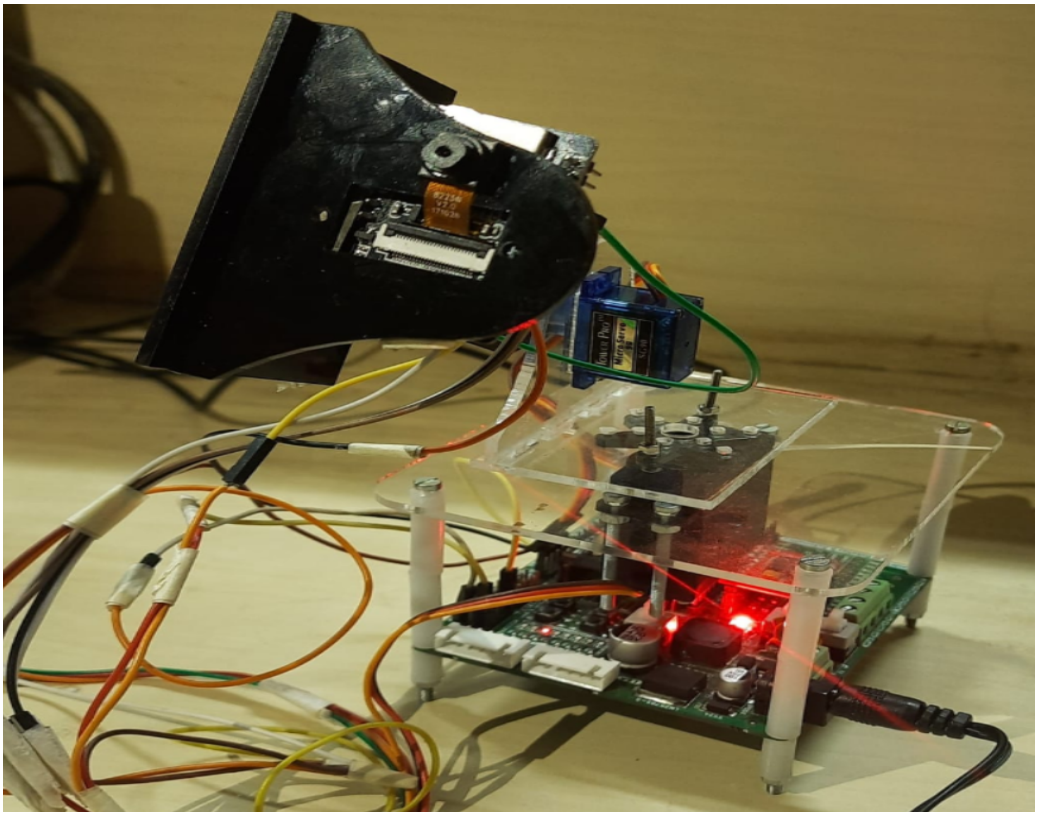


Project Report for Eklavya ‘22 at SRA VJTI

Smart Camera Stand

Yatharth Dedhia | Medha Sati

Acknowledgement

We would like to express our gratitude towards our very helpful and insightful seniors i.e the present members of Society of Robotics and Automation (SRA), VJTI for their kind cooperation, helpful resources and timely encouragement along with the proper guidance and help which helped us greatly in completion of this project. Although it seemed almost impossible, putting together our collective effort to reach this milestone and looking forward to many more to come is all thanks to you guys ! We would like to express our special gratitude and thanks to SRA for giving us this opportunity.

Special thanks to our amazing mentors:

- Sarrah Bastawala
- Marck Koothoor
- Aniruddha Thakare

Team:

Yatharth Dedhia - dedhiayatharth2004@gmail.com

Medha Sati - medhasati2002@gmail.com

Table of contents :

Project Overview

BRIEF IDEA

This project aims to make a self-sufficient phone stand with its own camera and microcontroller that should be able to process a lightweight object classification/ object detection model in TinyML that can track a person and rotate the stand in 2-axes to follow the person primarily in frame.

