Sign Language Detection

using LSTM model

DONE BY : DRISHTI
1928228
CSSE-03
GUIDED BY : PROF. PRADEEP KANDULA

OBJECTIVE

To create a sign language detection system using Keras' Long Short Term Memory model and Dense model. It's objective would be identifying words through actions in real time on any simplified system.

INTRODUCTION

1. What does sign language represnt and why is it important?
2. What purpose does the sign language detection system serve?
3. How does it work?

The community of the deaf and dumb uses gestures to communicate in sign language. This group relied on sign language for communication when it was hard to transmit audio or when typing and writing were challenging but there was still the potential of vision.
Sign language has three components

a. Fingerspelling : used to spell words letter by letter
b. Word level sign vocabulary : used for majority of communication
c. Non-manual features : facial expression and tongue, mouth and body position

Any movement of a bodily part, such as the hand or face, is a type of gesture. Here, we are utilizing sequence processing and deep learning for gesture identification. Gesture recognition makes it possible for computers to comprehend human behavior and serves as a translator between computers and people. This could make it possible for people to communicate organically with computers without coming into contact with any mechanical parts physically.

Image feed from the camera

Image resize

Normalization

Data store of RGB image

Partition training and testing data

Trainign LSTM and Dense based predictor

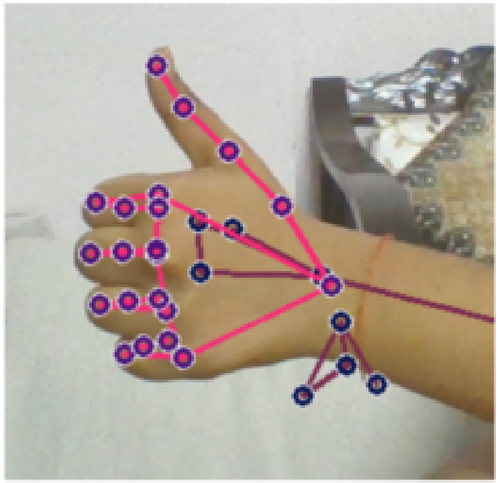
Fine tune the architecture and weights for the model

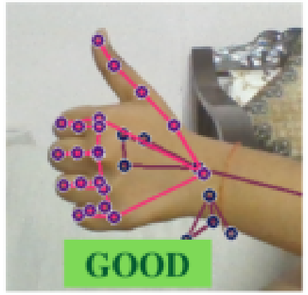
if no

Check if the results match the desired accuracy

If yes

Real time testing using LSTM and Dense model under tensorflow adn keras





Output

NEW MODULES UTILIZED

01

Tensorflow

- Python library for quick numerical computation.
- Can work with huge data set with unique properties.

02

OpenCV

- Can speed up the incorporation of A.I. into products.
- Written up in basic langaugae C, so can be tweaked to personlize wish.

03

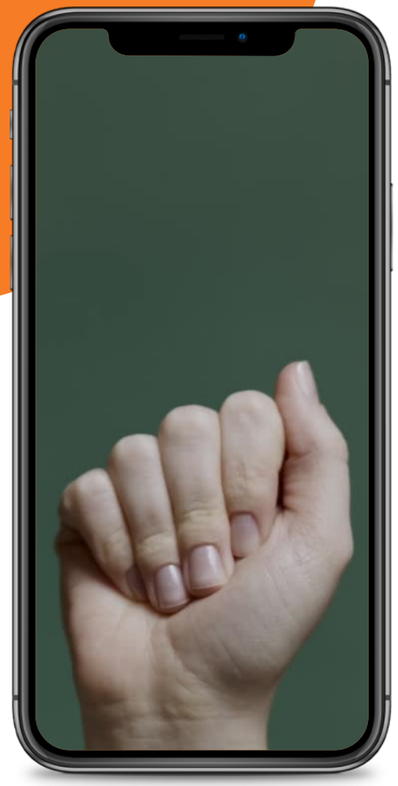
MediaPipe

- Quick prototyping for perception.
- For live and streaming videos it provides adaptable machine learning solutions.

