3rd AGATA-GRETINA/GRETA Collaboration meeting, 2-4 October 2019

GRETINA data ☛ AGATA Processing chain
&
Machine Learning

O. Stézowski

Workshop # 1 :

Following Amel & Torben's work,

Possibilities to process GRETINA data through the AGATA Data Processing chain ?

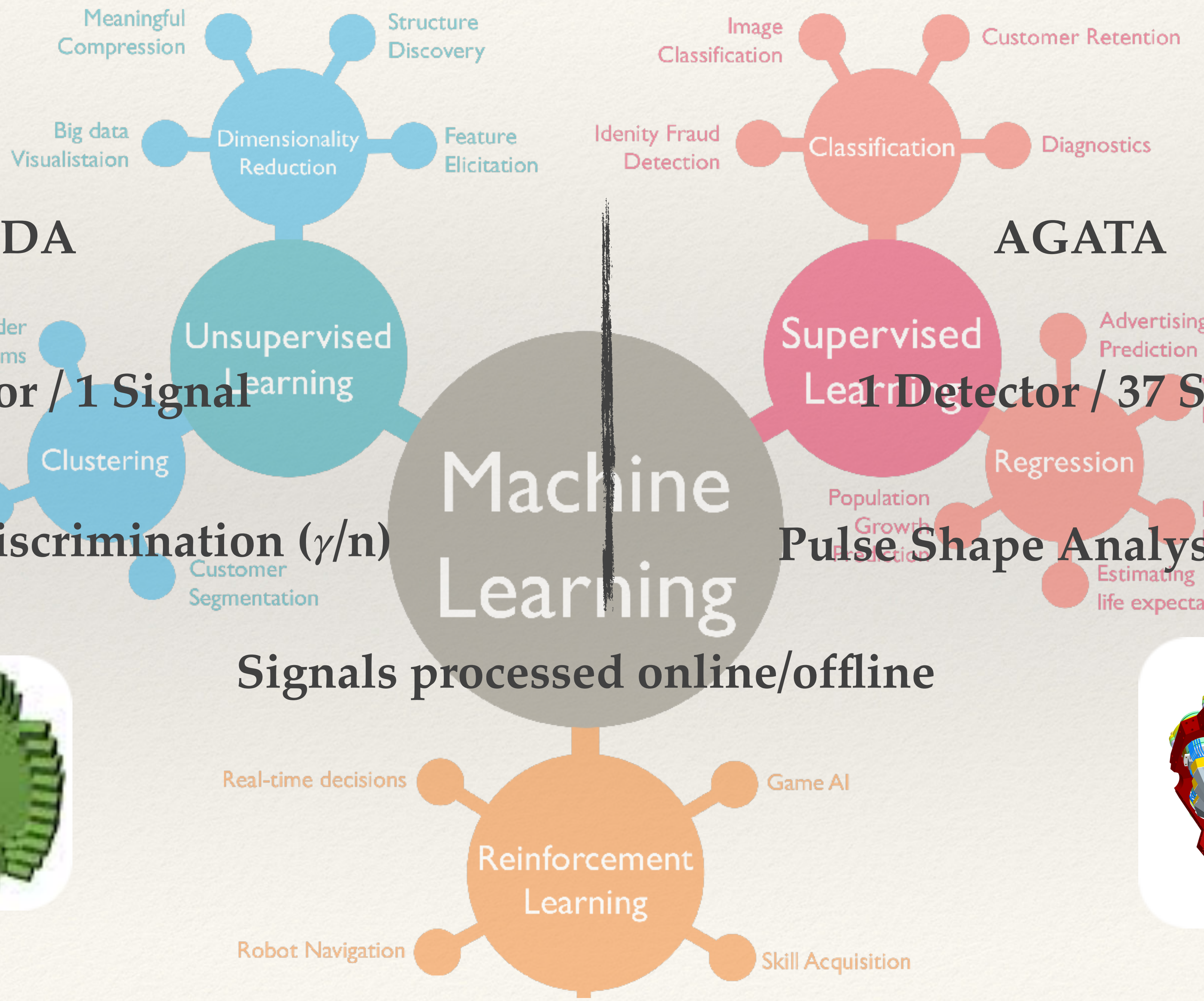
Workshop # 2:

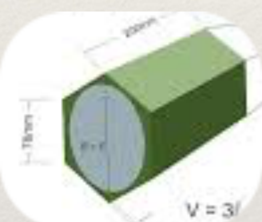
What about using Machine Learning for GRETINA/AGATA ?

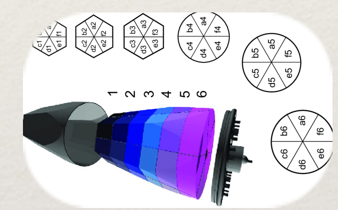
Machine Learning

Machine Learning, at lot to learn !

Our approach to learn machine learning NEDA ☛ AGATA







NEDA

1 Detector / 1 Signal

Pulse Shape Discrimination (γ/n)

AGATA

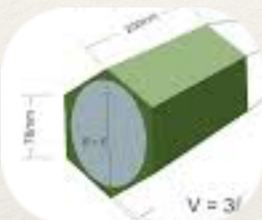
1 Detector / 37 Signals

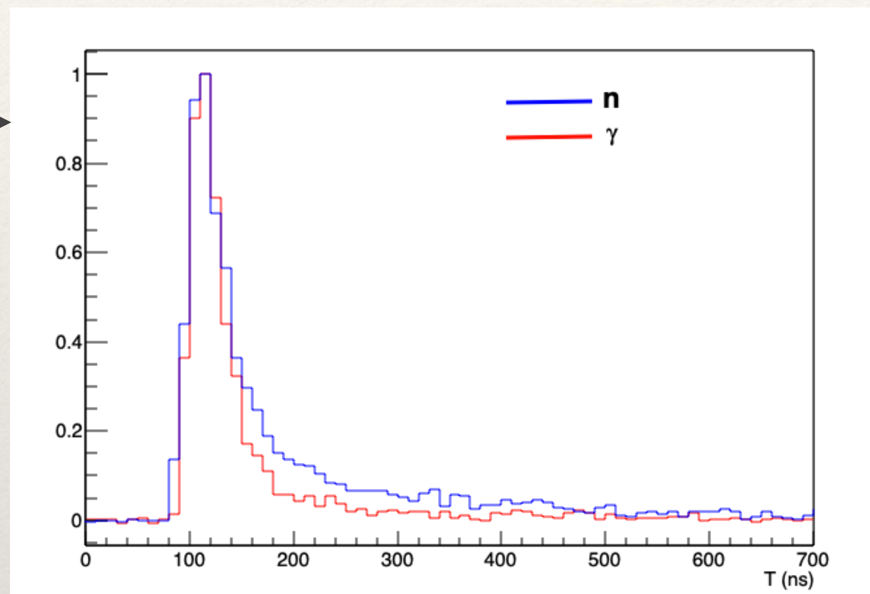
Pulse Shape Analysis (e,x,y,z)

Signals processed online/offline

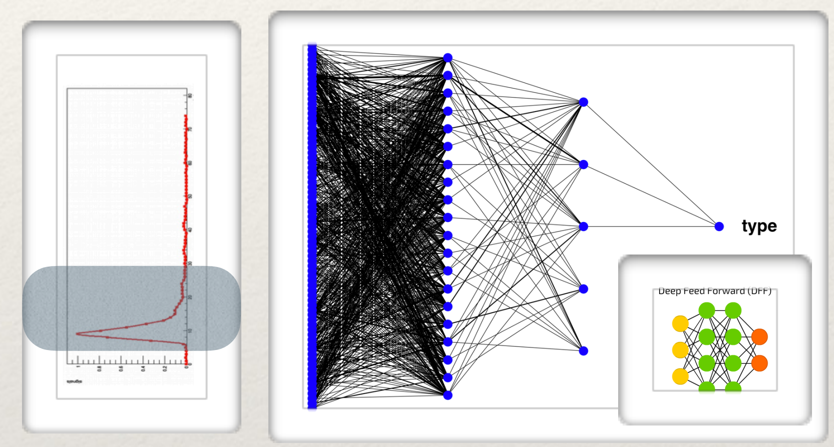
Machine Learning on signals

Pulse Shape Discrimination in NEDA





R&D NEDA, PSD with Neural network



Implementation with ROOT (monothread / CPU)

