Machine Learning Sensors

Applications of Machine Learning
No Good Data Left Behind

5 Quintillion bytes of data produced every day by IoT

<1% of unstructured data is analyzed or used at all

Cisco, Internet of Things (IoT) Data Continues to Explode Exponentially. Who Is Using That Data and How?, Feb 5, 2018
The “Classic” TinyML Paradigm
How to Stop Your Smart TV From Spying on You

A voice command starts your TV, face recognition, and viewing data. Interconnectivity has privacy implications for smart TVs and how the FBI warns about snoopy smart TVs spying on you.

Google Calls Hidden Microphone in Its Nest Home Security Devices an 'Error'

The company says it was an oversight, but it does little to stem paranoia.

Support the Guardian Make a year-end gift today

The Guardian

US World Environment Soccer US Politics Business Tech Science Newsletters Fight to vote

The Observer Smart homes How to stop your smart home spying on you

Everything in your smart home, from the lightbulb to the thermostat, could be recording you or collecting data about your habits without your knowledge.
How do we architect future Tiny Machine Learning (tinyML) sensors efficiently, effectively and robustly into the embedded ecosystem?
Machine Learning Sensors
An ML sensor is a **self-contained system** that utilizes **on-device machine learning** to extract **useful information** by observing some complex set of phenomena in the **physical world** and reports it through a **simple interface** to a wider system.
Machine Learning Sensors

by Useful Sensors
Machine Learning Sensors
Machine Learning Sensors

Machine Learning (ML) Sensor → Processor → Cloud

Sensor 2.0
ML Sensors - Guiding Set of Principles

1. We need to raise the level of abstraction to enable ease of use for scalable deployment of ML sensors; not everyone should be required to be a developer or an engineer to leverage ML sensors into their ecosystem.

2. The ML sensor’s design should be inherently data-centric and defined by its input-output behavior instead of exposing the underlying hardware and software mechanisms that support ML model execution.

3. An ML sensor’s implementation must be clean and complexity-free. Features such as reusability, software updates, and networking must be thought through to ensure data privacy and secure execution.

4. ML sensors must be transparent, indicating in a publicly and freely accessible ML sensor datasheet all the relevant information to supplement the traditional information available for hardware sensors.

5. We as a community should aim to foster an open ML sensors ecosystem by maximizing data, model, and hardware transparency where possible, without necessarily relinquishing any claim to intellectual property.
ML Sensors - Guiding Set of Principles

1. We need to **raise the level of abstraction** to enable ease of use for scalable deployment of ML sensors; not everyone should be required to be a developer or an engineer to leverage ML sensors into their ecosystem.

2. The ML sensor’s **design should be inherently data-centric** and defined by its input-output behavior instead of exposing the underlying hardware and software mechanisms that support ML model execution.

3. An ML sensor’s **implementation must be clean and complexity-free**. Features such as reusability, software updates, and networking must be thought through to ensure data privacy and secure execution.

4. ML sensors **must be transparent**, indicating in a publicly and freely accessible ML sensor datasheet all the relevant information to supplement the traditional information available for hardware sensors.

5. We as a community should aim to **foster an open ML sensors ecosystem by maximizing data, model, and hardware transparency** where possible, without necessarily relinquishing any claim to intellectual property.
ML Sensors - Guiding Set of Principles

1. **We need to raise the level of abstraction** to enable ease of use for scalable deployment of ML sensors; not everyone should be required to be a developer or an engineer to leverage ML sensors into their ecosystem.

2. The ML sensor’s design should be inherently data-centric and defined by its input-output behavior instead of exposing the underlying hardware and software mechanisms that support ML model execution.

3. An ML sensor’s implementation must be clean and complexity-free. Features such as reusability, software updates, and networking must be thought through to ensure data privacy and secure execution.

4. ML sensors must be transparent, indicating in a publicly and freely accessible ML sensor datasheet all the relevant information to supplement the traditional information available for hardware sensors.

5. We as a community should aim to foster an open ML sensors ecosystem by maximizing data, model, and hardware transparency where possible, without necessarily relinquishing any claim to intellectual property.
We need to raise the level of abstraction to enable ease of use for scalable deployment of ML sensors; not everyone should be required to be a developer or an engineer to leverage ML sensors into their ecosystem.

The ML sensor’s design should be inherently data-centric and defined by its input-output behavior instead of exposing the underlying hardware and software mechanisms that support ML model execution.

