

Proyecto Final

Sistemas Operativos y Laboratorio

Título del proyecto

Tiny ML para un mecanismo de detección de objetos.

Miembros del equipo

Oswald Daniel Gutierrez
Santiago Rivera Montoya
Leider Felipe Caicedo

Resumen

Este proyecto tiene como objetivo implementar modelos de aprendizaje automático como TinyML en dispositivos con recursos limitados, como microcontroladores, para procesar datos en tiempo real de manera local. Esto es importante en el desarrollo de dispositivos como portátiles, de monitoreo industrial y hogares inteligentes, etc. La implementación de Tiny permite ejecutar IA en estos dispositivos, ofreciendo tiempos de respuesta más rápidos y mejorando significativamente la eficiencia en diferentes escenarios. El marco teórico para TinyML implica conceptos fundamentales del aprendizaje automático, incluidas técnicas de optimización de modelos como la compresión de redes neuronales, la cuantificación de pesos y la poda. Estos métodos son importantes para crear modelos ligeros que mantengan la precisión sin importar las limitaciones de recursos. Este proyecto se realizará implementando TinyML en un microcontrolador ESP32-CAM para la clasificación de imágenes de frutas y verduras en tiempo real. Los objetivos específicos incluyen entrenar un modelo utilizando Edge Impulse, optimizarlo para el dispositivo, desplegarlo y evaluar su rendimiento. La metodología describe las herramientas necesarias, como ESP32-CAM, los componentes asociados y detalla las actividades requeridas para que el desarrollo sea exitoso. Para la validación de la funcionalidad y el rendimiento de la solución se realizarán pruebas de clasificación en tiempo real, monitoreo del consumo de recursos y evaluaciones de robustez en diferentes condiciones. Este enfoque garantiza la aplicación efectiva de TinyML en hardware asequible y accesible, facilitando el crecimiento y expansión de la inteligencia artificial en diferentes ámbitos y contextos.

Introducción

El desarrollo de soluciones TinyML (Tiny Machine Learning) aborda el desafío de implementar modelos de aprendizaje automático en dispositivos con recursos extremadamente limitados, como microcontroladores o sensores. A diferencia de sistemas convencionales, estos dispositivos tienen restricciones significativas en cuanto a capacidad de procesamiento, memoria y energía. La necesidad surge de la creciente demanda por inteligencia en el borde ("edge computing"), donde se procesan datos en tiempo real sin depender de infraestructura remota. Esto es crucial en aplicaciones como dispositivos portátiles, hogares inteligentes y monitoreo industrial.

La importancia del desarrollo de este desafío radica en que la inteligencia en el borde permite respuestas inmediatas, reducción en la latencia y menor consumo de ancho de banda. En el contexto tecnológico actual, donde la conectividad y el procesamiento en tiempo real son fundamentales, TinyML proporciona soluciones que pueden ejecutar tareas de IA de forma local en dispositivos con recursos limitados, permitiendo así la proliferación de la inteligencia artificial en nuevos escenarios.

Antecedentes o marco teórico

TinyML se basa en aspectos fundamentales de aprendizaje automático y optimización de modelos. La compresión de redes neuronales, la cuantificación de pesos y las técnicas de poda son claves para reducir el tamaño de los modelos sin comprometer su precisión. Además, se aprovechan arquitecturas ligeras, como MobileNet o redes neuronales convolucionales reducidas, que están diseñadas para operar con menos recursos.

En relación con Sistemas Operativos, TinyML depende del manejo eficiente de recursos como memoria, ciclos de CPU y consumo energético. Estos dispositivos ejecutan sistemas operativos en tiempo real (RTOS) que proporcionan un entorno optimizado para manejar procesos en tiempo real con restricciones estrictas. Esto conecta directamente con los temas del curso, como la gestión de memoria, la planificación de procesos y el manejo de interrupciones en dispositivos embebidos, elementos claves tanto en la teoría como en el laboratorio de sistemas operativos.

Objetivos (principal y específicos)

Objetivo principal

Implementar un sistema TinyML en una plataforma de microcontrolador (como el ESP32-CAM) para realizar tareas de clasificación de imágenes en tiempo real.

Objetivos específicos

- Entrenar y optimizar un modelo de clasificación de imágenes utilizando Edge Impulse.
- Implementar técnicas de compresión y cuantificación del modelo para ajustarlo a las limitaciones del dispositivo.
- Desplegar el modelo en el microcontrolador ESP32-CAM y evaluar su rendimiento en términos de precisión y eficiencia de recursos.

