Jyothi Engineering College
NAAC Accredited College with NBA Accredited Programmes*
Approved by AICTE & affiliated to APJ Abdul Kalam Technological University
A CENTRE OF EXCELLENCE IN SCIENCE & TECHNOLOGY BY THE CATHOLIC ARCHDIOCESE OF TRICHUR
JYOTHI HILLS, VETTIKATTIRI P.O, CHERUTHURUTHY, THRISSUR. PIN-679531 PH : +91- 4884-259000, 274423 FAX : 04884-274777
NBA accredited B.Tech Programmes in Computer Science & Engineering, Electronics & Communication Engineering, Electrical & Electronics Engineering and Mechanical Engineering valid for the academic years 2016-2022. NBA accredited B.Tech Programme in Civil Engineering valid for the academic years 2019-2022.

Mudra Classification

Department of CSE
Jyothi Engineering College
Thrissur

June 02, 2021

Group Number : 15

Project Title : Mudra Classification

https://github.com/saurav-alt/Group15

Group Guide : Ms. Ninu Francis
Asst. Professor
Department of CSE

Group Members :

Saurav Mundanatt Satheesh Kumar	JEC17CS091
Sandra David	JEC17CS084
Sruthi Elsa Shaji	JEC17CS100
Muhammed Afthab V U	JEC17CS070

Vision of the Department

- Creating eminent and ethical leaders in the domain of Computational Sciences through quality professional education with a focus on holistic learning and excellence.

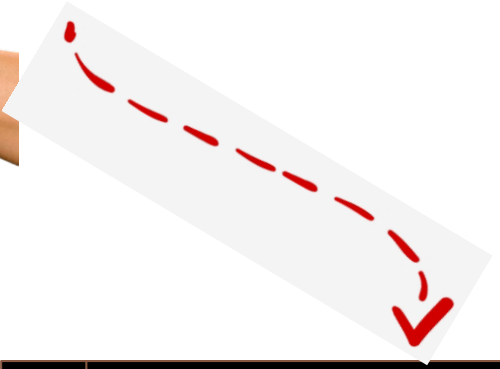
Mission of the Department

- To create technically competent and ethically conscious graduates in the field of Computer Science and Engineering by encouraging holistic learning and excellence.
- To prepare students for careers in Industry, Academia and the Government.
- To instill Entrepreneurial Orientation and research motivation among the students of the department.
- To emerge as a leader in education in the region by encouraging teaching, learning, industry and societal connect.

Abstract

- Indian classical dance has been part of Indian Culture from around 200 BC
- **The work inquires into the possibility of *identifying hasta mudras* in various classical dance forms of India**
- The review presents the details of implementation of the proposed work

Outline



Deep Learning Solution for classification of Hasta Mudras of Indian Classical Dance

Introduction

- Every classical dance form portrays a core idea, story, etc.
- Not everyone is expertise in understanding that idea, nor do everyone knows what each pose portrays
- The work attempts to find the feasibility of identifying the hasta mudra depicted in a classical dance form
- The hand data is to be preprocessed and features are to be extracted
- The algorithm used is the Faster R-CNN

Objectives

- To classify the hasta mudras of a classical dance performance
- To convey the exact meaning of a mudra to the viewer

Literature Survey

- ***On the classification of Kathakali hand gestures using support vector machines and convolutional neural networks***
  - The paper proposed an SVM model and CNN model which classifies the images
  - Considers multiple factors like background, positions, etc.
  - Classifies single hand mudras only

