cognitica-ucf-final

September 23, 2023

1 UCF 101 Action Recognition

INSTALLING LIBRARIES

[1]:
```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

[ ]:
```
!nvidia-smi
```

```
Sat Sep 23 07:49:53 2023
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 525.105.17   Driver Version: 525.105.17   CUDA Version: 12.0     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|                               |                      |               MIG M. |
|===============================+======================+======================|
|   0  Tesla T4            Off  | 00000000:00:04.0 Off |                    0 |
| N/A   34C    P8     9W /  70W |      0MiB / 15360MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                                  |
|  GPU   GI   CI        PID   Type   Process name                  GPU Memory |
|        ID   ID                                                   Usage      |
|=============================================================================|
|  No running processes found                                                 |
+-----------------------------------------------------------------------------+
```

[ ]:
```
print("Num GPUs Available: ", len(tf.config.experimental.
  ↪list_physical_devices('GPU')))

physical_devices = tf.config.experimental.list_physical_devices('GPU')
if len(physical_devices) > 0:
    tf.config.experimental.set_memory_growth(physical_devices[0], True)
```

```
Num GPUs Available:  1
```

[2]:
```
cd /content/drive/MyDrive/Cognitica
```

```
/content/drive/MyDrive/Cognitica
```

[ ]:
```
!pip install tensorflow
```

```
Requirement already satisfied: tensorflow in
c:\users\jaikr\.conda\envs\jupyter_\lib\site-packages (2.4.1)
Requirement already satisfied: flatbuffers~=1.12.0 in
c:\users\jaikr\.conda\envs\jupyter_\lib\site-packages (from tensorflow) (1.12)
Requirement already satisfied: opt-einsum~=3.3.0 in
c:\users\jaikr\.conda\envs\jupyter_\lib\site-packages (from tensorflow) (3.3.0)
Requirement already satisfied: tensorflow-estimator<2.5.0,>=2.4.0 in
c:\users\jaikr\.conda\envs\jupyter_\lib\site-packages (from tensorflow) (2.4.0)
Requirement already satisfied: grpcio~=1.32.0 in
c:\users\jaikr\.conda\envs\jupyter_\lib\site-packages (from tensorflow) (1.32.0)
Requirement already satisfied: astunparse~=1.6.3 in
c:\users\jaikr\.conda\envs\jupyter_\lib\site-packages (from tensorflow) (1.6.3)
Requirement already satisfied: h5py~=2.10.0 in
c:\users\jaikr\.conda\envs\jupyter_\lib\site-packages (from tensorflow) (2.10.0)
Requirement already satisfied: protobuf>=3.9.2 in
c:\users\jaikr\.conda\envs\jupyter_\lib\site-packages (from tensorflow) (3.20.3)
Requirement already satisfied: numpy~=1.19.2 in
c:\users\jaikr\.conda\envs\jupyter_\lib\site-packages (from tensorflow) (1.19.5)
Requirement already satisfied: wheel~=0.35 in
c:\users\jaikr\.conda\envs\jupyter_\lib\site-packages (from tensorflow) (0.38.4)
Requirement already satisfied: wrapt~=1.12.1 in
c:\users\jaikr\.conda\envs\jupyter_\lib\site-packages (from tensorflow) (1.12.1)
Requirement already satisfied: google-pasta~=0.2 in
c:\users\jaikr\.conda\envs\jupyter_\lib\site-packages (from tensorflow) (0.2.0)
Requirement already satisfied: tensorboard~=2.4 in
c:\users\jaikr\.conda\envs\jupyter_\lib\site-packages (from tensorflow) (2.11.2)
Requirement already satisfied: termcolor~=1.1.0 in
c:\users\jaikr\.conda\envs\jupyter_\lib\site-packages (from tensorflow) (1.1.0)
Requirement already satisfied: typing-extensions~=3.7.4 in
c:\users\jaikr\.conda\envs\jupyter_\lib\site-packages (from tensorflow)
(3.7.4.3)
Requirement already satisfied: absl-py~=0.10 in
c:\users\jaikr\.conda\envs\jupyter_\lib\site-packages (from tensorflow) (0.15.0)
Requirement already satisfied: keras-preprocessing~=1.1.2 in
c:\users\jaikr\.conda\envs\jupyter_\lib\site-packages (from tensorflow) (1.1.2)
Requirement already satisfied: six~=1.15.0 in
c:\users\jaikr\.conda\envs\jupyter_\lib\site-packages (from tensorflow) (1.15.0)
Requirement already satisfied: gast==0.3.3 in
c:\users\jaikr\.conda\envs\jupyter_\lib\site-packages (from tensorflow) (0.3.3)
Requirement already satisfied: google-auth-oauthlib<0.5,>=0.4.1 in
```

```
c:\users\jaikr\.conda\envs\jupyter_\lib\site-packages (from
tensorboard~=2.4->tensorflow) (0.4.6)
Requirement already satisfied: google-auth<3,>=1.6.3 in
c:\users\jaikr\.conda\envs\jupyter_\lib\site-packages (from
tensorboard~=2.4->tensorflow) (2.22.0)
Requirement already satisfied: werkzeug>=1.0.1 in
c:\users\jaikr\.conda\envs\jupyter_\lib\site-packages (from
tensorboard~=2.4->tensorflow) (2.2.3)
Requirement already satisfied: tensorboard-data-server<0.7.0,>=0.6.0 in
c:\users\jaikr\.conda\envs\jupyter_\lib\site-packages (from
tensorboard~=2.4->tensorflow) (0.6.1)
Requirement already satisfied: setuptools>=41.0.0 in
c:\users\jaikr\.conda\envs\jupyter_\lib\site-packages (from
tensorboard~=2.4->tensorflow) (65.6.3)
Requirement already satisfied: tensorboard-plugin-wit>=1.6.0 in
c:\users\jaikr\.conda\envs\jupyter_\lib\site-packages (from
tensorboard~=2.4->tensorflow) (1.8.1)
Requirement already satisfied: requests<3,>=2.21.0 in
c:\users\jaikr\.conda\envs\jupyter_\lib\site-packages (from
tensorboard~=2.4->tensorflow) (2.28.1)
Requirement already satisfied: markdown>=2.6.8 in
c:\users\jaikr\.conda\envs\jupyter_\lib\site-packages (from
tensorboard~=2.4->tensorflow) (3.4.4)
Requirement already satisfied: pyasn1-modules>=0.2.1 in
c:\users\jaikr\.conda\envs\jupyter_\lib\site-packages (from google-
auth<3,>=1.6.3->tensorboard~=2.4->tensorflow) (0.3.0)
Requirement already satisfied: rsa<5,>=3.1.4 in
c:\users\jaikr\.conda\envs\jupyter_\lib\site-packages (from google-
auth<3,>=1.6.3->tensorboard~=2.4->tensorflow) (4.9)
Requirement already satisfied: cachetools<6.0,>=2.0.0 in
c:\users\jaikr\.conda\envs\jupyter_\lib\site-packages (from google-
auth<3,>=1.6.3->tensorboard~=2.4->tensorflow) (5.3.1)
Requirement already satisfied: urllib3<2.0 in
c:\users\jaikr\.conda\envs\jupyter_\lib\site-packages (from google-
auth<3,>=1.6.3->tensorboard~=2.4->tensorflow) (1.26.14)
Requirement already satisfied: requests-oauthlib>=0.7.0 in
c:\users\jaikr\.conda\envs\jupyter_\lib\site-packages (from google-auth-
oauthlib<0.5,>=0.4.1->tensorboard~=2.4->tensorflow) (1.3.1)
Requirement already satisfied: importlib-metadata>=4.4 in
c:\users\jaikr\.conda\envs\jupyter_\lib\site-packages (from
markdown>=2.6.8->tensorboard~=2.4->tensorflow) (4.11.3)
Requirement already satisfied: certifi>=2017.4.17 in
c:\users\jaikr\.conda\envs\jupyter_\lib\site-packages (from
requests<3,>=2.21.0->tensorboard~=2.4->tensorflow) (2022.12.7)
Requirement already satisfied: idna<4,>=2.5 in
c:\users\jaikr\.conda\envs\jupyter_\lib\site-packages (from
requests<3,>=2.21.0->tensorboard~=2.4->tensorflow) (3.4)
Requirement already satisfied: charset-normalizer<3,>=2 in
```

```
c:\users\jaikr\.conda\envs\jupyter_\lib\site-packages (from
requests<3,>=2.21.0->tensorboard~=2.4->tensorflow) (2.0.4)
Requirement already satisfied: MarkupSafe>=2.1.1 in
c:\users\jaikr\.conda\envs\jupyter_\lib\site-packages (from
werkzeug>=1.0.1->tensorboard~=2.4->tensorflow) (2.1.1)
Requirement already satisfied: zipp>=0.5 in
c:\users\jaikr\.conda\envs\jupyter_\lib\site-packages (from importlib-
metadata>=4.4->markdown>=2.6.8->tensorboard~=2.4->tensorflow) (3.11.0)
Requirement already satisfied: pyasn1<0.6.0,>=0.4.6 in
c:\users\jaikr\.conda\envs\jupyter_\lib\site-packages (from
pyasn1-modules>=0.2.1->google-auth<3,>=1.6.3->tensorboard~=2.4->tensorflow)
(0.5.0)
Requirement already satisfied: oauthlib>=3.0.0 in
c:\users\jaikr\.conda\envs\jupyter_\lib\site-packages (from requests-
oauthlib>=0.7.0->google-auth-oauthlib<0.5,>=0.4.1->tensorboard~=2.4->tensorflow)
(3.2.2)
```

[3]:
```
import os
import cv2
import math
import random
import numpy as np
import datetime as dt
import tensorflow as tf
from collections import deque
import matplotlib.pyplot as plt

