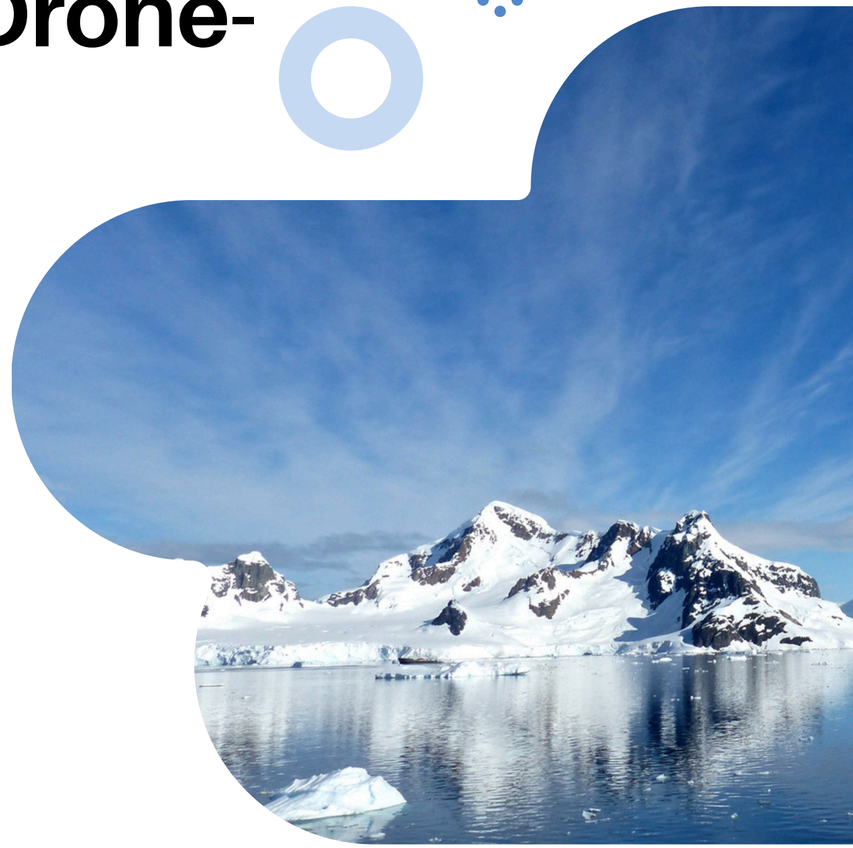
Intelligent IIoT Service Deployment using MEC and Federated Learning in Drone-Assisted Smart Agriculture



