B.TECH FINAL YEAR PROJECT PRESENTATION

Recognition of Sattriya Dance Double-Handed Mudra from Image and Video

– PRESENTED BY –

Dishanka Kalita (200102021)

Under the Supervision of
Assistant Prof. Parismita Sarma

Department of Information Technology,
Gauhati University, Assam

OUTLINE

- INTRODUCTION
- MOTIVATION
- PROBLEMS FORMULATED
- BLOCK DIAGRAM
- METHODOLOGY
- WORK DONE
- CONCLUSION

INTRODUCTION

This project focuses on the application of advanced image processing techniques to facilitate the recognition and analysis of Sattriya dance hand mudras, a significant aspect of the traditional dance form originating from the Indian state of Assam.

Key Points:

- **Gesture Recognition in Dance Forms**
- **Vision-Based System Overview**
- **Making it Accessible: Web App Deployment**
- **Cultural Heritage Digitization**

Figure 1: Sattriya Dance

MOTIVATION

Preserve Sattriya dance cultural essence using advanced recognition tech. Enhance educational experiences and enable global cultural exchange through an adaptable, open-source project.

- Cultural Preservation
- Technological Integration
- Community Empowerment
- Global Cultural Exchange

Recognition of Sattriya Dance Double-Handed Mudra from Image and Video

TOOLS USED

GOOGLE COLAB

FOR TRAINING THE ML MODELS

ROBOFLOW

DATA ANNOTATION , AUGMENTATION TOOL

YOLOv8

STATE OF THE ART OBJECT DETECTION ALGORITHM

YOLOv5

STATE OF THE ART OBJECT DETECTION ALGORITHM

RT-DETR

TRANSFORMER BASED OBJECT DETECTION ALGORITHM

OPEN CV

A real-time optimized Computer Vision library

TENSORFLOW

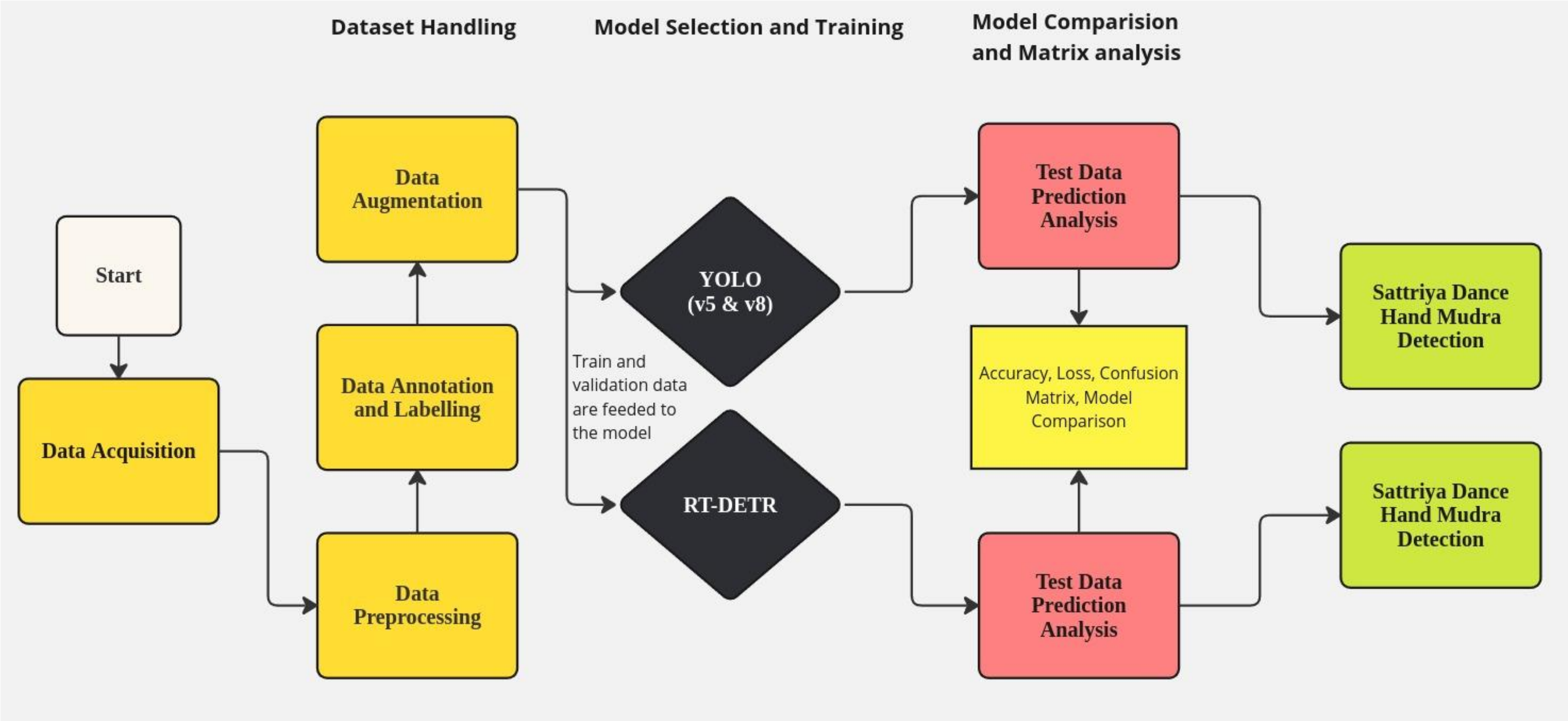
Python Machine Learning Library

FLASK

A popular Python web framework used for building web applications and APIs.

