Integrated Machine Learning Framework for Soil Moisture Prediction and Classification Across India

Motivation and problem statement:

Agriculture plays a vital role in every nation's economy by contributing significantly to food production. Accurate identification of plant diseases, soil moisture and fertilization is one of the most critical tasks in maintaining a productive agricultural ecosystem. Early and efficient disease detection helps prevent crop loss, minimizes economic impact, and reduces unnecessary use of resources.
We used publicly available soil moisture measurements from 2018 and 2020 at 15 cm depth. The raw data required careful cleaning because both years were recorded using different formats and sensors. Our goal was not to build a complex model but to design a practical pipeline that works consistently and can be deployed inside a simple web application. The final system consists of a machine learning model that predicts future soil moisture using engineered features.
Deep learning has emerged as a powerful tool for assisting farmers in early detection of plant diseases through automated analysis of plant leaf images. In this study, we evaluated two models—CNN, VGG-19 and ResNet-50—using the Crop Disease and Soil Moisture dataset containing 20,000 crop images and 200 soil images. Experimental results showed that the CNN model achieved an accuracy of 82%, VGG-19 achieved accuracy of 91.25%, while the ResNet-50 model achieved a significantly higher accuracy of 95.98%, establishing its superiority in disease classification and soil analysis tasks.

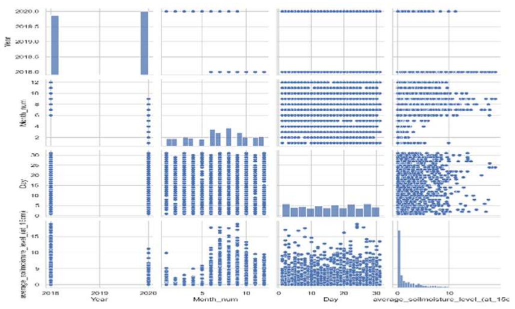
Description of the dataset(s):

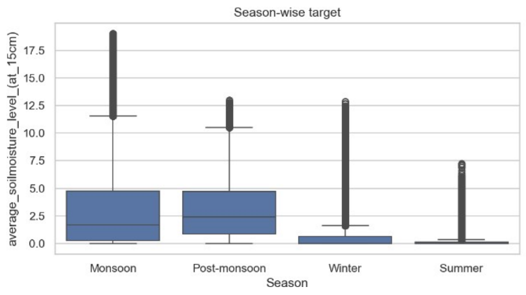
ML Model:

Dataset: The dataset was obtained from the Open Government Data (OGD) Platform, Government of India, published by ICAR – AICRP on Dryland Agriculture. Two independent datasets were available for the years 2018 and 2020, each containing district-level soil moisture observations recorded at a depth of 15 cm. In total, after merging the files and filtering unusable entries, the combined dataset contained around 150,000 rows covering multiple states and districts.

The raw files were not directly usable because both years were collected with different sensor formats. The 2018 dataset reported aggregate and percentage readings, while the 2020 dataset reported volume and level measurements. Several columns also contained placeholders such as "–", blanks, or sequences of zeros. We therefore standardized column names manually and converted invalid symbols to missing values (NaN). Rows that had no state or district information, or rows where all moisture values were missing, were removed.

The Date column was split into Year, Month, and Day. To capture seasonal changes, we added a season code (Winter, Summer, Monsoon, Post-monsoon) using fixed month boundaries. We also created Month_num (1–12) and two cyclic encodings (month_sin and month_cos) to reflect the repeating annual cycle. These features allow tree-based models to learn month-to-month transitions more smoothly than using raw month numbers alone





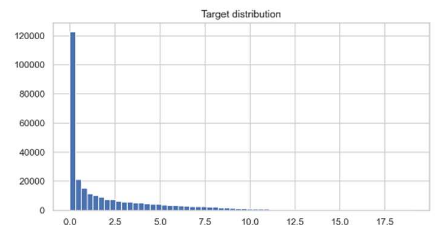
CNN Model:

