**Crop and Weed Detection System: Reducing Pesticide Waste and Enhancing Crop Production**

Aryan Shrivastav

National Institute of Technology, Goa

**Crop and Weed Detection System: Reducing Pesticide Waste and Enhancing Crop Production**

The agricultural industry faces significant challenges due to the presence of weeds, which compete with crops for essential resources such as nutrients, water, and land. This competition results in decreased crop productivity and inhibits the optimal growth of desired crops. To control weeds, farmers often resort to the use of pesticides. However, the indiscriminate application of pesticides can lead to adverse effects, such as the contamination of crops and potential health risks for humans. Therefore, there is a need for a more targeted approach to weed control in agriculture.

The aim of this research is to develop a system that can accurately detect and differentiate between crop plants and weeds in agricultural fields. By doing so, the system will enable targeted pesticide application exclusively to weeds, minimizing the mixing problem with crops and reducing the waste of pesticides. This paper presents the data preparation steps, including dataset collection, cleaning, image processing, data augmentation, and manual labeling. Additionally, it addresses the problem of weed interference in agriculture and emphasizes the importance of developing a system that can effectively mitigate the negative impacts of weeds and pesticides.

# Abstract

This paper focuses on the development of a crop and weed detection system to reduce pesticide waste and enhance crop production. The dataset used in this study contains 1300 images of sesame crops and different types of weeds, with each image labeled in YOLO format. The data preparation process involved collecting 589 images, cleaning the dataset to remove irrelevant or misleading data, resizing the images to a manageable size, and augmenting the dataset using data augmentation techniques. Manual labeling of the images was conducted by drawing bounding boxes to differentiate between crops and weeds. The paper addresses the problem of weed interference in agriculture, highlighting the negative impact on crop productivity and the potential risks associated with pesticide use. The aim of the study is to develop a system that can accurately detect and differentiate between crops and weeds, allowing targeted pesticide application exclusively to weeds, thereby reducing the mixing problem with crops and minimizing pesticide waste.

# Data Preparation

In this section, we discuss the process of collecting and cleaning the dataset. It involves capturing photos of crops and weeds, followed by dataset cleaning to ensure data quality. The resulting dataset comprises 546 images after the cleaning process.

## Dataset Collection.

The dataset used in this study consists of 589 images captured from agricultural fields, including photos of both crops and different types of weeds. These images serve as the foundation for training the crop and weed detection system.

## Dataset Cleaning.

Once the dataset was collected, a crucial step was to clean and refine it. Dataset cleaning is essential to ensure that only relevant and accurate data is used for training the detection model. In this step, any images that were deemed irrelevant or potentially misleading were removed from the dataset. By eliminating low-quality or misleading images, the cleaning process helps enhance the performance of the detection model. After the cleaning process, the dataset was refined, resulting in a reduced dataset containing 546 images.

## Data Augmentation

Data augmentation is a technique used to increase the dataset size and enhance the model's ability to generalize. In this research, data augmentation was applied to the dataset to create additional variations of the existing images. By applying transformations such as rotations, flips, zooms, and shifts, the augmented dataset provides a broader range of training examples for the model. The Keras ImageDataGenerator, a popular tool in deep learning, was utilized to perform these augmentations automatically. The augmentation process resulted in an expanded dataset containing 1300 images. The increased diversity and variability in the dataset contribute to improving the model's performance and its ability to handle real-world scenarios.

## Image Processing

The images obtained from the dataset were initially of high resolution, with a size of 4000x3000 pixels. However, using such large images for training can significantly increase computational requirements and training time. Therefore, image processing techniques were employed to resize the images to a more manageable size of 512x512 pixels. This resizing process allows for efficient training of the model while preserving the color information present in the images. By reducing the image size, the computational burden on the model is reduced, enabling faster and more efficient training.

## Manual Labelling

Accurate labeling of the image data is a critical step in training a supervised machine learning model. In this research, manual labeling was undertaken to annotate the images with bounding boxes that indicate the presence of crops and weeds. This labor-intensive process involved visually inspecting each image and drawing bounding boxes around the respective regions of interest. By manually labeling the images, the dataset becomes properly labeled, enabling the model to learn and distinguish between crops and weeds effectively. The labeled dataset serves as the ground truth for training the crop and weed detection system, allowing it to make accurate predictions in real-world scenarios.

By meticulously addressing each aspect of the data preparation phase, including dataset collection, cleaning, image processing, data augmentation, and manual labeling, the research ensures the availability of a high-quality dataset and sets the foundation for developing an effective crop and weed detection system.

# Model 1

I will be using Tensorflow 2.0 Api for all the model and training purposes

## Data Preprocessing

I divided the overall training dataset into training data and validation data. Then use the test data provided to check the accuracy of the model. For achieving this we will use ImageDataGenerator from tensorflow.