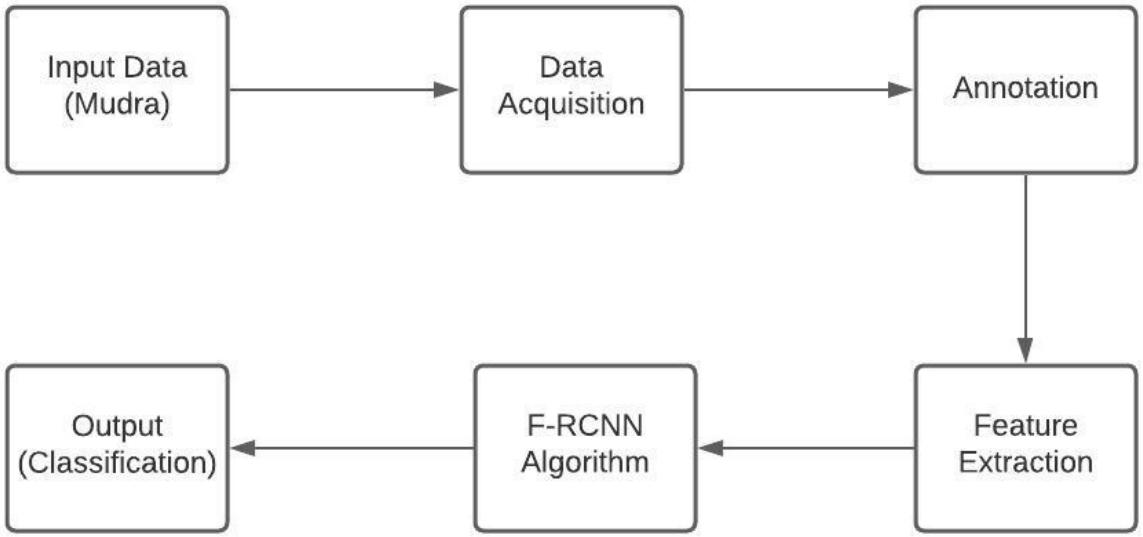
- ***Two-level classification scheme for single-hand gestures of Sattriya dance***
  - In the first level, SVM was used to classify an unknown hasta image into one of three groups and, in the second level Decision Tree classifier was used to recognize the hasta within the group
  - The accuracy during first level is 98%
  - The accuracy at second level of classification is not very encouraging

- ***Heterogeneous hand gesture recognition using 3D dynamic skeletal data***
  - The statistical representation, Fisher Vector and the temporal representation method, Temporal Pyramid are used for the encoding of descriptors
  - Takes into consideration the hand shapes, orientation and movement of hand
  - The task is time consuming as more steps are included
- ***Artificial neural network based identification and classification of images of Bharatanatya gestures***
  - Uses image- processing and also pattern recognition methods
  - The system works well with both single and double handed gestures
  - This project also identifies the health benefits of each mudra

Bharatanatyam hand gesture recognition using polygon representation

- This is effective for e-learning of 'Bharatanatyam' dance
- In the case of complex background the recognition rate falls

Architectural Diagram



Implementation

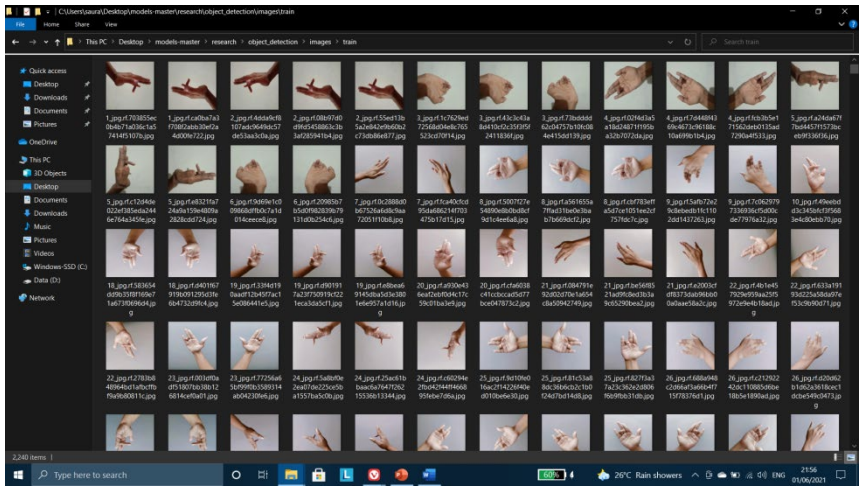
Module 1 Data Acquisition Module:

- The dataset was prepared from scratch
- Created 280 images of each mudra, totalling 1400
- 80% of the dataset was used as training data
- 20% of data images was used as test data