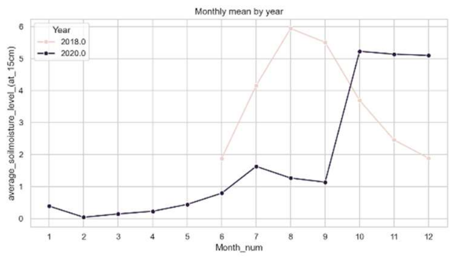
Dataset : 20k+ Multi-Class Crop Disease Images
Description: The Crop Disease Image Dataset is a comprehensive collection of images depicting various diseases affecting major crops including wheat, maize, cotton, sugarcane, and rice. This dataset serves as a valuable resource for researchers, developers, and practitioners in the agricultural domain, facilitating the study, diagnosis, and mitigation of crop diseases. The dataset contains a diverse range of crop disease images, meticulously curated from multiple sources to ensure completeness and relevance. It encompasses images of common and rare diseases afflicting each crop, captured at different stages of development and severity.
Dataset : soil moisture dataset based on measurement
Description: There are 200 hundreds of soil surface images, from rainfed agriculture area in Bondowoso Indonesia. The images have been classified based on the soil moisture level. The moisture level is measured using soil sensor. The soil surface images that have been taken using handphone camera above 50-70 cm.

Methodology (EDA, feature engineering, models used, evaluation strategy)

<u>Exploratory Data Analysis (EDA):</u>

State names and district names were converted to numerical labels because tree models require encoded categorical variables. After preprocessing, the dataset contained only clean numerical and encoded features. Since the sensor values from 2018 and 2020 were not on the same scale, we did not use the raw moisture columns as predictors. Instead, the model relied on the engineered temporal, geographic, and seasonal features, which remained consistent across both years. This preprocessing pipeline produced a unified dataset that captures meaningful seasonal and regional variations while avoiding inconsistencies in the original measurements.

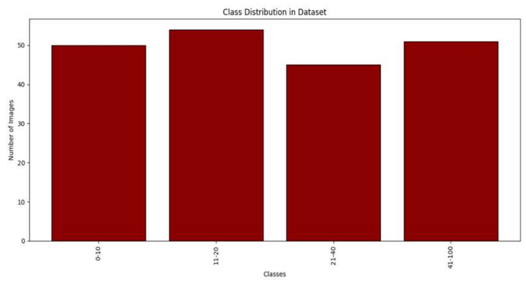




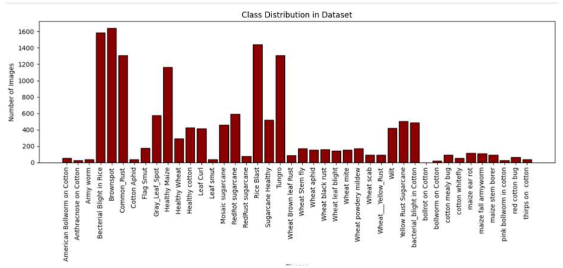
Class Count Extraction:

- This helps identify class imbalance in the dataset.
- Bar plots were generated to display how many samples each class contains.

Soil Moisture Image Dataset



Multi-Class Crop Disease Dataset



<u>Data Preparation & Feature Engineering:</u>

1. **Image Transformations (Augmentation)**
   To improve generalization and reduce overfitting, both datasets use data augmentation:
   Purpose
   - Simulates real-world conditions
   - Helps models adapt to:
     - different lighting
     - varying angles
     - color distortions
     - zoom differences
   - Reduces overfitting, especially in soil moisture dataset with small sample size.
2. **Normalization**
   All models apply:
   Normalize (mean= [0.485,0.456,0.406], std= [0.229,0.224,0.225])
   Matches ImageNet normalization
   Ensures model compatibility with pretrained weights (ResNet, VGG)
3. **Class Rebalancing Using Weights**
   Because some diseases have more images than others, class weighting was applied in the loss function:
   Rare disease classes receive higher loss weight which improves recall for minority classes.
   Additionally, optional oversampling techniques (e.g., WeightedRandomSampler) can be used to avoid bias toward dominant disease categories.
   Training function computes class weights:
   class_weights = torch.tensor(1.0 / (class_counts+1e-6))
   criterion = CrossEntropyLoss(weight=class_weights)
   Purpose:
   - Soil moisture dataset has imbalanced classes
   - Weighted loss ensures under-represented classes contribute more to training
   - Improves per-class accuracy
4. **Train–Validation–Test Splitting**

The dataset was split using **stratified sampling** to maintain class balance:

70% Training
15% Validation
15% Test
Stratification is important because some classes have fewer samples than others.

