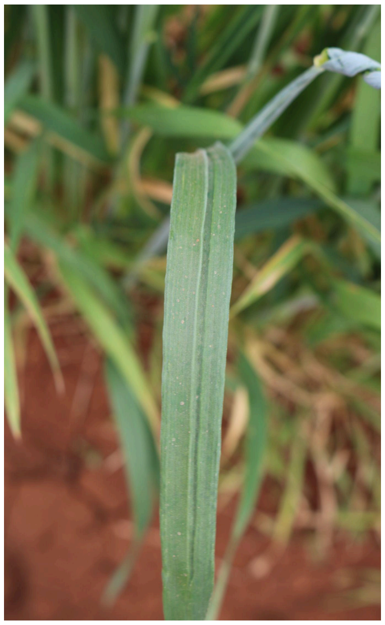
Leaf GAN

Dataset Description:

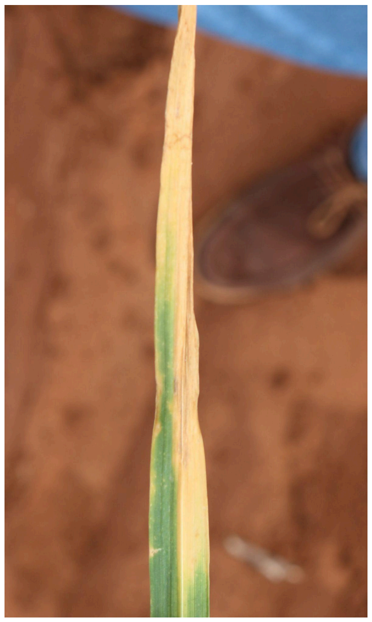
The leaf dataset comprises images categorized into three distinct classes: healthy, septoria, and stripe rust. These categories represent different conditions or diseases affecting the leaves.

(a) Healthy: This class consists of 102 images depicting healthy leaves, free from any diseases or abnormalities.
(b) Septoria: There are 97 images in this category, depicting leaves affected by septoria, a fungal disease characterized by small, dark spots on the leaves.
(c) Stripe Rust: The largest category in the dataset, comprising 208 images, depicts leaves afflicted with stripe rust, a common fungal disease recognizable by the yellowish-orange stripes or streaks on the leaf surface.

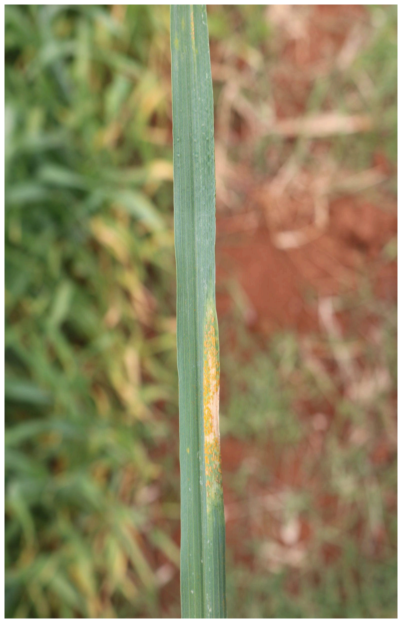
Total Images: The dataset contains a total of 407 images.



(a)



(b)



(c)

Novelty:

This Indian-origin dataset stands out as a pioneering resource in plant pathology and agriculture.

Proposed Algorithm:

```
Algorithm 1: Gradcam
  Function 1: Preprocessing
  Input: input_folder, output_folder, size
  Output: Resized images in the output folder
  if output_folder does not exist then
      Create output_folder;
  foreach filename in input_folder do
      input_image_path ← input_folder + filename;
      if filename is an image then
          Open original_image from input_image_path;
          Resize original_image to size × size using ANTIALIAS;
          Save resized_image to output_folder with same filename;
  Input: input_folder, output_folder, num_augmentations_per_image
  Output: Augmented images in the output folder
  Initialize Augmentor Pipeline with input_folder and output_folder;
  Define augmentation operations;
  Rotate images with probabilities and angles;
  Flip images left-right and top-bottom with probability 0.5;
  Perform augmentation for each image in input_folder with
    num_augmentations_per_image samples;
  Function 2: Resnet Classification
  Input: modelname, train_dataset, epoch
  Output: Fine-tuned model with 3 output classes
  model_ft = models.resnet101(pretrained=True);
  model_name = modelname;
  print(model_name);
  print(train_dataset);
  print("Number of epoch: %d" % epoch);
  num_ftrs = model_ft.fc.in_features;
  model_ft.fc = nn.Linear(num_ftrs, 3);
  for each param in model_ft.parameters() do
      param.requires_grad = True;
  Function 3: Gradcam masking
  GradCAM_Init model Initialize all_fmaps and all_grads as empty
  OrderedDicts for module in model.named_modules() do
  Register func_f as a forward hook for module[1] Register func_b as a
    backward hook for module[1] generate target_layer
    fmaps ← self._find(all_fmaps, target_layer)
    grads ← self._find(all_grads, target_layer)
    weights ← self._compute_grad_weights(grads)
    gcam ← (fmaps[0] * weights[0]).sum(dim = 0)
    gcam ← clamp(gcam, min = 0.) gcam− = gcam.min()
    gcam/ = gcam.max() return gcam.detach().cpu().numpy()
```

Implementation and Result:

1. Dataset Preparation:

Resizing: Original leaf images were uniformly resized to 500x500 pixels to standardize dimensions and streamline subsequent processing steps. (PIL Lib)

Augmentation: Eight new images were generated from each original leaf image through augmentation- total 9, resulting in a total of 3663 images. Augmentation techniques enhance dataset diversity and model robustness.