Metodología

Herramientas principales

- **ESP32-CAM:** Una pequeña cámara, ideal para capturar imágenes de objetos y realizar inferencias en tiempo real utilizando modelos TinyML.
- **USB to Serial Converter:** Permite programar y enviar datos al ESP32-CAM desde tu computadora.
- **Breadboard:** Facilita las conexiones de los componentes.
- **0.96" OLED Display:** Pantalla que se puede usar para mostrar los resultados de la clasificación de imágenes.
- **Jumper wires:** Cables de conexión para interconectar los componentes.
- **Frutas o vegetales para clasificar:** Objetos físicos que se usarán para entrenar y probar el modelo de clasificación.

Actividades

1. **Recopilación de datos:** Usar la cámara del ESP32-CAM para capturar imágenes de frutas y cargarlas en la plataforma **Edge Impulse** para su procesamiento.
2. **Entrenamiento del modelo:** Crear y entrenar un modelo de clasificación de imágenes en Edge Impulse, diferenciando entre diferentes tipos de frutas.
3. **Optimización del modelo:** Aplicar técnicas de compresión y cuantificación del modelo utilizando TensorFlow Lite para que se ajuste a las limitaciones del ESP32-CAM.
4. **Implementación:** Programar el ESP32-CAM con **Arduino IDE**, integrar el modelo entrenado y mostrar los resultados de la clasificación en el **OLED Display**.
5. **Validación y ajustes:** Evaluar el rendimiento del sistema clasificador en tiempo real, ajustando los parámetros según sea necesario para mejorar la eficiencia de los recursos y la precisión del modelo.

Esta metodología garantiza un proceso ordenado para la implementación de un sistema de clasificación de frutas en dispositivos con recursos limitados, usando TinyML en hardware económico y accesible.

Diseño de Experimentos

Para validar la funcionalidad y el rendimiento de la solución desarrollada en este desafío de TinyML utilizando la ESP32-CAM, se pueden emplear varias estrategias clave como:

Validación de la funcionalidad

- **Pruebas de clasificación en tiempo real:** Se utilizará la cámara de la ESP32-CAM para capturar imágenes de frutas o vegetales en tiempo real y el modelo entrenado deberá clasificar correctamente los objetos con los que fue entrenado.
- **Verificación en el OLED Display:** El sistema debe mostrar las predicciones en el display OLED en tiempo real.
- **Comparación de precisión del modelo:** Antes de implementarlo en la ESP32-CAM, se puede probar el modelo directamente en Edge Impulse para verificar su precisión. Para verificar que después de implementarse los resultados sean similares.

Validación del rendimiento

- **Evaluación de la latencia de predicción:** Se medirá el tiempo que tarda el sistema en realizar una predicción desde que se captura la imagen hasta que el resultado se muestra en la pantalla OLED.
- **Consumo de recursos:** Se supervisará el uso de la memoria y la CPU de la ESP32-CAM para asegurarse de que el modelo se ejecuta eficientemente dentro de las limitaciones del dispositivo.
- **Pruebas de robustez:** Se pueden realizar pruebas con distintas condiciones de iluminación y diferentes ángulos de las frutas para evaluar cómo el sistema maneja variaciones en las condiciones del entorno.

Validación final


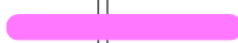






La validación final consistirá en comparar la tasa de error del modelo en el entorno real con la precisión esperada obtenida durante el entrenamiento. El rendimiento y la funcionalidad se ajustarán si el sistema no cumple con los estándares definidos inicialmente, optimizando el modelo o ajustando las condiciones de captura de imágenes.

Al seguir este enfoque, se asegura que el sistema TinyML funcione no solo de manera precisa, sino también eficiente dentro de las limitaciones de hardware de la ESP32-CAM.

Como se debería hacer

- <https://circuitdigest.com/microcontroller-projects/object-recognition-using-esp32-cam-and-edge-impulse>

Cronograma

ETAPA	SEMANA 1	SEMANA 2	SEMANA 3	SEMANA 4	SEMANA 5	SEMANA 6
DEFINICIÓN DEL PROYECTO						
ENTRENAMIENTO DEL MODELO						
OPTIMIZACIÓN DEL MODELO						
PROGRAMACIÓN DEL ESP32-CAM						
INTEGRACIÓN OLED DISPLAY						
VALIDACIONES						
AJUSTES DE RENDIMIENTO						
DOCUMENTO FINAL						

Referencias

- DataScientest. (n.d.). *TinyML: Todo lo que necesitas saber*. DataScientest. <https://datascientest.com/es/tinyml-todo-sobre>
- Mjrovai. (n.d.). *ESP32-TinyML: Exploring TinyML with ESP32 MCUs* [Repository]. GitHub. <https://github.com/Mjrovai/ESP32-TinyML>
- Hackster.io. (n.d.). *ESP32-CAM: TinyML Image Classification - Fruits vs Veggies*. Hackster.io. <https://www.hackster.io/mjrobot/esp32-cam-tinyml-image-classification-fruits-vs-veggies-4ab970>