Lesson 4

Use this model to solve this assignment:

```
# 6.3 ANN Library
import tensorflow as tf
from tensorflow.keras.datasets import mnist
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Flatten
from tensorflow.keras.utils import to_categorical

# Load the dataset
(train_images, train_labels), (test_images, test_labels) =
mnist.load_data()

# Normalize the images to have pixel values between 0 and 1
train_images = train_images / 255.0
test_images = test_images / 255.0

# Convert labels to one-hot encoded vectors
train_labels = to_categorical(train_labels)
test_labels = to_categorical(test_labels)

# Define the model architecture
model = Sequential([
    Flatten(input_shape=(28, 28)),  # Flattens the input
    Dense(10, activation='Softmax')  # A softmax layer with 10 output
units for each digit
])

# Compile the model
model.compile(optimizer='adam',
              loss='categorical_crossentropy',
              metrics=['accuracy'])

# Train the model
model.fit(train_images, train_labels, epochs=5, batch_size=32)

# Evaluate the model
test_loss, test_acc = model.evaluate(test_images, test_labels)
print(f'Test accuracy: {test_acc}')
```

Task 1: Experiment with Different Model Architectures

- Add at least two more Dense layers to the existing model architecture. Experiment with different numbers of neurons in these layers (e.g., 64, 128).
- Include a Dropout layer with a dropout rate of 0.2 or 0.5 after each Dense layer to see if it helps in reducing overfitting.
- Compare the performance (accuracy and loss) of the new models with the original model. Reflect on how the changes in the architecture influenced the model's performance.

Task 2: Adjust Training Parameters

- Train the original model with different numbers of epochs (e.g., 10, 20) and different batch sizes (e.g., 64, 128) to observe the effects on training time and accuracy.
- Change the optimizer from 'adam' to 'SGD' and 'RMSprop', and compare the model's performance using these different optimizers. Discuss the differences in training behavior and final accuracy among the optimizers.

Task 3: Implement and Evaluate a Model for a Different Dataset

- Train the model on Fashion MNIST and CIFAR-10 datasets for a classification task.
- Load and preprocess the data, adjusting the input shape if necessary (e.g., for CIFAR-10, the input shape would be `(32, 32, 3)`).
- Modify the model architecture as necessary to accommodate the new dataset (e.g., add layers, change activation functions).
- Train the model on the new dataset and evaluate its performance on the test set. Compare the results with those obtained on the MNIST dataset and discuss any challenges or interesting observations made during the process.

Interview Questions

- *Question*: Imagine you are tasked with designing an ANN-based system for a self-driving car that needs to process real-time data from multiple sensors to navigate through urban environments. Describe the architecture of the neural network you would propose. Include details on the type of layers, the number of nodes in each layer, and the activation functions you would use. How would you ensure that the network can effectively handle varying lighting and weather conditions?
- Question: You are involved in creating an ANN model to assist in diagnosing diseases from complex medical images, such as MRIs or CT scans. Outline the design of your ANN, specifying the types of layers and activation functions that would be most suitable for this task. How would you train your network to differentiate between very subtle variations in medical images that indicate different stages of a disease? Describe the loss function you would choose and the rationale behind this choice, considering the critical nature of accurate medical diagnostics.