RNS INSTITUTE OF TECHNOLOGY

BENGALURU - 98

DEPARTMENT OF INFORMATION SCIENCE & ENGINEERING

Presentation on Internship

Hand Gesture Recognition System

Arnav Kashyap
USN: 1RN18IS025

New Age Solutions Technologies

Internal Guide
Dr. Suresh L
HOD of ISE, RNSIT

External Guide
Mr. Deepak Garg
CEO, NASTECH

AGENDA

- Abstract
- About the Company
- Introduction
- Literature Survey
- Requirements
- System Design
- Detailed Design
- Implementation
- Results
- Conclusion and Future Enhancements
- References
- Q & A

ABSTRACT

- Hand gesture recognition is of great importance for human computer interaction (HCI) because of its extensive applications in virtual reality and sign language recognition
- Human hand is very smaller with very complex articulations comparing with the entire human body and therefore errors can be easily affected.
- A Hand Gesture Recognition software reduces many efforts in the tasks which can be automated.
- Hand gesture recognition system received great attention in the recent few years because of its manifoldness applications and the ability to interact with machine efficiently through human computer interaction.
- In our framework, the hand region is extracted from the background with the background subtraction method. Then, the palm and fingers are segmented so as to detect and recognize the fingers.

ABOUT THE COMPANY

NASTECH – New Age Solutions & Technologies

- ***NASTECH*** is formed with the purpose of bridging the gap between Academia and Industry.
- Nastech is one of the leading Global Certification and Training service providers for technical and management programs for educational institutions.
- They collaborate with educational institutes to understand their requirements and form a strategy in consultation with all stakeholders to fulfill those by skilling , reskilling and upskilling the students and faculties on new age skills and technologies.
- Industry and project oriented student training programs.
- Certification programs mapped to Global Certification Exams from Microsoft/EC- Council/Google/AWS/ Adobe).

INTRODUCTION

- In Hand Gesture Recognition, the hand region is extracted from the background with the background subtraction method. Then, the palm and fingers are segmented so as to detect and recognize the fingers.
- The proposed system focuses on how to identify the different hand Gestures with the help of computer vision and deep learning algorithm by using the OpenCV, Tensor flow and Keras library
- The detection is carried out to see what is the meaning of particular hand gesture,with the help of Convolution Neural Networks(CNN) algorithm that the Sequential model uses.

LITERATURE SURVEY

- An Efficient Framework for Indian Sign Language Recognition Using Wavelet Transform :The proposed ISLR system is considered as a pattern recognition technique that has two important modules: feature extraction and classification. The joint use of Discrete Wavelet Transform (DWT) based feature extraction and nearest neighbour classifier is used to recognize the sign language. The experimental results show that the proposed hand gesture recognition system achieves maximum 99.23% classification accuracy while using cosine distance classifier.
- Hand Gesture Recognition System For Dumb People : Authors presented the static hand gesture recognition system using digital image processing. For hand gesture feature vector SIFT algorithm is used. The SIFT features have been computed at the edges which are invariant to scaling, rotation, addition of noise.
- An Automated System for Indian Sign Language Recognition in : In this paper a method for automatic recognition of signs on the basis of shape based features is presented.

LITERATURE SURVEY

- Hand Gesture Recognition for Sign Language Recognition: A Review in : Authors presented various method of hand gesture and sign language recognition proposed in the past by various researchers. For deaf and dumb people, Sign language is the only way of communication. With the help of sign language, these physical impaired people express their emotions and thoughts to other person.
- Design Issue and Proposed Implementation of Communication Aid for Deaf & Dumb People in : In this paper author proposed a system to aid communication of deaf and dumb people communication using Indian sign language (ISL) with normal people where hand gestures will be converted into appropriate text message. Main objective is to design an algorithm to convert dynamic gesture to text at real time.