# from moviepy.editor import *
%matplotlib inline

from sklearn.model_selection import train_test_split

from tensorflow.keras.layers import *
from tensorflow.keras.models import Sequential
from tensorflow.keras.utils import to_categorical
from tensorflow.keras.callbacks import EarlyStopping
from tensorflow.keras.utils import plot_model
```

[ ]:
```
!pip install scikit-learn
```

```
Collecting scikit-learn
  Downloading scikit_learn-1.0.2-cp37-cp37m-win_amd64.whl (7.1 MB)
     ---------------------------------------- 7.1/7.1 MB 6.7 MB/s eta 0:00:00
Collecting threadpoolctl>=2.0.0
  Downloading threadpoolctl-3.1.0-py3-none-any.whl (14 kB)
Requirement already satisfied: scipy>=1.1.0 in
c:\users\jaikr\.conda\envs\jupyter_\lib\site-packages (from scikit-learn)
(1.7.3)
```

```
Requirement already satisfied: numpy>=1.14.6 in
c:\users\jaikr\.conda\envs\jupyter_\lib\site-packages (from scikit-learn)
(1.19.5)
Collecting joblib>=0.11
  Using cached joblib-1.3.2-py3-none-any.whl (302 kB)
Installing collected packages: threadpoolctl, joblib, scikit-learn
Successfully installed joblib-1.3.2 scikit-learn-1.0.2 threadpoolctl-3.1.0
Note: you may need to restart the kernel to use updated packages.
```

2 VISUALISATION

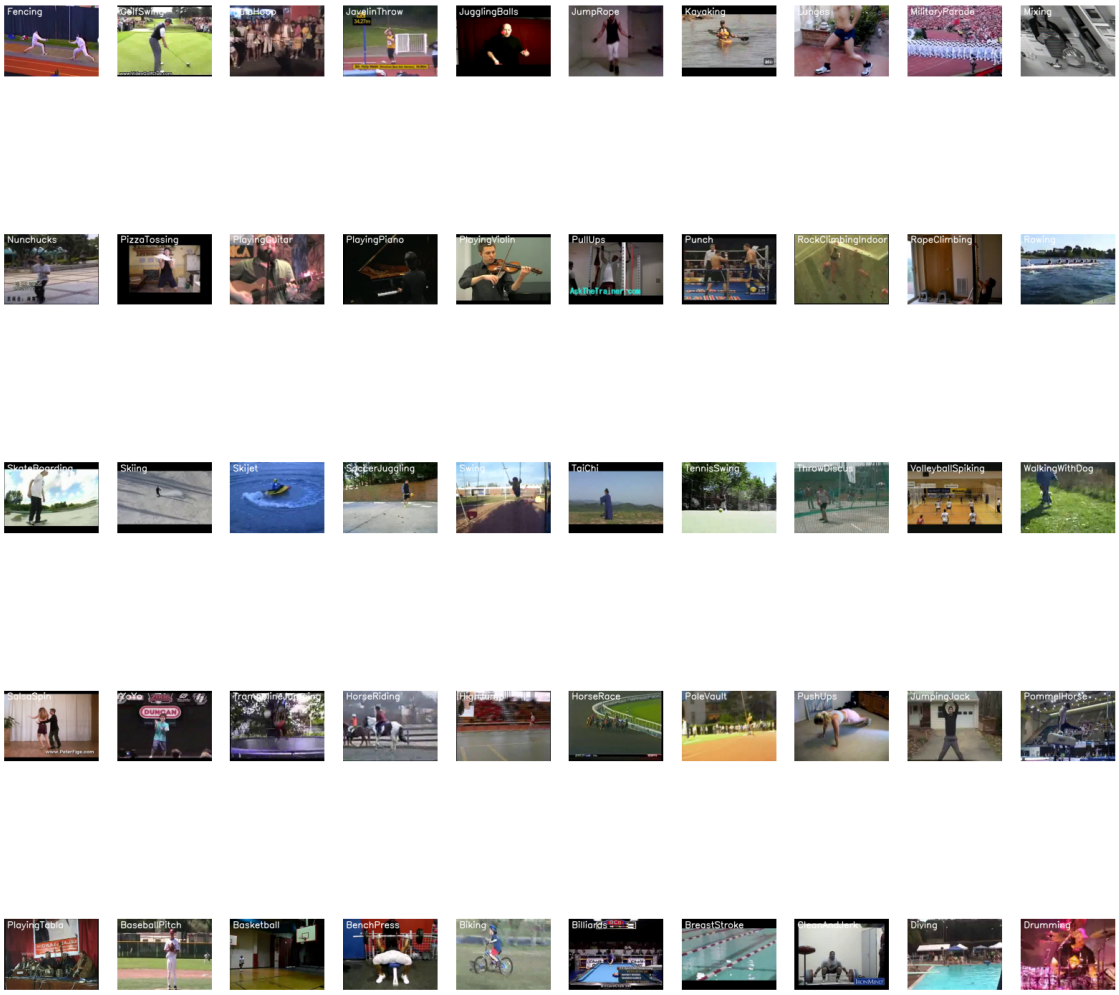
[ ]:
```python
import random

plt.figure(figsize=(20, 20))
all_classes_names = os.listdir('UCF50')

for counter, selected_class_Name in enumerate(all_classes_names, 1):
    video_files_names_list = os.listdir(f'UCF50/{selected_class_Name}')

    # Check if there are video files in the folder
    if video_files_names_list:
        selected_video_file_name = random.choice(video_files_names_list)
        video_reader = cv2.VideoCapture(f'UCF50/{selected_class_Name}/
 ↪{selected_video_file_name}')
        _, bgr_frame = video_reader.read()
        video_reader.release()
        rgb_frame = cv2.cvtColor(bgr_frame, cv2.COLOR_BGR2RGB)
        cv2.putText(rgb_frame, selected_class_Name, (10, 30), cv2.
 ↪FONT_HERSHEY_SIMPLEX, 1, (255, 255, 255), 2)
        plt.subplot(5, 10, counter)
        plt.imshow(rgb_frame)
        plt.axis('off')
    else:
        print(f"No video files found in folder {selected_class_Name}")

plt.show()
```



[4]:
```
IMAGE_HEIGHT,IMAGE_WIDTH = 64,64
SEQUENCE_LENGTH = 20
DATASET_DIR="UCF50"
CLASSES_LIST=["PlayingTabla","PommelHorse","JumpingJack","PushUps","PoleVault","HorseRace","Hi
                ␣
 ↪"Basketball","TrampolineJumping","YoYo","SalsaSpin","WalkingWithDog","VolleyballSpiking","T
                ␣
 ↪"SoccerJuggling","Skijet","Skiing","SkateBoarding","Rowing","RopeClimbing","RockClimbingInd
                ␣
 ↪"PizzaTossing","Nunchucks","Mixing","MilitaryParade","Lunges","Kayaking","JumpRope","Juggli
                 "CleanAndJerk","Billiards","Biking","BenchPress","BaseballPitch"]
```

3 VIDEO PREPROCESSING

[5]:
```
def frames_extraction(video_path):
  frames_list=[]
  video_reader = cv2.VideoCapture(video_path)
  video_frames_count = int(video_reader.get(cv2.CAP_PROP_FRAME_COUNT))
  skip_frames_window = max(int(video_frames_count/SEQUENCE_LENGTH),1)
  for frame_counter in range(SEQUENCE_LENGTH):
    video_reader.set(cv2.CAP_PROP_POS_FRAMES,frame_counter*skip_frames_window)
    success,frame = video_reader.read()
    if not success:
      break

    resized_frame = cv2.resize(frame,(IMAGE_HEIGHT,IMAGE_WIDTH))
    normalized_frame = resized_frame / 255
    frames_list.append(normalized_frame)
  video_reader.release()

  return frames_list
```

[6]:
```
def create_dataset():
  features = []
  labels = []
  video_files_paths = []
  for class_index,class_name in enumerate(CLASSES_LIST[:40]):
    print(f'Extracting Data of CLass: {class_name}')

    files_list = os.listdir(os.path.join(DATASET_DIR,class_name))
    for file_name in files_list:
      video_file_path = os.path.join(DATASET_DIR,class_name,file_name)

      frames = frames_extraction(video_file_path)
      if len(frames) == SEQUENCE_LENGTH:
        features.append(frames)
        labels.append(class_index)
        video_files_paths.append(video_file_path)
  features = np.asarray(features)
  labels = np.array(labels)

  return features,labels,video_files_paths
```

4 FEATURE EXTRACTION

[ ]:
```
features,labels,video_files_paths = create_dataset()
```

