Machine Learning en Dispositivos de Bajo Consumo

Freddy Rospigliosi

Agenda del día

- ¿Qué es TensorFlow Lite?
- ¿Qué es TinyML?
- Raspberry Pi para proyectos de visión y ML

¿Qué es TensorFlow Lite?

- **TensorFlow Lite** es una versión ligera y optimizada de **TensorFlow**, el popular *framework* de código abierto de Google para el aprendizaje automático (Machine Learning). Su propósito principal es permitir que los modelos de Machine Learning (incluyendo los de visión por computadora) se ejecuten de manera eficiente en dispositivos con recursos limitados.
- Imagina que entrenas un modelo de reconocimiento facial súper potente en una computadora de escritorio con una tarjeta gráfica de última generación. Ese modelo podría ser muy grande y requerir mucha potencia de cálculo. ¿Qué pasa si quieres usar ese modelo en un *smartphone*, una Raspberry Pi, o incluso un microcontrolador? Ahí es donde entra TensorFlow Lite.

¿Qué es TensorFlow Lite?

- Versión optimizada y ligera de TensorFlow
- Diseñada para móviles, microcontroladores y dispositivos embebidos
- Permite ejecutar modelos entrenados con consumo mínimo de recursos

¿Por qué necesitamos TensorFlow Lite?

- Los modelos de Machine Learning suelen ser:
- **Grandes:** Pueden ocupar muchos megabytes o incluso gigabytes.
- **Intensivos en cómputo:** Requieren mucha memoria y potencia de procesamiento para hacer inferencias (es decir, hacer predicciones).
- La mayoría de los dispositivos móviles, embebidos o de Internet de las Cosas (IoT) tienen:
  - Poca memoria RAM.
  - Procesadores menos potentes.
  - Batería limitada.
  - Conectividad a menudo intermitente o inexistente (lo que hace inviable depender siempre de la nube para las inferencias).
- TensorFlow Lite resuelve estos problemas al optimizar los modelos para que sean **más pequeños y más rápidos**, permitiendo que las aplicaciones de IA funcionen directamente en el dispositivo (**inferencia en el borde o *on-device inference***).

¿Cómo funciona TensorFlow Lite?

- El proceso para usar TensorFlow Lite generalmente sigue estos pasos:
- **Entrenamiento del modelo:** Primero, entrenas tu modelo de Machine Learning (por ejemplo, un modelo de clasificación de imágenes o detección de objetos) usando TensorFlow estándar en una computadora potente o en la nube.
- **Conversión a formato TensorFlow Lite (.tflite)**: Una vez entrenado, el modelo se convierte al formato .tflite. Durante esta conversión, se aplican varias optimizaciones:
  - **Cuantización**: Reduce la precisión de los números que representan los pesos del modelo (por ejemplo, de punto flotante de 32 bits a enteros de 8 bits). Esto reduce drásticamente el tamaño del modelo y acelera el cálculo, aunque puede haber una pequeña pérdida de precisión.
  - **Podado (Pruning)**: Elimina conexiones poco importantes en la red neuronal para reducir el tamaño y la complejidad.
  - **Fusión de operaciones**: Combina varias operaciones pequeñas en una sola para mejorar la eficiencia.

¿Cómo funciona TensorFlow Lite?

- **Implementación en el dispositivo**: El archivo .tflite optimizado se implementa en la aplicación del dispositivo móvil o embebido.
- **Ejecución del inferencia:** El **intérprete de TensorFlow Lite** carga el modelo y ejecuta las inferencias (predicciones) en tiempo real. Está diseñado para ser liviano y compatible con una variedad de *hardware* y sistemas operativos (Android, iOS, Linux, microcontroladores, etc.).
- También puede aprovechar aceleradores de *hardware* específicos del dispositivo (como GPUs, DSPs, NPUs) si están disponibles, lo que mejora aún más el rendimiento.

Ventajas de usar TensorFlow Lite

- **Bajo tamaño de modelo**: Los modelos son significativamente más pequeños, ocupando menos espacio de almacenamiento en el dispositivo.
- **Baja latencia**: Las inferencias se realizan directamente en el dispositivo sin necesidad de enviar datos a la nube, lo que reduce el retraso. Esto es crítico para aplicaciones en tiempo real (como la detección facial o la realidad aumentada).Privacidad: Los datos del usuario no salen del dispositivo, lo que mejora la privacidad.
- **Confiabilidad**: La aplicación funciona incluso sin conexión a internet.
- **Menor consumo de energía**: Al ser más eficiente, prolonga la vida útil de la batería del dispositivo.
- **Flexibilidad**: Admite una amplia gama de casos de uso y arquitecturas de modelos.