04

Keras

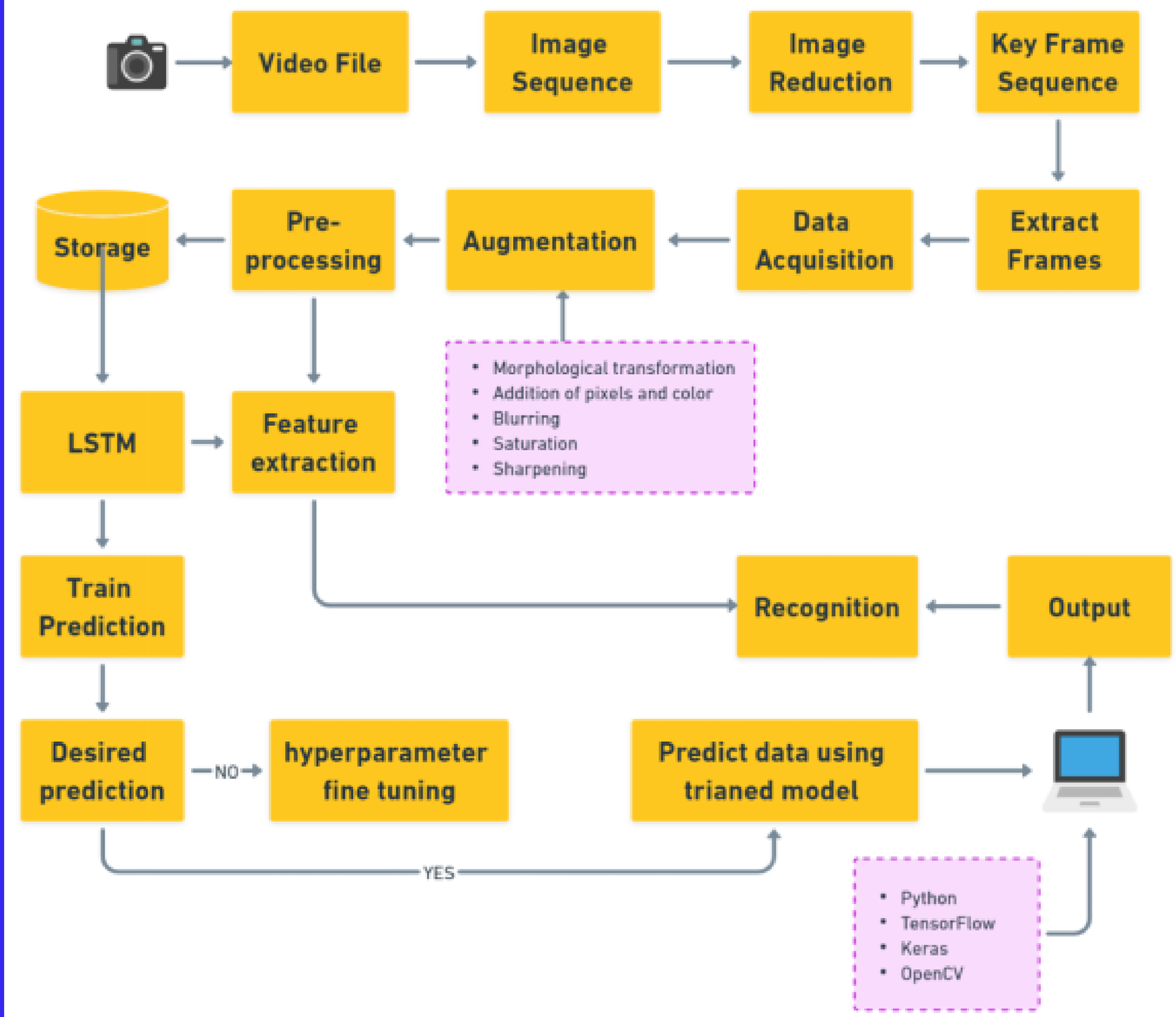
- Implementation of neural network is simple
- Multiple backend NN computation is supported.