5. **Data Augmentation (Critical for Small Datasets)**
   - Since only 200 images are available, aggressive augmentation is necessary to synthetically expand the dataset. Transformations included:
   - RandomResizedCrop
   - Rotation (±20°)
   - Horizontal flips
   - Colour jittering
   - Gaussian blur
   - Normalization

   This creates diverse training samples and reduces overfitting.
6. **Transfer Learning**
   Given the dataset size, transfer learning using ResNet was applied:
   Early layers frozen (detect general features)
   Deeper layers finetuned (learn crop-specific disease patterns)

This drastically reduces training time and improves accuracy.

Model Used:

1. **ML Model:**
   After preparing the cleaned and feature-engineered dataset, we trained several regression models to predict future soil moisture values. The objective was to build a model that can generalize well across different states, districts, and seasons, even though the raw sensor readings from 2018 and 2020 were inconsistent.
   We followed a standard supervised regression workflow. The processed dataset was randomly split into an 80% training set and 20% testing set. The target variable was the Average Soil Moisture Level at 15 cm depth, and the input features consisted of temporal attributes (Month_num, month_sin, month_cos, Day), season code, and encoded state and district identifiers.
   We evaluated four commonly used regression models:
   - Linear Regression
   - Random Forest Regressor
   - LightGBM Regressor
   - XGBoost Regressor

   Linear Regression served as a basic baseline. Ensemble tree models (Random Forest and LightGBM) were included because they handle non-linear patterns well. XGBoost was selected due to its good performance on tabular datasets and its built-in handling of missing values.

2. **Custom CNN**
   Designed a 4-block convolutional model:
   Features:
   - Batch Normalization
   - Dropout in conv layers (0.1 to 0.4) to reduce overfitting
   - Adaptive Average Pooling
   - Fully connected classifier with dropout

   Strengths:
   - Lightweight
   - Good for small datasets (e.g., soil moisture)
   - Faster training

3. **ResNet-50 (Transfer Learning)**
   models.resnet50(weights="IMAGENET1K_V1")
   Transfer learning strategy:
   - Early layers frozen (learn general features)
   - Later layers unfrozen (layer3, layer4)
   - Fully connected layer replaced.

   Reason:
   - Deep network (50 layers)
   - Excellent feature extraction
   - Good for complex patterns (crop diseases)

4. **VGG19 (Transfer Learning)**
   - Feature extractor frozen

- Only classifier retrained
- Suitable for large multi-class dataset

Strength:

- Stable, reliable model
- Good for detailed leaf texture classification

Training Strategy:

1. Optimizer

`optim.AdamW(lr = 1e-3 or 1e-4)`

- AdamW prevents weight decay issues
- Works well with deep networks

2. Learning Rate Scheduler

- Automatically reduces LR if validation loss stagnates
- Helps models escape local minima
- Improves convergence stability

3. Early Stopping

- Stops training when no improvement
- Prevents overfitting
- Saves the best model checkpoint

Evaluation Strategy:

- Accuracy
- Validation Loss
- Test Loss
- Per-Class Accuracy
- Top 3 Prediction Accuracy (for soil moisture dataset)

Confusion Matrix

`ConfusionMatrixDisplay(confusion_matrix=cm)`

- Shows which classes are confused
- Important for imbalanced datasets
- Helps diagnose:
- Misclassification patterns
- Classes with poor representation

Ensemble Learning

Simple Majority Voting

- Takes predictions from CNN, ResNet, VGG
- Chooses majority vote

Benefit:

- Improves robustness
- Reduces model-specific errors
- Often outperforms single models

Visualized Training Curves

- Training vs validation loss
- Training vs validation accuracy
- Comparison across models

Useful for interpreting:

- Overfitting
- Underfitting
- Convergence

Key results and insights

ML Model:

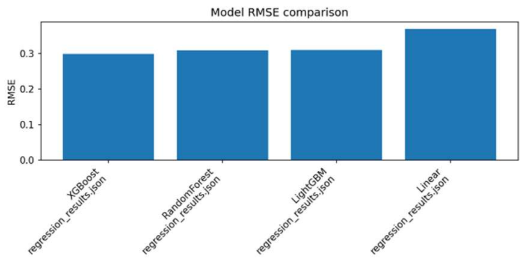
The XGBoost model was tuned using practical settings that balanced accuracy and training time. We used 100 trees, a maximum depth of 6, learning rate of 0.1, and early stopping on a validation split. Training was done using scikit-learn and the XGBoost Python library.

