Data Fusion in Tinyml and Applications in Biology and Federated Learning

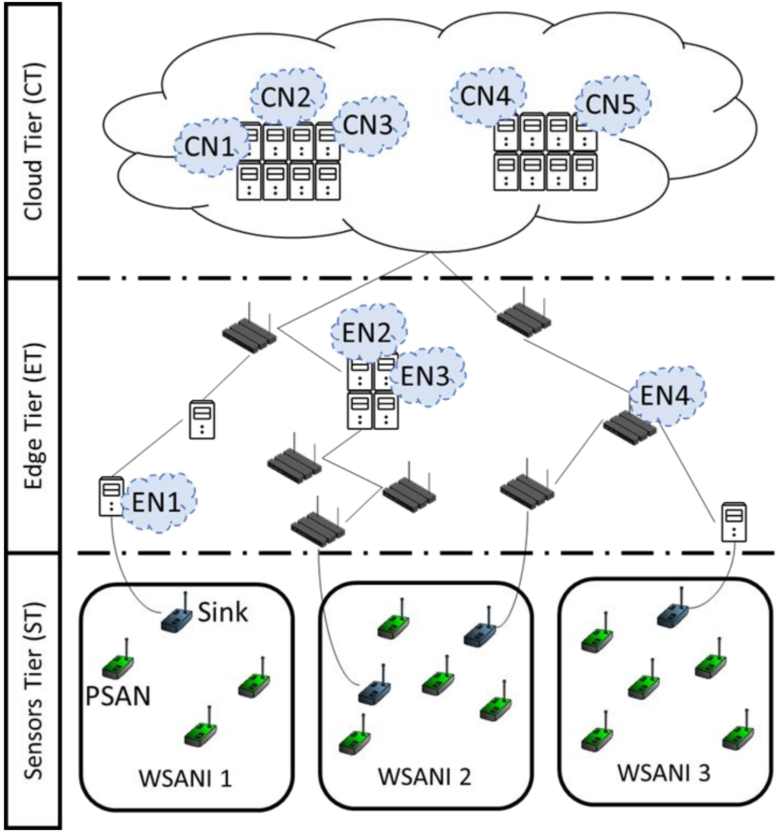
Claudio Miceli de Farias -Universidade Federal do Rio de Janeiro
cmicelifarias@cos.ufrj.br

UFRJ faz 100 ANOS 1920 | 2020
UNIVERSIDADE FEDERAL DO RIO DE JANEIRO

Engenharia de Sistemas e Computação PESC

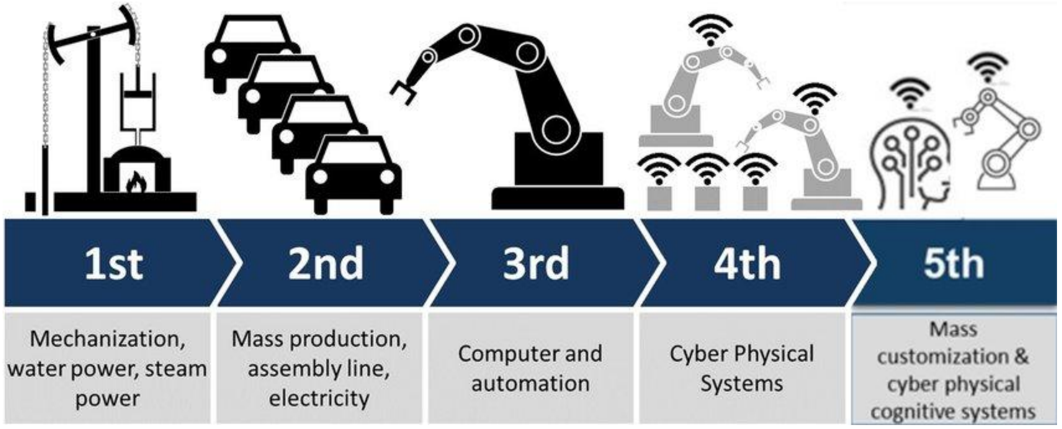
Instituto Tércio Pacitti de Aplicações e Pesquisas Computacionais

Internet Of Things



Industry 5.0

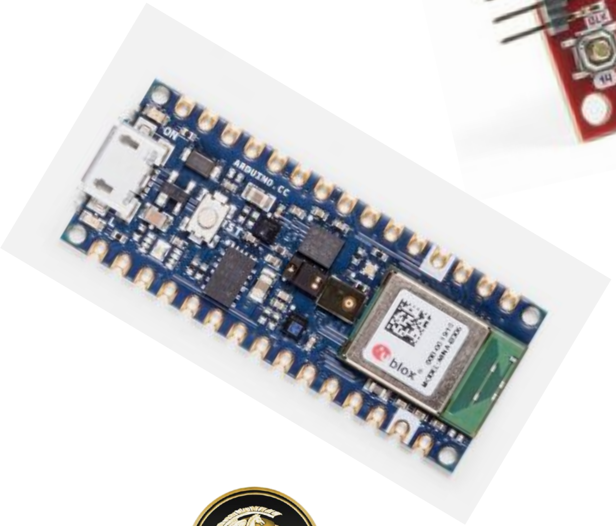

Human-centric
Industry 5.0
Resilient
Sustainable

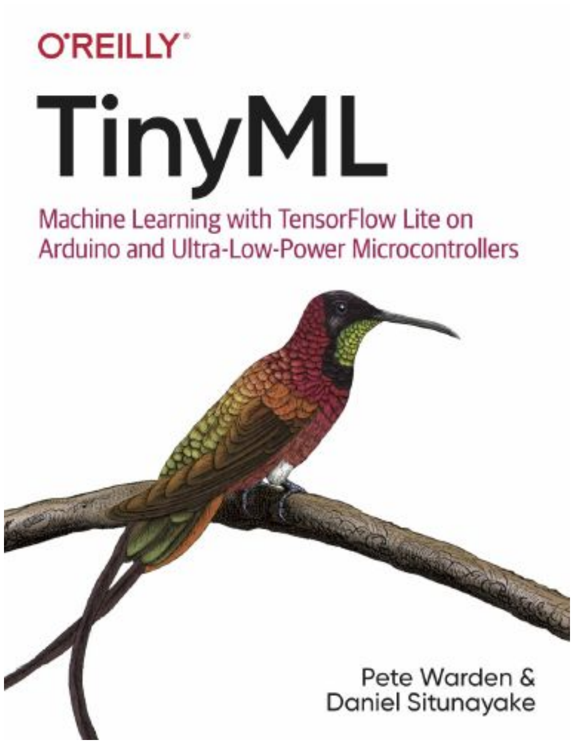

1st
2nd
3rd
4th
5th
Mechanization, water power, steam power
Mass production, assembly line, electricity
Computer and automation
Cyber Physical Systems
Mass customization & cyber physical cognitive systems


VS
5.0
4.0

TinyML - Computer Vision

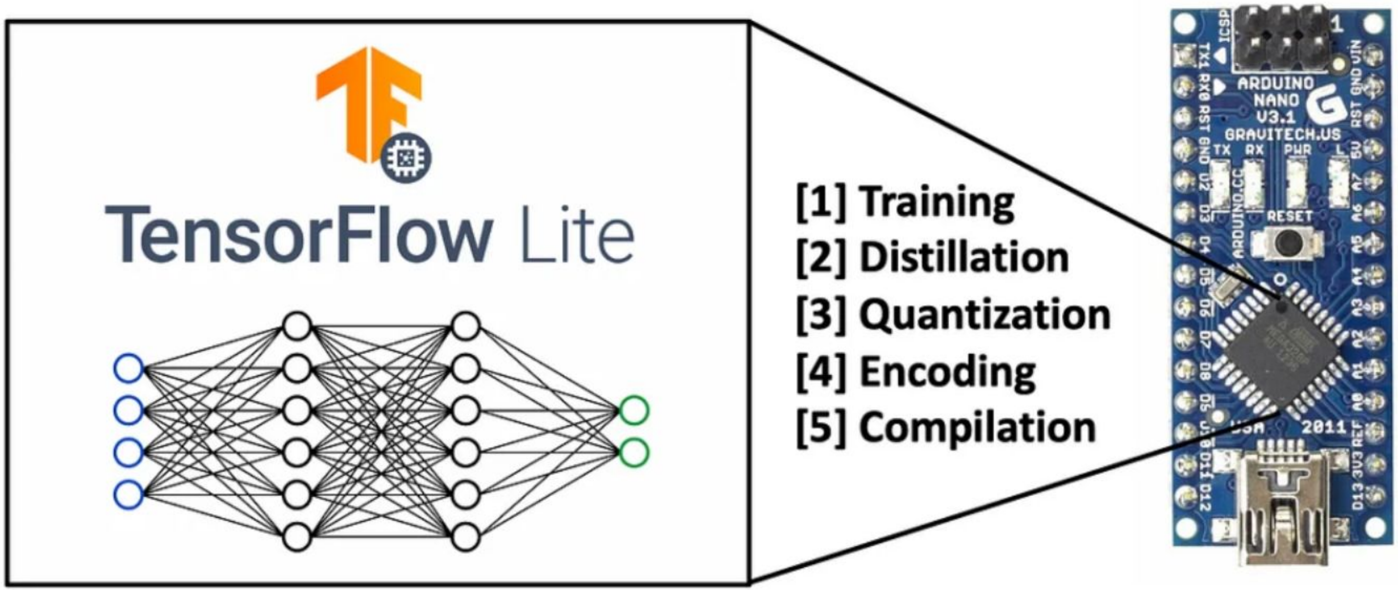
- Tensor flow lite micro





Universidade Federal do Rio de Janeiro

NCE UFRJ Instituto Tércio Pacitti de Aplicações e Pesquisas Computacionais


TensorFlow Lite
[1] Training
[2] Distillation
[3] Quantization
[4] Encoding
[5] Compilation
ARDUINO
NANO
V3.1
GRAVITECH.US
RESET

TinyML

Problem - not a new one

- Resource constrained environment
- Decision making
- CNN is the traditional way

Weightless Neural Networks

- A different type of Neural Network
- Low cost
- Without weights :)
- Wisard architecture - bethoween has been proposed as a related work!

