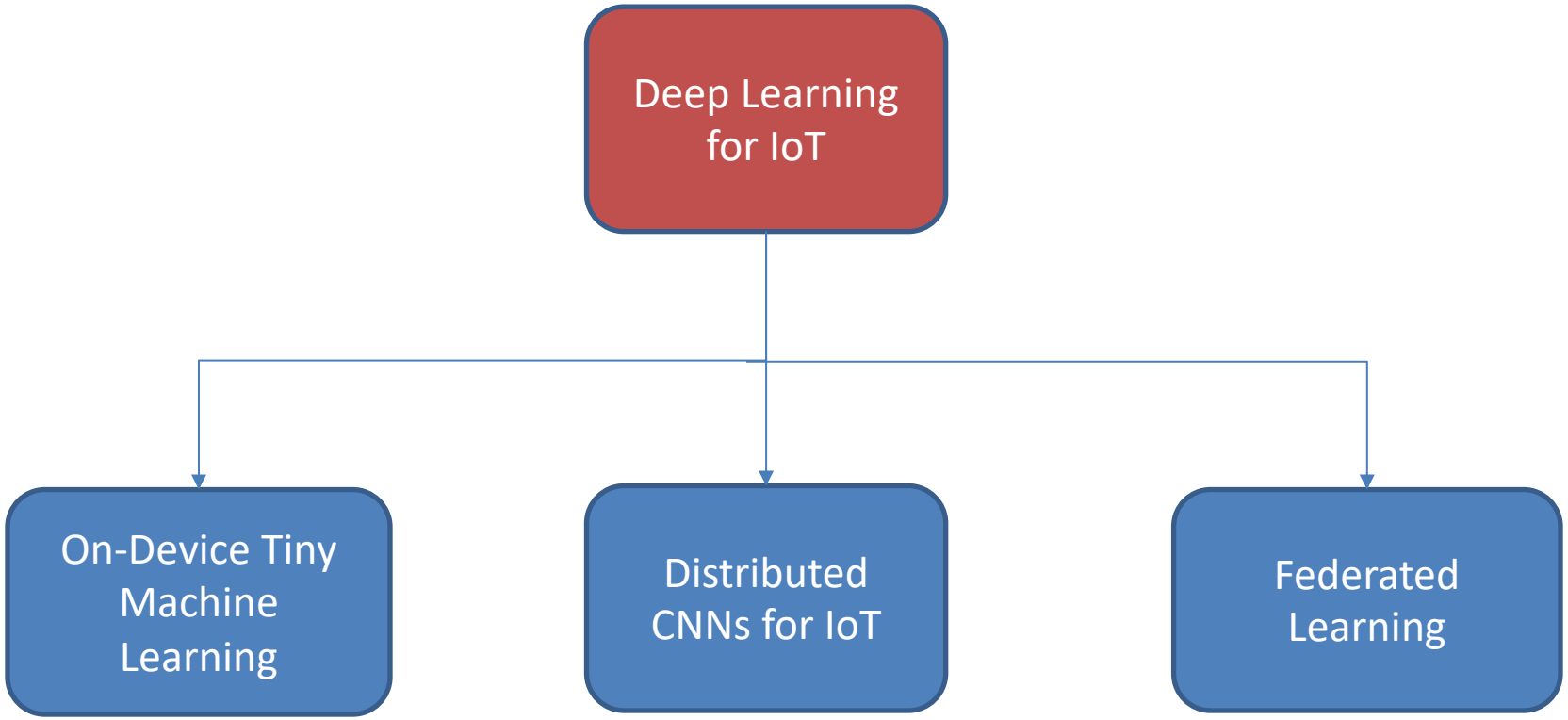
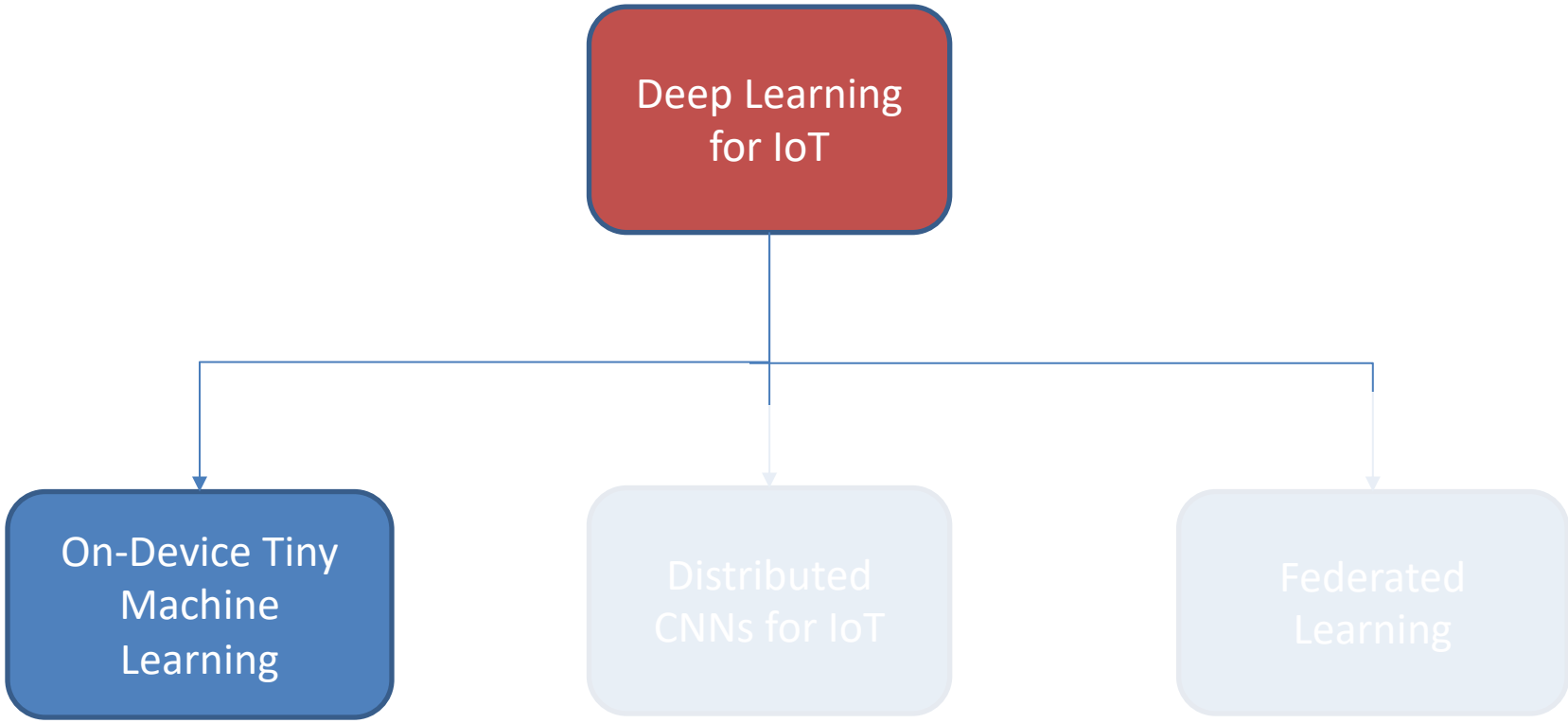
POLITECNICO
MILANO 1863

Hardware Architectures for Embedded and Edge AI

Prof Manuel Roveri – manuel.roveri@polimi.it

Lecture 10 – Deep Learning for IoT:
from the single unit to the ecosystem perspective


Deep Learning for IoT
On-Device Tiny Machine Learning
Distributed CNNs for IoT
Federated Learning

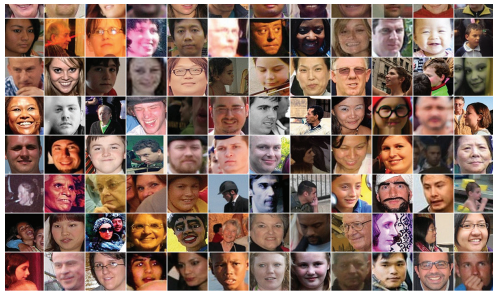

Deep Learning for IoT
On-Device Tiny Machine Learning
Distributed CNNs for IoT
Federated Learning

Let's start with a provocative question …

Is On-Device Learning
the Next Big Thing in TinyML?

Let's start with a provocative question …

Is On-Device Learning
the ~~Next~~ Big Thing in TinyML?



FINDING ONE FACE IN A MILLION

A new benchmark test shows that even Google's facial recognition algorithm is far from perfect

Helen of Troy may have had the face that launched a thousand ships, but even the best facial recognition algorithms might have had trouble finding her in a crowd of a million strangers. The first public benchmark test based on 1 million faces has shown how facial recognition algorithms from Google and other research groups around the world still fall well short of perfection.

Facial recognition algorithms that had previously performed with more than 95 percent accuracy on a popular benchmark test involving 13,000 faces saw significant drops in accuracy when taking on the new MegaFace Challenge. The best performer, Google's FaceNet algorithm, dropped from near-perfect accuracy on the five-figure data set to 75 percent on the million-face test. Other top algorithms dropped from above 90 percent to below 60 percent. Some algorithms made the proper identification as seldom as 35 percent of the time.

"MegaFace's key idea is that algorithms should be evaluated at large scale," says Ira Kemelmacher-Shlizerman, an assistant professor of computer science at the University of Washington, in Seattle, and the project's principal investigator. "And we make a number of discoveries that are now has been the Labeled Faces in the Wild (LFW) test created in 2007. LFW includes 13,000 images of just 5,000 people. Many facial recognition algorithms have been fine-tuned to the point that they scored near-perfect accuracy when picking through the LFW images. Most researchers say that new benchmark challenges have been long overdue.

"The big disadvantage is that [the field] is saturated—that is, there are many, many algorithms that perform above 95 percent on LFW," Kemelmacher-Shlizerman says. "This gives the impression that face recognition is solved and working perfectly."

With that in mind, University of Washington researchers raised the bar by creating the MegaFace Challenge using 1 million Flickr images of 690,000 unique faces that are publicly available under a Creative Commons license.

The MegaFace Challenge forces facial recognition algorithms to do verification and identification, two separate but related tasks. Verification involves trying to correctly determine whether two faces presented to the facial recognition algorithm belong to the same person. Identification involves trying to find a matching photo of the same person among a million "distractor" faces. Initial results on algorithms developed by Google and four other research groups were presented at the IEEE Conference on Computer Vision and Pattern Recognition on 30 June. (One of MegaFace's developers also heads a computer vision team at Google's Seattle office.)

The results presented were a mix of the intriguing and the expected. Nobody was surprised that the algorithms' performances suffered as the number of distractor faces increased. And the fact that algorithms had trouble identifying the same person at different ages was a known problem. However, the results also showed that algorithms trained on relatively small data sets can compete with those trained on very large ones, such as Google's FaceNet, which was trained on more than 500 million photos of 10 million people.

For example, the FaceN algorithm from Russia's N-TechLab performed well on certain tasks in comparison with FaceNet, despite having trained on 18 million photos of 200,000 people. The SIAT MMLab algorithm, created by a Chinese team under the leadership of Yu Qiao, a professor with Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, also performed well on certain tasks.

Nevertheless, FaceNet has so far performed the best overall. It delivered the most consistent performance across all testing.

The huge drops in accuracy when scanning a million faces matter because facial recognition algorithms inevitably face such challenges in the real world. People increasingly trust these algorithms to correctly identify them in security verification scenarios, and law enforcement may also rely on facial recognition to pick suspects out of the hundreds of thousands of faces captured on surveillance cameras.

UNIVERSITY OF WASHINGTON

The huge drops in accuracy when scanning a million faces matter because facial recognition algorithms inevitably face such challenges in the real world. People increasingly trust these algorithms to correctly identify them in security verification scenarios, and law enforcement may also rely on facial recognition to pick suspects out of the hundreds of thousands of faces captured on surveillance cameras.

The most popular benchmark until

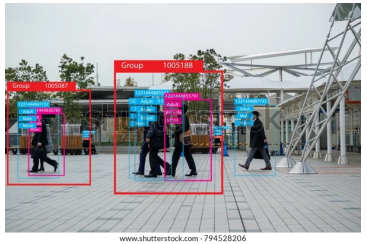
IEEE Spectrum
Aug, 2016

TinyML: an example in smart environmental monitoring

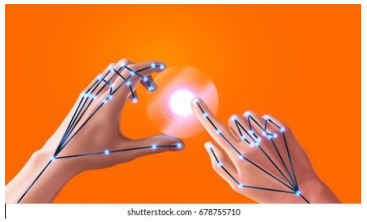
A rock-collapse forecasting system based on intelligent IoT and Edge Units (Northern Italy)

TinyML: the “always-on” Machine Learning

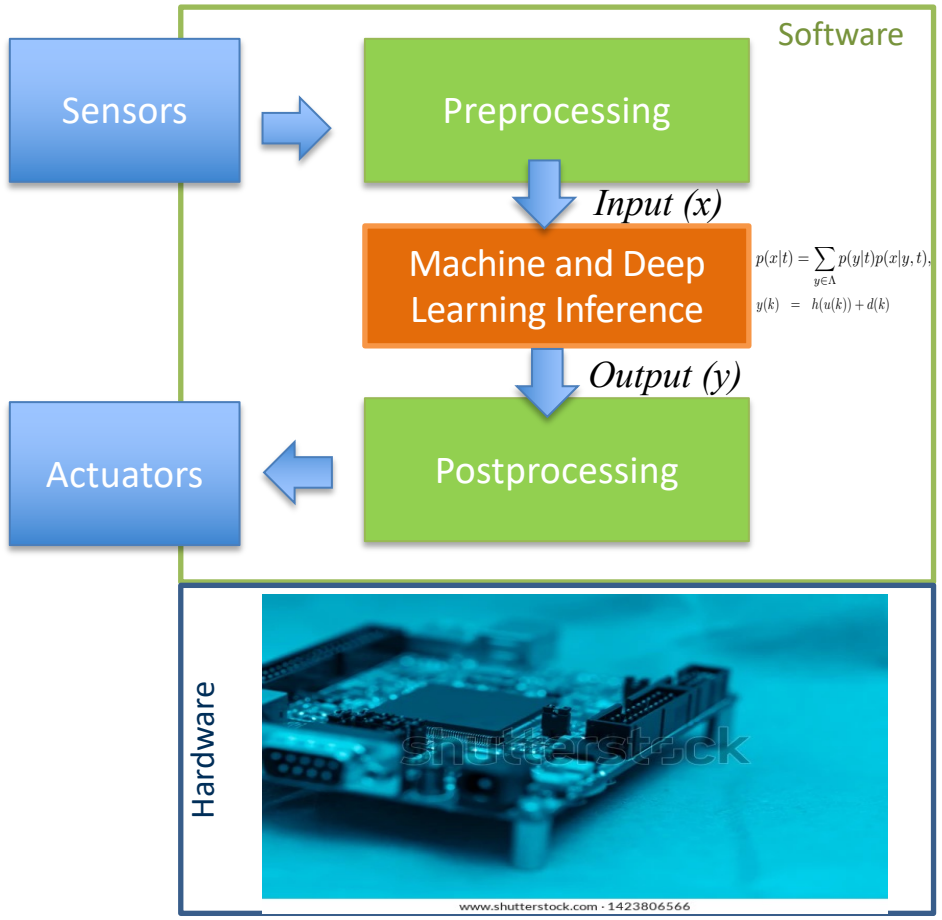
Wake-word detection



Person detection



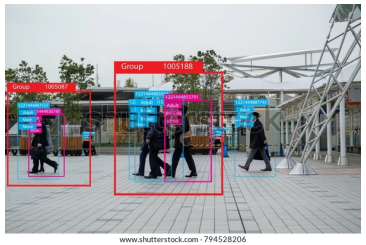
Gesture recognition



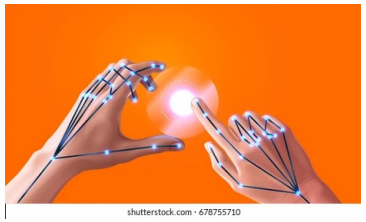
TinyML: the “always-on” Machine Learning