An ML sensor’s implementation must be clean and complexity-free. Features such as reusability, software updates, and networking must be thought through to ensure data privacy and secure execution.

ML sensors must be transparent, indicating in a publicly and freely accessible ML sensor datasheet all the relevant information to supplement the traditional information available for hardware sensors.

We as a community should aim to foster an open ML sensors ecosystem by maximizing data, model, and hardware transparency where possible, without necessarily relinquishing any claim to intellectual property.
ML Sensors - Guiding Set of Principles

1. We need to raise the **level of abstraction** to enable ease of use for scalable deployment of ML sensors; not everyone should be required to be a developer or an engineer to leverage ML sensors into their ecosystem.

2. The ML sensor’s **design should be inherently data-centric** and defined by its input-output behavior instead of exposing the underlying hardware and software mechanisms that support ML model execution.

3. An ML sensor’s **implementation must be clean and complexity-free**. Features such as reusability, software updates, and networking must be thought through to ensure data privacy and secure execution.

4. ML sensors **must be transparent**, indicating in a publicly and freely accessible ML sensor datasheet all the relevant information to supplement the traditional information available for hardware sensors.

5. We as a community should aim to **foster an open ML sensors ecosystem by maximizing data, model, and hardware transparency** where possible, without necessarily relinquishing any claim to intellectual property.
1. We need to raise the level of abstraction to enable ease of use for scalable deployment of ML sensors;
2. The ML sensor's design should be inherently data-centric and defined by its input-output behavior instead of exposing the underlying hardware and software mechanisms that support ML model execution.

ML Sensors - Guiding Set of Principles
ML Sensors - Guiding Set of Principles

1. We need to raise the level of abstraction to enable ease of use for scalable deployment of ML sensors; not everyone should be required to be a developer or an engineer to leverage ML sensors into their ecosystem.

2. The ML sensor’s design should be inherently data-centric and defined by its input-output behavior instead of exposing the underlying hardware and software mechanisms that support ML model execution.

3. An ML sensor’s implementation must be clean and complexity-free. Features such as reusability, software updates, and networking must be thought through to ensure data privacy and secure execution.

4. ML sensors must be transparent, indicating in a publicly and freely accessible ML sensor datasheet all the relevant information to supplement the traditional information available for hardware sensors.

5. We as a community should aim to foster an open ML sensors ecosystem by maximizing data, model, and hardware transparency where possible, without necessarily relinquishing any claim to intellectual property.
ML Sensors - Guiding Set of Principles

1. We need to **raise the level of abstraction** to enable ease of use for scalable deployment of ML sensors; not everyone should be required to be a developer or an engineer to leverage ML sensors into their ecosystem.

2. The ML sensor’s **design should be inherently data-centric** and defined by its input-output behavior instead of exposing the underlying hardware and software mechanisms that support ML model execution.

3. An ML sensor’s **implementation must be clean and complexity-free**. Features such as **reusability**, software updates, and networking must be thought through to ensure data privacy and secure execution.

---

*We need to define or rely on standard interfaces and mechanisms for communication with sensors.*

Source: https://github.com/usefulsensors/person_sensor_docs
ML Sensors - Guiding Set of Principles

1. We need to **raise the level of abstraction** to enable ease of use for scalable deployment of ML sensors; not everyone should be required to be a developer or an engineer to leverage ML sensors into their ecosystem.

2. The ML sensor’s **design should be inherently data-centric** and defined by its input-output behavior instead of exposing the underlying hardware and software mechanisms that support ML model execution.

3. An ML sensor’s **implementation must be clean and complexity-free**. Features such as **reusability**, software updates, and networking must be thought through to ensure data privacy and secure execution.

---

**We need to define data formats to enable interoperability and exchange of ML sensors across manufacturers**

Source: https://github.com/usefulsensors/person_sensor_docs
ML Sensors - Guiding Set of Principles

1. We need to **raise the level of abstraction** to enable ease of use for scalable deployment of ML sensors; not everyone should be required to be a developer or an engineer to leverage ML sensors into their ecosystem.

2. The ML sensor’s **design should be inherently data-centric** and defined by its input-output behavior instead of exposing the underlying hardware and software mechanisms that support ML model execution.

3. An ML sensor’s **implementation must be clean and complexity-free**. Features such as reusability, software updates, and networking must be thought through to ensure data privacy and secure execution.