Team No. 16

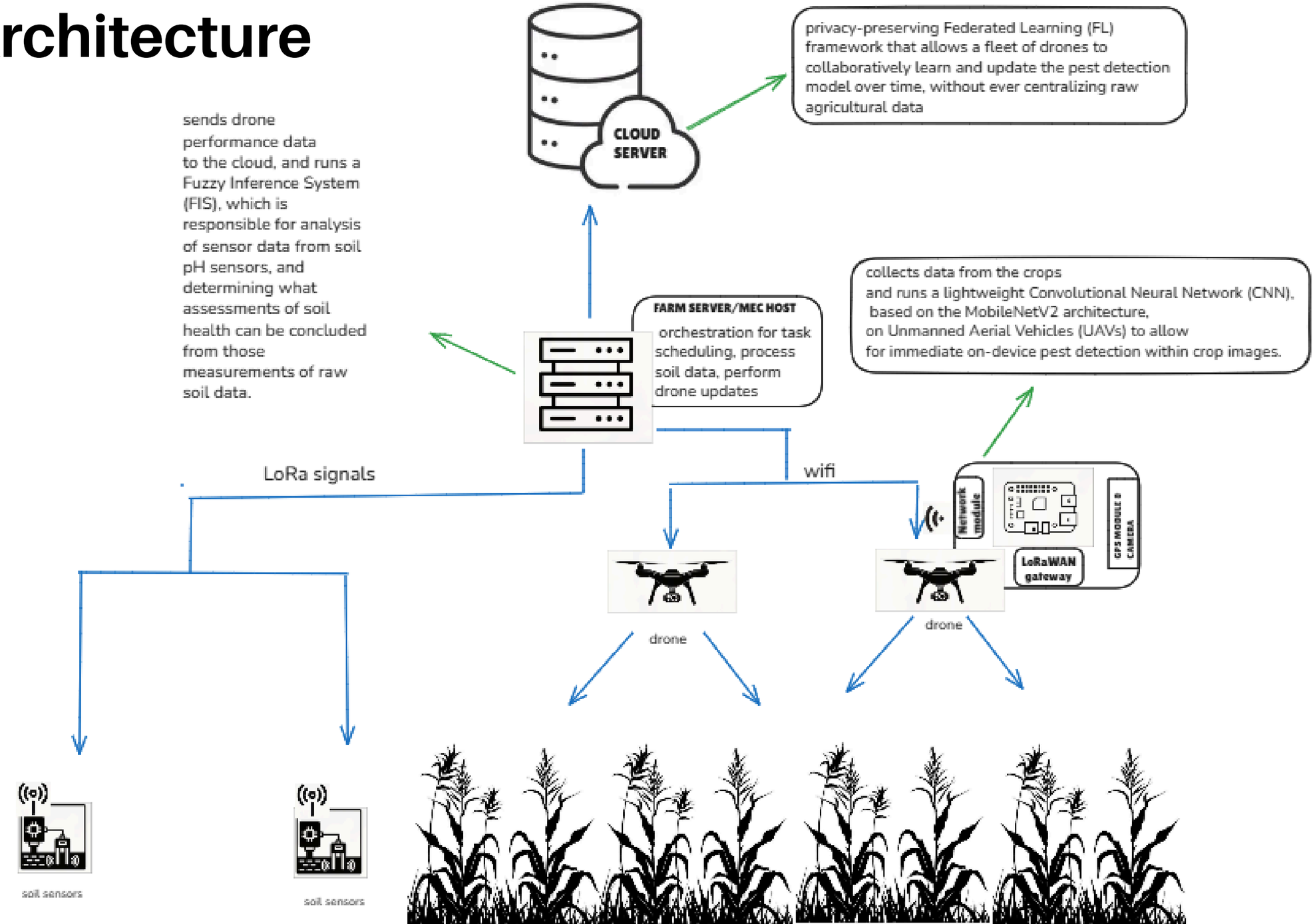
EDGLERS

PROBLEM STATEMENT

Traditional cloud-based farming systems suffer from high latency, unreliable connectivity, and limited scalability, making real-time decision-making difficult in large or remote fields.

An edge computing framework can overcome these challenges by processing data locally, enabling faster, smarter, and more responsive farm automation and pest detection.

Edge Architecture



Task Scheduling:

The MEC (Multi-Access Edge Computing) engine dynamically allocates inspection tasks to drones through an auction-based scheduler, balancing energy, distance, and mission priority.

Bid Calculation

BID = **distance**+0.3(100−**Battery**)+0.15(n**o. of inspections completed**)+1.5N(**no. of targets in mission**)+0.5(10−(**mission priority**))

- The drone with the minimum bid wins the mission, ensuring proximity, workload balance, and battery constraints are considered dynamically.

Normal Flow

Soil sensors periodically send environmental data to MEC.
MEC generates routine patrol missions using low-priority plants.
Drones bid for missions → lowest bid executes optimized route.
Inspection results update plant priorities and risk maps.

Anomaly Flow

MEC detects soil or visual anomalies (e.g., pest/disease zones).
High-priority plants are clustered → reactive missions created.
Auction re-runs immediately for urgent drone dispatch.
Anomaly data feedback refines future mission generation.

Protocols

1. Edge Layer: Data Collection and Real-Time Pest Detection

Soil Monitoring: Soil Sensors collect crucial environmental parameters (moisture, pH, nutrients). They transmit this data via LoRaWAN to the Farm Server (MEC Host).
LoRaWAN Gateway Protocol: Used for connecting soil sensors to the MEC server, leveraging low-power, long-range wireless capability for efficient field data gathering.
Reasoning: Selected for its long-range and extremely low power consumption, which is essential for battery-operated soil sensors distributed across a large field. Its low bandwidth is sufficient for transmitting small sensor data packets.