1) Changes in the environments

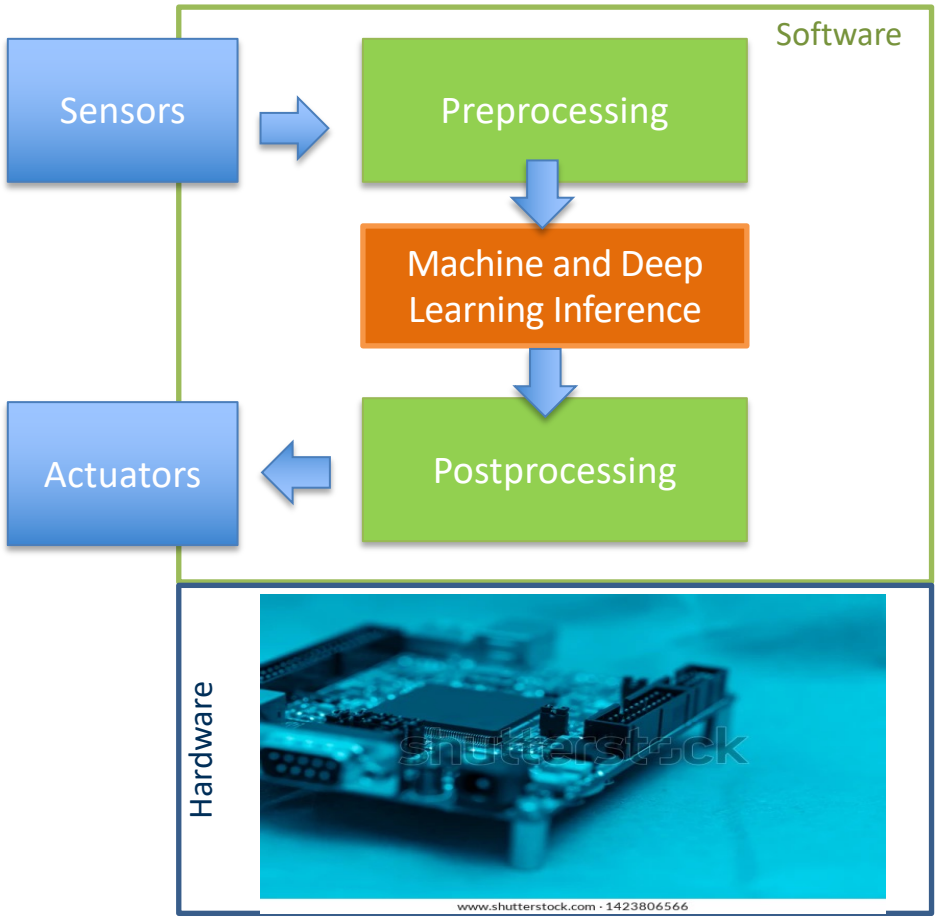
Wake-word detection



Person detection

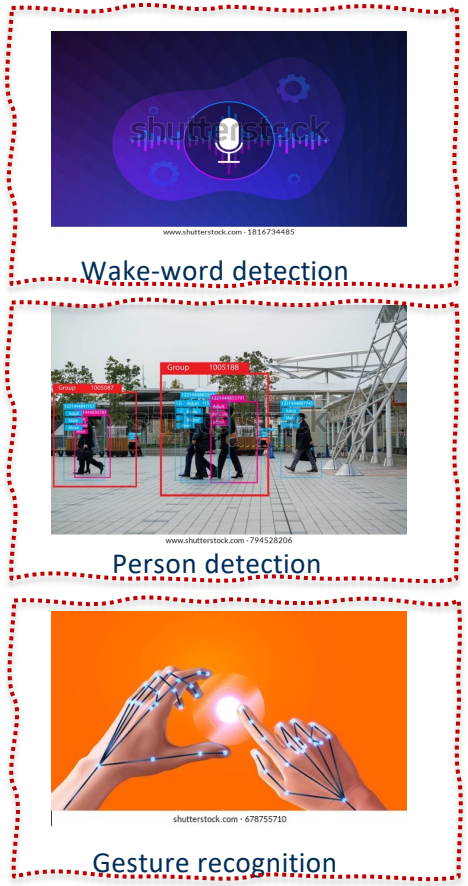


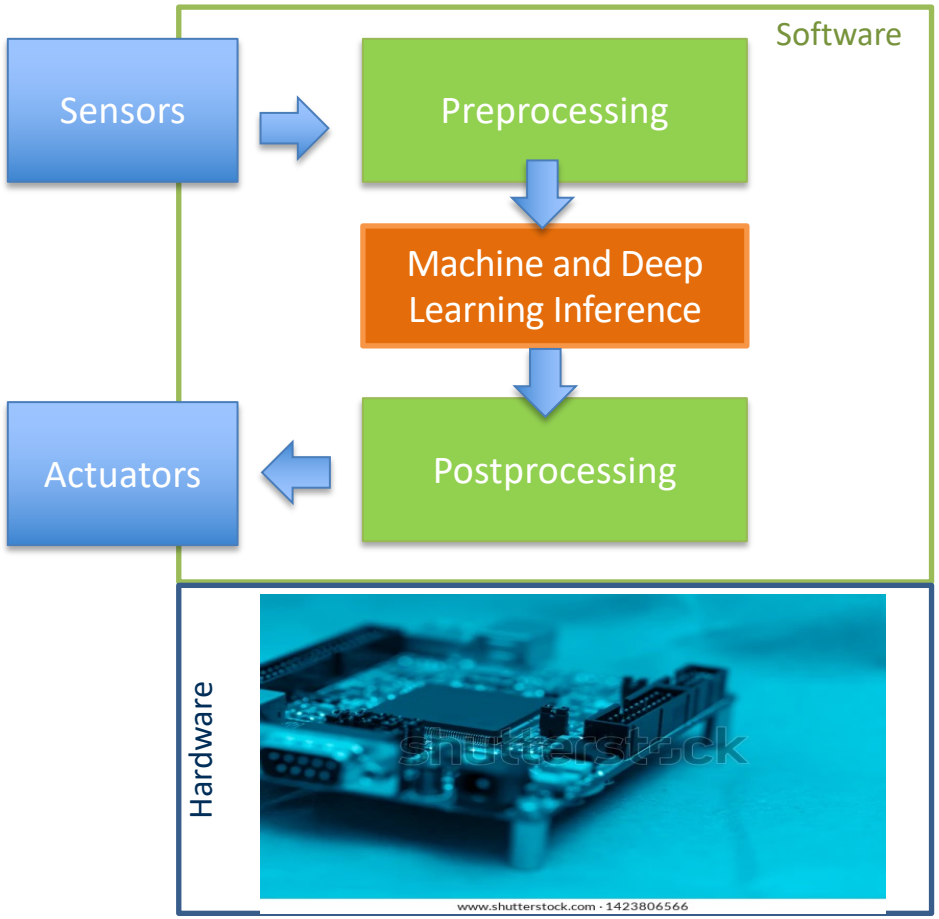
Gesture recognition



TinyML: the “always-on” Machine Learning

1) Changes in the environments
2) Changes in the users’ behaviour/interest

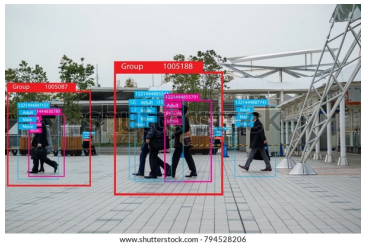




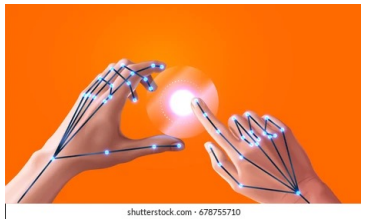
TinyML: the “always-on” Machine Learning

1) Changes in the environments
2) Changes in the users’ behaviour/interest
3) Faults/malfunctioning affecting the hardware or sensors/actuators

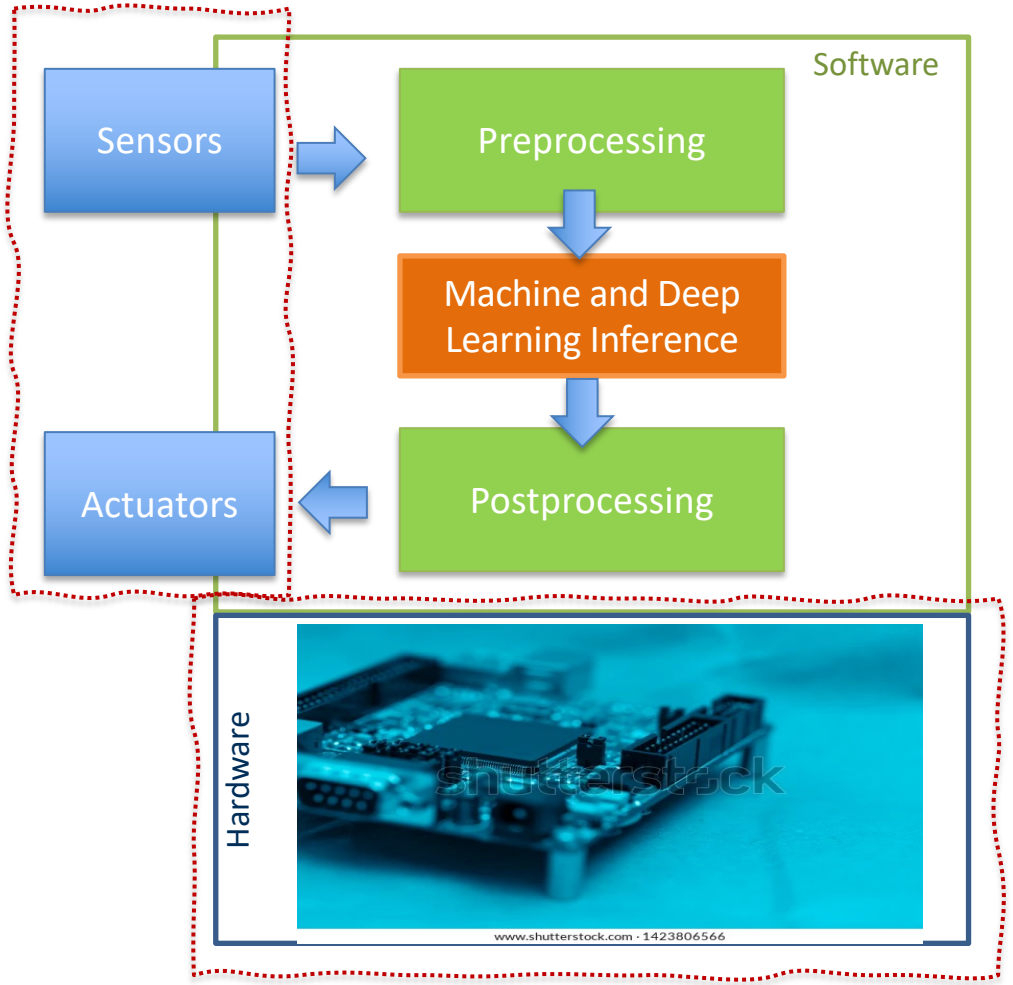
Wake-word detection



Person detection



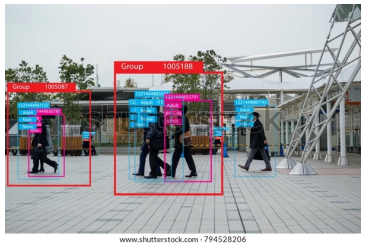
Gesture recognition



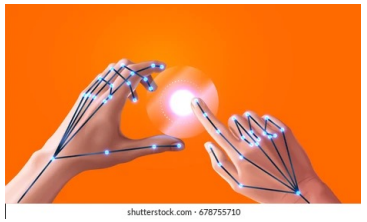
TinyML: the “always-on” Machine Learning

1) Changes in the environments
2) Changes in the users’ behaviour/interest
3) Faults/malfunctioning affecting the hardware or sensors/actuators
4) Ageing effects or thermal drifts affecting the sensors

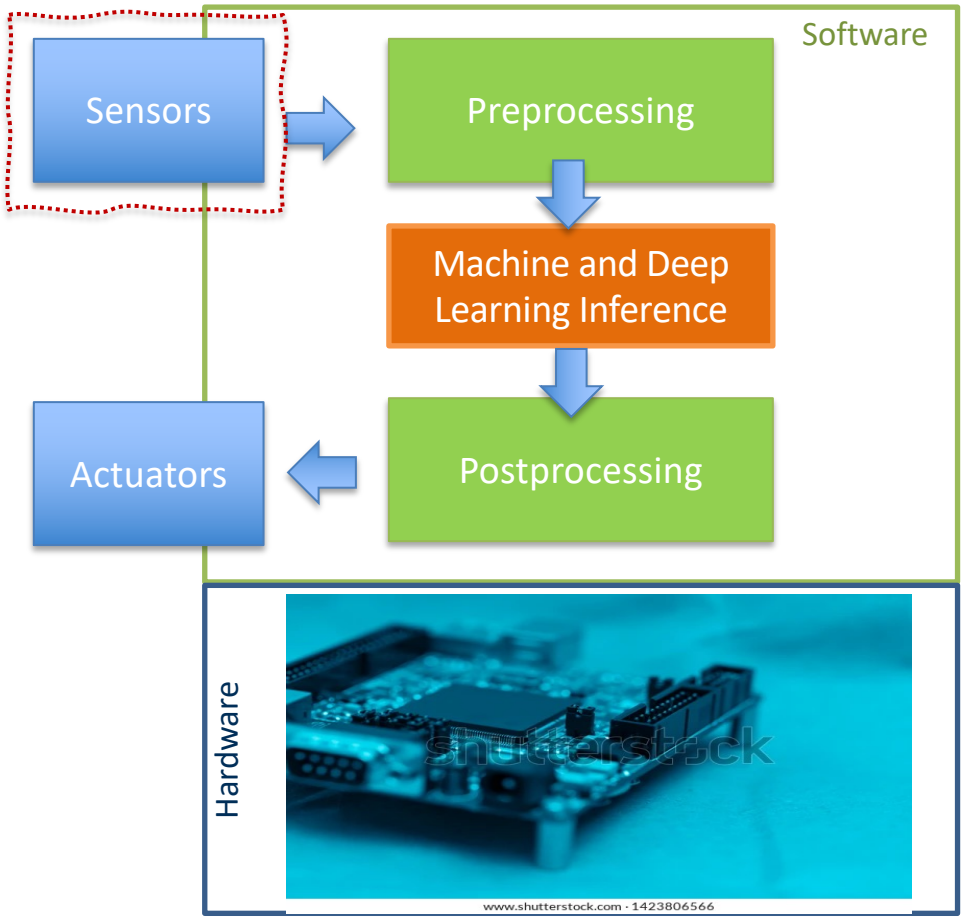
Wake-word detection



Person detection



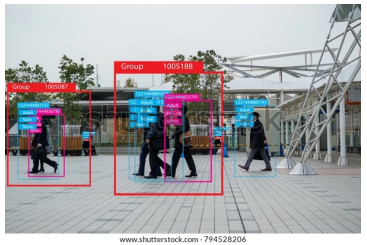
Gesture recognition



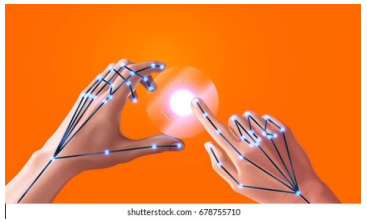
TinyML: the “always-on” Machine Learning