Best discrimination for low energy

Ronchi et al., A 610 (2009) 534–539

Signal parametrisation

s(t) = **A** [exp(-t/**td1**) - exp(-t/**tr**) + **R***(exp(-t/**td2**) - exp(-t/**tr**)] si t > **T0**

A amplitude

td1, **td2**, **tr** 'identical' γ & **n**

T0 depend of signals alignments

R different between γ & **n**

Machine Learning on signals

Our first work has been to run NN PSD online / offline

GANPRO

PSD

Tensorflow

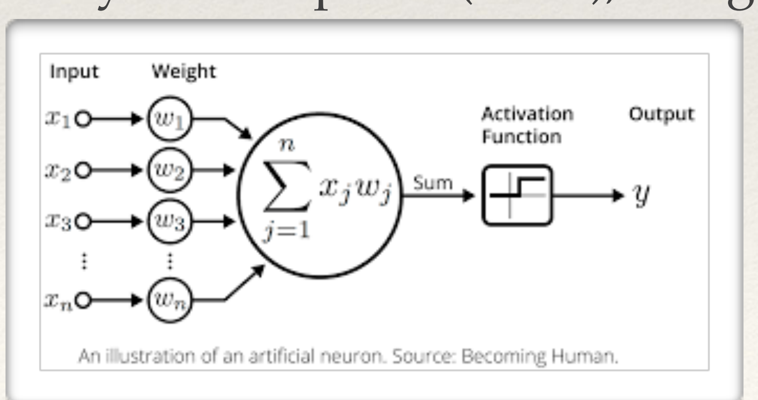
☛ We have moved from ROOT to Tensorflow/keras (python / C++)
  Python interface for training, C++ interface for inference
  The library deals with hardware, transparent to users (multi-core/CPU, GPU)
  Facteur 50 gained [on CPU] , **online inférence !**

TDC is an input of the network

We have decided to study other NN architectures

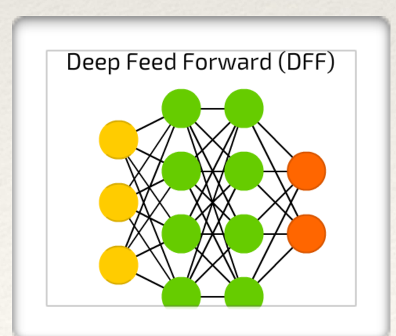
Three types of networks has been compared :

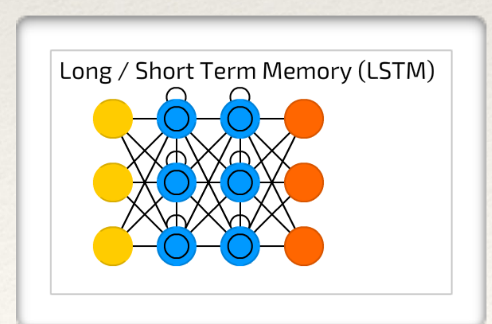
Multi Layer Perceptron (MLP), Long Short Term Memory (LSTM), Convolutional Neural Network (CNN)



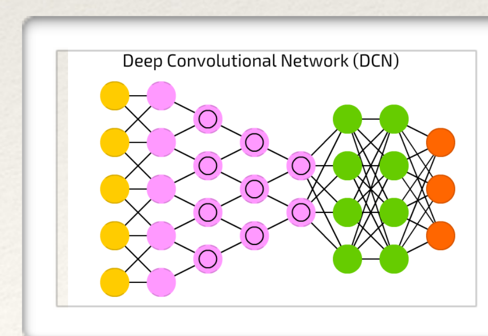
An illustration of an artificial neuron. Source: Becoming Human.

One neuron





👍 time series



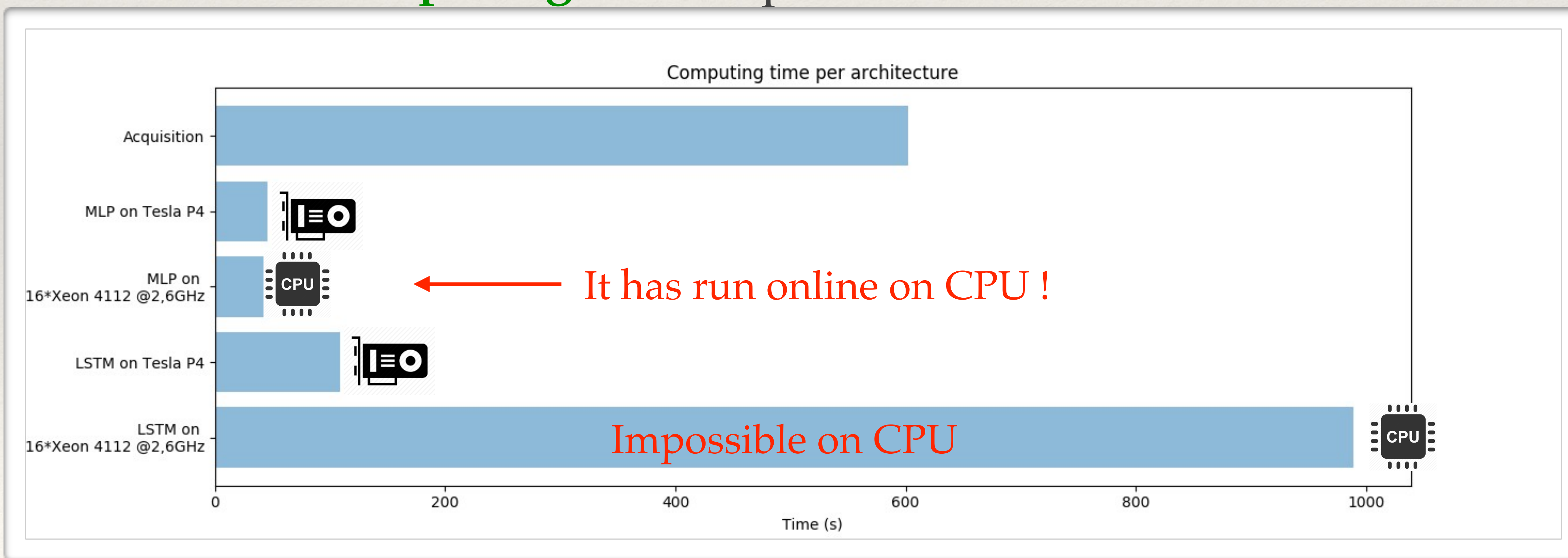
👍 pattern

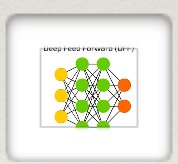
Machine Learning on signals

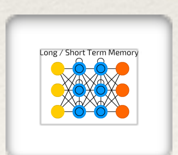
Network configurations

Network type	Structure	Activation functions	Number of trainable parameters
MLP	3 Dense layers (75 x 10 x 4 x 2)	Relu x ReLu x SoftMax	814
LSTM	75 x 1 LSTM layer (50 hidden units) x 1 Dense layer (50 x 2)	SoftMax	10 502
Convolution	75 x 3 (Conv1D+Max_Pooling) layers x 2 Dense layers (100 x 20 x 2)	ReLu x ReLu x ReLu x ReLu x SoftMax	7 042