BLOCK DIAGRAM

Sattriya Dance Hand Mudra Detection



METHODOLOGY

Satriya Dance Hand Mudra Detection

STEP 1 : Data Collection and Annotation

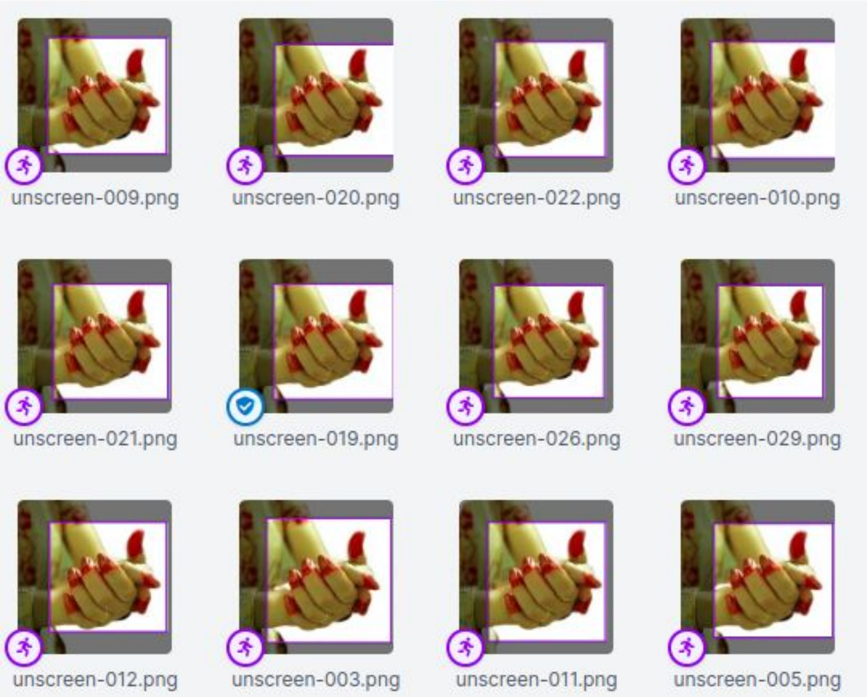


Figure 2: Dataset Handling and Annotation

METHODOLOGY

Satriya Dance Hand Mudra Detection

STEP 2 : Data Augmentation

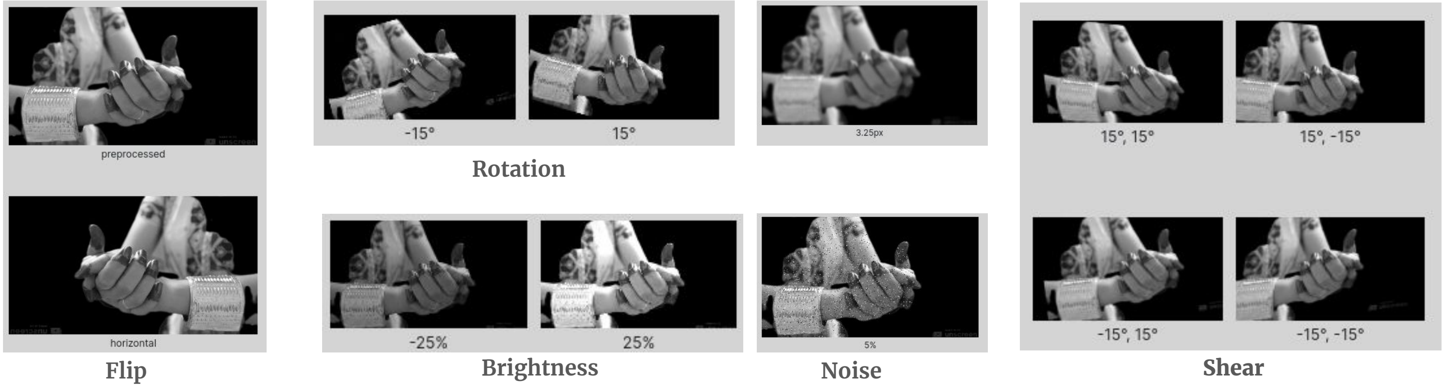


Figure 3: Data Augmentation

METHODOLOGY

Satriya Dance Hand Mudra Detection

STEP 3 : Vision Model Implementation

Here have used two state of the art models

1. You Only Look Once (YOLO v8 & v5)
2. Real Time Detection Transformer (RT-DETR)

METHODOLOGY

Sattriya Dance Hand Mudra Detection

YOLO :

- YOLO (You Only Look Once) stands out for its speed and accuracy.
- YOLO directly predicts bounding boxes and class probabilities from an image, bypassing the complex two-stage architecture of traditional object detection methods.
- YOLO version 8 is faster Compared to R-CNN, SSD, Faster R-CNN and earlier versions of Yolo

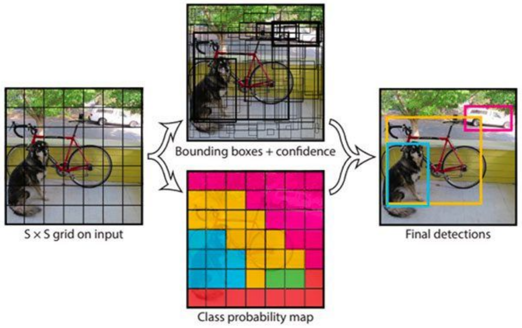


Figure 4: Overview of YOLO

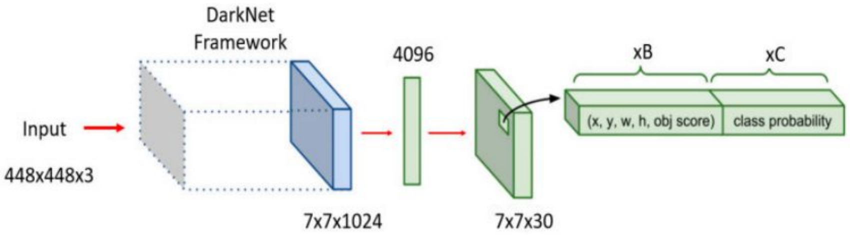


Figure 5: Architecture of Yolo

METHODOLOGY

Sattriya Dance Hand Mudra Detection

RT - DETR :

- Real-Time Detection Transformer (RT-DETR) is a state-of-the-art object detection algorithm that leverages the power of transformers in the field of Object Detection.
- Future Proof Technology

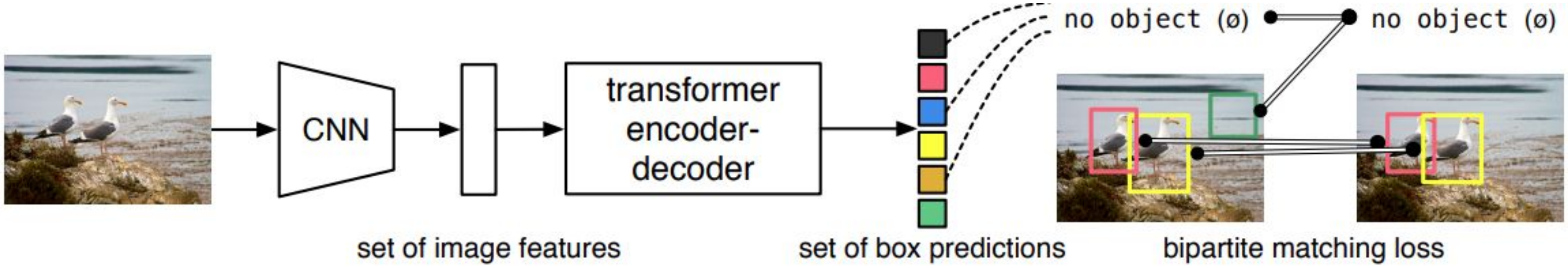


Figure 6: Structure of Detection Transformer

WORK DONE AND RESULTS

DATASETS:

- Total 5232 images are used, annotated and labeled using RoboFlow.
- The images are splitted into three sets:
    - **Training Set** : 4578 images
    - **Validation Set** : 436 images
    - **Test Set** : 218 images
- **Preprocessing** : Auto-Orient: Applied
  Resize: Stretch to 640x360
  Grayscale: Applied

- **Augmentation** : Flip: Horizontal
  Rotation: Between -15° and +15°
  Shear: ±15° Horizontal, ±15° Vertical
  Brightness: Between -25% and +25%
  Blur: Up to 3.25px
  Noise: Up to 5% of pixels