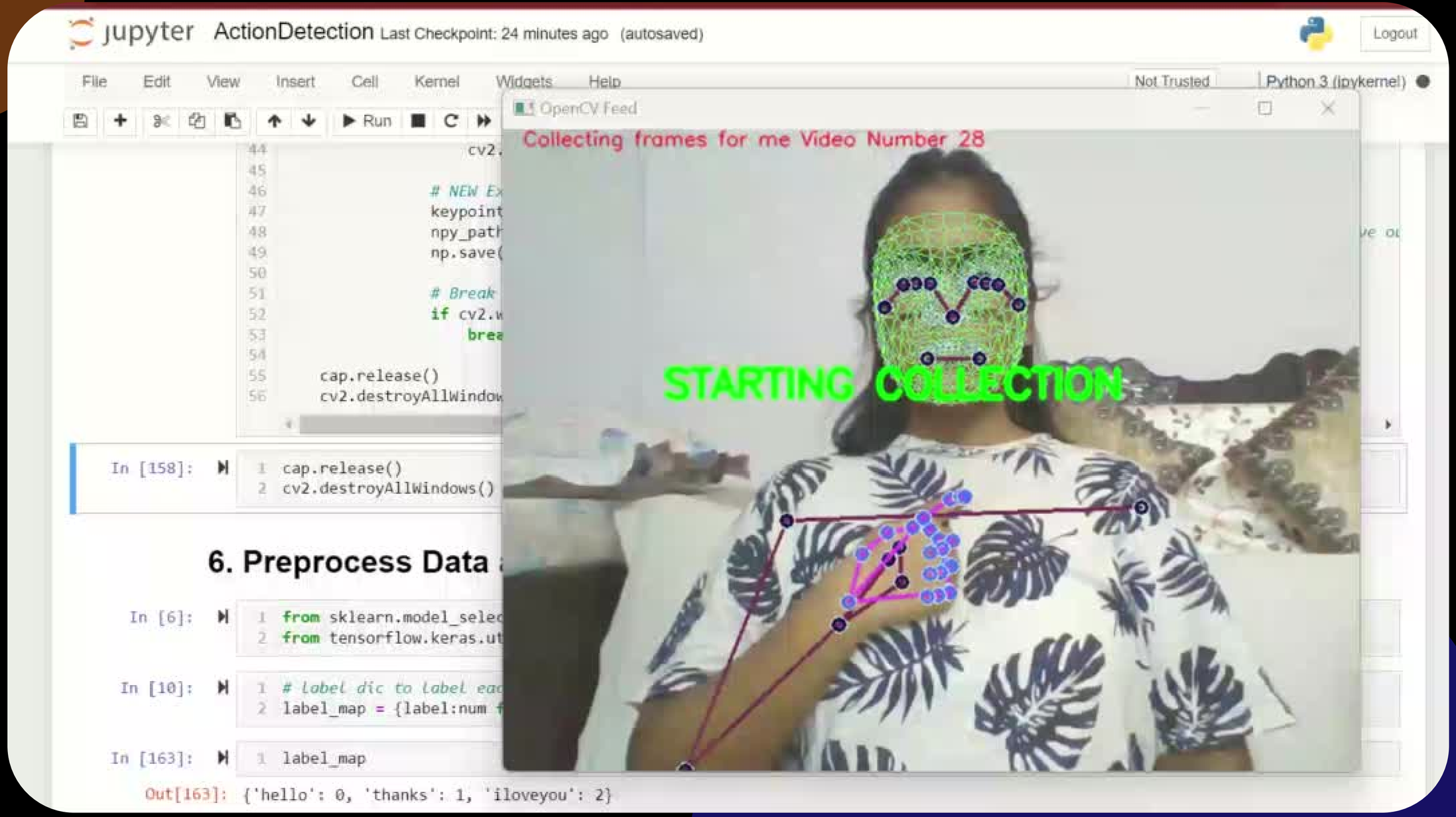


MOTIVATION BEHIND IT

Human computer interaction has several potential uses for sign language and hand gestures, which are effective forms of human communication. When compared to conventional devices, vision-based hand gesture recognition techniques have many demonstrated advantages. Nevertheless, sign language gesture recognition is a challenging problem, and the current work only makes a small contribution to getting the outcomes required. This pproject introduced a vision-based system capable of deciphering discrete hand gestures and assisting with action-based communication. In order to forecast the activity and even indicate the likelihood, this study relied on LSTM and Dense models.