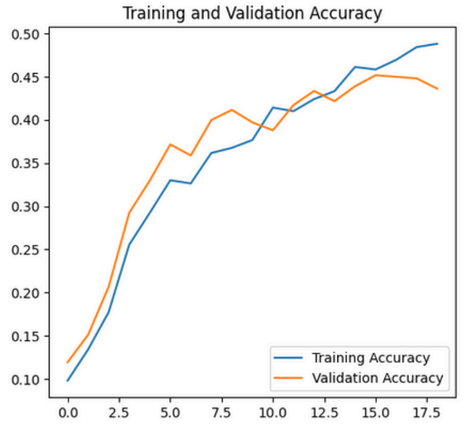
2. **Fog Layer:** Multi-access Edge Computing (MEC) - Control and Analytics

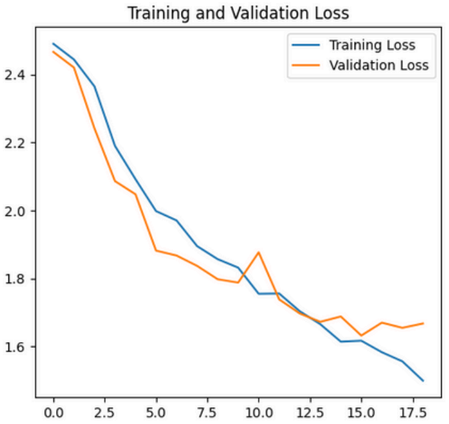
Local LTE Network: Used for drone-to-MEC server links, supporting higher data rates (e.g., image transfers) and low latency for mobile edge nodes (drones).
Reasoning: Chosen to provide the high bandwidth and low latency required for transmitting high-resolution images from drones and receiving real-time control commands from the MEC server.

CNN Model for Pest Detection:

Initial Attempt

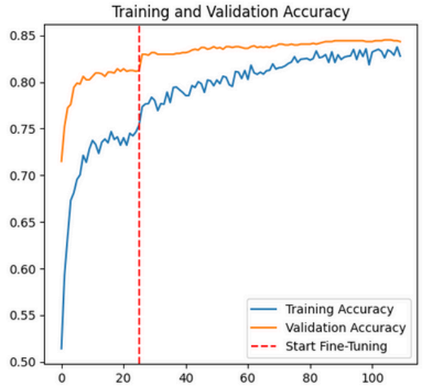
- Built CNN Model from Scratch
- Resized images (150×150×3), 80–20 split, data augmentation
- Architecture: Conv2D + MaxPooling → Dense layers → Softmax (12 pest classes)
- Trained with Adam optimizer, sparse categorical cross-entropy
- Achieved ~50% accuracy with ~60MB model (20–30 epochs, early stopping)