REQUIREMENTS

❖ HARDWARE REQUIREMENTS

- Processor : Any Processor above 500 MHz
- RAM : 512Mb
- Hard Disk : 4 GB
- Input device : Standard Keyboard and Mouse
- Output device : VGA and High Resolution Monitor

❖ SOFTWARE REQUIREMENTS

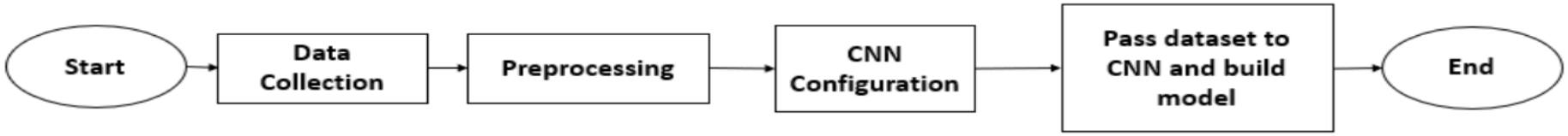
- Operating system : Windows 10
- IDE : Jupyter Notebook
- Tools/Technologies : Python, Computer Vision, OpenCv, Keras API, Numpy library

SYSTEM DESIGN

- This section provides the description of the dataset and CNN configuration that were used. The flowchart of methodology is shown on Figure 1. The approach is the combination of data collection, pre-processing, configuring the CNN and building the model.

- Images needed to train and validate the model were collected using a webcam. The gestures were performed by persons in front of the webcam. Then A minimal pre-processing was applied over the dataset to reduce the computational complication and achieve better efficiency.Then the images were converted to grayscale image. Since grayscale images contain only one color channel it will be easier for CNN to learn.

- The CNN that that has been considered in research to recognize hand gesture is composed of two convolution layers, two max pooling layers, two fully connected layers and output Sample Images from Hand Gesture Recognition Database layer.To implement the system, python was used as the programming language and a python IDE Spyder was used to write and run code. The library Keras was used for building the CNN classifier.



DETAILED DESIGN

THIS SECTION PROVIDES THE DESCRIPTION OF THE DATASET AND CNN CONFIGURATION THAT WERE USED. THE FLOWCHART OF METHODOLOGY IS SHOWN IN PREVIOUS FIGURE THE APPROACH IS THE COMBINATION OF DATA COLLECTION, PRE-PROCESSING, CONFIGURING THE CNN AND BUILDING THE MODEL.

1. **Input Data and Training Data**:

Images needed to train and validate the model were collected using a webcam. The gestures were performed by System Framework persons in front of the webcam. It is assumed that the input images exactly include one hand, gestures were made with right hand, the palm facing the camera and the hand were roughly vertical.

2. **Pre-Processing:**

A minimal pre-processing was applied over the dataset toreduce the computational complication and achieve better efficiency. Firstly, the background of the images was removed using the method of background subtraction proposed by Z. ZivKovic [9] [10]. The background subtraction is mainly based on K-gaussian distribution which selects appropriate gaussian distribution for each pixel and provides a better adaptability on varying scenes due to illumination changes. After subtracting background, only the image of hand remains.

3.**DATASET:**

We selected 10 static gestures (Index, Peace, Three, Palm Opened, Palm Closed, OK, Thumbs, Fist, Swing, Smile) to recognize. Each class has 800 images for training and 160 images for testing purpose. So total number of images is 8000 is provided.

DETAILED DESIGN

4. **CNN Configuration:**
The CNN that that has been considered in this research to recognize hand gesture is composed of two convolution layers, two max pooling layers, two fully connected layers and output
Fig. 5. Sample Images from Hand Gesture Recognition Database layer. There are three dropout performance in the network to prevent over-fitting.

The model is then compiled with Stochastic Gradient Descent (SGD) function with a learning rate 0.001. To evaluate loss, categorical cross-entropy function was used since the model is compiled for more than two classes. Finally, the metrics of loss and accuracy were specified to keep track on the evaluation process.

