Enabling energy efficient machine learning on a Ultra-Low-Power vision sensor for IoT

Francesco Paissan, Massimo Gottardi, Elisabetta Farella

FONDAZIONE BRUNO KESSLER

E3DA
ENERGY EFFICIENT EMBEDDED DIGITAL ARCHITECTURES

IRIS
INTEGRATED RA[illegible]N
AND IMAGE SENSO[illegible]

Contributions

- To address the problem of **redundancy** in video applications, we exploited the use of a Smart Vision Sensor (SVS);
- We designed a vision pipeline for detection, classification and tracking targeting **low power** consumption and computational complexity;
- We implemented and tested the pipeline on an ARM Cortex-M4 based board;

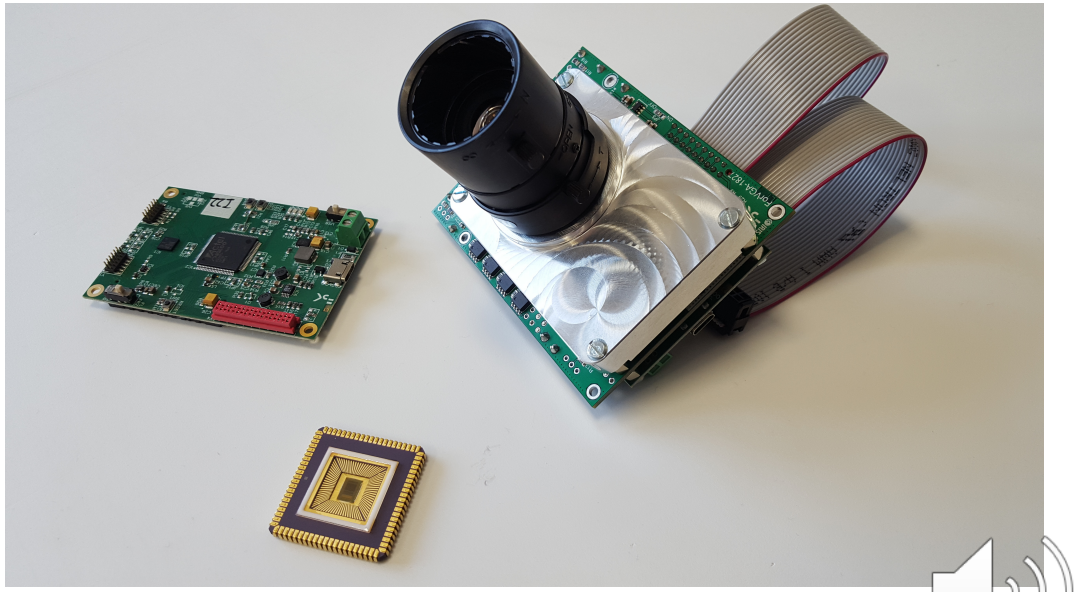
Related work

- Many applications of **vision on the edge** exist:
  - Robotics: target the Watt-scale power consumption, not sustainable in an IoT node powered by a solar harvester. For example, in [3] the authors present a variation of the YOLOv2 algorithm targeting the NVIDIA TX1 board; the same board is used also in [4];
  - IoT domain: some implementations using custom hardware for maximum power-efficiency. For example, in [1] the authors used a 4-core architecture to boost the node performance, whereas in [2] the authors use an 8-core architecture;

[1]"Always-ON visual node with a hardware-software event-based binarized neural network inference engine" - M. Rusci, et. al. - 2018 International Conference on Computing Frontiers;
[2] "A 64mW DNN-based Visual Navigation Engine for Autonomous Nano-Drones", Daniele Palossi, et. al. - 2019 IEEE Internet of Things Journal.
[3] "Fast YOLO: A Fast You Only Look Once System for Real-time Embedded Object Detection in Video", Javad Shafiee, et. al.
[4] "Object detection and recognition of intelligent service robot based on deep learning", Y. Zhang, et. al. - 2017 IEEE Conference on Cybernetics and Intelligent Systems.