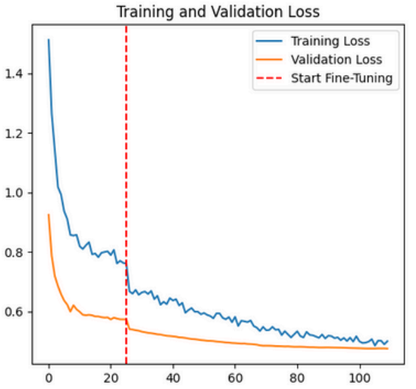




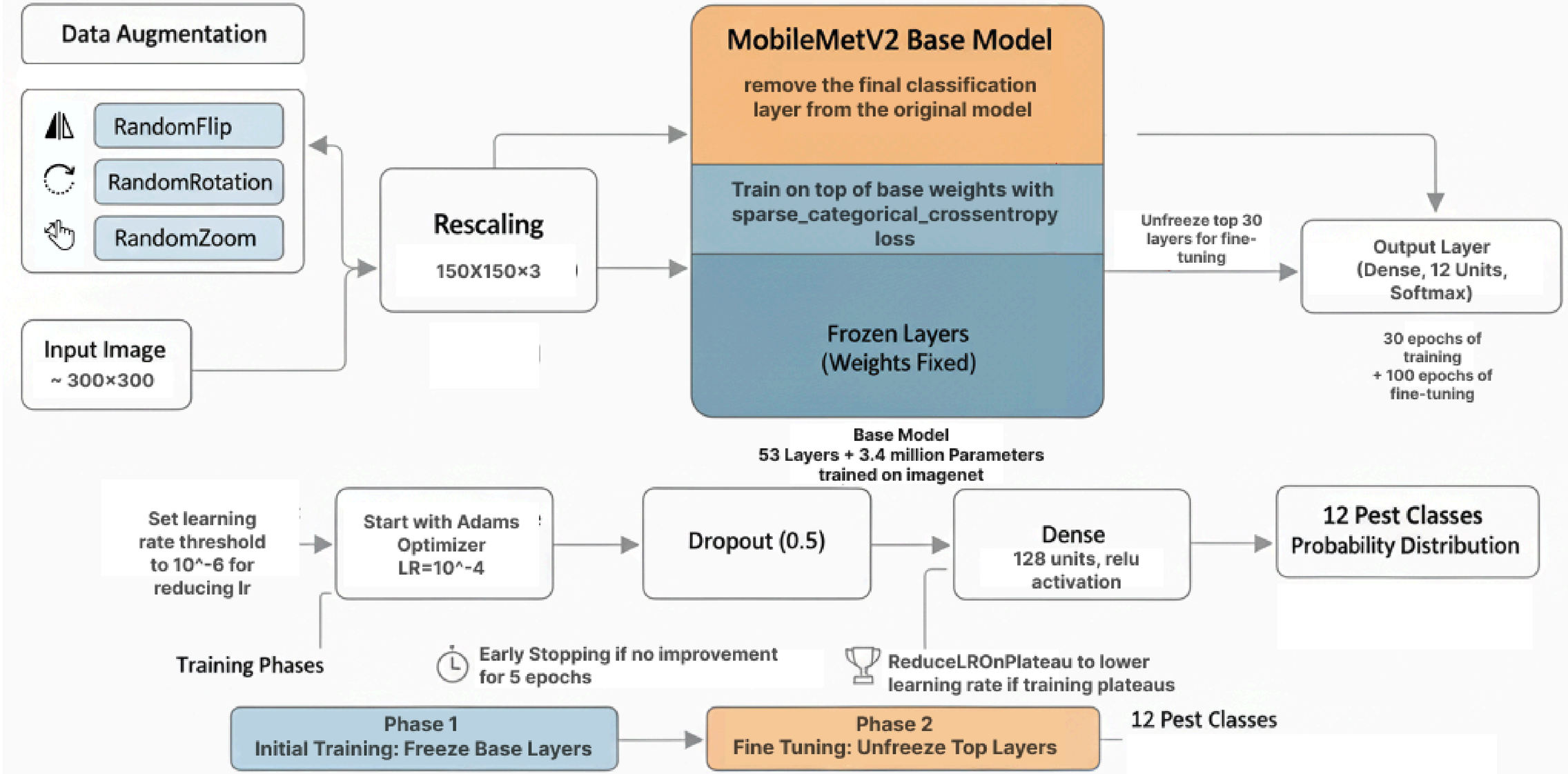
Update: Applied Transfer Learning with MobileNetV2:

- To create a pest detection model that is both accurate and efficient enough to run on a drone's resource-constrained hardware.
- Using Transfer Learning & MobileNetV2
- MobileNetV2 is a lightweight CNN designed specifically for mobile and embedded devices.
- We leverage the weights of pre-trained on the massive ImageNet dataset and fine-tune it for our specific pest classification task.
- This allows to achieve high accuracy with a relatively small dataset.