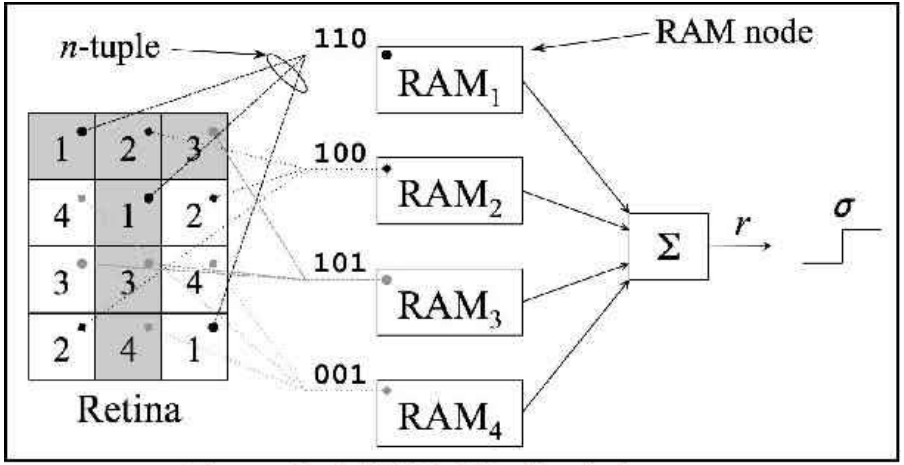
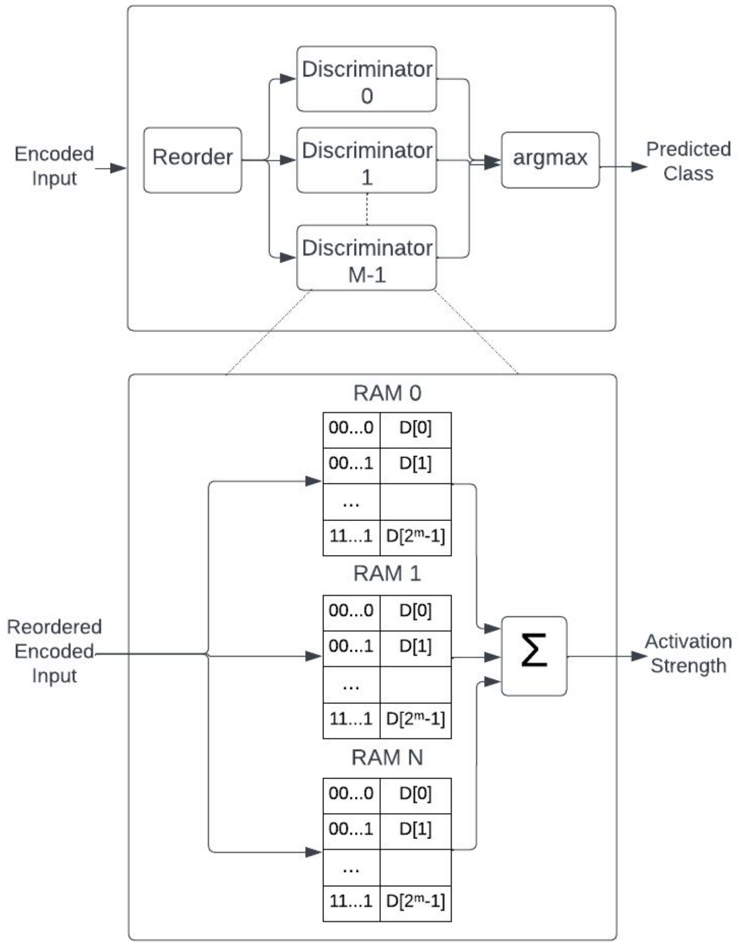


Figure 1. A WiSARD discriminator.

Proposal

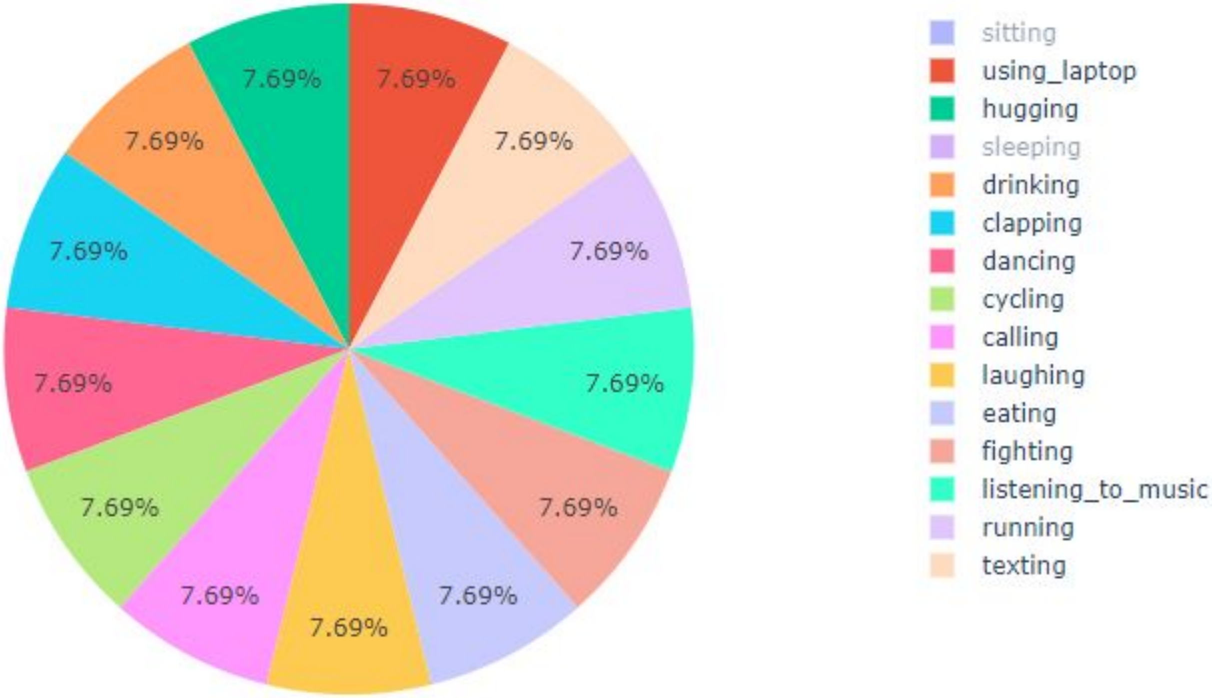
Apply WNN into Computer Vision for Edge AI and compare with traditional techniques!


Encoded Input
Reorder
Discriminator 0
Discriminator 1
Discriminator M-1
argmax
Predicted Class
RAM 0
00...0
D[0]
00...1
D[1]
...
11...1
D[2m-1]
RAM 1
00...0
D[0]
00...1
D[1]
...
11...1
D[2m-1]
RAM N
00...0
D[0]
00...1
D[1]
...
11...1
D[2m-1]
Reordered Encoded Input
Σ
Activation Strength

Simulation

- 2 setups
  - Google colab - baseline
  - Different boards
    - NVIDIA Jetson Nano 2GB
    - Google Coral Dev Board
    - Raspberry Pi 4Gb + Intel Neural Compute Stick 2
    - Raspberry Pi 4Gb + Google Coral TPU
    - Raspberry Pi 4Gb

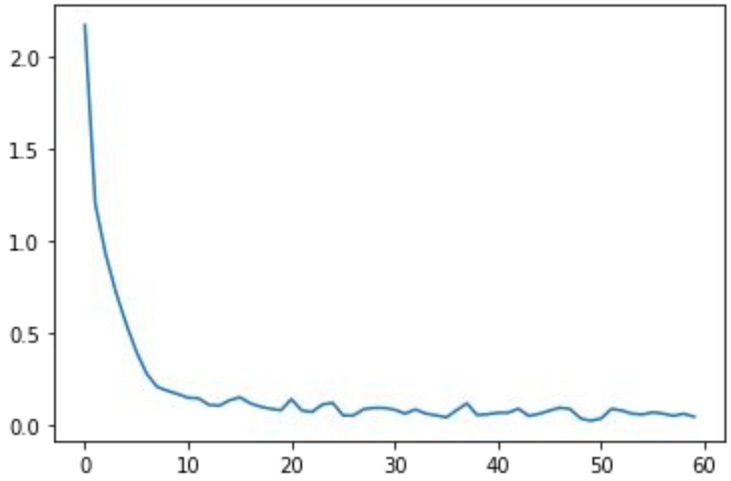
Dataset 1 - Human Activity Recognition

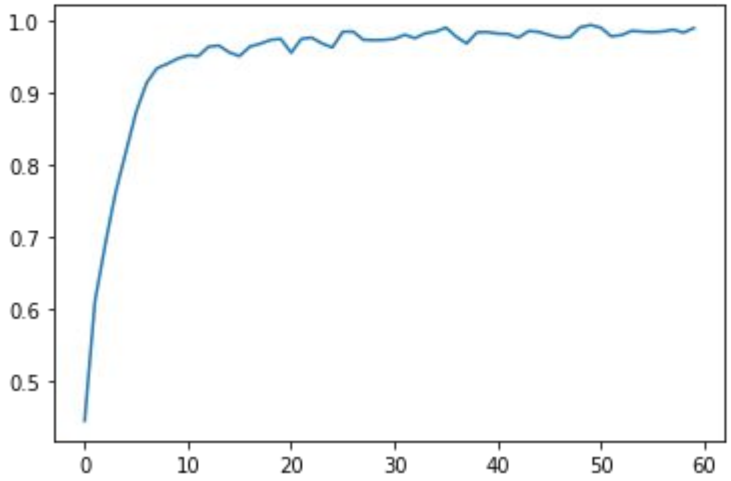


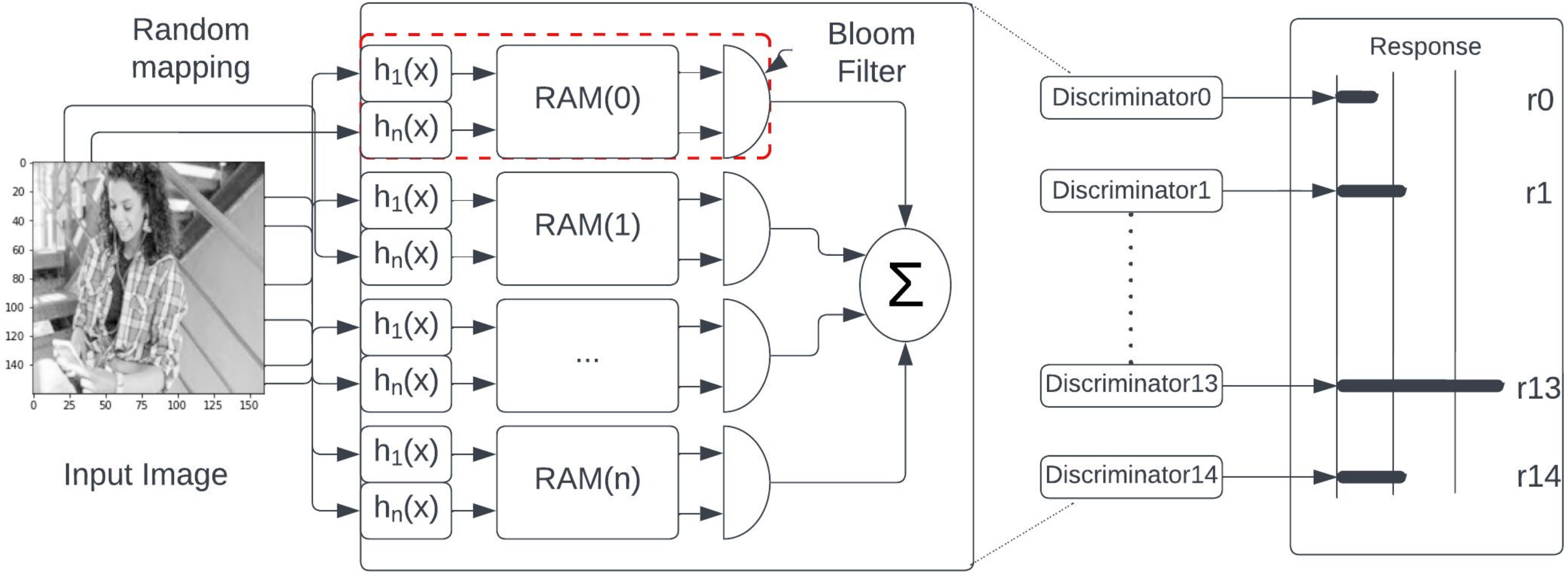
Model Summary

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 vgg16 (Functional)          (None, 512)               14714688

