Hardware Architectures For Embedded And Edge AI

Christian Rossi

Academic Year 2024-2025

Abstract

Contents

CHAPTER 1

Introduction

1.1 Introduction

Artificial Intelligence (AI) is a field of computer science focused on developing hardware and software systems capable of performing tasks that typically require human intelligence. These systems can autonomously pursue specific goals by making decisions that were traditionally made by humans.

A key distinction in AI-driven systems lies between smart objects and connected objects. While connected objects primarily send and receive data from the cloud, smart objects analyze data locally, enabling faster decision-making and reducing reliance on constant connectivity.

The definition of AI evolves rapidly, to the point that what was considered AI a decade ago may differ significantly from today's understanding.

AI hardware and software can be categorized similarly to traditional computing environments but are specifically designed to handle AI workloads. In this context, the development environment is often referred to as a framework, platform, or tool, rather than just a conventional programming environment.

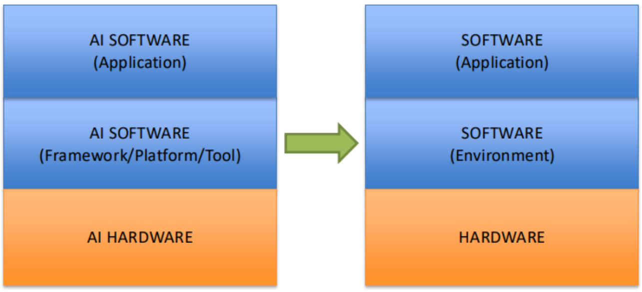

Figure 1.1: Artificial Intelligence stack

The AI stack consists of three main layers:

- *AI software* (application): AI-powered applications running within an IT system.
- *AI software* (framework, platform and tool): programs and libraries that manage physical resources and provide the necessary tools for building AI applications.

- *AI hardware*: the infrastructure supporting AI computation, including data centers, edge computing devices, IoT systems, and specialized processors like CPUs, GPUs, and TPUs.

1.2 Computing continuum

AI hardware ranges from small, low-power devices running on batteries to large-scale, high-performance systems in datacenters. This range represents the computing continuum:

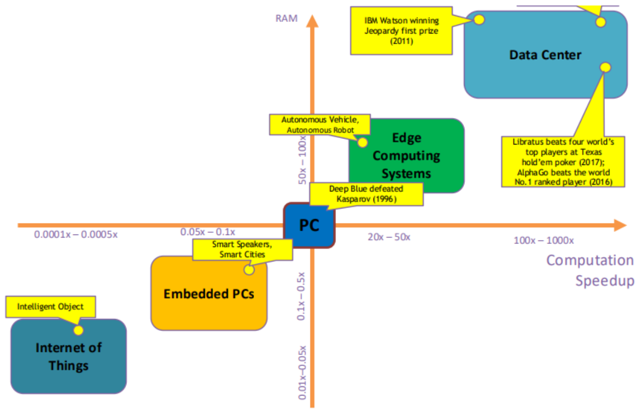

Figure 1.2: Computing continuum

1.2.1 Datacenters

Datacenters sit at the upper end of this spectrum, offering immense computational power for AI workloads. They provide cost-efficient IT infrastructure, high-performance computing capabilities, and instant software updates. Their vast storage capacity ensures data reliability and accessibility, allowing seamless collaboration across devices and locations. Furthermore, by decoupling AI processing from end-user devices, datacenters enable powerful AI applications that are not limited by local hardware constraints.

Datacenters require a constant internet connection, making them less viable in low-bandwidth environments. Their reliance on shared infrastructure can introduce privacy and security concerns, while the lack of direct hardware control may limit customization options. Additionally, the high energy consumption of large-scale AI operations raises both environmental and cost concerns. In latency-sensitive applications, delays in data transmission and processing can further impact real-time decision-making.

1.2.2 Edge computing systems

Edge computing delivers high computational power with the advantage of distributed processing. By bringing computation closer to where data is generated, it enhances privacy and security while significantly reducing latency in decision-making. However, these systems depend on a stable power supply and often integrate with cloud services to extend their processing capabilities.

By processing data locally, edge computing minimizes the need for constant data transmission, optimizing bandwidth and improving energy efficiency. This approach not only strengthens security and privacy but also enables real-time decision-making and adaptive learning across distributed networks. However, edge devices often operate with limited computing resources, constrained memory, and restricted energy availability. Their design requires careful coordination of hardware, software, and machine learning models, adding complexity to development and deployment.