4. ML sensors **must be transparent, indicating in a publicly and freely accessible ML sensor datasheet** all the relevant information to supplement the traditional information available for hardware sensors.

5. We as a community should aim to **foster an open ML sensors ecosystem by maximizing data, model, and hardware transparency** where possible, without necessarily relinquishing any claim to intellectual property.
We need to raise the level of abstraction to enable ease of use for scalable deployment of ML sensors; not everyone should be required to be a developer or an engineer to leverage ML sensors into their ecosystem.

The ML sensor’s design should be inherently data-centric and defined by its input-output behavior instead of exposing the underlying hardware and software mechanisms that support ML model execution.

An ML sensor’s implementation must be clean and complexity-free. Features such as reusability, software updates, and networking must be thought through to ensure data privacy and secure execution.

ML sensors must be transparent, indicating in a publicly and freely accessible ML sensor datasheet all the relevant information to supplement the traditional information available for hardware sensors.

We as a community should aim to foster an open ML sensors ecosystem by maximizing data, model, and hardware transparency where possible, without necessarily relinquishing any claim to intellectual property.
1. We need to raise the level of abstraction to enable ease of use for scalable deployment of ML sensors; not everyone should be required to be a developer or an engineer to leverage ML sensors into their ecosystem.

2. The ML sensor’s design should be inherently data-centric and defined by its input-output behavior instead of exposing the underlying hardware and software mechanisms that support ML model execution.

3. An ML sensor’s implementation must be clean and complexity-free. Features such as reusability, software updates, and networking must be thought through to ensure data privacy and secure execution.

4. ML sensors must be transparent, indicating in a publicly and freely accessible ML sensor datasheet all the relevant information to supplement the traditional information available for hardware sensors.

5. We as a community should aim to foster an open ML sensors ecosystem by maximizing data, model, and hardware transparency where possible, without necessarily relinquishing any claim to intellectual property.
E.g. ML Sensors Datasheets

- Description, Features and Use case
- Diagram and Form Factor
- Hardware Characteristics
- Model Characteristics
- Environment Impact
- Performance Analysis
- Communication Spec. & Pinout
- Data Labels
- Privacy & Security
- Compliance

E.g. ML Sensors
Datasheets

PA1 Person Detection Module

Description: The PA1 Person Detection Module enables you to quickly and easily add smart to your IoT deployment to monitor and detect for humans. You can use this module indoors and outdoors to understand where and when humans arrive at your deployment site.

Features:
- Real-time Person Detection with On-Device ML
- Inference latency of 100ms for 30fps
- Fixed dimension of 25 x 25 mm
- Minimum distance of 500mm (max 20000 Lux)
- Operates in a variety of environments: Indoor, Outdoor
- Supports Color and Black-and-White Detection Modules

Use Cases:
- Smart business and home security systems
- Multi-modal key word spotting for virtual assistants
- Occupancy sensors and other infrastructure sensors

Diagram and Form Factor

Hardware Characteristics

Model Characteristics

Communication Spec. & Pinout

Data Labels
Nutrition Label
Privacy & Security

IoT Security & Privacy Label

Performance Analysis

Detection Accuracy vs Distance to Nearest Dynamic Object

False Positive Rate vs Distance from Nearest Static Object

Environ. Impact

Compliance

Model performance: measured with Precision, Recall, F1-Score and Area Under the ROC Curve (AUC). Download for performance results.

Model performance: measured with Precision, Recall, F1-Score and Area Under the ROC Curve (AUC). Download for performance results.
"Model cards aim to provide a concise, holistic picture of a machine learning model. To start, a model card explains what a model does, its intended audience, and who maintains it. A model card also provides insight into the construction of the model, including its architecture and the training data used." – Google Cloud
There’s a missing step in the AI development pipeline: assessing datasets based on standard quality measures that are both qualitative and quantitative. We are working on packaging up these measures into an easy to use Dataset Nutrition Label.” - Dataset Nutrition Project
“... designing a **usable security and privacy label** for smart devices to **help consumers** make informed choices about Internet of Things device purchases and encourage manufacturers to **disclose their privacy and security practices.**” – IoT Security & Privacy
We require **systematic methodologies** to evaluate how an **end-to-end system** performs under **real-world conditions**.
ML sensors ought to be **tested by 3rd party certification agencies or bodies that specialize in AI/ML technologies.**
We must quantify the effects of ML sensors in terms of carbon emissions. Carbon emissions have two sources: (1) **operational energy consumption**, and (2) **hardware manufacturing and infrastructure**. The former has been decreasing thanks to software and hardware innovations but the **total footprint is growing**.
Assessing the Environmental Impact of an MCU

