Optimizing AI Models for Embedded Systems

Surendra Panpaliya

Agenda

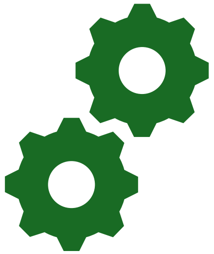
- Power Management and Efficiency
- Techniques for reducing power consumption in
- AI applications on embedded devices
- Balancing performance and
- power efficiency for longer battery life

Agenda

- Edge AI Frameworks and Tools
- Overview of Edge Impulse,
- TensorFlow Lite, Other frameworks for edge AI
- Comparing different frameworks
- for embedded AI development

Agenda

Hands-on Lab



Optimizing and Deploying AI Models

on Embedded Systems

Power Management and Efficiency

The goal is

to balance performance

with power efficiency to ensure

AI models run optimally

while extending the device’s battery life.

Model Optimization Techniques for Power Efficiency

Model Quantization

Reduces the precision of

Model’s weights and Activations

from 32-bit floating-point to

16-bit or 8-bit integers

Benefits

Require fewer bits per operation,

Reducing memory bandwidth and computation

Leads to power savings

Pruning

Removing unnecessary

weights or neurons from a model,

making the model lightweight

Benefits

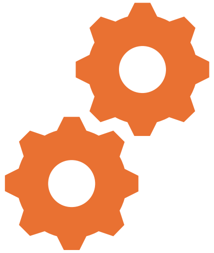
SPARSE MODELS USE LESS MEMORY

FEWER COMPUTATIONAL RESOURCES,

LEADING TO REDUCED POWER CONSUMPTION.

Software-Level Power Management Techniques

Power-Aware Scheduling



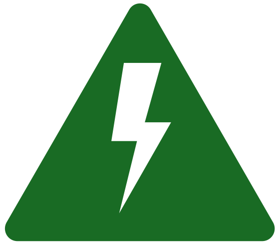
Optimizes

the scheduling of tasks

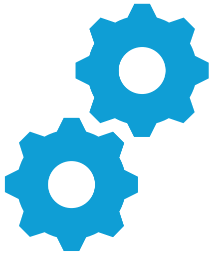
to reduce power consumption.

Power-Aware Scheduling

AI inference tasks can be scheduled



during periods of low power demand or



combined with other tasks to optimize energy use.

Use Case

EMBEDDED DEVICES THAT PERFORM MULTIPLE TASKS

SMARTWATCHES OR IOT SENSORS

CAN USE THIS TO RUN

AI INFERENCE TASKS EFFICIENTLY.

Idle Power Management

- Implementing
- low-power idle states
- when the AI model is
- not actively running

Idle Power Management

POWERING DOWN CERTAIN CORES

REDUCING CLOCK SPEEDS

WHEN THE SYSTEM IS IDLE.

Benefits

MINIMIZES POWER DRAW

DURING INACTIVE PERIODS.

Use Case

Ideal for systems that operate intermittently,

such as environmental monitoring systems

that only need to run inference a few times a day.

Hardware Optimization Techniques for Power Efficiency

Using Low-Power Hardware Accelerators

NPUs (Neural Processing Units)

TPUs (Tensor Processing Units)

Benefits

ACCELERATORS REDUCE

POWER CONSUMPTION AND

COMPUTATION TIME,

IDEAL FOR EMBEDDED AI APPLICATIONS.

Use Case

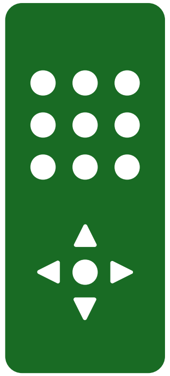
NPUs are used in smartphones

for tasks like face recognition and

object detection.

Use Case

TPUs are common in edge AI devices



low-power IoT systems.

Low-Power Memory Access Optimization

Optimizing memory access patterns,

Using on-chip memory or cache,

reduces power usage.

Use Case

Real-time AI applications

on microcontrollers or

battery-powered IoT devices.

Balancing Performance and Power Efficiency

Ensuring that

AI models meet real-time requirements

while conserving battery life

Dynamic Workload Scaling

High-load periods

When the system is plugged in or

has abundant power,

it can run the AI model

at full performance.

Low-load periods

When the device is on battery,

the system can downscale the workload

by reducing the frequency of model inference or

offloading certain tasks to lower-power components.

Case Study
Power Management in AI-based Devices

Smartphones

Scenario

Smartphones need to run AI tasks

such as facial recognition and voice processing

while balancing power consumption

for longer battery life.