```
Extracting Data of CLass: PlayingTabla
Extracting Data of CLass: PommelHorse
```

```
Extracting Data of CLass: JumpingJack
Extracting Data of CLass: PushUps
Extracting Data of CLass: PoleVault
Extracting Data of CLass: HorseRace
Extracting Data of CLass: HighJump
Extracting Data of CLass: Drumming
Extracting Data of CLass: HorseRiding
Extracting Data of CLass: Diving
Extracting Data of CLass: BreastStroke
Extracting Data of CLass: Basketball
Extracting Data of CLass: TrampolineJumping
Extracting Data of CLass: YoYo
Extracting Data of CLass: SalsaSpin
Extracting Data of CLass: WalkingWithDog
Extracting Data of CLass: VolleyballSpiking
Extracting Data of CLass: ThrowDiscus
Extracting Data of CLass: TennisSwing
Extracting Data of CLass: TaiChi
Extracting Data of CLass: Swing
Extracting Data of CLass: SoccerJuggling
Extracting Data of CLass: Skijet
Extracting Data of CLass: Skiing
Extracting Data of CLass: SkateBoarding
Extracting Data of CLass: Rowing
Extracting Data of CLass: RopeClimbing
Extracting Data of CLass: RockClimbingIndoor
Extracting Data of CLass: Punch
Extracting Data of CLass: PullUps
Extracting Data of CLass: PlayingViolin
Extracting Data of CLass: PlayingPiano
Extracting Data of CLass: PlayingGuitar
Extracting Data of CLass: PizzaTossing
Extracting Data of CLass: Nunchucks
Extracting Data of CLass: Mixing
Extracting Data of CLass: MilitaryParade
Extracting Data of CLass: Lunges
Extracting Data of CLass: Kayaking
Extracting Data of CLass: JumpRope
```

[ ]:
```
one_hot_encoded_labels = to_categorical(labels)
```

[7]:
```
seed_constant = 27
np.random.seed(seed_constant)
random.seed(seed_constant)
tf.random.set_seed(seed_constant)
```

5 SPLITTING DATA FOR TRAIN TEST

[10]:
```
features_train,features_test,labels_train,lables_test =␣
↪train_test_split(features,one_hot_encoded_labels,
                                                                    ␣
↪test_size = 0.25,shuffle = True,
                                                                    ␣
↪random_state = seed_constant)
```

6 CNN-LSTM MODEL LAYERS

[11]:
```
def create_LRCN_model():


    model = Sequential()

    #Model Architecture.
    ␣
↪#---------------------------------------------------------------------------------------------

    model.add(TimeDistributed(Conv2D(16, (3, 3), padding='same',activation =␣
↪'relu'),
                              input_shape = (SEQUENCE_LENGTH, IMAGE_HEIGHT,␣
↪IMAGE_WIDTH, 3)))

    model.add(TimeDistributed(MaxPooling2D((4, 4))))
    model.add(TimeDistributed(Dropout(0.25)))

    model.add(TimeDistributed(Conv2D(32, (3, 3), padding='same',activation =␣
↪'relu')))
    model.add(TimeDistributed(MaxPooling2D((4, 4))))
    model.add(TimeDistributed(Dropout(0.25)))

    model.add(TimeDistributed(Conv2D(64, (3, 3), padding='same',activation =␣
↪'relu')))
    model.add(TimeDistributed(MaxPooling2D((2, 2))))
    model.add(TimeDistributed(Dropout(0.25)))

    model.add(TimeDistributed(Conv2D(64, (3, 3), padding='same',activation =␣
↪'relu')))
    model.add(TimeDistributed(MaxPooling2D((2, 2))))
    #model.add(TimeDistributed(Dropout(0.25)))

    model.add(TimeDistributed(Flatten()))

    #K-LAYERED LSTM K=1
```

```
    model.add(LSTM(32))

    model.add(Dense(len(CLASSES_LIST[:10]), activation = 'softmax'))

    ␣
↪#--------------------------------------------------------------------------------

    model.summary()


    return model
```

1. Number of Layers: The choice of the number of convolutional and pooling layers in the TimeDistributed layers is based on a common architecture pattern for extracting spatial features from video frames, increasing the number of filters while reducing spatial dimensions to capture hierarchical features.
2. Activation Function: ReLU (Rectified Linear Unit) activation functions was chosen for convolutional layers because to introduce non-linearity to the model, allowing it to learn complex patterns in the data effectively.
3. Dropout: Dropout layers with a dropout rate of 0.25 was added after each MaxPooling2D layer to prevent overfitting by randomly deactivating a fraction of neurons during training.
4. LSTM Layer: A single LSTM layer with 32 units was chosen to capture temporal dependencies in the video sequences.
5. Output Layer Activation: The softmax activation function was used in the output layer to convert the model's predictions into class probabilities, suitable for multi-class classification tasks.

[12]:
```
LRCN_model = create_LRCN_model()


print("Model Created Successfully!")
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 time_distributed (TimeDist  (None, 20, 64, 64, 16)    448
 ributed)

 time_distributed_1 (TimeDi  (None, 20, 16, 16, 16)    0
 stributed)

 time_distributed_2 (TimeDi  (None, 20, 16, 16, 16)    0
 stributed)

 time_distributed_3 (TimeDi  (None, 20, 16, 16, 32)    4640
```

```
 stributed)

 time_distributed_4 (TimeDi  (None, 20, 4, 4, 32)      0
 stributed)

 time_distributed_5 (TimeDi  (None, 20, 4, 4, 32)      0
 stributed)

 time_distributed_6 (TimeDi  (None, 20, 4, 4, 64)      18496
 stributed)

 time_distributed_7 (TimeDi  (None, 20, 2, 2, 64)      0
 stributed)

 time_distributed_8 (TimeDi  (None, 20, 2, 2, 64)      0
 stributed)

 time_distributed_9 (TimeDi  (None, 20, 2, 2, 64)      36928
 stributed)

 time_distributed_10 (TimeD  (None, 20, 1, 1, 64)      0
 istributed)

 time_distributed_11 (TimeD  (None, 20, 64)            0
 istributed)

 lstm (LSTM)                 (None, 32)                12416

 dense (Dense)               (None, 10)                330

=================================================================
Total params: 73258 (286.16 KB)
Trainable params: 73258 (286.16 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
Model Created Successfully!
```

7 MODEL PLOT VISUALISATION

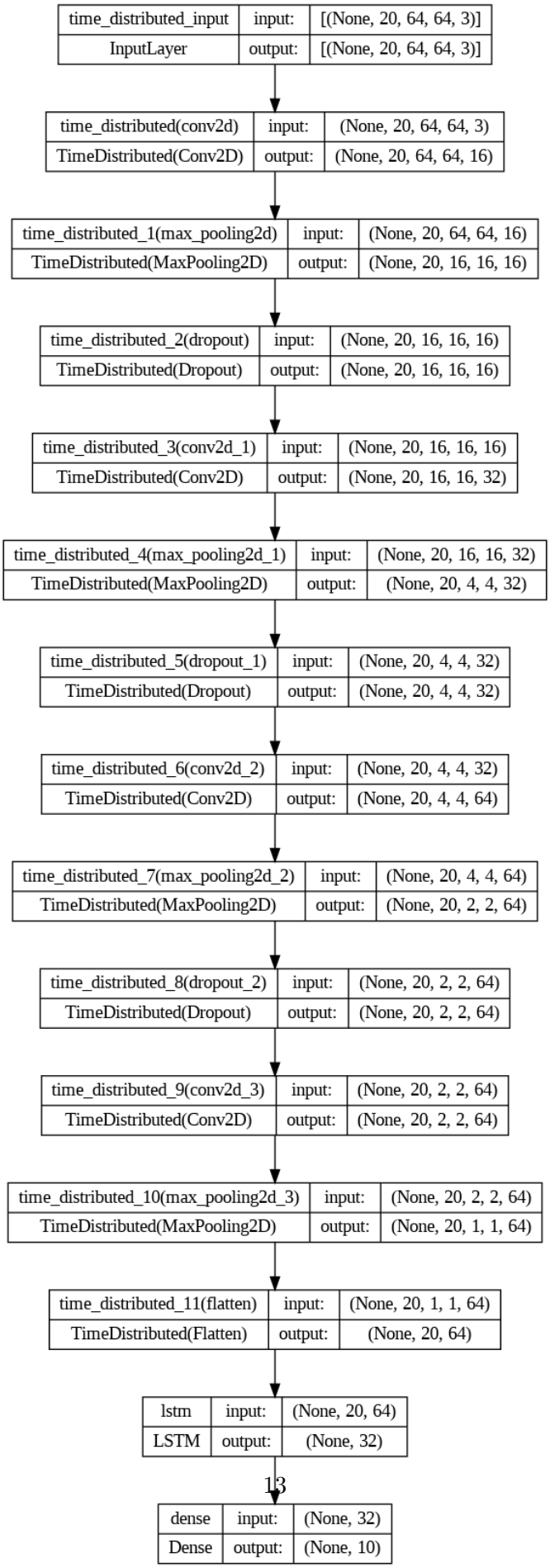
[ ]:
```
!pip install pydot
!pip install graphviz
import pydot
```