Perturbed, incorrect and missing data *can hence heavily* ***affect the subsequent processing phase*** *so as to possibly induce wrong decisions or on-the-field reactions.*

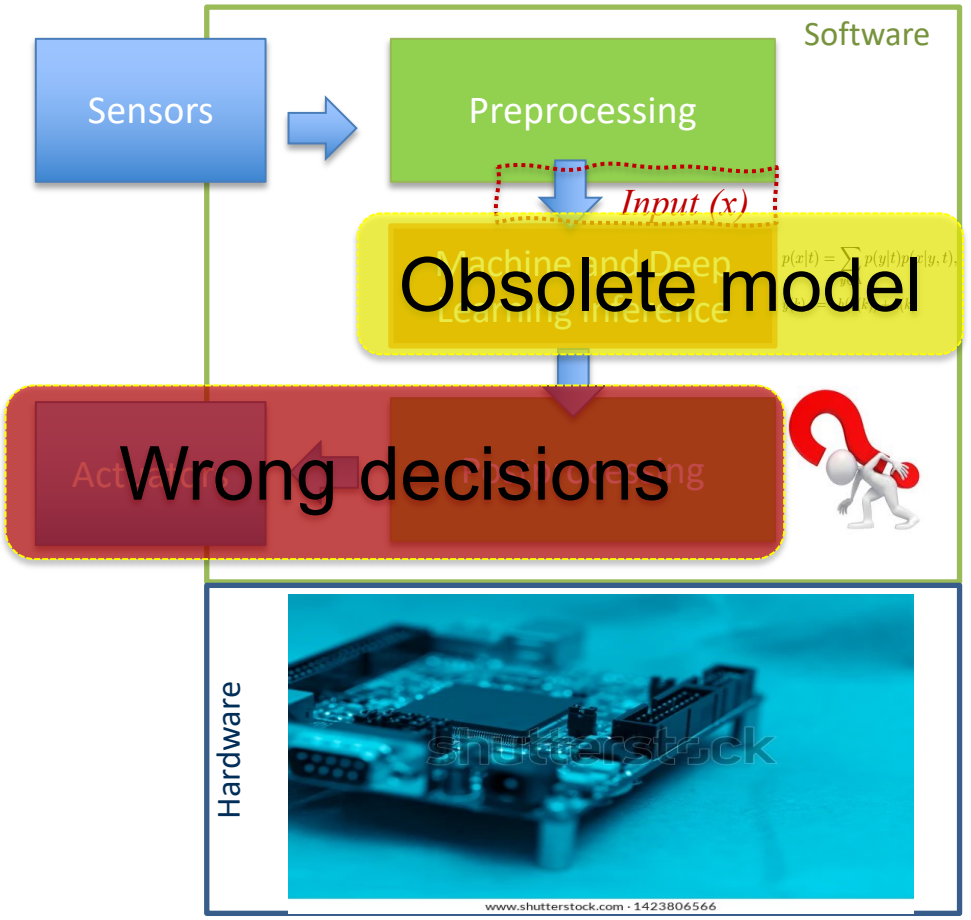
Wake-word detection



Person detection



Gesture recognition



Two (integrated) challenges to be addressed

Learning in presence of concept drift

Efficient on-device learning

Two (integrated) challenges to be addressed

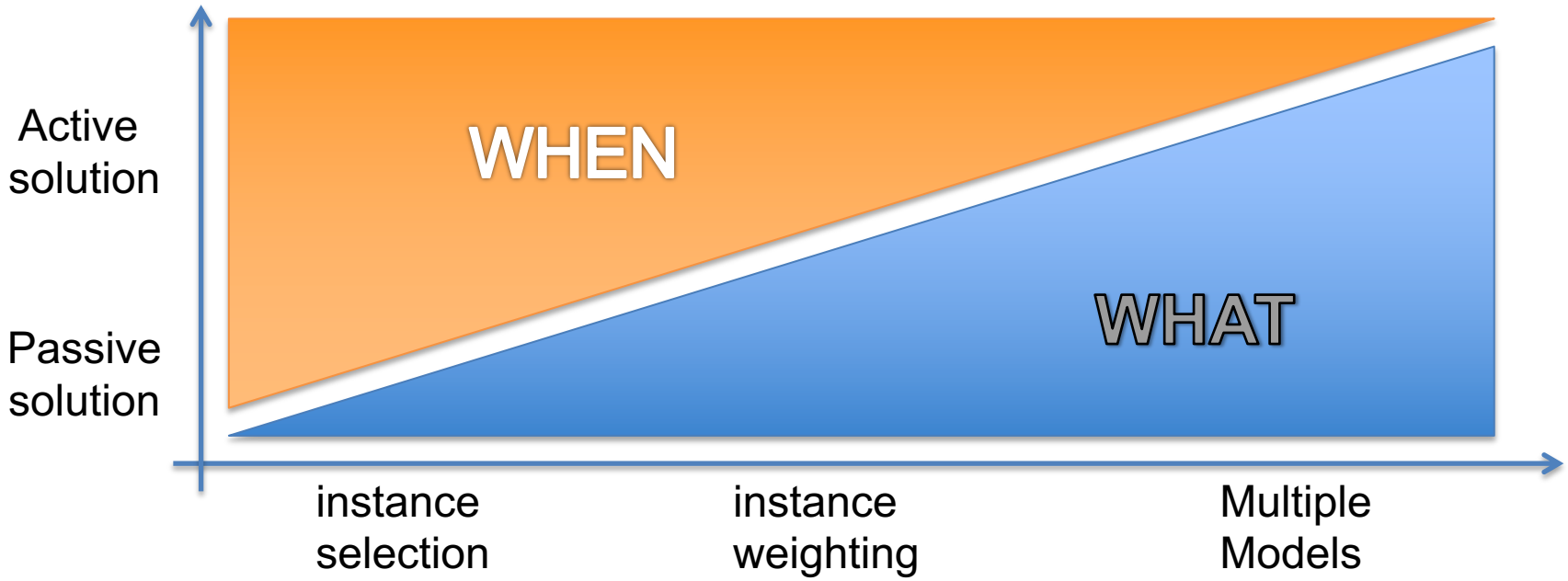
Learning in presence of concept drift

Efficient on-device learning

Already explored in the previous lecture

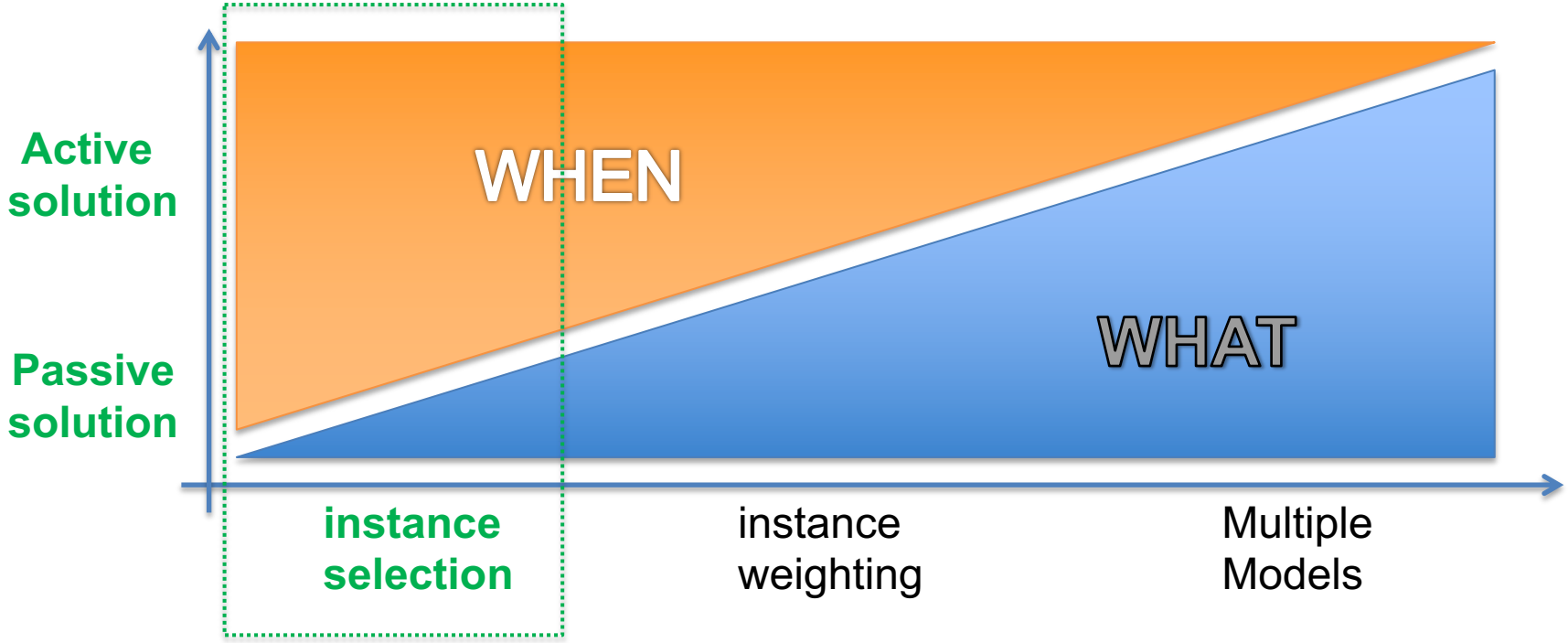
Adaptation is the game changer

- Traditional assumption: stationarity hypothesis
- Adaptive solutions in a non-stationary framework:



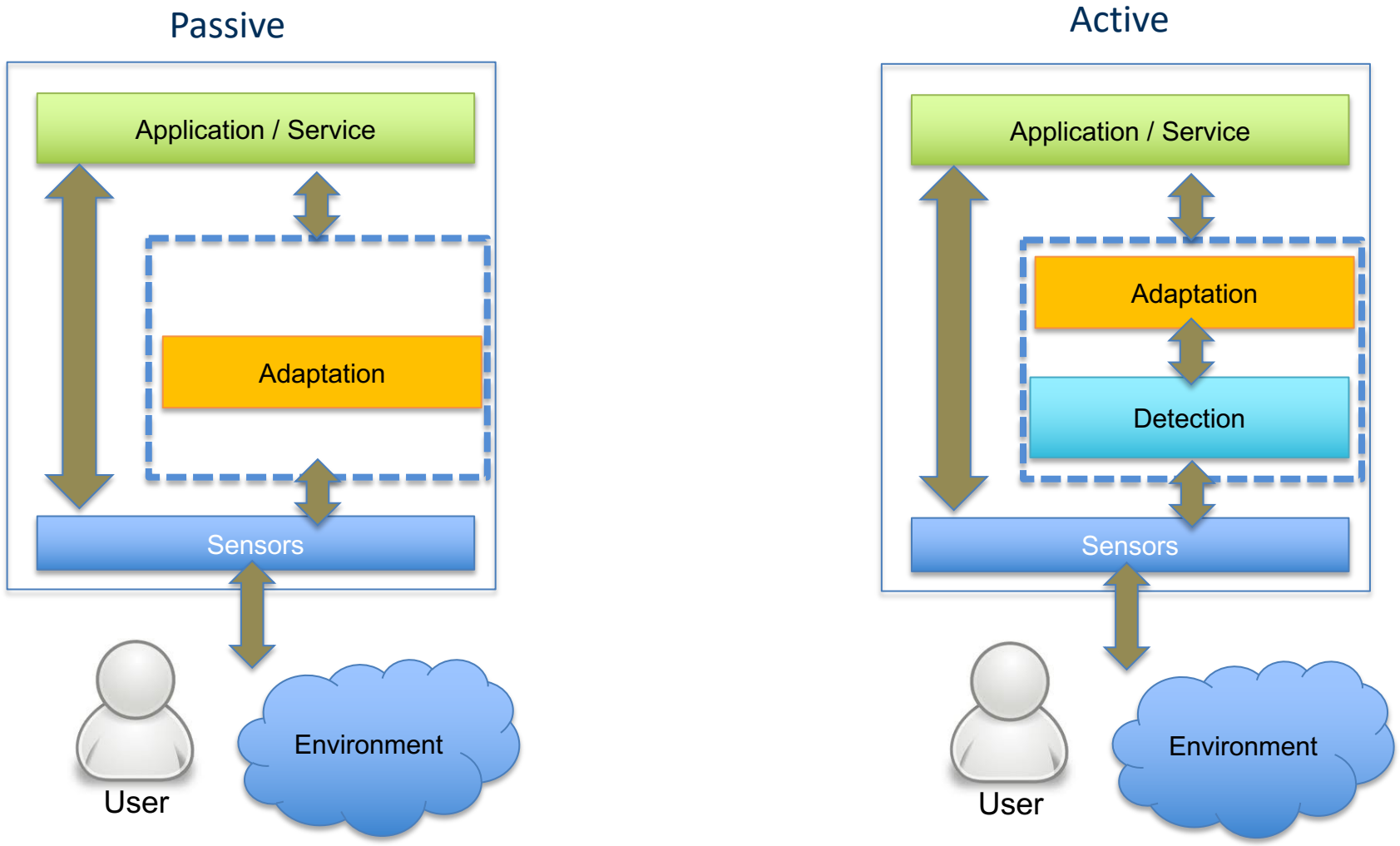
Adaptation is the game changer

- Traditional assumption: stationarity hypothesis
- Adaptive solutions in a non-stationary framework:



