1-score   support

           0       0.93      0.97      0.95       980
           1       0.94      0.97      0.96      1135
           2       0.93      0.89      0.91      1032
           3       0.90      0.90      0.90      1010
           4       0.89      0.93      0.91       982
           5       0.88      0.86      0.87       892
           6       0.93      0.94      0.94       958
           7       0.92      0.90      0.91      1028
           8       0.88      0.84      0.86       974
           9       0.88      0.88      0.88      1009

    accuracy                           0.91     10000
   macro avg       0.91      0.91      0.91     10000
weighted avg       0.91      0.91      0.91     10000

Confusion Matrix:
[[ 951    0    5    1    0    8    9    2    4    0]
 [   0 1103    2    3    0    3    4    1   19    0]
 [  15    9  920   10   17    1   11   14   26    9]
 [   4    2   20  904    1   30    4   18   17   10]
 [   1    6    3    1  915    2   10    1    7   36]
 [  11    4    2   36   13  763   18    8   29    8]
 [  14    3    5    1   12   15  905    1    2    0]
 [   3   20   22    4    9    1    0  929    2   38]
 [  10   20   10   30   13   35   12   12  817   15]
 [  16    6    3   13   43    8    0   27    7  886]]
313/313 [==============================] - 1s 3ms/step
Model with sigmoid activation, categorical_crossentropy loss, batch size 64, learning
              precision    recall  f1-score   support

           0       0.96      0.98      0.97       980
           1       0.97      0.98      0.98      1135
           2       0.94      0.93      0.93      1032
           3       0.93      0.92      0.93      1010
           4       0.93      0.94      0.94       982
           5       0.94      0.90      0.92       892
           6       0.94      0.96      0.95       958
           7       0.95      0.94      0.94      1028
           8       0.91      0.93      0.92       974
           9       0.92      0.93      0.92      1009

    accuracy                           0.94     10000
   macro avg       0.94      0.94      0.94     10000
weighted avg       0.94      0.94      0.94     10000

Confusion Matrix:
[[ 959    0    0    2    0    4    9    1    3    2]
 [   0 1112    4    2    1    1    4    2    9    0]
 [   7    5  956    9    8    2   11    9   20    5]
 [   0    3   22  929    1   14    4   14   19    4]
 [   1    1    4    1  926    0   10    2    4   33]
 [   8    4    5   26   12  800   11    4   17    5]
 [   8    3    4    0    7   13  917    2    4    0]
 [   1    9   20    4    7    0    0  964    1   22]
 [   4    1    5   16    9   15   11    6  901    6]
 [  10    7    1    9   25    3    1   10    9  934]]
313/313 [==============================] - 1s 3ms/step
Model with sigmoid activation, categorical_crossentropy loss, batch size 64, learning
              precision    recall  f1-score   support

           0       0.97      0.99      0.98       980
           1       0.98      0.99      0.99      1135
           2       0.98      0.97      0.98      1032
           3       0.97      0.99      0.98      1010
           4       0.96      0.98      0.97       982
           5       0.98      0.96      0.97       892
           6       0.97      0.97      0.97       958
           7       0.98      0.97      0.97      1028
           8       0.97      0.97      0.97       974
           9       0.97      0.96      0.96      1009

    accuracy                           0.97     10000
   macro avg       0.97      0.97      0.97     10000
weighted avg       0.97      0.97      0.97     10000

Confusion Matrix:
[[ 973    0    1    1    1    1    1    1    1    0]
 [   0 1124    2    3    0    2    2    1    1    0]
 [   3    3 1003    4    2    1    5    5    6    0]
 [   0    0    1  995    0    1    0    6    3    4]
 [   2    0    1    0  961    1    7    1    0    9]
 [   5    0    1   14    2  854    7    1    6    2]
 [   7    3    1    1    6    4  932    1    3    0]
 [   2   12    9    0    1    1    0  995    2    6]
 [   7    1    0    5    5    4    3    3  941    5]
 [   5    3    1    7   18    2    0    4    4  965]]
313/313 [==============================] - 1s 3ms/step
Model with sigmoid activation, categorical_crossentropy loss, batch size 128, learnin
              precision    recall  f1-score   support

           0       0.92      0.97      0.94       980
           1       0.92      0.97      0.94      1135
           2       0.91      0.86      0.89      1032
           3       0.88      0.88      0.88      1010
           4       0.88      0.92      0.90       982
           5       0.86      0.82      0.84       892
           6       0.91      0.93      0.92       958
           7       0.89      0.90      0.90      1028
           8       0.87      0.83      0.85       974
           9       0.87      0.86      0.87      1009

    accuracy                           0.89     10000
   macro avg       0.89      0.89      0.89     10000
weighted avg       0.89      0.89      0.89     10000