5. **System Implementation:**
To implement the system, python was used as the programming language and a python IDE Spyder was used to write and run code. The library Keras was used for building the CNN classifier. The library PIL was used for image preprocessing. Sklearn was used to calculate the confusion matrix. Matplotlib was used to visualize model accuracy and loss values and confusion matrix. NumPy was used for array operations. The training process on dataset is composed of two phases.

1) Training with Base Dataset: In this phase, the model was trained using the base dataset achieved after pre-processing.
2) Training with Expanded Dataset: In this phase, the dataset was augmented. Data augmentation is a technique to increase the number of data by applying zoom, shear, rotation, flip etc. This process not only increases the data but also brings variation in dataset which is essential for CNN to learn sophisticated differences of images.

IMPLEMENTATION

❖ TRAINING CODE

```
import cv2
import os
import time
import uuid
import tensorflow
```

```
IMAGES_PATH = 'Tensorflow/workspace/images/collectedImages'
```

```
labels = ['hello','thanks','yes','no','iloveyou']
number_imgs = 15
```

```
for label in labels:
    !mkdir {'Tensorflow\workspace\images\collectedImages\\'+label}
    cap = cv2.VideoCapture(0)
    print('Collecting Images for {}'.format(label))
    time.sleep(5)
    for imgnum in range(number_imgs):
        ret, frame = cap.read()
        imgname = os.path.join(IMAGES_PATH, label, label+'{}.jpg'.format(str(uuid.uuid1())))
        cv2.imwrite(imgname,frame)
        cv2.imshow('frame',frame)
        time.sleep(2)
        if cv2.waitKey(1) & 0xFF == ord('q'):
            break
    cap.release()
```