The TinyML perspective

Passive vs Active approaches

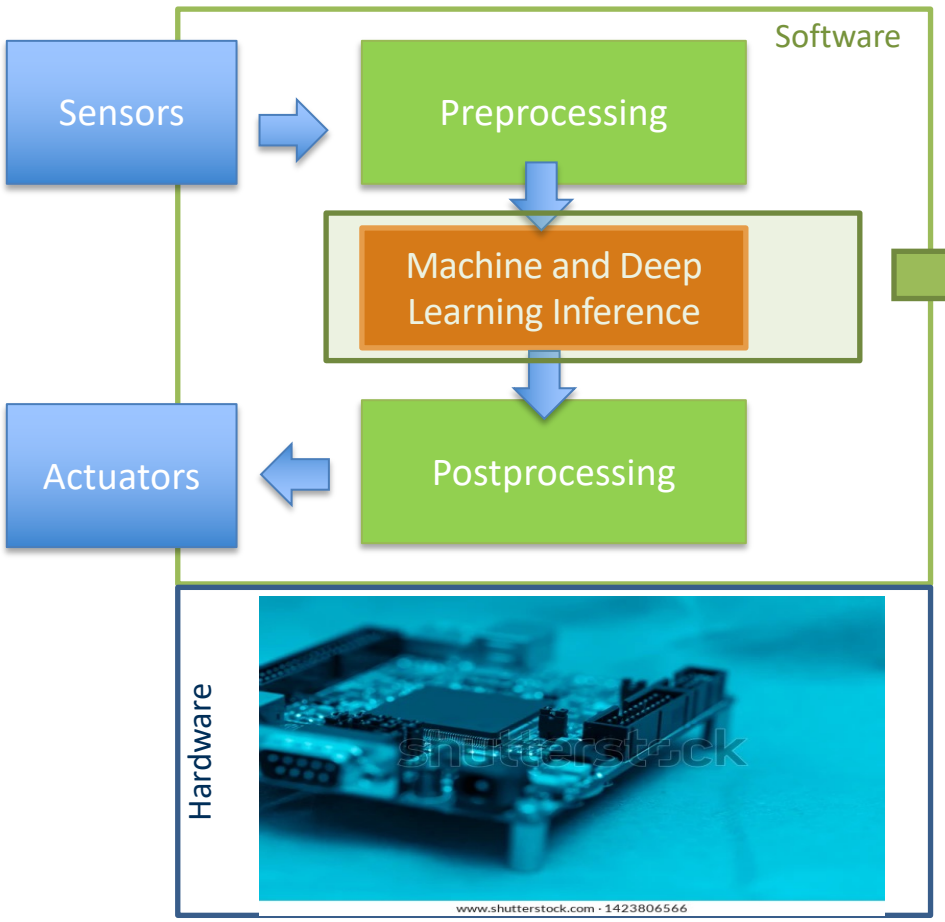


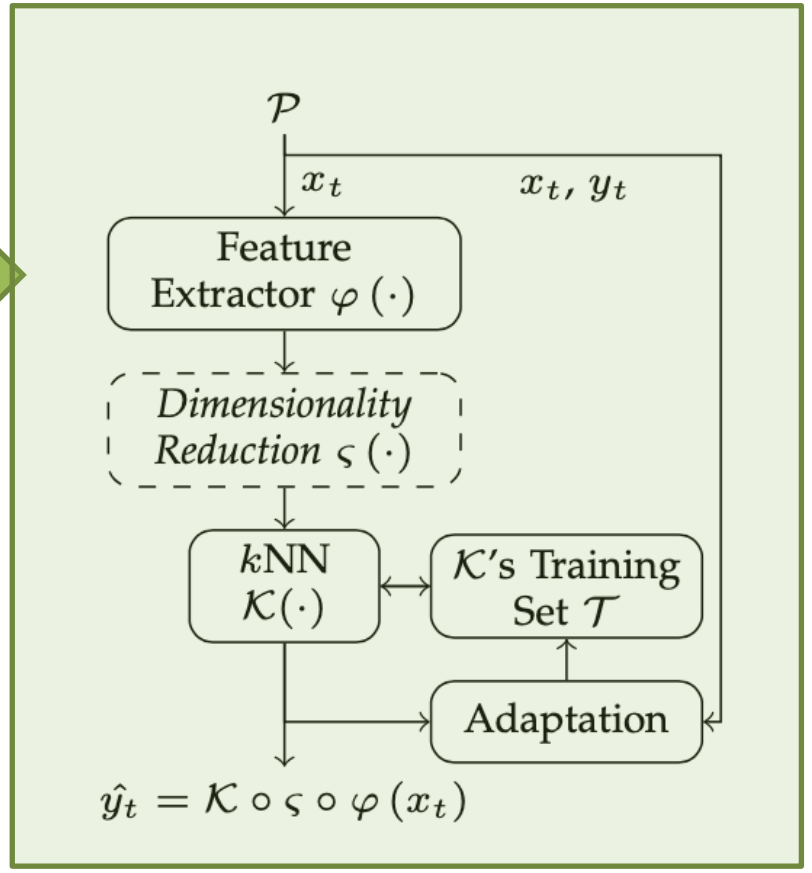
Two (integrated) challenges to be addressed

Learning in presence of concept drift

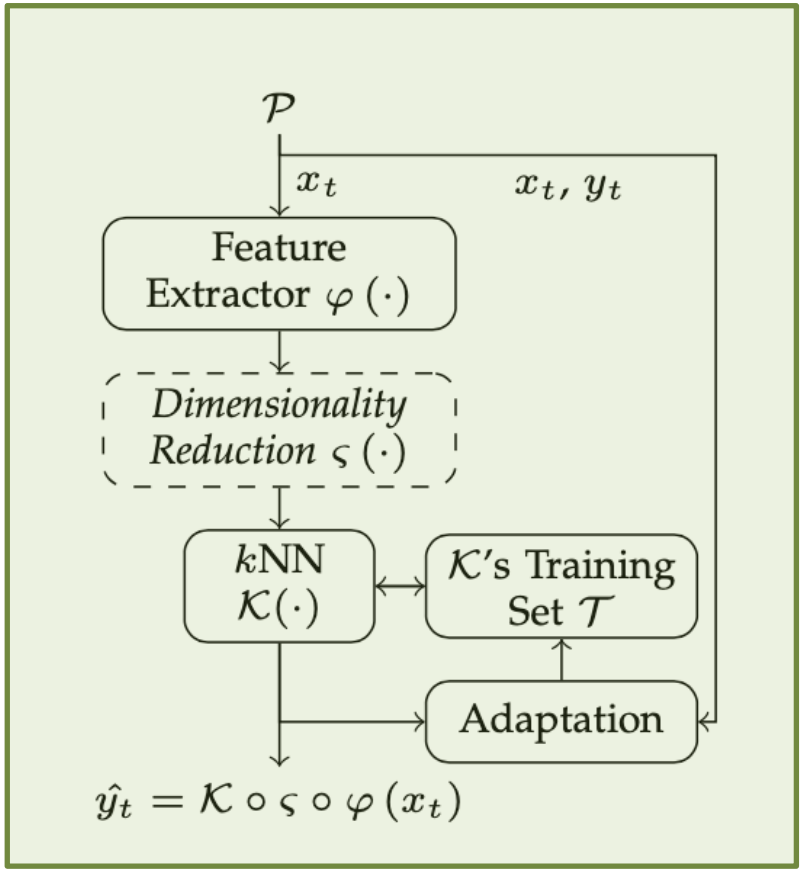
Efficient on-device learning

On-device TinyML for Concept Drift



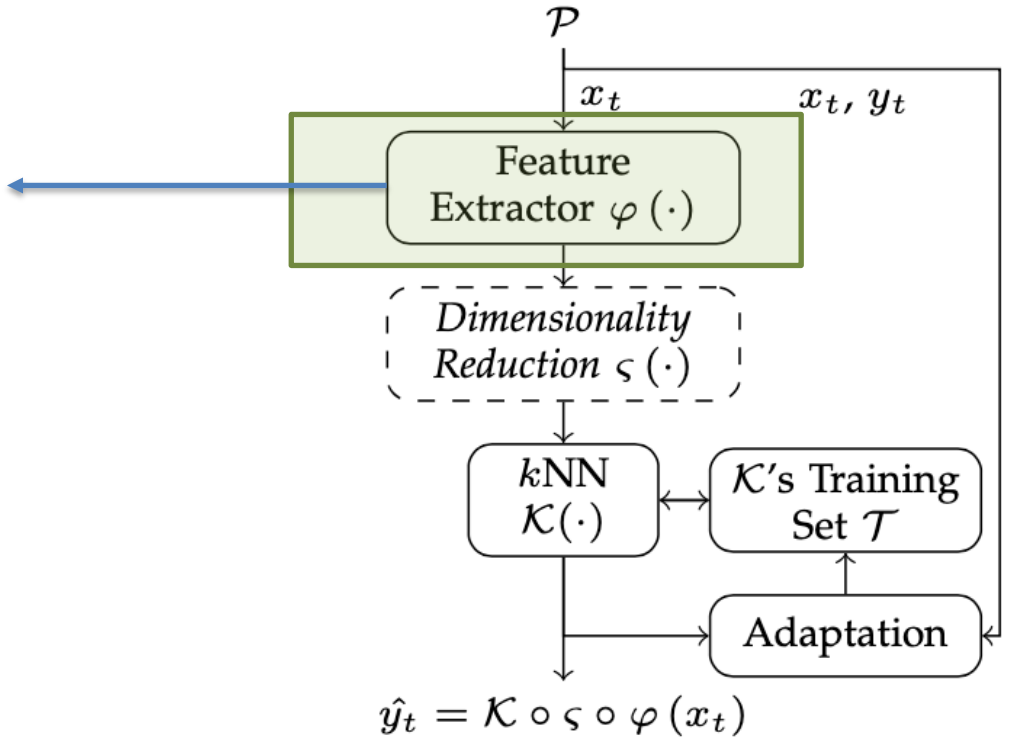


On-device TinyML for Concept Drift



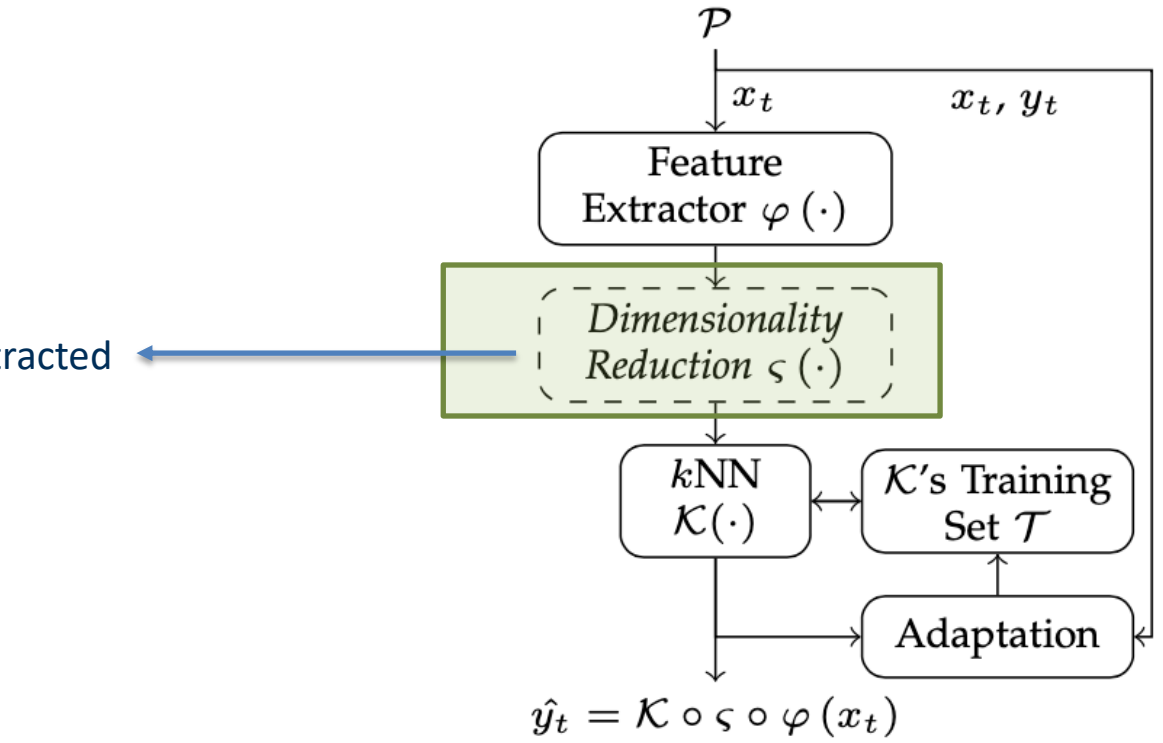
On-device TinyML for Concept Drift

- The Feature Extractor extracts features from the input x_t
- A pre-trained DL model (transfer learning) approximated by means of:
  - Task-Dropping
  - Precision Scaling



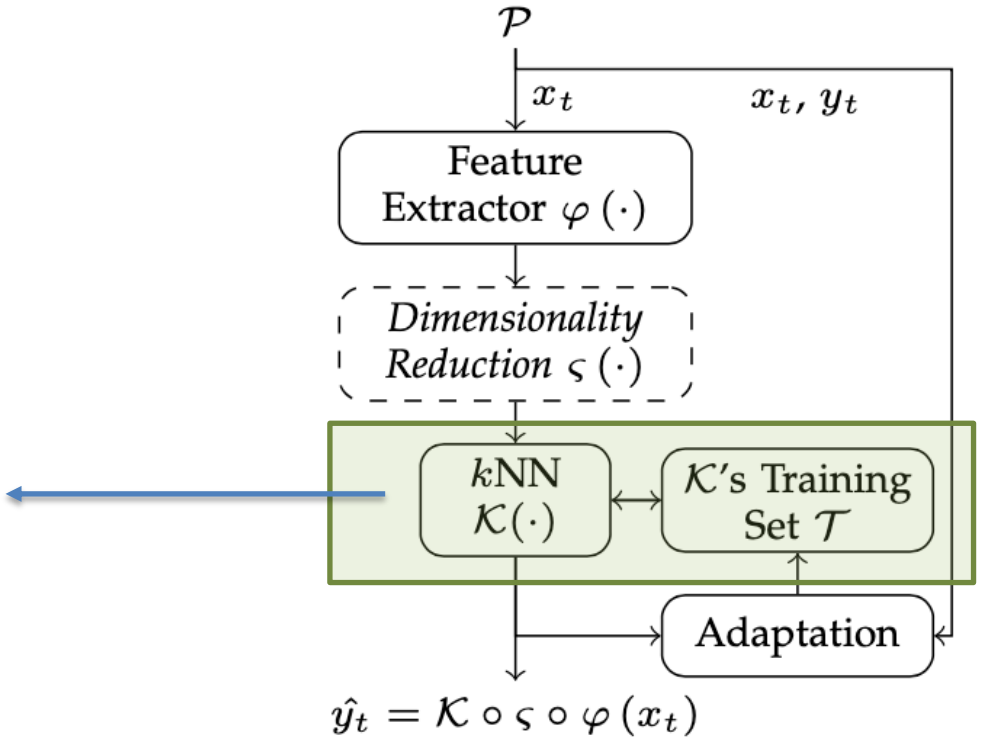
On-device TinyML for Concept Drift

- Reduces the dimensionality of extracted features (e.g., filter-selection mechanisms)



