

Instructivo de instalación y configuración de la Inteligencia Artificial

Este instructivo describe el desarrollo y funcionamiento del módulo de inteligencia artificial (IA) implementado en el sistema. Incluye la predicción automática de variables ambientales y la detección de objetos en imágenes mediante visión por computador.

Requisitos previos

- Python 3.10 o superior
- MongoDB 4.x o superior
- Broker MQTT (Mosquitto)

Dependencias Python:

```
pip install numpy pandas scikit-learn ultralytics opencv-python pillow paho-mqtt pymongo
```

- Archivo del modelo YOLOv8 ([yolov8n.pt](#)), que puede descargarse automáticamente al usar [ultralytics](#).

```
from ultralytics import YOLO
import cv2

modelo_yolo = YOLO("yolov8n.pt")
```

Predicción de temperatura y humedad

Se implementa un modelo de regresión lineal simple para predecir el valor futuro de las variables ambientales (temperatura y humedad). Los datos provienen de sensores conectados a dispositivos IoT y se almacenan en MongoDB.

Cada 5 minutos, un hilo en segundo plano accede a los datos históricos, entrena el modelo y realiza la predicción. Los resultados se publican automáticamente por MQTT a los siguientes topics:

```
TOPIC_DETECCION = "deteccion/personas"
TOPIC_TEMP_PRED = "iot/temperatura/prediccion"
TOPIC_HUM_PRED = "iot/humedad/prediccion"
```

Las funciones encargadas de esta lógica se encuentran en el archivo `ai_model.py`, y la automatización se ejecuta desde `mqtt_listener.py`.

Ejemplo de publicación:

```
{
  "datos_utilizados": 198,
  "mensaje": "Predicción completada",
  "prediccion_temperatura": 21.92
}
```

Detección de objetos con YOLOv8

Se utiliza el modelo preentrenado YOLOv8 para detectar personas y objetos en imágenes recibidas desde un ESP32 mediante MQTT.

Cuando se recibe una imagen en base64 en el topic `camara/foto`, el sistema:

1. La guarda en MongoDB usando GridFS.
2. La convierte a un array NumPy.
3. La analiza con YOLOv8.
4. Publica los objetos detectados en el topic `deteccion/personas`.

Toda esta lógica está implementada en `mqtt_listener.py`, y la función de detección se encuentra en `ai_model.py`.

Ejemplo de resultado publicado:

```
{
  "mensaje": "Detección automática desde MQTT",
  "objetos_detectados": ["person", "bottle"],
  "file_id": "6655cc08a6e4b45c74e5bc27",
}
```

```
"fecha_deteccion": "2025-05-28T17:50:01.000Z"
}
```

Automatización

El archivo `mqtt_listener.py` coordina la funcionalidad automática de la IA. Se suscribe a los topics de sensores e imágenes, guarda la información en la base de datos, realiza inferencias y publica los resultados por MQTT sin intervención manual.

La función `prediccion_automatica_loop()` corre en un hilo paralelo y se encarga de ejecutar las predicciones cada 5 minutos.

Para iniciar el sistema:

```
(venv) PS C:\Users\andre\invernadero> python mqtt_listener.py
```






Verificación

Para verificar que la IA está funcionando correctamente, se puede suscribir a los topics

```
(venv) PS C:\Users\andre\invernadero> python mqtt_listener.py
mosquitto_sub -h <IP_BROKER> -t iot/humedad/prediccion
mosquitto_sub -h <IP_BROKER> -t iot/humedad/prediccion
mosquitto_sub -h <IP_BROKER> -t deteccion/personas
```

También se puede comprobar la inserción de datos e imágenes en MongoDB (colecciones `Temperatura`, `Humedad`, y `fs.files`).

Archivos relevantes

Archivo	Descripción
 <code>ai_model.py</code> 	Lógica de entrenamiento, predicción y detección con YOLOv8
 <code>mqtt_listener.py</code>	Subscripción MQTT, automatización de IA y publicación de resultados
 <code>routes.py</code> 	Endpoints REST para pruebas manuales

Consideraciones finales

- El modelo de predicción requiere mínimo tres datos para funcionar.
- Las imágenes deben ser válidas y llegar correctamente codificadas en base64.
- La detección y la predicción están completamente automatizadas y no dependen de llamadas manuales.