WORK DONE AND RESULTS

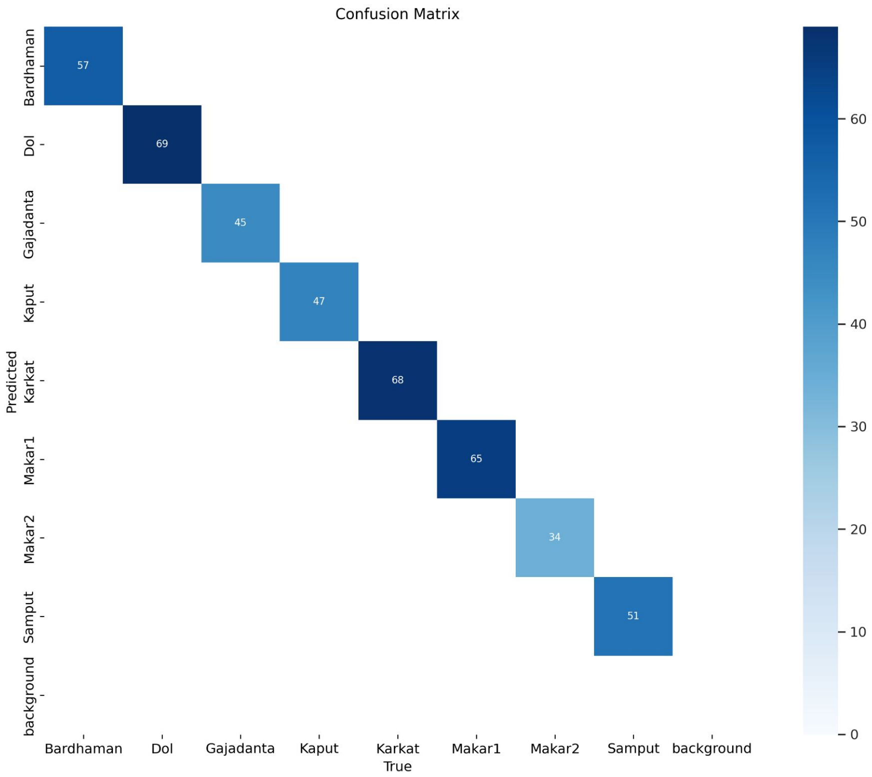


Figure 7: YOLOv8 Confusion Matrix

WORK DONE AND RESULTS

Table 1: YOLOv8 Train vs Validation Data

Category	Train Data	Validation data
mAP 50	0.995	0.995
mAP (50-90)	0.838	0.845

WORK DONE AND RESULTS

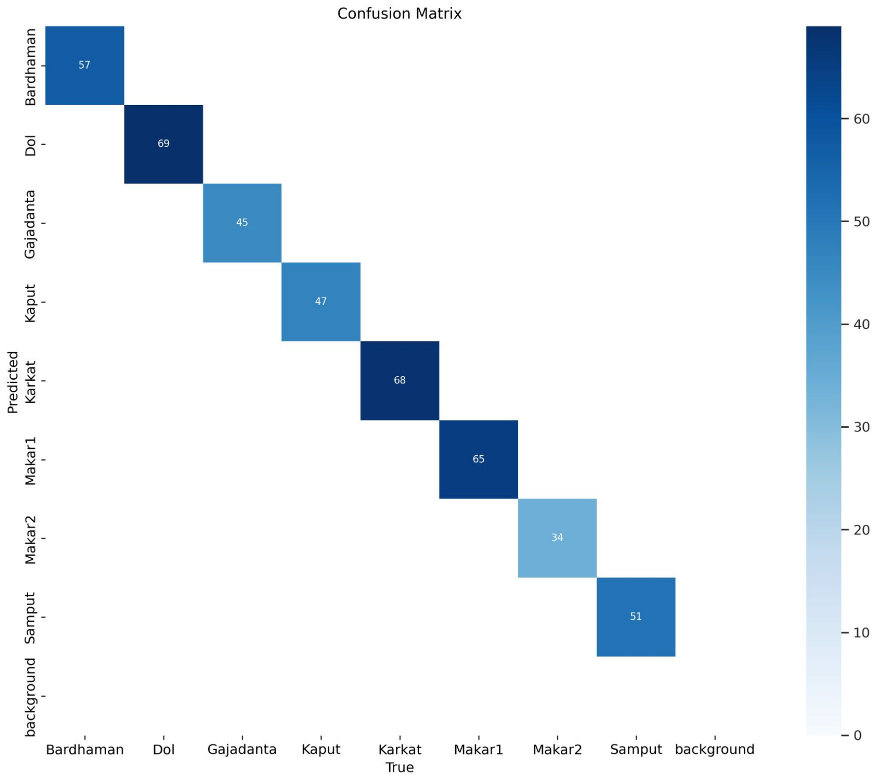


Figure 8: YOLOv5 Confusion Matrix

WORK DONE AND RESULTS

Table 2: YOLOv5 Train vs Validation Data

Category	Train Data	Validation data
mAP 50	0.995	0.995
mAP (50-90)	0.820	0.831

WORK DONE AND RESULTS

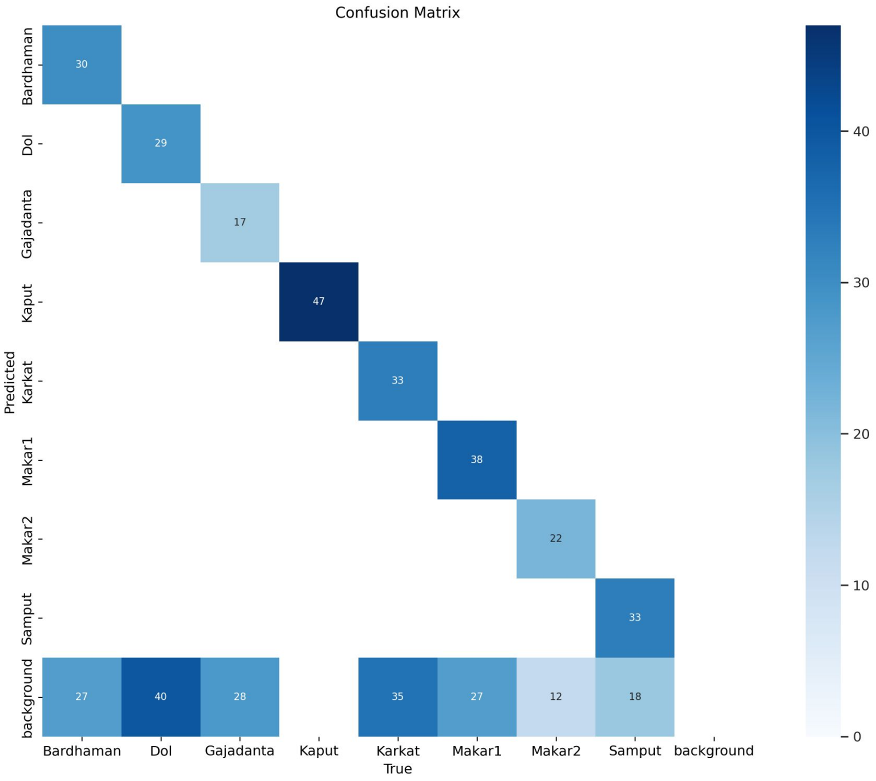


Figure 9: RT-DETR Confusion Matrix

WORK DONE AND RESULTS

Table 3: RT-DETR Train vs Validation Data