Computing time required for inference

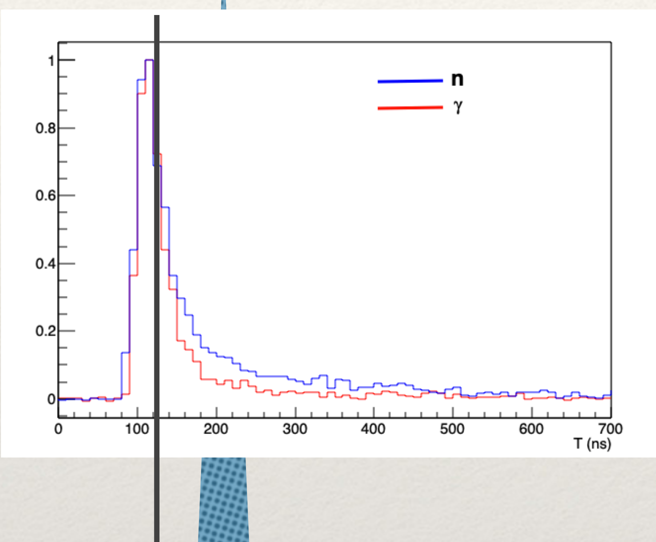




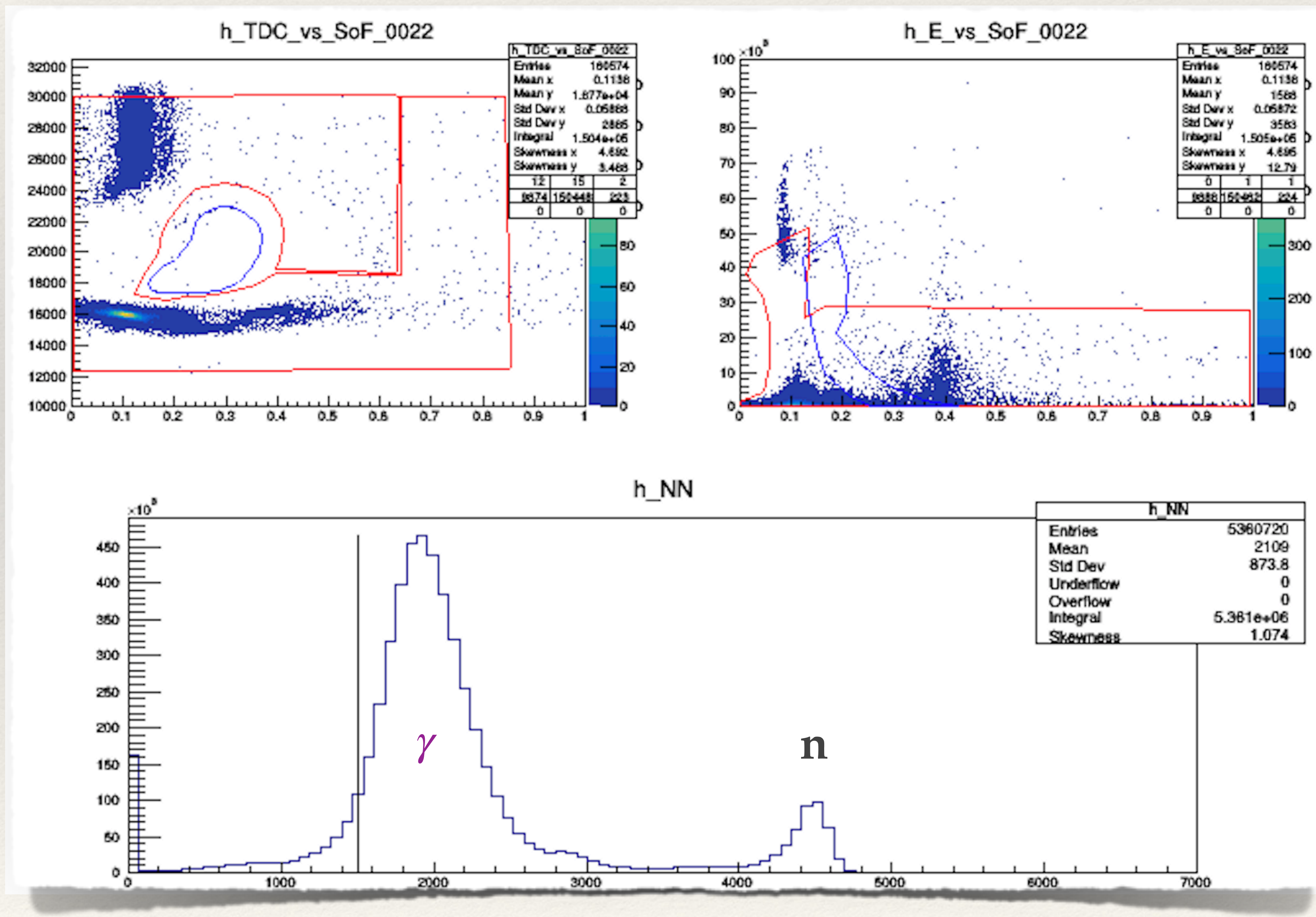


Machine Learning on signals

Training of the networks using 2 2D cuts on SoF/TDC, A/SoF



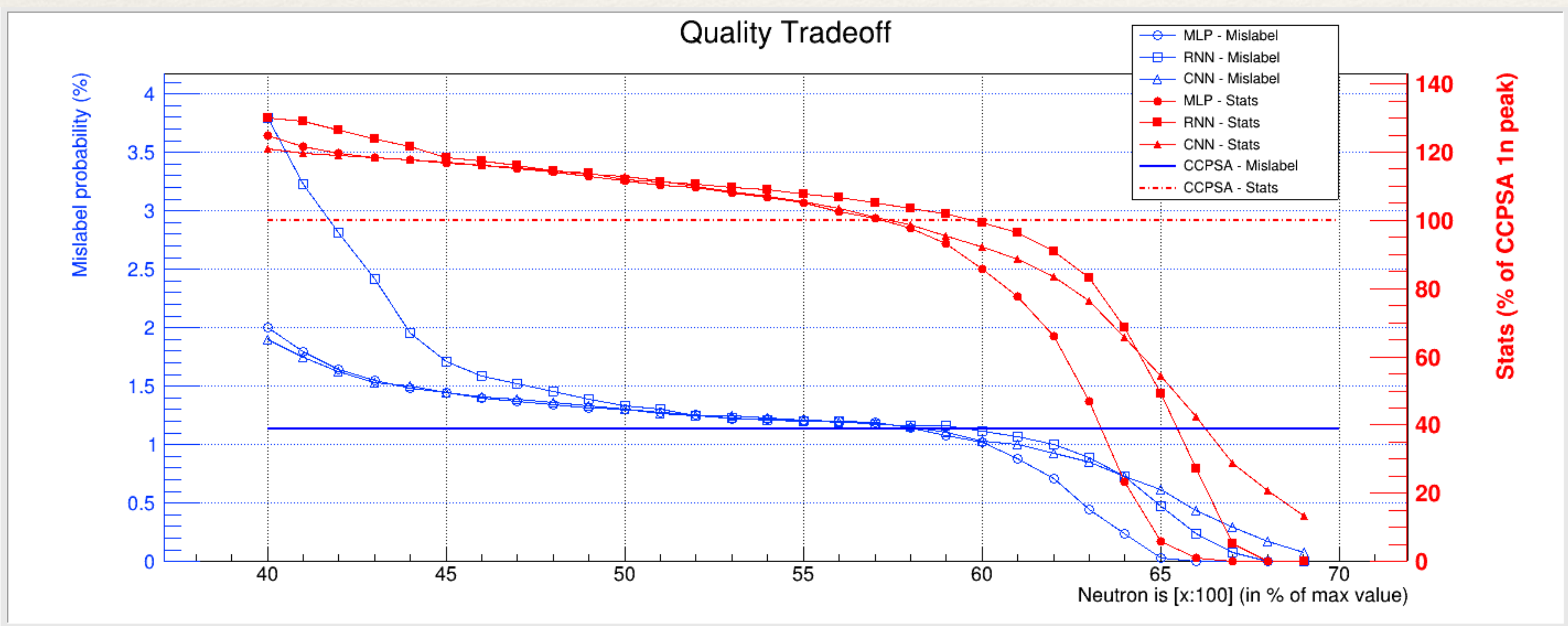
Training done using the python of Tensorflow