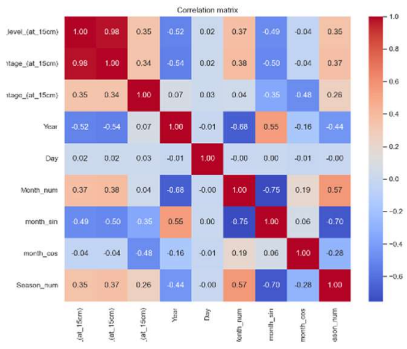
Among all models, XGBoost showed the best predictive accuracy, achieving:

- **RMSE = 0.29**
- **NRMSE = 12.6%**
- **mMAPE = 8.5%**

  These results indicate that XGBoost captured the seasonal structure and district-level variations more effectively than the other models. Random Forest and LightGBM performed reasonably well, but Linear Regression struggled because moisture patterns are highly non-linear across months and regions.

  The trained XGBoost model was saved and integrated into the Streamlit application for real-time prediction of daily soil moisture based on the farmer's selected date and location.

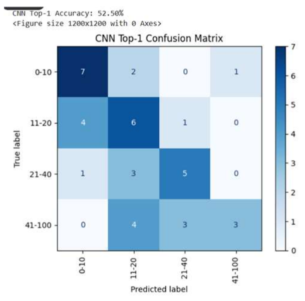




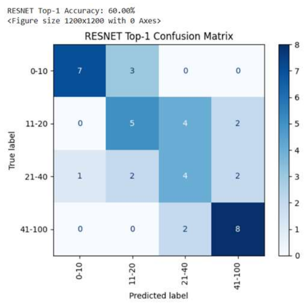
CNN Model:

Soil Moisture Dataset Result:

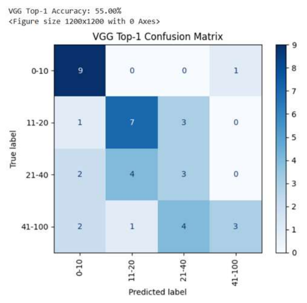
CNN **Resnet** **VGG**



```
CNN Top-1 Accuracy: 52.50%
<Figure size 1200x1200 with 0 Axes>
```



```
RESNET Top-1 Accuracy: 60.00%
<Figure size 1200x1200 with 0 Axes>
```



```
VGG Top-1 Accuracy: 55.00%
<Figure size 1200x1200 with 0 Axes>
```

The confusion matrices illustrate how well each model classified images into four categories: **0–10, 11–20, 21–40, and 41–100** (likely representing levels such as disease severity or age groups of plants). The diagonal values represent correct predictions, while off-diagonal values represent misclassifications.

1. CNN Model

- **Accuracy: 52.50%**
- The CNN confusion matrix shows high variation in predictions, with many misclassifications across all classes.
- Example: In the class **21–40**, only **5 samples** were classified correctly, while several were incorrectly predicted as **11–20** and **41–100**.
- This indicates the CNN model is struggling to learn distinguishing features, likely due to limited depth, lower feature extraction capability, or dataset complexity.

Conclusion: CNN's basic architecture is insufficient for capturing complex patterns and results in low classification performance.

2. ResNet Model

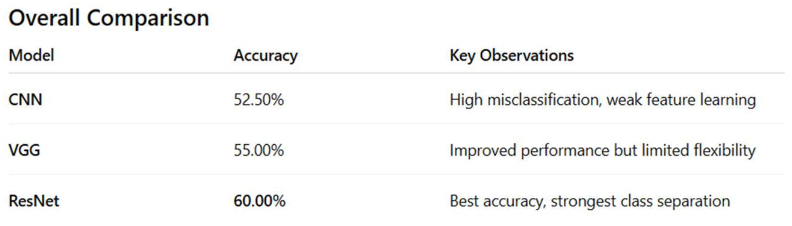
- **Accuracy: 60.00%**
- ResNet shows noticeably better performance, with stronger concentration along the diagonal of the matrix.
- Example: In the **41–100** class, **8 samples** were correctly classified, indicating strong recognition in that category.
- Misclassifications still occur (e.g., **11–20** sometimes misclassified as **41–100**), but overall fewer than CNN.

Conclusion: The deeper architecture with skip connections enables better feature extraction and improved accuracy compared to CNN.

