Anomaly Detection with Spiking Neural Networks

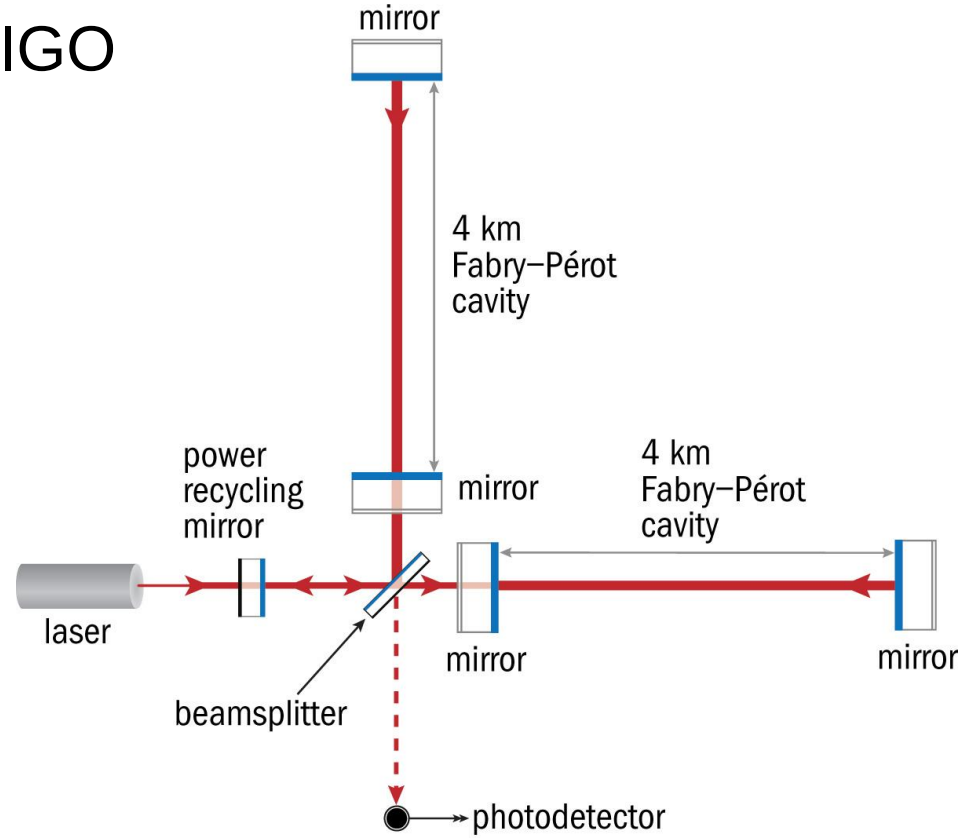
- Bartlomiej Borzyszkowski (Gdansk University of Technology, Intel Poland)
- Eric Moreno (Caltech)

Mentors: Jean-Roch Vlimant, Maurizio Pierini, Vladimir Loncar

September 4th, 2020

Introduction to the project

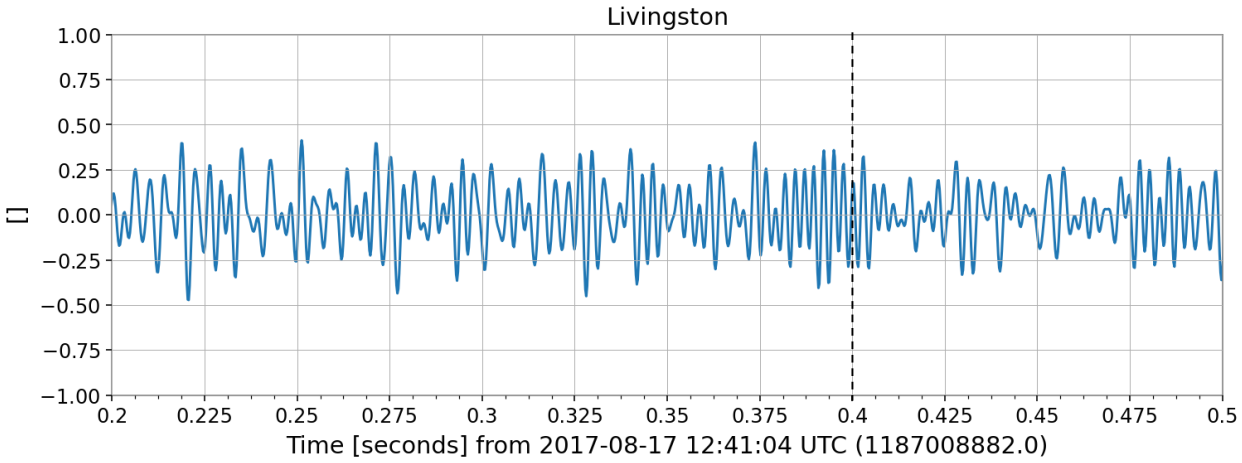
- Detection of gravitational waves (GWs) at LIGO