DESCRIPTION OF USE CASE AND PROJECT

Developing a general understanding of Convolutional Neural Networks and understanding their working. Running the model on ESP32 and using a servo motor to rotate the camera stand.

PROJECT DOMAIN

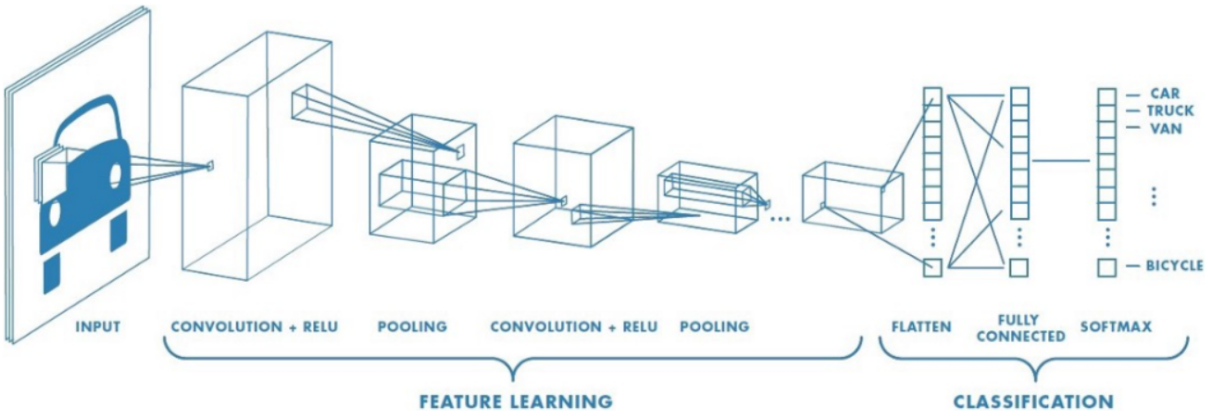
- Embedded Systems
- Convolutional Neural Network
- Machine Learning
- TinyML

Introduction

- CONVOLUTIONAL NEURAL NETWORK

A Convolutional Neural Network (ConvNet/CNN) is a Deep Learning algorithm which can take in an input image, assign importance (**learnable weights and biases**) to various aspects/objects in the image and be able to differentiate one from the other. The pre-processing required in a ConvNet is much lower as compared to other classification algorithms. While in primitive methods filters are hand-engineered, with enough training, ConvNets have the ability to learn these filters/characteristics.

The architecture of a ConvNet is analogous to that of the connectivity pattern of Neurons in the Human Brain and was inspired by the organization of the Visual Cortex. Individual neurons respond to stimuli only in a restricted region of the visual field known as the Receptive Field. A collection of such fields overlap to cover the entire visual area.



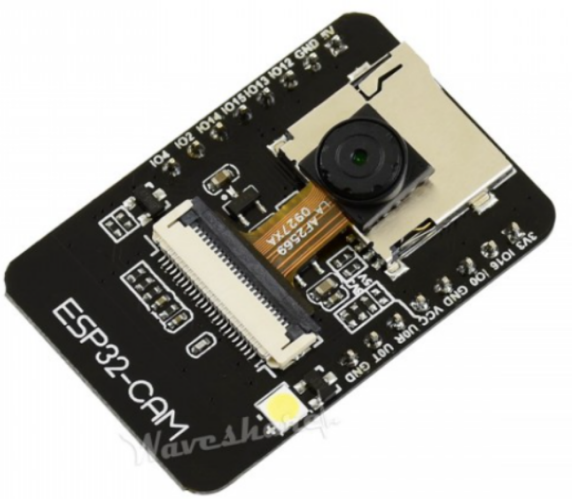
- TRANSFER LEARNING

Transfer learning is a machine learning method where a model developed for a task is reused as the starting point for a model on a second task. It is a popular approach in deep learning where pre-trained models are used as the starting point on computer vision and natural language processing tasks given the vast compute and time resources required to develop neural network models on these problems and from the huge jumps in skill that they provide on related problems.