CNN Model for Pest Detection-Architecture



CNN Model for Pest Detection:

Results:

- Final Model weights only ~20MB
- Fit for running on the Drone.
- Achieves a balance between accuracy and computational cost

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 2s 2s/step
Model predictions: [[0.01018984 0.02700407 0.0893319  0.08830439 0.20445232 0.07952978
  0.10970672 0.00559862 0.35035127 0.02213354 0.0061905  0.00720698]]
Predicted class index: 8
```

CNN Model for Pest Detection:

Local Incremental Updation of the model

- **Self-Supervised Learning**: The drone uses its current model to make predictions. It randomly selects a subset of images with very high confidence (>95%) and uses them to train itself for a few epochs (~5). This allows it to adapt to its local environment.
- **Human-in-the-Loop:** For images where the model has low confidence, it sends them back to the server for human labeling. This prevents the model from reinforcing its own mistakes.

```
[starkiller@starkiller-linux fl_model]$ python drone_client.py
2025-09-18 11:02:41.603617: I tensorflow/core/util/port.cc:153] oneDNN custom operations are on. You may see slightly different numerical results due to floating-point round-off errors from different computation orders. To turn t
, set the environment variable `TF_ENABLE_ONEDNN_OPTS=0`.
2025-09-18 11:02:41.669858: I tensorflow/core/platform/cpu_feature_guard.cc:210] This TensorFlow binary is optimized to use available CPU instructions in performance-critical operations.
To enable the following instructions: SSE3 SSE4.1 SSE4.2 AVX AVX2 AVX_VNNI FMA, in other operations, rebuild TensorFlow with the appropriate compiler flags.
AttributeError: 'MessageFactory' object has no attribute 'GetPrototype'
AttributeError: 'MessageFactory' object has no attribute 'GetPrototype'
AttributeError: 'MessageFactory' object has no attribute 'GetPrototype'
AttributeError: 'MessageFactory' object has no attribute 'GetPrototype'
AttributeError: 'MessageFactory' object has no attribute 'GetPrototype'

--- Starting Local Training on drone_2744 ---
/usr/lib/python3.13/site-packages/keras/src/saving/saving_lib.py:797: UserWarning: Skipping variable loading for optimizer 'adam', because it has 2 variables whereas the saved optimizer has 74 variables.
  saveable.load_own_variables(weights_store.get(inner_path))
Successfully loaded global weights from global_model_initial.weights.h5
Simulating collection of new local data...
Found 30 files belonging to 3 classes.
Starting local fine-tuning...
Epoch 1/5
4/4 ━━━━━━━━━━━━━━━━━━━━ 13s 129ms/step - accuracy: 0.1667 - loss: 3.5472
Epoch 2/5
4/4 ━━━━━━━━━━━━━━━━━━━━ 0s 118ms/step - accuracy: 0.1333 - loss: 3.3055
Epoch 3/5
4/4 ━━━━━━━━━━━━━━━━━━━━ 0s 121ms/step - accuracy: 0.1667 - loss: 2.8302
Epoch 4/5
4/4 ━━━━━━━━━━━━━━━━━━━━ 0s 116ms/step - accuracy: 0.2000 - loss: 3.0405
Epoch 5/5
4/4 ━━━━━━━━━━━━━━━━━━━━ 0s 118ms/step - accuracy: 0.0667 - loss: 3.0590
Local fine-tuning complete.
Saved updated weights to drone_2744_update_1758173583.weights.h5
```

Federated Learning Framework:

Built a Federated Learning system that trains a new neural network taking weights from CNN models from drone fleet, improving the model's pest detection accuracy.

Collaborative Learning with Federated Learning (FL)

The main objective of this framework is to continuously improve the model across the entire drone fleet without centralizing sensitive image data.

Used Federated Averaging (FedAvg) as the Core Algorithm.

The FL Cycle:

- Distribute: The central server sends the current global model to all drones.
- Local Training: Each drone fine-tunes the model on new data it collects in the field.
- Aggregate: Drones send only their updated model weights (not images) back to the server.
- Update & Repeat: The server averages the weights to create an improved global model, which is then redistributed.

- Self-Supervised Learning: Drones learn from their own high-confidence (>95%) predictions, adapting to their local environment.
- Human-in-the-Loop: Low-confidence images are flagged for human review. This provides a " improvised check " and prevents the model from reinforcing its own errors.

Key Result: Accuracy increase from 80% to 95% on test dataset, proving the effectiveness of this approach.

Soil QualityAssessment with Fuzzy Logic

For the achieving the goal of translating numerical soil pH data into intuitive, qualitative assessments (e.g., "Good," "Moderate," "Poor").

Why choose Fuzzy ?

Fuzzy excels at handling real-world ambiguity. The transition from "acidic" to "optimal" is gradual, not a hard cutoff, and fuzzy logic models this human-like reasoning.

Working of FIS:

- 1. Fuzzification: The system takes a crisp input (e.g., pH 6.8) and determines its degree of membership in predefined linguistic sets like Acidic or Optimal using membership functions.
- 2. Inference Engine: A simple rule base, containing expert knowledge, is evaluated (e.g., IF Soil_pH is Optimal THEN Soil_Quality is Good).
- 3. Defuzzification: The results from all activated rules are combined and translated back into a single, crisp output score.

- Key Result: The system provides nuanced assessments, correctly identifying optimal (pH 6.8 -> score 88.2), borderline (pH 6.1 -> score 62.5), and poor (pH 4.5 -> score 15.0) conditions.

RESULT:

```
===  SOIL SENSOR DATA (Sparse Zone Monitoring) ===
Total sensor readings: 208
Active sensors: 16
Soil anomalies detected: 9 (4.3%)
Zones with soil anomalies: [np.int64(100), np.int64(102), np.int64(106), np.int64(109), np.int64(110), np.int64(111)]

===  MEC SERVER INTELLIGENCE ===
Sensor data processed: 208 packets
Average processing latency: 0.9 time units
Anomaly zones tracked: 9

=== DRONE VISUAL INSPECTIONS (Onboard AI) ===
Total plant inspections: 725
Unique plants inspected: 293
Visual anomalies detected: 98 (13.5%)
```

```
 Drone Performance:
  Drone 50: 82 inspections, 10 anomalies (12.2%)
  Drone 51: 271 inspections, 31 anomalies (11.4%)
  Drone 52: 141 inspections, 30 anomalies (21.3%)
  Drone 53: 231 inspections, 27 anomalies (11.7%)

===  MISSION GENERATION (Multi-Source Intelligence) ===
Total missions generated: 40

Mission Types:
  • PATROL: 15 missions (avg priority: 3.6, avg targets: 24.3)
  • REACTIVE: 13 missions (avg priority: 35.7, avg targets: 14.8)
  • ZONE_INVESTIGATION: 12 missions (avg priority: 5.5, avg targets: 14.0)

Missions completed: 27
```

```
=== AUCTION-BASED TASK ALLOCATION ===
Total auctions: 40
Average winning bid: 93.37
Average competition: 2.9 bidders/auction

Winning Drones:
  Drone 50: 8 missions won (reactive: 3, patrol: 2)
  Drone 51: 11 missions won (reactive: 2, patrol: 5)
  Drone 52: 10 missions won (reactive: 5, patrol: 3)
  Drone 53: 11 missions won (reactive: 3, patrol: 5)

=== DRONE FLEET PERFORMANCE ===
Total inspections: 725
Total anomalies detected: 98
Total distance traveled: 8982.28 units
Average distance per drone: 2245.57 units
Load balancing efficiency: 52.8%

=== PLANT MONITORING COVERAGE ===
Total plants: 300
Plants inspected: 293 (97.7%)
Plants with detected issues: 76 (25.3%)
High-risk plants: 292
Average inspections per plant: 2.42

===  ZONE-BASED ANALYSIS (Soil Sensor Zones) ===
Total zones: 16
Average plants per zone: 18.8
Average zone coverage: 82.9%
```

Data Sources:
- Soil sensors: 16 sparse sensors
- Visual inspections: 293 plants scanned
- Sensor:Plant ratio: 1:18

Anomaly Detection:
- Soil-based anomalies: 9 zones
- Vision-based anomalies: 98 plants
- Detection rate: 13.5% of inspected plants

Task Scheduling:
- Auction efficiency: 2.9 avg bidders
- Mission completion: 67%
- Reactive missions: 13

Coverage & Efficiency:
- Plant inspection coverage: 97.7%
- Total flight distance: 8982.28 units
- Inspections per distance unit: 0.08