Technologies Used to Build a Vision Application for Weed Classification

- In this presentation, I will discuss the technologies I used to build a vision application that can Identify the type of weed growing in the farmlands of the Karnataka Region.

- The dataset used in the project contains 3 different classes of weeds, and I trained a deep learning model to recognize the weeds.

- The vision application uses the OpenCV module to implement live webcam inferencing with different background noise.

- Through this presentation, I will walk you through the various technologies I used in the project, including the deep learning model, data preparation techniques, and OpenCV for real-time inferencing.

Data Preparation

- The dataset used in the project consists of images of weeds growing in the farmlands of the Karnataka Region.
- The dataset was provided for the project and contains 3 different classes of weeds: "CELOSIA ARGENTEA L", "CROWFOOT GRASS", and "PURPLE CHLORIS".
- I used the ImageDataGenerator function from the TensorFlow library to prepare the data for model training.
- The data was rescaled, sheared, zoomed, and flipped horizontally to increase the dataset size and improve the model's ability to generalize.
- I also used a validation split of 20% to create separate training and validation sets.

Deep Learning Model

- For the project, I used the MobileNetV2 architecture for the deep learning model.
- MobileNetV2 is a lightweight and efficient neural network architecture designed for mobile and embedded vision applications.
- I used pre-trained weights from the “mobilenet_v2_weights_tf_dim_ordering_tf_kernels_1.0_224_no_top.h5" file, which is a pre-trained model without the classification head.
- I added a GlobalAveragePooling2D layer, a Dense layer with 128 units and ReLU activation, a Dropout layer with a rate of 0.5, and a Dense output layer with a softmax activation function.
- The model was compiled with the Adam optimizer, categorical cross-entropy loss function, and accuracy metric.
- The model was trained using the training and validation sets and saved as "weed_classification_karnataka_model.h5".