The environmental impact of an MCU includes:

- **390g CO₂-eq**: 1.6km by car
- **23L**: 23 bottles of water
- **120mg P-eq**: 0.2 washing cycles
- **823mg NMVOC**: 2.7km by car

**Total Impact**

**Climate Change**

- End of Life: <1%
- Logistics: 1%
- Use: 8%
- Raw Materials: 10%
- Production: Other: 24%
- Production: Energy Consumption: 56%

**Water Demand**

- End of Life: <1%
- Logistics: <1%
- Use: 6%
- Raw Materials: 41%
- Production: Other: 15%
- Production: Energy Consumption: 39%

**Freshwater Eutrophication**

- End of Life: <1%
- Logistics: <1%
- Use: 28%
- Raw Materials: 27%
- Production: Other: 18%
- Production: Energy Consumption: 27%

**Protochemical Oxidant Formation**

- End of Life: <1%
- Logistics: 1%
- Use: 8%
- Raw Materials: 10%
- Production: Other: 2%
- Production: Energy Consumption: 71%

Source:
Figure 4. A breakdown of different TinyML system footprints highlights that the footprint is largely attributable to the embodied footprint of the power supply, onboard sensors, and transportation. Note that actuator and connectivity blocks from Pirson and Bol [21] are encapsulated in “Other” and “Processing”, respectively, while “Product Use” captures the operational footprint. The carbon footprint of TinyML Systems was also compared with Apple’s Series 7 Watch [12] and 16-inch MacBook Pro [11] as baseline references. For more details and to compute the footprint of your own TinyML system see github.com/harvard-edge/TinyML-Footprint.
1. We need to **raise the level of abstraction** to enable ease of use for scalable deployment of ML sensors; not everyone should be required to be a developer or an engineer to leverage ML sensors into their ecosystem.

2. The ML sensor’s **design should be inherently data-centric** and defined by its input-output behavior instead of exposing the underlying hardware and software mechanisms that support ML model execution.

3. An ML sensor’s **implementation must be clean and complexity-free**. Features such as reusability, software updates, and networking must be thought through to ensure data privacy and secure execution.

4. ML sensors **must be transparent**, indicating in a publicly and freely accessible ML sensor datasheet all the relevant information to supplement the traditional information available for hardware sensors.

5. We as a community should aim to **foster an open ML sensors ecosystem by maximizing data, model, and hardware transparency** where possible, without necessarily relinquishing any claim to intellectual property.
Machine Learning Sensors

An ML sensor is a self-contained system that utilizes on-device machine learning to extract useful information by observing some complex set of phenomena in the physical world and reports it through a simple interface to a wider system.

Machine learning sensors represent a paradigm shift for the future of embedded machine learning applications. Current instantiations of embedded ML suffer from complex integration, lack of modularity, and privacy and security concerns from data storage and transmission.
Machine Learning Sensors

An ML sensor is a self-contained system that utilizes on-device machine learning to extract useful information by observing some complex set of phenomena in the physical world and reports it through a simple interface to a wider system.

Machine learning sensors represent a paradigm shift for the future of embedded machine learning applications. Current instantiations of embedded ML suffer from complex integration, lack of modularity, and privacy and security concerns from data movement. ML sensors provide a more data-centric paradigm for embedding sensor intelligence on edge devices to combat these challenges.

Our vision for "sensor 2.0" entails segregating sensor input data and ML processing from the wider system at the hardware level and providing a thin interface that mimics traditional sensors in functionality. This separation leads to a modular and easy-to-use ML sensor device. ML sensors increase privacy and accuracy while making it easier for system builders to integrate ML into their products as a simple component.

To learn more about our approach, check out our whitepaper on arXiv.