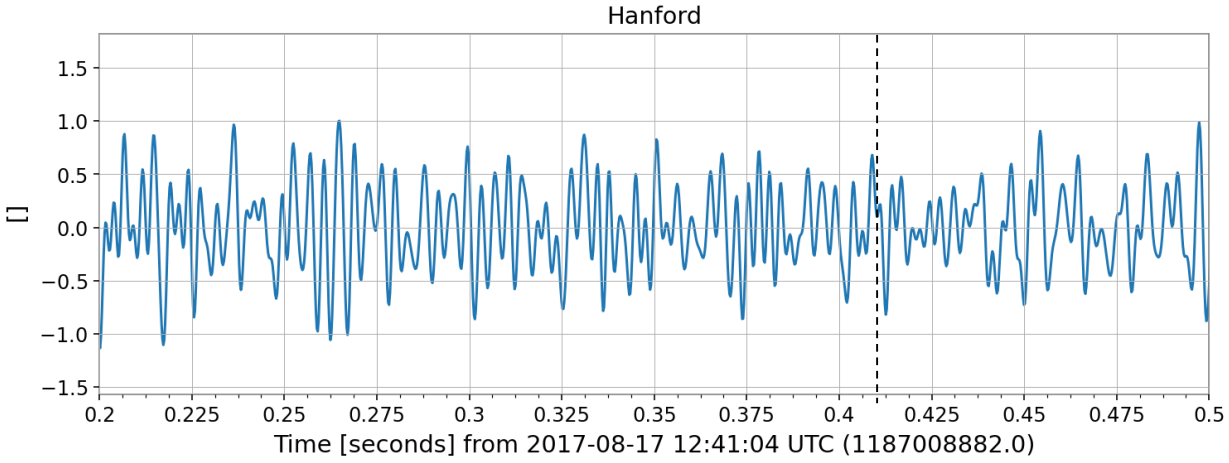


Produces: 1-D time-series strain

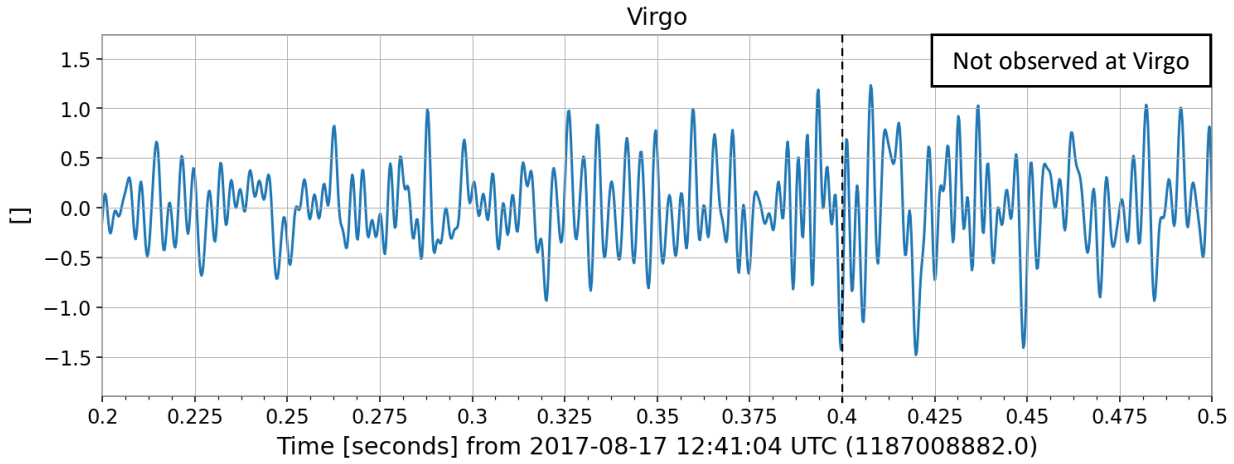
Source: Elena Cuoco - Real Time Classifier for transient signals in Gravitational Waves, From raw data to classified triggers

LIGO Data Generation





Data is whitened and bandpassed



Source: https://www.gw-openscience.org/data/

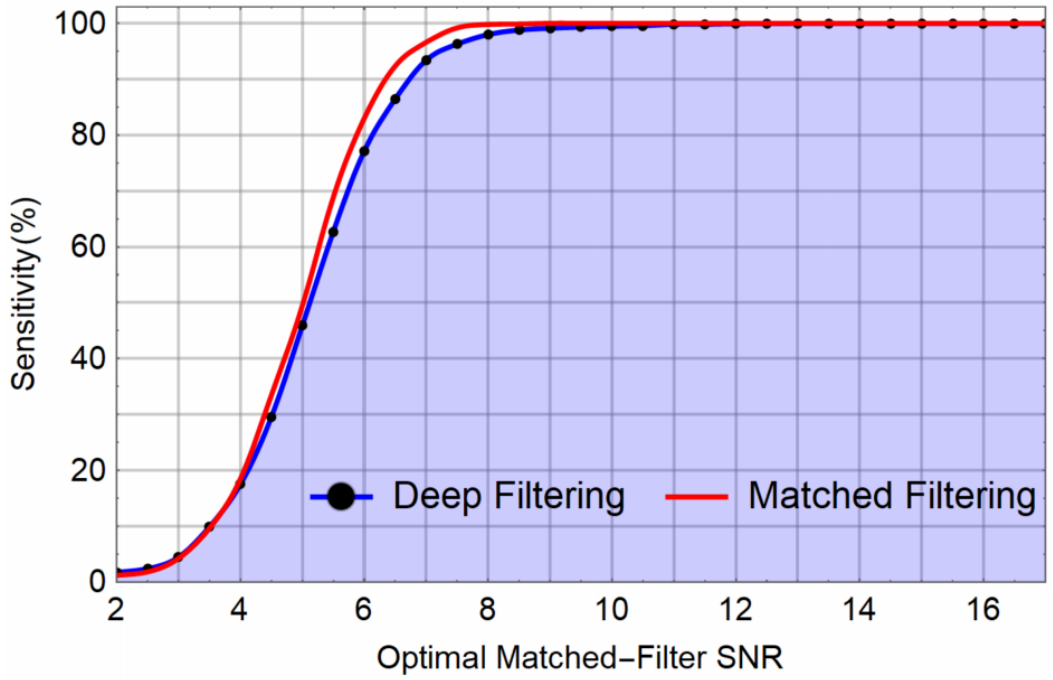
Current methods

Matched Filtering (used at LIGO)

- Compares incoming GW data to bank of simulated waveforms
- Can only identify GWs that are available in GW banks (no exotic events)

Deep Filtering (CNNs)

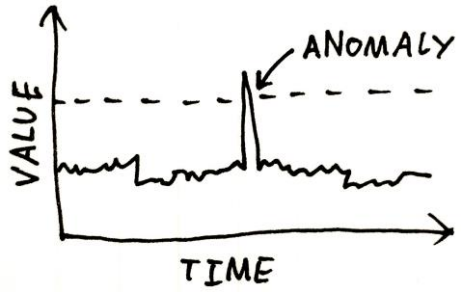
- Take time-series inputs, can determine detections and estimate parameters of events
- Still can miss events that aren't included in training set