Aplicaciones comunes de TensorFlow Lite:

- **Dispositivos móviles:** Filtros y efectos de cámara en tiempo real.
  - Reconocimiento de voz y comandos.
  - Detección de objetos y clasificación de imágenes en aplicaciones.
  - Personalización en el dispositivo (teclados inteligentes, recomendaciones).
- **Dispositivos IoT y embebidos:**
  - Sistemas de seguridad (detección de movimiento, reconocimiento de personas).
  - Hogares inteligentes (control por voz, detección de actividad).
  - Automatización industrial (monitoreo de equipos, detección de anomalías).
  - Drones y robótica (navegación, reconocimiento de objetos).

Flujo general

- Entrenas un modelo en TensorFlow tradicional.
- Lo conviertes a formato .tflite
- Lo despliegas en tu dispositivo.(SmartPhone, Raspberry, etc)
- El modelo realiza inferencia (predicción) en tiempo real.

¿Qué es TinyML?

- TinyML = Machine Learning en dispositivos muy pequeños y de bajo consumo
- Procesamiento en el borde (on-device) **sin conexión a internet**
- Ejemplos:
  - Detectar sonidos (aplausos, ronquidos)
  - Clasificar imágenes simples
  - Activar alarmas con sensores de movimiento

TinyML

- **TinyML** es un campo emergente dentro de la inteligencia artificial que se enfoca en llevar el **Machine Learning (ML)** a los dispositivos más pequeños y con recursos más limitados. Piensa en microcontroladores, sensores y otros sistemas embebidos de muy bajo costo y consumo de energía.
- Mientras que **TensorFlow Lite** (como te expliqué antes) permite ejecutar modelos de ML en *smartphones* o Raspberry Pi, **TinyML** va un paso más allá, apuntando a dispositivos que tienen solo unos pocos kilobytes (KB) de memoria RAM y que operan con muy poca energía, a menudo con una batería de larga duración.

¿Por qué es importante TinyML?

- La mayoría de los sistemas de IA tradicionales requieren una gran cantidad de potencia de cómputo y energía, lo que limita su despliegue a la nube o a dispositivos más robustos. Sin embargo, hay miles de millones de dispositivos pequeños en el mundo que podrían beneficiarse de la inteligencia artificial si esta pudiera ejecutarse directamente en ellos.
- **TinyML busca superar las siguientes limitaciones**:
  - **Recursos extremadamente limitados**: Microcontroladores con muy poca RAM, capacidad de almacenamiento y potencia de procesamiento.
  - **Bajo consumo de energía**: Muchos de estos dispositivos funcionan con baterías diminutas y deben operar durante meses o años sin recargarse.
  - **Falta de conectividad**: A menudo operan en ubicaciones remotas sin acceso constante a internet o la nube.

¿Cómo funciona TinyML?

- El funcionamiento de TinyML se basa en una combinación de técnicas:
- **Modelos ultra-optimizados:** Se entrenan modelos de ML (a menudo redes neuronales convolucionales o redes recurrentes simples) en máquinas potentes, pero luego se someten a una **optimización extrema** para reducir su tamaño y complejidad.
- **Herramientas especializadas:** Existen *frameworks* y bibliotecas diseñadas específicamente para TinyML, como **TensorFlow Lite for Microcontrollers (TFLite Micro)**, que es una versión aún más ligera de TensorFlow Lite. Estas herramientas permiten convertir los modelos optimizados a un formato que puede ser ejecutado directamente en el microcontrolador.
- **Hardware de bajo consumo:** Los modelos se despliegan en microcontroladores y procesadores específicos diseñados para la eficiencia energética, a menudo con unidades de procesamiento de IA (NPUs) o aceleradores de propósito específico.

Aplicaciones de TinyML

- TinyML está abriendo la puerta a una nueva generación de dispositivos inteligentes en una amplia gama de sectores:
- Monitoreo de salud personal: Dispositivos wearables (relojes inteligentes, parches) que monitorean constantes vitales, patrones de sueño o actividad física, detectando anomalías y alertando al usuario o a profesionales de la salud sin necesidad de conexión constante.
- **Mantenimiento predictivo industrial**: Sensores en maquinaria que detectan sonidos, vibraciones o temperaturas anómalas, prediciendo fallos antes de que ocurran y optimizando el mantenimiento.
- **Agricultura de precisión**: Sensores de suelo que analizan las condiciones para optimizar el riego y el uso de fertilizantes.
- **Hogares y edificios inteligentes**: Sensores de presencia para gestionar la iluminación o la climatización, o detectores de fugas de agua o gas.
- **Detección de palabras clave y gestos**: Dispositivos que se activan por voz ("Oye, Siri" o "Alexa") o por un gesto específico sin enviar el audio/video a la nube.
- **Monitoreo ambiental**: Sensores que detectan la calidad del aire, la presencia de contaminantes o patrones de vida silvestre.