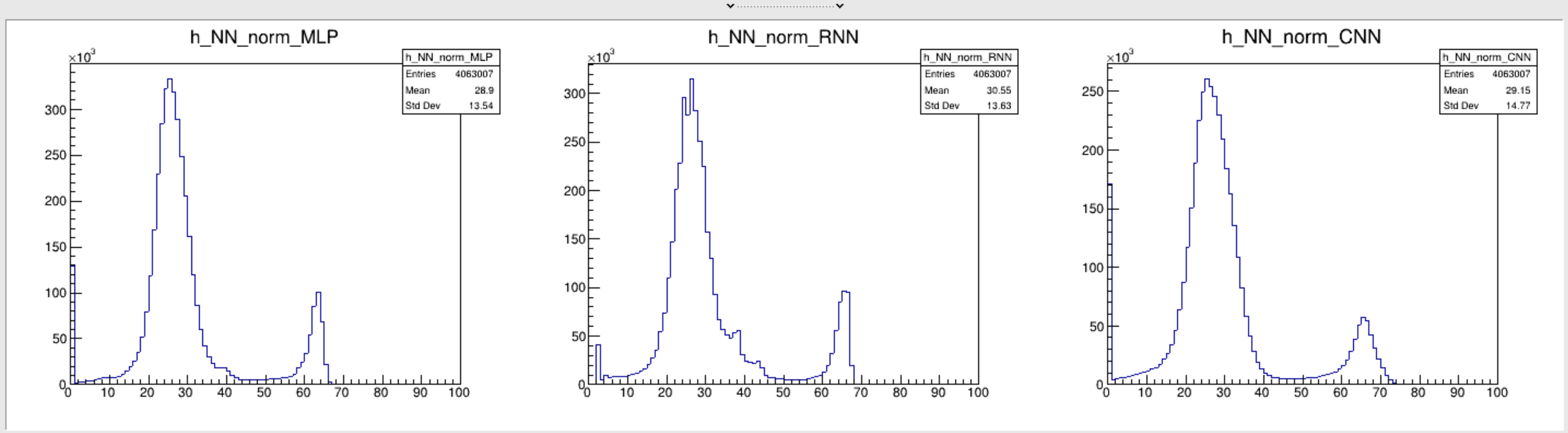
Machine Learning on signals

We have AGATA / NEDA / DIAMANT Data,

↪ AGATA γ spectra to evaluate wrong n discriminations in NEDA



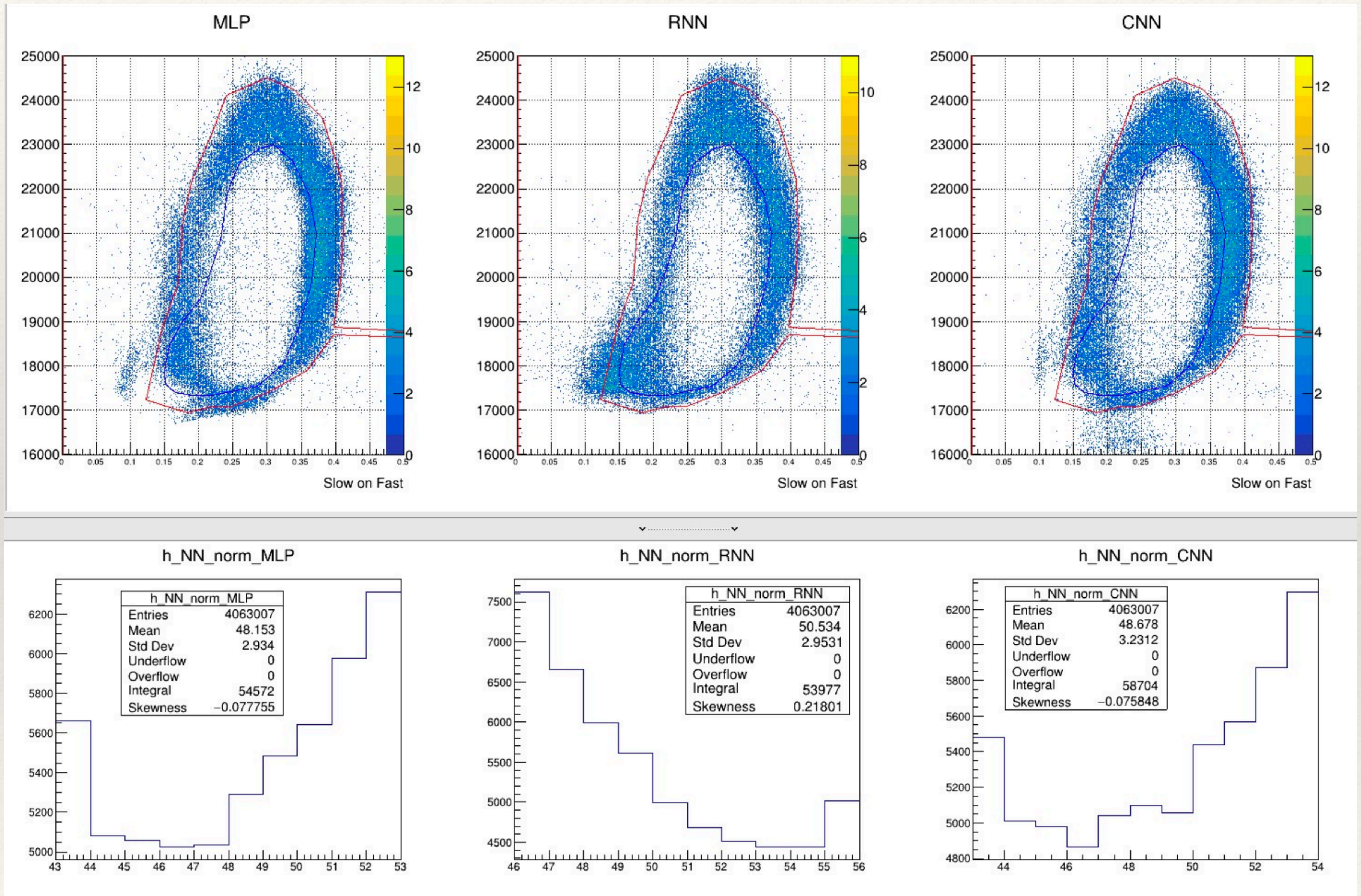
Mislabel probability



Neural Network answer to is γ or **n** ?

Machine Learning on signals

How networks extrapolate on data out of the cuts used for training ?



☛ We are working on the qualification of those sub-events using γ spectra

Machine Learning on signals

We have moved to simulations to check for strengths/weaknesses of the different NN

↪ labels on γ or **n** are 100 % sure !

