NAME : SRAVAN KUMAR REDDY

SID : 2465382

ASSIGNMENT -10

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import LSTM, Dense, Dropout
from tensorflow.keras.callbacks import EarlyStopping, ModelCheckpoint
from sklearn.metrics import classification_report, accuracy_score
import matplotlib.pyplot as plt
import numpy as np

# Example: dummy dataset (replace with your actual data)
# X_train, X_test shape: (samples, timesteps, features)
# y_train, y_test: one-hot encoded labels
# Example: 1000 samples, 10 timesteps, 5 features, 3 classes
X_train = np.random.rand(800, 10, 5)
y_train = np.eye(3)[np.random.choice(3, 800)]
X_test = np.random.rand(200, 10, 5)
y_test = np.eye(3)[np.random.choice(3, 200)]

# Define LSTM model
model = Sequential()
model.add(LSTM(64, input_shape=(X_train.shape[1], X_train.shape[2]), return_sequences=False))
model.add(Dropout(0.2))
model.add(Dense(32, activation='relu'))
model.add(Dense(y_train.shape[1], activation='softmax'))

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

# Training parameters
epochs = 32
batch_size = 12
```

```python
# Callbacks
early_stop = EarlyStopping(monitor='val_loss', patience=5, restore_best_weights=True)
checkpoint = ModelCheckpoint('best_lstm_model.h5', monitor='val_accuracy', save_best_only=True, verbose=1)

# Train model
history = model.fit(
    X_train, y_train,
    epochs=epochs,
    batch_size=batch_size,
    validation_split=0.2,
    callbacks=[early_stop, checkpoint],
    verbose=1
)

# Evaluate
y_pred = model.predict(X_test)
y_pred_classes = y_pred.argmax(axis=1)
y_true_classes = y_test.argmax(axis=1)

print("Classification Report:")
print(classification_report(y_true_classes, y_pred_classes))
print("Test Accuracy:", accuracy_score(y_true_classes, y_pred_classes))

# Plot detailed accuracy
plt.figure(figsize=(10,6))
plt.plot(history.history['accuracy'], label='Training Accuracy')
plt.plot(history.history['val_accuracy'], label='Validation Accuracy')
plt.title('Detailed Accuracy over Epochs')
plt.xlabel('Epochs')
```

```python
plt.plot(history.history['accuracy'], label='Training Accuracy')
plt.plot(history.history['val_accuracy'], label='Validation Accuracy')
plt.title('Detailed Accuracy over Epochs')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()
plt.show()
```

```
ent to a layer. When using Sequential models, prefer using an `Input(shape)` object as the first layer in the model instead.
  super().__init__(**kwargs)
Epoch 1/32
54/54 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step - accuracy: 0.3330 - loss: 1.1050
Epoch 1: val_accuracy improved from None to 0.33125, saving model to best_lstm_model.h5
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is considered leg
We recommend using instead the native Keras format, e.g. `model.save('my_model.keras')` or `keras.saving.save_model(model, 'my_model.keras')`.
54/54 ━━━━━━━━━━━━━━━━━━━━ 6s 18ms/step - accuracy: 0.3297 - loss: 1.1040 - val_accuracy: 0.3313 - val_loss: 1.1004
Epoch 2/32
48/54 ━━━━━━━━━━━━━━━━━━━━ 0s 5ms/step - accuracy: 0.3708 - loss: 1.0916
Epoch 2: val_accuracy did not improve from 0.33125
54/54 ━━━━━━━━━━━━━━━━━━━━ 0s 8ms/step - accuracy: 0.3469 - loss: 1.0969 - val_accuracy: 0.3313 - val_loss: 1.0997
Epoch 3/32
50/54 ━━━━━━━━━━━━━━━━━━━━ 0s 5ms/step - accuracy: 0.3245 - loss: 1.1073
Epoch 3: val_accuracy did not improve from 0.33125
54/54 ━━━━━━━━━━━━━━━━━━━━ 1s 9ms/step - accuracy: 0.3547 - loss: 1.1046 - val_accuracy: 0.3313 - val_loss: 1.1055
Epoch 4/32
```

```
Epoch 5: val_accuracy improved from 0.33125 to 0.37500, saving model to best_lstm_model.h5
```

```
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is considered legacy.
We recommend using instead the native Keras format, e.g. `model.save('my_model.keras')` or `keras.saving.save_model(model, 'my_model.keras')`.
```

```
54/54 ━━━━━━━━━━━━━━━━━━━━ 1s 9ms/step - accuracy: 0.3578 - loss: 1.0974 - val_accuracy: 0.3750 - val_loss: 1.0960
Epoch 6/32
46/54 ━━━━━━━━━━━━━━━━━━━━ 0s 5ms/step - accuracy: 0.3939 - loss: 1.0890
Epoch 6: val_accuracy did not improve from 0.37500
54/54 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - accuracy: 0.3656 - loss: 1.0946 - val_accuracy: 0.3187 - val_loss: 1.1053
Epoch 7/32
47/54 ━━━━━━━━━━━━━━━━━━━━ 0s 5ms/step - accuracy: 0.3491 - loss: 1.0946
Epoch 7: val_accuracy did not improve from 0.37500
54/54 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - accuracy: 0.3469 - loss: 1.0947 - val_accuracy: 0.3313 - val_loss: 1.1103
Epoch 8/32
44/54 ━━━━━━━━━━━━━━━━━━━━ 0s 5ms/step - accuracy: 0.3423 - loss: 1.0936
Epoch 8: val_accuracy did not improve from 0.37500
54/54 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - accuracy: 0.3500 - loss: 1.0950 - val_accuracy: 0.3313 - val_loss: 1.0980
Epoch 9/32
```

```
54/54 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - accuracy: 0.3734 - loss: 1.0892 - val_accuracy: 0.3500 - val_loss: 1.1054
7/7 ━━━━━━━━━━━━━━━━━━━━ 0s 31ms/step
Classification Report:
              precision    recall  f1-score   support

           0       0.44      0.32      0.37        74
           1       0.00      0.00      0.00        59
           2       0.32      0.70      0.44        67

    accuracy                           0.35       200
   macro avg       0.25      0.34      0.27       200
weighted avg       0.27      0.35      0.29       200

Test Accuracy: 0.355
```

```
C:\Users\B sravan kumar reddy\anaconda3\Lib\site-packages\sklearn\metrics\_classification.py:1565: UndefinedMetricWarning: Precision is ill-defin
d being set to 0.0 in labels with no predicted samples. Use `zero_division` parameter to control this behavior.
```