PROJECT REPORT:

AI-POWERED CONVEYOR MODEL FOR INDUSTRIAL WASTE CLASSIFICATION

Group 6

Supporter	**Phạm Minh Tâm**
Member	**Phan Ngọc Anh Thư**
Member	**Nguyễn Thiên Phúc**
Member	**Trương Khôi Nguyên**
Member	**Lê Doãn Dương**

I. INTRODUCTION:

1. Reason for choosing the Topic:

- The model has high practical applicability, is accessible to high school students, and possesses real-world relevance.
- The model serves as a foundation for members to develop academically while also raising awareness about waste classification.
- It is one of the classic and well-documented models, making it easy to research and implement.

2. Objectives of the Topic:

- Apply computer vision and deep learning to build an AI model capable of processing images and classifying waste into two main categories: inorganic and organic, thereby enhancing knowledge about AI.
- Apply hardware-related techniques to learn and gain a deeper understanding of hardware systems.
- Classify waste to contribute to current environmental protection efforts.
- Develop a forward-looking model that supports environmental protection in industrial applications.

II. HARDWARE:

1. ELECTRICAL CIRCUIT STRUCTURE AND COMPONENTS:

a) Components and Their Functions:

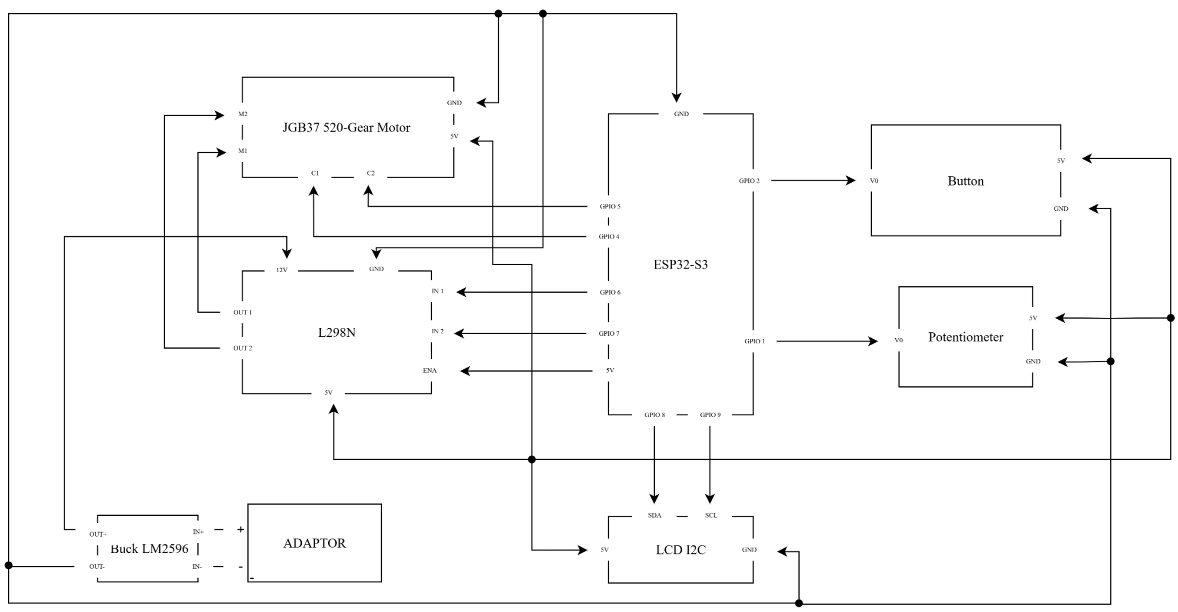
- ESP32-CAM S3:
  - Specifications:
    + Processor: Dual-core Tensilica LX7, clock speed 240MHz
    + RAM: 512KB SRAM, supports extended PSRAM
    + Supports Wi-Fi 802.11 b/g/n and Bluetooth BLE 5.0
    + Integrated OV2640/OV5640 camera
    + Multi-purpose GPIO, UART, SPI, I2C, PWM
  - Functions: Acts as the central controller, collecting data from the camera and sensors, and communicating via Wi-Fi or Bluetooth
- L298N H-Bridge Motor Driver:
  - Specifications:
    + Operating voltage: 5–35V
    + Maximum load current: 2A per channel
    + Two channels for controlling DC motors or one stepper motor
    + Built-in flyback diode protection
  - Functions: Controls the direction and speed of the DC motor
- DC-DC Buck Converter LM2596 3A:
  - Specifications:

