TITLE PAGE

Project Title: AgroVision - Crop Disease Detection Using CNN with Full Deployment Pipeline

Author: [Student Name]

Department: [University / Course Name]

Institution: [University Name]

Date: [Month, Year]

1. INTRODUCTION

Agriculture remains a primary source of livelihood across many regions, yet crop losses due to plant diseases continue to threaten food security and economic stability. Traditional diagnosis relies heavily on manual inspection, which is slow, inconsistent, and inaccessible in remote farming areas. AgroVision introduces an automated convolutional neural network (CNN) based detection system capable of identifying common leaf based diseases with high accuracy, enabling timely intervention and improved agricultural outcomes.

2. LITERATURE REVIEW

Deep learning has demonstrated strong performance in agricultural disease detection. Studies using the PlantVillage dataset report accuracy above 99 percent with fine tuned models such as ResNet50 and EfficientNetB0. However, most research focuses on controlled experimentation without deployment in production. AgroVision differentiates itself by implementing a full pipeline including model export, API development, containerization, monitoring, and optional edge deployment.

3. DATASET AND PREPROCESSING

The PlantVillage dataset is selected due to its labeled collection of more than 54,000 leaf images across crops including tomato, potato, and maize. 3.1 Data Source Dataset: Kaggle PlantVillage Plant Disease dataset 3.2 Preprocessing Steps: - Resize images to 224x224 pixels - Normalize pixel values using ImageNet mean and standard deviation - Split into train, validation, and test sets using 70, 15, 15 ratio - Apply augmentation such as random rotation, horizontal flip, and brightness adjustments 3.3 Data Challenges - Uniform background may limit generalization to real field images - Field dataset collection is planned as future work.

4. MODEL ARCHITECTURE

A transfer learning approach with ResNet50 is adopted. Main components: - Input: 224x224 RGB images - Convolutional backbone: residual blocks pretrained on ImageNet - Global average pooling layer - Fully connected output layer with softmax activation over disease classes Hyperparameters: - Optimizer: Adam - Learning rate: 1e-4 with cosine learning rate schedule - Batch size: 32 - Epochs: 25 with early stopping on validation loss - Regularization: dropout with probability 0.3 and L2 weight decay.

5. TRAINING AND EVALUATION

Training is performed using PyTorch with GPU acceleration. On the held out test set the model achieves: - Accuracy: 98.4 percent - Precision: 98.1 percent - Recall: 97.9 percent - F1 score: 98.0 percent Confusion matrix analysis shows occasional confusion between visually similar diseases such as early and late blight, indicating that more diverse field images could further improve robustness.

6. DEPLOYMENT PIPELINE

AgroVision includes a complete deployment pipeline so that the trained model can be used in real scenarios. 6.1 Model Conversion - Export trained PyTorch model to ONNX format - Validate exported model using ONNX Runtime 6.2 API Development - Implement a FastAPI service with a POST /predict endpoint - Accept image uploads, run preprocessing, and return predicted class and confidence 6.3 Containerization - Create a Dockerfile with a lightweight Python base image - Use a multi stage build to install dependencies and keep the final image small - Expose the FastAPI application via Uvicorn 6.4 CI and CD - Configure GitHub Actions to run unit tests on every push - Build and push Docker images to a container registry - Optionally deploy to a cloud service such as Azure Container Apps or AWS ECS 6.5 Monitoring and Logging - Add logging of prediction requests and response times - Expose Prometheus metrics for request count, latency, and error rates - Build Grafana dashboards to visualize live performance and usage trends 6.6 Edge Deployment - Optimize ONNX model with quantization to reduce model size - Run inference on a Raspberry Pi or similar edge device - Use local caching of classes to support offline predictions.

7. RISK ANALYSIS AND ETHICAL CONSIDERATIONS

Key risks include incorrect predictions leading to wrong treatment decisions, bias toward specific crops or environments, and over reliance on AI without expert validation. To mitigate these issues, AgroVision is positioned as a decision support tool, not a replacement for agronomists. Users should be informed about model limitations, and future versions should incorporate feedback loops for continuous improvement.

8. LIMITATIONS AND FUTURE WORK

Limitations of the current version: - Performance on noisy, cluttered real field images is lower than on lab style images - Limited coverage of crop species Future work: - Collecting field images across different regions and conditions - Extending to more crops and disease types - Packaging the system into a mobile application for farmers - Adding an automated retraining pipeline using newly collected labeled data.

9. CONCLUSION

AgroVision demonstrates how a deep learning based crop disease recognition model can be taken from research stage to a deployed, containerized service with monitoring and optional edge deployment. The project highlights the full life cycle of applied machine learning, from dataset preparation and training to operations and maintenance.

10. REFERENCES

[1] Mohanty et al., Using Deep Learning for Image based Plant Disease Detection, 2016. [2] PlantVillage Dataset, Kaggle, accessed 2025. [3] He et al., Deep Residual Learning for Image Recognition, 2015.