- ESP32-CAM

The ESP32-CAM is a small size, **low power consumption camera module based on ESP32**. It comes with an OV2640 camera and provides an onboard TF card slot.

The ESP32-CAM can be widely used in intelligent IoT applications such as wireless video monitoring, WiFi image upload, QR identification, and so on.



- SERVO MOTOR

A servo motor is a type of motor that can rotate with great precision. Normally this type of motor consists of a control circuit that provides feedback on the current position of the motor shaft, this feedback allows the servo motors to rotate with great precision. If you want to rotate an object at some specific angles or distance, then you use a servo motor.

- TINY ML

Tiny machine learning is broadly defined as a fast growing field of machine learning technologies and applications including hardware, algorithms and software capable of performing on-device sensor data analytics at extremely low power, typically in the mW range and below, and hence enabling a variety of always-on use-cases and targeting battery operated devices.

- MODEL COMPRESSION
  - QUANTIZATION

Quantization is the process of **reducing the precision of the weights, biases, and activations** such that they consume less memory.

In other words, the process of quantization is the process of taking a neural network, which generally uses 32-bit floats to represent parameters, and instead converts it to use a smaller representation, like 8-bit integers.

  - PRUNING

Pruning is the process of **deleting parameters** from an existing neural network, which might involve removing individual parameters or groups of parameters, such as neurons. This procedure aims to keep the network's accuracy while enhancing its efficiency. This can be done to reduce the size or to cut down on the amount of computing power necessary to run the neural network.

Workflow

Our first aim was to learn about neural networks, convolutional neural net and transfer learning. After completing our learning phase, we started preparing our model.

- TRANSFER LEARNING

We used a cifar100 dataset. This dataset contains 32x32 images which are divided into 100 classes with 600 images per class. Each class is under a superclass which is a superset of 5 classes. For our use-case, we only need to train the model for the presence of a person, we only need 2 classes for binary classification. Hence, all the images from the superclass 'person' are labeled as 1, hence it contains 3000 images. Initially, we labeled all the remaining 57000 images as 0 or 'not-person', which caused the model to overfit and give false accuracy of 95% as it was trained more towards 'not-people' class.

Here is the list of the 100 classes in the CIFAR-100:

Classes:
1-5) beaver, dolphin, otter, seal, whale
6-10) aquarium fish, flatfish, ray, shark, trout
11-15) orchids, poppies, roses, sunflowers, tulips
16-20) bottles, bowls, cans, cups, plates
21-25) apples, mushrooms, oranges, pears, sweet peppers
26-30) clock, computer keyboard, lamp, telephone, television
31-35) bed, chair, couch, table, wardrobe
36-40) bee, beetle, butterfly, caterpillar, cockroach
41-45) bear, leopard, lion, tiger, wolf
46-50) bridge, castle, house, road, skyscraper
51-55) cloud, forest, mountain, plain, sea
56-60) camel, cattle, chimpanzee, elephant, kangaroo
61-65) fox, porcupine, possum, raccoon, skunk
66-70) crab, lobster, snail, spider, worm
71-75) baby, boy, girl, man, woman
76-80) crocodile, dinosaur, lizard, snake, turtle
81-85) hamster, mouse, rabbit, shrew, squirrel
86-90) maple, oak, palm, pine, willow
91-95) bicycle, bus, motorcycle, pickup truck, train
96-100) lawn-mower, rocket, streetcar, tank, tractor

We rectified this by taking an equal number of images for both 1 and 0 class. This is done by taking 31 images from each class, hence we got 2945 images for the 'not-people' class. These images were then split into 80-20% for training and testing.