 flatten (Flatten)           (None, 512)               0

 dense (Dense)               (None, 512)               262656

 dense_1 (Dense)             (None, 15)                7695

=================================================================
Total params: 14,985,039
Trainable params: 270,351
Non-trainable params: 14,714,688
_________________________________________________________________
```

Results - Loss and Accuracy







Random
mapping
Input Image
Bloom
Filter
h1(x)
hn(x)
RAM(0)
RAM(1)
...
RAM(n)
Σ
Response
Discriminator0
Discriminator1
Discriminator13
Discriminator14
r0
r1
r13
r14

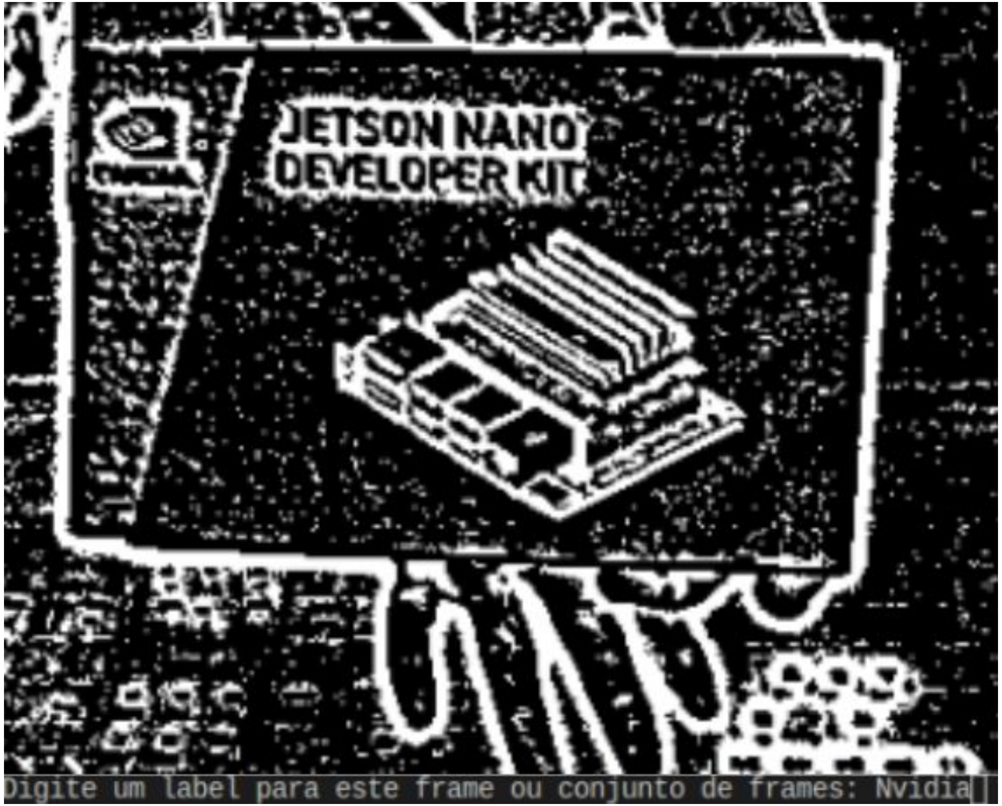

JETSON NANO
DEVELOPER KIT
Digite um label para este frame ou conjunto de frames: Nvidia

TABLE I
CLASSIFICATION REPORT - DATASET 1 - INPUT SIZE: 64 X 64

precision	recall	f1-score	support	Support
calling	0.2143	0.1364	0.1667	154
clapping	0.4113	0.3473	0.3766	167
cycling	0.5107	0.7346	0.6025	162
dancing	0.3438	0.2865	0.3125	192
drinking	0.2542	0.1875	0.2158	160
eating	0.3209	0.6013	0.4185	158
fighting	0.4500	0.3387	0.3865	186
hugging	0.2432	0.2711	0.2564	166
laughing	0.3529	0.3313	0.3418	163
listening_to_music	0.2323	0.2130	0.2222	169
running	0.3839	0.4355	0.4081	186
sitting	0.2153	0.2500	0.2314	180
sleeping	0.5533	0.5000	0.5253	166
texting	0.1923	0.1212	0.1487	165
using_laptop	0.2723	0.3059	0.2881	170
accuracy			0.3369	2544
macro avg	0.3301	0.3373	0.3267	2544
weighted avg	0.3313	0.3369	0.3274	2544

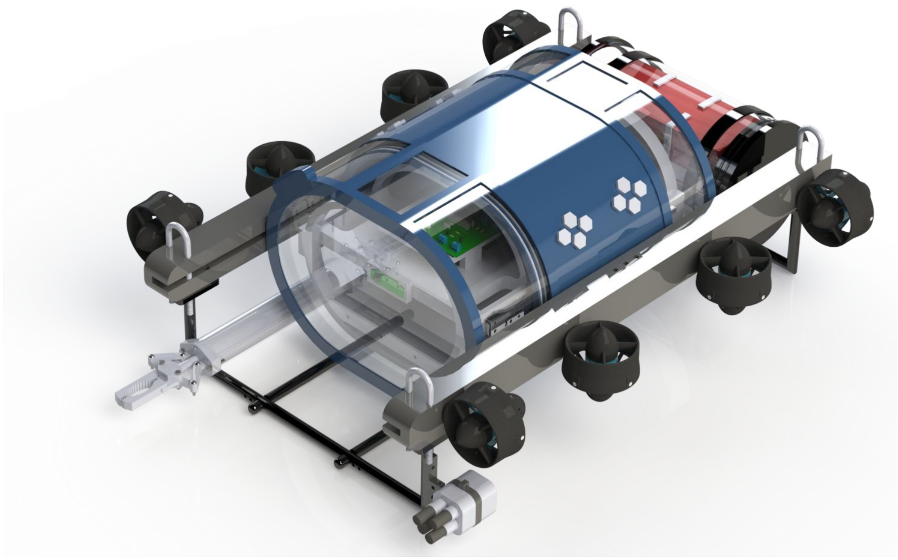
TABLE II
CLASSIFICATION REPORT - DATASET 2

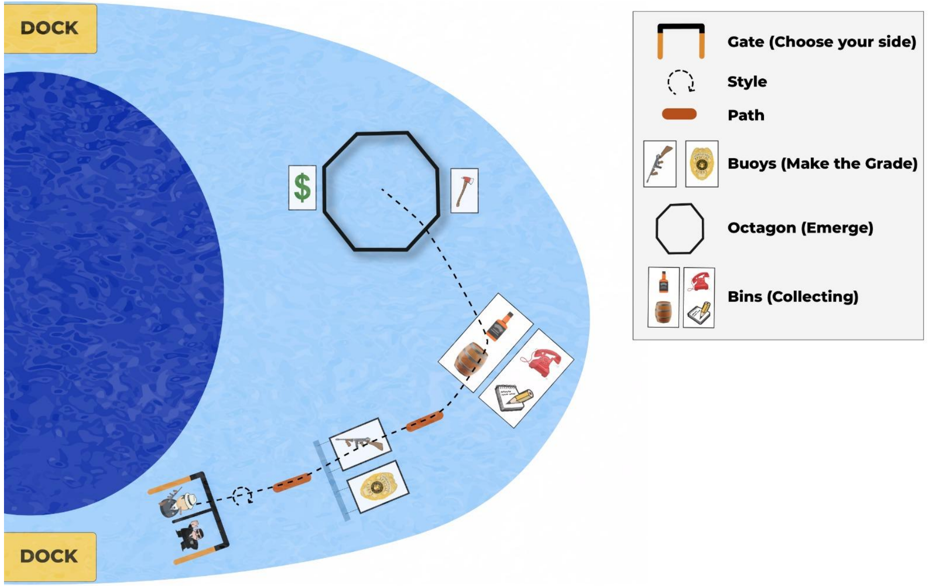
precision	recall	f1-score	support	Support
WALKING	0.89	0.85	0.87	259
UPSTAIRS	0.76	0.89	0.82	216
DOWNSTAIRS	0.92	0.81	0.87	210
SITTING	0.67	0.75	0.71	261
STANDING	0.73	0.65	0.69	270
LAYING	1.00	1.00	1.00	256
accuracy			0.82	1472
macro avg	0.83	0.82	0.82	1472
weighted avg	0.83	0.82	0.82	1472

Conclusions and Future works

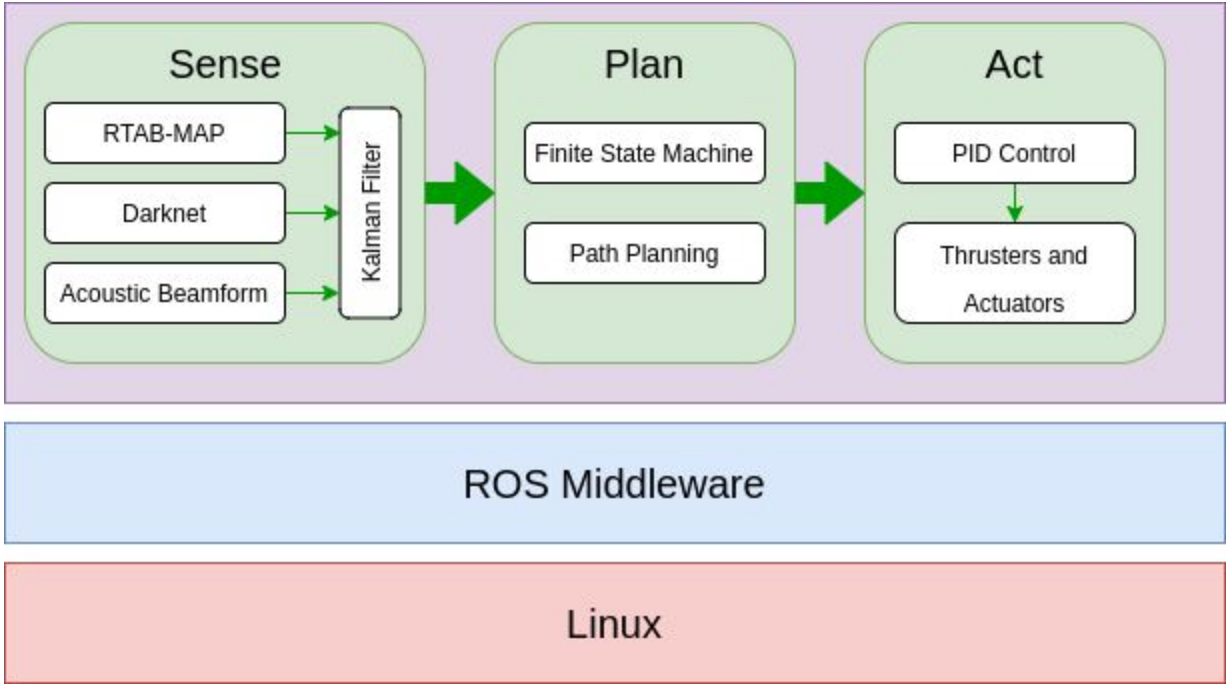
- WNN are effective for Edge AI
- The results had close results to traditional CNN
- As future works:
    - test different types of neural network as classifiers for human action recognition;
    - develop a technique that uses a WNN as input to adapt to network conditions;
    - use a WNN to try to fix distortions in a video in realtime;

LUA and the RoboSub Competition





LUA's Software Design



Proposal

- A fast implementation of the beamforming algorithm in the time and frequency domains
- Errors:
  - (i) the first derives from the noise of the sensors and signals,
  - (ii) arrangement of the sensors, which is a function of the true azimuth and elevation angles.

- Proposal: A TinyML algorithm [4]
- Able to learn about angles combination
- Two approaches:
  - (i) a Convolutional Neural Network and
  - (ii) a clustering algorithm.

Proposal - Neural Networks