IMPLEMENTATION

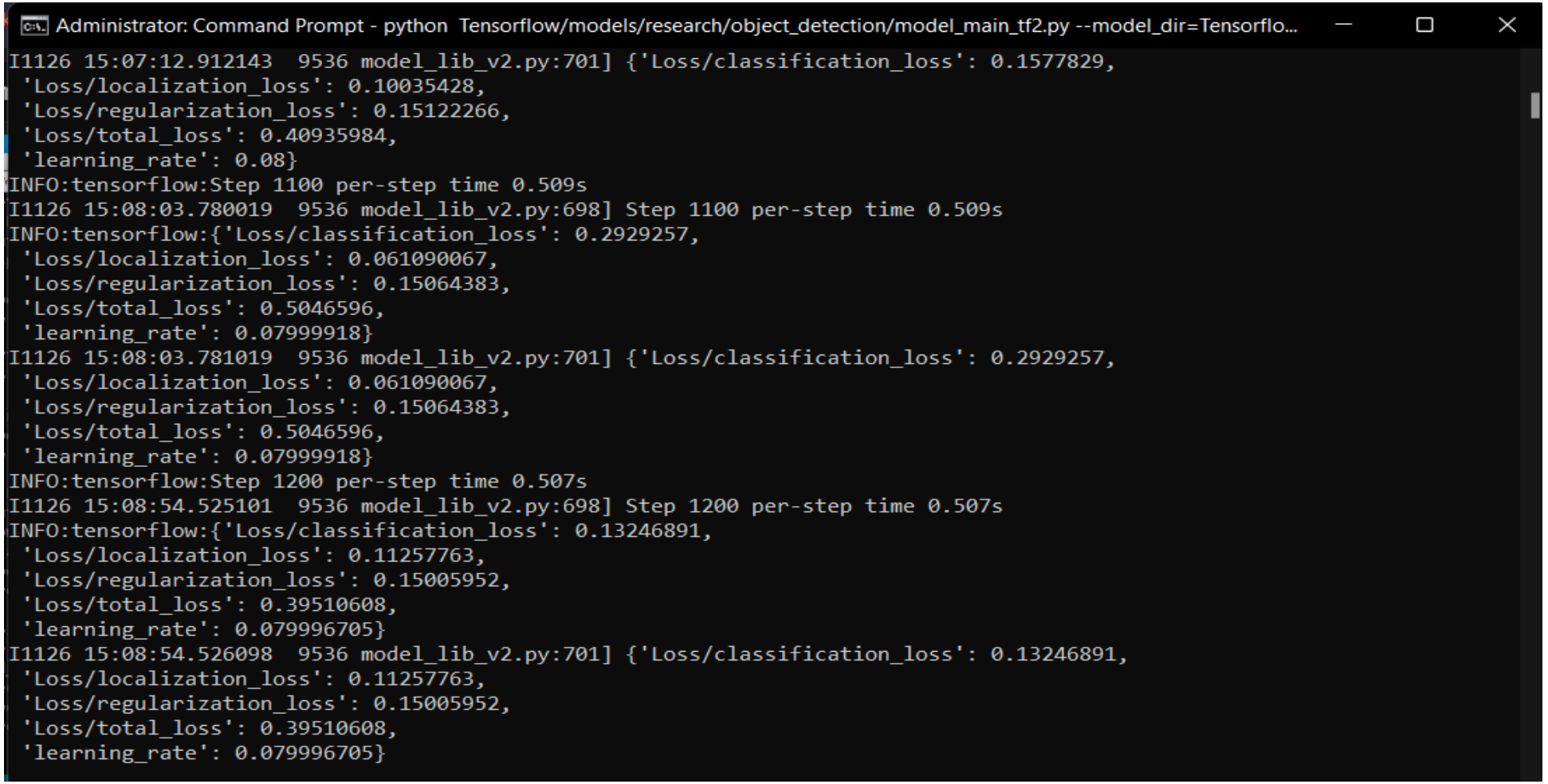
❖ TRAINING CODE

```
print("""python {}/research/object_detection/model_main_tf2.py --model_dir={}/{}
    --pipeline_config_path={}/{}/pipeline.config
    --num_train_steps=10000""".format(APIMODEL_PATH, MODEL_PATH,CUSTOM_MODEL_NAME,MODEL_PATH,CUSTOM_MODEL_NAME))
```

```
python Tensorflow/models/research/object_detection/model_main_tf2.py --model_dir=Tensorflow/workspace/models/my_ssd_mobnet --pi
peline_config_path=Tensorflow/workspace/models/my_ssd_mobnet/pipeline.config --num_train_steps=10000
python Tensorflow/models/research/object_detection/model_main_tf2.py --model_dir=Tensorflow/workspace/models/my_ssd_mobnet --pi
peline_config_path=Tensorflow/workspace/models/my_ssd_mobnet/pipeline.config --num_train_steps=10000
```