Confusion Matrix:
[[ 946    1    3    2    1   11   11    2    3    0]
 [    0 1101    4    2    1    2    4    2   19    0]
 [  15   22  889   18   17    0   13   21   30    7]
 [   7    3   18  884    0   43    5   22   19    9]
 [   1    6    4    0  907    1   13    3    6   41]
 [  16    6    6   51   13  730   22    8   30   10]
 [  12    4   10    2   15   18  890    3    4    0]
 [   2   19   28    6    9    0    1  924    0   39]
 [  13   24    9   31   15   32   13   13  804   20]
 [  12   11    5    9   54    9    1   35    5  868]]
313/313 [==============================] - 1s 3ms/step
Model with sigmoid activation, categorical_crossentropy loss, batch size 128, learnir
              precision    recall  f1-score   support

           0       0.95      0.98      0.96       980
           1       0.96      0.98      0.97      1135
           2       0.94      0.90      0.92      1032
           3       0.90      0.91      0.91      1010
           4       0.93      0.93      0.93       982
           5       0.91      0.87      0.89       892
           6       0.93      0.95      0.94       958
           7       0.94      0.93      0.93      1028
           8       0.90      0.88      0.89       974
           9       0.91      0.92      0.91      1009

    accuracy                           0.93     10000
   macro avg       0.93      0.93      0.93     10000
weighted avg       0.93      0.93      0.93     10000

Confusion Matrix:
[[ 961    0    2    1    1    4    9    1    1    0]
 [    0 1113    2    3    1    1    4    2    9    0]
 [  11    7  925   17    5    4   12   10   33    8]
 [   2    0   19  924    1   23    4   10   18    9]
 [   1    4    3    1  917    0   10    2    5   39]
 [   8    1    2   34    9  780   15   10   25    8]
 [   9    4    4    0    7   15  914    3    2    0]
 [   3   13   17    7    7    2    0  953    1   25]
 [   6   10    6   26   10   28   12   11  858    7]
 [  11    6    1   13   30    4    0   14    5  925]]
313/313 [==============================] - 1s 3ms/step
Model with sigmoid activation, categorical_crossentropy loss, batch size 128, learnir
              precision    recall  f1-score   support

           0       0.97      0.98      0.98       980
           1       0.99      0.99      0.99      1135
           2       0.97      0.97      0.97      1032
           3       0.96      0.98      0.97      1010
           4       0.98      0.97      0.97       982
           5       0.98      0.95      0.96       892
           6       0.97      0.98      0.97       958
           7       0.97      0.96      0.96      1028
           8       0.95      0.98      0.96       974
           9       0.96      0.95      0.96      1009

    accuracy                           0.97     10000
   macro avg       0.97      0.97      0.97     10000
weighted avg       0.97      0.97      0.97     10000

Confusion Matrix:
[[ 963    1    2    2    0    3    5    2    1    1]
 [    0 1122    1    2    0    2    5    1    2    0]
 [   2    2 1000    3    3    0    4    6   12    0]
 [   1    0    4  988    0    1    0    5    7    4]
 [   2    0    4    1  949    0    2    4    2   18]
 [   5    0    2   15    2  849   10    2    5    2]
 [   7    2    0    0    6    4  935    1    3    0]
 [   2    4   11    5    4    0    0  987    6    9]
 [   6    0    2    4    0    3    3    4  951    1]
 [   4    4    1    7    4    6    2    8   10  963]]
313/313 [==============================] - 1s 3ms/step
Model with sigmoid activation, mean_squared_error loss, batch size 32, learning rate
              precision    recall  f1-score   support

           0       0.85      0.95      0.90       980
           1       0.82      0.96      0.89      1135
           2       0.87      0.78      0.82      1032
           3       0.85      0.82      0.83      1010
           4       0.77      0.85      0.80       982
           5       0.80      0.61      0.69       892
           6       0.83      0.89      0.86       958
           7       0.82      0.85      0.84      1028
           8       0.83      0.74      0.78       974
           9       0.78      0.75      0.77      1009

    accuracy                           0.82     10000
   macro avg       0.82      0.82      0.82     10000
weighted avg       0.82      0.82      0.82     10000

Confusion Matrix:
[[ 928    0    4    5    1   12   25    2    2    1]
 [   0 1089    5    6    6    3    4    1   20    1]
 [  33   35  804   27   21    1   36   25   35   15]
 [  10   19   18  825    6   58   11   27   20   16]
 [   7   12    7    0  830    2   28    6    6   84]
 [  53   40   17   57   48  540   36   44   35   22]
 [  23   17   19    1   19   17  855    2    5    0]
 [   7   39   25    5   22    0    5  873    7   45]
 [  16   51   13   41   38   32   19   16  721   27]
 [  16   21   11    8   91   13    9   63   20  757]]
313/313 [==============================] - 1s 4ms/step
Model with sigmoid activation, mean_squared_error loss, batch size 32, learning rate
              precision    recall  f1-score   support