BLOCK DIAGRAM




jupyter ActionDetection Last Checkpoint: 24 minutes ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
OpenCV Feed
Collecting frames for me Video Number 28
STARTING COLLECTION
cap.release()
cv2.destroyAllWindows()
6. Preprocess Data
label_map
{'hello': 0, 'thanks': 1, 'iloveyou': 2}

Jupyter ActionDetection-Copy1 Last Checkpoint: 3 hours ago (unsaved changes) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted gputest

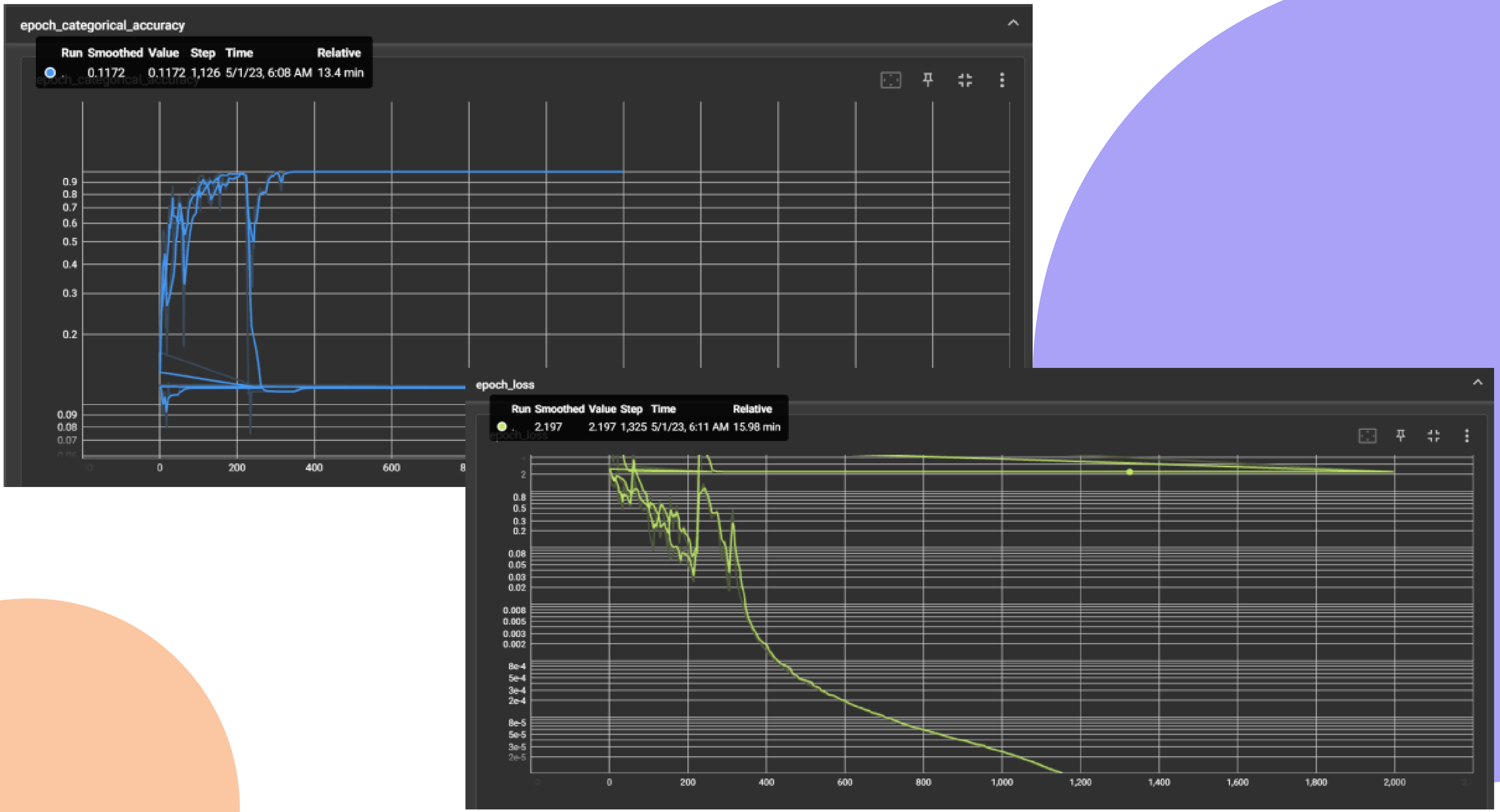
Run Code

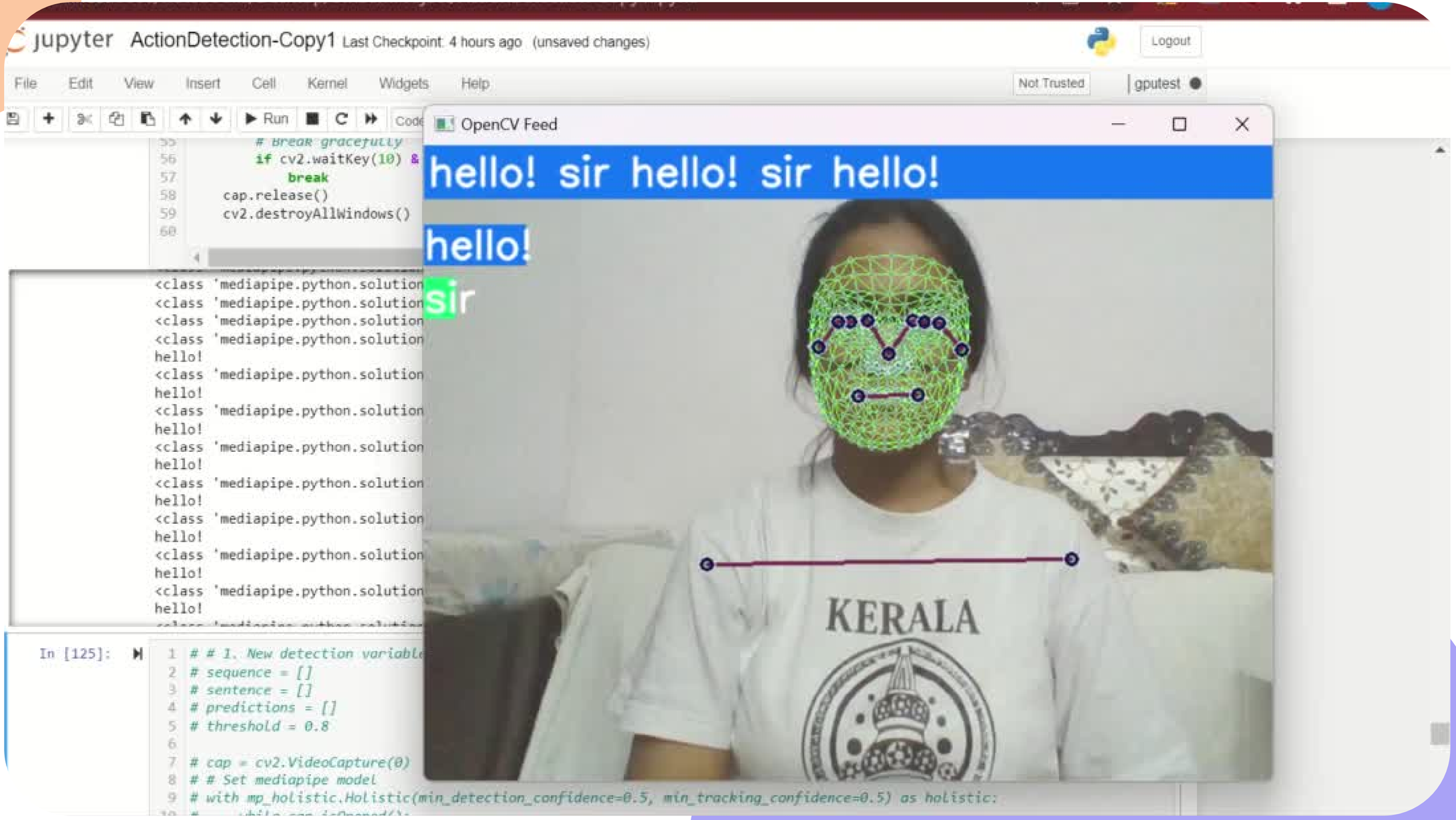
```
Epoch 1737/2000
4/4 [==============================] - 0s 101ms/step - loss: 3.7645e-07 - categorical_accuracy: 1.0000
Epoch 1738/2000
4/4 [==============================] - 0s 100ms/step - loss: 3.7645e-07 - categorical_accuracy: 1.0000
Epoch 1739/2000
4/4 [==============================] - 0s 101ms/step - loss: 3.7645e-07 - categorical_accuracy: 1.0000
Epoch 1740/2000
4/4 [==============================] - 0s 109ms/step - loss: 3.7018e-07 - categorical_accuracy: 1.0000
Epoch 1741/2000
4/4 [==============================] - 0s 108ms/step - loss: 3.7018e-07 - categorical_accuracy: 1.0000
Epoch 1742/2000
4/4 [==============================] - 0s 97ms/step - loss: 3.6808e-07 - categorical_accuracy: 1.0000
Epoch 1743/2000