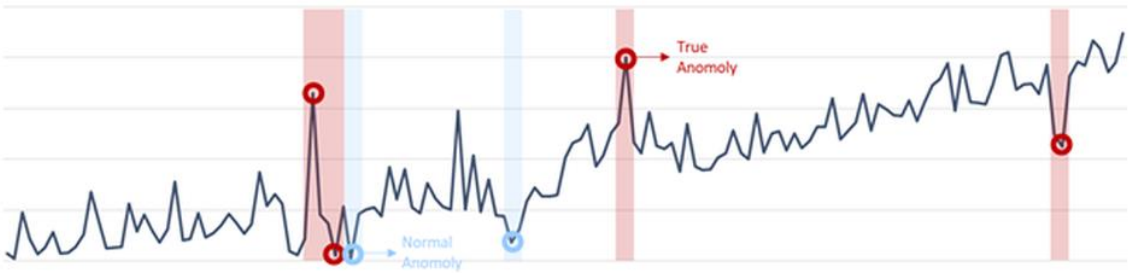
Source: https://arxiv.org/pdf/1711.03121.pdf, https://arxiv.org/pdf/2005.06534.pdf

Artificial Neural Networks

1. *Unsupervised learning*
   - *unlabeled data*
   - *autoencoders*



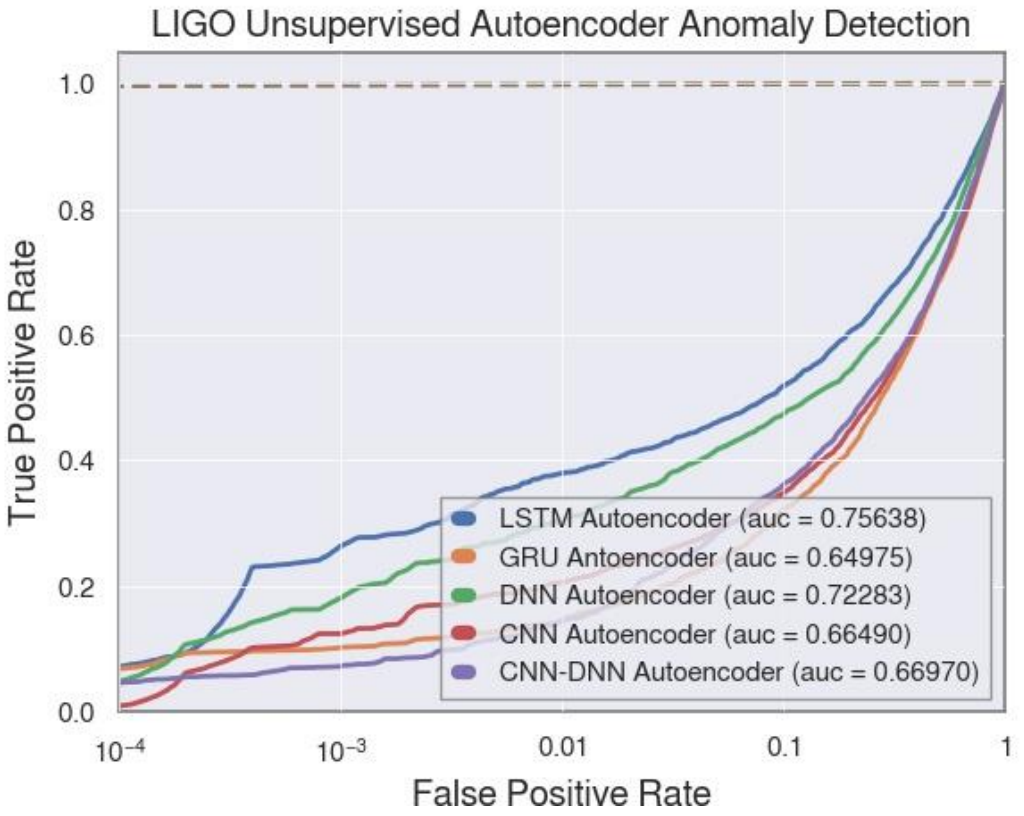
2. *Supervised learning*
   - *labels for gravitational waves*
   - *deep neural networks*

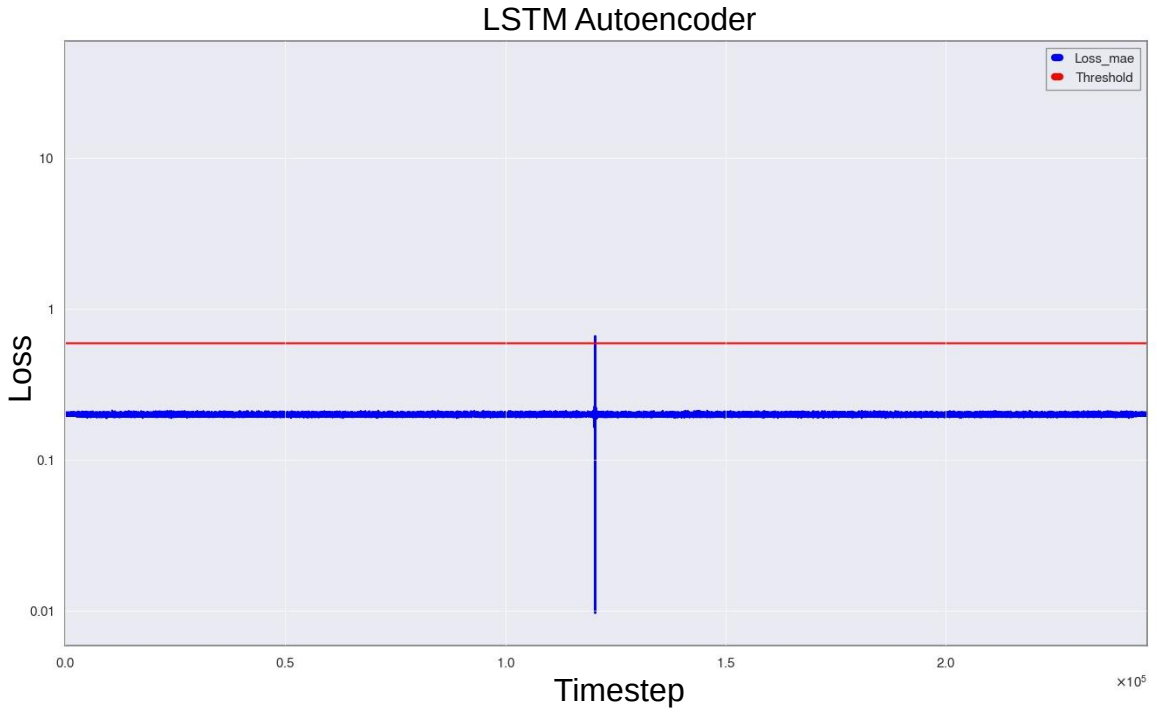


3. *Fast inference*
   - *hls4ml with oneAPI backend*
   - *hardware acceleration*