Function used to generate signals

s(t) = **A** [exp(-t/**td1**) - exp(-t/**tr**) + **R***(exp(-t/**td2**) - exp(-t/**tr**)] si t > **T0**

Study 1 : sensibility to **T0**

Training done with gaussian distribution for T0, $\boldsymbol{\sigma} = 2$

Test done with gaussian distribution for T0, $\boldsymbol{\sigma} = 20$

Study 2 : using NN to tag pileup signals

ΔT between two signals, random distribution

Almost same networks, just more categories, more outputs

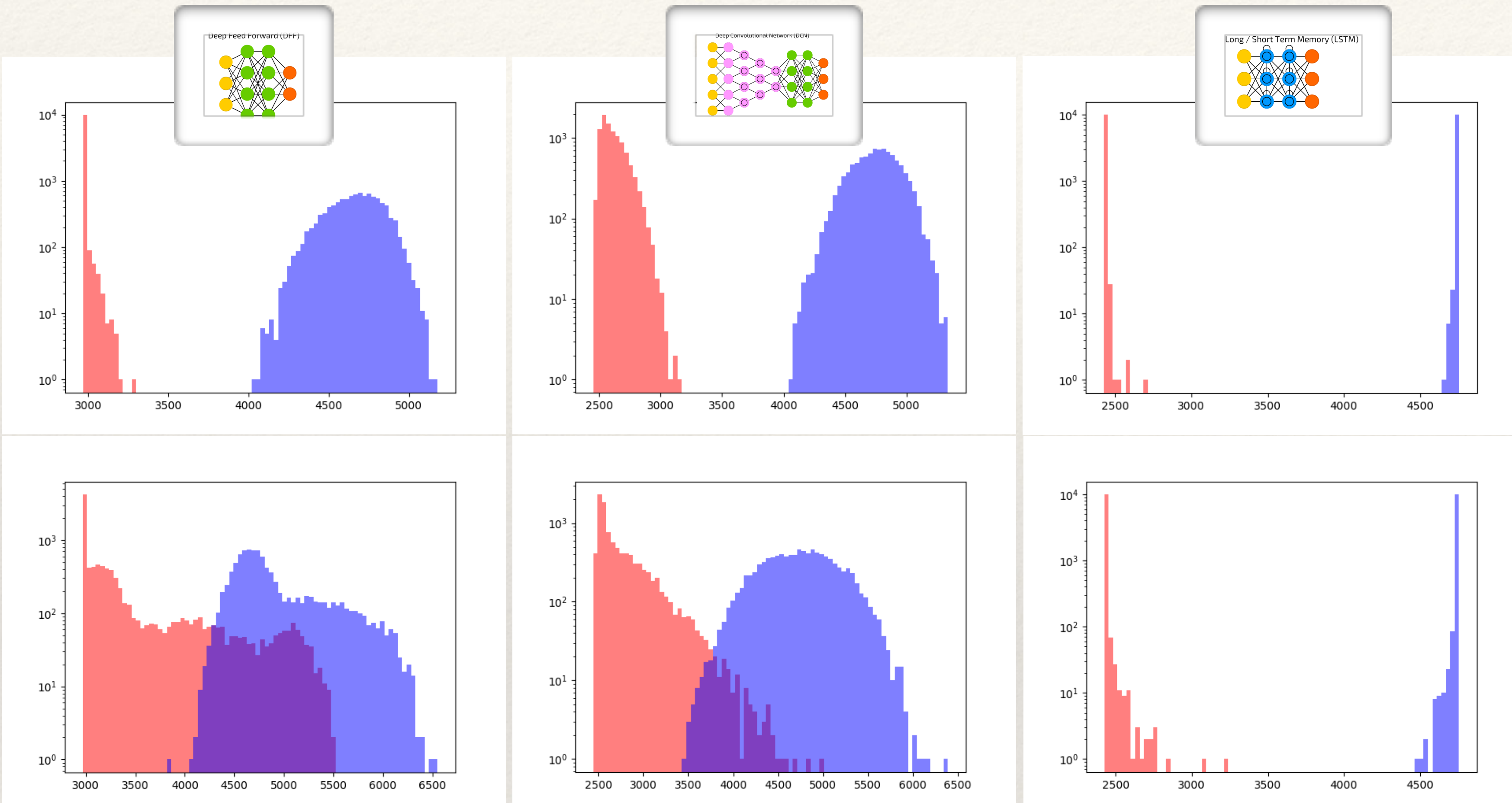
Network type	Structure	Activation functions	Number of trainable parameters
MLP	3 Dense layers (75 x 10 x 4 x 2)	Relu x ReLu x SoftMax	814
LSTM	75 x 1 LSTM layer (50 hidden units) x 1 Dense layer (50 x 2)	SoftMax	10 502
Convolution	75 x 3 (Conv1D+Max_Pooling) layers x 2 Dense layers (100 x 20 x 2)	ReLu x ReLu x ReLu x ReLu x SoftMax	7 042

Machine Learning on signals

Study 1 : sensibility to **T0**

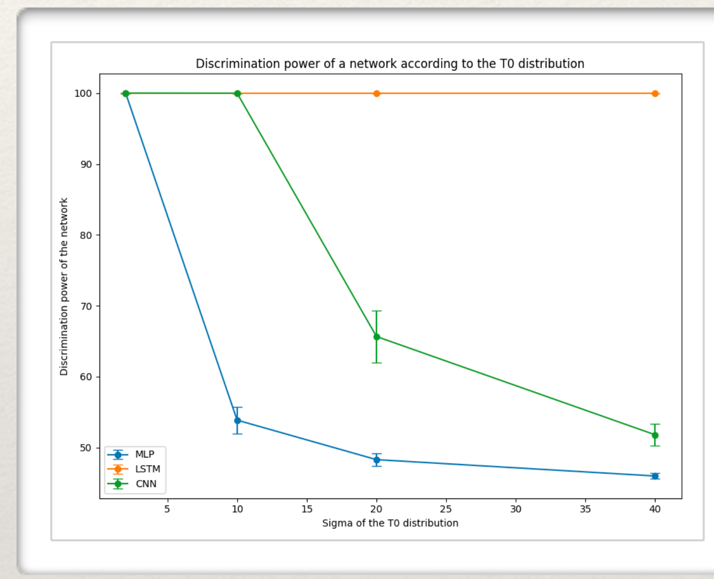
$\sigma = 2$

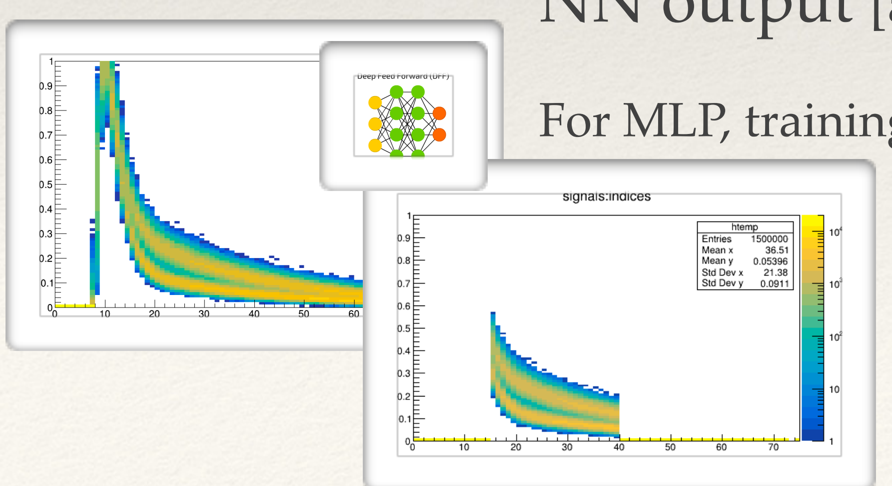
$\sigma = 20$



NN output [as coded in the data flow]

LSTM the most robuste !





For MLP, training with full signals, test with partial signals

☛ the network works fine with partial signals !