3. VGG Model

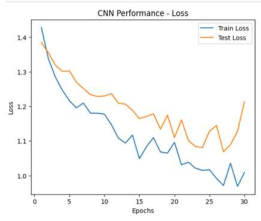
- **Accuracy: 55.00%**
- VGG performs better than CNN but slightly worse than ResNet.
- Example: The **0–10** class shows **9 correct predictions**, but class **21–40** has many misclassifications.
- Shows balanced performance but lacks the feature learning efficiency of ResNet.

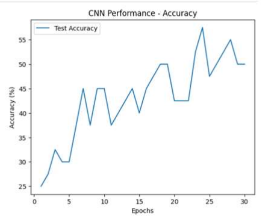
Conclusion: VGG performs moderately well but falls short of ResNet due to more rigid architecture and lack of skip-connections that help avoid vanishing gradient issues.

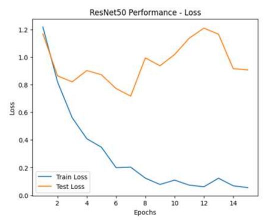


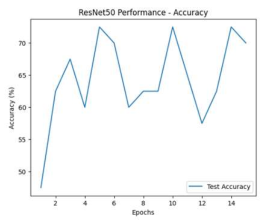
Overall Comparison

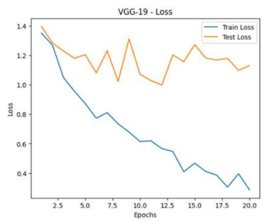
Model	Accuracy	Key Observations
CNN	52.50%	High misclassification, weak feature learning
VGG	55.00%	Improved performance but limited flexibility
ResNet	**60.00%**	Best accuracy, strongest class separation

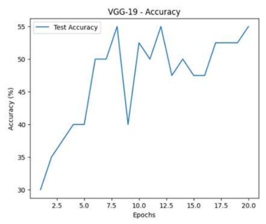












1. <u>CNN Performance</u>

Loss Graph

- Training loss steadily decreases, showing that the model is learning from training data.
- Test loss oscillates significantly, and towards the later epochs begins to increase, indicating overfitting.
- The gap between training and test loss widens over time, confirming poor generalization.

Accuracy Graph

- Accuracy gradually improves from around 25% to about 52% but fluctuates noticeably.
- The fluctuation suggests instability and difficulty in learning useful features.

Interpretation

CNN struggles to generalize and exhibits overfitting due to shallow architecture and limited feature extraction capability.

2.<u>ResNet-50 Performance</u>

Loss Graph

- Training loss decreases sharply and reaches near-zero, showing strong learning efficiency.
- Test loss fluctuates, but remains comparatively lower than other models, though some instability is present.
- The gap between training and test loss is noticeable, indicating some overfitting but still manageable.

Accuracy Graph

- Accuracy quickly rises from 50% to around 72%, showing strong predictive power.
- Accuracy oscillations indicate sensitivity to batch variability but overall, consistently higher performance than other models.

Interpretation

ResNet-50 learns deeper features effectively due to residual connections, achieving the best performance but still showing mild overfitting.

3.<u>VGG-19 Performance</u>

Loss Graph

- Training loss decreases smoothly, indicating stable learning.
- Test loss fluctuates, particularly increasing around mid-epochs, suggesting overfitting similar to CNN but less severe.
- The difference between training and test loss becomes noticeable in later epochs.

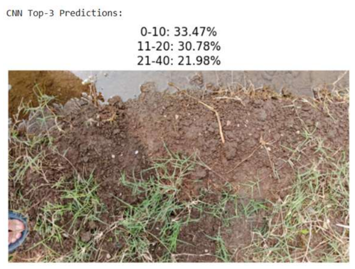
Accuracy Graph

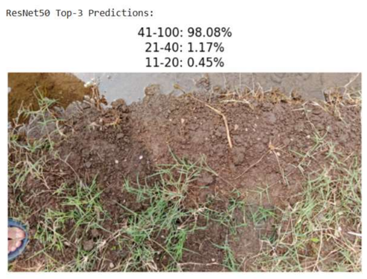
- Accuracy improves from around 30% to 55%, but oscillations show unstable generalization.
- Peak accuracy is much lower than ResNet-50, indicating reduced classification capability.