On-device TinyML for Concept Drift

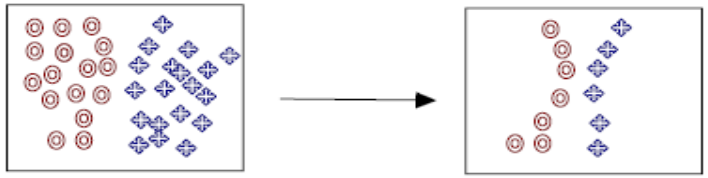
- kNN classifier
- **Pros: It does not require a training phase**
- **Cons: Memory demand and computational complexity**

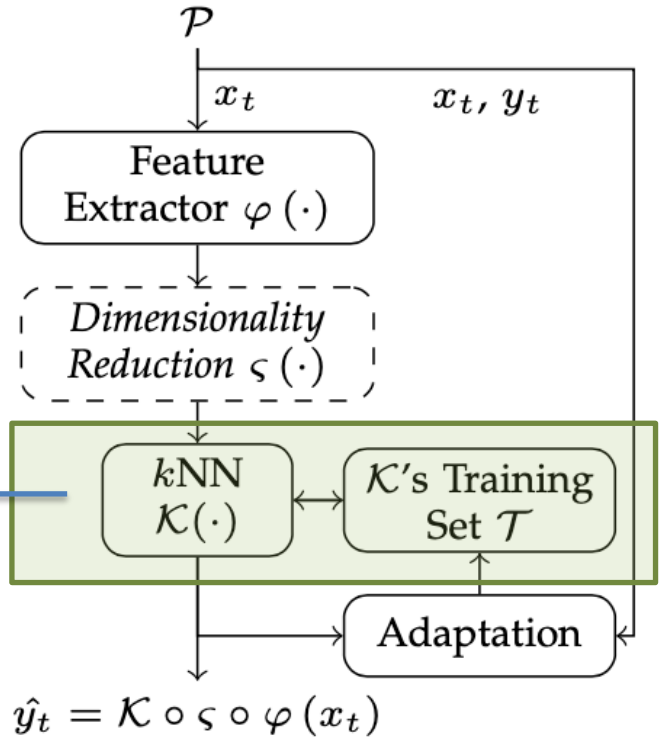


On-device TinyML for Concept Drift

- kNN classifier
- **Pros: It does not require a training phase**
- **Cons: Memory demand and computational complexity**

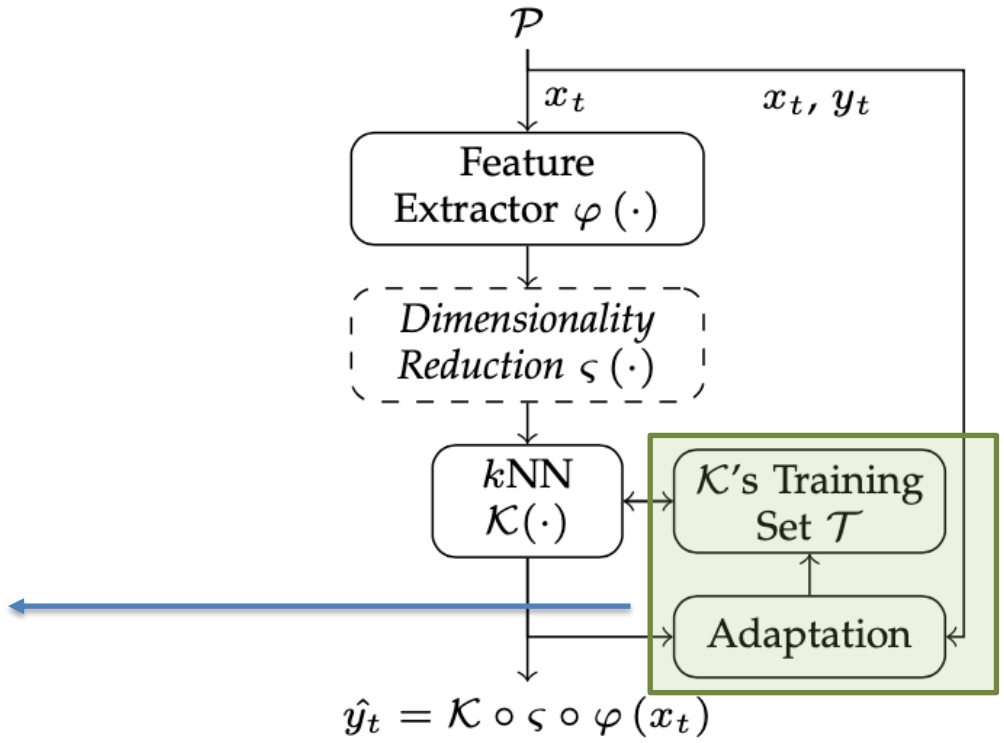
Condensing algorithm for kNN:
Identifying the smallest subset of training data that can correctly classify all the training samples





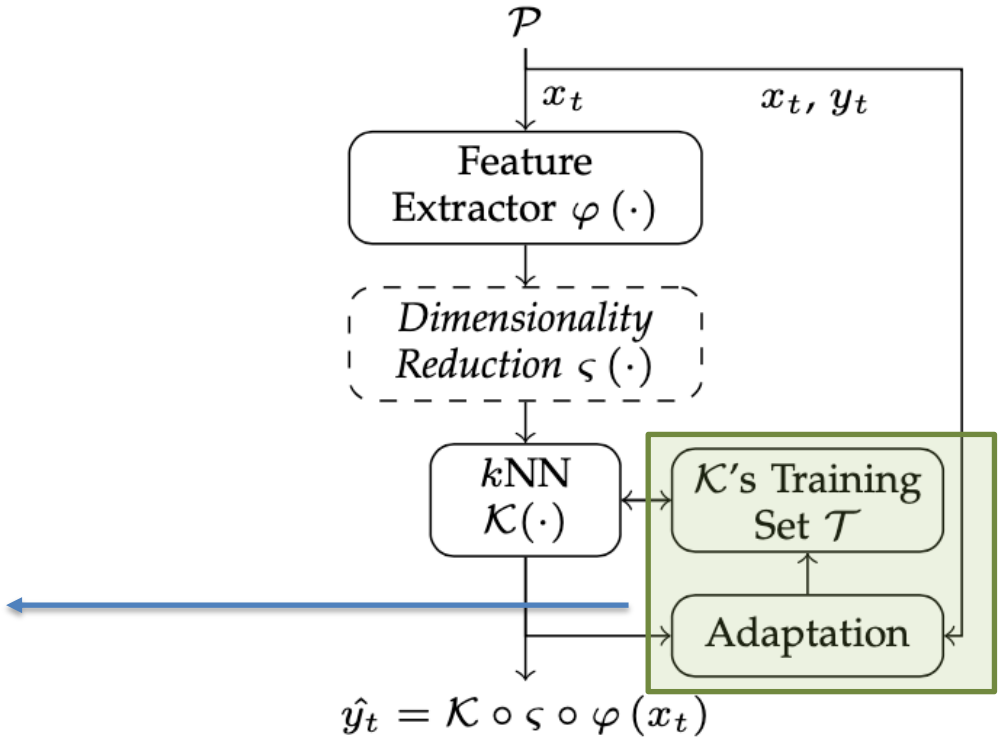
On-device TinyML for Concept Drift

- The adaptation module receives as input the sample x and the kNN prediction y
- **When the supervised information y_t is available, it updates the knowledge base of kNN**



On-device TinyML for Concept Drift

- **Passive Update**: the adaptation is carried out at each new supervised sample
- **Active Update**: the adaptation is triggered by a Change-detection test
- **Hybrid Update:** integrates passive and active update



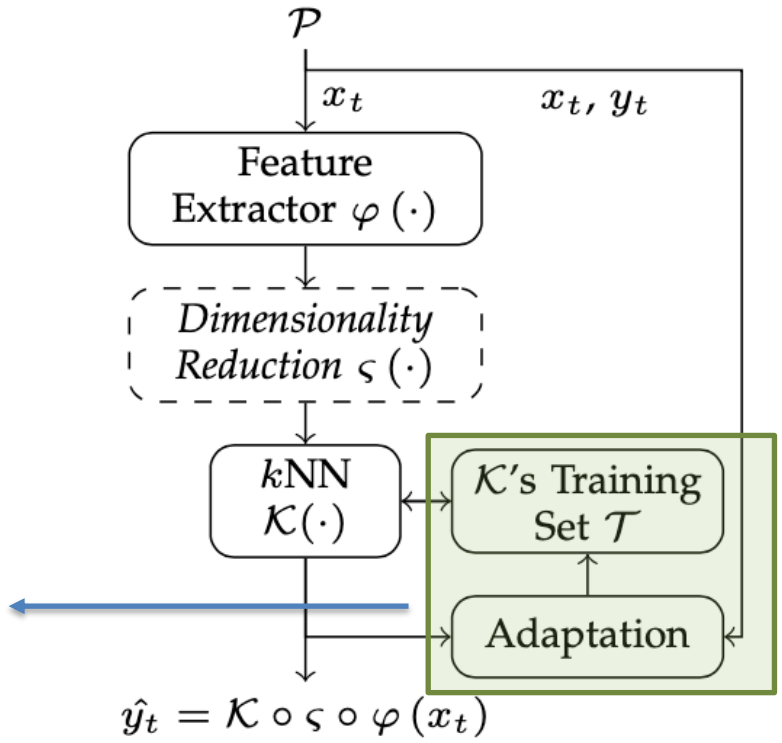
Passive update: condensing-in-time

- The adaptation is carried out at each new supervised sample

Algorithm 3: The Condensing-in-Time (Passive).

Input: Training Set $\mathcal{T}$, Feature Extractor $\varsigma \circ \varrho$.
Parameters : Maximum number of training samples p.

Compute $\mathcal{T} \leftarrow \bar{\mathcal{T}}$ with Algorithm 2. ▷ Condense $\mathcal{T}$.
Initialize the k-NN classifier $\mathcal{K}$ with $\varsigma \circ \varrho\,(\mathcal{T})$.
Define $\mathcal{D} = \mathcal{K} \circ \varsigma \circ \varrho$.
▷ Loop over samples arriving at time t.
foreach $(x_t, y_t) \sim \mathcal{P}, t = 1, 2, \ldots$ **do**
  Predict $\hat{y}_t \leftarrow \mathcal{D}(x_t)$
  if $\hat{y}_t \neq y_t$ **then** ▷ Passive Update.
    $\mathcal{T} \leftarrow \mathcal{T} \sqcup (x_t, y_t)$
    if $|\mathcal{T}| > p$ **then** ▷ Window Size Check.
      $(x_{\tilde{t}}, y_{\tilde{t}}) \leftarrow \arg\min_{\bar{t}} \{(x_{\bar{t}}, y_{\bar{t}}) \in \mathcal{T}\}$
      $\mathcal{T} \leftarrow \mathcal{T} \setminus \{(x_{\tilde{t}}, y_{\tilde{t}})\}$
    Update $\mathcal{D}$ with $\mathcal{T}$.



Active update: Active Tiny *k*NN

- **T**he adaptation is triggered by a CUSUM-based change-detection mechanism

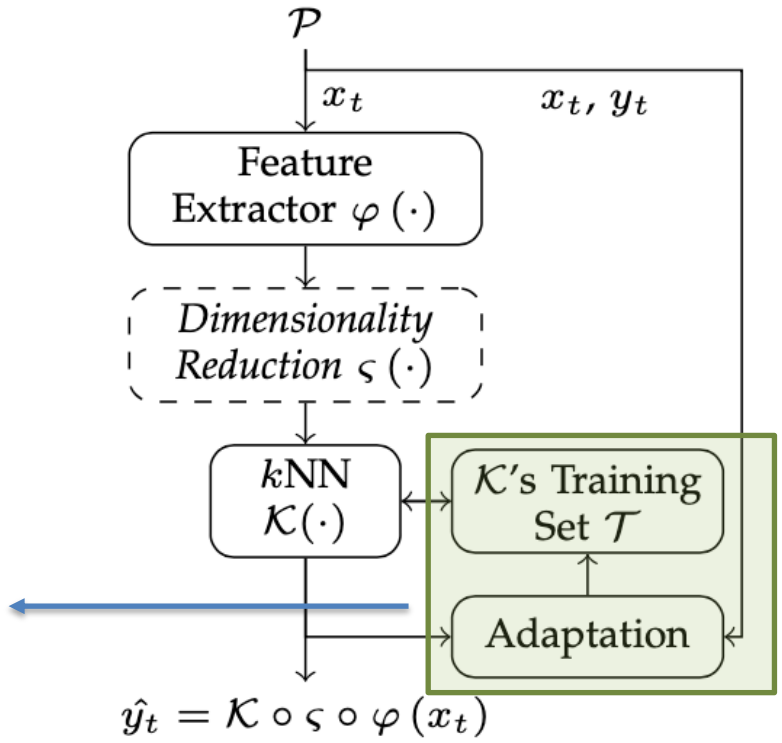
$$s_t = \ln \frac{p_{\upsilon_1}(\zeta_t)}{p_{\upsilon_0}(\zeta_t)}, \qquad S_j^t(\upsilon_1) = \sum_{i=j}^{t} s_i,$$

$$g_t = \vartheta(s_1, \ldots, s_t) = \max_{1 \le j \le t} \sup_{\upsilon_1 \in \Upsilon_1} S_j^t(\upsilon_1),$$

MultiDim-CUSUM cdt

$$(\tilde{j}, \tilde{\upsilon}_1) = \arg\max_{1 \le j \le t} \sup_{\upsilon_1 \in \Upsilon_1} S_j^t(\upsilon_1).$$