           0       0.93      0.98      0.95       980
           1       0.95      0.97      0.96      1135
           2       0.92      0.87      0.89      1032
           3       0.91      0.90      0.90      1010
           4       0.89      0.93      0.91       982
           5       0.91      0.83      0.87       892
           6       0.92      0.95      0.93       958
           7       0.91      0.91      0.91      1028
           8       0.87      0.86      0.87       974
           9       0.89      0.89      0.89      1009

    accuracy                           0.91     10000
   macro avg       0.91      0.91      0.91     10000
weighted avg       0.91      0.91      0.91     10000

Confusion Matrix:
[[ 956    0    3    1    0    3   12    2    3    0]
 [   0 1102    3    3    1    3    4    2   17    0]
 [  15    8  896   14   19    1   10   24   34   11]
 [   4    1   22  908    2   22    6   18   19    8]
 [   1    5    5    0  912    2   10    1    3   43]
 [  11    5    6   39   17  738   22   12   33    9]
 [  14    4    4    0   11   11  911    2    1    0]
 [   5   20   23    6    8    1    1  931    0   33]
 [  11   14   11   23   15   20   17   15  837   11]
 [  14    7    2    8   40   10    1   21   12  894]]
313/313 [==============================] - 1s 4ms/step
Model with sigmoid activation, mean_squared_error loss, batch size 32, learning rate
              precision    recall  f1-score   support

           0       0.96      0.98      0.97       980
           1       0.97      0.98      0.98      1135
           2       0.95      0.91      0.93      1032
           3       0.92      0.93      0.92      1010
           4       0.92      0.95      0.93       982
           5       0.93      0.88      0.91       892
           6       0.94      0.95      0.95       958
           7       0.93      0.93      0.93      1028
           8       0.90      0.92      0.91       974
           9       0.92      0.91      0.92      1009

    accuracy                           0.94     10000
   macro avg       0.94      0.94      0.94     10000
weighted avg       0.94      0.94      0.94     10000

Confusion Matrix:
[[ 962    0    0    2    1    2    8    3    1    1]
 [   0 1113    3    2    1    1    4    2    9    0]
 [   7    5  943   14   10    3    9   11   26    4]
 [   0    0   13  940    4   16    0   10   20    7]
 [   1    2    2    1  932    0    9    2    4   29]
 [   8    1    1   30   10  786   15    9   23    9]
 [   8    4    6    2    8   14  910    3    3    0]
 [   2    9   21    4    7    1    0  959    1   24]
 [   5    5    3   16   10   14    8   14  896    3]
 [   8    6    2   12   31    6    0   13   10  921]]
313/313 [==============================] - 1s 3ms/step
Model with sigmoid activation, mean_squared_error loss, batch size 64, learning rate
              precision    recall  f1-score   support

           0       0.78      0.86      0.82       980
           1       0.66      0.97      0.78      1135
           2       0.70      0.57      0.62      1032
           3       0.72      0.80      0.75      1010
           4       0.71      0.76      0.73       982
           5       0.64      0.49      0.55       892
           6       0.72      0.81      0.77       958
           7       0.78      0.82      0.80      1028
           8       0.80      0.39      0.53       974
           9       0.68      0.60      0.64      1009

    accuracy                           0.71     10000
   macro avg       0.72      0.71      0.70     10000
weighted avg       0.72      0.71      0.70     10000

Confusion Matrix:
[[ 844    4   30    8    2   32   36   15    7    2]
 [   0 1101    9   14    1    3    5    0    0    2]
 [  31  146  584   53   35    7  104   36    9   27]
 [  26   33   26  803    2   28   19   29   18   26]
 [   7   47    6    3  745   14   30   10    7  113]
 [  84   85    4  129   44  438   32   19   27   30]
 [  46   37   41    1   16   32  778    3    3    1]
 [   8   77   23    6   10    3    3  844    1   53]
 [  29  116  107   84   43  105   41   29  382   38]
 [  13   31    9   20  152   25   27   98   25  609]]
313/313 [==============================] - 1s 3ms/step
Model with sigmoid activation, mean_squared_error loss, batch size 64, learning rate
              precision    recall  f1-score   support

           0       0.91      0.97      0.94       980
           1       0.92      0.97      0.95      1135
           2       0.91      0.85      0.88      1032
           3       0.89      0.89      0.89      1010
           4       0.87      0.92      0.90       982
           5       0.88      0.79      0.83       892
           6       0.90      0.94      0.92       958
           7       0.90      0.90      0.90      1028
           8       0.87      0.82      0.85       974
           9       0.88      0.88      0.88      1009

    accuracy                           0.89     10000
   macro avg       0.89      0.89      0.89     10000
weighted avg       0.89      0.89      0.89     10000