4/4 [==============================] - 0s 106ms/step - loss: 3.6599e-07 - categorical_accuracy: 1.0000
Epoch 1744/2000
4/4 [==============================] - 0s 100ms/step - loss: 3.6181e-07 - categorical_accuracy: 1.0000
Epoch 1745/2000
4/4 [==============================] - 0s 100ms/step - loss: 3.6286e-07 - categorical_accuracy: 1.0000
Epoch 1746/2000
4/4 [==============================] - 0s 100ms/step - loss: 3.5972e-07 - categorical_accuracy: 1.0000
Epoch 1747/2000
4/4 [==============================] - 0s 97ms/step - loss: 3.5763e-07 - categorical_accuracy: 1.0000
Epoch 1748/2000
4/4 [==============================] - 0s 105ms/step - loss: 3.5658e-07 - categorical_accuracy: 1.0000
Epoch 1749/2000
4/4 [==============================] - 0s 98ms/step - loss: 3.5763e-07 - categorical_accuracy: 1.0000
Epoch 1750/2000
4/4 [==============================] - 0s 102ms/step - loss: 3.5658e-07 - categorical_accuracy: 1.0000
Epoch 1751/2000
4/4 [==============================] - 0s 103ms/step - loss: 3.5554e-07 - categorical_accuracy: 1.0000
Epoch 1752/2000
4/4 [==============================] - 0s 104ms/step - loss: 3.4612e-07 - categorical_accuracy: 1.0000
Epoch 1753/2000
3/4 [=====================>........] - ETA: 0s - loss: 3.1913e-07 - categorical_accuracy: 1.0000
```