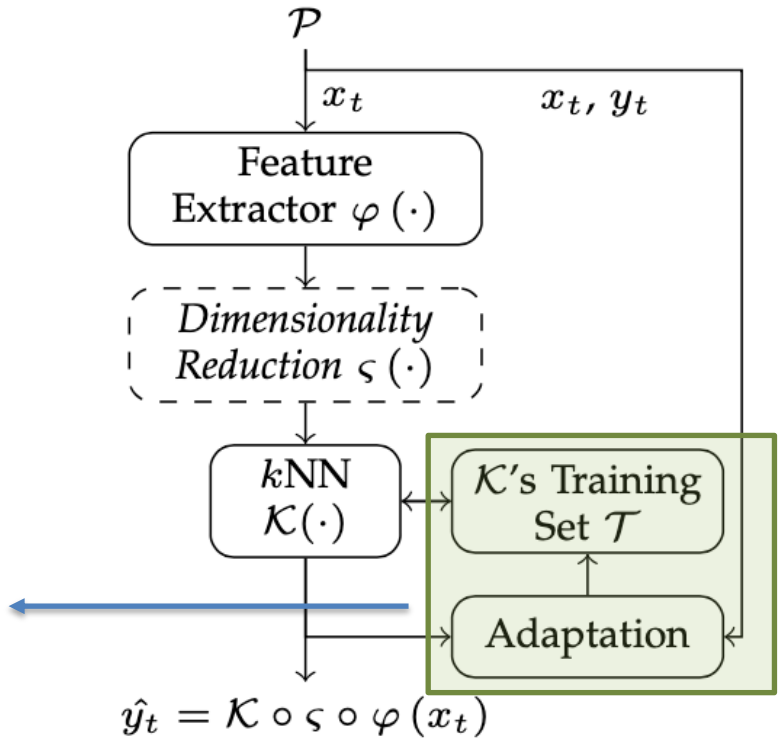
Estimation of the change time instant



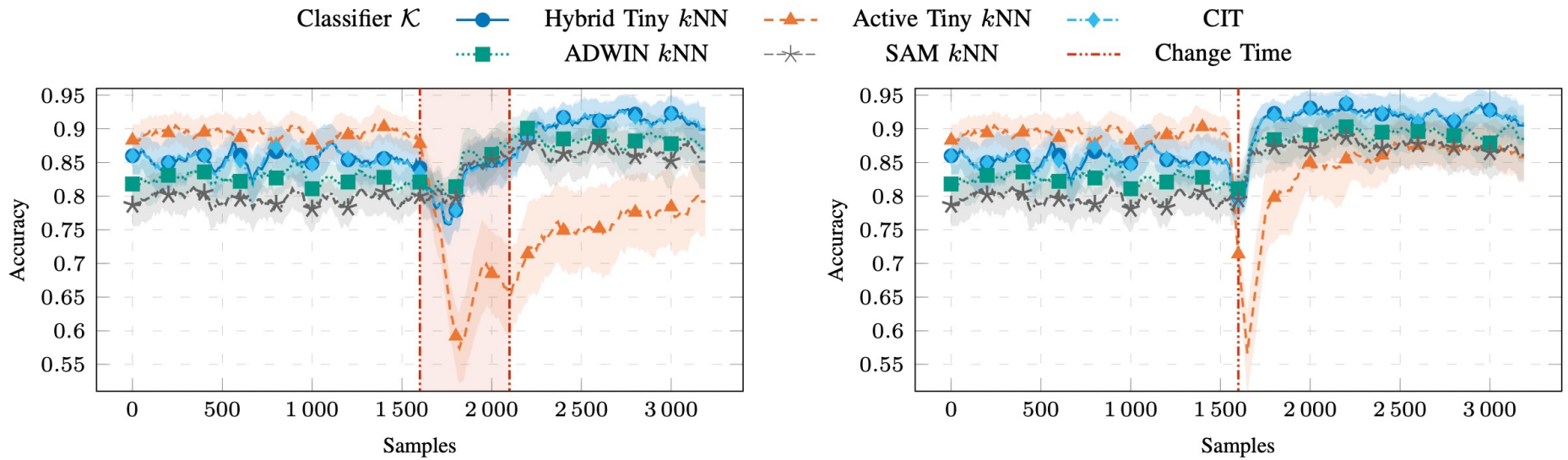
Hybrid update

- **Integrating passive and active updates**

- The hybrid update continuously adapts $\mathcal{T}$ over time thanks to the passive adaptation
- The active adaptation present in the hybrid update can quickly discard obsolete knowledge when a change is detected and set a bound on the memory footprint of $\mathcal{T}$.



Some relevant results (Speech Command Identification)



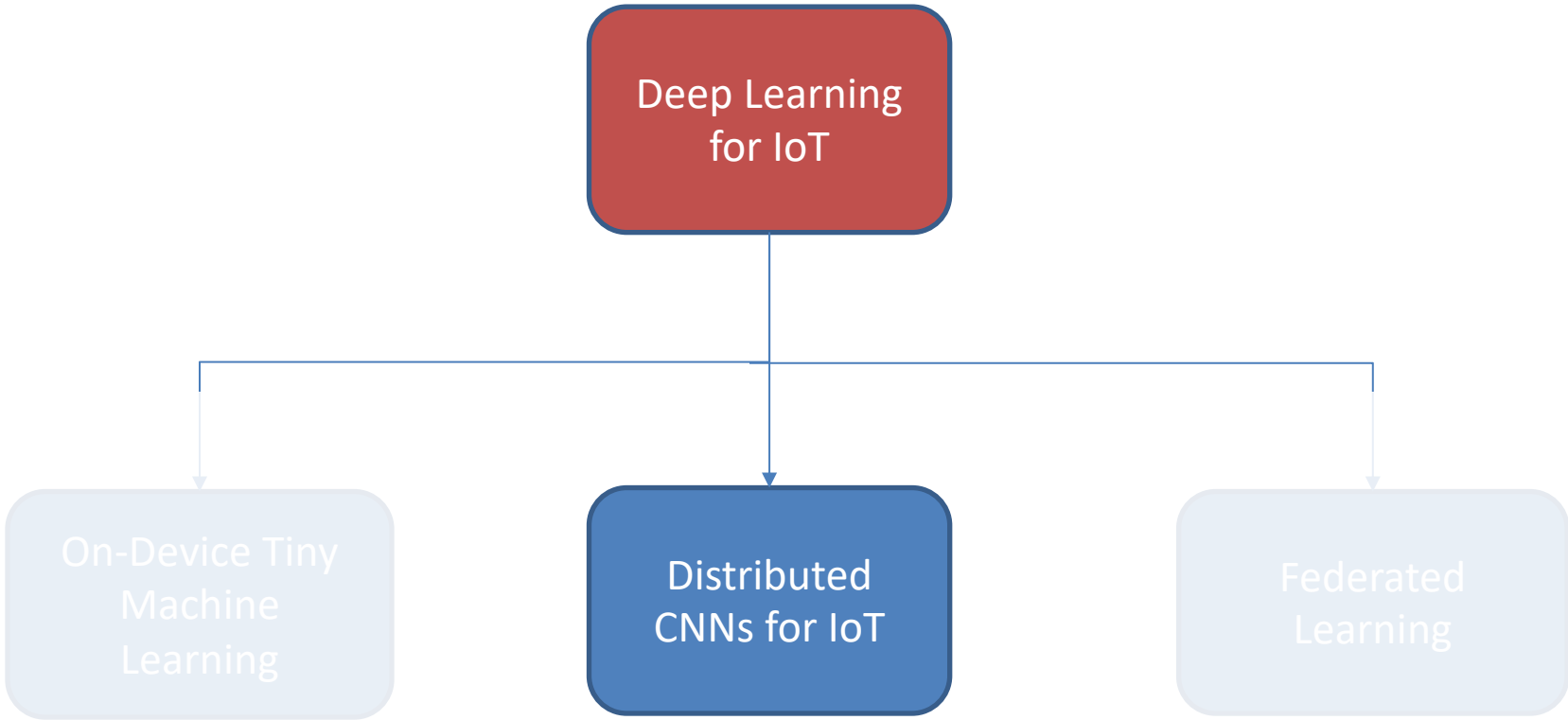
Concept drift refers to the introduction of noise (distortion, reverb, and echoes)

MCU		Execution Times (ms) t_p	$t_{\varsigma \circ \varrho}$	$t_{\mathcal{K},10}$	$t_{\mathcal{K},50}$
STM32H743ZI	480Mhz 1024KB RAM	22.9	18.6	2.0	2.9
STM32F767ZI	216Mhz 512KB RAM	53.8	41.5	2.2	4.0
STM32F401RE	84Mhz 96KB RAM	34.6	43.1	2.2	4.6
		Spectrogram Processing	*Feature Extraction*	*KNN classification with 10 and 50 samples*	

Two current research directions

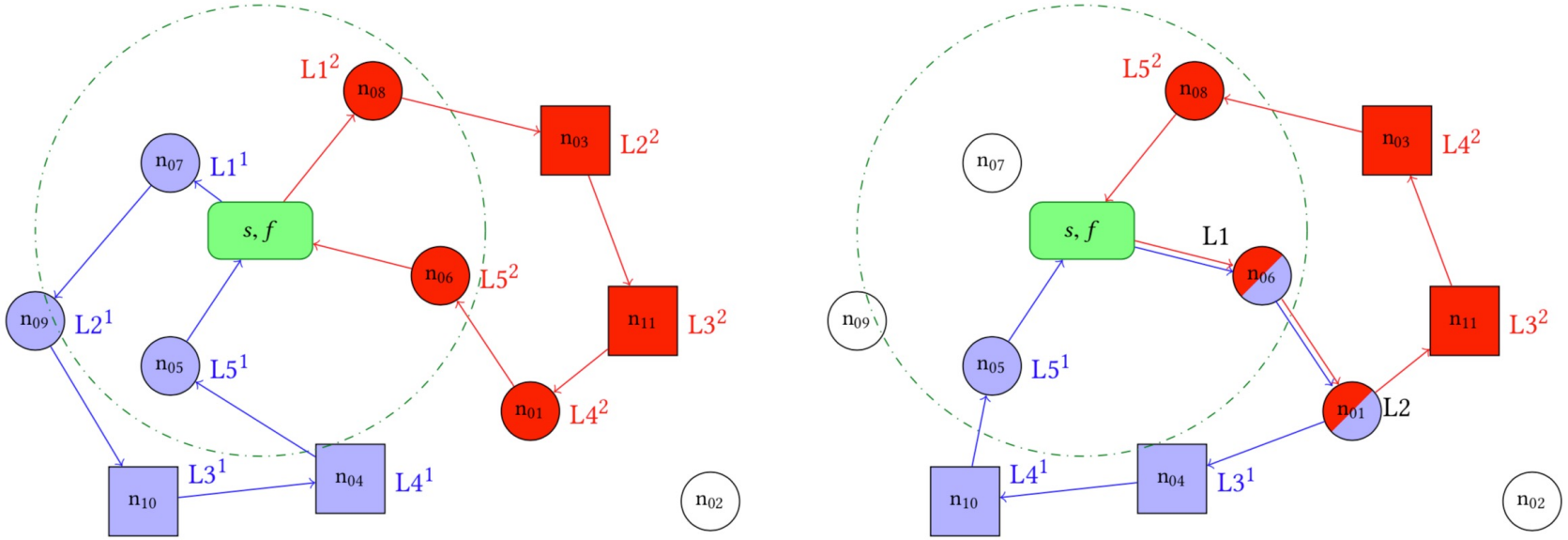
Semi Supervised TinyML

Automatic design, development and deployment for On-line Tiny Machine Learning


Deep Learning for IoT
On-Device Tiny Machine Learning
Distributed CNNs for IoT
Federated Learning

Distributed CNNs for IoTs

Identify the optimal allocation of CNN layers on a set of heterogeneous IoT units



The figure of merit aims at minimizing the latency between when the image is acquired and the decision is taken by the sink

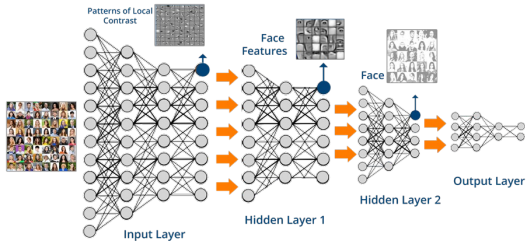
$$\sum_{u=1}^{C}\sum_{i=1}^{N}\sum_{k=i}^{N}\sum_{j=1}^{M-1}\alpha_{u,i,j}\cdot\alpha_{u,k,j+1}\cdot p_{u,j+1}\cdot\frac{K_{u,j}}{\rho}\cdot d_{i,k}$$

$$+\sum_{i=1}^{N}t_i^{(p)}+t_s+t_f$$

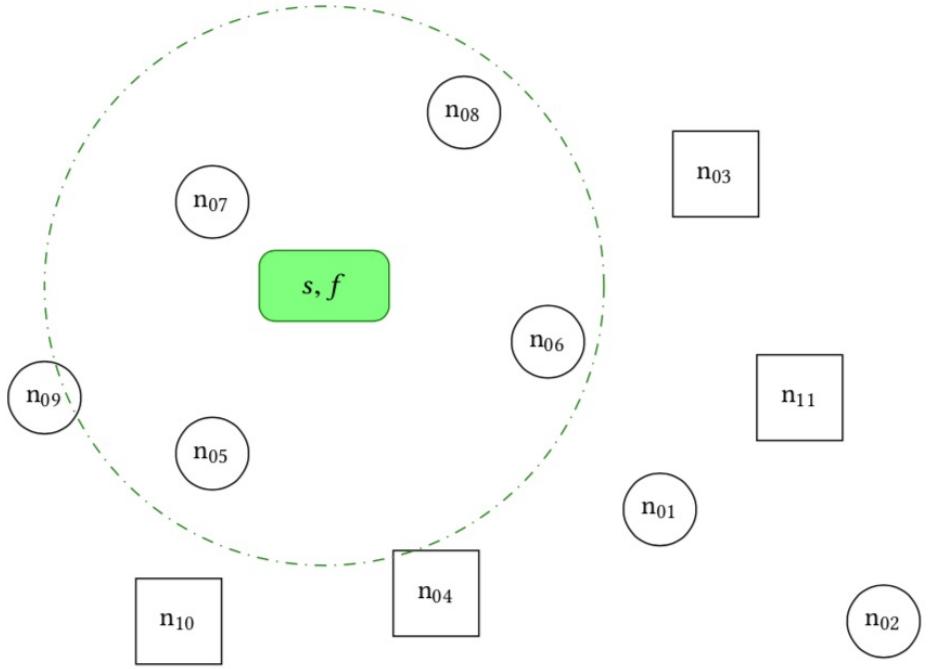
Disabato, Simone, Manuel Roveri, and Cesare Alippi. "Distributed Deep Convolutional Neural Networks for the Internet-of-Things." *arXiv preprint arXiv:1908.01656* (2019). Currently under revision.