1.2.3 Embedded systems

Embedded systems, widely used in AI applications, provide high-performance computing in a compact form. They benefit from the availability of development boards and can be programmed similarly to traditional computers, making them accessible to a broad community of developers. Despite these advantages, they tend to consume relatively high power, and in some cases, require custom hardware design to meet specific application needs.

1.2.4 Internet of Things

At the smallest scale, the Internet of Things (IoT) enables AI integration into pervasive, low-cost, battery-powered devices. These systems support wireless connectivity and often include sensing and actuating capabilities, making them essential for smart environments. However, IoT devices face limitations in computing power, energy efficiency, and memory capacity, which can complicate programming and constrain their ability to run advanced AI models.

CHAPTER 2

Hardware

2.1 Introduction

In embedded and edge AI systems, a typical setup includes sensors that capture data from the physical world, software that processes this data, and actuators that execute actions based on computational outcomes. All processing tasks rely on specialized hardware optimized for efficiency and performance.

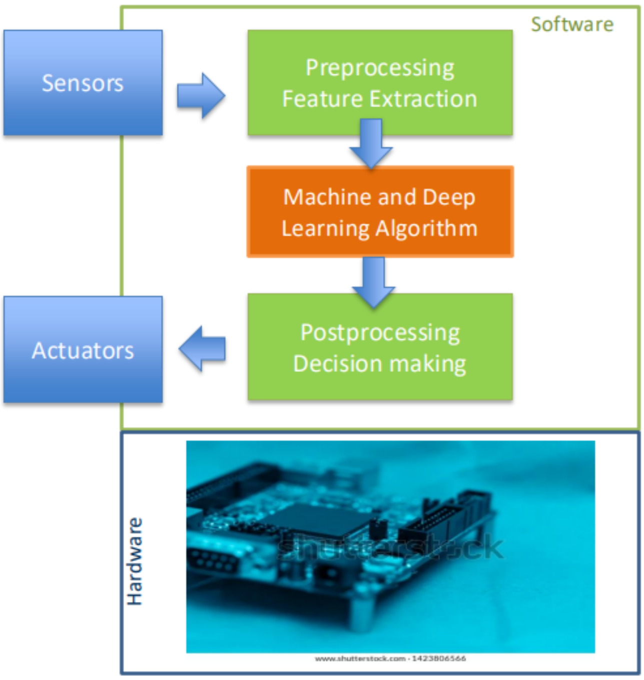

Figure 2.1: AI systems

Embedded systems are computers designed to control and manage the electronics within various physical devices. Embedded software refers to the programs that run on these systems, enabling their functionality.

Unlike general-purpose computers such as laptops or smartphones, embedded systems are typically designed for a specific, dedicated task, ensuring optimized performance, reliability, and energy efficiency for their intended application.

2.2 Architecture

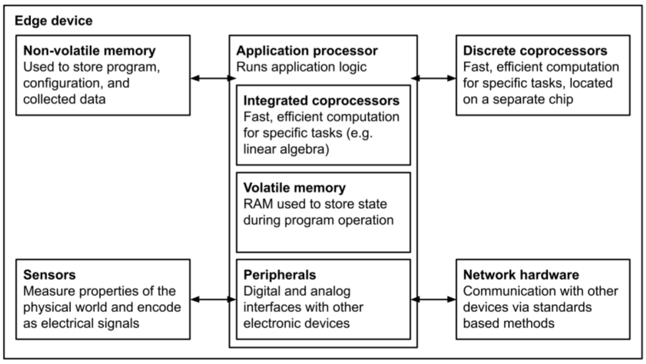

Figure 2.2: Hardware architecture

The hardware architecture of embedded and edge AI systems consists of several key components:

- *Non-volatile memeory*: used to store programs, configurations, and collected data. Flash memory is typically used for this purpose, as it retains data even when the system is powered off. It is ideal for storing information that does not change frequently but is slow to read and extremely slow to write.

- *Application processor*: runs the application logic and manages program execution. It includes an integrated coprocessor for efficient computation of specific tasks, volatile memory (RAM) for storing the system state during operation, and various digital and analog peripherals that allow interaction with other electronic components.

- *Discrete coprocessors*: external chips designed for high-speed, efficient mathematical computations. They provide additional processing power for specialized AI workloads that require high performance.

- *Sensors*: measure physical-world properties and convert them into electrical signals for processing. They enable real-time data collection, which is essential for AI-driven decision-making.

- *Network hardware*: ensures communication with other devices using standardized protocols. Reliable connectivity is crucial for data exchange in distributed AI systems.

RAM is often the performance bottleneck in embedded and edge AI systems. It is very fast but consumes significant energy, making efficiency critical in power-sensitive applications. Since it is volatile, data is lost when power is turned off. RAM is also costly and takes up a large physical footprint, impacting the overall design of embedded AI devices.