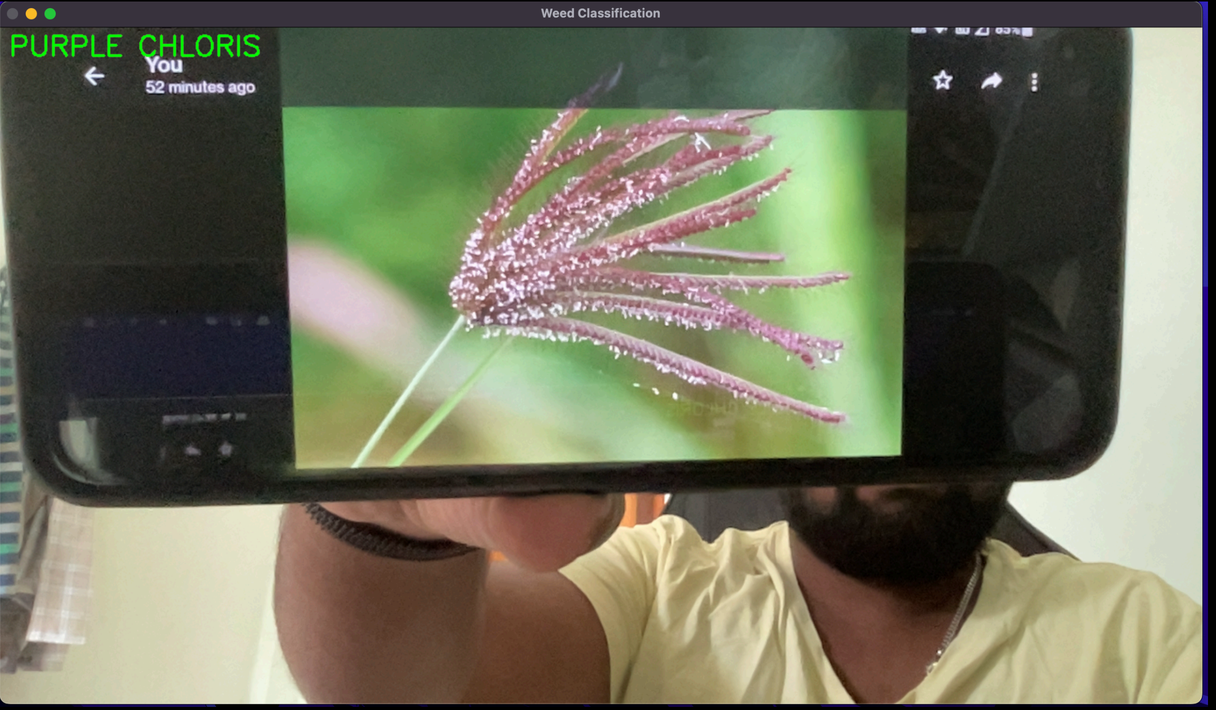
Model Training and Validation

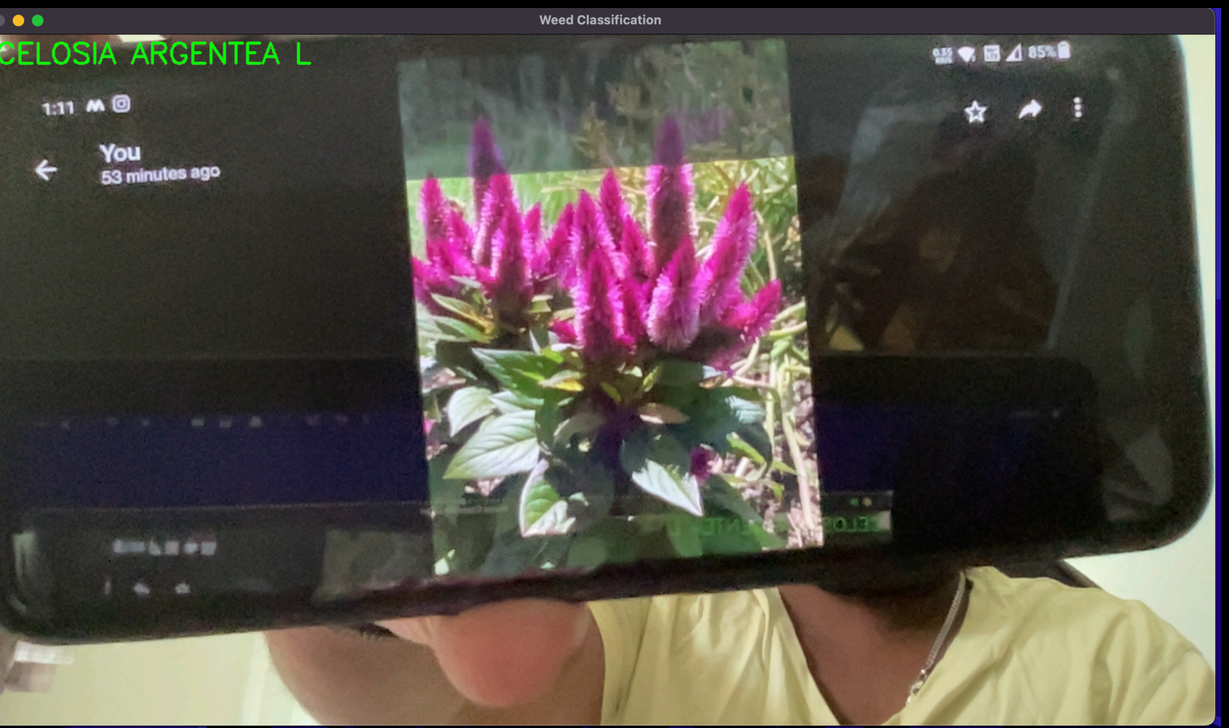
- The deep learning model was trained using the ImageDataGenerator function from TensorFlow, which preprocessed and augmented the dataset.
- The training process used a validation split of 20% to create separate training and validation sets.
- The MobileNetV2 architecture was used for the model with pre-trained weights from the "mobilenet_v2_weights_tf_dim_ordering_tf_kernels_1.0_224_no_top.h5" file.
- The model was compiled with the Adam optimizer, categorical cross-entropy loss function, and accuracy metric.
- The model was trained for 10 epochs using the training set and validation set.
- The model achieved a high level of accuracy on the validation set, demonstrating its ability to generalize well to new data.