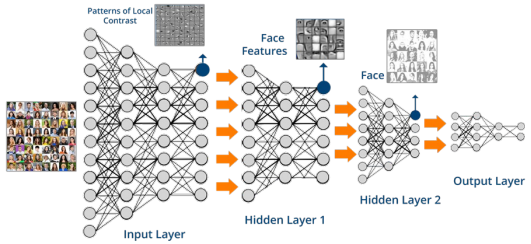
An example in the smart home

n_{07}

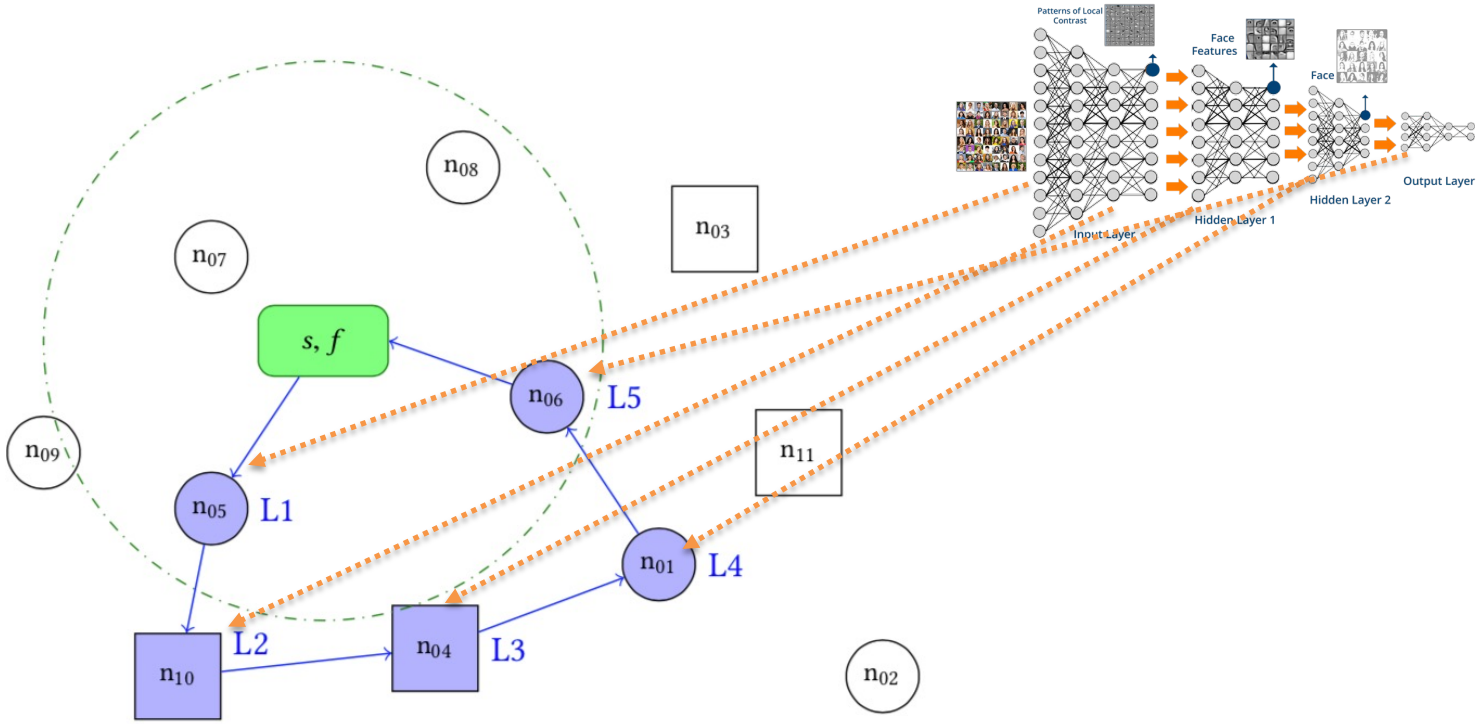


An example in the smart home

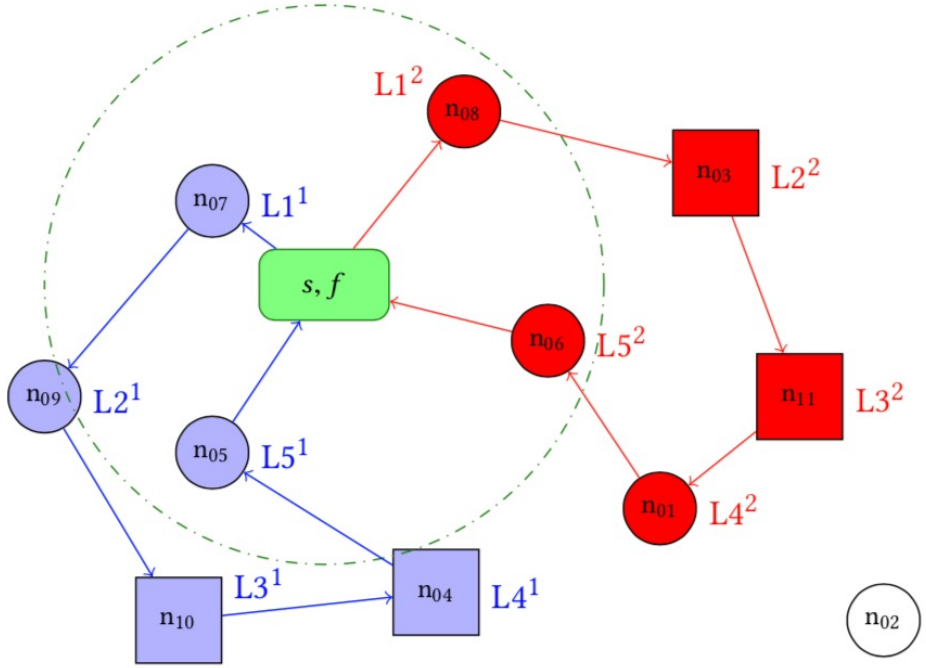


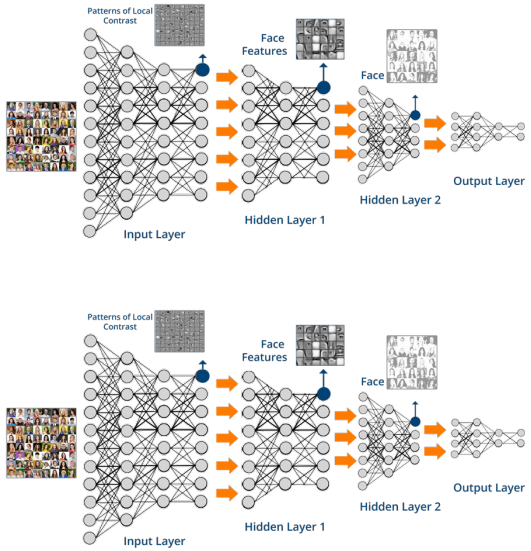


An example in the smart home

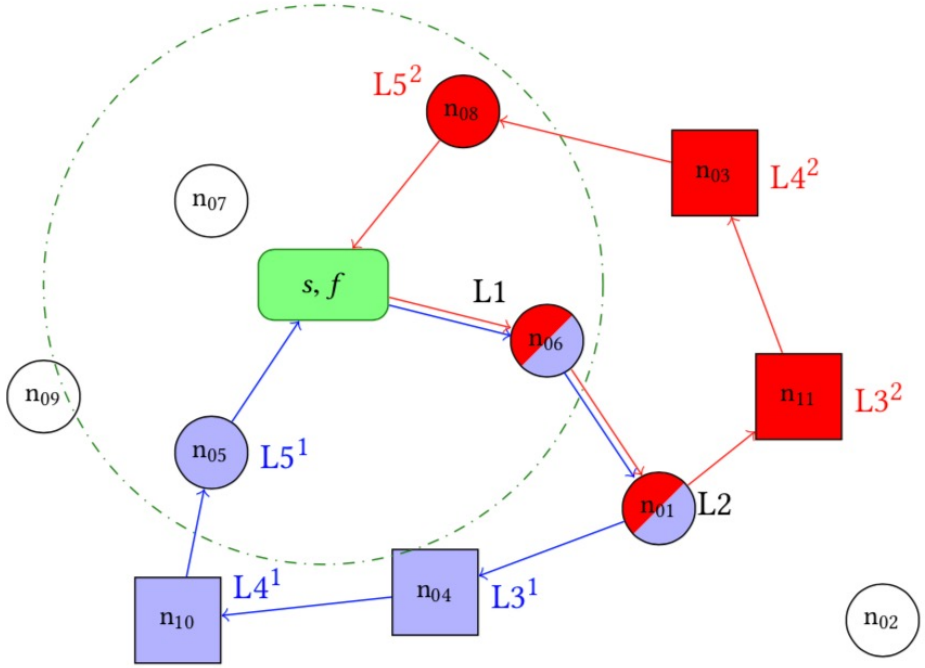


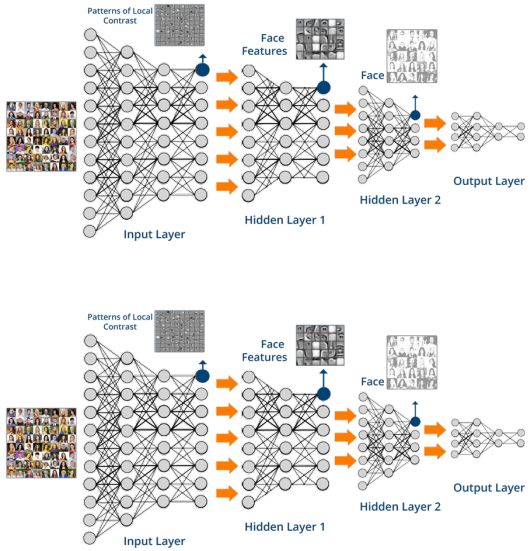
An example in the smart home



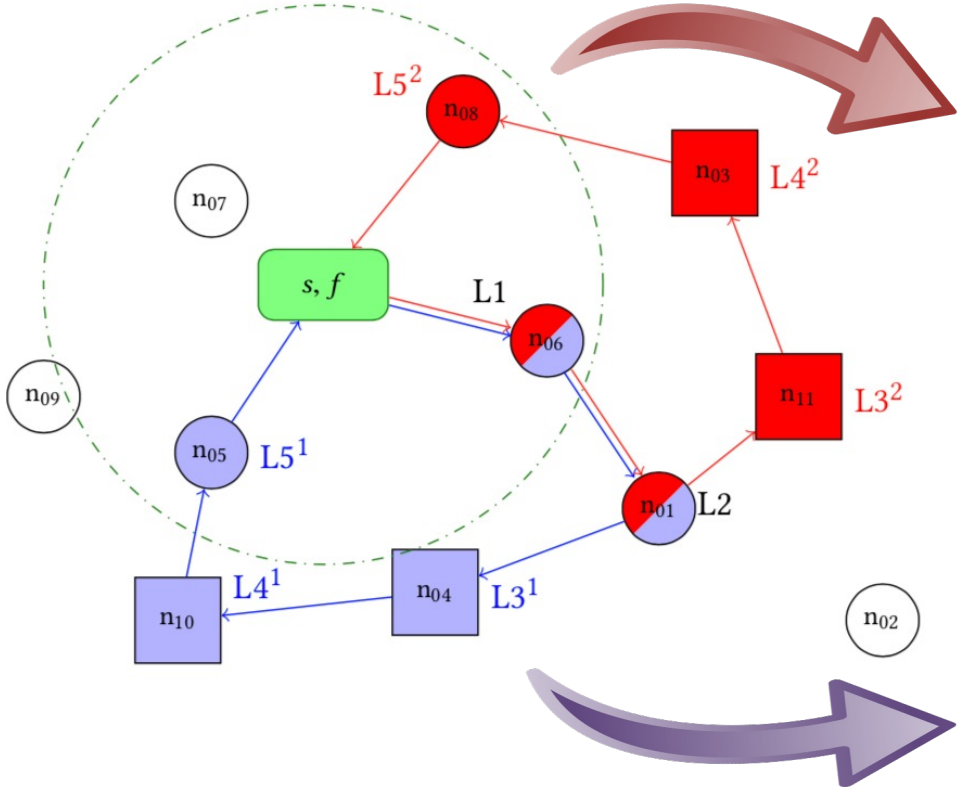


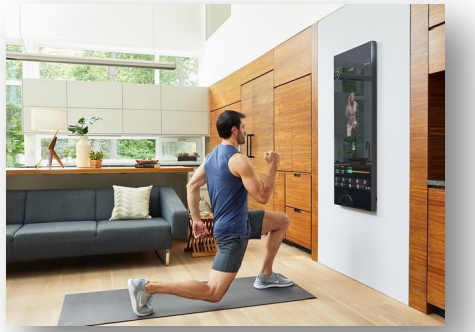
An example in the smart home

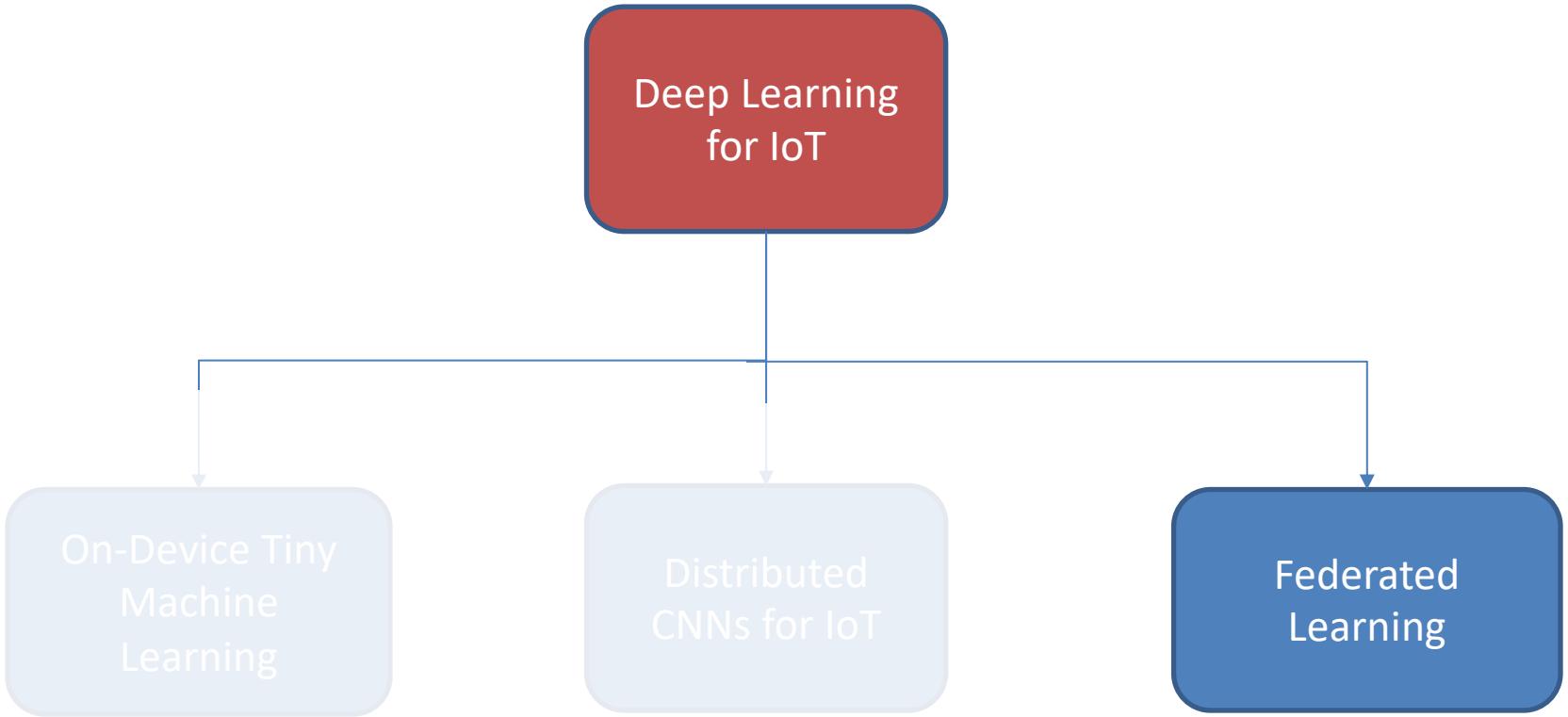




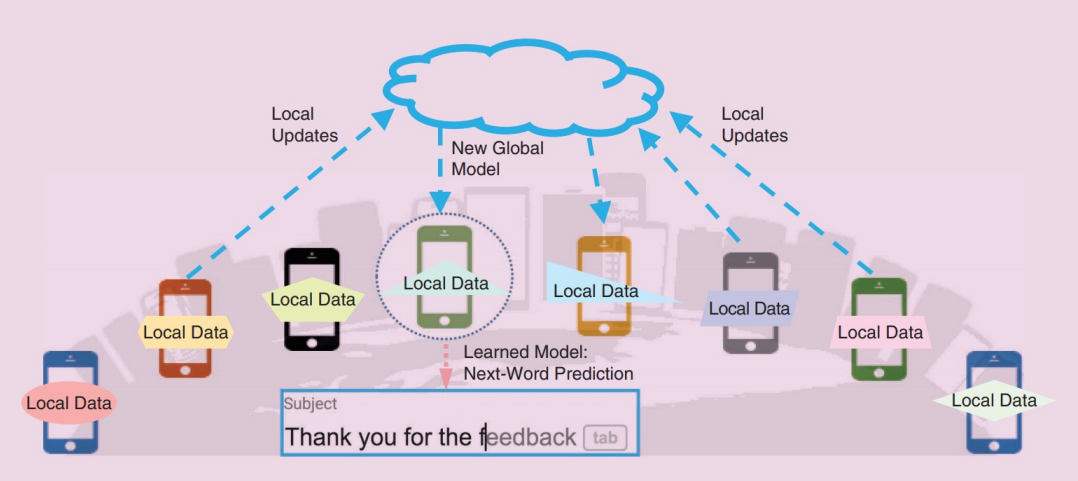
An example in the smart home



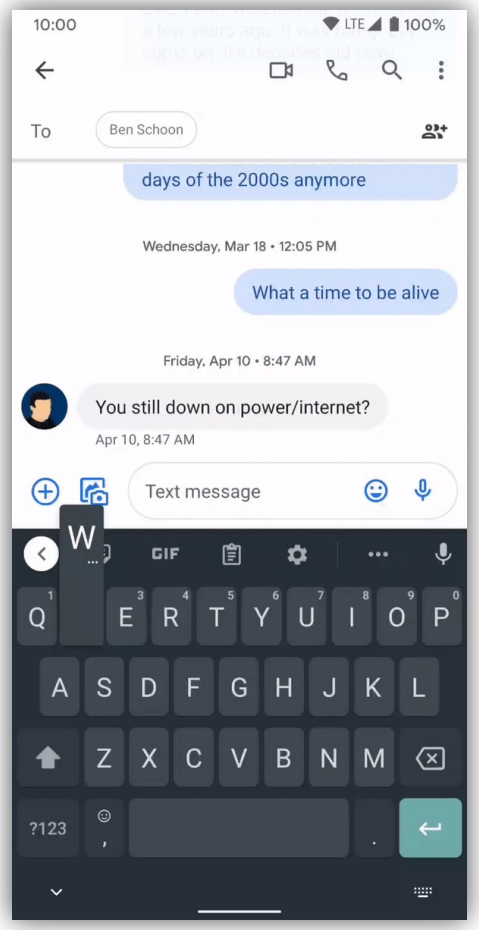



Deep Learning for IoT
On-Device Tiny Machine Learning
Distributed CNNs for IoT
Federated Learning

The idea of Federated Learning



Federated Learning for Next-Word prediction.
T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions", IEEE Signal Processing Magazine, vol. 37, no. 3, pp. 50-60, 2020.



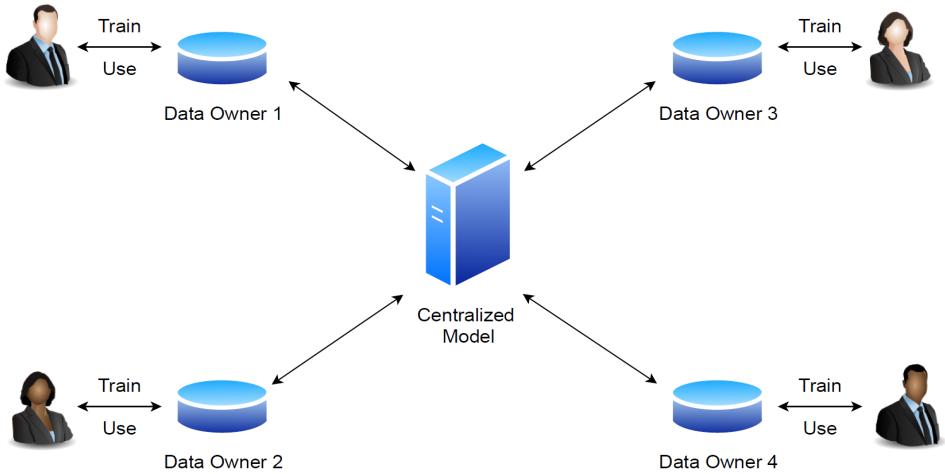
G-Board Next-Words prediction. Image from https://mspoweruser.com/google-starts-testing-smart-compose-in-gboard-for-android/

The Scenario of Federate Learning

- ✓ **Pervasively Distributed devices** (smartphones, IoT devices)
- ✓ **Privacy Issues, Sensitive Data** (personal, medical, banking data)
- ✓ **User-needs require cooperation** (data and computational power)
- ✓ **System Heterogeneity** (devices can be different in HW characteristics)
- ✓ **Statistical Heterogeneity** (possible non-independent and identically distributed (non-IID) data across the set of devices)
- ➢ **Non-Stationary Environment** (seasonality effects, changes in users' behaviour)

An overview of Federated Learning

- **Privacy Preserving Technique:** Data don't leave the devices (On-Device Data Set)
- **Communication:** The connection between devices and server is only used to send model's updates
- **Heterogeneity:** System and Statistical Heterogeneity
- **Training Data:** large number of participant devices, and each of them contains a different amount and type of samples. The number of devices can be much bigger than the number of training samples on each device
- **Secure Aggregation:** Homomorphic Encryption, Secure Multi-Party Computation and Differential Privacy



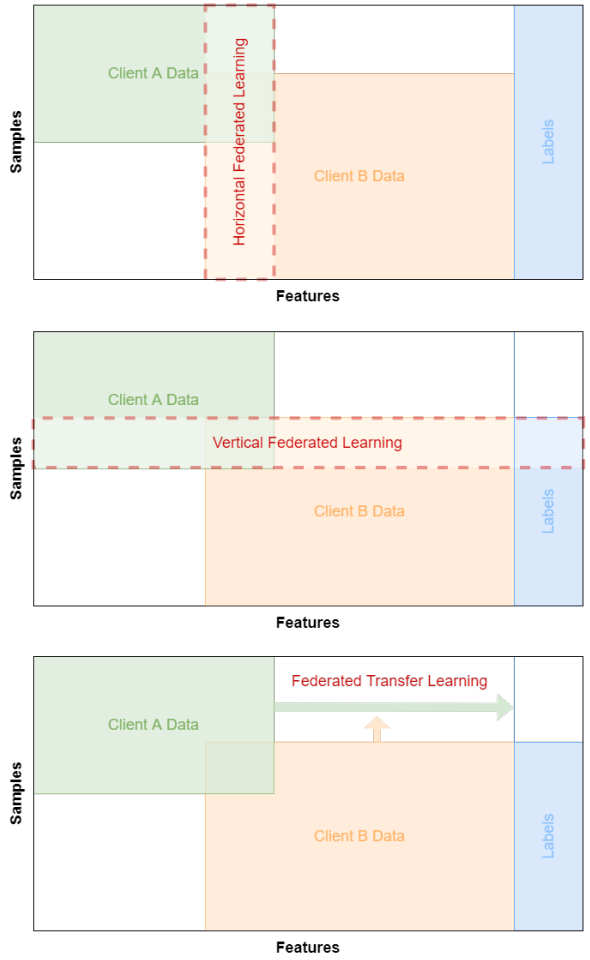
Federated Learning: General distributed scenario with several clients, with their respective local data, which want to build an aggregated model.

Federated Learning - Architectures

1. **Horizontal FL:** Local datasets share the same feature space but they are different in samples

2. **Vertical FL:** The sample space of the local data sets is the same (or similar) but the feature space is different. The clients are trained with the same data, using different features

3. **Federated Transfer Learning:** Local datasets differ both in samples and features



- Global **Objective Function:**

$$\min_{w} F(w), \quad where \;\; F(w) \coloneqq \sum_{k=1}^{K} p_k F_k(w) \qquad p_k : weight\ coefficient$$

Federated Average (FedAVG)

1. Server sends **initialized** model's parameters to the clients
2. Clients perform their **local updates**
3. Clients **send back** their updated parameters
4. Server **aggregates** the parameters (simple average)
5. Then, the **parameters are updated** and sent to the clients
6. Goto 2

Algorithm 1 Federated Averaging. K: number of clients; B: batch size; E: number of local epochs; η: learning rate

procedure ServerExecution
  initialize w_0
  for each round $t = 1, 2, \ldots$ **do**
    for each client k **in parallel do**
      $w_{t+1}^k \leftarrow \text{ClientUpdate}(k, w_t)$
    end for
    $w_{t+1} \leftarrow \sum_{k=1}^{K} \frac{n_k}{n} w_{t+1}^k$
  end for
end procedure

procedure ClientUpdate(k, w)
  $\mathcal{B} \leftarrow$ (split D_k into batches of size B)
  for each local epoch i from 1 to E **do**
    for batch $b \in \mathcal{B}$ **do**
      $w \leftarrow w - \eta \nabla \ell(w; b)$
    end for
  end for
  return w to server
end procedure

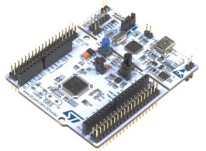
The Last Mile

Towards an unified vision …

MACHINE AND DEEP LEARNING AS A SERVICE

MACHINE AND DEEP LEARNING PLAFORMS FOR IOT AND EDGE

Data Center



Internet-of-Things

PC Embedded

Edge Computing

nytimes.com

Bio | Manuel Roveri,... | Corriere della Sera | Google News | Top Sites | Spectrum: Technolog... | Xplore Full-Text PDF: | Just How Accurate A...

SECTIONS HOME SEARCH

The New York Times

SUBSCRIBE NOW LOG IN

PERSONAL TECH

Just How Accurate Are Fitbits? The Jury Is Out

By MIKE McPHATE MAY 25, 2016

Fitbit Charge HR Tony Cenicola/The New York Times

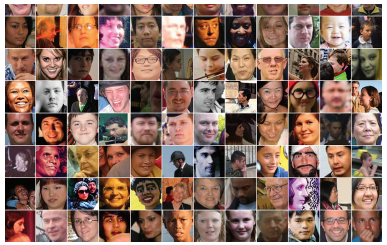
Many users of activity trackers have always harbored suspicions: How accurate are these things?

A handful of tests by journalists and researchers have tried to bring clarity to the issue. Results, alas, have been mixed.

The latest study, released by the plaintiffs in a class-action lawsuit against Fitbit, found that the pulse-monitoring technology used in the company's wrist-bound Surge and Charge devices was "highly inaccurate during elevated physical activity."

Researchers from California State Polytechnic University, Pomona, had 43 subjects wear the devices as they ran, jogged and jumped rope, among other activities, and then compared the readings with those of an electrocardiogram.

New York Times