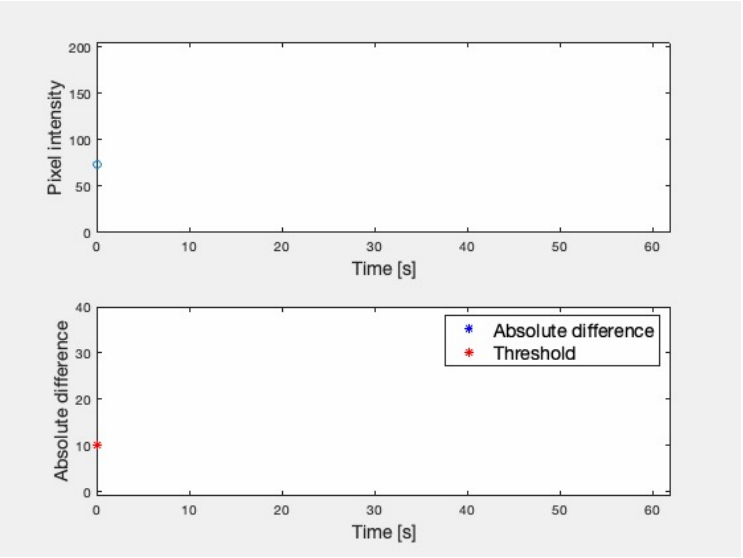
Smart Vision Sensors

- Use motion detection algorithms to reduce image size by showing only moving objects
- Enable event driven computation
- Power consumption:
  - Idle: 344 µW
  - Active: 1350 µW
- Frame rate: 8 fps
- Resolution: 160x120



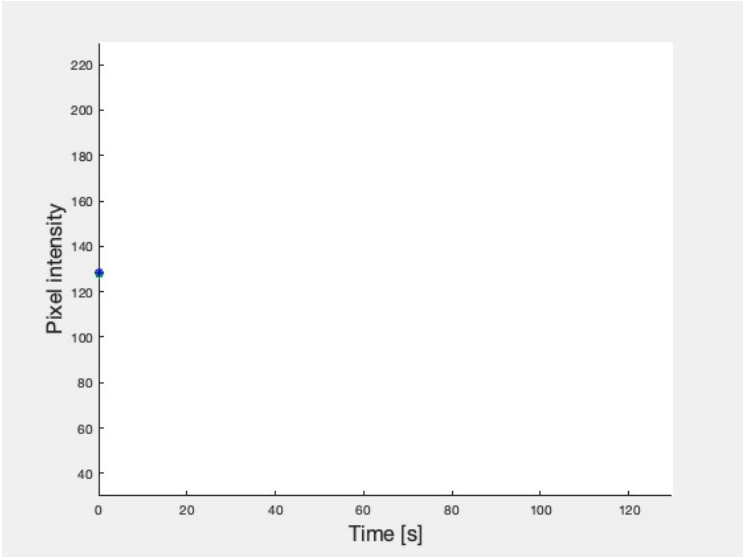
Motion detection

- Insensitive to periodic changes
- Not robust to periodic changes



State of the art

- Efficient in temporal thresholding
- Robust to periodic changes



Our approach

Simulation of output

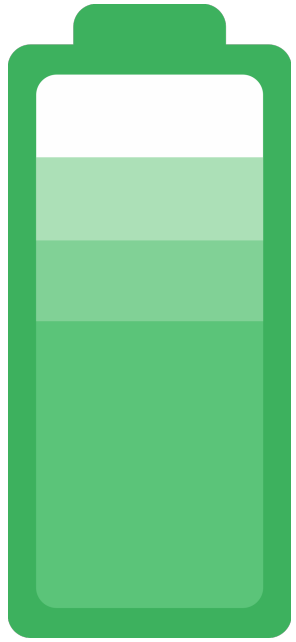
Frame difference  Double threshold

How can we use data from this sensor?