```
/* Training */
b = Beamforming()
X = List()
y = List()
for sound in sounds do
    RMS = b.frequency_beamforming(sound)
    X.append(RMS)
    (az_pred, el_pred) = b.angle_of_arrival(sound)
    error = (az_real - az_pred, el_real - el_pred)
    y.append(error)
end for
model = NeuralNetwork()
model.train(X,y)
/* Usage */
sound = new_sound()
RMS = b.frequency_beamforming(sound)
error_pred = model.predict(RMS)
(az, el) = b.angle_of_arrival(sound) - error_pred
```

Proposal - Clustering

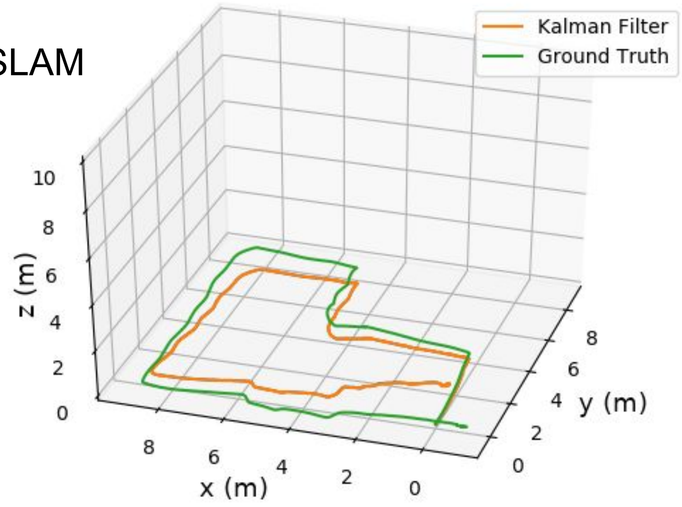
```
/* Training */
b = Beamforming()
angles = [(az, el) for az in range(361) for el in range(181)]
kmeans = KMeans(number_of_clusters).fit(angles)
errors = HashMap(k: List() for k in range(number_of_clusters))
for sound in sounds do
    (az_pred, el_pred) = b.angle_of_arrival(sound)
    angles_cluster = kmeans.predict((az_pred, el_pred))
    error = (az_real - az_pred, el_real - el_pred)
    errors[angles_cluster].append(error)
end for
median_errors = HashMap(k: median(errs) for k, errs in errors)
/* Usage */
sound = new_sound()
(az_pred, el_pred) = b.angle_of_arrival(sound)
angles_cluster = kmeans.predict((az_pred, el_pred))
error_pred = median_errors[angles_cluster]
(az, el) = (az_pred, el_pred) - error_pred
```

Experimental Design

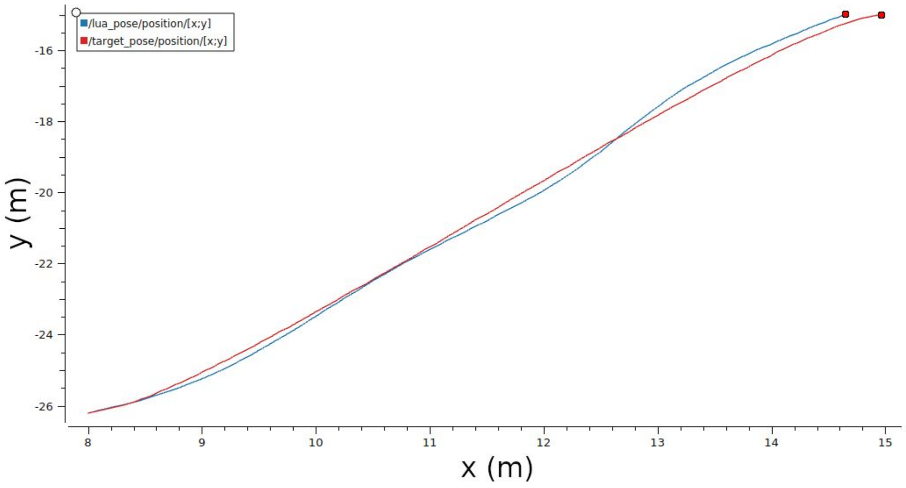
- Nvidia Jetson Nano
- Python
- ROS
- 30 repetitions
- 95% confidence interval
- Synthetic and real data

