Student Information

Name: Satish Veduruvada

ID: 700756488

CRN : 23850

Question 1: Tensor Manipulations & Reshaping

```python
import tensorflow as tf
import numpy as np
tf.random.set_seed(42)
# Create random tensor
random_tensor = tf.random.normal([4, 6])
print(random_tensor)
# Find rank and shape
print(tf.rank(random_tensor).numpy())
print(random_tensor.shape)
# Reshape and transpose
reshaped_tensor = tf.reshape(random_tensor, [2, 3, 4])
print(reshaped_tensor)
transposed_tensor = tf.transpose(reshaped_tensor, perm=[1, 0, 2])
print(transposed_tensor)
# Broadcasting
small_tensor = tf.constant([[1.0, 2.0, 3.0, 4.0]])
print(small_tensor)
broadcasted_result = small_tensor + random_tensor[:, :4]
```

```
print(broadcasted_result)
```

Expected Output

- Random tensor with shape (4, 6)
- Rank and shape of the tensor
- Reshaped tensor with shape (2, 3, 4)
- Transposed tensor with shape (3, 2, 4)
- Broadcasted result with shape (4, 4)

Explanation

- Tensor manipulation and reshaping using TensorFlow
- Broadcasting and tensor operations

Question 2: Loss Functions & Hyperparameter Tuning

```python
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
# Load iris dataset
iris = load_iris()
X = iris.data
y = iris.target
# Split dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

```
# Define model architecture

model = Sequential([

    Dense(64, activation='relu', input_shape=(4,)),

    Dense(32, activation='relu'),

    Dense(3, activation='softmax')

])

# Compile model with different loss functions

model.compile(loss='sparse_categorical_crossentropy', optimizer='adam', metrics=['accuracy'])

# Train model

model.fit(X_train, y_train, epochs=10, batch_size=32, validation_data=(X_test, y_test))
```

Expected Output

- Model architecture and compilation

- Training and validation accuracy and loss

Explanation

- Loss functions and hyperparameter tuning using TensorFlow

- Model architecture and compilation

- Training and validation

Question 3: Training Models with Different Optimizers

```python
import tensorflow as tf

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import Dense
from tensorflow.keras.datasets import mnist

# Load MNIST dataset

(X_train, y_train), (X_test, y_test) = mnist.load_data()

# Normalize pixel values

X_train = X_train.astype('float32') / 255.0

X_test = X_test.astype('float32') / 255.0

# Define model architecture

model = Sequential([

    Dense(64, activation='relu', input_shape=(784,)),

    Dense(32, activation='relu'),

    Dense(10, activation='softmax')

])

# Compile model with different optimizers

model.compile(loss='sparse_categorical_crossentropy', optimizer='adam', metrics=['accuracy'])

# Train model

model.fit(X_train, y_train, epochs=10, batch_size=32, validation_data=(X_test, y_test))
```

Expected Output

- Model architecture and compilation

- Training and validation accuracy and loss

Explanation

- Training models with different optimizers using TensorFlow

- Model architecture and compilation

- Training and validation

Question 4: TensorBoard Logging

```python
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
from tensorflow.keras.callbacks import TensorBoard
# Define model architecture
model = Sequential([
    Dense(64, activation='relu', input_shape=(784,)),
    Dense(32, activation='relu'),
    Dense(10, activation='softmax')
])
# Compile model
model.compile(loss='sparse_categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
# Define TensorBoard callback
tensorboard_callback = TensorBoard(log_dir='./logs', histogram_freq=1)
# Train model with TensorBoard logging
model.fit(X_train, y_train, epochs=10, batch_size=32, validation_data=(X_test, y_test), callbacks=[tensorboard_callback])
```

Expected Output

- Model architecture and compilation
- TensorBoard logging setup
- Training and validation accuracy and loss

Explanation

- TensorBoard logging using TensorFlow
- Model architecture and compilation

- Training and validation with TensorBoard logging