```
Requirement already satisfied: pydot in
c:\users\jaikr\.conda\envs\jupyter_\lib\site-packages (1.4.2)
Requirement already satisfied: pyparsing>=2.1.4 in
c:\users\jaikr\.conda\envs\jupyter_\lib\site-packages (from pydot) (3.1.1)
```

```
Requirement already satisfied: graphviz in
c:\users\jaikr\.conda\envs\jupyter_\lib\site-packages (0.20.1)
```

[ ]:
```
plot_model(LRCN_model,to_file = 'LRCN_model_Structure_plot.
↪png',show_shapes=True,show_layer_names = True)
```

[ ]:


time_distributed_input | input: | [(None, 20, 64, 64, 3)]
InputLayer | output: | [(None, 20, 64, 64, 3)]
time_distributed(conv2d) | input: | (None, 20, 64, 64, 3)
TimeDistributed(Conv2D) | output: | (None, 20, 64, 64, 16)
time_distributed_1(max_pooling2d) | input: | (None, 20, 64, 64, 16)
TimeDistributed(MaxPooling2D) | output: | (None, 20, 16, 16, 16)
time_distributed_2(dropout) | input: | (None, 20, 16, 16, 16)
TimeDistributed(Dropout) | output: | (None, 20, 16, 16, 16)
time_distributed_3(conv2d_1) | input: | (None, 20, 16, 16, 16)
TimeDistributed(Conv2D) | output: | (None, 20, 16, 16, 32)
time_distributed_4(max_pooling2d_1) | input: | (None, 20, 16, 16, 32)
TimeDistributed(MaxPooling2D) | output: | (None, 20, 4, 4, 32)
time_distributed_5(dropout_1) | input: | (None, 20, 4, 4, 32)
TimeDistributed(Dropout) | output: | (None, 20, 4, 4, 32)
time_distributed_6(conv2d_2) | input: | (None, 20, 4, 4, 32)
TimeDistributed(Conv2D) | output: | (None, 20, 4, 4, 64)
time_distributed_7(max_pooling2d_2) | input: | (None, 20, 4, 4, 64)
TimeDistributed(MaxPooling2D) | output: | (None, 20, 2, 2, 64)
time_distributed_8(dropout_2) | input: | (None, 20, 2, 2, 64)
TimeDistributed(Dropout) | output: | (None, 20, 2, 2, 64)
time_distributed_9(conv2d_3) | input: | (None, 20, 2, 2, 64)
TimeDistributed(Conv2D) | output: | (None, 20, 2, 2, 64)
time_distributed_10(max_pooling2d_3) | input: | (None, 20, 2, 2, 64)
TimeDistributed(MaxPooling2D) | output: | (None, 20, 1, 1, 64)
time_distributed_11(flatten) | input: | (None, 20, 1, 1, 64)
TimeDistributed(Flatten) | output: | (None, 20, 64)
lstm | input: | (None, 20, 64)
LSTM | output: | (None, 32)
dense | input: | (None, 32)
Dense | output: | (None, 10)

[16]:
```
def plot_metric(model_training_history,metric_name1,metric_name2,plot_name):
  metric_value1 = model_training_history.history[metric_name1]
  metric_value2 = model_training_history.history[metric_name2]

  epochs = range(len(metric_value1))

  plt.plot(epochs,metric_value1,'blue',label=metric_name1)
  plt.plot(epochs,metric_value2,'red',label=metric_name2)
  plt.title(str(plot_name))

  plt.legend()
```

8 MODEL TRAINING

1. Learning Rate: The choice of the learning rate (lr),a value of 0.001 (default value for Adam optimizer) model to converge effectively without causing divergence.
2. Optimizer: The Adam optimizer was chosen because it combines the benefits of both AdaGrad and RMSProp, providing effective optimization for training.
3. Epochs: The number of epochs (100) was selected based on early stopping to prevent overfitting while allowing the model to train until convergence.
4. Batch Size: A batch size of 4 was chosen to have a balance between computation efficiency and model stability during training.
5. Loss Function: Categorical Crossentropy was chosen as the loss function because it is suitable for multi-class classification tasks and helps the model to minimize the difference between predicted and actual class probabilities.

[13]:
```
import time

# before training
start_time = time.time()

early_stopping_callback = EarlyStopping(monitor='accuracy', patience=10,␣
 ↪mode='max', restore_best_weights=True)

LRCN_model.compile(loss='categorical_crossentropy', optimizer='Adam',␣
 ↪metrics=["accuracy"])

# Start training
LRCN_model_training_history = LRCN_model.fit(x=features_train, y=labels_train,␣
 ↪epochs=100, batch_size=4,
                                              shuffle=True, validation_split=0.
 ↪2, callbacks=[early_stopping_callback])
```

```python
# end time after training
end_time = time.time()

# total training time
total_training_time = end_time - start_time
print(f"Total training time: {total_training_time:.2f} seconds")
```

```
Epoch 1/100
210/210 [==============================] - 64s 285ms/step - loss: 2.2857 -
accuracy: 0.1637 - val_loss: 2.2180 - val_accuracy: 0.1429
Epoch 2/100
210/210 [==============================] - 46s 218ms/step - loss: 1.9896 -
accuracy: 0.2867 - val_loss: 1.7623 - val_accuracy: 0.2905
Epoch 3/100
210/210 [==============================] - 48s 227ms/step - loss: 1.6800 -
accuracy: 0.4146 - val_loss: 1.6667 - val_accuracy: 0.4429
Epoch 4/100
210/210 [==============================] - 49s 233ms/step - loss: 1.4914 -
accuracy: 0.4934 - val_loss: 1.3236 - val_accuracy: 0.5667
Epoch 5/100
210/210 [==============================] - 46s 220ms/step - loss: 1.4182 -
accuracy: 0.5030 - val_loss: 1.2893 - val_accuracy: 0.5810
Epoch 6/100
210/210 [==============================] - 46s 219ms/step - loss: 1.2485 -
accuracy: 0.5687 - val_loss: 1.2132 - val_accuracy: 0.6000
Epoch 7/100
210/210 [==============================] - 45s 216ms/step - loss: 1.1579 -
accuracy: 0.6081 - val_loss: 1.0671 - val_accuracy: 0.6476
Epoch 8/100
210/210 [==============================] - 45s 216ms/step - loss: 1.0425 -
accuracy: 0.6344 - val_loss: 1.0284 - val_accuracy: 0.6619
Epoch 9/100
210/210 [==============================] - 45s 216ms/step - loss: 0.8958 -
accuracy: 0.7001 - val_loss: 0.8957 - val_accuracy: 0.7143
Epoch 10/100
210/210 [==============================] - 46s 217ms/step - loss: 0.7761 -
accuracy: 0.7503 - val_loss: 0.8505 - val_accuracy: 0.7333
Epoch 11/100
210/210 [==============================] - 45s 217ms/step - loss: 0.7110 -
accuracy: 0.7634 - val_loss: 0.7115 - val_accuracy: 0.7810
Epoch 12/100
210/210 [==============================] - 48s 229ms/step - loss: 0.6340 -
accuracy: 0.7945 - val_loss: 1.0735 - val_accuracy: 0.6429
Epoch 13/100
210/210 [==============================] - 46s 218ms/step - loss: 0.5469 -
accuracy: 0.8172 - val_loss: 0.7026 - val_accuracy: 0.8048
```

```
Epoch 14/100
210/210 [==============================] - 46s 218ms/step - loss: 0.4584 -
accuracy: 0.8578 - val_loss: 0.5640 - val_accuracy: 0.8476
Epoch 15/100
210/210 [==============================] - 45s 217ms/step - loss: 0.4626 -
accuracy: 0.8519 - val_loss: 0.5883 - val_accuracy: 0.8238
Epoch 16/100
210/210 [==============================] - 48s 229ms/step - loss: 0.3564 -
accuracy: 0.9092 - val_loss: 0.5712 - val_accuracy: 0.8333
Epoch 17/100
210/210 [==============================] - 48s 228ms/step - loss: 0.2942 -
accuracy: 0.9188 - val_loss: 0.5062 - val_accuracy: 0.8571
Epoch 18/100
210/210 [==============================] - 46s 220ms/step - loss: 0.2647 -
accuracy: 0.9200 - val_loss: 0.5300 - val_accuracy: 0.8238
Epoch 19/100
210/210 [==============================] - 48s 229ms/step - loss: 0.3197 -
accuracy: 0.9008 - val_loss: 0.4969 - val_accuracy: 0.8619
Epoch 20/100
210/210 [==============================] - 48s 228ms/step - loss: 0.2042 -
accuracy: 0.9462 - val_loss: 0.6027 - val_accuracy: 0.8238
Epoch 21/100
210/210 [==============================] - 46s 218ms/step - loss: 0.2420 -
accuracy: 0.9235 - val_loss: 0.5591 - val_accuracy: 0.8571
Epoch 22/100
210/210 [==============================] - 48s 231ms/step - loss: 0.2564 -
accuracy: 0.9211 - val_loss: 0.4882 - val_accuracy: 0.8619
Epoch 23/100
210/210 [==============================] - 47s 224ms/step - loss: 0.2946 -
accuracy: 0.9128 - val_loss: 0.3511 - val_accuracy: 0.9000
Epoch 24/100
210/210 [==============================] - 46s 221ms/step - loss: 0.1230 -
accuracy: 0.9701 - val_loss: 0.4159 - val_accuracy: 0.8857
Epoch 25/100
210/210 [==============================] - 46s 219ms/step - loss: 0.1897 -
accuracy: 0.9403 - val_loss: 0.3959 - val_accuracy: 0.8905
Epoch 26/100
210/210 [==============================] - 48s 229ms/step - loss: 0.2191 -
accuracy: 0.9319 - val_loss: 0.3297 - val_accuracy: 0.9000
Epoch 27/100
210/210 [==============================] - 46s 218ms/step - loss: 0.0691 -
accuracy: 0.9869 - val_loss: 0.3615 - val_accuracy: 0.9286
Epoch 28/100
210/210 [==============================] - 48s 230ms/step - loss: 0.0541 -
accuracy: 0.9857 - val_loss: 0.3190 - val_accuracy: 0.9238
Epoch 29/100
210/210 [==============================] - 46s 219ms/step - loss: 0.2235 -
accuracy: 0.9367 - val_loss: 0.5476 - val_accuracy: 0.8429
```