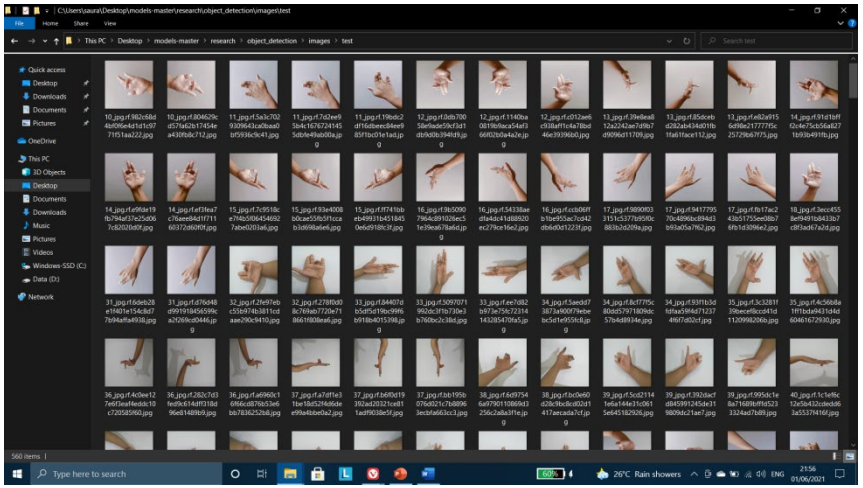
Module 2 Data Pre-processing Module:

- The process of preparing raw data and making it suitable for training
- The images were annotated
- The LabelImg tool was used for labelling of data images

Dataset

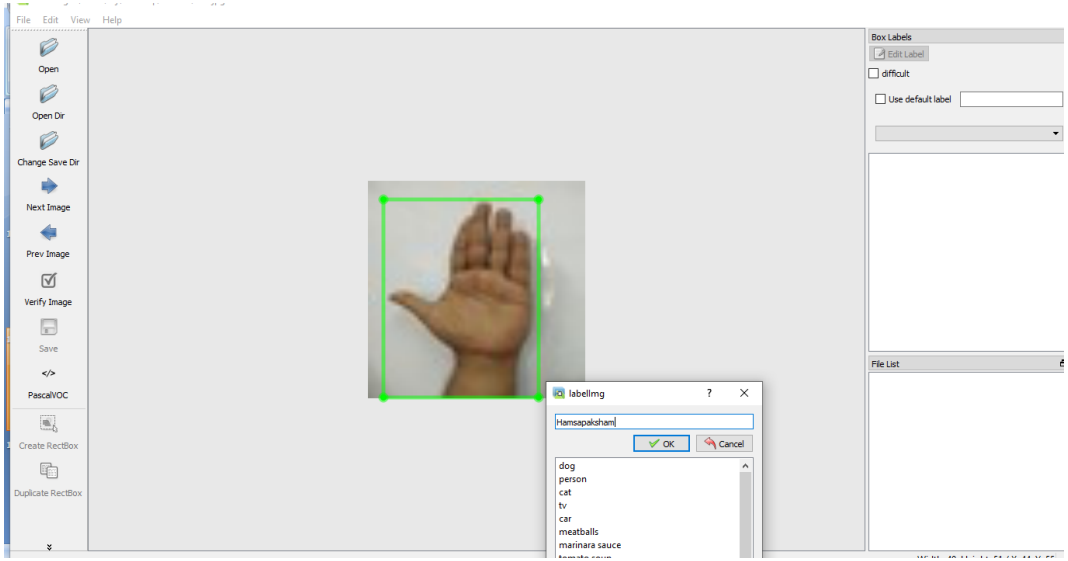


Training set



Testing set

Annotation Using LabelImg




```
<?xml version="1.0"?>
- <annotation>
    <folder>Dataset</folder>
    <filename>144.jpg</filename>
    <path>                                \Dataset\144.jpg</path>
  - <source>
      <database>Unknown</database>
    </source>
  - <size>
      <width>56</width>
      <height>56</height>
      <depth>3</depth>
    </size>
    <segmented>0</segmented>
  - <object>
      <name>Hamsapaksham</name>
      <pose>Unspecified</pose>
      <truncated>0</truncated>
      <difficult>0</difficult>
    - <bndbox>
        <xmin>5</xmin>
        <ymin>4</ymin>
        <xmax>43</xmax>
        <ymax>55</ymax>
      </bndbox>
    </object>
</annotation>
```