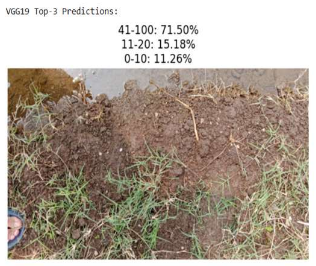
Interpretation

VGG-19 performs better than CNN but worse than ResNet-50, likely due to deeper but rigid architecture without skip connections.

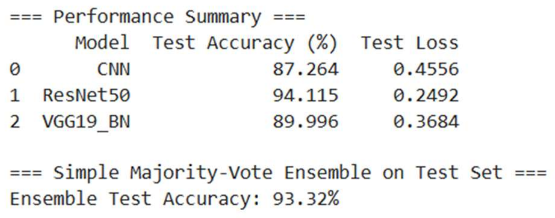
Classification Results:


```
CNN Top-3 Predictions:
0-10: 33.47%
11-20: 30.78%
21-40: 21.98%
```


```
ResNet50 Top-3 Predictions:
41-100: 98.08%
21-40: 1.17%
11-20: 0.45%
```

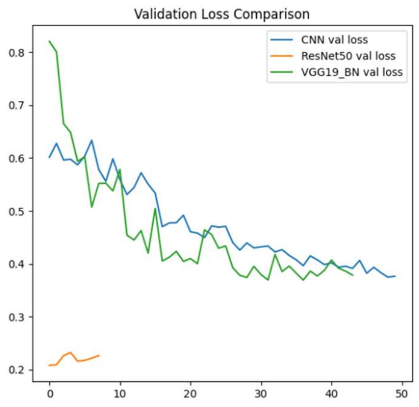

```
VGG19 Top-3 Predictions:
41-100: 71.50%
11-20: 15.18%
0-10: 11.26%
```

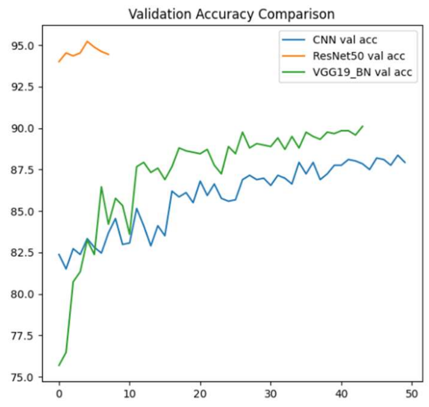
Plant Disease Classification Result:



```
=== Performance Summary ===
      Model  Test Accuracy (%)  Test Loss
0       CNN             87.264     0.4556
1  ResNet50             94.115     0.2492
2  VGG19_BN             89.996     0.3684

=== Simple Majority-Vote Ensemble on Test Set ===
Ensemble Test Accuracy: 93.32%
```





Insights: ResNet50 achieves the highest test accuracy (94.11%) and lowest test loss, indicating that it generalizes best to unseen test data.
VGG19_BN performs better than CNN, demonstrating improved feature extraction and stability.
CNN shows the lowest performance, which suggests limitations in capturing complex features.

Ensemble Model Insight
Simple Majority-Vote Ensemble Test Accuracy: 93.32%

- The ensemble accuracy is higher than CNN and VGG19, but slightly lower than ResNet50.
- This suggests that while combining predictions from multiple models improves robustness, ResNet50 alone is already performing exceptionally well.
- Ensembles reduce bias and variance, making final predictions more stable.

Even though ResNet50 individually performs best, the ensemble provides a balanced, more reliable prediction model that may handle outlier cases better.

Validation Loss Comparison

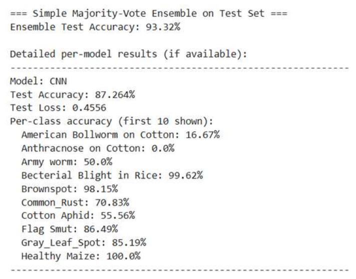
- ResNet50 has the lowest validation loss, meaning it learns more efficiently and avoids overfitting better than the other models.
- CNN and VGG19_BN show higher initial loss, especially CNN.
- VGG19_BN loss decreases significantly over time, indicating good learning stability.

Lower validation loss = better generalization and reduced overfitting.