Experimental Design

SLAM



Path Planning



Experimental Design

Beamforming	Neural Network	Clusterization
9.88	12.18	9.36
0.00%	−23.28%	5.26%

TABLE I
SELD [17] DATASET EVALUATION MEAN ABSOLUTE ERROR - AZIMUTH

Beamforming	Neural Network	Clusterization
16.45	5.57 ± 0.38	13.18 ± 0.13
0.00%	66.14%	19.89%

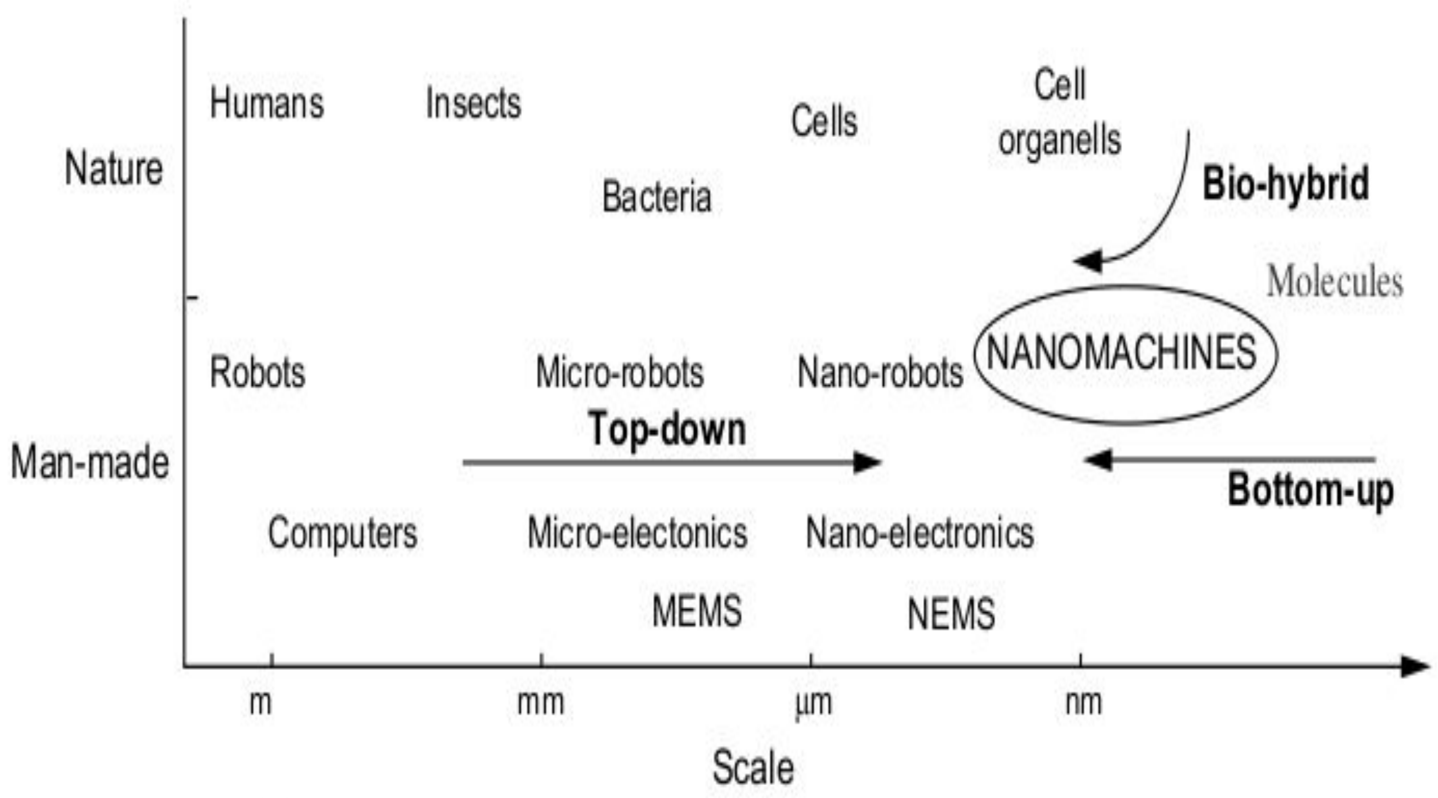
TABLE II
SYNTHETIC DATASET EVALUATION MEAN ABSOLUTE ERROR - AZIMUTH

Conclusions

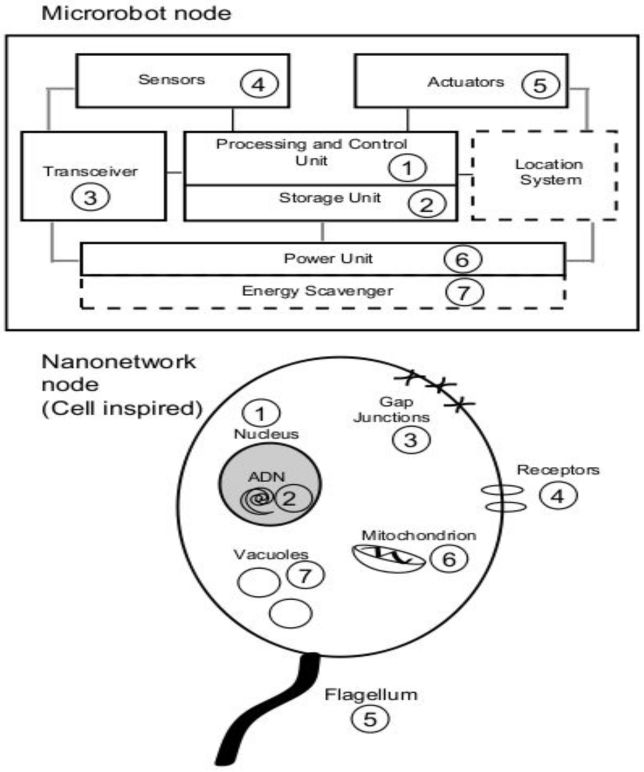
- Lua, a low-cost AUV developed by the UFRJ Nautilus team
- Software components of the AUV
- New beamforming algorithms
- Use time and frequency simultaneously

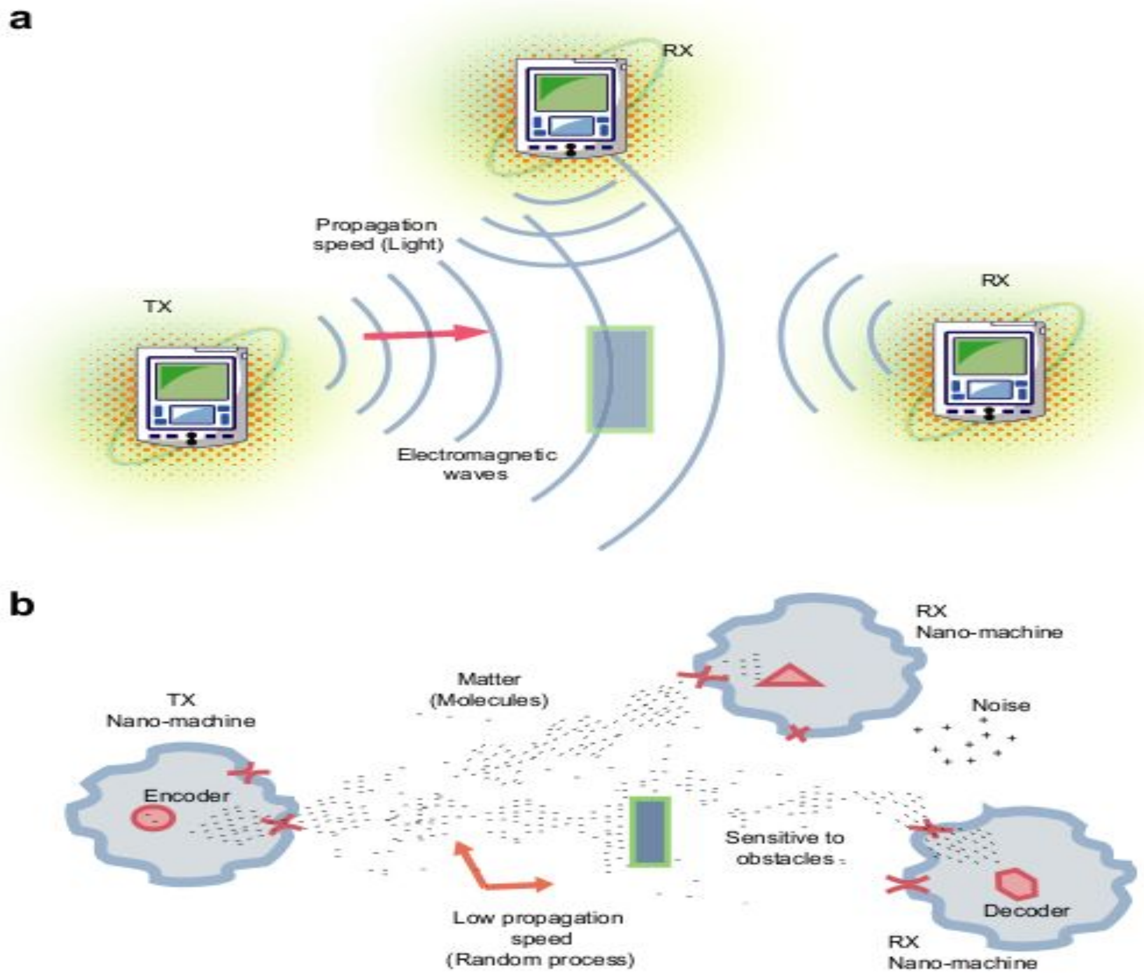
Future works

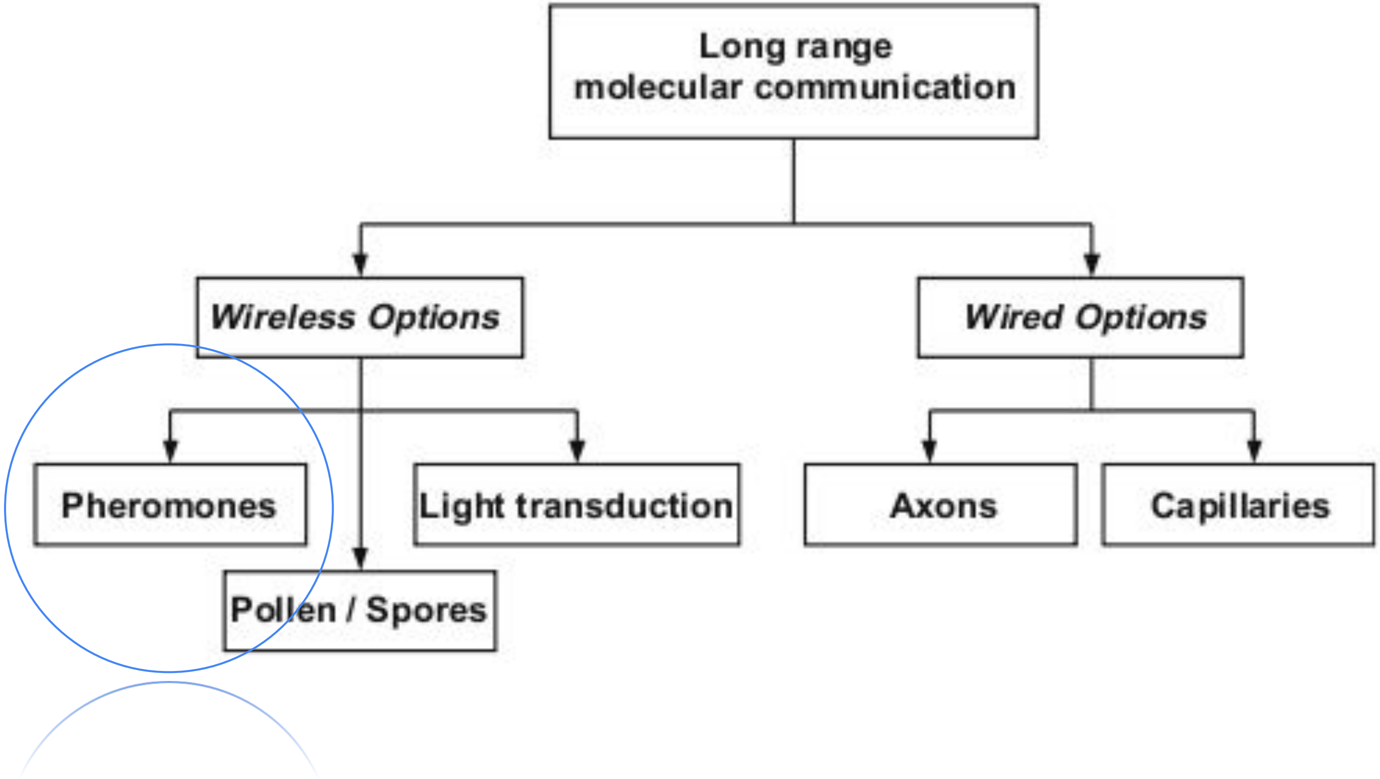
- Increase the amount of training data and/or different Network architectures.
- A better synthetic dataset generator
- Correlation between μ and statistical metrics
- Federated Learning among a group of AUV

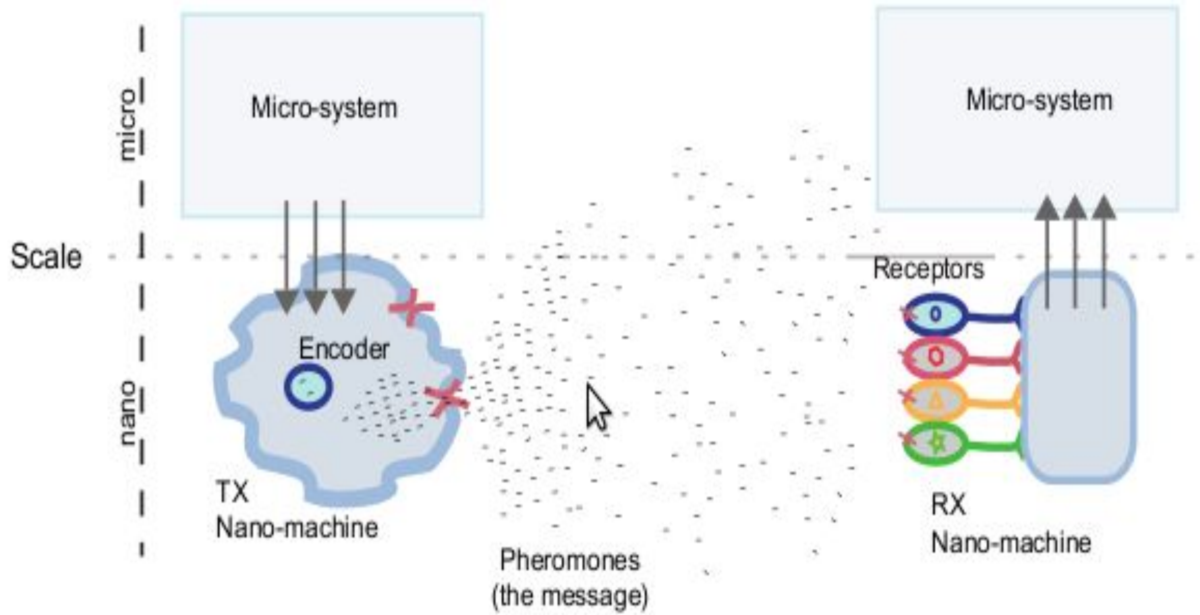

Nature
Man-made
Humans
Insects
Bacteria
Cells
Cell organells
Bio-hybrid
Molecules
NANOMACHINES
Robots
Micro-robots
Nano-robots
Top-down
Bottom-up
Computers
Micro-electonics
Nano-electronics
MEMS
NEMS
m
mm
µm
nm
Scale

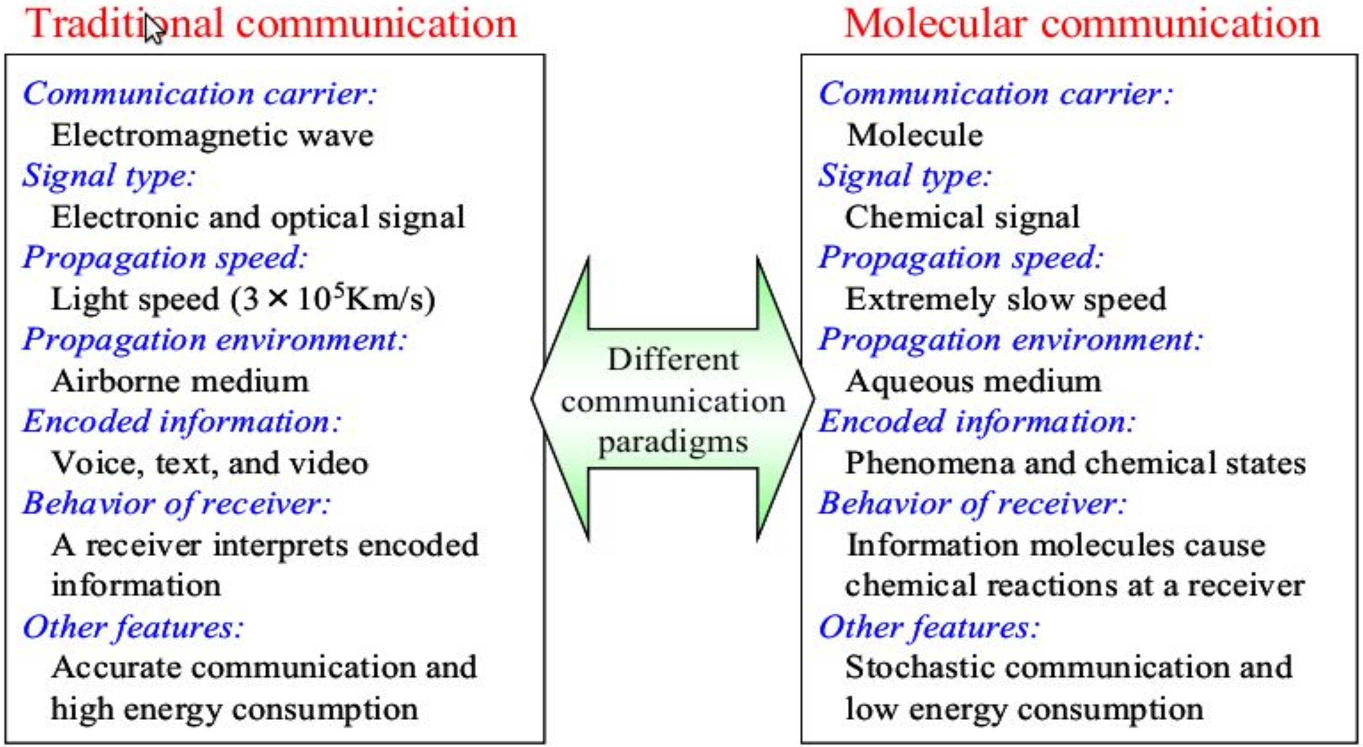
Internet of Bionanomachines




a
RX
Propagation
speed (Light)
TX
RX
Electromagnetic
waves
b
RX
Nano-machine
Matter
(Molecules)
TX
Nano-machine
Noise
Encoder
Sensitive to
obstacles