Module 3 Classification Module:

- The module identifies and classifies the given data
- Faster R-CNN is used for classification
- The model is trained which took a time between 7-8 hours
- Libraries used: Tensorflow, Numpy, Matplotlib, OpenCV
- Faster R-CNN makes testing section faster and accuracy greater than other methods
- Inception V2 acts as the feature extractor
- Real time input is given through the web camera
- The system is tested with unknown data for accuracy

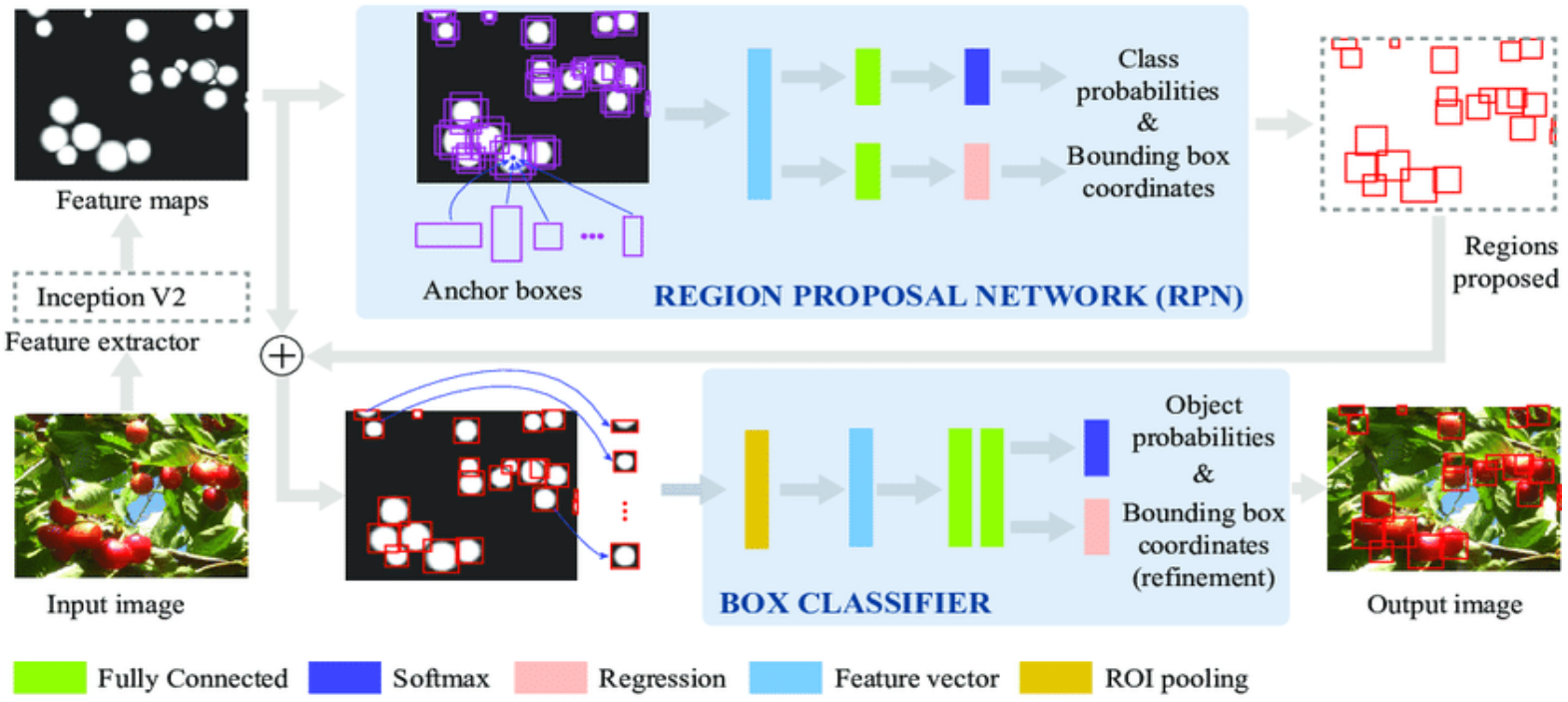
Faster R-CNN:

- Faster R-CNN is a deep convolutional network used for object detection
- Faster R-CNN fixes the problem of Selective Search algorithm by replacing it with Region Proposal Network (RPN)

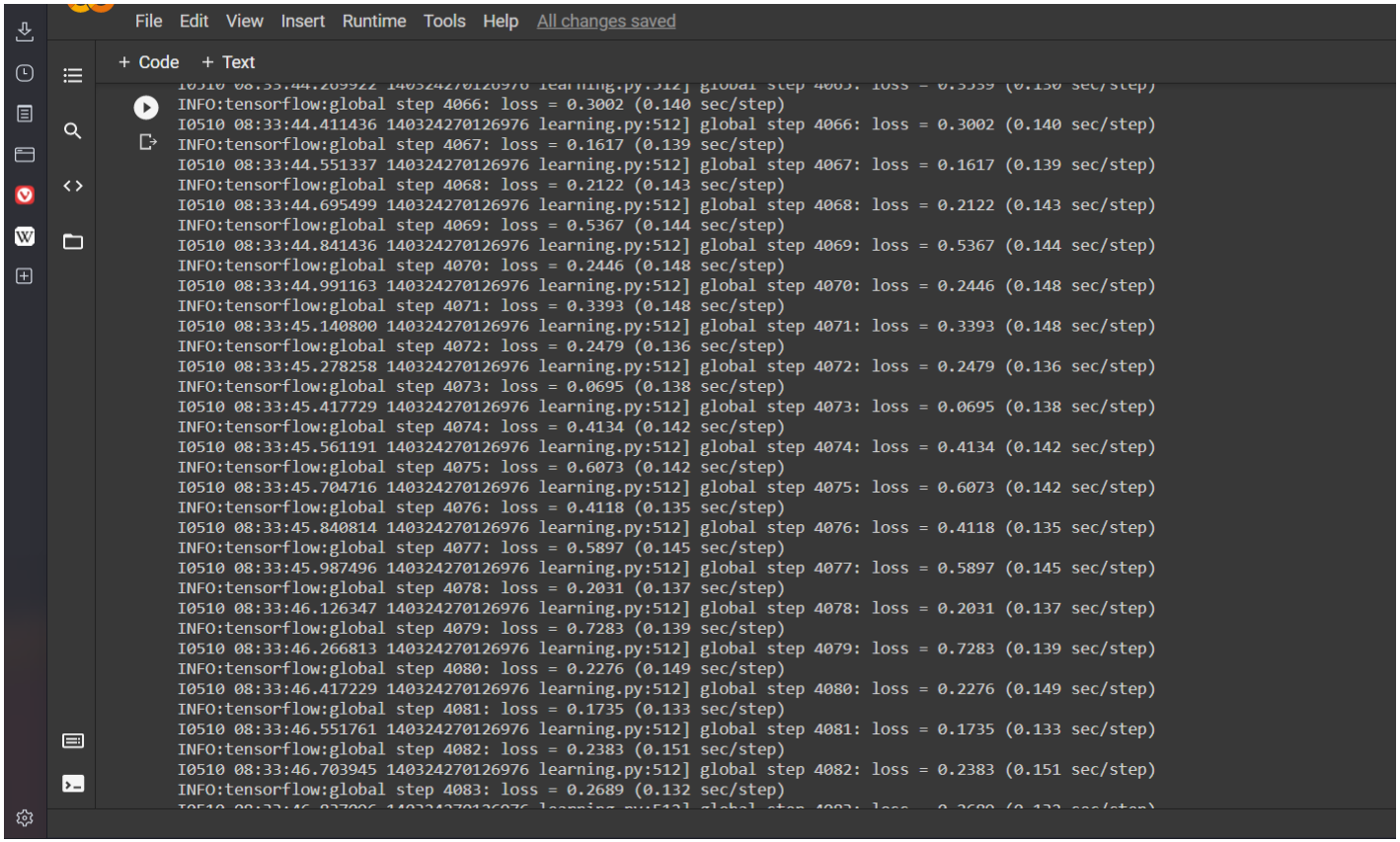
Inception V2:

- The features are extracted from the pre-processed images, helps to generate a training pattern
- Designed to reduce the dimensionality of its feature map, then performing the larger convolution