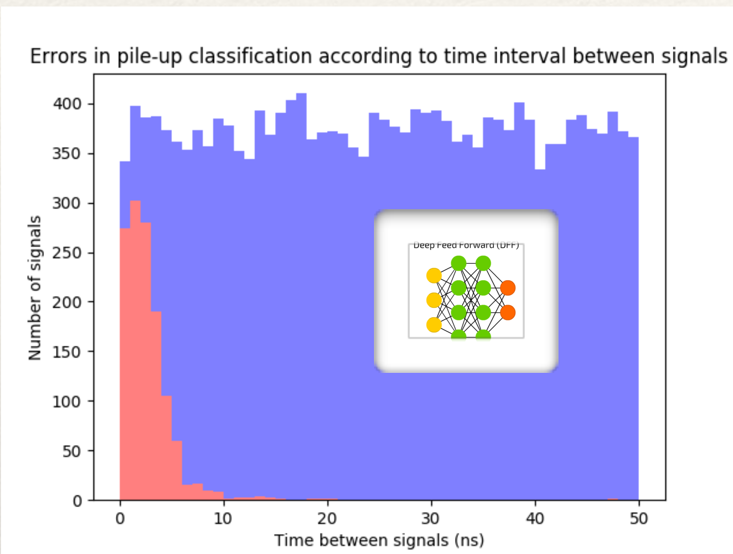
☛ Important to have well 'calibrated'* signals for that kind of NN

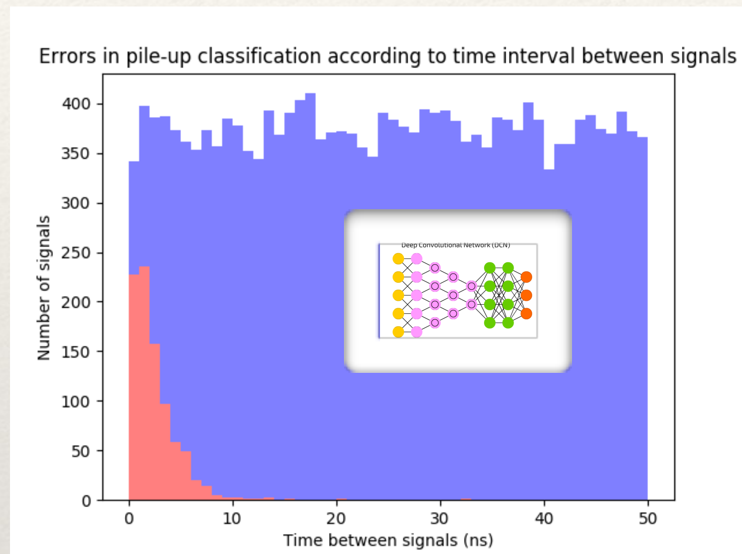
* Feature extraction in machine learning language

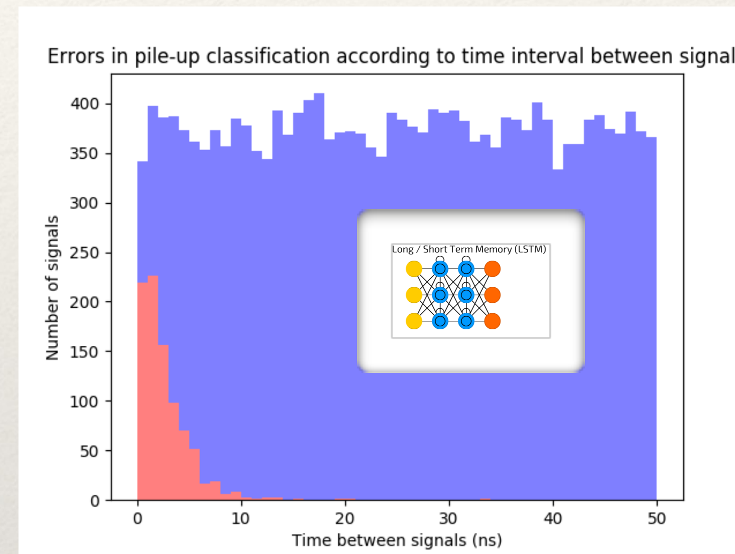
Machine Learning on signals

Study 2: Pileup identification

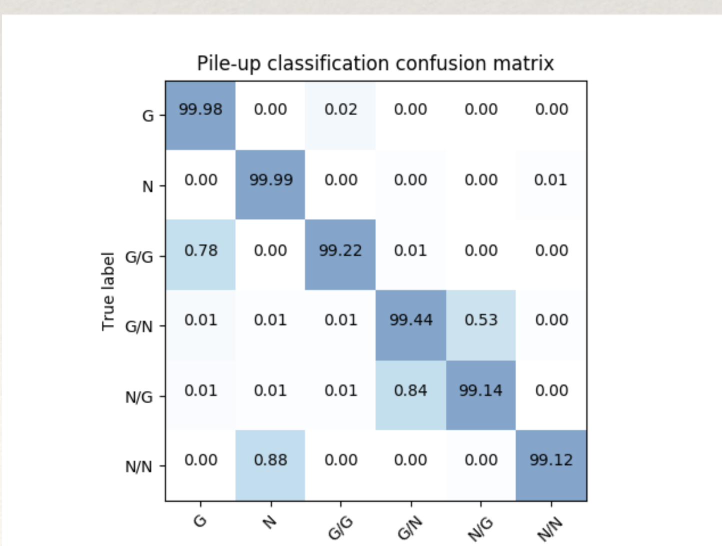
Error as fonction of the time between signals