## Model Architecture and Training

We create a convolutional neural network (CNN) architecture for image classification using the VGG16 model. The VGG16 model is loaded with pre-trained weights from the ImageNet dataset, excluding the fully connected layers. The head of the network consists of a flatten layer, followed by two dense layers with 4096 units and ReLU activation. Dropout layers are incorporated to prevent overfitting, and a final dense layer with softmax activation assigns probabilities to each class. The base model layers are frozen to retain the pre-trained weights, while only the head model is trainable. The model is compiled with the categorical cross-entropy loss function and Adam optimizer with a learning rate of 0.001. This architecture leverages the powerful feature extraction capabilities of VGG16 and demonstrates the effectiveness of the proposed approach for image classification tasks. By freezing the base model and training only the head, the model can efficiently learn and classify images with high accuracy.

## Model Evaluation

The trained model showed a precision of 98.3% while accuracy was about 97%

# Model 2(SVM)

## Data Preprocessing

### Image processing

The feature extraction process plays a crucial role in various computer vision tasks. In this project, we aimed to extract meaningful features from ground truth labeled images using a pre-trained model. The provided code snippet illustrates the workflow involved in this process. The code begins by iterating over the labeled images and retrieving their corresponding filenames. Each image is then read using the OpenCV library and converted from the default BGR color space to RGB.

### Region Extraction

To focus on the relevant content of the image, bounding box coordinates are extracted from the labels, indicating the region of interest. These coordinates are used to crop the region of interest from the original image, which is then resized to a fixed dimension of 224x224 pixels. This resizing step ensures consistency in the input size for the subsequent feature extraction. The resized image is passed through a pre-trained model, where the last two fully connected layers have been removed, leaving only the feature extraction layers. The model predicts the feature vector for the image, which captures the essential characteristics of the content.

### Feature Extraction

Based on the type of image (training or testing), the code segregates the feature vectors along with their corresponding class labels into separate lists, namely 'train\_features' and 'test\_features'. This separation facilitates the subsequent use of these features for training and evaluating machine learning models. In summary, this code snippet outlines the process of extracting features from ground truth labeled images. By utilizing a pre-trained model, we obtain compact and informative feature vectors that encapsulate the content of the images. These features serve as a foundation for training and testing machine learning models in various computer vision applications.

## Model Architecture and Training

Support Vector Machines (SVMs) have proven to be effective tools for classification tasks in various domains. In this research paper, we explore the utilization of SVMs in the field of image analysis. The presented code snippet focuses on the construction and training of an SVM model with a linear kernel for image classification. We describe the model architecture, including the linear kernel and regularization parameter, and discuss the training process. The trained model is then used to predict class labels for the testing data, and the accuracy of the model is evaluated. This research contributes to the advancement of image analysis techniques by leveraging the robustness and efficiency of SVMs.

The SVM model implemented in this research employs a linear kernel, enabling the definition of a hyperplane that separates different classes in the feature space. The linear kernel is well-suited for scenarios where the data is linearly separable. Additionally, a regularization parameter, denoted as C, is introduced to control the trade-off between maximizing the margin and minimizing classification errors. A careful selection of C is vital for achieving optimal classification performance, as a small C value allows for a wider margin but may lead to increased misclassifications, while a large C value reduces the margin but minimizes misclassifications.

The training process involves fitting the SVM model to the training data. The training dataset consists of labeled images represented as feature vectors (X\_train) and their corresponding class labels (y\_train). By applying the 'fit' method on the SVM model, the optimal hyperplane is learned, which effectively separates the different classes based on the provided training data. Moreover, to enable probability estimation, the 'probability=True' argument is set, allowing the model to estimate the likelihood of an instance belonging to each class.

## Model Evaluation

The trained model showed a precision of 98.8% while accuracy was about 96.8%

# Detecting the Crops and Weeds

I implemented an object detection algorithm using the Selective Search algorithm for region proposal and an SVM classifier for object classification. The algorithm takes an input image and performs selective search to generate region proposals. For each region proposal, the algorithm extracts features using a pre-trained model and then classifies the regions as either "crop" or "weed" using an SVM classifier. The algorithm applies non-maximum suppression to eliminate redundant bounding boxes and outputs a list of detected objects. The algorithm visualizes the detected objects by drawing bounding boxes and labels on the input image. This object detection pipeline can be used in various applications such as agriculture, surveillance, and image analysis. By leveraging the power of selective search and a trained SVM classifier, the algorithm provides an efficient and accurate approach for detecting and localizing objects of interest within an image.