Target and goal of the work

Smart cities scenario

P/C Tracking



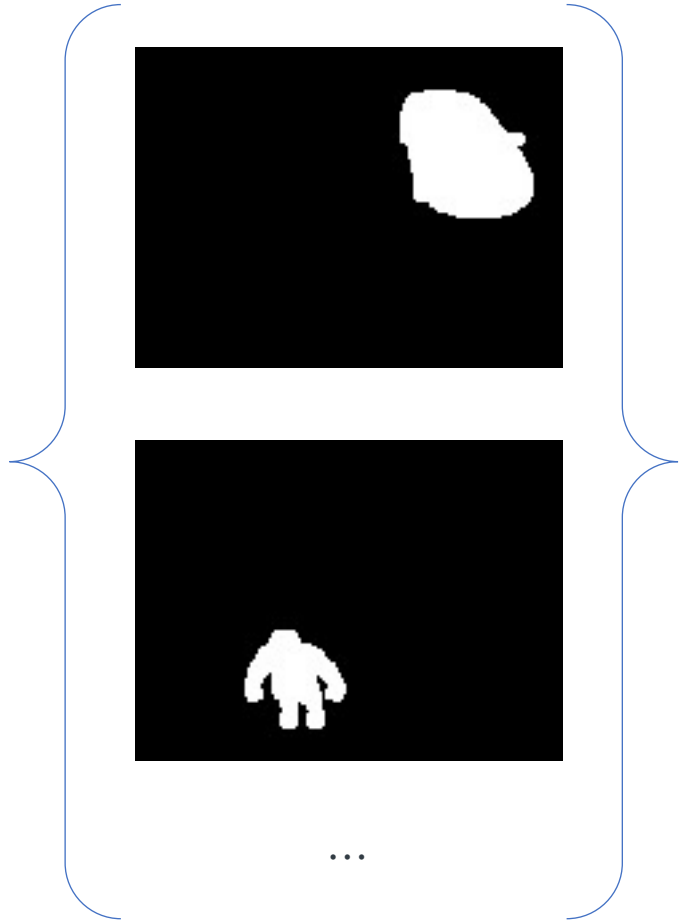
Low power

Setup and assumptions

- The entire work is tailored to work as **IoT vision node** in a smart city scenario. The system is therefore tuned to work as solution with sensors which output is compressed using background subtraction techniques;
- Therefore, the pipeline proposed can not compared with SotA or other RGB benchmarks that don't satisfy this constraint;

Dataset creation

$$P = \begin{bmatrix} \frac{\cot\frac{fovy}{2}}{aspect} & 0 & 0 & 0 \\ 0 & \cot\frac{fovy}{2} & 0 & 0 \\ 0 & 0 & \frac{n+f}{n-f} & \frac{2*n*f}{n-f} \\ 0 & 0 & -1 & 0 \end{bmatrix}$$