```
Epoch 30/100
210/210 [==============================] - 48s 228ms/step - loss: 0.1736 -
accuracy: 0.9486 - val_loss: 0.3049 - val_accuracy: 0.9190
Epoch 31/100
210/210 [==============================] - 46s 220ms/step - loss: 0.0753 -
accuracy: 0.9809 - val_loss: 0.3990 - val_accuracy: 0.9048
Epoch 32/100
210/210 [==============================] - 46s 221ms/step - loss: 0.0457 -
accuracy: 0.9881 - val_loss: 0.3929 - val_accuracy: 0.9238
Epoch 33/100
210/210 [==============================] - 48s 229ms/step - loss: 0.0403 -
accuracy: 0.9928 - val_loss: 0.3990 - val_accuracy: 0.9143
Epoch 34/100
210/210 [==============================] - 48s 230ms/step - loss: 0.0362 -
accuracy: 0.9904 - val_loss: 0.3804 - val_accuracy: 0.9190
Epoch 35/100
210/210 [==============================] - 48s 230ms/step - loss: 0.0724 -
accuracy: 0.9809 - val_loss: 0.5057 - val_accuracy: 0.8095
Epoch 36/100
210/210 [==============================] - 47s 225ms/step - loss: 0.1255 -
accuracy: 0.9594 - val_loss: 0.3483 - val_accuracy: 0.9238
Epoch 37/100
210/210 [==============================] - 46s 219ms/step - loss: 0.0502 -
accuracy: 0.9892 - val_loss: 0.3175 - val_accuracy: 0.9476
Epoch 38/100
210/210 [==============================] - 48s 228ms/step - loss: 0.3029 -
accuracy: 0.9152 - val_loss: 0.3650 - val_accuracy: 0.9000
Epoch 39/100
210/210 [==============================] - 46s 218ms/step - loss: 0.0698 -
accuracy: 0.9785 - val_loss: 0.4175 - val_accuracy: 0.9238
Epoch 40/100
210/210 [==============================] - 45s 217ms/step - loss: 0.0319 -
accuracy: 0.9928 - val_loss: 0.3273 - val_accuracy: 0.9286
Epoch 41/100
210/210 [==============================] - 48s 229ms/step - loss: 0.0312 -
accuracy: 0.9928 - val_loss: 0.4579 - val_accuracy: 0.9143
Epoch 42/100
210/210 [==============================] - 44s 209ms/step - loss: 0.0327 -
accuracy: 0.9904 - val_loss: 0.4689 - val_accuracy: 0.8810
Epoch 43/100
210/210 [==============================] - 49s 232ms/step - loss: 0.0907 -
accuracy: 0.9761 - val_loss: 0.5111 - val_accuracy: 0.8762
Epoch 44/100
210/210 [==============================] - 48s 228ms/step - loss: 0.1932 -
accuracy: 0.9415 - val_loss: 0.5547 - val_accuracy: 0.8810
Epoch 45/100
210/210 [==============================] - 47s 222ms/step - loss: 0.0551 -
accuracy: 0.9833 - val_loss: 0.3284 - val_accuracy: 0.9286
```

```
Epoch 46/100
210/210 [==============================] - 46s 220ms/step - loss: 0.0130 -
accuracy: 0.9976 - val_loss: 0.3669 - val_accuracy: 0.9286
Epoch 47/100
210/210 [==============================] - 46s 218ms/step - loss: 0.0111 -
accuracy: 0.9988 - val_loss: 0.3517 - val_accuracy: 0.9333
Epoch 48/100
210/210 [==============================] - 48s 231ms/step - loss: 0.1648 -
accuracy: 0.9498 - val_loss: 0.5305 - val_accuracy: 0.8571
Epoch 49/100
210/210 [==============================] - 46s 218ms/step - loss: 0.1302 -
accuracy: 0.9558 - val_loss: 0.6336 - val_accuracy: 0.8429
Epoch 50/100
210/210 [==============================] - 48s 230ms/step - loss: 0.0754 -
accuracy: 0.9809 - val_loss: 0.6056 - val_accuracy: 0.8619
Epoch 51/100
210/210 [==============================] - 48s 226ms/step - loss: 0.0548 -
accuracy: 0.9857 - val_loss: 0.3637 - val_accuracy: 0.9286
Epoch 52/100
210/210 [==============================] - 46s 220ms/step - loss: 0.0251 -
accuracy: 0.9940 - val_loss: 0.3174 - val_accuracy: 0.9286
Epoch 53/100
210/210 [==============================] - 49s 231ms/step - loss: 0.0257 -
accuracy: 0.9916 - val_loss: 0.3625 - val_accuracy: 0.9286
Epoch 54/100
210/210 [==============================] - 48s 231ms/step - loss: 0.0620 -
accuracy: 0.9821 - val_loss: 0.4055 - val_accuracy: 0.9286
Epoch 55/100
210/210 [==============================] - 48s 230ms/step - loss: 0.1171 -
accuracy: 0.9606 - val_loss: 0.3175 - val_accuracy: 0.9333
Epoch 56/100
210/210 [==============================] - 46s 220ms/step - loss: 0.0277 -
accuracy: 0.9952 - val_loss: 0.3580 - val_accuracy: 0.9476
Epoch 57/100
210/210 [==============================] - 47s 224ms/step - loss: 0.0197 -
accuracy: 0.9928 - val_loss: 0.3732 - val_accuracy: 0.9381
Epoch 58/100
210/210 [==============================] - 49s 232ms/step - loss: 0.0094 -
accuracy: 0.9976 - val_loss: 0.3506 - val_accuracy: 0.9333
Epoch 59/100
210/210 [==============================] - 46s 219ms/step - loss: 0.0304 -
accuracy: 0.9881 - val_loss: 0.4057 - val_accuracy: 0.9190
Epoch 60/100
210/210 [==============================] - 46s 218ms/step - loss: 0.1785 -
accuracy: 0.9522 - val_loss: 0.6308 - val_accuracy: 0.8238
Epoch 61/100
210/210 [==============================] - 46s 218ms/step - loss: 0.0973 -
accuracy: 0.9654 - val_loss: 0.3607 - val_accuracy: 0.9190
```

```
Epoch 62/100
210/210 [==============================] - 46s 220ms/step - loss: 0.0229 -
accuracy: 0.9928 - val_loss: 0.3812 - val_accuracy: 0.9286
Epoch 63/100
210/210 [==============================] - 48s 230ms/step - loss: 0.0121 -
accuracy: 0.9964 - val_loss: 0.4065 - val_accuracy: 0.9190
Epoch 64/100
210/210 [==============================] - 47s 223ms/step - loss: 0.0067 -
accuracy: 0.9988 - val_loss: 0.4554 - val_accuracy: 0.9238
Epoch 65/100
210/210 [==============================] - 47s 225ms/step - loss: 0.0081 -
accuracy: 0.9976 - val_loss: 0.4030 - val_accuracy: 0.9238
Epoch 66/100
210/210 [==============================] - 48s 230ms/step - loss: 0.0049 -
accuracy: 1.0000 - val_loss: 0.4229 - val_accuracy: 0.9333
Epoch 67/100
210/210 [==============================] - 46s 219ms/step - loss: 0.0026 -
accuracy: 1.0000 - val_loss: 0.4242 - val_accuracy: 0.9333
Epoch 68/100
210/210 [==============================] - 49s 231ms/step - loss: 0.0296 -
accuracy: 0.9928 - val_loss: 0.4893 - val_accuracy: 0.9095
Epoch 69/100
210/210 [==============================] - 48s 228ms/step - loss: 0.2302 -
accuracy: 0.9295 - val_loss: 0.4162 - val_accuracy: 0.9048
Epoch 70/100
210/210 [==============================] - 46s 221ms/step - loss: 0.0547 -
accuracy: 0.9833 - val_loss: 0.4762 - val_accuracy: 0.8952
Epoch 71/100
210/210 [==============================] - 46s 218ms/step - loss: 0.0186 -
accuracy: 0.9952 - val_loss: 0.4165 - val_accuracy: 0.9000
Epoch 72/100
210/210 [==============================] - 46s 217ms/step - loss: 0.0067 -
accuracy: 0.9988 - val_loss: 0.4274 - val_accuracy: 0.9238
Epoch 73/100
210/210 [==============================] - 49s 232ms/step - loss: 0.0048 -
accuracy: 0.9988 - val_loss: 0.4575 - val_accuracy: 0.9095
Epoch 74/100
210/210 [==============================] - 48s 229ms/step - loss: 0.0057 -
accuracy: 0.9988 - val_loss: 0.4688 - val_accuracy: 0.9238
Epoch 75/100
210/210 [==============================] - 46s 218ms/step - loss: 0.2237 -
accuracy: 0.9319 - val_loss: 0.4128 - val_accuracy: 0.9190
Epoch 76/100
210/210 [==============================] - 48s 229ms/step - loss: 0.0514 -
accuracy: 0.9821 - val_loss: 0.2922 - val_accuracy: 0.9429
Epoch 77/100
210/210 [==============================] - 47s 223ms/step - loss: 0.0106 -
accuracy: 0.9988 - val_loss: 0.3224 - val_accuracy: 0.9333
```