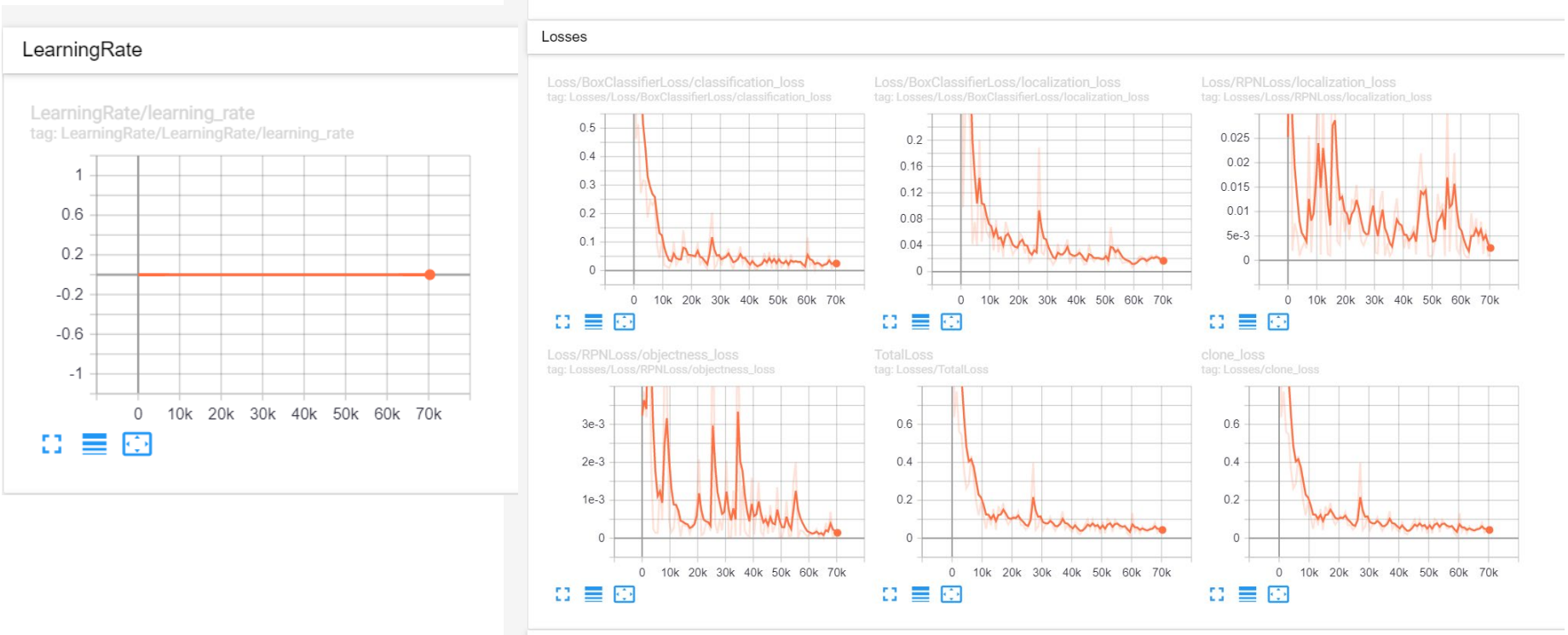
Faster R-CNN with Inception V2

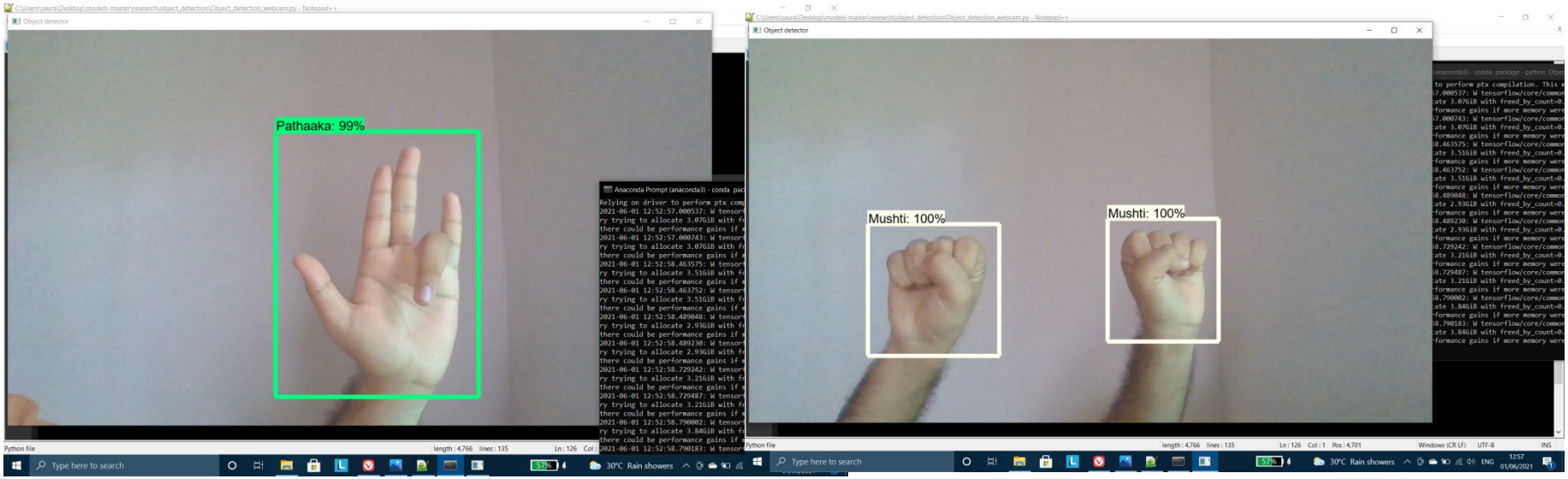


Training


File Edit View Insert Runtime Tools Help All changes saved
+ Code + Text
INFO:tensorflow:global step 4066: loss = 0.3002 (0.140 sec/step)
I0510 08:33:44.411436 140324270126976 learning.py:512] global step 4066: loss = 0.3002 (0.140 sec/step)
INFO:tensorflow:global step 4067: loss = 0.1617 (0.139 sec/step)
I0510 08:33:44.551337 140324270126976 learning.py:512] global step 4067: loss = 0.1617 (0.139 sec/step)
INFO:tensorflow:global step 4068: loss = 0.2122 (0.143 sec/step)
I0510 08:33:44.695499 140324270126976 learning.py:512] global step 4068: loss = 0.2122 (0.143 sec/step)
INFO:tensorflow:global step 4069: loss = 0.5367 (0.144 sec/step)
I0510 08:33:44.841436 140324270126976 learning.py:512] global step 4069: loss = 0.5367 (0.144 sec/step)
INFO:tensorflow:global step 4070: loss = 0.2446 (0.148 sec/step)
I0510 08:33:44.991163 140324270126976 learning.py:512] global step 4070: loss = 0.2446 (0.148 sec/step)
INFO:tensorflow:global step 4071: loss = 0.3393 (0.148 sec/step)
I0510 08:33:45.140800 140324270126976 learning.py:512] global step 4071: loss = 0.3393 (0.148 sec/step)
INFO:tensorflow:global step 4072: loss = 0.2479 (0.136 sec/step)
I0510 08:33:45.278258 140324270126976 learning.py:512] global step 4072: loss = 0.2479 (0.136 sec/step)
INFO:tensorflow:global step 4073: loss = 0.0695 (0.138 sec/step)
I0510 08:33:45.417729 140324270126976 learning.py:512] global step 4073: loss = 0.0695 (0.138 sec/step)