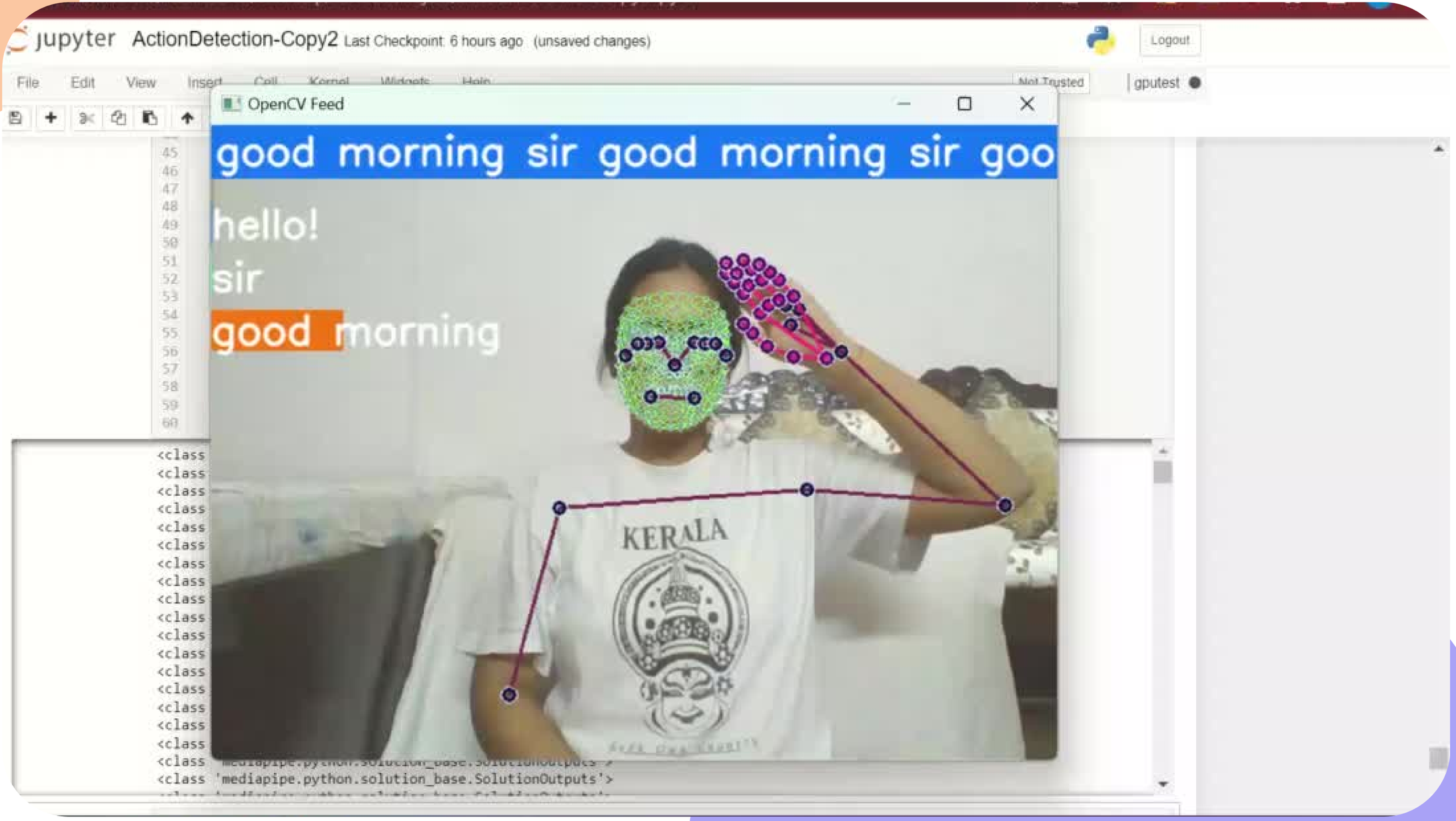
In [83]:

```
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm (LSTM)                  (None, 30, 64)            442112
_________________________________________________________________
lstm_1 (LSTM)                (None, 30, 128)           98816
_________________________________________________________________
```