Category	Train Data	Validation data
mAP 50	0.995	0.995
mAP (50-90)	0.818	0.821

WORK DONE AND RESULTS

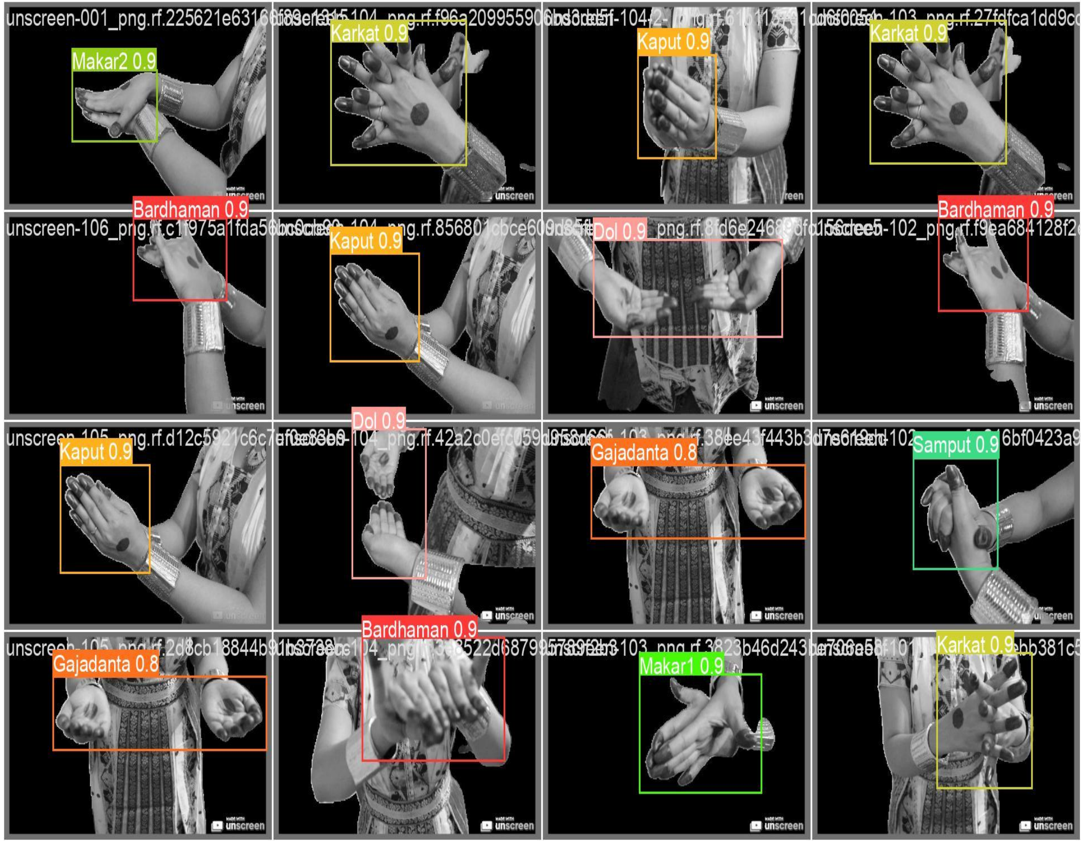


Figure 10:YOLOv8 validation data Prediction

WORK DONE AND RESULTS

Figure 11:YOLOv5 validation data Prediction

WORK DONE AND RESULTS

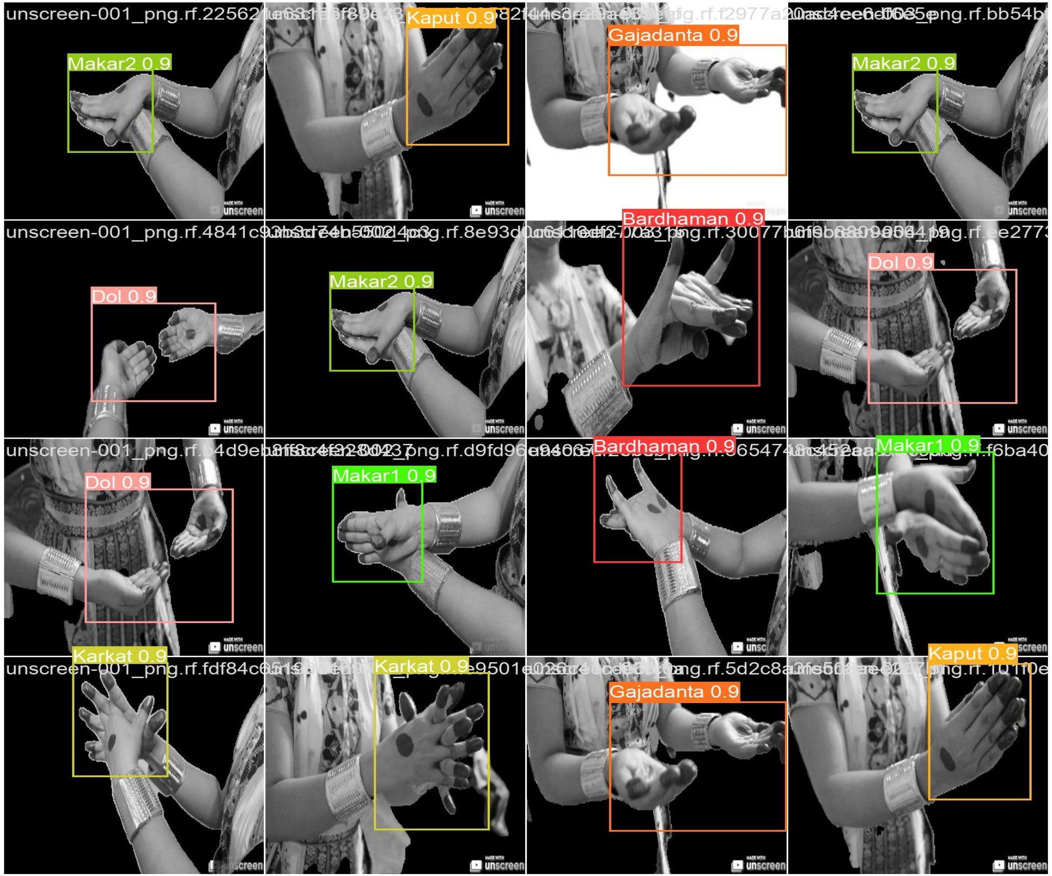


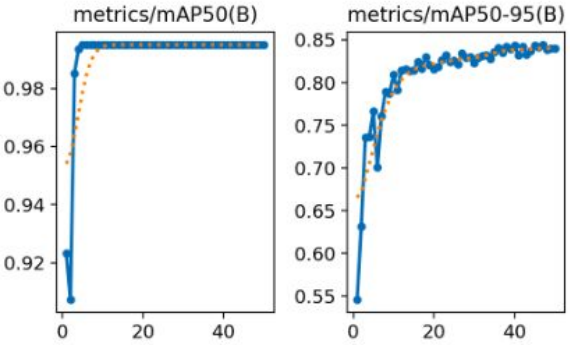
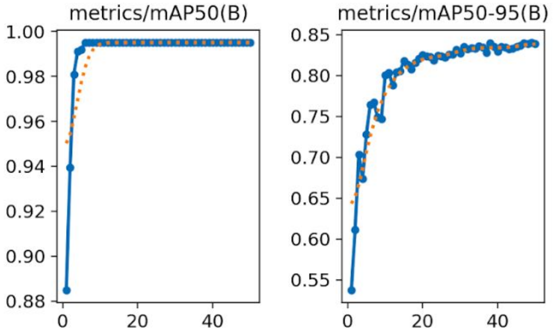
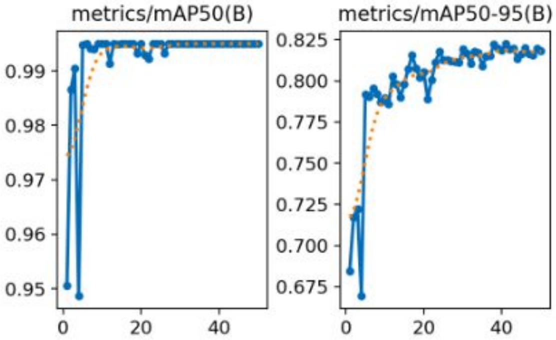
Figure 12:RT-DETR validation data Prediction

WORK DONE AND RESULTS

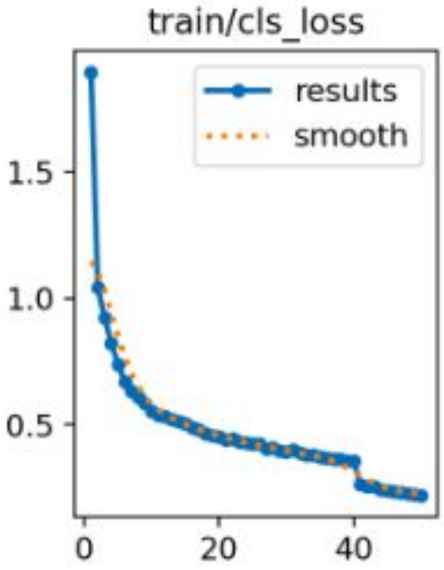
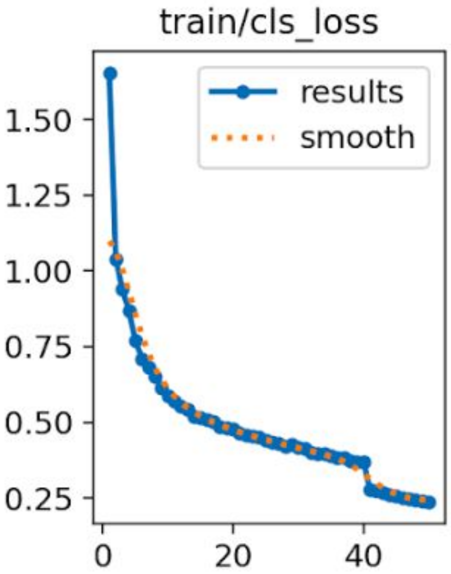
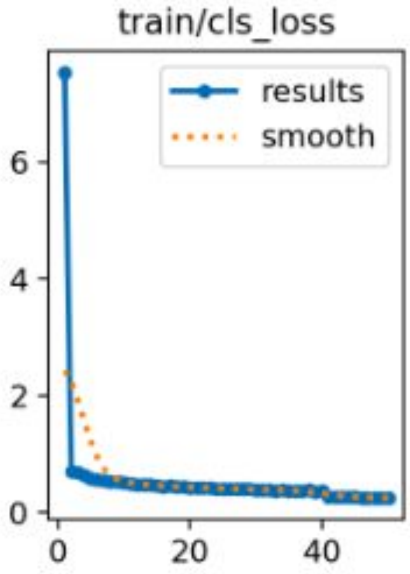
Table 4:Summary of YOLOv8, YOLOv5 and RT-DETR