After selecting and making a suitable dataset, the next task was to select a model. We will use a pre-trained model for our work. This is called Transfer Learning. There were a few options that we were considering from the start. We wanted a model with less number of parameters but high accuracy. Few networks that we were considering were:

1. ResNet 50
2. MobileNetv2

3. Deep-Learning Models (EfficientNet Lite & MobileNetV2)

We read a few articles to compare the working of some of these networks after being trained.

Model	Model Size (MB)	Inference time (sec)	Testing Accuracy
MobileNetV2	8.54	0.035	0.888
EfficientNet Lite-0	12.58	0.042	0.924
EfficientNet Lite-1	15.8	0.064	0.913
EfficientNet Lite-2	18.37	0.085	0.907
EfficientNet Lite-3	26.38	0.128	0.907
EfficientNet Lite-4	44.69	0.221	0.891

After analyzing the performance and model size of various models, we decided to use EfficientNetLite0.

Even though the accuracy of the **EfficientNetLite0** model was better than MobileNetV2, the size of the model was our primary concern. After using EfficientNetLite0, our model size was around 41Mb even after quantization. We could not prune our model as pruning is not supported in the Tensorflow Lite API.

We tried looking for some alternatives but in the end, decided to switch to **MobileNetV2** as the model size is smaller than EfficientNetLite0 and we can prune our MobileNetV2 model. We also pruned and quantized our model to reduce the size. We tried different methods and compared the performance and size of our model.

EfficientNetLite0 Inferences on Image Classification

prune	quantization	accuracy (CIFAR100)	target_sparsity	model size	c array size
		88.16	0.9	2.7	16.7
	Full int			2.7	16.7
	dynamic			2.6	16.3
	float16			4.4	27.6
const sparsity	int only			2.9	17.9
		86	0.7		
	Dynamic			2.6	16.3
	full int			2.7	16.7
	float16			4.4	27.5
	int only	87.47	0.5,0.8	2.9	17.9
		87.64	0.5-0.9	2.6	16.3
Poly Decay		85.2	0.5-0.955	2.6	16.3
	dynamic	86	0.4-0.9	2.6	16.3

For pruning, we used the prune_low_magnitude function. This function wraps a tf.keras model or layer with pruning functionality which sparsifies the layer's weights during training. For quantization, we used the Dynamic range quantization method. This type of quantization, statically quantizes only the weights from floating point to integer at conversion time, which provides 8-bits of precision. After pruning and quantizing, our MobileNetV2 model the model size was reduced to around 16Mb.

- ESP DL

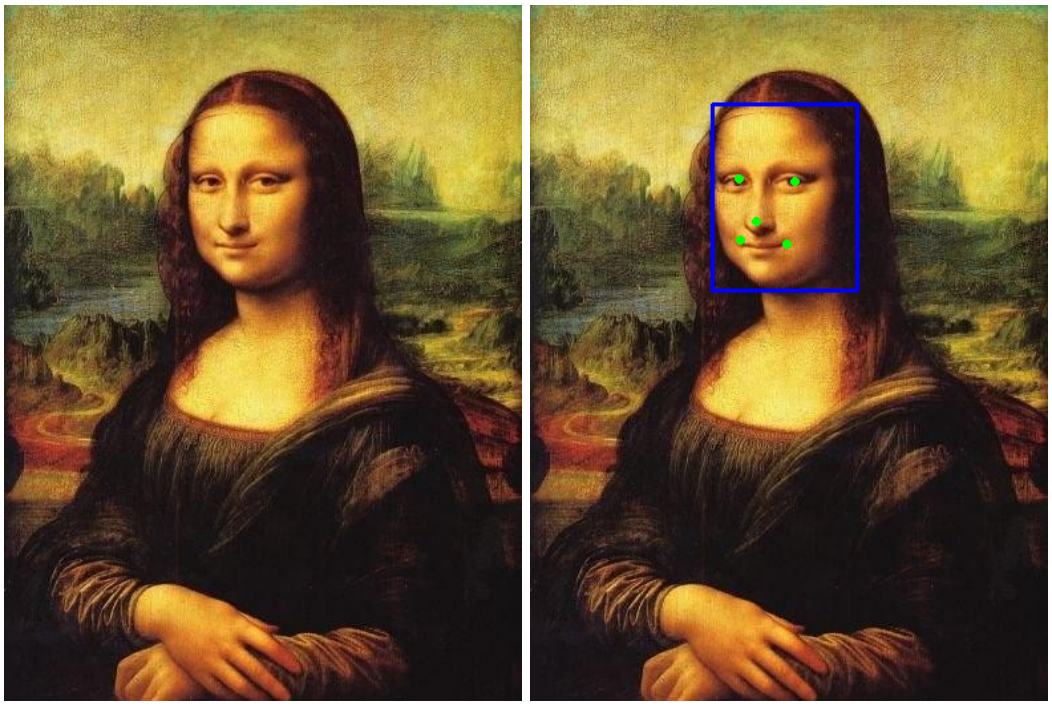
ESP-DL is a library for high-performance deep learning resources dedicated to ESP32, ESP32-S2, ESP32-S3 and ESP32-C3.