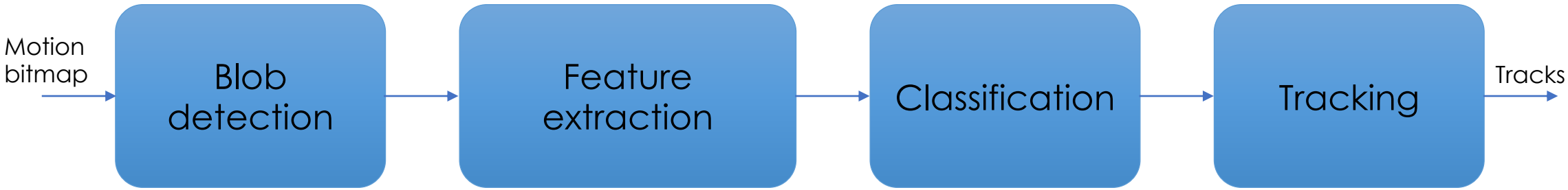


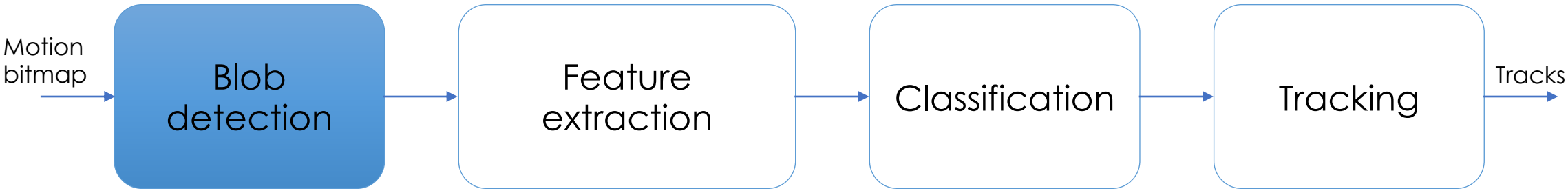
3D models

Perspective projection

Artificial dataset

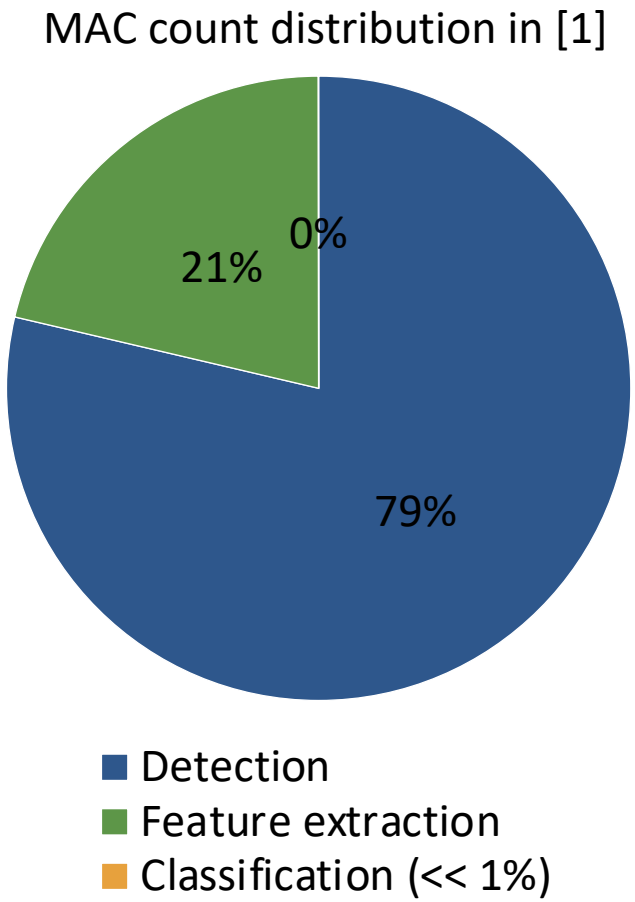
Algorithm outline




Motion bitmap
Blob detection
Feature extraction
Classification
Tracking
Tracks

Detection

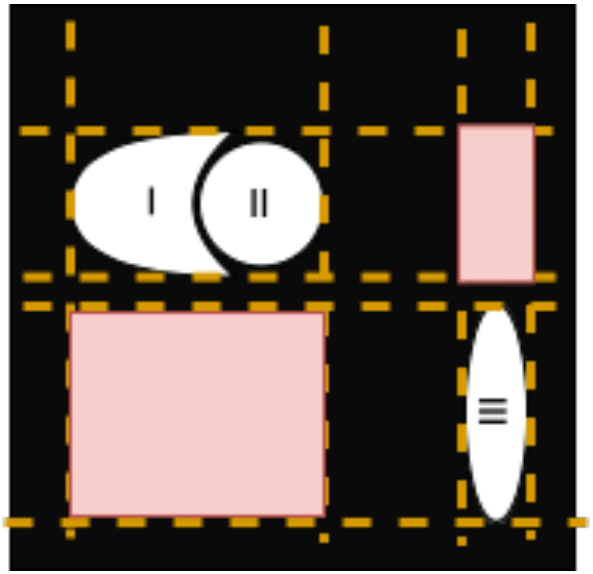
- We innovated on the approach presented in [5] which was accounting for **nearly 80%** of the total MAC count;
- Since the sensor outputs the (x,y)-projections, we exploit the Cartesian product as region proposal and filter the regions considering a minimum threshold on the number of white pixels present in the frame.