Decoder
Low propagation
speed
(Random process)
RX
Nano-machine


Long range
molecular communication
Wireless Options
Wired Options
Pheromones
Light transduction
Axons
Capillaries
Pollen / Spores

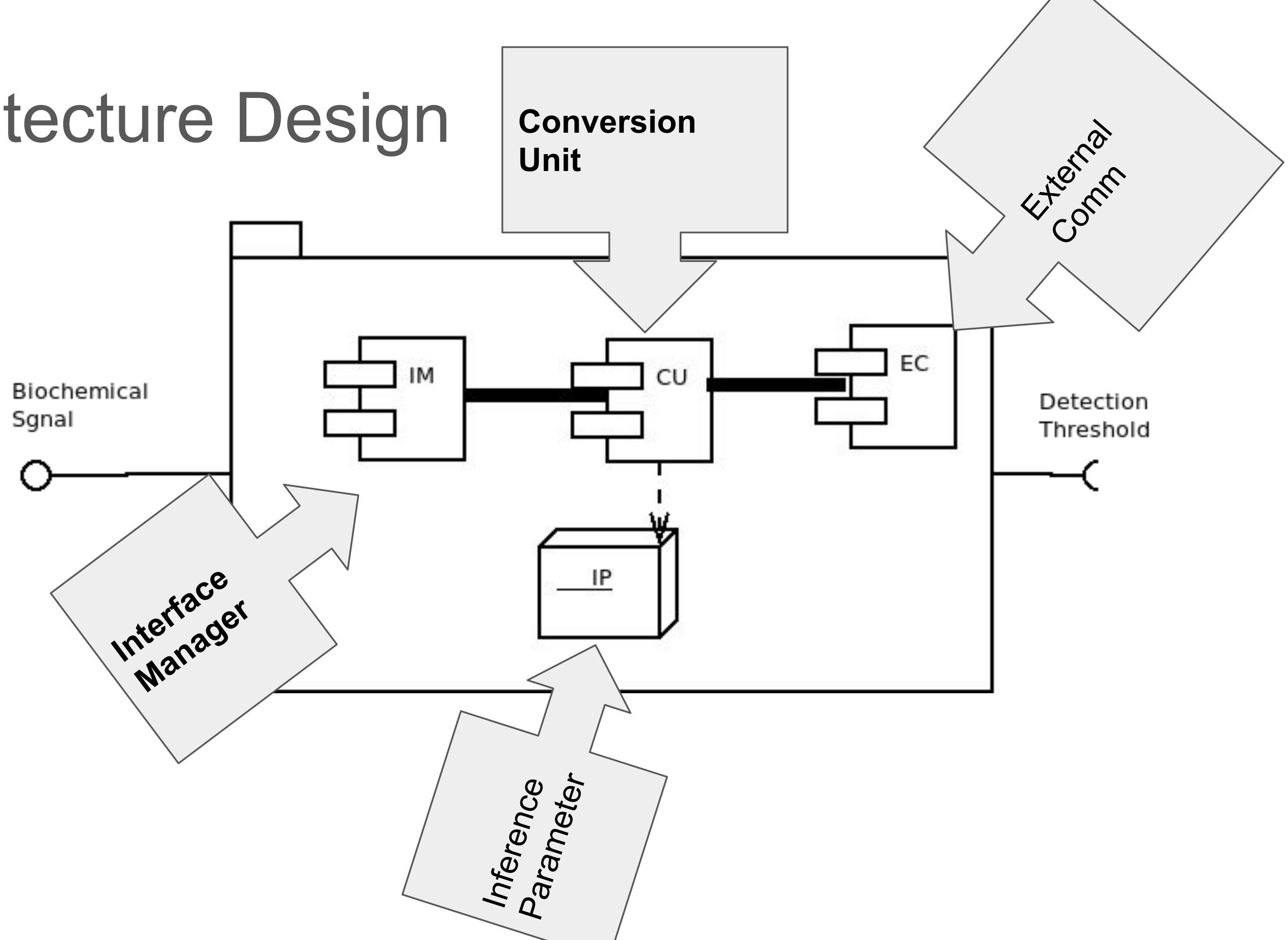

Micro-system
Micro-system
micro
Scale
nano
Encoder
Receptors
TX
Nano-machine
RX
Nano-machine
Pheromones
(the message)

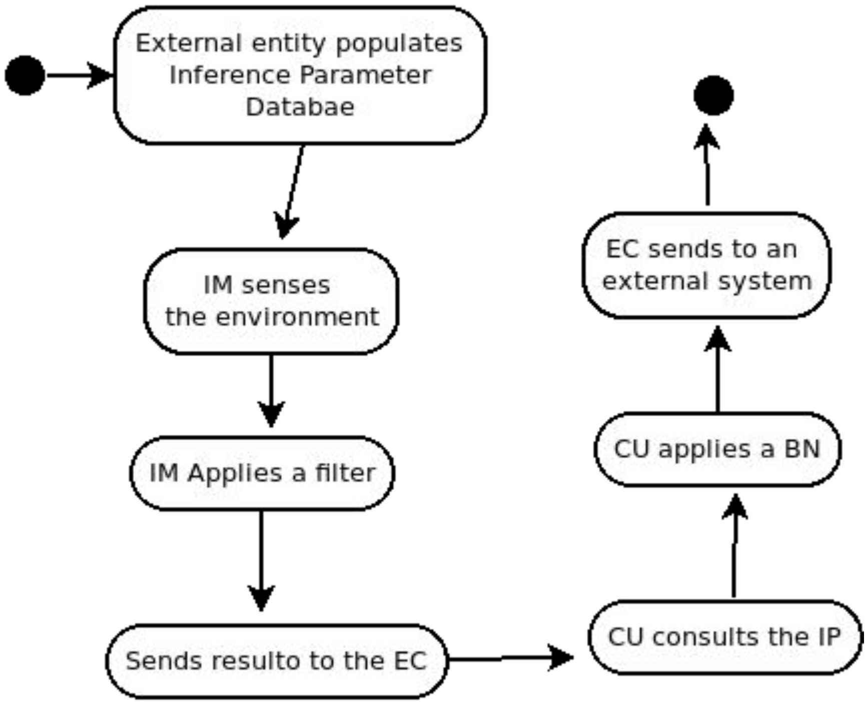


Traditional communication
Communication carrier:
Electromagnetic wave
Signal type:
Electronic and optical signal
Propagation speed:
Light speed (3 × 10^5 Km/s)
Propagation environment:
Airborne medium
Encoded information:
Voice, text, and video
Behavior of receiver:
A receiver interprets encoded information
Other features:
Accurate communication and high energy consumption
Different communication paradigms
Molecular communication
Communication carrier:
Molecule
Signal type:
Chemical signal
Propagation speed:
Extremely slow speed
Propagation environment:
Aqueous medium
Encoded information:
Phenomena and chemical states
Behavior of receiver:
Information molecules cause chemical reactions at a receiver
Other features:
Stochastic communication and low energy consumption

So, we are proposing a
low-cost interface for
long-range IoBNT
communication through indirect
sending using High-Level Data
Fusio

Architecture Design

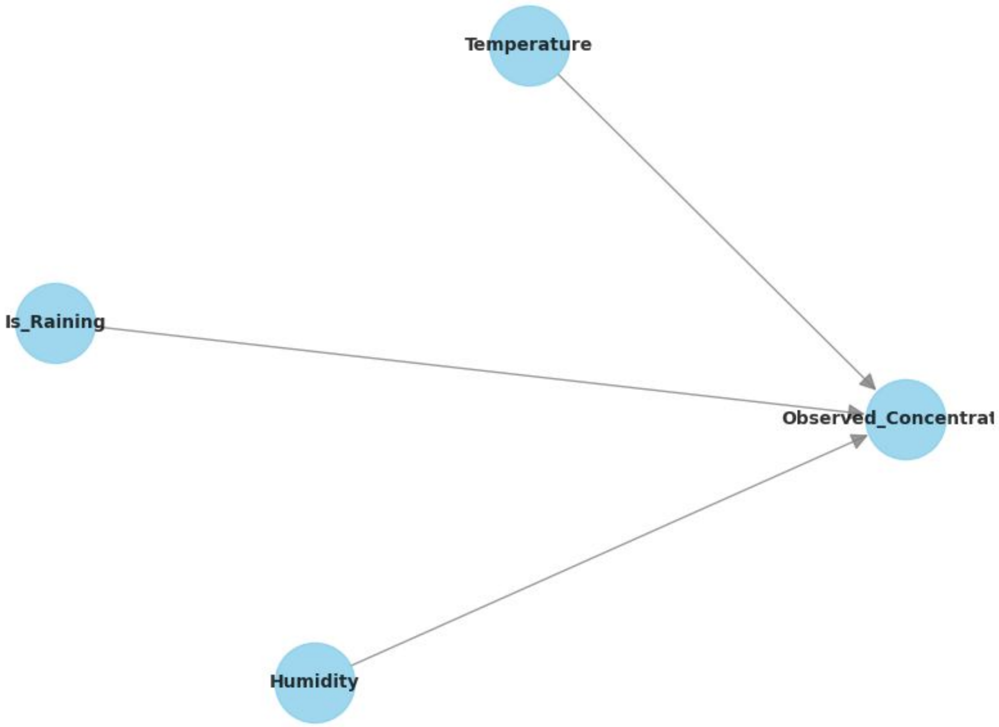



External entity populates
Inference Parameter