YOLOv8	YOLOv5	RT-DETR
❑ The model is trained for 50 epochs ❑ Class Loss : 0.22104 ❑ mAP 50 (Mean Average Precision at IoU threshold 0.5): 0.995 ❑ mAP 50-95 (Mean Average Precision over IoU thresholds 0.5 to 0.95): 0.83758	❑ The model is trained for 50 epochs ❑ Class Loss : 0.2365 ❑ mAP 50 (Mean Average Precision at IoU threshold 0.5): 0.995 ❑ mAP 50-95 (Mean Average Precision over IoU thresholds 0.5 to 0.95): 0.820	❑ The model is trained for 50 epochs ❑ Class Loss : 0.24785 ❑ mAP 50 (Mean Average Precision at IoU threshold 0.5): 0.995 ❑ mAP 50-95 (Mean Average Precision over IoU thresholds 0.5 to 0.95): 0.81813

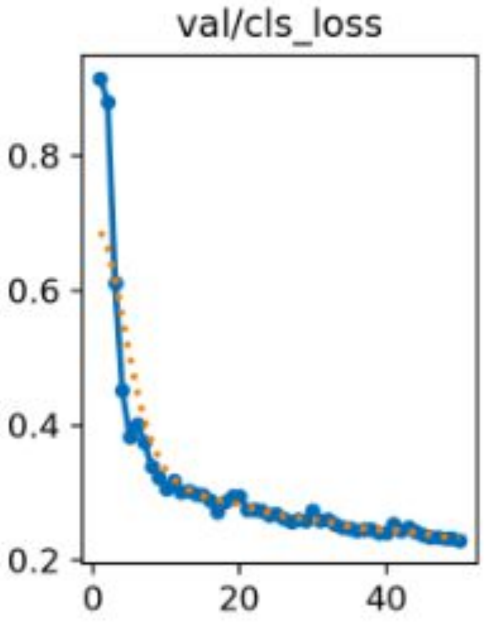
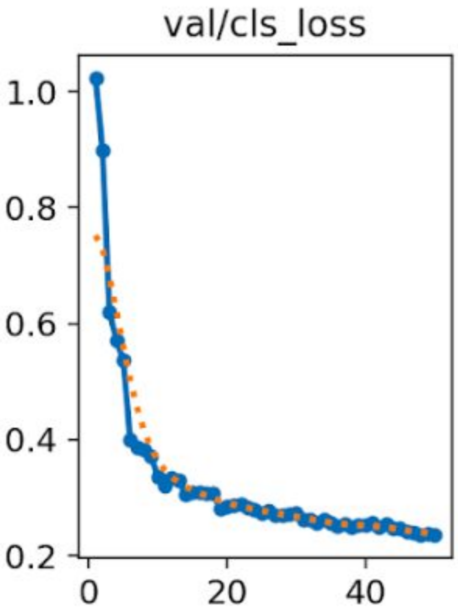
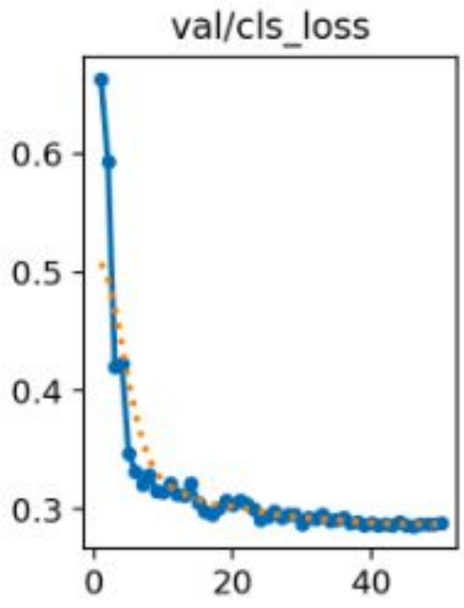
WORK DONE AND RESULTS

YOLOv8	YOLOv5	RT-DETR
		

WORK DONE AND RESULTS

YOLO v8	YOLOv5	RT-DETR
		

WORK DONE AND RESULTS

YOLOv8	YOLOv5	RT-DETR
		

WORK DONE AND RESULTS

Table 5:Comparison of YOLOv8, YOLOv5 and RT-DETR

Models	Epochs	Batch Size	CLS_Loss	Mean Average Precision(mAP50-95)	Recall
YOLOv8	50	16	0.22104	0.83758	1
YOLOv5	50	16	0.2365	0.820	1
RT-DETR	50	16	0.24785	0.8181	1