[5] "People/Car Classification using an Ultra-Low-Power Smart Vision Sensor", Paissan et al. - 2019 IEEE International Workshop on Advances in Sensor and Interfaces.

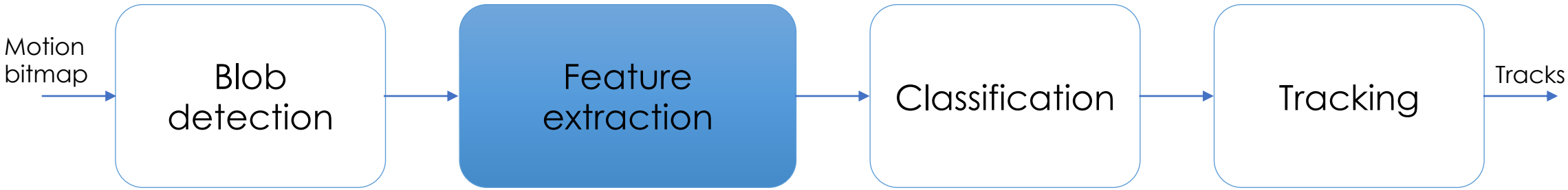
Detection - drawbacks

- There are some drawbacks on the global level (as shown in the figure on the right), which rarely occur in the test sequences acquired due to the sensor's limitation in resolution;
- With respect to the previously used approach [6], we detect bounding boxes which size is **1.5 times bigger**;



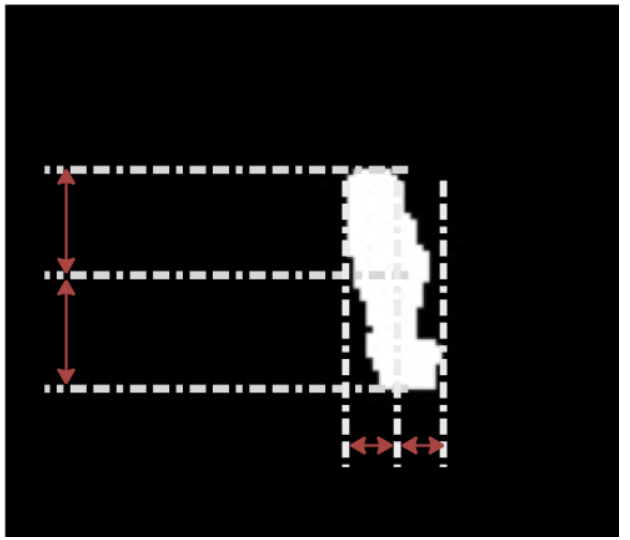
An example in which the proposed algorithm fails. In fact, blob II is not detected as its projections are merged with the ones from blob I, therefore a single blob is considered in the particular bounding box.

[6] "Topological structural analysis of digitized binary images by border following", Suzuki et al. - 1985 Journal of Computer Vision, Graphics and Image Processing.