Arquitectura típica de TinyML

- Recolección de datos (sensor/cámara)
- Preprocesamiento (normalización, redimensionado)
- Inferencia (modelo .tflite)
- Acción o respuesta

Ventajas de TinyML

- **Consumo de energía ultrabajo**: Permite dispositivos que funcionan con baterías durante años.
- **Latencia cero**: Las decisiones se toman al instante en el dispositivo, sin retrasos por la red.
- **Privacidad**: Los datos sensibles permanecen en el dispositivo y no se envían a la nube.
- **Fiabilidad**: Funcionamiento autónomo sin depender de la conectividad a internet.
- **Costo reducido**: Utiliza hardware de bajo costo y minimiza el uso de recursos de red.

Limitaciones de TinyML

- **Modelos muy simples**: Solo pueden ejecutarse modelos ML con una complejidad muy limitada debido a las restricciones de memoria y procesamiento. No esperes ejecutar una red neuronal masiva para procesamiento de lenguaje natural en un microcontrolador.
- **Precisión comprometida**: La agresiva cuantización y optimización pueden llevar a una ligera pérdida de precisión en comparación con los modelos de tamaño completo.
- **Entrenamiento complejo**: El proceso de optimización del modelo y su despliegue requiere conocimientos especializados en ML y sistemas embebidos.
- **Menos flexibilidad:** Una vez que un modelo se ha optimizado y desplegado, cambiarlo o actualizarlo puede ser más complejo.

TensorFlow Lite vs. TinyML: La Diferencia Clave

TensorFlow Lite	TinyML
conjunto de herramientas y bibliotecas de software desarrollado por Google para ejecutar modelos de Machine Learning en **dispositivos *edge*** (dispositivos periféricos como Smartphones, Raspberrys, etc).	se enfoca en hacer posible el Machine Learning en **dispositivos de ultra bajo consumo y con recursos *extremadamente* limitados**. (Microcontroladores)
Optimizar modelos de TensorFlow para que sean más pequeños y rápidos, adecuados para inferencia en el dispositivo.	Llevar la inferencia de ML a microcontroladores y sensores con unos pocos **kilobytes de RAM** y que puedan funcionar con miliwatts (mW) o incluso microwatts (µW) de energía durante meses o años con una batería.
Dispositivos Típicos: smartphones (Android, iOS), computadoras de placa única (como Raspberry Pi), hasta sistemas embebidos de Linux.	microcontroladores más pequeños y de menor costo (como algunas placas Arduino, ESP32, y chips Cortex-M de ARM)

TensorFlow Lite vs. TinyML:

Característica	TensorFlow Lite	TinyML
Naturaleza	**Herramienta/Framework** de software	**Campo/Ecosistema** interdisciplinario (software + hardware + algoritmos)
Escala de Dispositivos	Smartphones, Raspberry Pi, Linux embebido (mayor potencia que MCUs)	**Microcontroladores** y sistemas embebidos de ultra bajo consumo y recursos **muy limitados**
Tamaño de Modelo Típico	Cientos de KB a pocos MB	**Pocos KB** (extremadamente pequeños)
Consumo de Energía	Bajo a moderado	**Ultra bajo** (miliwatts o microwatts)
Objetivo Principal	Optimizar ML para el *edge device* general	Habilitar ML para el *ultra-low-power edge device* (los más pequeños y eficientes)
Relación con TFLite	Puede usarse **dentro** del campo TinyML, específicamente **TensorFlow Lite Micro** es la implementación clave de TFLite para TinyML.	**Utiliza** herramientas como TensorFlow Lite Micro para lograr sus objetivos.

Edge computing

- La computación de borde (edge computing) es un paradigma informático que acerca el procesamiento y almacenamiento de datos a donde se generan, es decir, al borde de la red, **en lugar de centralizarlo en la nube o centros de datos**. Esto permite una **reducción de la latencia**, optimización del ancho de banda y una mayor eficiencia en la toma de decisiones, especialmente en aplicaciones que requieren respuestas rápidas, como el Internet de las Cosas (IoT).

- https://youtube.com/shorts/1cGDd3GTMyA?si=EhC7F-qwIZX6pFCG