```
Epoch 78/100
210/210 [==============================] - 46s 220ms/step - loss: 0.0057 -
accuracy: 0.9988 - val_loss: 0.3340 - val_accuracy: 0.9476
Epoch 79/100
210/210 [==============================] - 45s 217ms/step - loss: 0.0140 -
accuracy: 0.9940 - val_loss: 0.3494 - val_accuracy: 0.9286
Epoch 80/100
210/210 [==============================] - 46s 219ms/step - loss: 0.1501 -
accuracy: 0.9582 - val_loss: 0.3997 - val_accuracy: 0.8952
Epoch 81/100
210/210 [==============================] - 48s 230ms/step - loss: 0.1061 -
accuracy: 0.9689 - val_loss: 0.4375 - val_accuracy: 0.8857
Epoch 82/100
210/210 [==============================] - 49s 231ms/step - loss: 0.0451 -
accuracy: 0.9857 - val_loss: 0.3103 - val_accuracy: 0.9429
Epoch 83/100
210/210 [==============================] - 48s 231ms/step - loss: 0.0116 -
accuracy: 0.9964 - val_loss: 0.4622 - val_accuracy: 0.9190
Epoch 84/100
210/210 [==============================] - 48s 229ms/step - loss: 0.0085 -
accuracy: 0.9976 - val_loss: 0.3584 - val_accuracy: 0.9429
Epoch 85/100
210/210 [==============================] - 48s 228ms/step - loss: 0.0035 -
accuracy: 1.0000 - val_loss: 0.3679 - val_accuracy: 0.9476
Epoch 86/100
210/210 [==============================] - 46s 221ms/step - loss: 0.0028 -
accuracy: 1.0000 - val_loss: 0.3619 - val_accuracy: 0.9476
Epoch 87/100
210/210 [==============================] - 48s 231ms/step - loss: 0.0020 -
accuracy: 1.0000 - val_loss: 0.3559 - val_accuracy: 0.9476
Epoch 88/100
210/210 [==============================] - 46s 220ms/step - loss: 0.0018 -
accuracy: 1.0000 - val_loss: 0.3505 - val_accuracy: 0.9429
Epoch 89/100
210/210 [==============================] - 49s 231ms/step - loss: 0.0016 -
accuracy: 1.0000 - val_loss: 0.3447 - val_accuracy: 0.9429
Epoch 90/100
210/210 [==============================] - 47s 223ms/step - loss: 0.0014 -
accuracy: 1.0000 - val_loss: 0.3494 - val_accuracy: 0.9429
Epoch 91/100
210/210 [==============================] - 47s 226ms/step - loss: 0.0019 -
accuracy: 1.0000 - val_loss: 0.3493 - val_accuracy: 0.9429
Epoch 92/100
210/210 [==============================] - 46s 220ms/step - loss: 0.0019 -
accuracy: 1.0000 - val_loss: 0.3481 - val_accuracy: 0.9429
Epoch 93/100
210/210 [==============================] - 49s 231ms/step - loss: 0.0020 -
accuracy: 1.0000 - val_loss: 0.3695 - val_accuracy: 0.9476
```

```
Epoch 94/100
210/210 [==============================] - 49s 232ms/step - loss: 0.0988 -
accuracy: 0.9761 - val_loss: 0.6008 - val_accuracy: 0.8810
Epoch 95/100
210/210 [==============================] - 48s 228ms/step - loss: 0.2240 -
accuracy: 0.9367 - val_loss: 0.5412 - val_accuracy: 0.8905
Epoch 96/100
210/210 [==============================] - 46s 221ms/step - loss: 0.0890 -
accuracy: 0.9713 - val_loss: 0.4062 - val_accuracy: 0.9095
Epoch 97/100
210/210 [==============================] - 46s 219ms/step - loss: 0.0346 -
accuracy: 0.9892 - val_loss: 0.5169 - val_accuracy: 0.8952
Epoch 98/100
210/210 [==============================] - 46s 220ms/step - loss: 0.0343 -
accuracy: 0.9940 - val_loss: 0.3616 - val_accuracy: 0.9381
Epoch 99/100
210/210 [==============================] - 49s 233ms/step - loss: 0.0093 -
accuracy: 0.9976 - val_loss: 0.3909 - val_accuracy: 0.9333
Epoch 100/100
210/210 [==============================] - 49s 231ms/step - loss: 0.1524 -
accuracy: 0.9570 - val_loss: 0.3472 - val_accuracy: 0.9095
Total training time: 4769.18 seconds
```

[14]:
```
model_evaluation_history = LRCN_model.evaluate(features_test,lables_test)
```

```
11/11 [==============================] - 6s 523ms/step - loss: 0.5794 -
accuracy: 0.8800
```

9 SAVING MODEL

[ ]:
```
model_evaluation_loss,model_evaluation_accuracy = model_evaluation_history
date_time_format = '%Y_%m%_%d_%H_%M_%S'
current_date_time_dt = dt.datetime.now()
current_date_time_string = dt.datetime.
 ↪strftime(current_date_time_dt,date_time_format)

model_file_name =␣
 ↪f'LRCN_model_Date_Time_{current_date_time_string}_Loss{model_evaluation_loss}_Accuracy_{mod
 ↪h5'

LRCN_model.save(model_file_name)
```

```
/usr/local/lib/python3.10/dist-packages/keras/src/engine/training.py:3000:
UserWarning: You are saving your model as an HDF5 file via `model.save()`. This
file format is considered legacy. We recommend using instead the native Keras
format, e.g. `model.save('my_model.keras')`.
  saving_api.save_model(
```

10 LOAD TRAINED MODEL

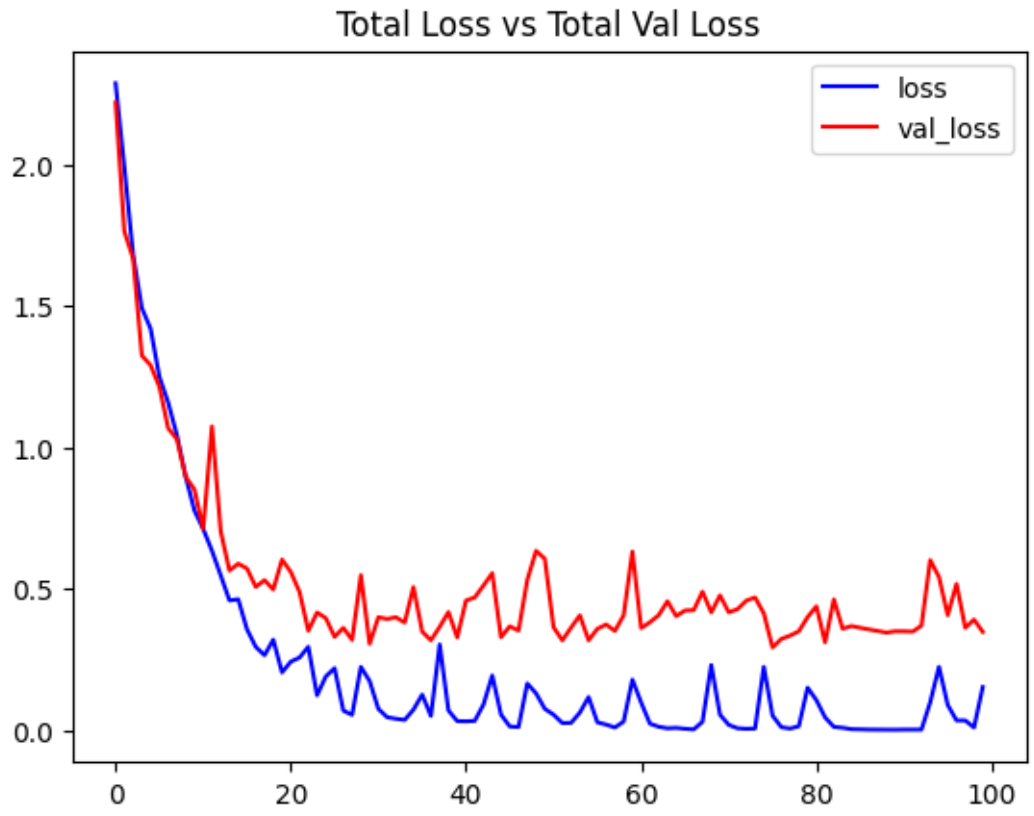
[17]:
```
from keras.models import load_model