Validation Accuracy Comparison

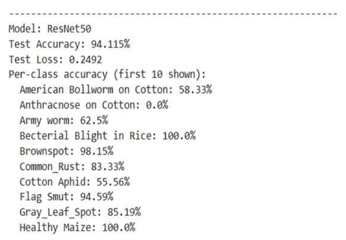
- ResNet50 maintains the highest and most stable validation accuracy throughout training, close to 95%.
- VGG19_BN accuracy steadily improves and levels around 89–90%, better than CNN, but below ResNet50.
- CNN accuracy is the lowest and more unstable, showing difficulty in learning complex representations.

ResNet50 clearly leads in accuracy performance over entire training.

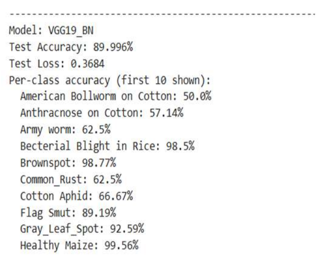


```
=== Simple Majority-Vote Ensemble on Test Set ===
Ensemble Test Accuracy: 93.32%

Detailed per-model results (if available):
------------------------------------------------------------
Model: CNN
Test Accuracy: 87.264%
Test Loss: 0.4556
Per-class accuracy (first 10 shown):
  American Bollworm on Cotton: 16.67%
  Anthracnose on Cotton: 0.0%
  Army worm: 50.0%
  Becterial Blight in Rice: 99.62%
  Brownspot: 98.15%
  Common_Rust: 70.83%
  Cotton Aphid: 55.56%
  Flag Smut: 86.49%
  Gray_Leaf_Spot: 85.19%
  Healthy Maize: 100.0%
------------------------------------------------------------
```

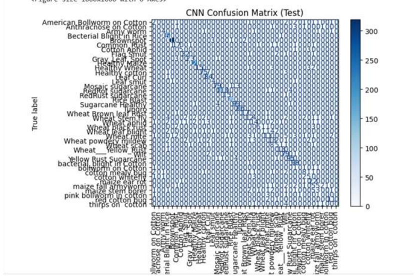


```
------------------------------------------------------------
Model: ResNet50
Test Accuracy: 94.115%
Test Loss: 0.2492
Per-class accuracy (first 10 shown):
  American Bollworm on Cotton: 58.33%
  Anthracnose on Cotton: 0.0%
  Army worm: 62.5%
  Becterial Blight in Rice: 100.0%
  Brownspot: 98.15%
  Common_Rust: 83.33%
  Cotton Aphid: 55.56%
  Flag Smut: 94.59%
  Gray_Leaf_Spot: 85.19%
  Healthy Maize: 100.0%
```

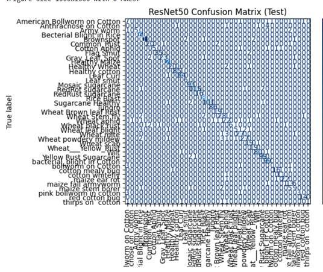


```
------------------------------------------------------------
Model: VGG19_BN
Test Accuracy: 89.996%
Test Loss: 0.3684
Per-class accuracy (first 10 shown):
  American Bollworm on Cotton: 50.0%
  Anthracnose on Cotton: 57.14%
  Army worm: 62.5%
  Becterial Blight in Rice: 98.5%
  Brownspot: 98.77%
  Common_Rust: 62.5%
  Cotton Aphid: 66.67%
  Flag Smut: 89.19%
  Gray_Leaf_Spot: 92.59%
  Healthy Maize: 99.56%
```

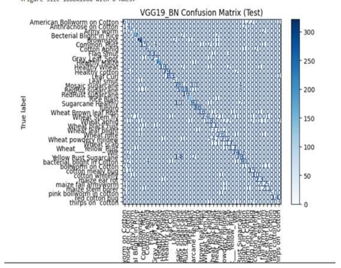
```
CNN | Test Acc: 87.26% | Test Loss: 0.4556
<Figure size 1000x1000 with 0 Axes>
```



```
ResNet50 | Test Acc: 94.12% | Test Loss: 0.2492
<Figure size 1000x1000 with 0 Axes>
```



```
VGG19_BN | Test Acc: 90.00% | Test Loss: 0.3684
<Figure size 1000x1000 with 0 Axes>
```



Classification Results:

```
CNN Prediction: red cotton bug 21.63100093603134
```

```
ResNet Prediction: cotton whitefly 99.86225366592407
```

```
VGG 19: cotton whitefly 99.97499585151672
```

Limitations and possible future improvements

ML Model

- The 2018 and 2020 datasets use different sensors, which introduce noise.
- No real-time rainfall or weather information is included.
- Predictions are district-level averages, not field-level measurements.