WORK DONE AND RESULTS

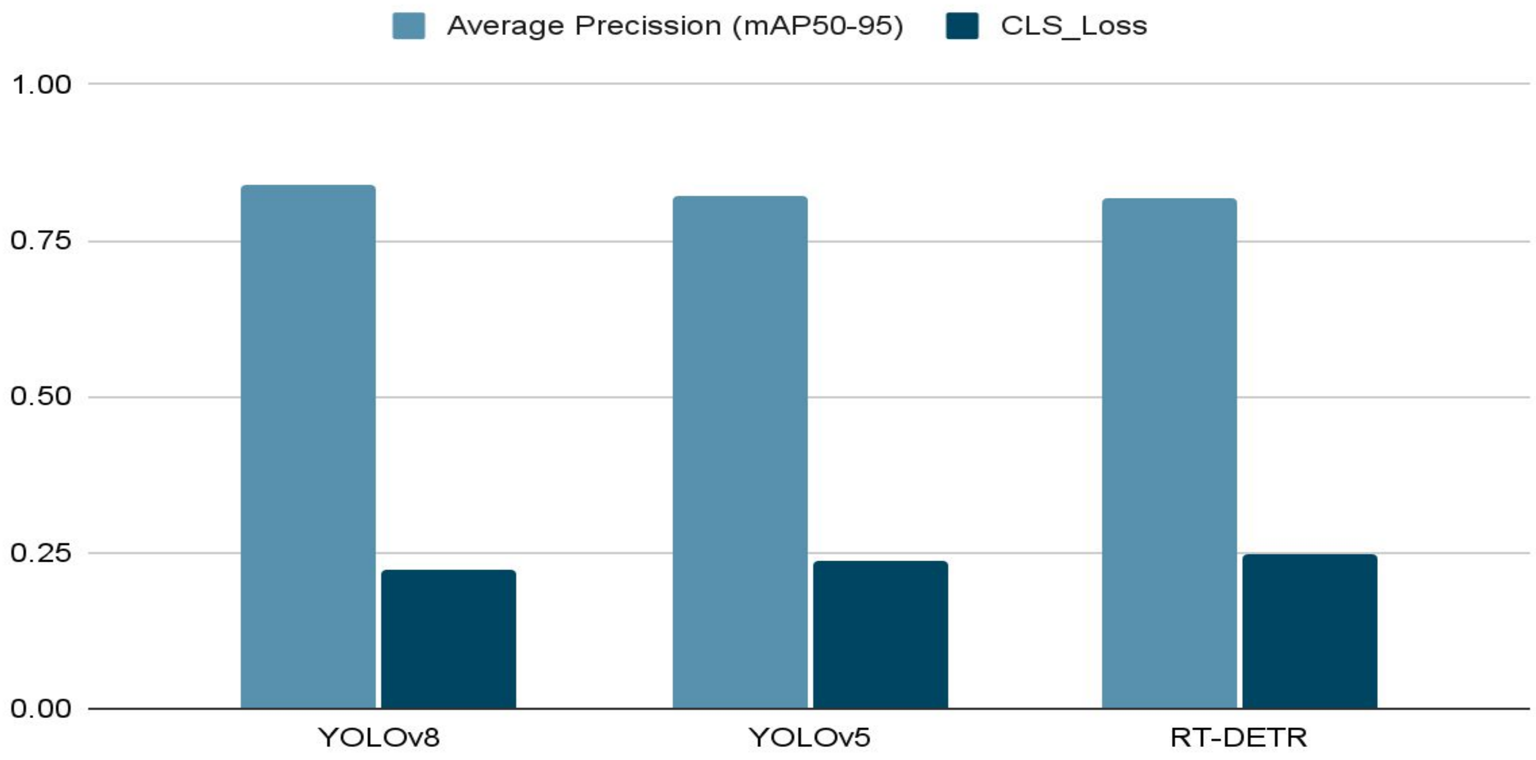


Figure 13:Comparison Between three models based on mAp50-95 and CLS_Loss

WORK DONE AND RESULTS

Table 6:Mean Average Precision(mAp50-95) of all the models based on different classes

Class	YOLOv8	YOLOv5	RT-DETR
Bardhaman	0.781	0.77	0.762
Dol	0.829	0.80	0.809
Gajadanta	0.849	0.835	0.835
Kaput	0.828	0.807	0.777
Karkat	0.827	0.816	0.804
Makar1	0.891	0.87	0.881
Makar2	0.92	0.906	0.896
Samput	0.822	0.81	0.803

WORK DONE AND RESULTS

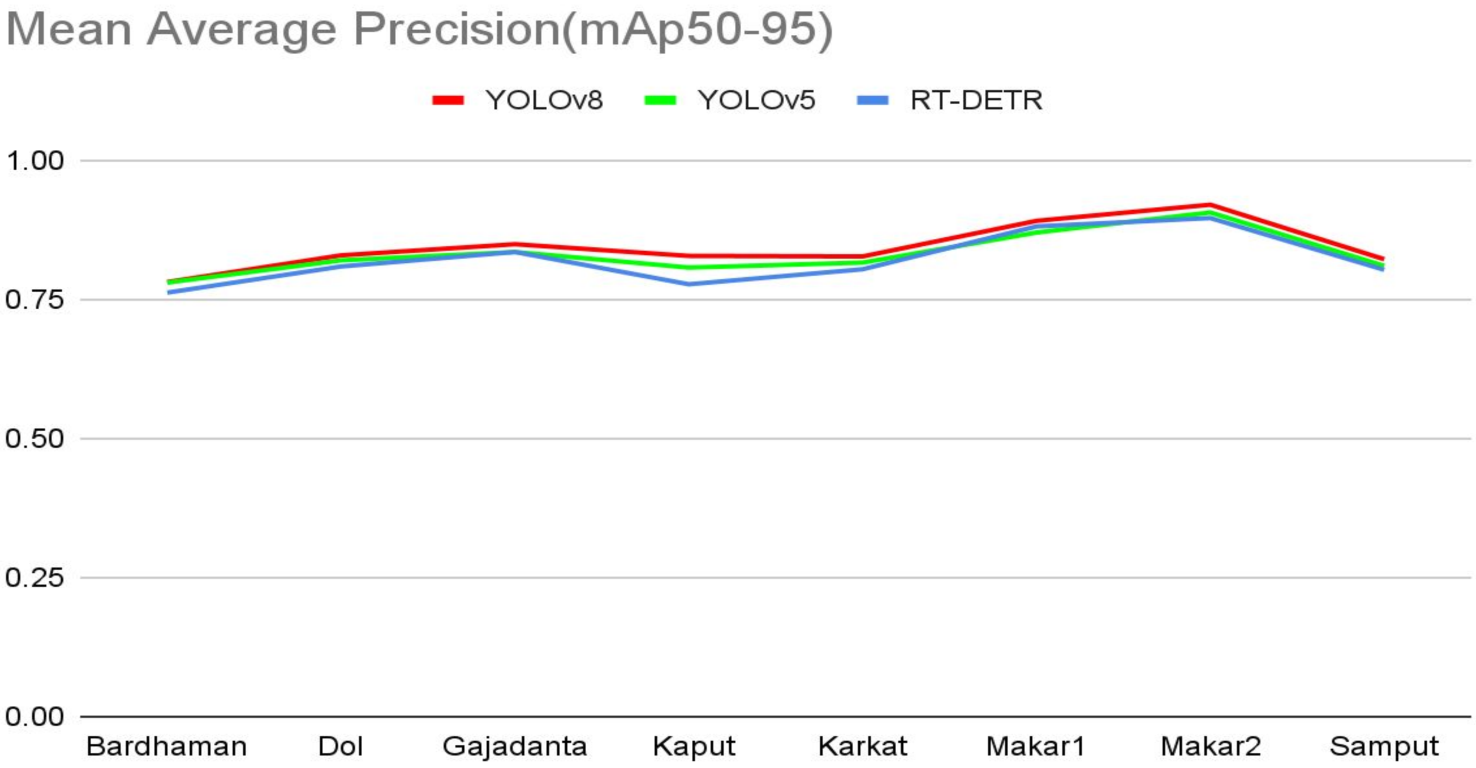


Figure 14:Mean Average Precision(mAp50-95) of all the models based on different classes

YOLOv8 Shines for Web Deployment:

A comprehensive evaluation of YOLOv8, YOLOv5, and RT-DETR models, focusing on precision and classification loss, identified YOLOv8 as the most suitable choice for deployment in our web application. YOLOv8's superior performance in these critical areas translates to efficient and accurate mudra recognition. To ensure a user-friendly experience, we developed a web application utilizing Flask, HTML, CSS, and JavaScript. This combination provides a seamless interface for users to interact with the YOLOv8 model and access mudra recognition results, fostering wider accessibility and appreciation for this unique aspect of Sattriya dance.

Making it Accessible: Web App Deployment

- Building a User-Friendly Interface: HTML, CSS, JavaScript, and Flask

- Integrated the trained YOLOv8 model with the UI using Flask, a Python web framework.

Flask Integration

- Flask App Creation: Created a Python script to initialize the Flask application.

```
from flask import Flask, render_template, request, redirect, send_file, url_for, Response

from ultralytics import YOLO



app = Flask(__name__)

@app.route("/")
def home():
    return render_template('index.html')


opdir='runs/detect'
@app.route("/", methods=["GET", "POST"])
```

Flask Integration

- Model Loading: Loaded the pre-trained YOLOv8 model within the Flask app.