Real-Time Inference with OpenCV

- For real-time inference with the model, I used the OpenCV library.
- I set up a live webcam feed using the cv2.VideoCapture() function from OpenCV.
- The model was loaded from the "weed_classification_karnataka_model.h5" file using the tf.keras.models.load_model() function from TensorFlow.
- The class labels were mapped to class indices using a dictionary.
- The webcam feed was pre-processed by resizing the frame to (224, 224) and expanding the dimensions to (1, 224, 224, 3) using the numpy library.
- The model made predictions on the input frame and returned a predicted class.
- The predicted class was displayed on the frame using the cv2.putText() function from OpenCV.
- The result was displayed in real-time on the screen until the user exits the program.

Results

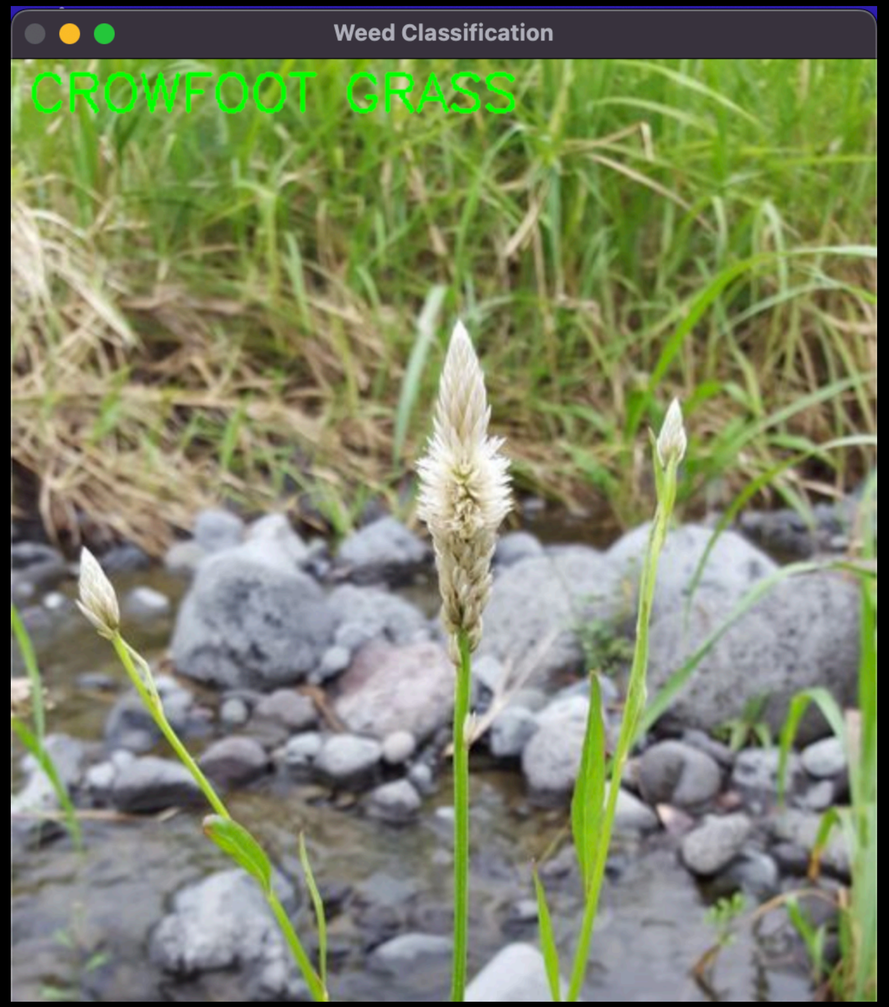

Weed Classification
PURPLE CHLORIS
You
52 minutes ago


Weed Classification
CELOSIA ARGENTEA L
You
53 minutes ago


Weed Classification
CROWFOOT GRASS

Test Image Input

- In addition to the live webcam feed, a single test image was used to verify the model's predictions.
- The test image was manually labeled and resized to 224x224 pixels to match the input size of the model.
- The model was able to make accurate predictions on the test image, demonstrating its ability to generalize to new data.
- This test image can be useful for validating the model's performance and for testing the model in situations where a live webcam feed is not available.
- Overall, this extra test image input adds a valuable component to the project and showcases the model's versatility.



Thank You

Name: Sai Pranay

Reg.no: 20bci7061