Confusion Matrix:
[[ 955    0    1    1    1    4   14    1    3    0]
 [   0 1100    4    3    1    2    5    2   18    0]
 [  14   18  881   17   17    1   16   23   34   11]
 [   6    5   27  899    1   29    5   16   15    7]
 [   0    5    3    0  904    1   15    2    5   47]
 [  23   11    5   39   29  701   26   15   31   12]
 [  20    3    5    2   13   12  896    5    2    0]
 [   3   25   21    6    9    0    0  926    2   36]
 [  13   18   15   28   16   33   18   17  803   13]
 [  14    8    6   11   46   10    0   25    6  883]]
313/313 [==============================] - 1s 3ms/step
Model with sigmoid activation, mean_squared_error loss, batch size 64, learning rate
              precision    recall  f1-score   support

           0       0.94      0.99      0.96       980
           1       0.97      0.98      0.97      1135
           2       0.93      0.91      0.92      1032
           3       0.93      0.91      0.92      1010
           4       0.93      0.94      0.93       982
           5       0.92      0.87      0.89       892
           6       0.93      0.96      0.95       958
           7       0.92      0.92      0.92      1028
           8       0.91      0.89      0.90       974
           9       0.91      0.91      0.91      1009

    accuracy                           0.93     10000
   macro avg       0.93      0.93      0.93     10000
weighted avg       0.93      0.93      0.93     10000

Confusion Matrix:
[[ 966    0    1    1    0    2    7    1    2    0]
 [   0 1110    3    3    1    1    4    2   11    0]
 [  11    6  939    9    9    2   11   16   22    7]
 [   3    1   25  919    0   24    3   14   12    9]
 [   2    3    6    0  920    0   11    2    4   34]
 [   8    1    2   28    8  779   19   10   29    8]
 [  11    3    5    0    7    9  919    2    2    0]
 [   3    8   21    4    9    1    0  950    0   32]
 [   9   11    6   21   11   23   10   15  865    3]
 [  15    5    0    7   25    9    0   18    8  922]]
313/313 [==============================] - 1s 3ms/step
Model with sigmoid activation, mean_squared_error loss, batch size 128, learning rate
              precision    recall  f1-score   support

           0       0.64      0.85      0.73       980
           1       0.61      0.86      0.71      1135
           2       0.56      0.37      0.45      1032
           3       0.55      0.45      0.50      1010
           4       0.63      0.47      0.53       982
           5       0.30      0.29      0.29       892
           6       0.64      0.70      0.67       958
           7       0.61      0.72      0.66      1028
           8       0.49      0.40      0.44       974
           9       0.46      0.42      0.44      1009

    accuracy                           0.56     10000
   macro avg       0.55      0.55      0.54     10000
weighted avg       0.55      0.56      0.55     10000

Confusion Matrix:
[[832  11  16   4  20  42  17  22   5  11]
 [  2 981  26  39   1  48  15  14   5   4]
 [ 70 189 381  19  39  12 162  37 108  15]
 [ 44 100  57 459  13 136  69  43  64  25]
 [  7  37  67  25 457  88  37  46  43 175]
 [204  65  15 106  17 258  49  25  57  96]
 [ 57  52  34   5  50  31 668  12  45   4]
 [  4  60  11   7  23  17   6 744  34 122]
 [ 55  92  61 114   8 154  15  48 387  40]
 [ 29  29  11  60 103  75  11 231  41 419]]
313/313 [==============================] - 1s 3ms/step
Model with sigmoid activation, mean_squared_error loss, batch size 128, learning rate
              precision    recall  f1-score   support

           0       0.90      0.97      0.93       980
           1       0.91      0.97      0.94      1135
           2       0.90      0.83      0.86      1032
           3       0.87      0.86      0.87      1010
           4       0.83      0.90      0.87       982
           5       0.86      0.75      0.80       892
           6       0.90      0.92      0.91       958
           7       0.87      0.89      0.88      1028
           8       0.86      0.80      0.83       974
           9       0.86      0.83      0.85      1009

    accuracy                           0.88     10000
   macro avg       0.88      0.87      0.87     10000
weighted avg       0.88      0.88      0.87     10000

Confusion Matrix:
[[ 947    0    5    3    0    5   11    2    7    0]
 [   0 1104    1    4    0    2    4    3   17    0]
 [  16   19  860   19   22    1   22   34   34    5]
 [   6    7   22  871    4   39   11   20   21    9]
 [   2    5    3    1  886    4   16    2    9   54]
 [  25   15   11   47   42  671   22   22   31    6]
 [  23    6    9    0   12   17  886    3    2    0]
 [   5   28   23    4   11    3    3  912    2   37]
 [  19   28   13   34   23   25   13   14  784   21]
 [  12    7   12   15   62   17    1   38    6  839]]
313/313 [==============================] - 1s 3ms/step
Model with sigmoid activation, mean_squared_error loss, batch size 128, learning rate
              precision    recall  f1-score   support