ESP-DL provides APIs for Neural Network (NN) Inference, Image Processing, Math Operations and some Deep Learning Models. With ESP-DL, you can use Espressif's SoCs for AI applications easily and fast. As ESP-DL does not need any peripherals, it can be used as a component of some projects.

We used it as a component of ESP-WHO, which contains several project-level examples of image application. ESP-DL also provides a quantization toolkit that helps in deploying the quantized inference on ESP SoCs with models using ESP-DL. This toolkit runs based on Open Neural Network Exchange (ONNX), an open source format for AI models.

ESP-DL Object Detection Inference

Bounding Box with facial features



- ESP-WHO

ESP-WHO is an image processing development platform based on Espressif chips. It contains development examples that may be applied in practical applications. ESP-WHO provides examples such as Human Face Detection, Human Face Recognition, Cat Face Detection, Gesture Recognition, etc. You can develop a variety of practical applications based on these examples. ESP-WHO runs on

ESP-IDF. ESP-DL provides rich deep learning related interfaces for ESP-WHO, which can be implemented with various peripherals to realize many interesting applications. We used the Human Face Detection model for our project.

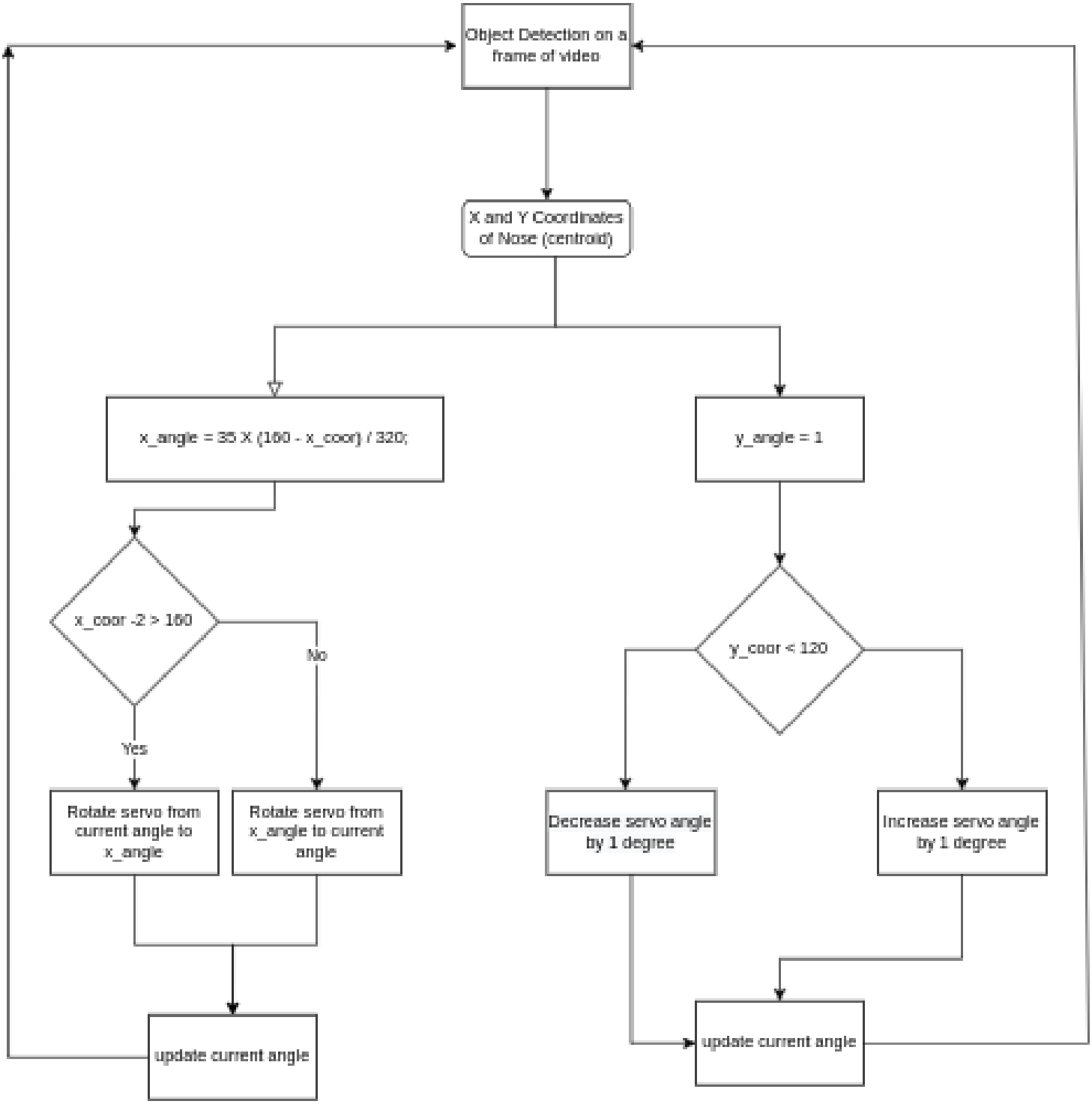
- ESP-CAM and SERVO

We used an ESP32-CAM for our project. The camera is used to capture frames and use it as an input for the Object Detection Model. The detected and face features coordinated are used as the input for the servo.