IMPLEMENTATION

❖ TRAINING CODE



IMPLEMENTATION

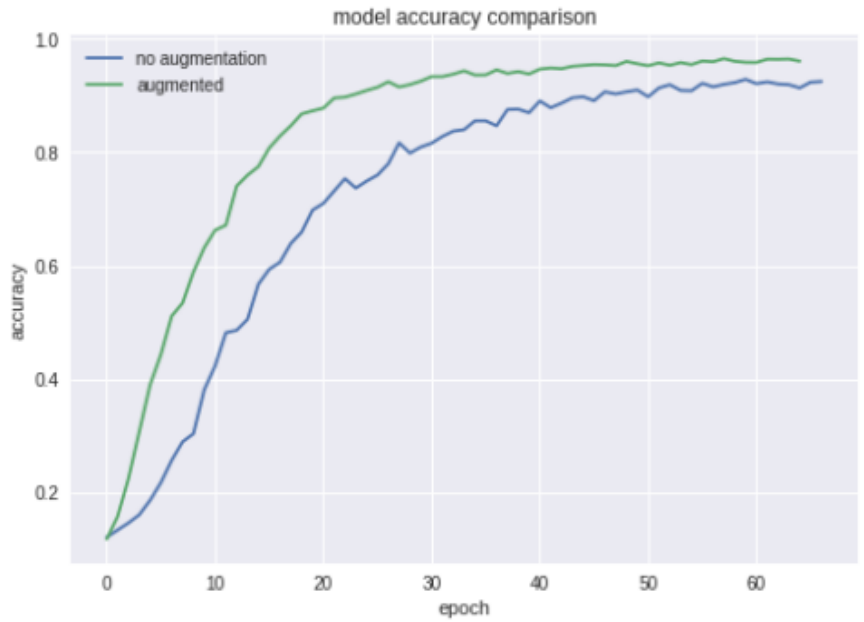
❖ TESTING AND LIVE DETECTION

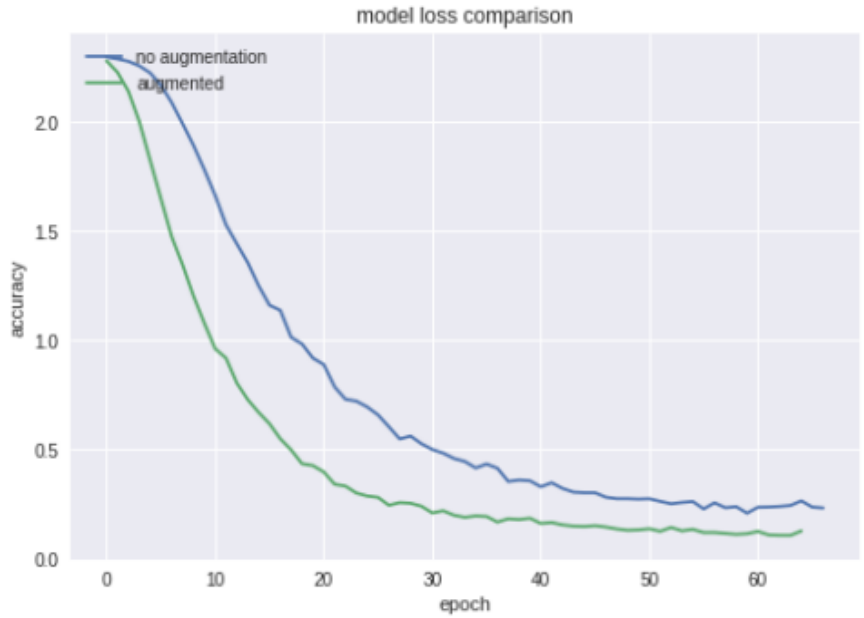
```
# Setup capture
cap = cv2.VideoCapture(0)
width = int(cap.get(cv2.CAP_PROP_FRAME_WIDTH))
height = int(cap.get(cv2.CAP_PROP_FRAME_HEIGHT))
```

```
while True:
    ret, frame = cap.read()
    image_np = np.array(frame)

    input_tensor = tf.convert_to_tensor(np.expand_dims(image_np, 0), dtype=tf.float32)
    detections = detect_fn(input_tensor)

    num_detections = int(detections.pop('num_detections'))
    detections = {key: value[0, :num_detections].numpy()
                  for key, value in detections.items()}
    detections['num_detections'] = num_detections

    # detection_classes should be ints.
    detections['detection_classes'] = detections['detection_classes'].astype(np.int64)

    label_id_offset = 1
    image_np_with_detections = image_np.copy()

    viz_utils.visualize_boxes_and_labels_on_image_array(
                image_np_with_detections,
                detections['detection_boxes'],
                detections['detection_classes']+label_id_offset,
                detections['detection_scores'],
                category_index,
                use_normalized_coordinates=True,
                max_boxes_to_draw=5,
                min_score_thresh=.5,
                agnostic_mode=False)

    cv2.imshow('object detection',  cv2.resize(image_np_with_detections, (800, 600)))

    if cv2.waitKey(1) & 0xFF == ord('q'):
        cap.release()
        break
```