Techniques

Use NPUs for AI inference

Quantizing models

to reduce power consumption

offloading non-essential tasks

to low-power cores.

Wearable Devices

Scenario

Health-monitoring wearables

must continuously track data

heart rate while maintaining battery life.

Techniques

Event-driven AI,

Energy-efficient models like MobileNet,

low-power accelerators like FPGAs or

NPUs for on-device inference.

Conclusion

Optimizing power management and efficiency

in AI applications for embedded systems

is essential to achieving longer battery life and

better overall performance.

Edge AI Frameworks and Tools

Agenda

- Overview of Edge Impulse,
- TensorFlow Lite, and
- Other frameworks for edge AI
- Comparing different frameworks
- for embedded AI development

Overview of Edge AI Frameworks and Tools

Designed to enable

AI model development,

optimization

deployment directly

on edge devices

Overview of Edge AI Frameworks and Tools

Focus on low-latency inference,

power efficiency, and

small memory footprint

Overview and comparison of Most popular Edge AI frameworks

Edge Impulse

Platform designed specifically

for embedded machine learning (ML) and

edge AI applications.

Edge Impulse

Provides a complete development pipeline,

including dataset acquisition,

model training, optimization, and deployment.

Edge Impulse

Widely used in IoT, wearables,

Sensor-based applications

where low-power and

real-time inference are essential

Key Features

Data collection tools

AutoML capabilities

Data collection tools

Edge Impulse allows developers

to collect, label, and

preprocess sensor data

directly from edge devices.

Data collection tools

AutoML capabilities

Automatic model optimization

quantization for edge devices.

Prebuilt libraries for microcontrollers

Supports MCUs like

Arm Cortex-M,

Nordic Semiconductor,

Espressif, and Arduino.

Deployment-ready

Easy-to-deploy models

on microcontrollers and

edge devices with

low power and memory constraints.

Edge Optimized Model Zoo

A collection of pre-built,

optimized models for vision,

audio, and motion data.

Advantages

End-to-end solution

TinyML support

End-to-end solution

Complete toolchain

from data acquisition

to deployment.

TinyML support

OPTIMIZED FOR TINYML

MACHINE LEARNING ON MICROCONTROLLERS

WITH TINY MEMORY FOOTPRINTS.

Low-code/No-code interface

Easy for beginners and

Non-experts to use.

Cloud-based model training

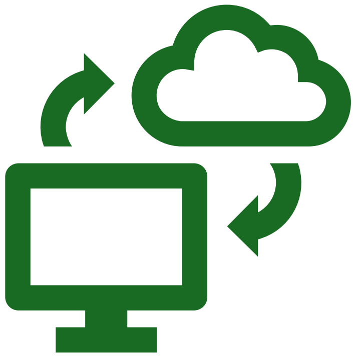
Training is done on the cloud,

but models are optimized

for deployment on edge devices.

Limitations

Limited to edge devices



Cloud-dependent training

Limited to edge devices

Focused mainly on

small-scale devices,

not ideal for large-scale

cloud-based models.

Cloud-dependent training

Although optimized for the edge,

model training is typically

performed in the cloud.

Use Cases

WEARABLES FOR MOTION TRACKING AND HEALTH MONITORING.

IOT SENSORS FOR ENVIRONMENTAL MONITORING.

AUDIO RECOGNITION IN LOW-POWER DEVICES

LIKE SMART HOME ASSISTANTS.

TensorFlow Lite

TensorFlow Lite is a lightweight version of

TensorFlow optimized for mobile and

embedded devices.

TensorFlow Lite

Widely used for deploying ML models

on edge devices like smartphones,

IoT devices, and microcontrollers.

TensorFlow Lite

TensorFlow Lite supports

model quantization, pruning, and

optimization techniques

to make models suitable

for edge deployment.

Key Features

Model conversion:

TensorFlow models can be easily

converted to TensorFlow Lite

format for edge deployment.

Key Features

MODEL OPTIMIZATION:

SUPPORTS POST-TRAINING QUANTIZATION,

QUANTIZATION-AWARE TRAINING, AND

PRUNING TO REDUCE MODEL SIZE AND

IMPROVE INFERENCE SPEED.

Key Features

Hardware acceleration:

Supports Edge TPUs, NPUs, and GPU

acceleration for optimized inference

on supported hardware.

Key Features

Cross-platform support:

TensorFlow Lite runs on a variety of devices,

from mobile (Android, iOS)

to microcontrollers (e.g., Arm Cortex-M).