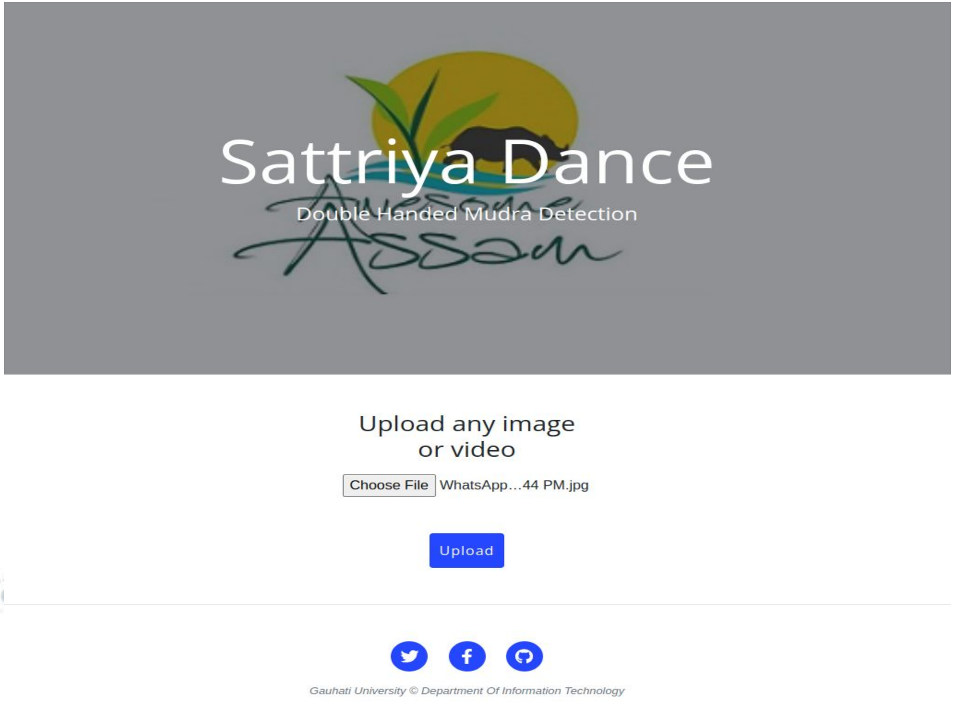
```
#initialize the YOLOv8 model here
model = YOLO('best.pt')
```

- Route Definition: Defined a route in Flask to handle user requests for mudra recognition.

```
def predict_img():
    if request.method == "POST":
        if 'file' in request.files:
            f = request.files['file']
            basepath = os.path.dirname(__file__)
            filepath = os.path.join(basepath, 'uploads', f.filename)
            print("upload folder is", filepath)
            f.save(filepath)
            global imgpath
            predict_img.imgpath = f.filename
            # print("printing predict_img :::::::", predict_img)
            print("printing predict_img :::::::", predict_img.imgpath)
```

Flask Integration

- Image Uploading: Designed the UI using HTML and JavaScript to allow users to upload images containing Sattriya dance mudras.



Flask Integration

- ❑ Image Processing: Within the Flask route, implemented functions to handle uploaded images:
- ❑ Convert image to a format suitable for the YOLOv8 model.

```
if file_extension == 'jpg':
    img = cv2.imread(filepath)
    frame = cv2.imencode('.jpg', cv2.UMat(img)) [1].tobytes()
    image = Image.open(io.BytesIO(frame))

    return display(f.filename)

elif file_extension == 'mp4':
    video_path = filepath #replace with your video path
    cap = cv2.VideoCapture(video_path)

    #get video dimensions
    frame_width = int(cap.get(cv2.CAP_PROP_FRAME_WIDTH))
    frame_height = int(cap.get(cv2.CAP_PROP_FRAME_HEIGHT))

    #Define the codec and create VideoWriter object
    fourcc = cv2.VideoWriter_fourcc(*'mp4v')
    out = cv2.VideoWriter('output.mp4', fourcc, 30.0, (frame_width, frame_height))
```

Flask Integration

- Run the YOLOv8 model on the image to detect mudras.

```
# perfome the detection
yolo = YOLO('best.pt')
# detections = yolo.predict(image, save=True)
results = yolo(filepath, conf=0.4, save=True, project=opdir)
print(results)
return display(f.filename)
```

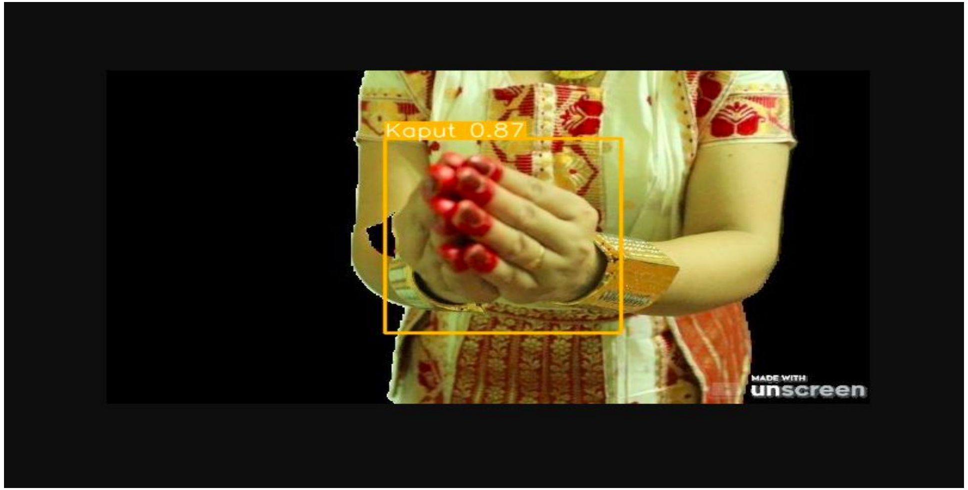
Flask Integration

- Result Display:Flask code to route the output from the model in a separate tab

```python
## The display function is used to serve the image or video from the folder_path directory.
@app.route('/<path:filename>')
def display(filename):
    folder_path = 'runs/detect'
    subfolders = [f for f in os.listdir(folder_path)if os.path.isdir(os.path.join(folder_path, f))]
    latest_subfolder = max(subfolders, key=lambda x: os.path.getctime(os.path.join(folder_path, x)))

    directory = folder_path+'/'+latest_subfolder
    print("printing directory:", directory)
    files = os.listdir(directory)
    latest_file = files[0]

    print(latest_file)

    filename = os.path.join(folder_path, latest_subfolder, latest_file)

    file_extension = filename.rsplit('.', 1)[1].lower()

    environ = request.environ
    if file_extension == 'jpg':
        return send_from_directory(directory, latest_file, environ) # shows the result in seperate tab

    else:
        return "Invalid file format"
```