           0       0.93      0.98      0.96       980
           1       0.96      0.98      0.97      1135
```

```
1
```

Regression

DATASET03 - AUTO-MPG

1. Load and preprocess the given data.
2. Build ANN model with regularization and skip connections and train it on the given data.
3. Analyze ANN model performance with different batch sizes (test of 3 different batch size) and learning rates (3 different learning rates).
4. Plot mse, mae and rmse for different batch size and learning rates.

Loading and Preprocessing the data

```
data = pd.read_csv('/content/auto-mpg.csv')
```

```
data.head(5)
```

	mpg	cylinders	displacement	horsepower	weight	acceleration	model year	origin	car name
0	18.0	8	307.0	130	3504	12.0	70	1	chevrolet chevelle malibu
1	15.0	8	350.0	165	3693	11.5	70	1	buick skylark 320
2	18.0	8	318.0	150	3436	11.0	70	1	plymouth satellite
3	16.0	8	304.0	150	3433	12.0	70	1	amc rebel sst

```
data['car name'].nunique()
```

```
305
```

```
data.shape
```

```
(398, 9)
```

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 398 entries, 0 to 397
Data columns (total 9 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   mpg           398 non-null    float64
 1   cylinders     398 non-null    int64  
 2   displacement  398 non-null    float64
 3   horsepower    398 non-null    object 
 4   weight        398 non-null    int64  
 5   acceleration  398 non-null    float64
 6   model year    398 non-null    int64  
 7   origin        398 non-null    int64  
 8   car name      398 non-null    object 
dtypes: float64(3), int64(4), object(2)
memory usage: 28.1+ KB
```

```
# Convert 'horsepower' column to numeric
data['horsepower'] = pd.to_numeric(data['horsepower'], errors='coerce')
```

```
data.isnull().sum()
```

```
mpg             0
cylinders       0
displacement    0
horsepower      6
weight          0
acceleration    0
model year      0
origin          0
car name        0
dtype: int64
```

```
# Impute missing values with the mean for each column
data['horsepower'] = data['horsepower'].fillna(data['horsepower'].median())
```

```
data.isnull().sum()
```

```
mpg             0
cylinders       0
displacement    0
horsepower      0
weight          0
acceleration    0
model year      0
origin          0
car name        0
dtype: int64
```

Car name is of no use

```
data = data.drop('car name', axis = 1)
```

```
data.shape
```

```
(398, 8)
```

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler

# Split the data into features (X) and target (y)
X = data.drop(['mpg'], axis=1)
y = data['mpg']

# Split the data into train and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Standardize features
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

Build an ANN Model with Regularization and Skip Connections:

```
import tensorflow as tf
from tensorflow.keras import layers

def create_model():
    input_layer = layers.Input(shape=(X_train.shape[1],))

    x = layers.Dense(64, activation='relu', kernel_regularizer=tf.keras.regularizers.l2(0.001))(input_
    x = layers.BatchNormalization()(x)

    # Add a Dense layer to match input and intermediate layer dimensions
    identity = layers.Dense(64, activation='linear')(input_layer)

    # Create a skip connection by adding the input to the intermediate layer
    x = layers.Add()([x, identity])

    x = layers.Dense(64, activation='relu', kernel_regularizer=tf.keras.regularizers.l2(0.001))(x)
    x = layers.BatchNormalization()(x)

    output_layer = layers.Dense(1)(x)  # Output layer for regression

    model = tf.keras.Model(inputs=input_layer, outputs=output_layer)

    return model

model = create_model()
```

```
model
```

```
<keras.src.engine.functional.Functional at 0x7fc01017bc10>
```