Motion bitmap
Blob detection
Feature extraction
Classification
Tracking
Tracks

Feature extraction

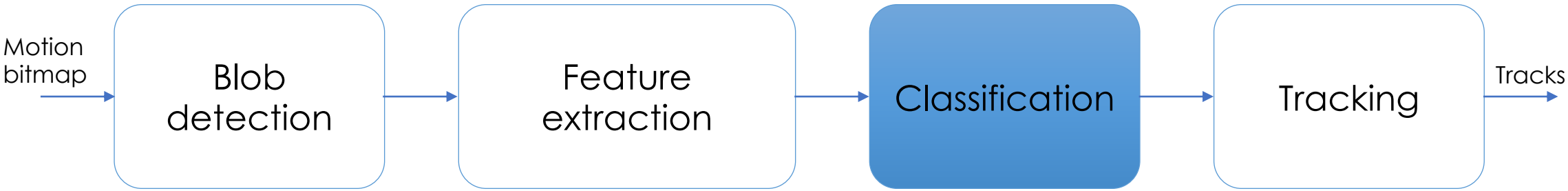
- Chosen after an analysis on objects' geometry:
  - Area of the blob;
  - Variances on the two axis;
  - Minumum vertical coordinate.



Feature	Weigth
σ_y^2	0.38 $\pm$0.05
Area	0.16 $\pm$0.04
Blob position	0.11 $\pm$0.03
σ_x^2	0.06 $\pm$0.03

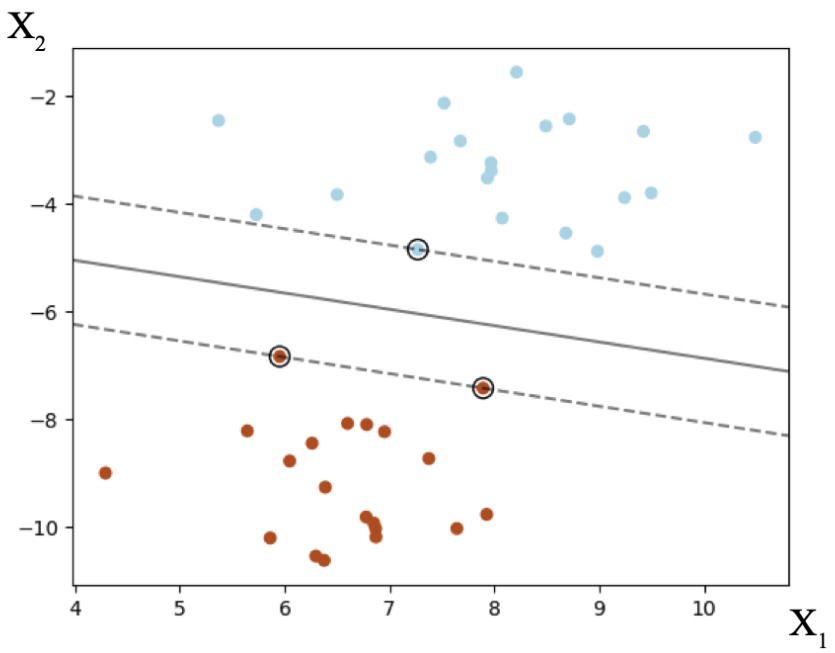
Permutation importance of classification features

*Image moments: statistical property of an image, used to compute area and variances

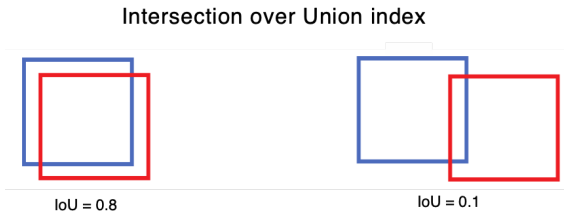

Motion bitmap
Blob detection
Feature extraction
Classification
Tracking
Tracks

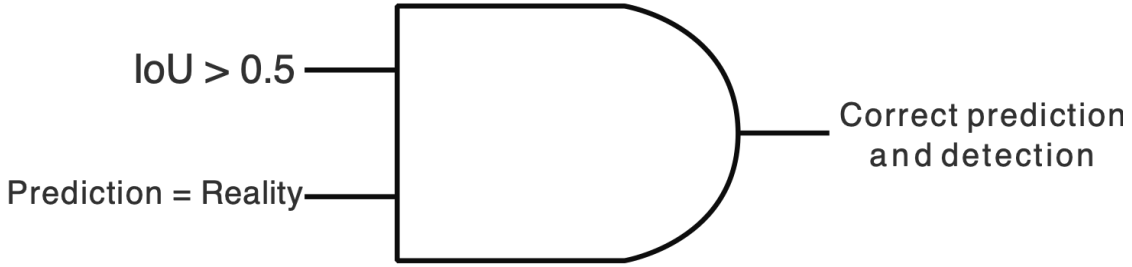
Classification

- Support Vector Machine, a pattern analysis algorithm
- Performs well in experiments for this task
- Suitable computational complexity for our resource-constraint platform
- Trained on the artificial dataset for the classification of real data