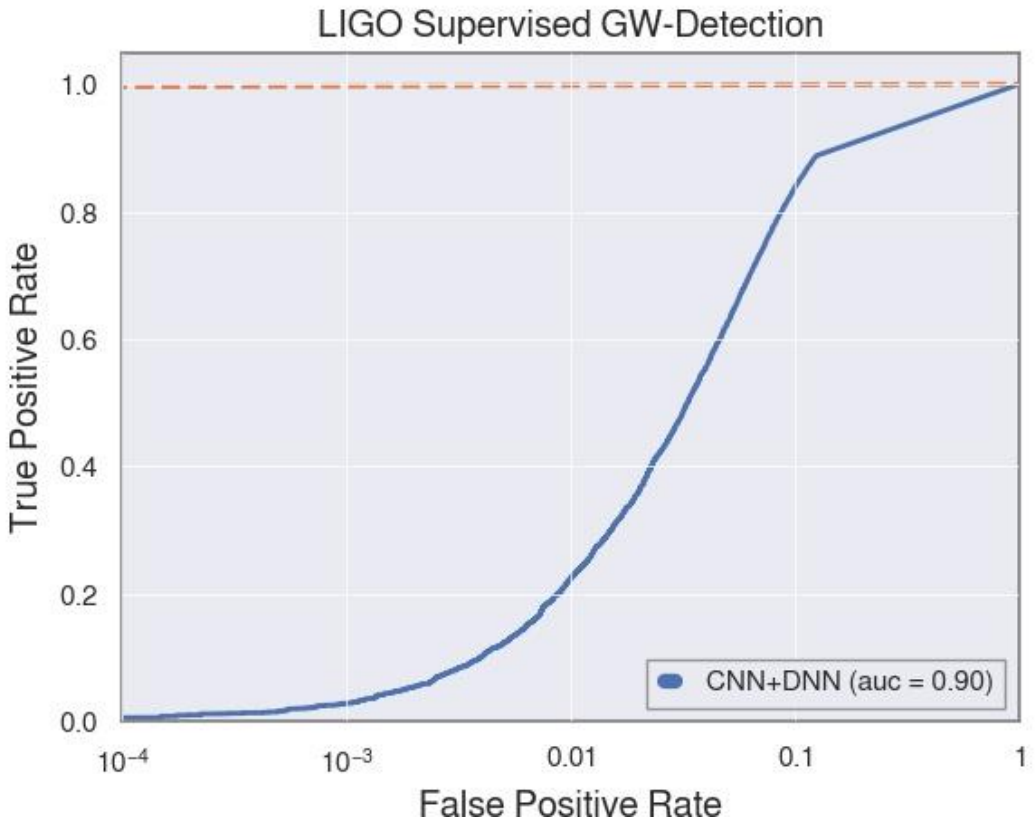
hls4ml + oneAPI

ANNs: Unsupervised learning



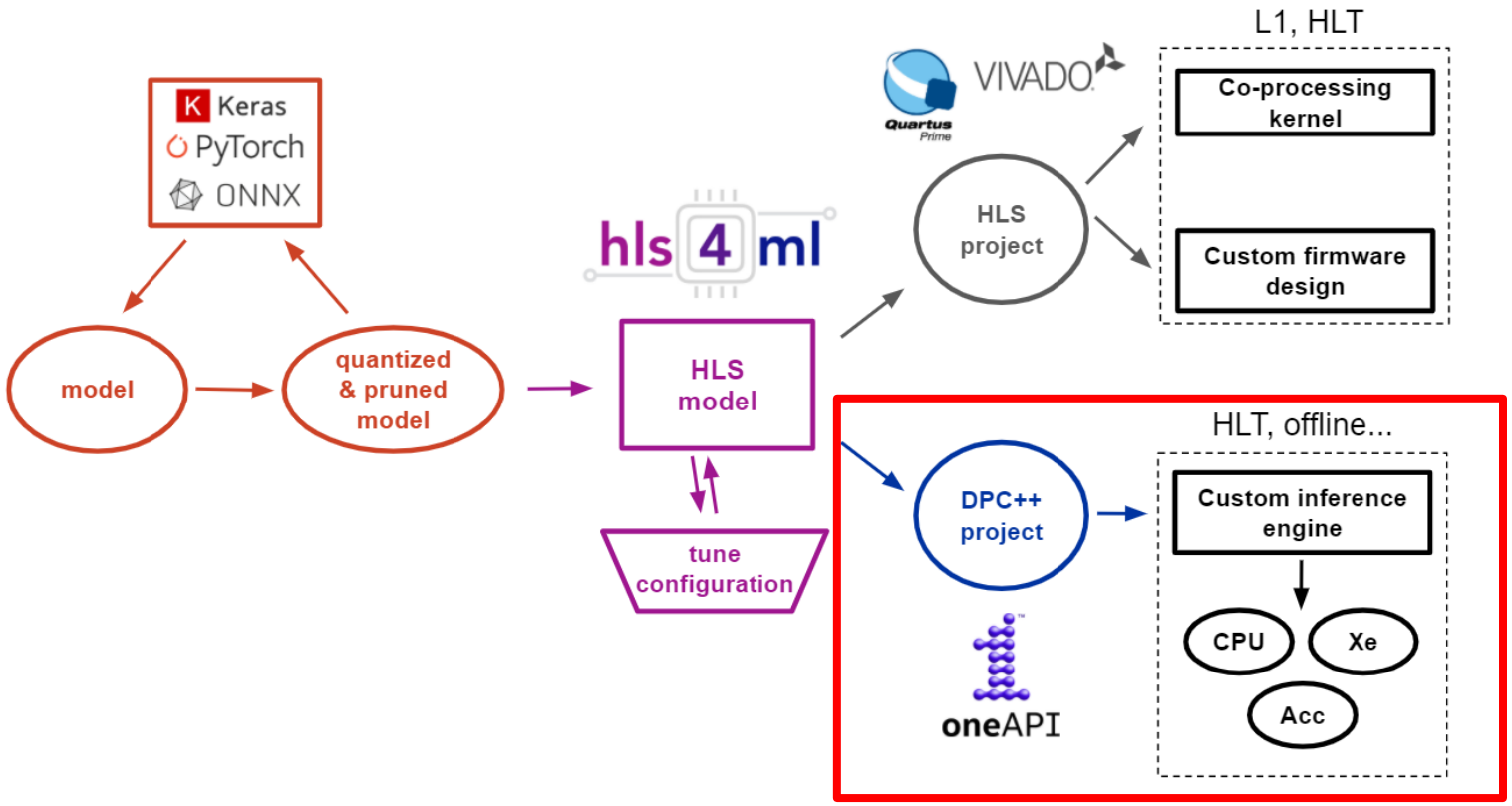


ANNs: Supervised learning



	Input	vector (size: 8192)
1	Reshape	matrix (size: 1×8192)
2	Convolution	matrix (size: 64×8177)
3	Pooling	matrix (size: 64×2044)
4	ReLU	matrix (size: 64×2044)
5	Convolution	matrix (size: 128×2014)
6	Pooling	matrix (size: 128×503)
7	ReLU	matrix (size: 128×503)
8	Convolution	matrix (size: 256×473)
9	Pooling	matrix (size: 256×118)
10	ReLU	matrix (size: 256×118)
11	Convolution	matrix (size: 512×56)