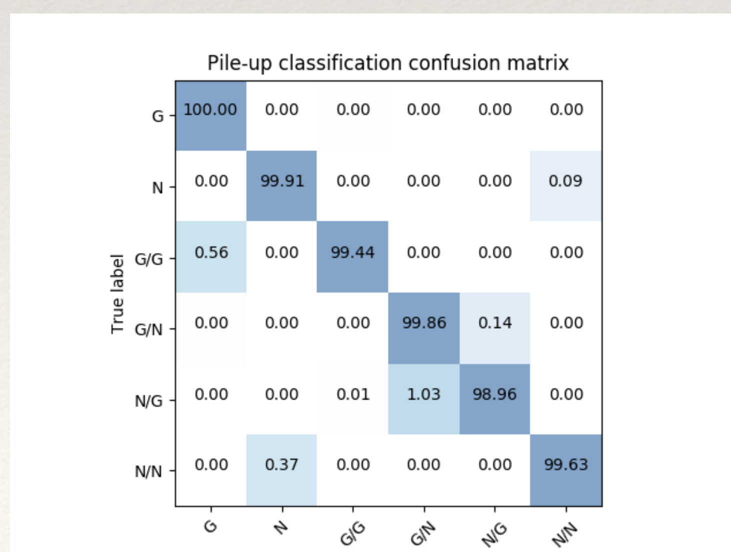


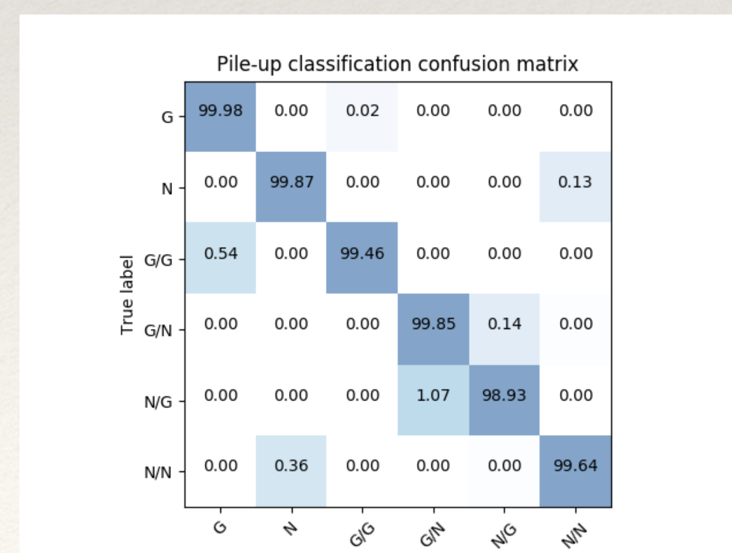




Confusion matrix



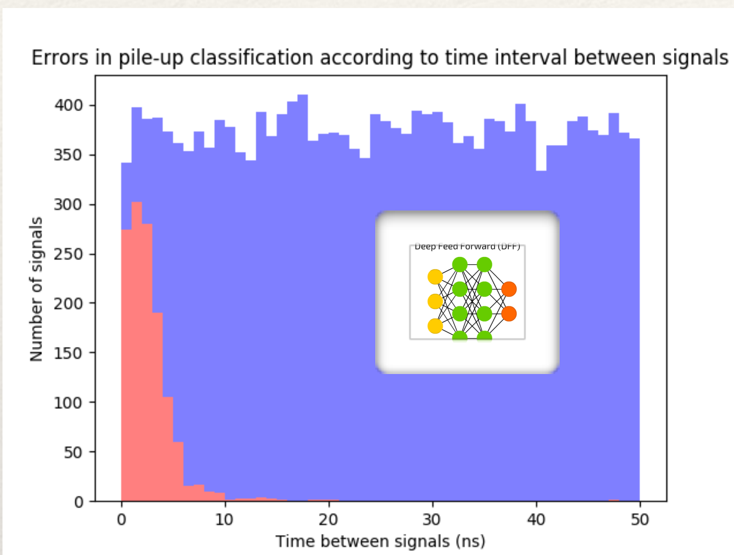


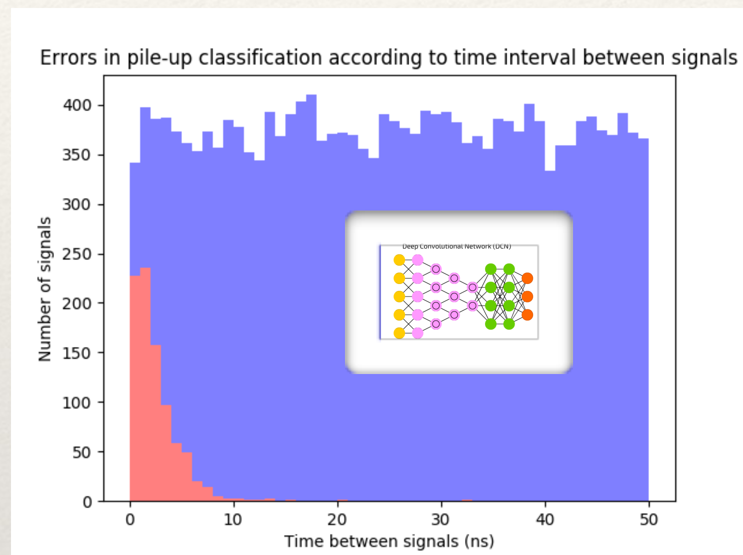


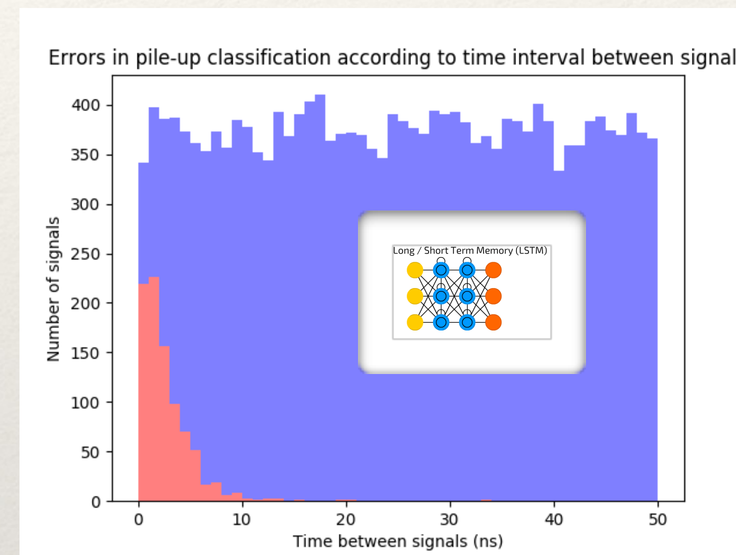
Machine Learning on signals

Study 2: Pileup identification

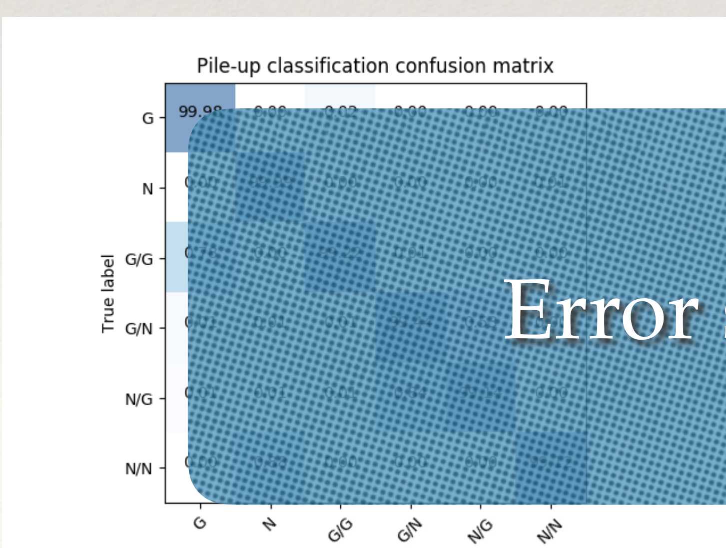
Error as fonction of the time between signals