epoch_categorical_accuracy
Run Smoothed Value Step Time Relative
. 0.1172 0.1172 1,126 5/1/23, 6:08 AM 13.4 min
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.09
0.08
0.07
0
200
400
600
epoch_loss
Run Smoothed Value Step Time Relative
. 2.197 2.197 1,325 5/1/23, 6:11 AM 15.98 min
2
0.8
0.5
0.3
0.2
0.08
0.05
0.03
0.02
0.008
0.005
0.003
0.002
8e-4
5e-4
3e-4
2e-4
8e-5
5e-5
3e-5
2e-5
0
200
400
600
800
1,000
1,200
1,400
1,600
1,800
2,000


jupyter ActionDetection-Copy1 Last Checkpoint: 4 hours ago (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
gputest
Run
OpenCV Feed
hello! sir hello! sir hello!
hello!
sir
KERALA
break gracefully
if cv2.waitKey(10) &
break
cap.release()
cv2.destroyAllWindows()
<class 'mediapipe.python.solution
hello!
In [125]:
1. New detection variable
sequence = []
sentence = []
predictions = []
threshold = 0.8
cap = cv2.VideoCapture(0)
Set mediapipe model
with mp_holistic.Holistic(min_detection_confidence=0.5, min_tracking_confidence=0.5) as holistic:


jupyter ActionDetection-Copy2 Last Checkpoint: 6 hours ago (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
gputest
OpenCV Feed
good morning sir good morning sir goo
hello!
sir
good morning
KERALA
<class 'mediapipe.python.solution_base.SolutionOutputs'>

LINK TO THE PROJECT :
https://github.com/Drishti228/Sign-Language-detection-using-LSTM

LINK TO MY WEBSITE :
https://drishti228.github.io/portfolio/

THANK YOU