12	Pooling	matrix (size: 512×14)
13	ReLU	matrix (size: 512×14)
14	Flatten	vector (size: 7168)
15	Linear Layer	vector (size: 128)
16	ReLU	vector (size: 128)
17	Linear Layer	vector (size: 64)
18	ReLU	vector (size: 64)
19	Linear Layer	vector (size: 2)
	Output	vector (size: 2)

Inference engine with hls4ml and Intel® oneAPI



Currently supported layers:

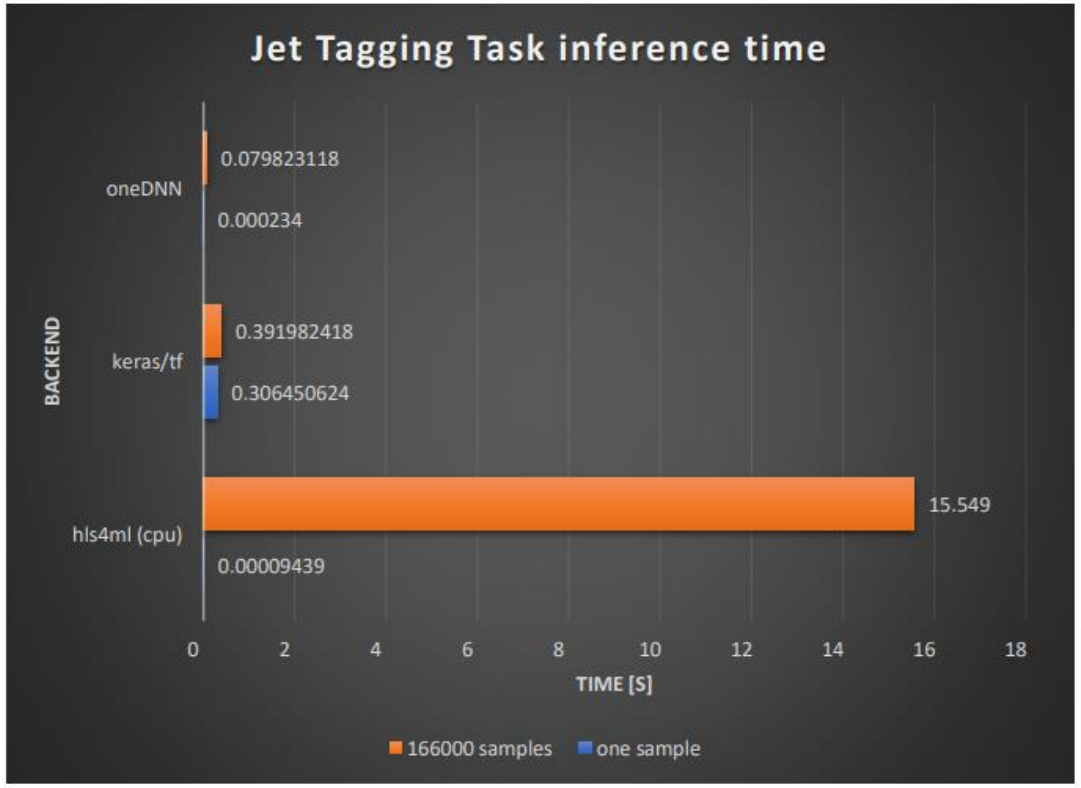
- Dense
- Element-wise functions (ReLU, tanh, log, linear exp, sqrt and others)
- Softmax
- Conv1d, Conv2d
- Pooling1d, Pooling2d