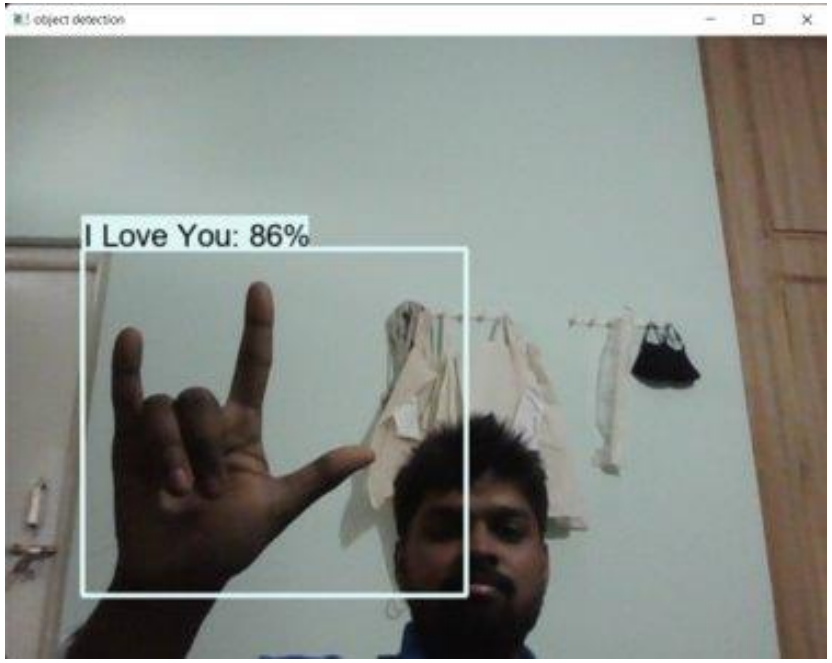
RESULTS

- This section describes the results obtained form the experiment using the CNN configuration according to The experimental result shows that the model which was augmented with temporary data achieved 97.12% accuracy which Effects of Data Augmentation is about 4% higher than the model without any augmented data.