        + Input voltage: 4–40V DC
        + Adjustable output voltage: 1.25–37V DC
        + Maximum output current: 2–3A
        + Conversion efficiency: 75–90%
    - Function: A buck converter that reduces voltage while increasing output current, supplying higher current at lower voltage
- DC Motor JGB37-520 with Gearbox and Encoder:
    - Specifications:
        + Operating voltage: 6V – 12V (rated 12V)
        + No-load current: ~70–100 mA
        + Maximum load current: ~1.5 – 2A
        + Torque: ~5 – 15 kg·cm
    - Functions: Converts electrical energy into mechanical energy, generating rotational motion to operate the conveyor belt
- LCD Display with I2C Interface:
    - Specifications:
        + Voltage: 5V
        + Interface: I2C (2-wire SDA/SCL)
        + Display type: 16x2 or 20x4 characters
        + Default I2C address: 0x27 or 0x3F
    - Function: Displays system information such as motor speed, operation mode, potentiometer values, etc.
- Infrared Obstacle Avoidance Sensor (IR Sensor):
    - Specifications:
        + Operating voltage: 3.3–5V
        + Detection range: 2–30 cm (adjustable)
        + Interface: Digital Output (0/1)
    - Function: Detects nearby obstacles to prevent collisions during conveyor operation.
- Rotary Potentiometer:
    - Specifications:
        + Resistance: Typically 10KΩ
        + Type: Manual adjustable, 3 pins
        + Output signal: Analog (0–3.3V or 0–5V)
    - Function: Adjusts the motor speed by varying the analog signal
- Servo Motor SG90:
    - Specifications:
        + Operating voltage: 4.8–6V
        + Torque: ~1.8 kg·cm
        + Rotation angle: 0–180°
        + Signal type: PWM
    - Function: Pushes inorganic waste into a designated collection bin
- Push Button:
    - Specifications:
        + Type: Normally Open (NO), 4 pins
        + Operating voltage: 3.3V or 5V
        + Signal type: Digital (0 or 1)

- Functions: Switches between conveyor belt operation modes, displayed on the LCD screen.

- Power Adapter:
    - Specifications:
        + Input voltage: 220V AC
        + Output voltage: 5V, 9V, or 12V DC
        + Output current: 1A–5A depending on type
        + Connector standard: DC 5.5x2.1mm
    - Function: Provides power supply for the L298N module
- Ceramic Capacitor:
    - Specifications:
        + Typical value: 0.1µF – 100nF
        + Voltage rating: 50V or higher
        + Type: Non-polarized
    - Function: Helps filter noise and stabilize the signal when pressing the button.

b) ELECTRICAL CIRCUIT DIAGRAM



2. CONVEYOR SYSTEM ALGORITHM:

a) PID Algorithm:

- Purpose: Control the motor speed so that the actual speed (from the encoder) approaches the preset speed (from the potentiometer).
- Operation:
    + The **input variable** receives the actual speed value currentRPM
    + The **setpoint variable** receives the target speed targetRPM.

+ The function myPID.Compute() performs calculations using the PID controller.
+ The **output value** is used to generate PWM signals for the motor (via pwmOut(output)).