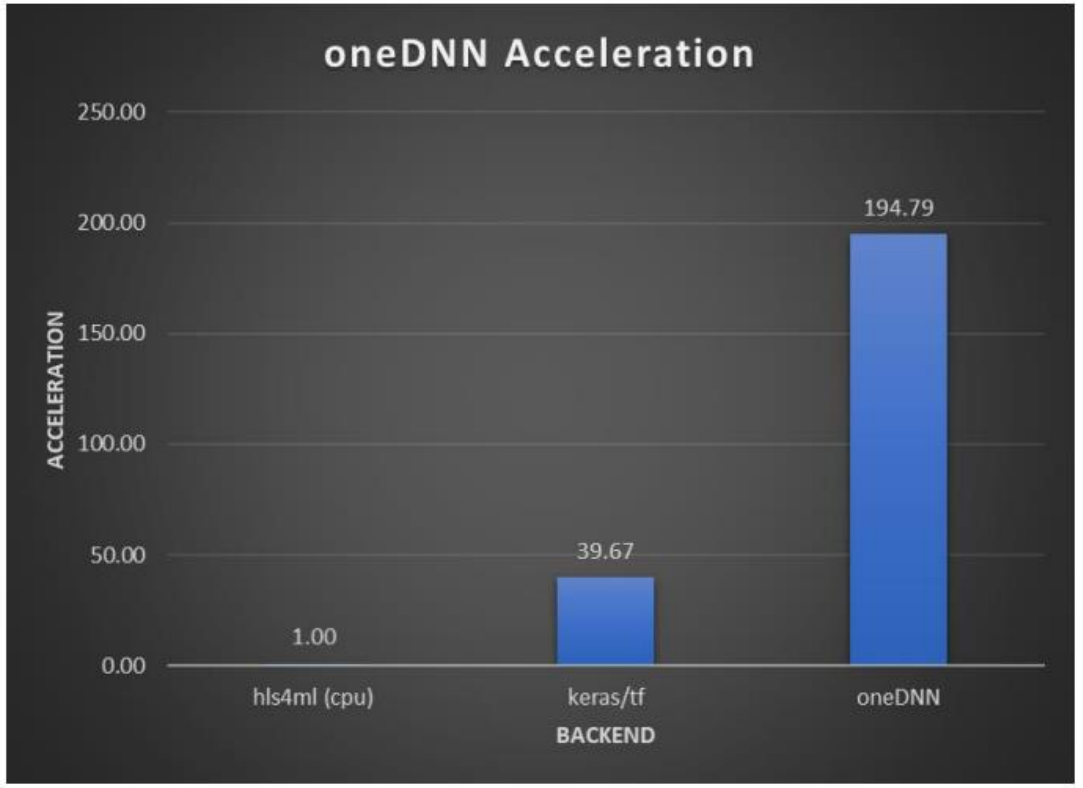
Anomaly Detection
runs in hls4ml with oneAPI backend

Credits: Marcin Świniarski, Vladimir Loncar;

Intel® oneAPI baseline model performance

Three hidden-layer neural network on Xeon Gold 6128 3.4Ghz

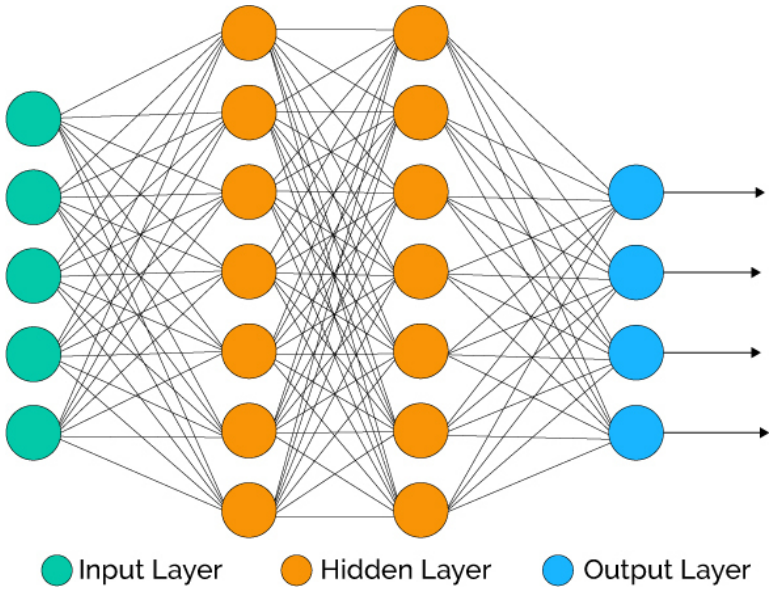


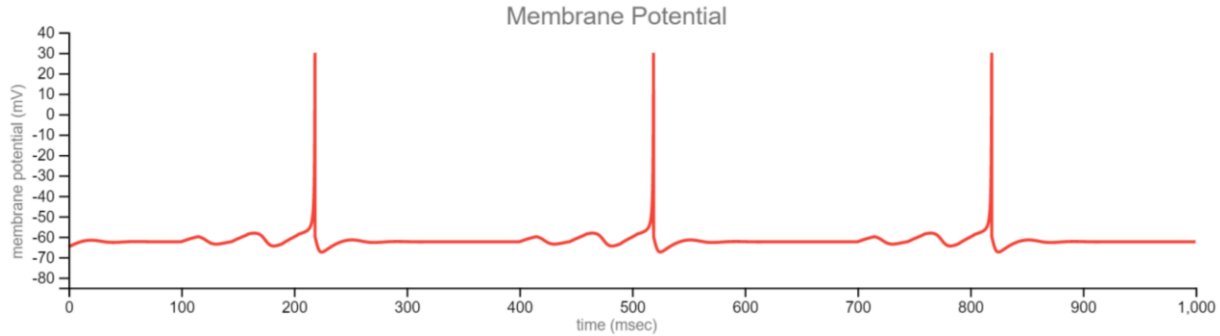


Credits: Marcin Świniarski, Vladimir Loncar;

Spiking Neural Networks (SNNs)

- Inspired by biological processing of information (mimic human brains)
- Neurons consist of signal and activate at a certain threshold (non-differentiable)