INFO:tensorflow:global step 4074: loss = 0.4134 (0.142 sec/step)
I0510 08:33:45.561191 140324270126976 learning.py:512] global step 4074: loss = 0.4134 (0.142 sec/step)
INFO:tensorflow:global step 4075: loss = 0.6073 (0.142 sec/step)
I0510 08:33:45.704716 140324270126976 learning.py:512] global step 4075: loss = 0.6073 (0.142 sec/step)
INFO:tensorflow:global step 4076: loss = 0.4118 (0.135 sec/step)
I0510 08:33:45.840814 140324270126976 learning.py:512] global step 4076: loss = 0.4118 (0.135 sec/step)
INFO:tensorflow:global step 4077: loss = 0.5897 (0.145 sec/step)
I0510 08:33:45.987496 140324270126976 learning.py:512] global step 4077: loss = 0.5897 (0.145 sec/step)
INFO:tensorflow:global step 4078: loss = 0.2031 (0.137 sec/step)
I0510 08:33:46.126347 140324270126976 learning.py:512] global step 4078: loss = 0.2031 (0.137 sec/step)
INFO:tensorflow:global step 4079: loss = 0.7283 (0.139 sec/step)
I0510 08:33:46.266813 140324270126976 learning.py:512] global step 4079: loss = 0.7283 (0.139 sec/step)
INFO:tensorflow:global step 4080: loss = 0.2276 (0.149 sec/step)
I0510 08:33:46.417229 140324270126976 learning.py:512] global step 4080: loss = 0.2276 (0.149 sec/step)
INFO:tensorflow:global step 4081: loss = 0.1735 (0.133 sec/step)
I0510 08:33:46.551761 140324270126976 learning.py:512] global step 4081: loss = 0.1735 (0.133 sec/step)
INFO:tensorflow:global step 4082: loss = 0.2383 (0.151 sec/step)
I0510 08:33:46.703945 140324270126976 learning.py:512] global step 4082: loss = 0.2383 (0.151 sec/step)
INFO:tensorflow:global step 4083: loss = 0.2689 (0.132 sec/step)