2.3 Sensors and signals

Sensors are used to acquire measurements from the environment or from human interactions. They generate continuous streams of data, which can be used for various AI-driven applications.

In addition to sensor-generated data, other sources such as digital device logs, network packets, and radio transmissions can also provide valuable information. Sensors can output data in different formats depending on their purpose and design.

Data storage Data values can be stored in various formats, depending on precision and memory constraints. Boolean values (1 bit) represent binary states with two possible values. An 8-bit integer can store up to 256 distinct values, while a 16-bit integer extends this range to 65,536 possible values. A 32-bit floating point number can represent a wide range of values with up to seven decimal places, reaching a maximum of approximately 3.4×10^{38}. Quantization techniques help optimize memory usage by reducing the required storage for each value while maintaining sufficient precision for AI computations.

2.3.1 Data

Time series data A time series is a sequence of data points recorded in chronological order:

$$X = (x_1, x_2, \ldots, x_N)$$

Essentially, it represents observations collected at consistent time intervals. Key factors to consider include:

- *Sampling period*: the time gap between consecutive data points.
- *Bit depth* (n): the number of bits used to represent each value.
- *Memory usage*: each sample requires n bits of storage.

Audio data Audio data is a specific type of time series, representing sound wave oscillations as they travel through air. Key parameters are:

- *Sampling rate* (Hz): the number of samples taken per second.
- *Quantization* (bit depth): the precision of each sample.
- *Signal duration* (s): the total length of the recording.
- *Number of channels*: mono (single channel) or stereo (two channels).

Memory consumption is calculated as:

$$\text{length} \times \text{sampling} \times \text{bit} \times \text{channel}$$

Image data Images capture visual information as a grid of pixels, where each pixel represents a specific property of the scene. Key characteristics are:

- *Dimensions* ($W \times H$): width and height of the image.
- *Bit depth* (N): the number of bits used to store each pixel.
- *Number of channels*: typically 1 (grayscale) or 3 (RGB).

Memory usage is given by:

$$W \times K \times N \times \text{channels}$$

Video data Videos are sequences of images displayed rapidly to create motion. They share the same structure as images but add an extra dimension: time. Critical parameters:

- *Resolution* ($W \times H$): width and height of each frame.
- *Bit depth* (N): bits per pixel.
- *Number of channels*: defines color representation.
- *Frame rate* (fps): number of frames per second.
- *Duration* (s): total length of the video.

Memory requirements are determined by:

$$W \times K \times N \times \text{channels} \times \text{frame rate} \times \text{length}$$

2.3.2 Sensors

There are thousands of different types of sensors available, each designed to capture specific kinds of data. In the context of embedded and edge AI, sensor technologies can be categorized into six main families:

1. *Acoustic and vibration*: detects sound and mechanical vibrations.
2. *Visual and scene*: captures images, video, and environmental light data.
3. *Motion and position*: measures movement, acceleration, and spatial positioning.
4. *Force and tactile*: detects pressure, touch, and force.
5. *Optical, electromagnetic, and radiation*: measures light, radio waves, and radiation levels.
6. *Environmental and chemical*: monitors temperature, humidity, gases, and other environmental factors.

Acoustic and vibration Detecting vibrations is a crucial capability in embedded and edge AI. These sensors allow systems to perceive movement, structural vibrations, and even communication signals from humans and animals at a distance. Acoustic sensors measure vibrations traveling through different media: air (microphones), water (hydrophones), and ground (geophones and seismometers). Since acoustic data is distributed across different frequencies, the sampling frequency plays a key role in ensuring accurate representation for a given application. These sensors typically produce audio data as their output.

Visual and scene Visual sensors capture information about the environment without direct contact. These range from tiny, low-power cameras to high-resolution multi-megapixel sensors. Key characteristics of image sensors: color channels, spectral response (infrared sensors), pixel size, resolution, and frame rate. The output of these sensors can be 2D or 3D images or video data, depending on the application.

Motion and position Motion and position sensors track movement and spatial positioning in various ways:

- *Tilt sensors*: simple mechanical switches that detect orientation changes.
- *Accelerometers*: measure acceleration along one or more axes.
- *Gyroscopes*: detect rotational movement.
- *Time-of-flight sensors*: emit light or radio waves to measure distances to objects.
- *Real-time locating systems*: use multiple transceivers placed around a space to track object positions.
- *Global Positioning System*: uses satellites to determine an object's precise location.

These sensors typically generate time-series data, tracking movement and positioning over time.