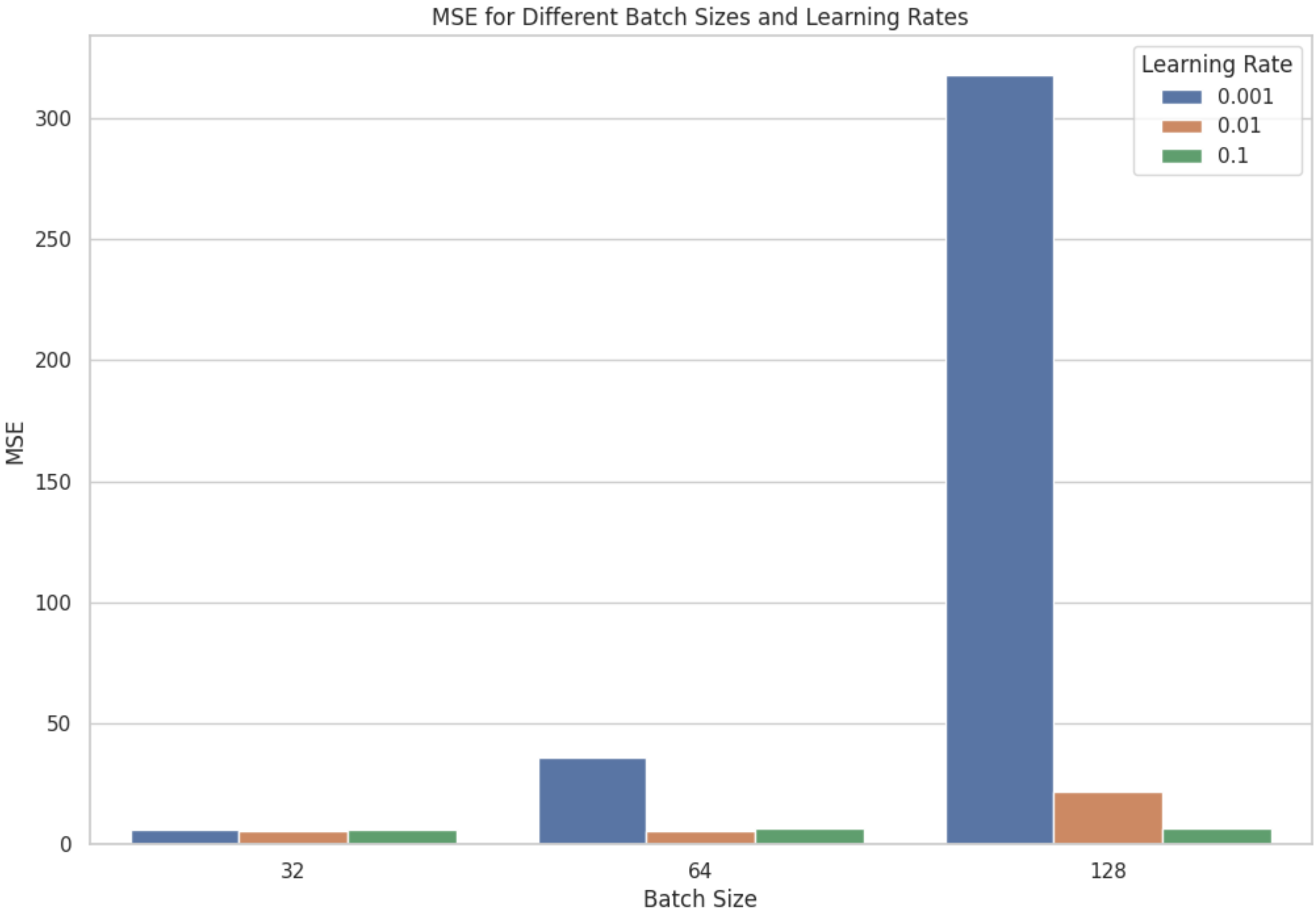
Train the ANN Model with Different Batch Sizes and Learning Rates:

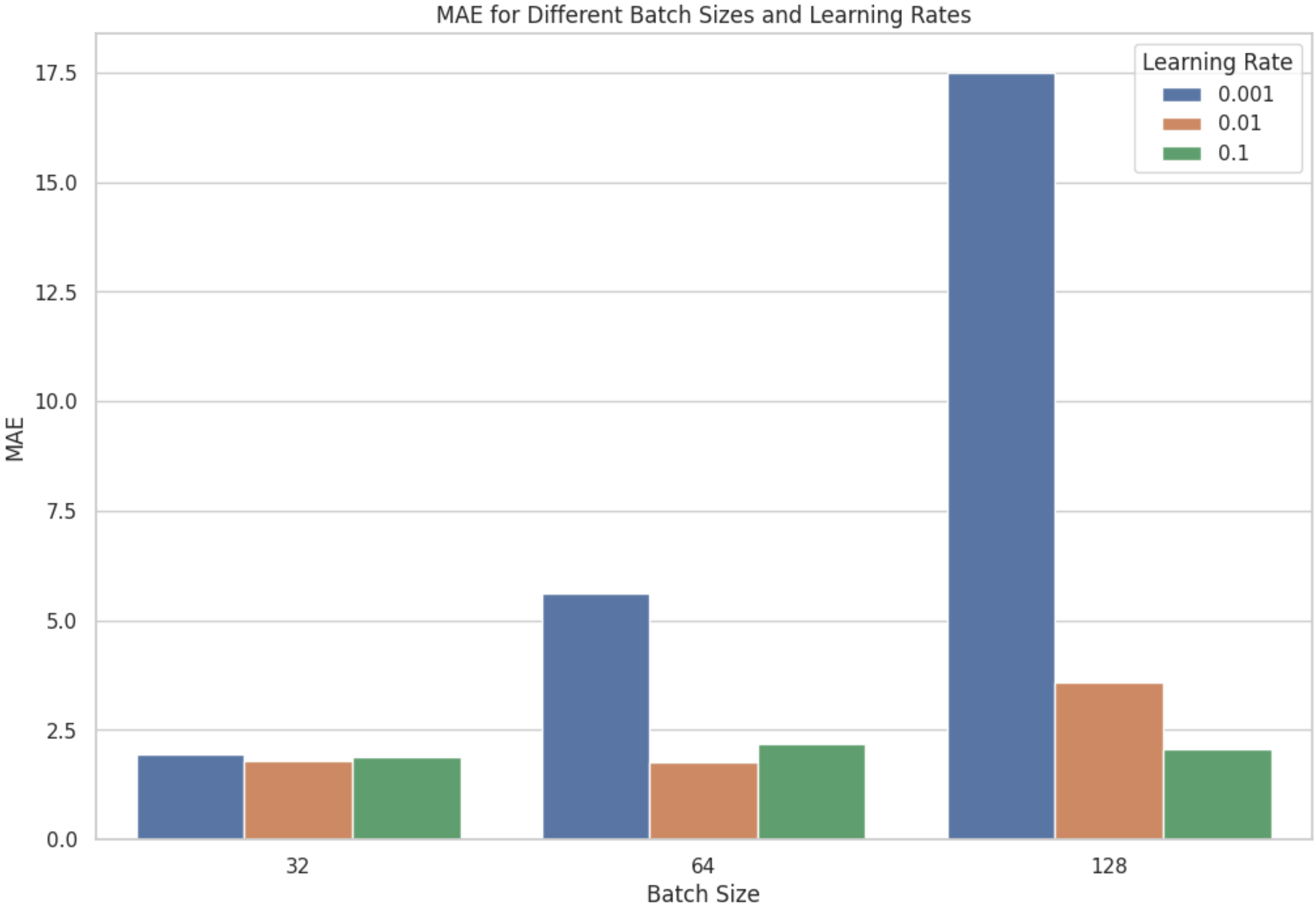
```
from tensorflow.keras.losses import MeanSquaredError, MeanAbsoluteError
from tensorflow.keras.optimizers import Adam
from sklearn.metrics import mean_squared_error, mean_absolute_error

# Define a function to train and evaluate the model with different parameters
def train_and_evaluate(batch_size, learning_rate):
    model = create_model()  # Create a new model for each iteration
    optimizer = Adam(learning_rate=learning_rate)
    model.compile(optimizer=optimizer, loss='mean_squared_error', metrics=['mae', 'mse'])

    history = model.fit(X_train, y_train, batch_size=batch_size, epochs=100, validation_split=0.2, ver

    y_pred = model.predict(X_test)
    mse = mean_squared_error(y_test, y_pred)
    mae = mean_absolute_error(y_test, y_pred)
    rmse = np.sqrt(mse)

    return mse, mae, rmse

# Define lists to store results for different batch sizes and learning rates
batch_sizes = [32, 64, 128]
learning_rates = [0.001, 0.01, 0.1]
results = []

# Iterate through different batch sizes and learning rates
for batch_size in batch_sizes:
    for learning_rate in learning_rates:
        mse, mae, rmse = train_and_evaluate(batch_size, learning_rate)
        results.append((batch_size, learning_rate, mse, mae, rmse))

# Convert results to a DataFrame for analysis
results_df = pd.DataFrame(results, columns=['Batch Size', 'Learning Rate', 'MSE', 'MAE', 'RMSE'])
```

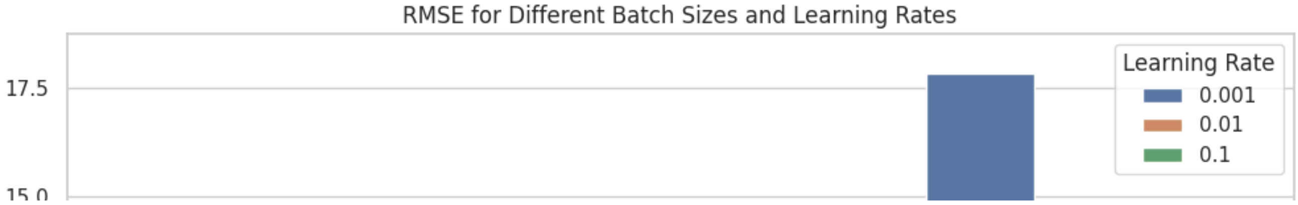
```
3/3 [==============================] - 0s 6ms/step
3/3 [==============================] - 0s 4ms/step
3/3 [==============================] - 0s 6ms/step
WARNING:tensorflow:5 out of the last 13 calls to <function Model.make_predict_function.<locals>.predict_function at 0x7fc00811af80>
3/3 [==============================] - 0s 8ms/step
WARNING:tensorflow:5 out of the last 13 calls to <function Model.make_predict_function.<locals>.predict_function at 0x7fc00811acb0>
3/3 [==============================] - 0s 5ms/step
3/3 [==============================] - 0s 5ms/step
3/3 [==============================] - 0s 5ms/step
3/3 [==============================] - 0s 5ms/step
3/3 [==============================] - 0s 4ms/step
```