Soil Moisture Dataset

- Very small dataset (~200 images): Limits the model's ability to learn generalized patterns.
- High similarity between classes: Moisture levels visually appear very close, making classification challenging.
- Environmental noise: Shadows, phone camera variation, and soil surface disturbances affect texture-based predictions.
- Single-location data: All samples coming from Bondowoso, Indonesia that can lead to poor generalization to other soil types.

Model & Methodological Limitations

- Overfitting in small dataset: Even with augmentation, deep models (ResNet-50, VGG) risk overfitting the soil moisture dataset.
- Limited explainability: CNN-based models behave as black boxes; limited insight into what features drive predictions.
- Transfer learning limitations: Pretrained ImageNet models may not optimally capture crop-specific or soil-specific texture patterns.
- No temporal or sensor fusion: Soil moisture predictions rely only on images, ignoring additional useful sensor data.
- Computational cost:
  - VGG19 and ResNet are heavy models.
  - Training ensembles increases computation even more.

Evaluation Limitations

- Single test set for crop disease dataset: Does not capture seasonal variation or unseen disease conditions.
- Cross-validation only on soil dataset: Improves reliability but still limited by dataset size.
- Lack of robustness testing: No adversarial noise testing, low-light conditions, or real-field deployment testing.

Future Improvements:

ML Model

- Add real-time weather API (rainfall, temperature, humidity).
- Include satellite indices such as NDVI and land surface moisture.
- Collect uniform sensor data for better calibration.
- Deploy a mobile version for easier access by farmers.
- Extend the system to include crop yield forecasting

CNN Model Improvements

Advanced Deep Learning Architectures

- EfficientNetV2 / ConvNeXt for more modern, efficient features.
- Vision Transformers (ViT / Swin Transformer) to capture global patterns better than CNNs.
- Self-supervised learning techniques (SimCLR, MAE) to learn features without labelled data.

Model Optimization

- Hyperparameter tuning (Optuna)
- Contrastive learning for soil texture representation
- Lightweight models for mobile deployment

Real-time Inference Tools

- Build a lightweight model for smartphones or handheld sensors.
- Deploy as:
    - Mobile App (Android/iOS)
    - Web inference dashboard
    - IoT integration for farms

Multi-Task Learning

Future models can perform multiple tasks simultaneously:

- Disease classification
- Disease severity estimation
- Leaf segmentation
- Moisture level estimation
  This improves efficiency and overall performance.

Contributions by each team member

Code Github Repo: https://github.com/parthiisc/da_204o_Project/tree/main

Payal Dey: I contributed to the project by integrating deep learning models to support irrigation and crop health decision-making. I developed and trained multiple CNN-based architectures—including a custom CNN, VGG-19, and ResNet-50—to predict soil moisture levels and perform plant disease classification. I compared their performances using accuracy, loss trends, and confidence scores to identify the most reliable model for real-world deployment. I built a Streamlit application to deliver these insights through an intuitive interface and integrated LLM-based reasoning modules to generate context-aware irrigation and crop recommendations.

Patel Parth Pravinkumar: I developed a decision-support system that integrates machine learning with LLM-based reasoning to improve irrigation planning. I trained an XGBoost model that accurately predicts soil moisture trends using temporal and geographic features. I also built a Streamlit application and designed the LLM-powered crop recommendation module, making the system practical and scalable for farmers. Additionally, I demonstrated that the methodology can be expanded to other agricultural decision-making tasks by incorporating weather, satellite, and IoT data.

Inderjit Singh Chauhan: I contributed to the project by developing a deep-learning–based preprocessing pipeline designed specifically for IoT agricultural image uploads. Since images captured by farmers using mobile cameras or low-cost IoT sensors often contain noise, blur, lighting distortions, compression artifacts, and environmental interference (e.g., dust, shadows, moisture), I implemented a CNN-powered denoising module capable of removing these residual distortions before classification.
I designed and trained a series of convolutional models that use multi-stage filtering operations—such as Gaussian smoothing, bilateral filtering, median filtering, and learned convolutional feature extraction—to automatically enhance image quality.