Performance analysis



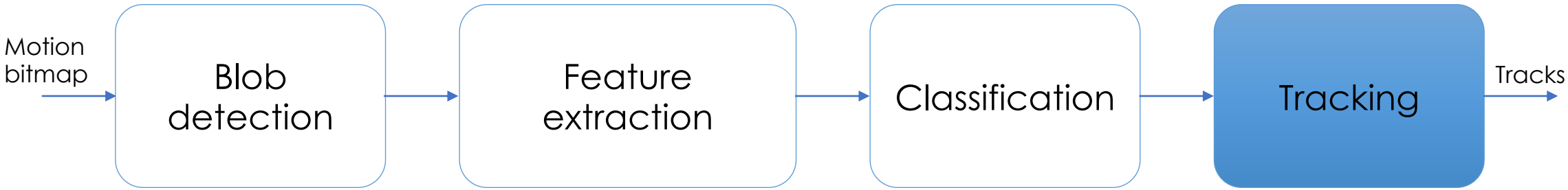


		Actual values	
		Positive	Negative
Predictions	Positive	188	10
	Negative	75	362

Accuracy: ~90%

	Actual value			
	Correct	Incorrect	Total	Error Rate
FORENSOR	239	33	272	0.12
FD	151	121	272	0.44

ER drop (false triggers): 32%


Motion bitmap
Blob detection
Feature extraction
Classification
Tracking
Tracks

Tracking

- The algorithm implemented is based on the work in [7] and it's composed in 4 stages:
  - Detection;
  - Estimation: performed by a Kalman filter implemented using Single Instruction Multiple Data (SIMD);
  - Association: core of the tracking algorithm, solved using the Munkres assignment algorithm;
  - Track management: modify the active object list based on threshold N_{hit} and T_{lost};
  - Time consistency and counter: try to address the 10% error rate in classification by averaging on time.

[7] "Simple online and realtime tracking", A. Bewley et. al - 2016 IEEE International Conference on Image Processing (ICIP)

Tracking performance

- Computed on a sample dataset in [8] and a PETS2009 [9] sequence preprocessed to meet the sensor's criteria;

	Ground Truth		Tracking	
	Car	Pedestrian	Car	Pedestrian
Sequence 1	4	-	4	-
Sequence 2	1	2	1	2
PETS2009 Seq	-	6	-	5

[8] https://github.com/fpaissan/SVS_BS-dataset NB: the authors will keep updating and expanding the dataset at this link;
[9] "Pets2009: Dataset and challenge", J. Ferryman et al - 2009 Twelfth IEEE International Workshop on Performance Evaluation of Tracking and Surveillance.

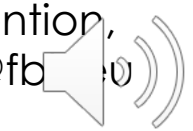
Power consumption

- The processing is implemented using the Nucleo L496ZG, ARM Cortex-M4 based development board and the described sensor;
- The overall power consumption of the node is **7.5 mW** with an 8 ms processing window.

Work in progress

- We are developing an equivalent approach based on YOLOv2 [10] which exploits a custom backbone for processing;
- This future approach will not have the hard constraint of the specific image that can be processed, in fact simply by re-training the architecture on RGB data the same task can be accomplished.

Thank you for your attention,
Francesco. – fpaissan@fb[illegible]0



[10] J. Redmon and A. Farhadi, "Yolo9000: Better, faster, stronger." arXiv preprint arXiv:1612.08242, 2016.