Labeling: Each augmented leaf image was labeled with '0' to denote its origin from the leaf dataset. This labeling simplifies classification and evaluation tasks.

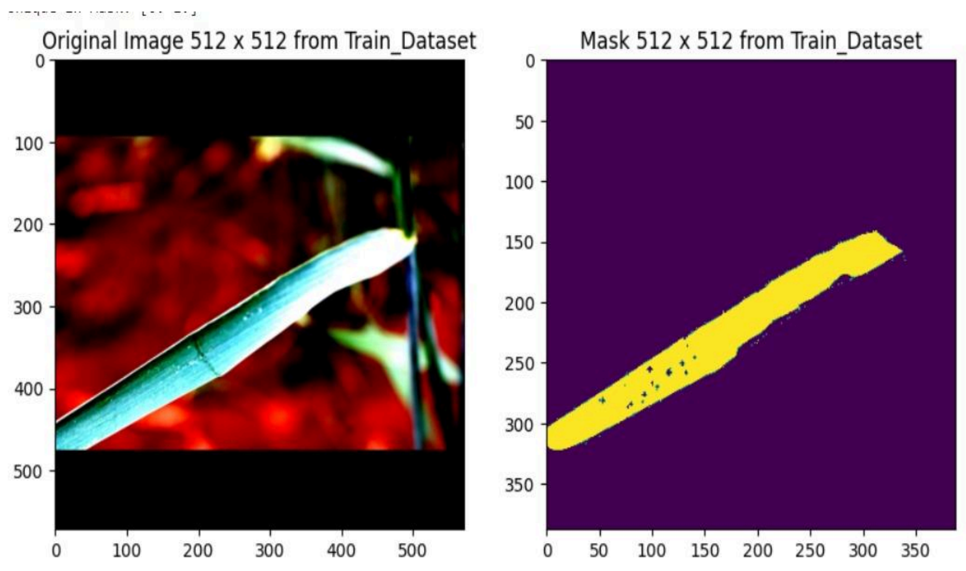
Partitioning Images: Original images were resized to 1500x1500 pixels and then partitioned into nine 500x500 images, each assigned the label '1'. This partitioning strategy ensures consistency and uniformity in dataset composition.

ImageNet Dataset Integration: An ImageNet dataset subset containing 3663 images was integrated into the dataset. Each ImageNet image was partitioned into nine equivalent images to match the structure of the resized leaf images and labeled '2'. This integration enhances dataset diversity and complexity.

2. Model Training:

1. Unet with Pretrained Network (VGG16):

Utilizes a U-Net architecture for semantic segmentation, initializing the encoder with pre-trained VGG16 weights.
Benefits from learned feature representations from VGG16, enhancing segmentation performance.
Allows for fine-tuning to adapt pre-trained weights to the segmentation task.