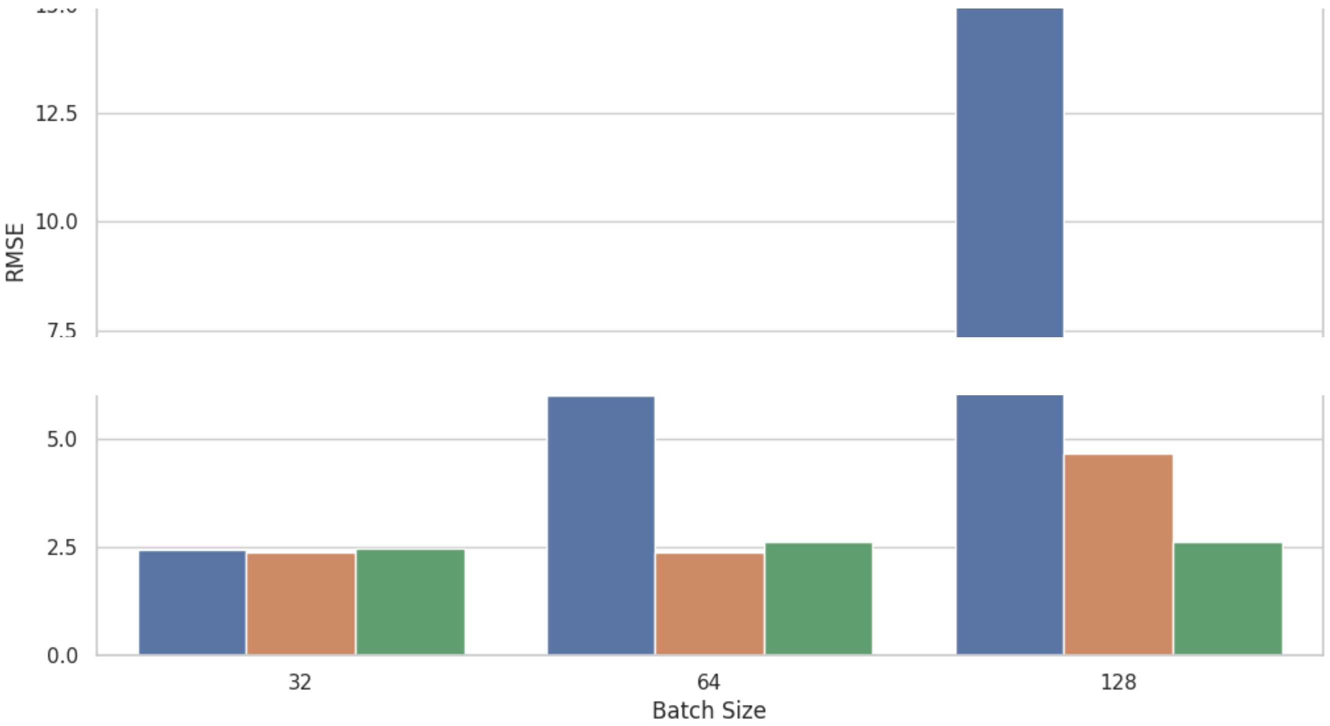
Plot MSE, MAE, and RMSE for Different Batch Sizes and Learning Rates:

```
import matplotlib.pyplot as plt
import seaborn as sns

# Plot MSE, MAE, and RMSE for different batch sizes and learning rates
plt.figure(figsize=(12, 8))
sns.set(style="whitegrid")
sns.barplot(x="Batch Size", y="MSE", hue="Learning Rate", data=results_df)
plt.title("MSE for Different Batch Sizes and Learning Rates")
plt.show()

plt.figure(figsize=(12, 8))
sns.set(style="whitegrid")
sns.barplot(x="Batch Size", y="MAE", hue="Learning Rate", data=results_df)
plt.title("MAE for Different Batch Sizes and Learning Rates")
plt.show()

plt.figure(figsize=(12, 8))
sns.set(style="whitegrid")
sns.barplot(x="Batch Size", y="RMSE", hue="Learning Rate", data=results_df)
plt.title("RMSE for Different Batch Sizes and Learning Rates")
plt.show()
```


MSE for Different Batch Sizes and Learning Rates
Learning Rate
0.001
0.01
0.1
MSE
300
250
200
150
100
50
0
32
64
128
Batch Size


MAE for Different Batch Sizes and Learning Rates
Learning Rate
0.001
0.01
0.1
MAE
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
32
64
128
Batch Size


RMSE for Different Batch Sizes and Learning Rates
Learning Rate
0.001
0.01
0.1
17.5
15.0



1