## Algorithm

Steps

1. Start
2. Read the input image file path (img\_path), confidence threshold (confidence), and IoU threshold (iou\_thresh).
3. Load the input image using the specified image file path.
4. Apply Selective Search algorithm to generate region proposals on the input image.
5. Select the top 2000 region proposals for further processing.
6. Initialize empty lists: pred\_crop, pred\_weed, and final.
7. For each region proposal in the selected region proposals, do:
8. Extract the ROI from the input image based on the region proposal coordinates.
9. Resize the ROI to a fixed size of 224x224 pixels and normalize pixel values.
10. Extract features from the resized ROI using a pre-trained model without the last two fully connected layers.
11. Use the SVM classifier to predict the class probabilities and labels for the extracted features.
12. If the predicted label is "crop" and the maximum class probability is above the confidence threshold, add the region proposal and its maximum class probability to the pred\_crop list.
13. If the predicted label is "weed" and the maximum class probability is above the confidence threshold, add the region proposal and its maximum class probability to the pred\_weed list.
14. Apply Non-Maximum Suppression (NMS) for the predicted crop objects:
15. Initialize an empty list to store the final detected objects.
16. Sort the pred\_crop list based on the maximum class probabilities in descending order.
17. Iterate through each predicted crop object in the sorted list:
18. Check if the object's bounding box overlaps significantly with any previously selected bounding box using the IoU threshold.
19. If there is significant overlap, discard the current bounding box.
20. If there is no significant overlap, add the current bounding box, maximum class probability, and label ("crop") to the final list.
21. Apply Non-Maximum Suppression (NMS) for the predicted weed objects:
22. Perform the same steps as in the NMS for crop objects, but with the pred\_weed list.
23. Draw bounding boxes and labels on the input image for each object in the final list.
24. Display the annotated image.
25. Save the annotated image as "prediction.jpeg".
26. Return the final list of detected objects.
27. End.

## Code Description

The code implements an object detection algorithm using a combination of Selective Search and an SVM classifier. The objective of the algorithm is to detect and classify objects, specifically distinguishing between "crop" and "weed" classes, in an input image.

The code begins by loading the input image specified by the img\_path parameter. It then applies the Selective Search algorithm, available through the OpenCV library, to generate a set of region proposals. Selective Search is a region-based object proposal method that identifies potential object locations by grouping pixels based on similarity and proximity.

Next, the algorithm selects the top 2000 region proposals for further processing. For each selected region, referred to as a region of interest (ROI), the code extracts the corresponding image patch from the original image. The ROI is resized to a fixed size of 224x224 pixels, which is a common input size for many deep learning models.

Feature extraction is performed on each resized ROI using a pre-trained model. The model utilized in this code has its last two fully connected layers removed, as indicated by the model\_without\_last\_two\_fc variable. The features are extracted by passing the resized ROI through the model, resulting in a feature vector.

After feature extraction, the code employs an SVM classifier to predict the class probabilities and labels for the extracted features. The SVM model used for classification is referenced by the svm\_model variable. The predicted label and corresponding class probability are obtained for each ROI.

The algorithm then proceeds to filter and process the predicted results. It first checks if the predicted label is "crop" and if the maximum class probability exceeds the specified confidence threshold (default value: 0.9). If both conditions are met, the ROI is considered a predicted "crop" object and added to the pred\_crop list. Similarly, if the predicted label is "weed" and the maximum class probability exceeds the confidence threshold, the ROI is considered a predicted "weed" object and added to the pred\_weed list.

Non-Maximum Suppression (NMS) is applied to eliminate redundant bounding boxes that may overlap for the same object class. The algorithm iterates through the predicted "crop" objects and retains only the bounding box with the maximum class probability. If multiple bounding boxes overlap significantly (determined by the IoU threshold, default value: 0.1), only the one with the highest probability is kept. The same NMS process is applied to the predicted "weed" objects.

The final list of detected objects, stored in the final list, includes the bounding box coordinates, maximum class probabilities, and corresponding labels. The algorithm then visualizes the detected objects by drawing bounding boxes and labels on the input image using OpenCV. The annotated image is displayed and saved as "prediction.jpeg".

In summary, this code implements an object detection algorithm that utilizes Selective Search for region proposal and an SVM classifier for object classification. By combining these techniques, the algorithm can effectively detect and classify objects within an image, specifically focusing on distinguishing between "crop" and "weed" classes.

# Challenges/Hurdles

## Computation Resources

Selective Search involves generating a large number of region proposals, which can be computationally intensive. Processing a large number of regions and extracting features for each one can lead to longer execution times.

## Choice of parameters

The code relies on hyperparameters such as confidence threshold and IoU threshold. Setting appropriate values for these parameters can be challenging and may require careful tuning to achieve the desired trade-off between precision and recall.

## Sensitivity to image variations

The performance of the object detection algorithm can be affected by variations in lighting conditions, object sizes, orientations, and occlusions. These factors can influence the accuracy of the region proposals and the SVM classifier, leading to potential detection errors.

## Limited Handling of Overlapping Objects

Although Non-Maximum Suppression (NMS) is applied to remove redundant bounding boxes, it may not handle overlapping objects perfectly. Depending on the specific implementation and chosen IoU threshold, the algorithm may fail to accurately separate closely located objects, leading to fused or fragmented bounding boxes.