May 26, 2016



FINDING ONE FACE IN A MILLION

A new benchmark test shows that even Google's facial recognition algorithm is far from perfect

Helen of Troy may have had the face that launched a thousand ships, but even the best facial recognition algorithms might have had trouble finding her in a crowd of a million strangers. The first public benchmark test based on 1 million faces has shown how facial recognition algorithms from Google and other research groups around the world still fall well short of perfection.

Facial recognition algorithms that had previously performed with more than 95 percent accuracy on a popular benchmark test involving 13,000 faces saw significant drops in accuracy when taking on the new MegaFace Challenge. The best performer, Google's FaceNet algorithm, dropped from near-perfect accuracy on the five-figure data set to 75 percent on the million-face test. Other top algorithms dropped from above 90 percent to below 60 percent. Some algorithms made the proper identification as seldom as 35 percent of the time.

"MegaFace's key idea is that algorithms should be evaluated at large scale," says Ira Kemelmacher-Shlizerman, an assistant professor of computer science at the University of Washington, in Seattle, and the project's principal investigator. "And we make a number of discoveries that are only possible when evaluating at scale."

ng a million faces matter because fa
cognition algorithms inevitably f
ch challenges in the real world. Peo
creasingly trust these algorithms to
ctly identify them in security verif
n scenarios, and law enforcement m
so rely on facial recognition to pick

The most popular benchmark until now has been the Labeled Faces in the Wild (LFW) test created in 2007. LFW includes 13,000 images of just 5,000 people. Many facial recognition algorithms have been fine-tuned to the point that they scored near-perfect accuracy when picking through the LFW images. Most researchers say that new benchmark challenges have been long overdue.

"The big disadvantage is that [the field] is saturated—that is, there are many, many algorithms that perform above 95 percent on LFW," Kemelmacher-Shlizerman says. "This gives the impression that face recognition is solved and working perfectly."

With that in mind, University of Washington researchers raised the bar by creating the MegaFace Challenge using 1 million Flickr images of 690,000 unique faces that are publicly available under a Creative Commons license.

The MegaFace Challenge forces facial recognition algorithms to do verification and identification, two separate but related tasks. Verification involves trying to correctly determine whether two faces presented to the facial recognition algorithm belong to the same person. Identification involves trying to find a matching photo of the same person among a million "distractor" faces. Initial results on algorithms developed by Google and four other research groups were presented at the IEEE Conference on Computer Vision and Pattern Recognition on 30 June. (One of MegaFace's developers also heads a computer vision team at Google's Seattle office.)

The results presented were a mix of the intriguing and the expected. Nobody was surprised that the algorithms' performances suffered as the number of distractor faces increased. And the fact that algorithms had trouble identifying the same person at different ages was a known problem. However, the results also showed that algorithms trained on relatively small data sets can compete with those trained on very large ones, such as Google's FaceNet, which was trained on more than 500 million photos of 10 million people.

For example, the FaceN algorithm from Russia's N-TechLab performed well on certain tasks in comparison with FaceNet, despite having trained on 18 million photos of 200,000 people. The SIAT MMLab algorithm, created by a Chinese team under the leadership of Yu Qiao, a professor with Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, also performed well on certain tasks.

Nevertheless, FaceNet has so far performed the best overall. It delivered the most consistent performance across all testing.

UNIVERSITY OF WASHINGTON

The huge drops in accuracy when scanning a million faces matter because facial recognition algorithms inevitably face such challenges in the real world. People increasingly trust these algorithms to correctly identify them in security verification scenarios, and law enforcement may also rely on facial recognition to pick suspects out of the hundreds of thousands of faces captured on surveillance cameras. The most popular benchmark until

IEEE Spectrum
Aug, 2016

Tech Alert

IEEE SPECTRUM

JOIN IEEE

15 February 2018

A Sleeping Alexa Can Listen for More Than Just Her Name

How do you keep digital assistants like Amazon's Alexa from jumping into every conversation or reacting to every bit of noise when they should be sleeping? Researchers are refining the answer to that question even as they attempt to broaden the palette of sounds to which Alexa will react. So, they've got a big challenge ahead of them. Among the solutions being worked out is analyzing sounds and coding them into ideophones. At that point, it will be just as simple for a digital assistant to properly react to a sound — breaking glass, barking dogs, or alarms sounding — as it is to a wake word.

Thank you for the attention!