- Characteristics:
  + Closed-loop control system (feedback loop).
  + PID output is limited from -255 to 255 to match the 8-bit PWM range.
  + PID updates every 100 ms (SetSampleTime(100)).

b) RPM Calculation Algorithm from Encoder:

- Purpose: Determine the actual rotational speed of the motor to provide input for the PID controller.
- Operation:
  + Use attachInterrupt() to count pulses from the encoder (variable encoderPos).
  + Every 500 ms in the loop(), the system calculates the change in encoder position encoderDelta.
  + Then compute: currentRPM = (encoderDelta / PPR) * (60 / deltaTime)

c) ADC Signal Filtering Algorithm from Potentiometer:

- Purpose: Stabilize the signal read from the potentiometer to avoid RPM fluctuations caused by noise.
- Operation:
  + Read the analog value using analogRead(POTENTIOMETER) in each loop cycle.

d) PWM Output Algorithm for Motor Control:

- Purpose: Convert the PID output into PWM pulses to control the motor's direction and speed.
-

III. SOFTWARE:

1. Datasets:

- Datasets: 6 Kaggle datasets (moltean/fruits, mostafaabla/garbage-classification, utkarshsaxenadn/fruits-classification, sriramr/fruits-fresh-and-rotten-for-classification, chrisfilo/fruit-recognition, kritikseth/fruit-and-vegetable-image-recognition)
- Labels: Organic (1) = apple, orange, lime, lemon, banana; Inorganic (0) = glass, plastic, metal, trash, battery
- Image size: 224×224; Batch size: 32; Validation split: 20%

2. Methodology:

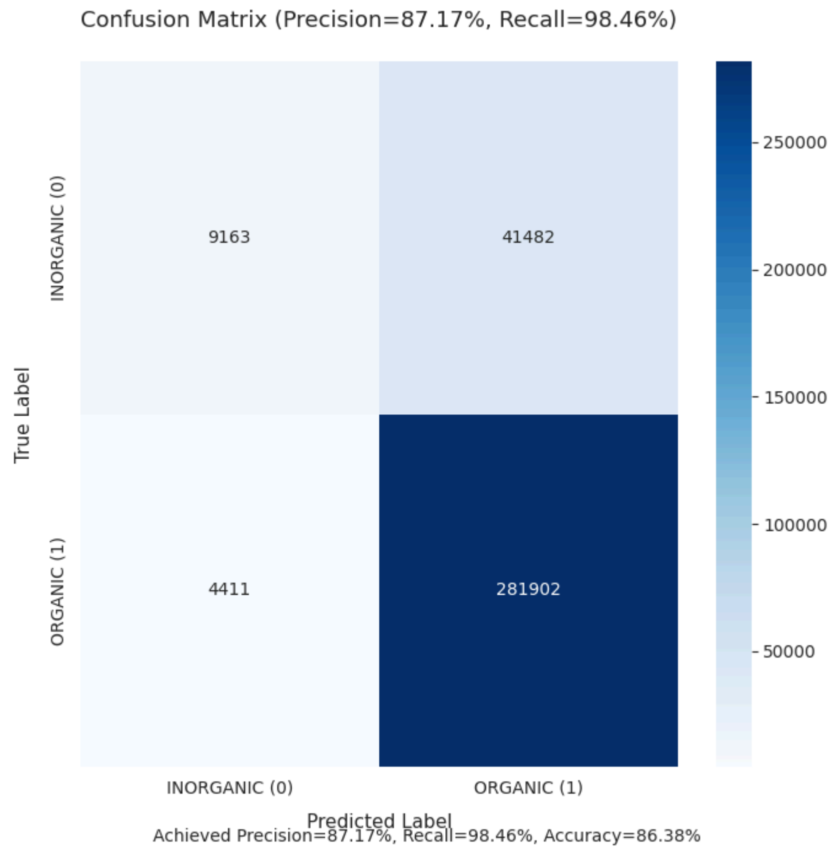
- Transfer Learning with EfficientNetB0:

+ Leverage EfficientNetB0 pretrained on ImageNet to retain general visual features; perform controlled fine-tuning for waste classification.