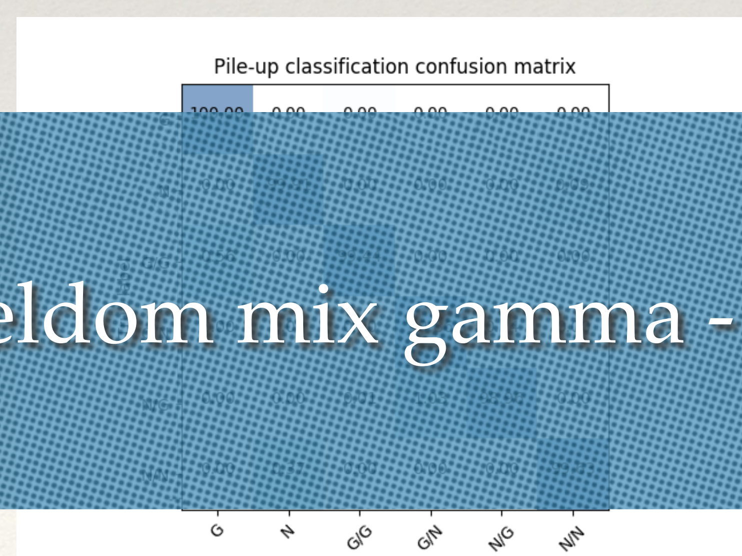


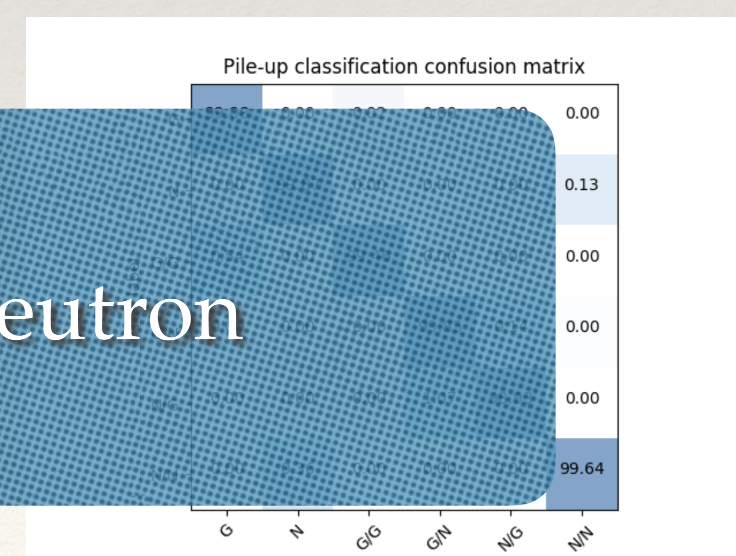




Confusion matrix

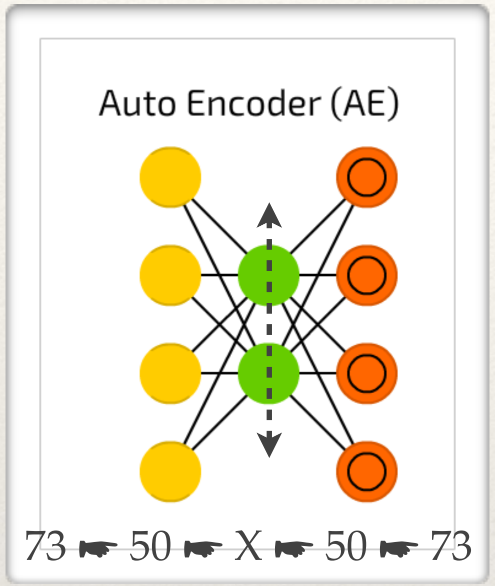






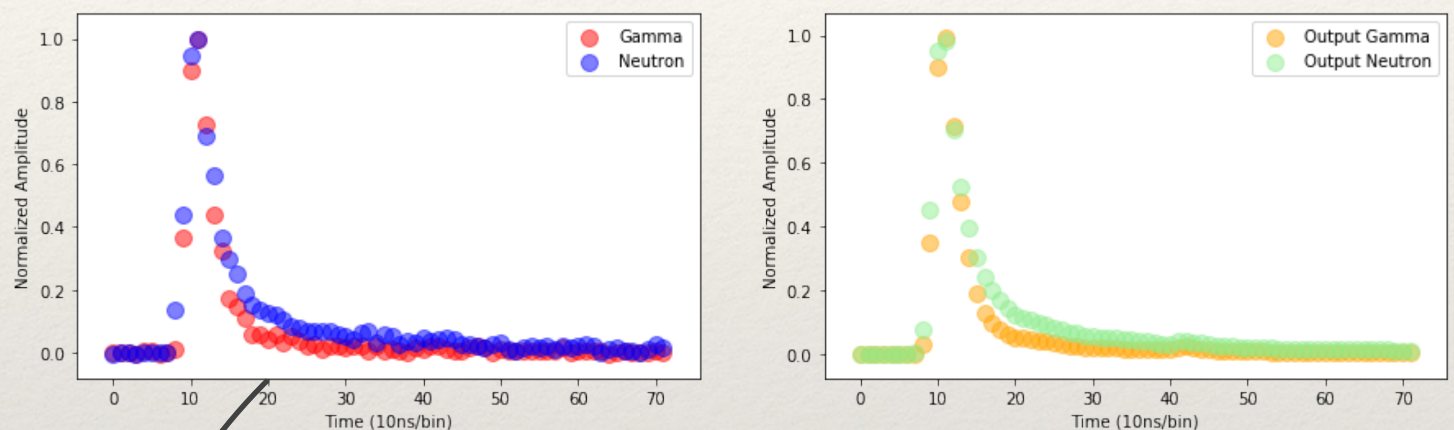
Error seldom mix gamma - neutron

Machine Learning on signals



Auto-encoder, unsupervised learning ☛ self learning !

➥ avoid problems of having labelled training data …



Objectives :

Denoising

Data reduction !

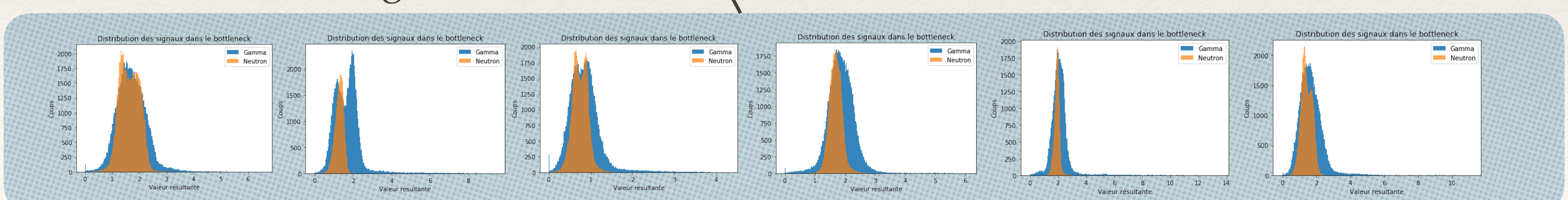
We found @ least 4 neurones needed in the bottleneck,
Is it link to

$s(t) = A\,[\exp(-t/td1) - \exp(-t/tr) + R^{*}(\exp(-t/td2) - \exp(-t/tr)]$?

Identification of anomalies ?

Simulation of signals ?

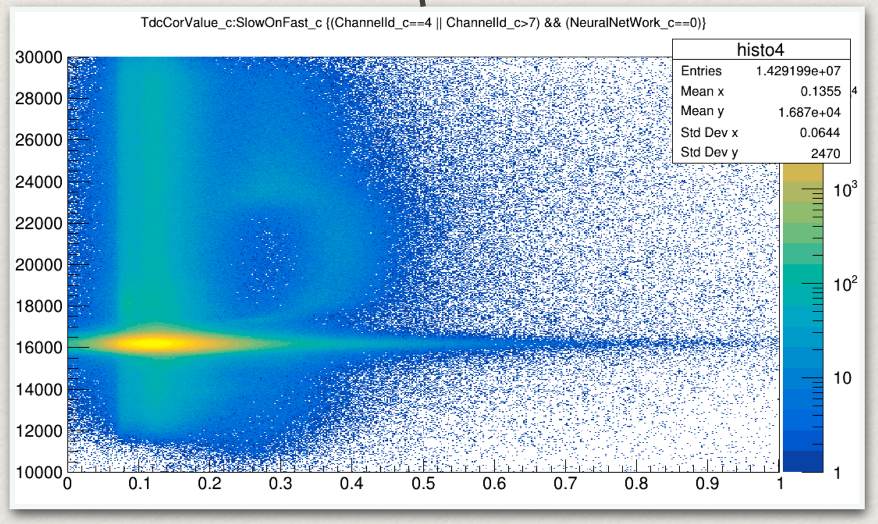
Distribution for 6 central neurons



Conclusions / perspectives

We have studied several Neural Network architectures on Traces (PSD/NEDA)

- ☛ **feature extraction** (calibration) could be a **sensible issue**
- ☛ Dependence on the training data set not fully studied
- ☛ **LSTM** networks **robust** to **T0 misalignment**



- ☛ Autoencoder (no training) studies for
  - ➥ denoising … effects to be quantified
  - ➥ Data reduction at hand ?

- ☛ Future directions are :
  - ➥ **denoising** … effects to be quantified
  - ➥ ML to identify '**anomalies**' in signals
  - ➥ Other applications, for instance clusters for tracking

On NEDA/AGATA/GRETINA