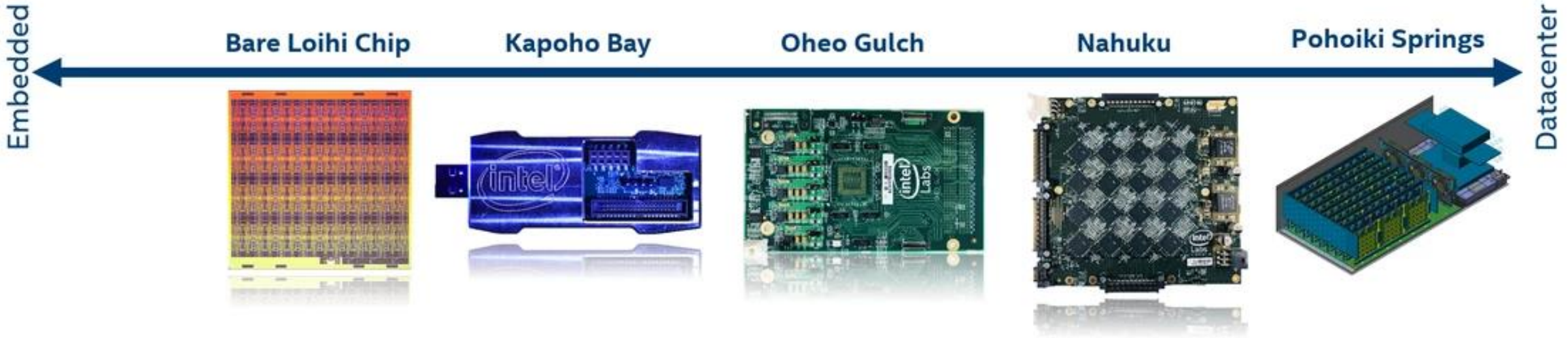




Membrane potential of a single spiking neuron. The neuron is firing when the action potential achieves a specific threshold. After activation the signal returns to the low, regular value.

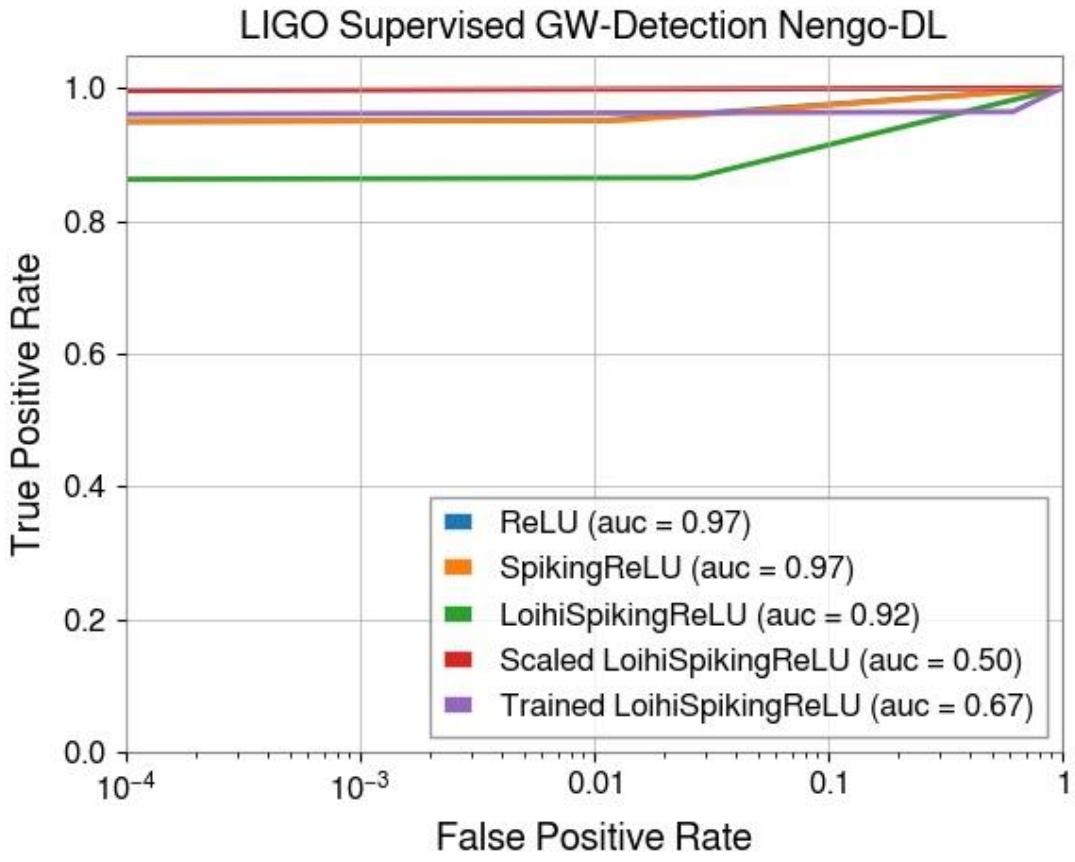
Source: http://jackterwilliger.com/biological-neural-networks-part-i-spiking-neurons/

Intel Loihi – Dedicated neuromorphic chip



Source: https://intel-ncl.atlassian.net/wiki/spaces/INRC/pages/2752539/Loihi+Systems+Overview

SNNs supervised learning - Nengo



Layer (type)	Output Shape	Param #
input (InputLayer)	[(None, 2048)]	0
reshape (Reshape)	(None, 2048, 1, 1)	0
conv2d (Conv2D)	(None, 2045, 1, 16)	64
conv2d_1 (Conv2D)	(None, 511, 1, 16)	1024
conv2d_2 (Conv2D)	(None, 127, 1, 32)	2048
conv2d_3 (Conv2D)	(None, 31, 1, 64)	8192
conv2d_4 (Conv2D)	(None, 6, 1, 128)	65536
flatten (Flatten)	(None, 768)	0
dense (Dense)	(None, 128)	98304
dense_1 (Dense)	(None, 64)	8192
output (Dense)	(None, 2)	130