# Load the LRCN model from a local file
model_path = '/content/drive/MyDrive/Cognitica/
  ↪LRCN_model_Date_Time_2023_09%d_09_38_43_Loss0.4353593587875366_Accuracy_0.
  ↪9142857193946838.h5'
LRCN_model = load_model(model_path)
```

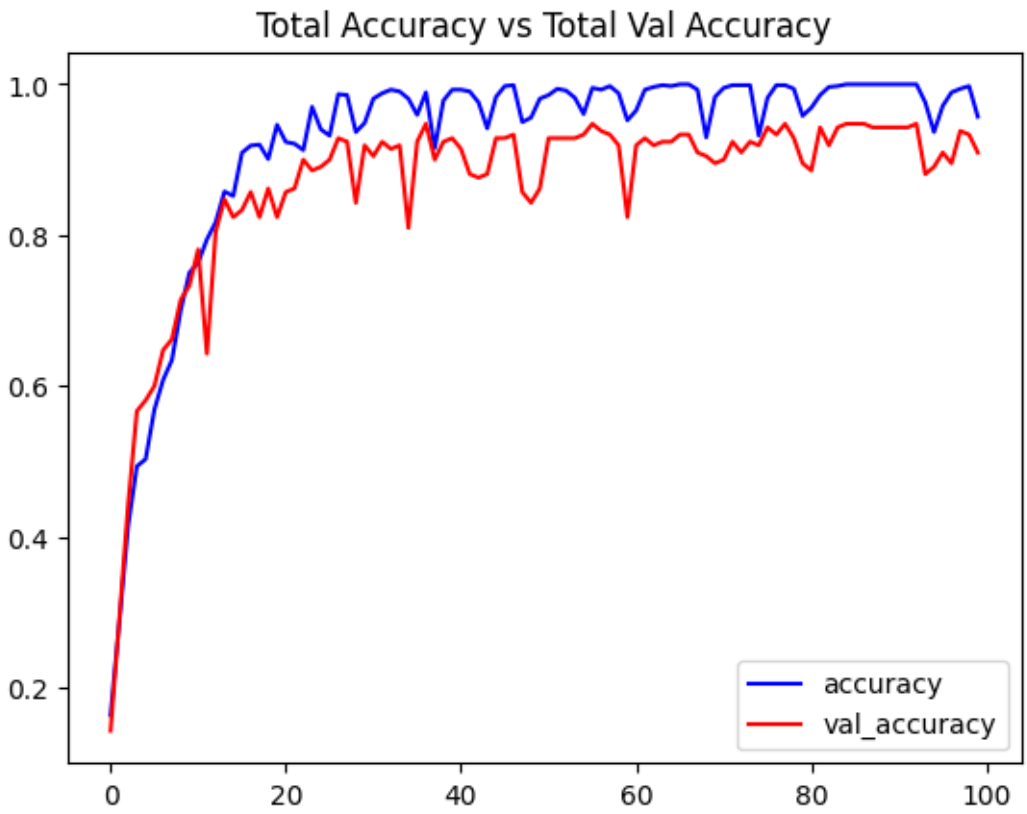
[ ]:
```
model_file_name
```

[ ]:
```
'LRCN_model_Date_Time_2023_09%d_08_47_14_Loss0.5378007292747498_Accuracy_0.88285
71438789368.h5'
```

[18]:
```
plot_metric(LRCN_model_training_history,'loss','val_loss','Total Loss vs Total␣
  ↪Val Loss')
```



[19]:
```
plot_metric(LRCN_model_training_history,'accuracy','val_accuracy','Total␣
  ↪Accuracy vs Total Val Accuracy')
```



11 TESTING

[12]:
```
!pip install moviepy
```

```
Requirement already satisfied: moviepy in /usr/local/lib/python3.10/dist-
packages (1.0.3)
Requirement already satisfied: decorator<5.0,>=4.0.2 in
/usr/local/lib/python3.10/dist-packages (from moviepy) (4.4.2)
Requirement already satisfied: tqdm<5.0,>=4.11.2 in
/usr/local/lib/python3.10/dist-packages (from moviepy) (4.66.1)
Requirement already satisfied: requests<3.0,>=2.8.1 in
/usr/local/lib/python3.10/dist-packages (from moviepy) (2.31.0)
Requirement already satisfied: proglog<=1.0.0 in /usr/local/lib/python3.10/dist-
packages (from moviepy) (0.1.10)
Requirement already satisfied: numpy>=1.17.3 in /usr/local/lib/python3.10/dist-
packages (from moviepy) (1.23.5)
Requirement already satisfied: imageio<3.0,>=2.5 in
/usr/local/lib/python3.10/dist-packages (from moviepy) (2.31.3)
Requirement already satisfied: imageio-ffmpeg>=0.2.0 in
/usr/local/lib/python3.10/dist-packages (from moviepy) (0.4.8)
```

```
Requirement already satisfied: pillow>=8.3.2 in /usr/local/lib/python3.10/dist-
packages (from imageio<3.0,>=2.5->moviepy) (9.4.0)
Requirement already satisfied: charset-normalizer<4,>=2 in
/usr/local/lib/python3.10/dist-packages (from requests<3.0,>=2.8.1->moviepy)
(3.2.0)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-
packages (from requests<3.0,>=2.8.1->moviepy) (3.4)
Requirement already satisfied: urllib3<3,>=1.21.1 in
/usr/local/lib/python3.10/dist-packages (from requests<3.0,>=2.8.1->moviepy)
(2.0.4)
Requirement already satisfied: certifi>=2017.4.17 in
/usr/local/lib/python3.10/dist-packages (from requests<3.0,>=2.8.1->moviepy)
(2023.7.22)
```

[14]:
```
import os
from moviepy.editor import VideoFileClip
test_videos_directory='test_videos'
os.makedirs(test_videos_directory,exist_ok=True)


input_video_file_path = '/content/drive/MyDrive/Cognitica/Test_dir/
 ↪v_Diving_g25_c02.avi'


video_title = os.path.splitext(os.path.basename(input_video_file_path))[0]

print(f"Video Name: {video_title}")
```

```
Video Name: v_Diving_g25_c02
```

[61]:
```
def predict_on_video(video_file_path, output_file_path, SEQUENCE_LENGTH):
    video_reader = cv2.VideoCapture(video_file_path)

    original_video_width = int(video_reader.get(cv2.CAP_PROP_FRAME_WIDTH))
    original_video_height = int(video_reader.get(cv2.CAP_PROP_FRAME_HEIGHT))

    video_writer = cv2.VideoWriter(output_file_path, cv2.
 ↪VideoWriter_fourcc('M', 'P', '4', 'V'),
                                   video_reader.get(cv2.CAP_PROP_FPS),␣
 ↪(original_video_width, original_video_height))

    frames_queue = deque(maxlen=SEQUENCE_LENGTH)

    predicted_class_name = ''

    while video_reader.isOpened():
        ok, frame = video_reader.read()
```

```
        if not ok:
            break  # Exit the loop when there are no more frames

        # Check if the frame is empty before resizing
        if not frame.size:
            continue

        resized_frame = cv2.resize(frame, (IMAGE_HEIGHT, IMAGE_WIDTH))
        normalized_frame = resized_frame / 255
        frames_queue.append(normalized_frame)

        if len(frames_queue) == SEQUENCE_LENGTH:
            predicted_labels_probabilities = LRCN_model.predict(np.
 ↪expand_dims(frames_queue, axis=0))[0]
            predicted_label = np.argmax(predicted_labels_probabilities)

            predicted_class_name = CLASSES_LIST[predicted_label]

            cv2.putText(frame, predicted_class_name, (10, 30), cv2.
 ↪FONT_HERSHEY_SIMPLEX, 1, (0, 255, 0), 2)
            video_writer.write(frame)

    video_reader.release()
    video_writer.release()
```

[16]:
```
%%capture


output_dir = "/content/drive/MyDrive/Cognitica/Output"
output_video_file_path = f'{output_dir}/
 ↪{video_title}-Output-SeqLen{SEQUENCE_LENGTH}.mp4'
predict_on_video(input_video_file_path, output_video_file_path, SEQUENCE_LENGTH)


processed_video = VideoFileClip(output_video_file_path, audio=False,␣
 ↪target_resolution=(300, None))
processed_video.ipython_display()
```

```
WARNING:py.warnings:/usr/local/lib/python3.10/dist-
packages/moviepy/video/io/ffmpeg_reader.py:123: UserWarning: Warning: in file
/content/drive/MyDrive/Cognitica/Output/v_Diving_g25_c02-Output-SeqLen20.mp4,
360000 bytes wanted but 0 bytes read,at frame 86/87, at time 2.87/2.87 sec.
Using the last valid frame instead.
  warnings.warn("Warning: in file %s, "%(self.filename)+
```

[17]:
```
processed_video.ipython_display()
```