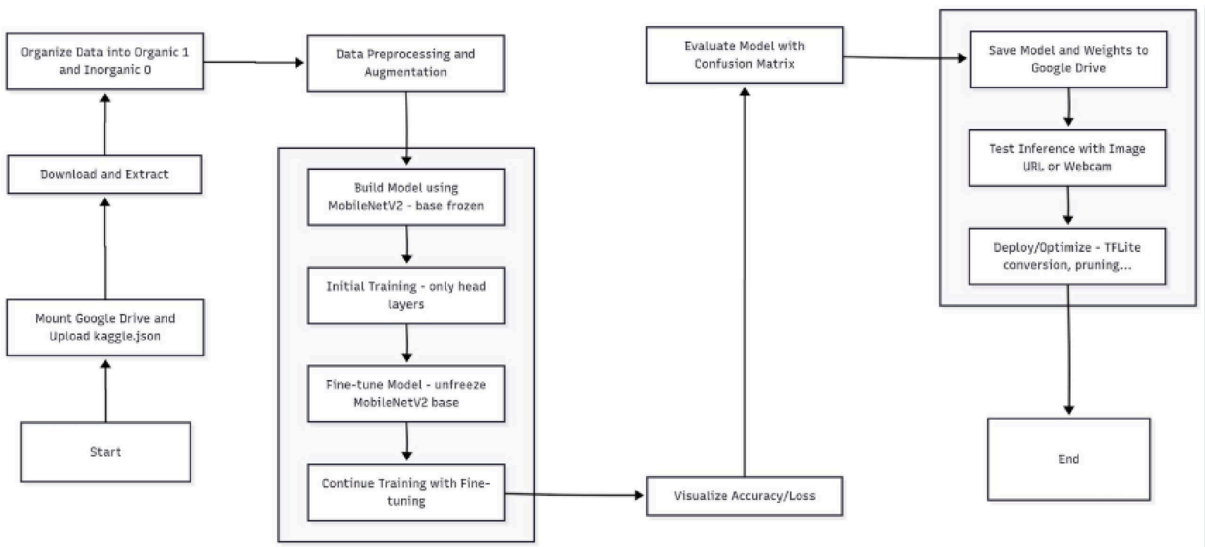
- Data Augmentation
  + Rotate ±25°, translate ±15%, shear ±15%, zoom ±15%, horizontal flip, brightness 0.85–1.15, channel shift 0.1.
  + Goal: increase data diversity, reduce overfitting, improve generalization.
- Anti-overfitting Head Architecture
  + GlobalAveragePooling2D → Dropout(0.5) → Dense(128, ReLU, L2=0.01) → BatchNorm → Dropout(0.3) → Dense(1, sigmoid).
  + Rationale: fewer parameters + stronger regularization (Dropout + L2).
- Optimizer – Adam with Gradient Clipping
  + Adam with clipnorm=1.0 for stable updates and to prevent gradient explosion.
- Scheduler – Cosine Decay (+ ReduceLROnPlateau)
  + CosineDecay (alpha=0.1) learning rate schedule; ReduceLROnPlateau lowers LR when val_accuracy plateaus.
  + Benefit: smoother convergence and escape from local minima.
- Loss Function – Binary Crossentropy
  + Metrics: accuracy, precision, recall.
  + Labels: Inorganic (0), Organic (1).
- EarlyStopping (patience=7, restore_best_weights=True)
  + Stops early on no improvement to avoid overfitting and save resources.
- Controlled Fine-Tuning (2 stages)
  + Stage 1: freeze backbone, train head on full dataset (no step limits).
  + Stage 2: unfreeze ~last 50 layers of EfficientNetB0, use low LR + clipping to refine features.
- Overfitting Monitoring
  + Custom OverfittingMonitor callback warns when train–val gap > 3%; tight per-epoch monitoring.
- Training Configuration
  + Image size 224×224, batch size 32, validation split 20%, use full training set each epoch.
  + Save best checkpoint by val_accuracy; visualize and evaluate with confusion matrix and classification report.

3. **Results:**

- Confusion Matrix:

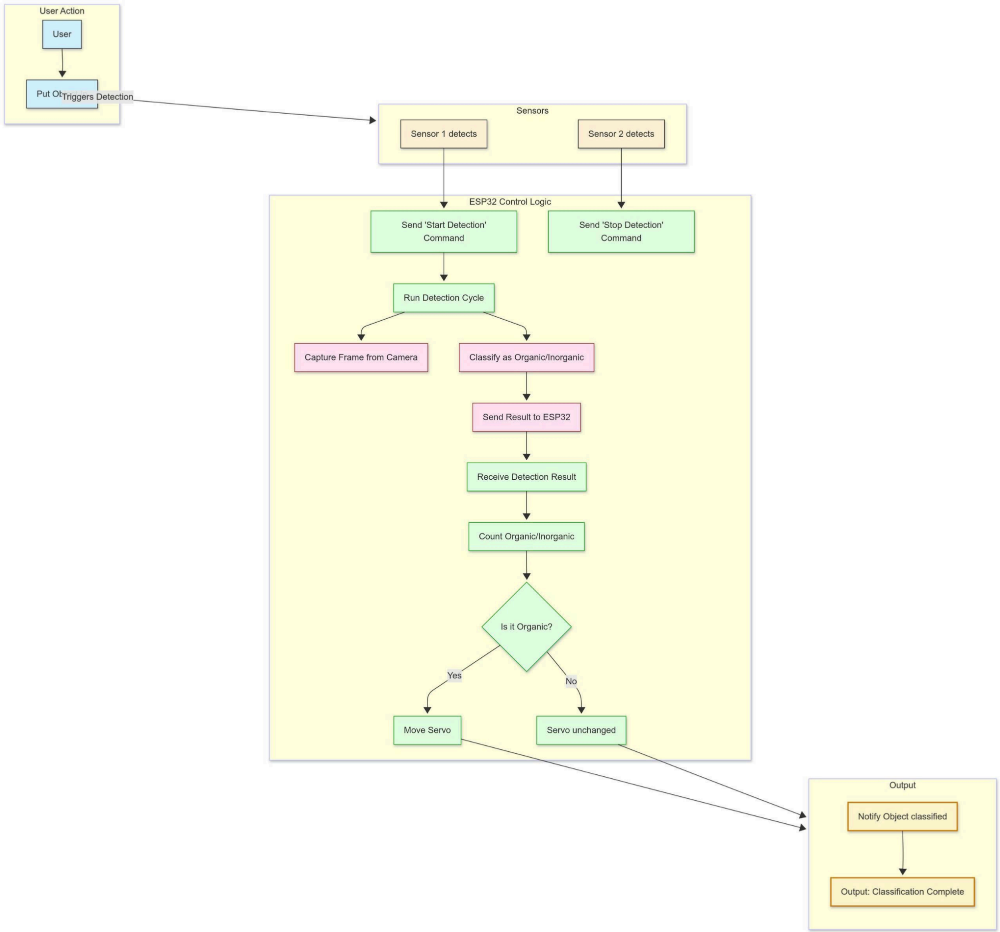


- Evaluate: Good at processing data, achieving a precision of 87.17% and recall of 98.46%
- Algorithm Flowchart:



4. Processing on the Conveyor Belt

- Algorithm Flowchart and functional diagram of the model using an external camera



IV. CONCLUSION:

1. Current Limitations:

- The AI training dataset is limited, and there are insufficient resources and infrastructure to train AI models with higher performance.
- The system can only analyze single objects on the conveyor belt; high-speed processing may cause data noise.
- There are not enough resources or funding to invest in materials such as cameras, etc.

2. Future Research Directions:

- **Sensors**: Replace IR with ToF sensors for more accurate detection and add auxiliary cameras to handle occluded objects.
- **AI Classification**: Upgrade to EfficientNetB0 or MobileViT to support more waste categories and enable learning from real-world data.
- **Cloud Connectivity**: Send results, images, and logs to Firebase or Google Sheets for online monitoring and alerts.
- **Automation**: Support tracking multiple objects, adjusting conveyor belt speed, and automatically stopping when predictions are uncertain.

- **Resources and Infrastructure**: Seek infrastructure development and additional investment to improve model accuracy and industrialize the system.

V. Task Assignment and Evaluation of Members' Performance

Member	Task	Completion level
Trương Khôi Nguyên	Construct the physical frame of the conveyor and design the circuit diagrams.	100%
Phan Ngọc Anh Thư	Develop the control program and assemble the electrical circuits.	100%
Lê Doãn Dương	Apply the model to operation on the conveyor belt; train AI model.	100%
Nguyễn Thiên Phúc	Train the AI model and handle final processing tasks.	100%