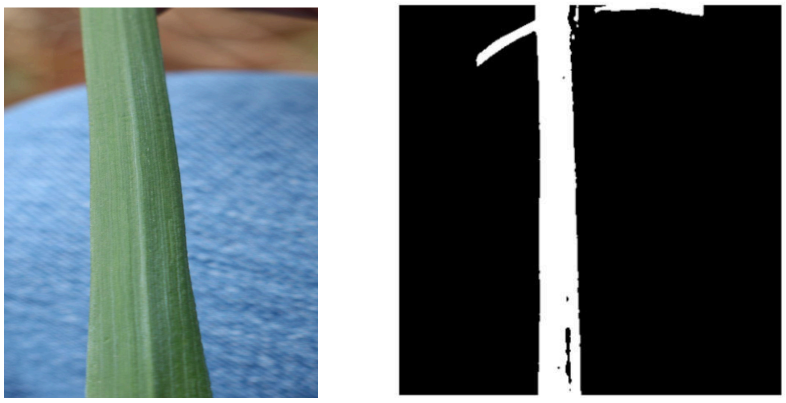

Original Image 512 x 512 from Train_Dataset
Mask 512 x 512 from Train_Dataset

Input

Output

2. Unet from Scratch:

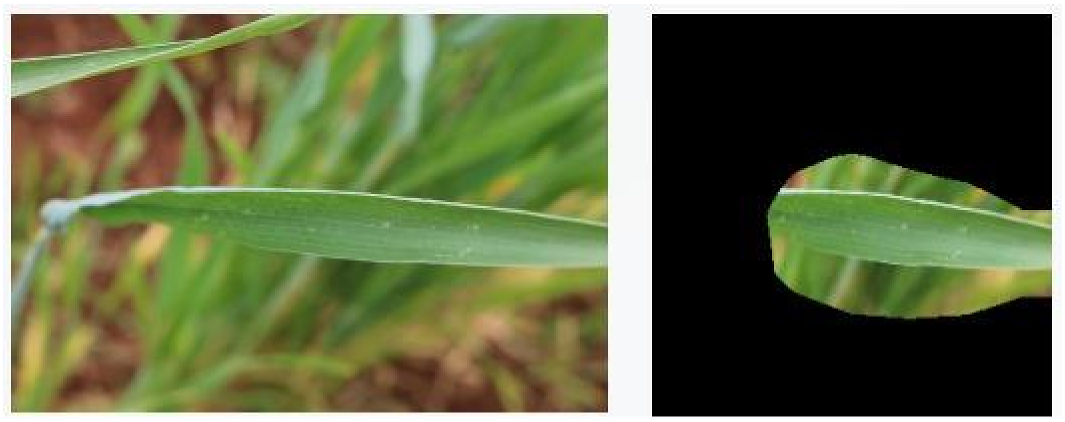
Trains a U-Net architecture from random initialization without using pre-trained weights.
Learns task-specific features directly from the dataset.
Offers flexibility and potential for better performance tailored to the dataset characteristics.



Input Output

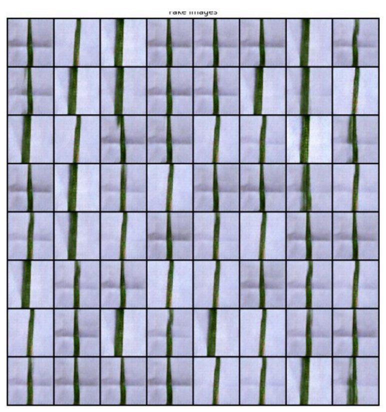
3. Grad-CAM:

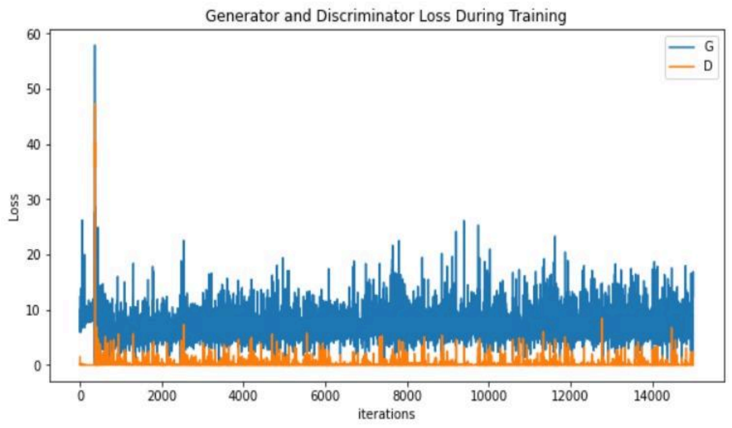
Visualizes and interprets CNN models, particularly classifiers, without training a separate model.
Generates heat maps highlighting regions crucial for the model's prediction.
Provides insights into model decisions, aiding in validation and interpretation of CNN behavior.



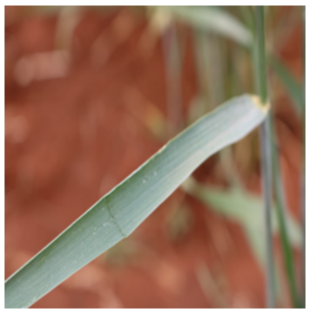
3. Results:

Output with raw image as input:

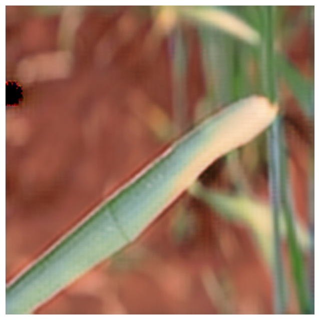




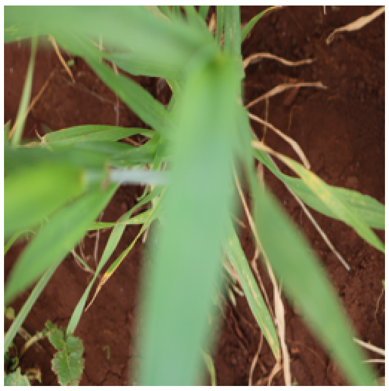
Output with applying the Preprocessing and applying gradCAM:



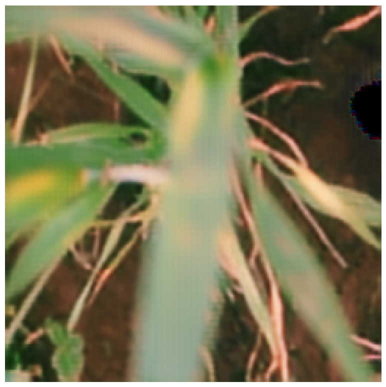
Input image
(healthy)



Output image
(stripe rust)



Input image
(healthy)



Output image
(stripe rust)