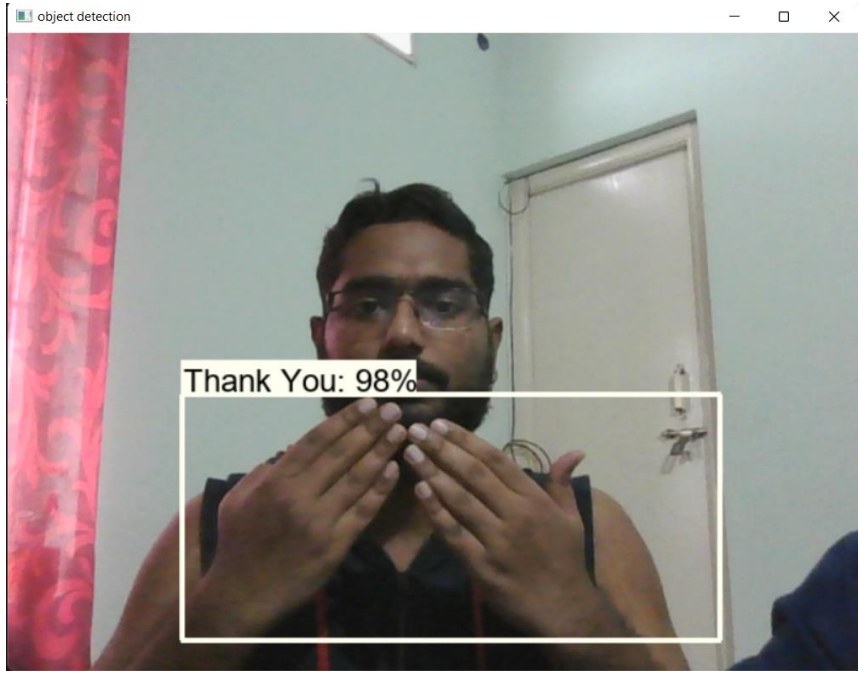


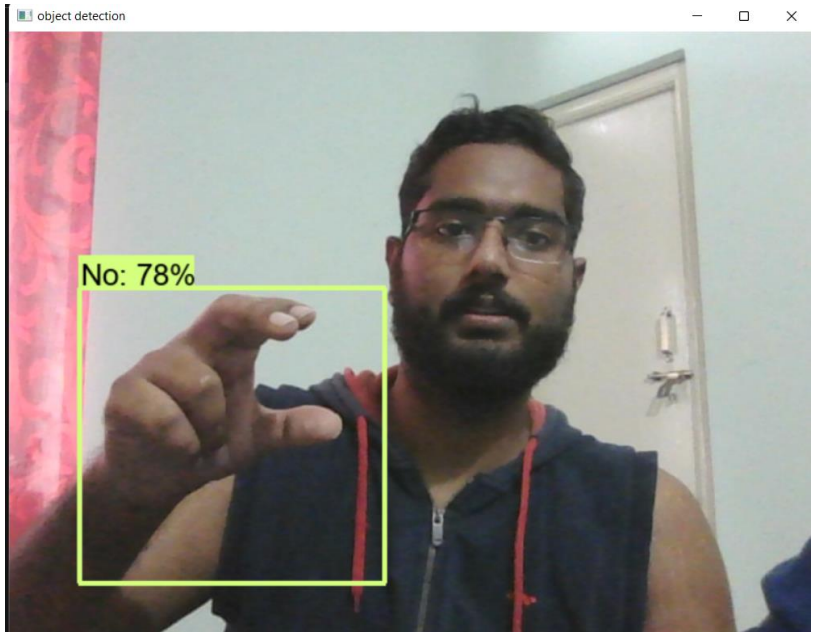


RESULTS



RESULTS





CONCLUSIONS

- This research explores the opportunity and challenges in recognition of hand gestures. It also analyzes the effect of data augmentation in deep learning. We can say after this research that CNN is a data driven methodology and data augmentation has a huge effect in deep learning.
- The user can interact with the virtual environment by hand gestures.
- Although the system can recognize gestures successfully, some extension is still possible. For example, by applying knowledge driven methodology such as Belief Rule Base (BRB), which is widely used where uncertainty becomes an issue . Hence, the recognition of gesture can be accomplished more accurately.

LIMITATIONS

- A major challenge in gesture recognition is the proper segmentation of skin coloured objects (e.g. hands, face) against a complex static background. The accuracy of skin segmentation algorithms is limited because of objects in the background that are similar in colour to human skin.
- It could also be observed that the face recognition stage is not robust when the detected Gesture is at a certain angle of inclination.
- However, this is not a problem of great impact, as this application is oriented to access control and at this point, the person must maintain a firm and straight posture in front of the device that acquires the image.

FUTURE ENHANCEMENTS

- Gesture recognition algorithm is relatively robust and accurate.
- Convolution can be slow, so there is tradeoff between speed and accuracy.
- In the future, we will investigate other methods of extracting feature vectors, without performing expensive convolution operations.
- More gestures can be added to list of recognition. It was assumed that the background should be less complex. Therefore, recognition of gestures in complex background can be another extension. Recognition of gestures made with both hands is not possible by this system. Therefore, another future work can be the recognition of gestures made with both hands.

REFERENCES

[1] A. D. Wilson and A. F. Bobick, "Learning visual behavior for gesture analysis," in Proceedings of International Symposium on Computer Vision-ISCV. IEEE, 1995, pp. 229–234.

[2] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," nature, vol. 521, no. 7553, p. 436, 2015.

[3] Y. LeCun, P. Haffner, L. Bottou, and Y. Bengio, "Object recognition with gradient-based learning," in Shape, contour and grouping in computer vision. Springer, 1999, pp. 319–345.

[4] E. Stergiopoulou and N. Papamarkos, "Hand gesture recognition using a neural network shape fitting technique," Engineering Applications of Artificial Intelligence, vol. 22, no. 8, pp. 1141–1158, 2009.

[5] T.-N. Nguyen, H.-H. Huynh, and J. Meunier, "Static hand gesture recognition using artificial neural network," Journal of Image and Graphics, vol. 1, no. 1, pp. 34–38, 2013.

THANK YOU