Databae
IM senses
the environment
IM Applies a filter
Sends resulto to the EC
CU consults the IP
CU applies a BN
EC sends to an
external system

Bayesian Networks

$$P(X) = \prod_{i}^{n} P(X_i | parents(X_i)). \qquad (1)$$

Bayesian Network for Ethylene Concentration Prediction

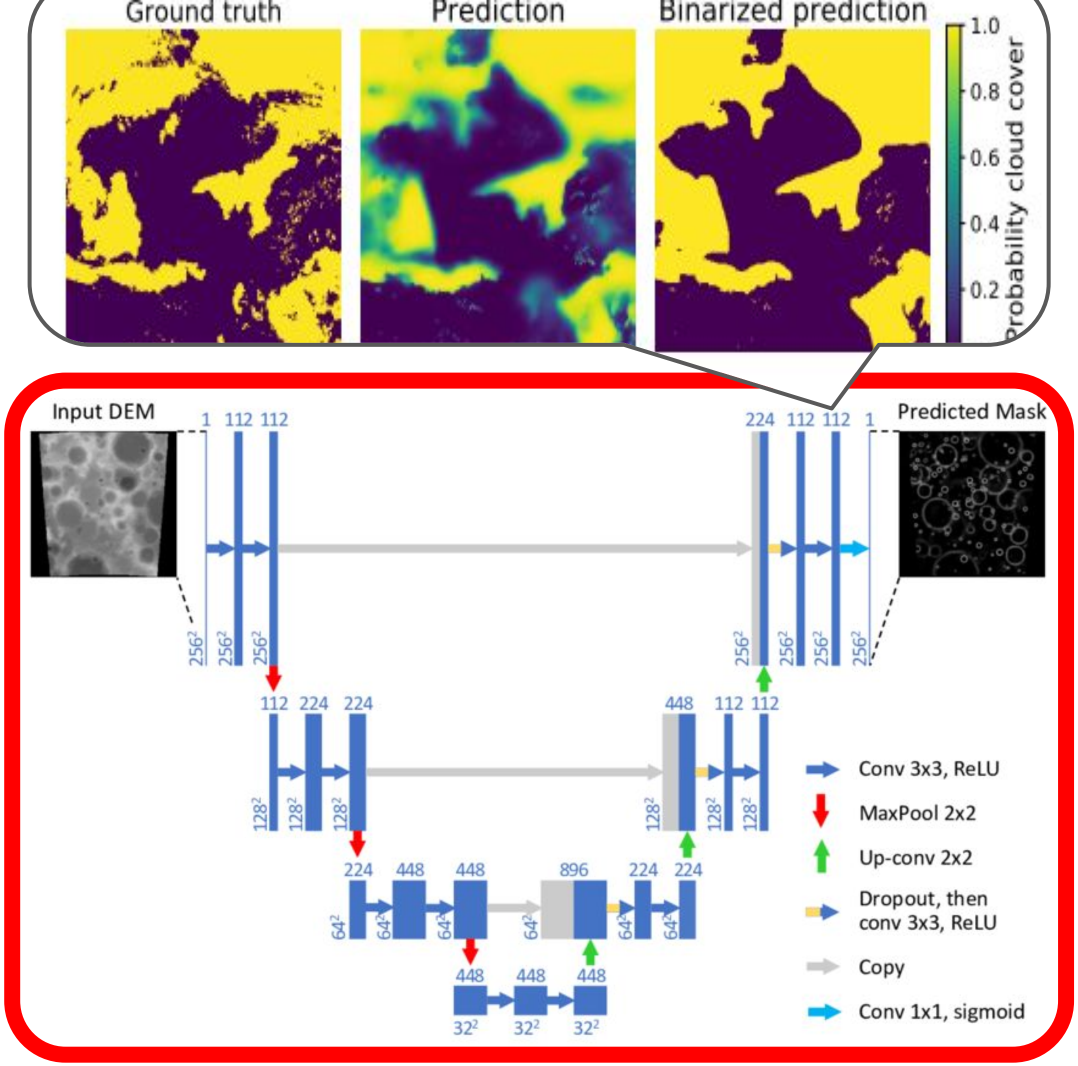


Conclusions and Future Works

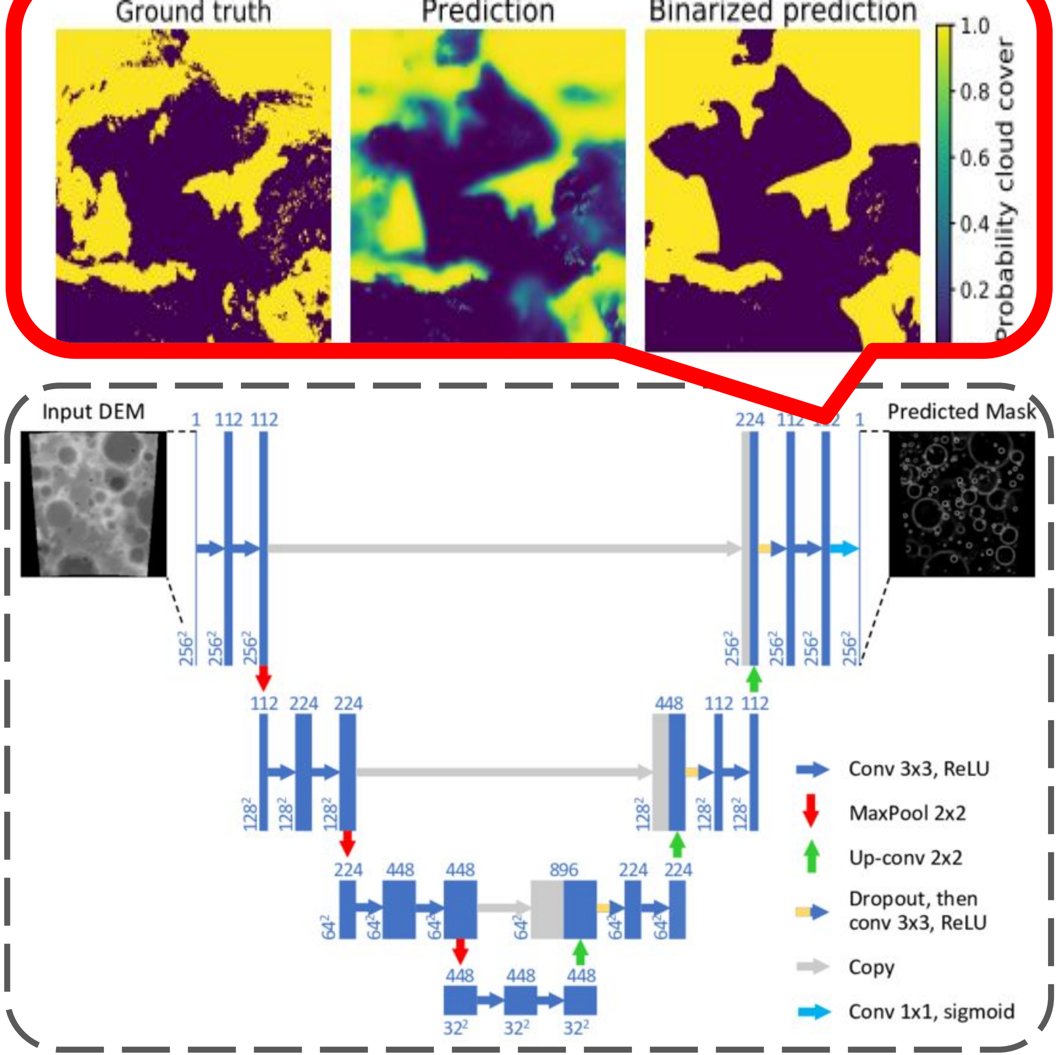
- Framework based on indirect sensing that uses high level information fusion techniques to build gateways to Internet of Bionano Things.
- Use indirect sensing measurements and high level information fusion techniques to infer about communications status.

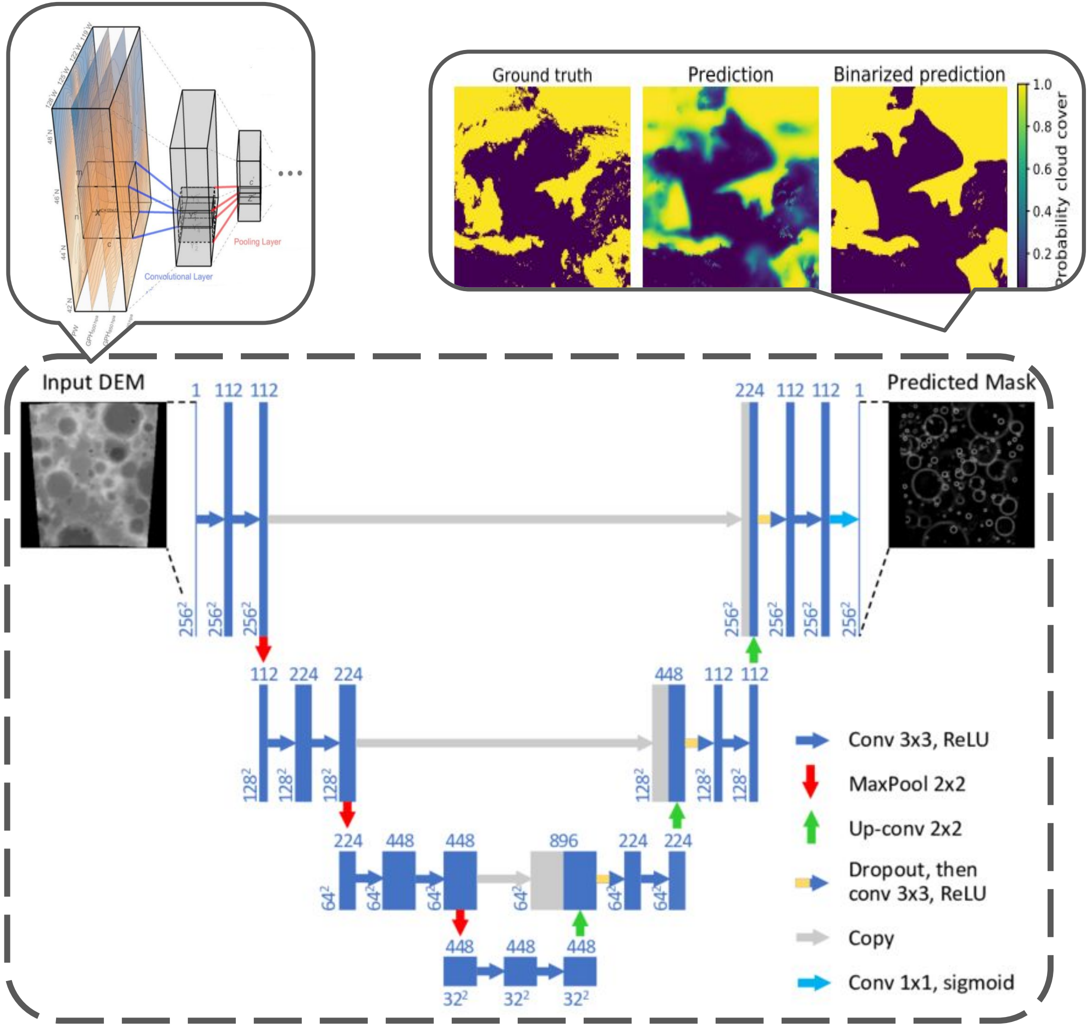
As future works we intend to:

1. Explore new DFTs to improve accuracy;
2. Use Machine Learning techniques to predict the environment behavior
3. Use the indirect sensing to enhance the biological sensor decisions by combining different types of measurements.

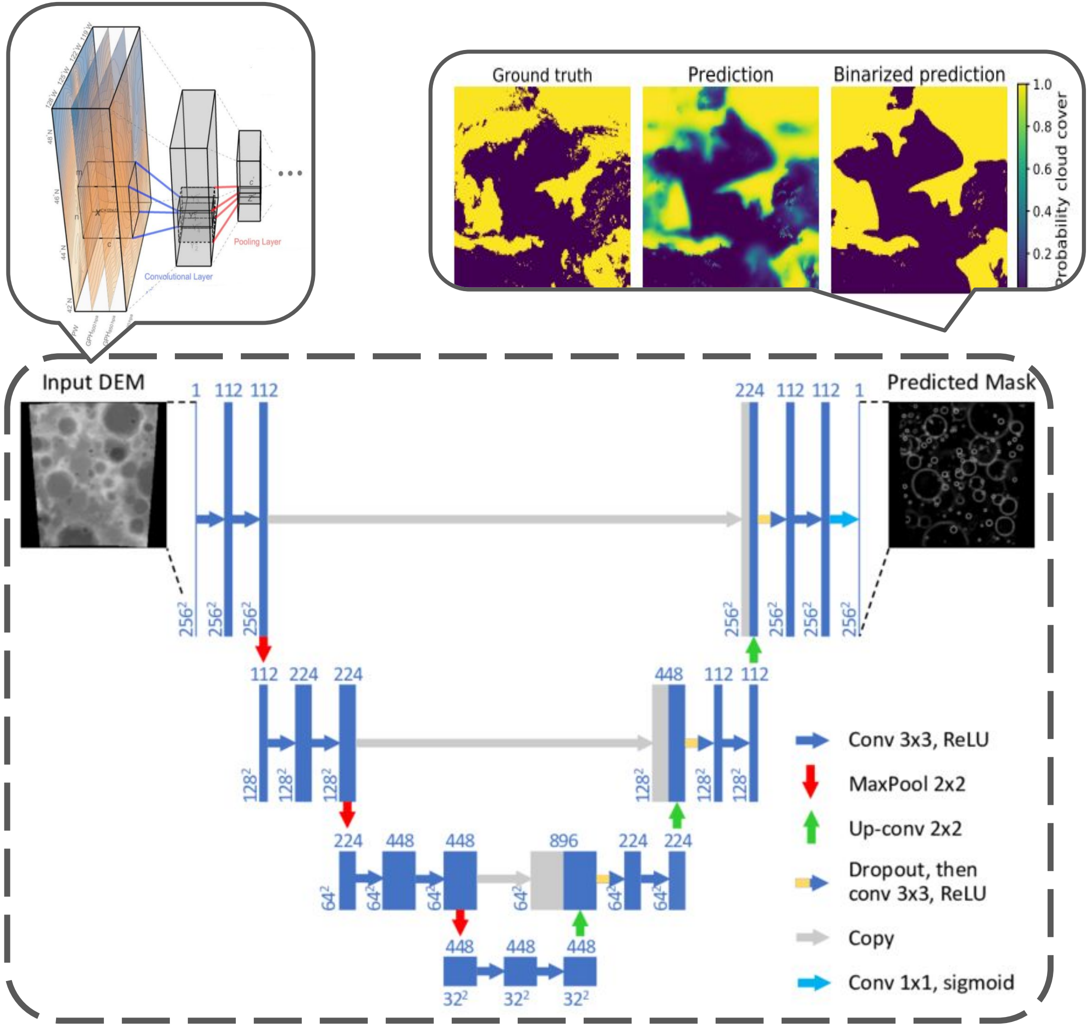


Ground truth
Prediction
Binarized prediction
1.0
0.8
0.6
0.4
0.2
Probability cloud cover
Input DEM
Predicted Mask
Conv 3x3, ReLU
MaxPool 2x2
Up-conv 2x2
Dropout, then conv 3x3, ReLU
Copy
Conv 1x1, sigmoid

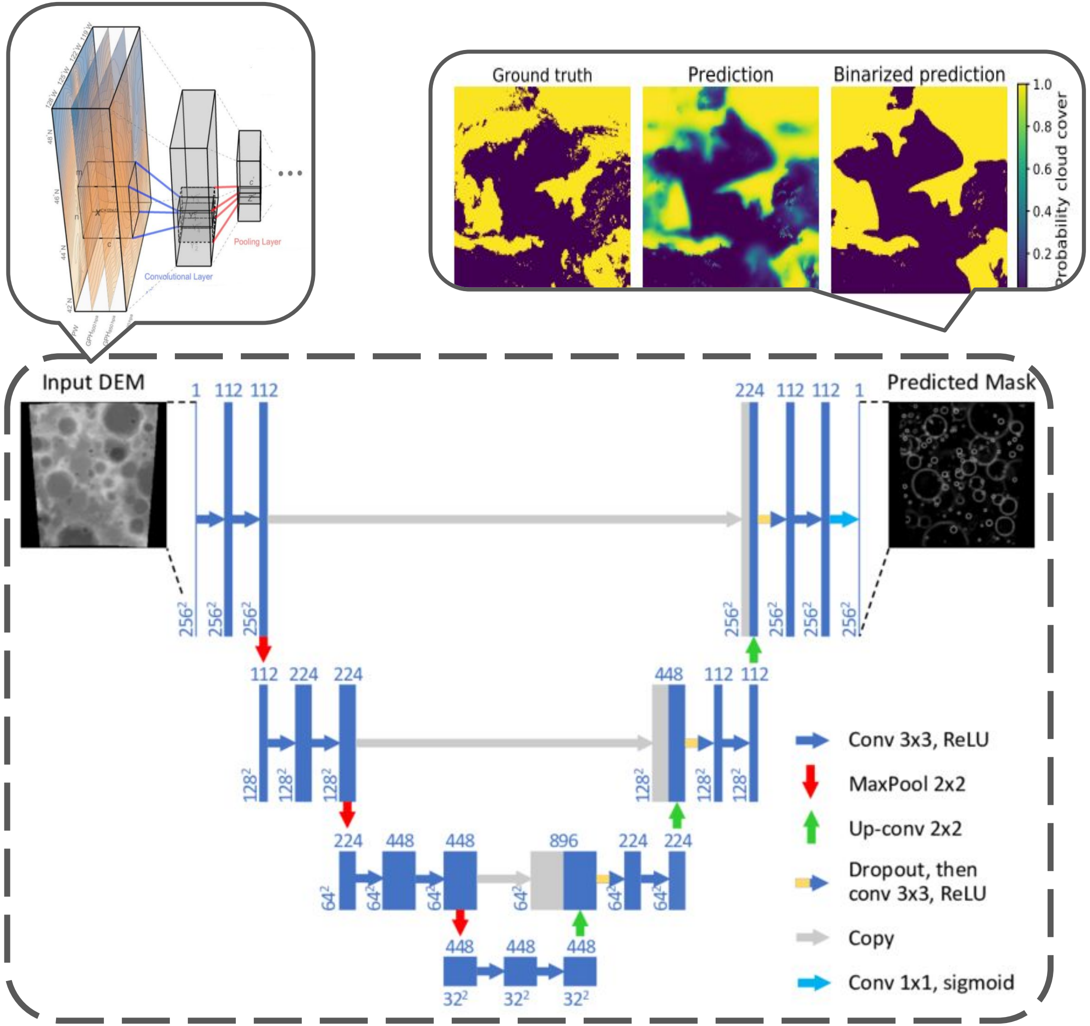

Ground truth
Prediction
Binarized prediction
1.0
0.8
0.6
0.4
0.2
Probability cloud cover
Input DEM
Predicted Mask
Conv 3x3, ReLU
MaxPool 2x2
Up-conv 2x2
Dropout, then conv 3x3, ReLU
