Onion Harvester Vision System Documentation

Ajit Mukund Joshi

August 24, 2025

1 Project Overview

Project Name: Onion Harvester Vision System (TIFAN 2K22)
Developer: Ajit Mukund Joshi
Version: 1.0
Description: An AI-based vision system for detecting onions in automated harvesting, contributing to 2nd runner-up in TIFAN 2K22.

2 Aim

The aim is to develop a vision-based system for detecting onions in fields, enabling efficient automated harvesting.

3 Summary

This project led AI innovation in a national competition:

- **Key Features:** Object detection using ResNet model.
- **Dataset:** Custom or PlantVillage-like for onions.
- **Achievement:** 2nd runner-up in TIFAN 2K22.

4 Technology Used

- **Programming Language:** Python 3.x
- **Libraries and Frameworks:**
  - PyTorch, Torchvision: For ResNet18 and transforms.
  - OpenCV: For image processing.
  - NumPy: For arrays.
- **Tools:** Jupyter Notebook.

5 Methodology

1. **Data Preparation:** Load and transform images.
2. **Model Architecture:** Pre-trained ResNet18 fine-tuned for 2 classes.
3. **Training:** Adam optimizer, CrossEntropy loss, 10 epochs.
4. **Detection:** Load model, predict on images.

6 Source Code

6.1 requirements.txt

```
torch
torchvision
opencv-python
numpy
```

6.2 train.py

```
import torch
from torch.utils.data import DataLoader
from torchvision import datasets, transforms, models

transform = transforms.Compose([transforms.Resize((224, 224)),
   transforms.ToTensor()])
dataset = datasets.ImageFolder('data/', transform=transform)
loader = DataLoader(dataset, batch_size=32, shuffle=True)

model = models.resnet18(pretrained=True)
model.fc = torch.nn.Linear(model.fc.in_features, 2)  # 2 classes:
   onion, not-onion

optimizer = torch.optim.Adam(model.parameters(), lr=0.001)
criterion = torch.nn.CrossEntropyLoss()

for epoch in range(10):
    for inputs, labels in loader:
        optimizer.zero_grad()
        outputs = model(inputs)
        loss = criterion(outputs, labels)
        loss.backward()
        optimizer.step()
    print(f"Epoch {epoch+1} complete")

torch.save(model.state_dict(), 'onion_model.pth')
```

6.3 detect.py

```
import torch
from torchvision import transforms, models
import cv2
import numpy as np
import argparse

parser = argparse.ArgumentParser()
parser.add_argument('--image', required=True)
args = parser.parse_args()

model = models.resnet18()
model.fc = torch.nn.Linear(model.fc.in_features, 2)
model.load_state_dict(torch.load('onion_model.pth'))
model.eval()

transform = transforms.Compose([transforms.Resize((224, 224)),
   transforms.ToTensor()])
img = cv2.imread(args.image)
img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
img = transform(img).unsqueeze(0)
pred = model(img)
label = 'Onion' if torch.argmax(pred) == 0 else 'Not Onion'
print(f"Detection: {label}")
```

7 Installation and Setup

1. Place dataset in `data/`.
2. Install dependencies: `pip install -r requirements.txt`.
3. Train: `python train.py`.
4. Detect: `python detect.py –image path/to/image.jpg`.

8 Testing and Validation

- **Test Cases:** Field images, varied conditions.
- **Metrics:** Detection accuracy, IoU (if extended).
- **Validation:** Competition performance.

9 Conclusion

The Onion Harvester Vision System innovates agricultural AI, securing a competition award. Future work could include YOLO for faster detection and robotic integration.

10 References

- PyTorch Documentation
- Resume: Ajit Mukund Joshi (Projects and Highlights Section)