```
Moviepy - Building video __temp__.mp4.
Moviepy - Writing video __temp__.mp4



Moviepy - Done !
Moviepy - video ready __temp__.mp4
```

[17]:
```
<moviepy.video.io.html_tools.HTML2 object>
```

[20]:
```
%%capture
input_video_file_path = '/content/drive/MyDrive/Cognitica/Test_dir/
 ↪v_HighJump_g25_c04.avi'


video_title = os.path.splitext(os.path.basename(input_video_file_path))[0]

print(f"Video Name: {video_title}")

output_dir="/content/drive/MyDrive/Cognitica/Output"
output_video_file_path = f'{output_dir}/
 ↪{video_title}-Output-SeqLen{SEQUENCE_LENGTH}.mp4'
predict_on_video(input_video_file_path,output_video_file_path,SEQUENCE_LENGTH)
processed_video = VideoFileClip(output_video_file_path, audio=False,␣
 ↪target_resolution=(300, None))
```

[21]:
```
processed_video.ipython_display()
```

```
Moviepy - Building video __temp__.mp4.
Moviepy - Writing video __temp__.mp4



Moviepy - Done !
Moviepy - video ready __temp__.mp4
```

[21]:
```
<moviepy.video.io.html_tools.HTML2 object>
```

[26]:
```
%%capture
input_video_file_path = '/content/drive/MyDrive/Cognitica/Test_dir/
 ↪v_Drumming_g25_c07.avi'


video_title = os.path.splitext(os.path.basename(input_video_file_path))[0]
```

```
print(f"Video Name: {video_title}")
output_dir="/content/drive/MyDrive/Cognitica/Output"
output_video_file_path = f'{output_dir}/
 ↪{video_title}-Output-SeqLen{SEQUENCE_LENGTH}.mp4'
predict_on_video(input_video_file_path,output_video_file_path,SEQUENCE_LENGTH)
processed_video = VideoFileClip(output_video_file_path, audio=False,␣
 ↪target_resolution=(300, None))
```

[27]:
```
processed_video.ipython_display()
```

```
Moviepy - Building video __temp__.mp4.
Moviepy - Writing video __temp__.mp4



Moviepy - Done !
Moviepy - video ready __temp__.mp4
```

[27]:
```
<moviepy.video.io.html_tools.HTML2 object>
```

[28]:
```
%%capture
input_video_file_path = '/content/drive/MyDrive/Cognitica/Test_dir/
 ↪v_HorseRace_g25_c04.avi'


video_title = os.path.splitext(os.path.basename(input_video_file_path))[0]


print(f"Video Name: {video_title}")
output_dir="/content/drive/MyDrive/Cognitica/Output"
output_video_file_path = f'{output_dir}/
 ↪{video_title}-Output-SeqLen{SEQUENCE_LENGTH}.mp4'
predict_on_video(input_video_file_path,output_video_file_path,SEQUENCE_LENGTH)
processed_video = VideoFileClip(output_video_file_path, audio=False,␣
 ↪target_resolution=(300, None))
```

[29]:
```
processed_video.ipython_display()
```

```
Moviepy - Building video __temp__.mp4.
Moviepy - Writing video __temp__.mp4


t:  97%|      | 245/253 [00:00<00:00, 279.82it/s,
now=None]WARNING:py.warnings:/usr/local/lib/python3.10/dist-
packages/moviepy/video/io/ffmpeg_reader.py:123: UserWarning: Warning: in file
/content/drive/MyDrive/Cognitica/Output/v_HorseRace_g25_c04-Output-SeqLen20.mp4,
```

```
360000 bytes wanted but 0 bytes read,at frame 252/253, at time 8.41/8.41 sec.
Using the last valid frame instead.
  warnings.warn("Warning: in file %s, "%(self.filename)+


Moviepy - Done !
Moviepy - video ready __temp__.mp4
```

[29]:
```
<moviepy.video.io.html_tools.HTML2 object>
```

[30]:
```
%%capture
input_video_file_path = '/content/drive/MyDrive/Cognitica/Test_dir/
 ↪v_PushUps_g26_c04.avi'


video_title = os.path.splitext(os.path.basename(input_video_file_path))[0]


print(f"Video Name: {video_title}")
output_dir="/content/drive/MyDrive/Cognitica/Output"
output_video_file_path = f'{output_dir}/
 ↪{video_title}-Output-SeqLen{SEQUENCE_LENGTH}.mp4'
predict_on_video(input_video_file_path,output_video_file_path,SEQUENCE_LENGTH)
processed_video = VideoFileClip(output_video_file_path, audio=False,␣
 ↪target_resolution=(300, None))
```

[31]:
```
processed_video.ipython_display()
```

```
Moviepy - Building video __temp__.mp4.
Moviepy - Writing video __temp__.mp4



Moviepy - Done !
Moviepy - video ready __temp__.mp4
```

[31]:
```
<moviepy.video.io.html_tools.HTML2 object>
```

MODEL EVALUATION

[73]:
```
%%capture

true_labels = []  # true labels for each frame
predicted_labels = []  # predicted labels for each frame
#predict labels for a video
def predict_on_video(video_file_path, true_label):
    video_reader = cv2.VideoCapture(video_file_path)
```

```
    frames_queue = deque(maxlen=SEQUENCE_LENGTH)

    while video_reader.isOpened():
        ok, frame = video_reader.read()
        if not ok:
            break


        if not frame.size:
            continue

        resized_frame = cv2.resize(frame, (IMAGE_HEIGHT, IMAGE_WIDTH))
        normalized_frame = resized_frame / 255
        frames_queue.append(normalized_frame)

        if len(frames_queue) == SEQUENCE_LENGTH:

            predicted_labels_probabilities = LRCN_model.predict(np.
 ↪expand_dims(frames_queue, axis=0))[0]
            predicted_label = np.argmax(predicted_labels_probabilities)

            predicted_class_name = CLASSES_LIST[predicted_label]


            true_labels.append(true_label)
            predicted_labels.append(predicted_class_name)

    video_reader.release()

    return true_labels, predicted_labels

#test vids
class_name_mapping = {
    'v_Diving_g25_c02': 'Diving',
    'v_Drumming_g25_c07': 'Drumming',
    'v_HighJump_g25_c04': 'HighJump',
    'v_HorseRace_g25_c04': 'HorseRace',
    'v_HorseRiding_g25_c21': 'HorseRiding',
    'v_JumpingJack_g25_c07': 'JumpingJack',
    'v_PlayingTabla_g22_c04': 'PlayingTabla',
    'v_PoleVault_g17_c09': 'PoleVault',
    'v_PommelHorse_g05_c04': 'PommelHorse',
    'v_PushUps_g26_c04': 'PushUps',
}

all_true_labels = []
all_predicted_labels = []
```

```
test_videos_directory = '/content/drive/MyDrive/Cognitica/Test_dir'

for video_file in os.listdir(test_videos_directory):
    if video_file.endswith(".avi"):
        video_file_path = os.path.join(test_videos_directory, video_file)
        video_title = os.path.splitext(os.path.basename(video_file_path))[0]

        # Map video_title
        class_name = class_name_mapping.get(video_title, 'Unknown')


        true_labels_video, predicted_labels_video =␣
 ↪predict_on_video(video_file_path, true_label=class_name)

        all_true_labels.extend(true_labels_video)
        all_predicted_labels.extend(predicted_labels_video)
```

CLASSIFICATION REPORT

[75]:
```
with warnings.catch_warnings():
    warnings.filterwarnings("ignore", category=UndefinedMetricWarning)

    report = classification_report(all_true_labels, all_predicted_labels)

    print(report)
```

```
              precision    recall  f1-score   support

      Diving       0.86      1.00      0.92       516
    Drumming       1.00      1.00      1.00      1405
    HighJump       1.00      0.39      0.56       231
   HorseRace       1.00      1.00      1.00       504
 HorseRiding       0.99      1.00      0.99      2210
 JumpingJack       1.00      1.00      1.00       483
PlayingTabla       1.00      1.00      1.00      2943
   PoleVault       0.97      0.98      0.98      1520
 PommelHorse       0.96      1.00      0.98       275
     PushUps       1.00      1.00      1.00       216

    accuracy                           0.98     10303
   macro avg       0.98      0.94      0.94     10303
weighted avg       0.99      0.98      0.98     10303
```

CONFUSION MATRIX

[76]:
```
confusion = confusion_matrix(all_true_labels, all_predicted_labels)

print("Confusion Matrix:")
print(confusion)
```

```
Confusion Matrix:
[[ 516    0    0    0    0    0    0    0    0    0]
 [   0 1405    0    0    0    0    0    0    0    0]
 [  84    0   90    0    0    0    0   45   12    0]
 [   0    0    0  504    0    0    0    0    0    0]
 [   0    0    0    0 2210    0    0    0    0    0]
 [   0    0    0    0    0  483    0    0    0    0]
 [   0    0    0    0    0    0 2943    0    0    0]
 [   0    0    0    0   24    0    0 1496    0    0]
 [   0    0    0    0    0    0    0    0  275    0]
 [   0    0    0    0    0    0    0    0    0  216]]
```