Advantages

COMPREHENSIVE SUPPORT:

EXTENSIVE INTEGRATION WITH

THE TENSORFLOW ECOSYSTEM,

MAKING IT EASY TO MOVE FROM

MODEL TRAINING TO DEPLOYMENT.

Advantages

Hardware acceleration

Optimized for performance

with support for Edge TPU, GPU

Model quantization

Significant reductions in model size and

inference latency using

8-bit or 16-bit integer quantization.

Large community and support

As part of TensorFlow,

it benefits from a large ecosystem of tools,

libraries, and community support.

Limitations

Larger footprint for microcontrollers

Manual optimizations needed

Larger footprint for microcontrollers

While it supports microcontrollers,

TensorFlow Lite models may still

be relatively large for the most

resource-constrained devices.

Manual optimizations needed

Developers may need

to manually optimize models

to fully leverage edge device capabilities.

Use Cases

Mobile apps

Edge AI devices

IoT applications

Use Cases

Mobile apps

Edge AI devices

IoT applications

Microsoft Azure Percept

MICROSOFT'S PLATFORM

FOR BUILDING AI MODELS ON EDGE DEVICES

LEVERAGING AZURE AI AND AZURE IOT

TO MANAGE AND DEPLOY MODELS.

Microsoft Azure Percept

Integrates edge AI hardware

Pre-trained models with

cloud-based tools for AI development

Key Features

Edge AI hardware

Provides edge-optimized hardware

like Azure Percept Vision and

Azure Percept Audio.

Key Features

SEAMLESS INTEGRATION WITH AZURE IOT

COMBINES AI AT THE EDGE WITH IOT SOLUTIONS,

ENABLING REMOTE MONITORING,

MANAGEMENT, UPDATES FOR DEPLOYED MODELS.

Key Features

Pre-built AI models

Offers pre-built AI models for

object detection,

speech recognition, and

anomaly detection.

Advantages

Deep cloud integration

Pre-trained models

IoT and AI fusion

Limitations

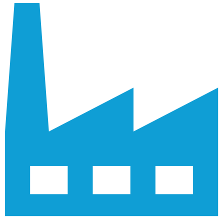
Azure cloud dependency

Limited hardware support

Use Cases

Smart cities

Retail



Manufacturing

AWS IoT Greengrass

Designed by Amazon

to extend AWS cloud capabilities

to edge devices.

AWS IoT Greengrass

Allows devices to act locally on

the data they generate while securely

communicating with AWS for management,

updates, and analytics.

AWS IoT Greengrass

Greengrass supports

deploying ML models on edge devices and

includes built-in integrations with

AWS SageMaker for training models in the cloud.

Key Features

- Edge AI inference
- Secure local execution
- Supports various edge hardware
- Built-in model management

Advantages

Integration with AWS ecosystem

Real-time local inference

Scalable

Limitations

Cloud dependency

Steeper learning curve

Use Cases

Smart agriculture

Predictive maintenance

Smart homes and cities

OpenVINO (Intel)

Open Visual Inference and Neural Network Optimization

is Intel’s toolkit for optimizing and

deploying deep learning models

on a variety of Intel hardware platforms.

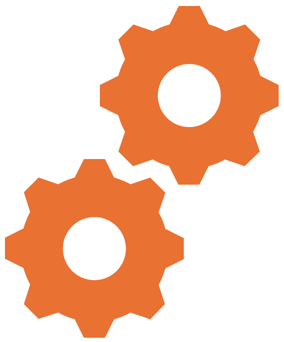
OpenVINO (Intel)

Focused on

computer vision applications

but can support other types of neural networks.

Key Features



Model optimization

Supports multiple AI frameworks

Key Features

Hardware acceleration

Extensive model zoo

Advantages

Hardware-specific optimizations

Low-latency inference

Cross-hardware deployment

Limitations

Limited hardware compatibility

Primarily focused on vision

Use Cases

Smart cameras

Retail analytics

Industrial automation

Conclusion

Each of these frameworks offers

unique strengths based on the use case,

hardware, and application needs.

Conclusion

Edge Impulse excels in TinyML and → sensor-based applications → on microcontrollers,

while TensorFlow Lite is ideal for → general-purpose edge AI across → mobile devices and IoT.

Conclusion

Azure Percept and AWS IoT Greengrass → offer robust cloud-integrated solutions → for managing edge devices at scale, → while OpenVINO is optimized for → Intel-based hardware and → high-performance computer vision applications.

Surendra Panpaliya

Founder and CEO

GKTCS Innovations

https://www.gktcs.com