Copy
Conv 1x1, sigmoid



Convolutional Layer
Pooling Layer
Ground truth
Prediction
Binarized prediction
1.0
0.8
0.6
0.4
0.2
Probability cloud cover
Input DEM
Predicted Mask
1 112 112
224 112 112 1
256²
112 224 224
448 112 112
128²
224 448 448
896 224 224
64²
448 448 448
32²
Conv 3x3, ReLU
MaxPool 2x2
Up-conv 2x2
Dropout, then conv 3x3, ReLU
Copy
Conv 1x1, sigmoid



Pooling Layer
Convolutional Layer
Ground truth
Prediction
Binarized prediction
Probability cloud cover
1.0
0.8
0.6
0.4
0.2
Input DEM
Predicted Mask
Conv 3x3, ReLU
MaxPool 2x2
Up-conv 2x2
Dropout, then conv 3x3, ReLU
Copy
Conv 1x1, sigmoid



Convolutional Layer
Pooling Layer
Ground truth
Prediction
Binarized prediction
Probability cloud cover
1.0
0.8
0.6
0.4
0.2
Input DEM
Predicted Mask
1
112
112
224
112
112
1
256²
256²
256²
256²
256²
256²
256²
112
224
224
448
112
112
128²
128²
128²
128²
128²
128²
224
448
448
896
224
224
64²
64²
64²
64²
64²
64²
448
448
448
32²
32²
32²
Conv 3x3, ReLU
MaxPool 2x2
Up-conv 2x2
Dropout, then conv 3x3, ReLU
Copy
Conv 1x1, sigmoid

new global
model
local
updates
local
updates
local data
local data
local data
local data
local data
local data
local data
local data
local data
learnt model:
next-word prediction
Subject
Thank you for the feedback
tab

Data Fusion in Tinyml and Applications in Biology and Federated Learning

Claudio Miceli de Farias -Universidade Federal do Rio de Janeiro
cmicelifarias@cos.ufrj.br