Challenges

Interface Standards Ethics

Example ML Sensor Datasheet

This illustrative example datasheet highlighting the various sections of an ML Sensor datasheet. On the top, we have the items currently found in standard datasheets: the description, features, use cases, diagrams and form factor, hardware characteristics, and communication specification and pinout. On the bottom, we have the new items that need to be included in an ML sensor datasheet: the ML model characteristics, dataset nutrition label, environmental impact analysis, and end-to-end performance analysis. While we compressed this datasheet into a one-page illustrative example by combining features and data from a mixture of sources, on a real datasheet, we assume each of these sections would be longer and include additional explanatory text to increase the transparency of the device to end-users. Interested users can find the most up-to-date version of the datasheet online at https://github.com/harvard-edge/ML-Sensors.
Recap of ML Sensors

1. We need to **raise the level of abstraction** to enable ease of use for scalable deployment of ML sensors; not everyone should be required to be a developer or an engineer to leverage ML sensors into their ecosystem.

2. The ML sensor’s **design should be inherently data-centric** and defined by its input-output behavior instead of exposing the underlying hardware and software mechanisms that support ML model execution.

3. An ML sensor’s **implementation must be clean and complexity-free**. Features such as reusability, software updates, and networking must be thought through to ensure data privacy and secure execution.

4. ML sensors **must be transparent, indicating in a publicly and freely accessible ML sensor datasheet** all the relevant information to supplement the traditional information available for hardware sensors.

5. We as a community should aim to **foster an open ML sensors ecosystem by maximizing data, model, and hardware transparency** where possible, without necessarily relinquishing any claim to intellectual property.
Call to Action

Radcliffe exploratory seminar to determine:

What ethical considerations are necessary when developing ML sensors?

What compliance standards must be met by ML sensor developer and manufacturers?

How should ML sensors interface with existing systems?
Machine Learning Sensors

Pete Ward Johnson Stewart 1 BrianPlanche2 ColbyBubarey3 ShivrajPrakash4 EmmaChen5 ZainAsgar1 SachinKatti6 VijayJanapaReddi7
1 Stanford University 2 Harvard University

Abstract

Machine learning sensors represent a paradigm shift for the future of embedded machine learning applications. Current incarnations of embedded machine learning (ML) suffer from complex integrations, lack of modularity, and privacy and security concerns from data movement. This article proposes a more data-centric paradigm for embedding sensor intelligence on edge devices to combat these challenges. Our vision for "sensor 2.0" retains disaggregating sensor input data and ML processing from the wider system at the hardware level and providing a thin interface that mimics traditional sensors in functionality. This separation leads to a modular and easy-to-use ML sensor device. We discuss challenges presented by the standard approach of building ML processing into the software stack of the existing microcontroller on an embedded system and how the modularity of ML sensors alleviates these problems. ML sensors increase privacy and accuracy while making it easier for system builders to integrate ML into their products as a simple component. We provide examples of prospective ML sensors and an illustrative dataset as a demonstration and hope that this will build a dialogue to progress us towards sensor 2.0.

1 Introduction

Since the advent of AlexNet [3], deep neural networks have proven to be robust solutions to many challenges that involve making sense of data from the physical world. Machine learning (ML) models can now run on low-cost, low-power hardware capable of deployment as part of an embedded device. Processing data close to the sensor on an embedded device allows for an expansive new variety of always-on ML use-cases that preserve bandwidth, latency, and energy while improving responsiveness and maintaining data privacy. This emerging field, commonly referred to as embedded ML or tiny machine learning (TinyML) [33, 48, 59, 90], is paving the way for a prosperous new array of use-cases, from personalized health initiatives to improving manufacturing productivity and everything in between.

However, the current practice for combining inference and sensing is cumbersome and raises the barrier of entry to embedded ML. At present, the general design practice is to design or leverage a board with decoupled sensors and compute (in the form of a microcontroller or DSP), and for the developer to figure out how to run ML on these embedded platforms. The developer is expected to trans and optimize ML models and fit them within the resource constraints of the embedded device. Once an acceptable prototype implementation is developed, the model is integrated with the rest of the software on the device. Finally, the widget is retrofitted to the device under test to run inference. The current approach is slow, manual, energy-inefficient, and error-prone.

Figure 1. The Sensor 1.0 paradigm tightly couples the ML model with the application processor and logic, making it difficult to provide hard guarantees about the ML sensor's ultimate behavior.

Figure 2. Our proposed Sensor 2.0 paradigm. The ML model is tightly coupled with the physical sensor, separate from the application processor, and comes with an ML sensor dataset that makes its behavior transparent to the system integrators and developers.

It requires a sophisticated understanding of ML and the intricacies of ML model implementations to optimize and fit a model within the constraints of the embedded device.