1. Project Overview

The **Fall Armyworm (Spodoptera frugiperda)** is a destructive agricultural pest that causes severe yield losses in maize and other crops. This project focuses on building an **AI-powered image detection model** to automatically identify **fall armyworm presence and damage** using computer vision techniques.

By combining **YOLOv8 object detection** with a **Streamlit web interface**, this system enables real-time pest detection from uploaded crop images, helping farmers and agricultural experts respond early and effectively.

2. Objectives

- Detect and classify fall armyworm and related symptoms (larvae, eggs, frass, leaf damage).
- Train and evaluate a YOLOv8 model using labeled agricultural images.
- Deploy an interactive web app using Streamlit for easy use.
- Contribute to precision agriculture and pest management solutions.

3. Dataset

Source:
Dataset was obtained from **Roboflow**, labeled under *Fall Armyworm dataset*.
It includes images categorized into:

- fall-armyworm-egg
- fall-armyworm-larva

- fall-armyworm-frass
- fall-armyworm-larval-damage
- healthy-maize
- maize-streak-disease

Dataset Composition:

- Training images: ~7,400
- Validation images: ~1,200
- Testing images: ~500

Format: YOLOv8 (.yaml structure with bounding box annotations)

4. Methodology

Step 1: Data Collection and Preparation

- Dataset imported from Roboflow using API key.
- Data split into train, valid, and test directories.
- Preprocessing: resizing, normalization, and augmentation handled via YOLO pipeline.

Step 2: Model Training

Framework: **YOLOv8n** (Ultralytics)
Environment: **Google Colab (GPU runtime)**
Epochs: 30
Batch size: 8

Training Command:

```
!yolo task=detect mode=train model=yolov8n.pt
data=/content/Fall-Armyworm-1/data.yaml epochs=30 imgsz=640
```

Step 3: Model Evaluation

Metrics achieved after training:

- **Precision:** 0.74
- **Recall:** 0.77
- **mAP50:** 0.76
- **mAP50-95:** 0.76

These results indicate that the model is capable of identifying multiple classes of armyworm presence with good accuracy.

Step 4: Model Deployment

- Exported trained weights (`best.pt`) from Colab.
- Developed a **Streamlit application** for interactive use.
- Deployed the app via **Streamlit Cloud**.

Key Libraries Used:

```
streamlit
ultralytics
opencv-python
numpy
pillow
```

Streamlit Demo Command:

```
streamlit run app.py
```

5. Streamlit App Description

Features:

- Upload any maize image (.jpg, .png)
- Model predicts and displays bounding boxes
- Real-time visualization of detected pest damage
- Simple interface for non-technical users

app.py snippet:

```
import streamlit as st
from ultralytics import YOLO
from PIL import Image

model = YOLO("best.pt")
st.title("Fall Armyworm Detection App")

uploaded_file = st.file_uploader("Upload an image", type=["jpg", "png"])

if uploaded_file:
    img = Image.open(uploaded_file)
    results = model(img)
    st.image(results[0].plot(), caption="Detection Results",
use_column_width=True)
```

6. Results and Discussion

Metric	**Score**
Precision	0.742
Recall	0.769
mAP50	0.767

mAP50-95	0.764

- The model performs well on detecting larvae and larval damage.
- Misclassifications occasionally occur between “healthy-maize” and “mildly infected” leaves — likely due to visual similarity.

Visual Output Example:
Detected classes are drawn on the image with confidence scores:

```
fall-armyworm-larva: 0.92
fall-armyworm-damage: 0.88
```

7. Challenges Faced

- Difficulty accessing GPU runtime in Colab for long training sessions.
- Streamlit deployment required dependency fixes (libGL.so.1, ultralytics, OpenCV).
- ngrok and localtunnel instability during early testing.
- Dataset imbalance among certain categories.

8. Conclusion

This project successfully demonstrates an **AI-based fall armyworm detection system** that can identify pest presence and crop damage with high accuracy.
The model can assist farmers and agricultural bodies to quickly detect infestations, thereby improving yield and reducing chemical misuse.

Next Steps:

- Train on more diverse datasets for generalization.
- Integrate with mobile camera systems or drones.
- Add multilingual interface for broader usability.

9. References

- Ultralytics YOLOv8 Documentation: https://docs.ultralytics.com
- Roboflow Dataset Portal: https://roboflow.com
- Streamlit Documentation: https://docs.streamlit.io
- Capstone Project Guide (AI Bootcamp, 2025)