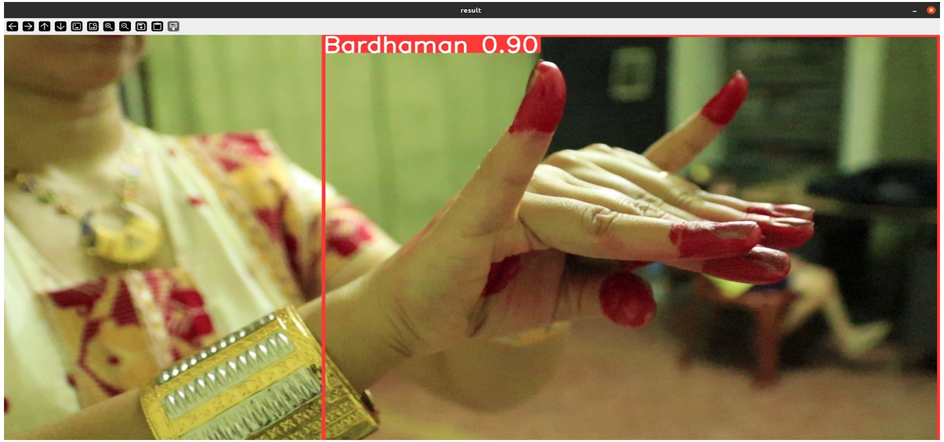
Outputs from the Web Application

- Image Detection made by the proposed model

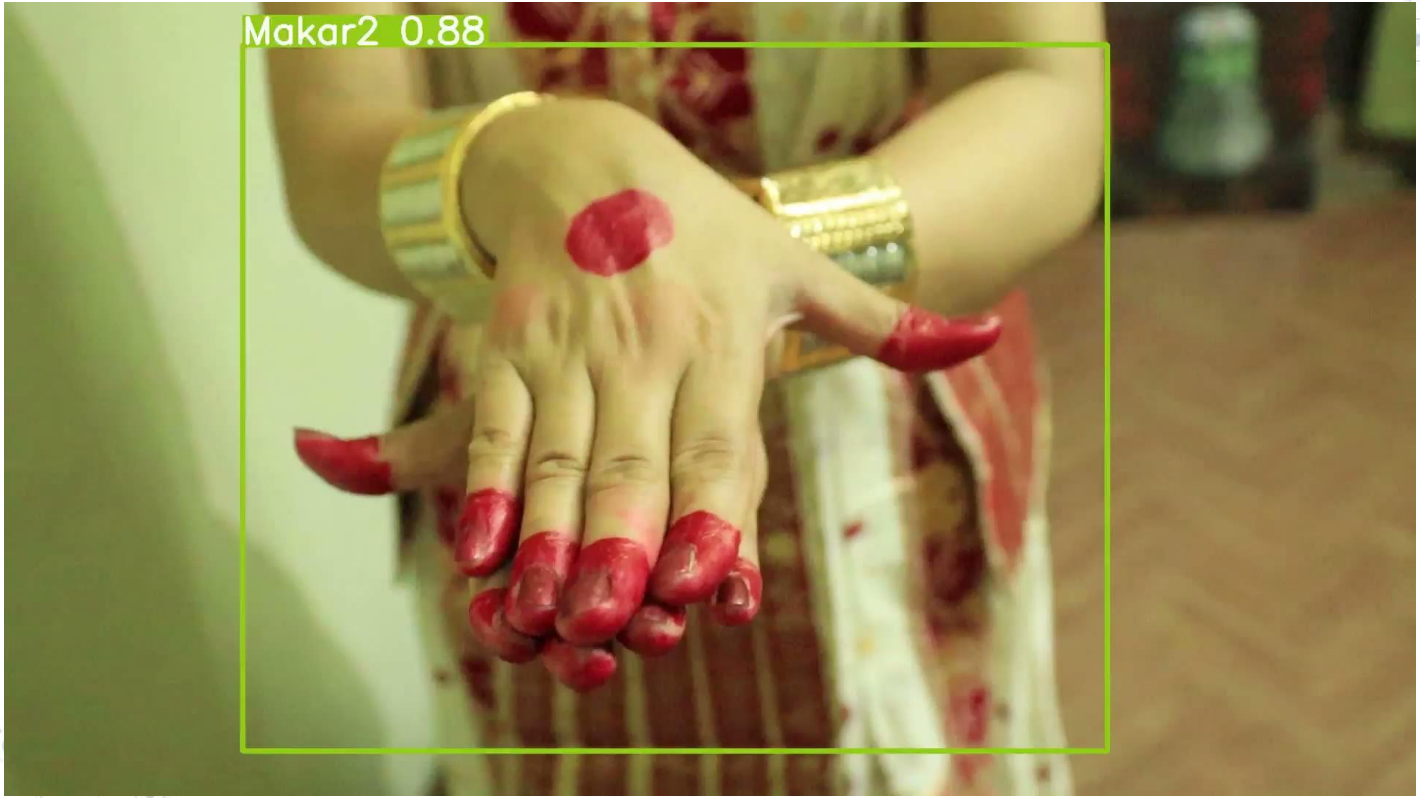


Outputs from the Web Application

- ❏ Video Detection made by the model



A Video Detection made by the YOLO model



CONCLUSION

- ❑ YOLO Model Success
- ❑ Diverse Recognition Approaches
- ❑ Intersection of cultural heritage and advanced technology

Bibliography

- [1] R. Pradeep, R. Rajeshwari, V. R. Ruchita, R. Bubna and H. R. Mamatha, "Recognition of Indian Classical Dance Hand Gestures," 2023 International Conference on Inventive Computation Technologies (ICICT), Lalitpur, Nepal, 2023, pp. 814-820, doi: 10.1109/ICICT57646.2023.10134484. keywords: {Training;Measurement;Humanities;Image recognition;Computational modeling;Neural networks;Object detection;Indian classical dance forms;hand gestures;mudra;samyukta hasta;asamyukta hasta;image classification;Convolutional Neural Network models;Inception v1;SqueezeNet;object detection;YOLO v5}

- [2] M. Safaldin, N. Zaghden and M. Mejdoub, "An Improved YOLOv8 to Detect Moving Objects," in IEEE Access, vol. 12, pp. 59782-59806, 2024, doi: 10.1109/ACCESS.2024.3393835. keywords: {YOLO;Feature extraction;Real-time systems;Detectors;Computer architecture;Object recognition;Task analysis;Deep learning;Deep learning;localization;object detection;segmentation;YOLO}

- [3] Diwan, Tausif, G. Anirudh, and Jitendra V. Tembhurne. "Object detection using YOLO: Challenges, architectural successors, datasets and applications." multimedia Tools and Applications 82.6 (2023): 9243-9275.

Bibliography

- [4] SANTHOSH, REMYA and KK, Rajkumar, Classification of Asmyukta Mudras in Indian Classical Dance Using Handcrafted and Pre-Trained Features with Machine Learning and Deep Learning Methods. Available at SSRN: https://ssrn.com/abstract=4818826 or http://dx.doi.org/10.2139/ssrn.4818826

- [5] M. B. Ullah, "CPU Based YOLO: A Real Time Object Detection Algorithm," 2020 IEEE Region 10 Symposium (TENSYMP), Dhaka, Bangladesh, 2020, pp. 552-555, doi: 10.1109/TENSYMP50017.2020.9230778. keywords: {Central Processing Unit;Computers;Object detection;Real-time systems;Graphics processing units;Task analysis;Computational modeling;object detection;real time;YOLO;CPU;deep learning}

- [6]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, India, 2016, pp. 193-196, doi: 10.1109/ICADW.2016.7942540. keywords: {Feature extraction;Support vector machines;Decision trees;Gesture recognition;Matrix converters;Indexes;Asamyukta hastas;hand gestures;Indian Classical Dance;Medial Axis Transformation}

Bibliography

- [7]H. Bhuyan, P. P. Das, J. K. Dash and J. Killi, "An Automated Method for Identification of Key frames in Bharatanatyam Dance Videos," in IEEE Access, vol. 9, pp. 72670-72680, 2021, doi: 10.1109/ACCESS.2021.3079397.
keywords: {Videos;Motion segmentation;Feature extraction;Support vector machines;Deep learning;Annotations;Image segmentation;Key frame;Adavu;three frame difference;bit-plane extraction;adaptive threshold;machine learning},

- [8]Joko S. Dance gesture recognition using space component and effort component of laban movement analysis. International Journal of Scientific & Technology Research. 2020;9(2):3389-94.

- [9]Kico I, Grammalidis N, Christidis Y, Liarokapis F. Digitization and visualization of folk dances in cultural heritage: A review. Inventions. 2018 Oct 23;3(4):72.

- [10]Bhushan S, Alshehri M, Keshta I, Chakraverti AK, Rajpurohit J, Abugabah A. An experimental analysis of various machine learning algorithms for hand gesture recognition. Electronics. 2022 Mar 21;11(6):968.

Bibliography

- [11]Naik AD, Supriya M. Classification of indian classical dance images using convolution neural network. In2020 International Conference on Communication and Signal Processing (ICCSP) 2020 Jul 28 (pp. 1245-1249). IEEE.

- [12]V . Janaranjani1, N. Megala2, R.Naveeth Kumar3 1,2,3Assistant Professor in Biomedical Engineering, Rathinam Technical Campus, Coimbatore Hand Gesture Recognition for Dance Movements. © March 2022| IJIRT | Volume 8 Issue 10 | ISSN: 2349-6002

THANK YOU