The Nose of the face is used as the centroid to get the x and y coordinates for calculating the angles for servo rotation.

We have mapped the coordinates to the angles of the servo.

Person Tracking Algorithm:



Project Conclusion & Future Work

- **WHAT WE HAVE ACHIEVED SO FAR**

- We are currently using the ESP WHO Human Face Detection model. The model returns 5 coordinate points. The coordinates of the edge of both the eyes and the nose.
- We are using the nose coordinates to move the servo motor.
- We are using two servo motors. One for horizontal motion and one for vertical motion.
- We also have a MobileNetv2 model that we trained using transfer learning. The model is 88% accurate. It can be used on single board computers like Raspberry pi but was not suitable for ESP32 due to the size.

• FUTURE PROSPECTS

1. Stream video feed from ESP32-CAM to a website using the onboard wifi module.
2. Implement other Object Detection models and compare results.
3. Explore other Object Tracking techniques.
4. Improve the design to accommodate a Smart Phone.

References

- GitHub Repo
- Courses:

1. 3b1b NN playlist

2. Neural Networks and Deep Learning by DeepLearning.AI (Andrew NG)

3. Convolutional Neural Networks by DeepLearning.ai (Andrew NG):

- Dataset
- ESP-DL
- ESP-WHO

EKLAVYA MENTORSHIP PROGRAMME

At

SOCIETY OF ROBOTICS AND AUTOMATION,

VEERMATA JIJABAI TECHNOLOGICAL INSTITUTE

MUMBAI

OCTOBER 2022