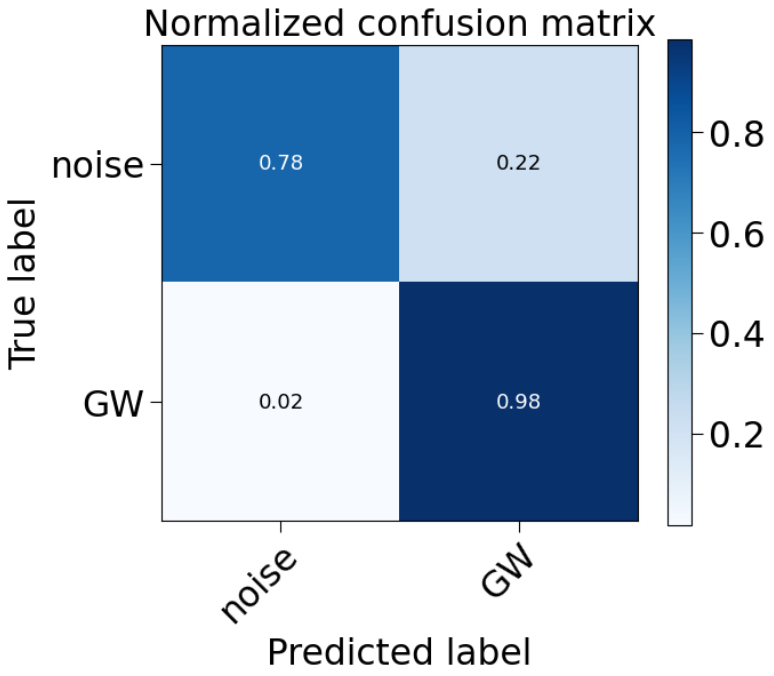
Total params: 183,490
Trainable params: 183,490
Non-trainable params: 0

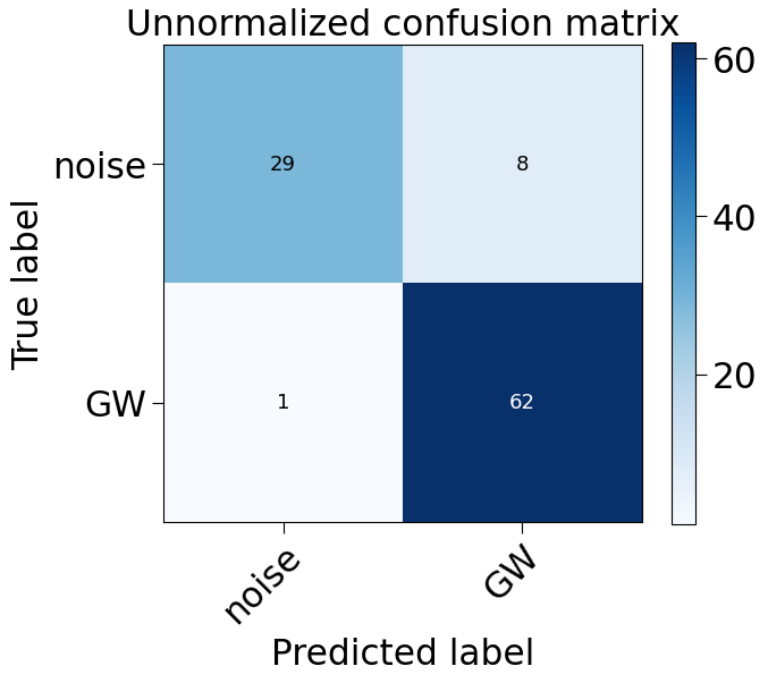
→ Off-chip

ON-chip

Executed on Loihi chip

SNNs supervised learning – SNN-TB






```
Layer (type)                 Output Shape              Param #
=================================================================
input (InputLayer)           [(10, 2025)]              0
_________________________________________________________________
00Reshape_2025x1x1 (Reshape) (10, 2025, 1, 1)          0
_________________________________________________________________
01Conv2D_2022x1x16 (Conv2D)  (10, 2022, 1, 16)         80
_________________________________________________________________
02Conv2D_505x1x32 (Conv2D)   (10, 505, 1, 32)          2080
_________________________________________________________________
03Conv2D_126x1x64 (Conv2D)   (10, 126, 1, 64)          8256
_________________________________________________________________
04Conv2D_31x1x128 (Conv2D)   (10, 31, 1, 128)          32896
_________________________________________________________________
05Flatten_3968 (Flatten)     (10, 3968)                0
_________________________________________________________________
06Dense_128 (Dense)          (10, 128)                 508032
_________________________________________________________________
07Dense_64 (Dense)           (10, 64)                  8256
_________________________________________________________________
08Dense_2 (Dense)            (10, 2)                   130
=================================================================
Total params: 559,730
Trainable params: 559,730
Non-trainable params: 0
_________________________________________________________________
Evaluating parsed model on 1000 samples...
Top-1 accuracy: 94.30%
Top-5 accuracy: 100.00%

Building spiking model...
Building layer: 00Reshape_2025x1x1
Building layer: 01Conv2D_2022x1x16
Building layer: 02Conv2D_505x1x32
Building layer: 03Conv2D_126x1x64
Building layer: 04Conv2D_31x1x128
Building layer: 05Flatten_3968
Building layer: 06Dense_128
Building layer: 07Dense_64
Building layer: 08Dense_2
Compiling spiking model...
```

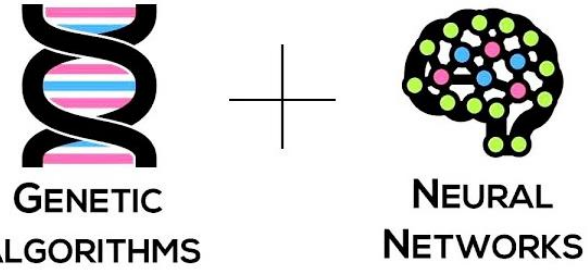
Accuracy results in simualtion:
ANN: 94%
SNN: 91%

Executed on Loihi chip

Work under development & opportunities

- Unsupervised learning with SNNs (Recurrent SNNs: LSTMs, LMUs)
- Backpropagation algorithm for direct training of SNNs (SLAYER)
- Evolutionary algorithims to adjust the parameters of SNNs

- Scientific Publication (ongoing)
- Work on LHC data for anomaly detection



Thank you for your attention!

Do you have any questions?

bartek_borzyszkowski@wp.pl

github.com/eric-moreno/Anomaly-Detection-Autoencoder