Results


LearningRate
LearningRate/learning_rate
tag: LearningRate/LearningRate/learning_rate
1
0.6
0.2
-0.2
-0.6
-1
0
10k
20k
30k
40k
50k
60k
70k
Losses
Loss/BoxClassifierLoss/classification_loss
tag: Losses/Loss/BoxClassifierLoss/classification_loss
0.5
0.4
0.3
0.2
0.1
0
Loss/BoxClassifierLoss/localization_loss
tag: Losses/Loss/BoxClassifierLoss/localization_loss
0.2
0.16
0.12
0.08
0.04
0
Loss/RPNLoss/localization_loss
tag: Losses/Loss/RPNLoss/localization_loss
0.025
0.02
0.015
0.01
5e-3
0
Loss/RPNLoss/objectness_loss
tag: Losses/Loss/RPNLoss/objectness_loss
3e-3
2e-3
1e-3
0
TotalLoss
tag: Losses/TotalLoss
0.6
0.4
0.2
0
clone_loss
tag: Losses/clone_loss
0.6
0.4
0.2
0


Pathaaka: 99%
Mushti: 100%
Mushti: 100%

- The dataset was populated with images for training and testing
- The model was trained with training data using Faster R-CNN algorithm which took time between 7-8 hours
- The model performed as expected with an appreciable accuracy rate for every mudra tested

Future Enhancements

- Using the existing algorithm, can be used to detect face expressions, body movements
- Detect the pattern and predict the storyline of the dance

Conclusions

- The classical dance forms play a vital role in enriching the heritage of India
- An attempt is made to explore the possibility of identifying the mudras in the classical dance performance
- The work makes an effort to help the viewer identify the exact mudra depicted in a dance form
- The viewer who aspires to enjoy the classical dance form to its fullest is benefited
- The Indian classical dance forms get proper recognition when more people get to appreciate the art forms

Reference

[1] Kumar K.V.V., Kishore P.V.V. (2018) Indian Classical Dance Mudra Classification Using HOG Features and SVM Classifier. In: Satapathy S., Bhateja V., Das S. (eds) Smart Computing and Informatics. Smart Innovation, Systems and Technologies, vol 77. Springer, Singapore

[2] L. T. Bhavanam and G. Neelakanta Iyer, "On the Classification of Kathakali Hand Gestures Using Support Vector Machines and Convolutional Neural Networks," 2020 International Conference on Artificial Intelligence and Signal Processing (AISP), Amaravati, India, 2020

[3] M. Devi and S. Saharia, "A two-level classification scheme for single-hand gestures of Sattriya dance," 2016 International Conference on Accessibility to Digital World (ICADW), Guwahati, 2016

[4] S. Saha et al., "Bharatanatyam hand gesture recognition using polygon representation," Proceedings of The 2014 International Conference on Control, Instrumentation, Energy and Communication (CIEC), Calcutta, 2014

Thank You