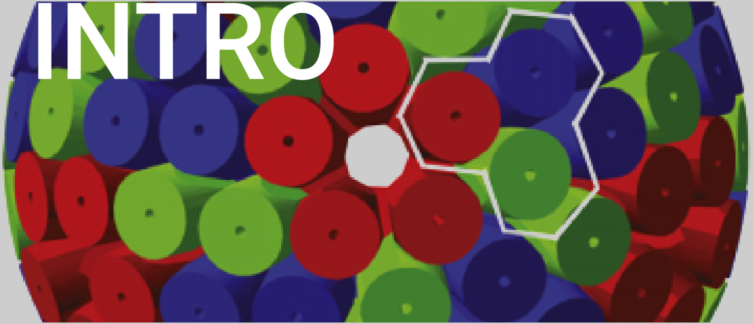
Machine Learning to count interactions in AGATA

Me: Danylo Kovalenko
Taras Shevchenko National University of Kiyv, Kyiv, Ukraine

Supervisor: Joa Ljungvall.
IJCLab, Orsay, France

Jan 17 2023

INTRO



There is a new age in nuclear structure research, and it requires the new generation of detector systems with unprecendent level of sensitivity and count-rate capabilities.

In AGATA combined:

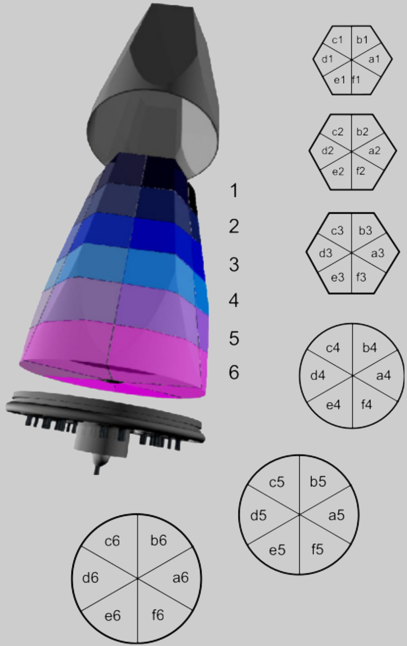
1. Large solid angle
   -> Increase efficency $(\varepsilon = \frac{N_{detected}}{N_{emitted}})$

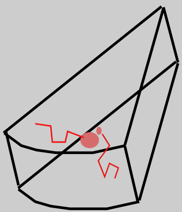
2. Electrically Segmented HPGe detectors and Gamma-ray “tracking”
   -> Peak-to-total (Escape supression)

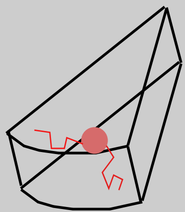
Gamma-ray “tracking” algorithm require:

- Position of interactions -> **PSA**
- Time of interactions
- Energy deposition

Pulse shape analysis (PSA) can tell more precisely about position of interaction within segment.
But it has the **flaw** - it can not separate two close interactions and sees one instead.







What the challenge is?

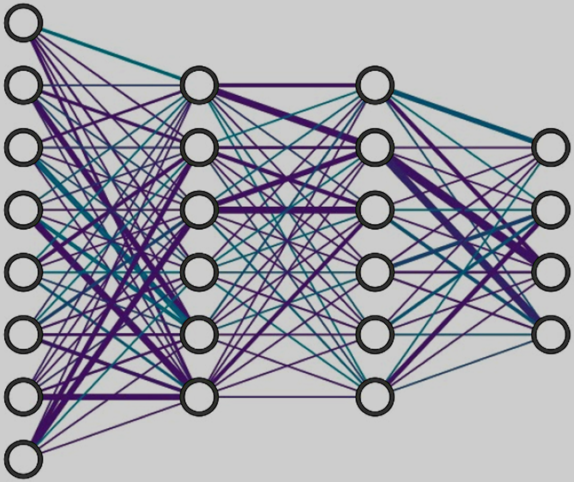
Prediction of number of interactions within segment:

Simulate data with GEANT4

Build and train neural network

Use it in case of real data

Short explanation of deep learning



$$f(\vec{X}, \mathbf{\Omega}) = \vec{\mathbf{Y}}$$

$\vec{X} - input$

$\vec{Y} - output$

The first approach

$$i \qquad \vec{X}_i = [Edep_i, \quad crystal_i, \quad Segment_i] \qquad \vec{Y}_i = Y_i$$

	edep	crystal	slice_sect	num_of_int
1	61.454	116	5	1

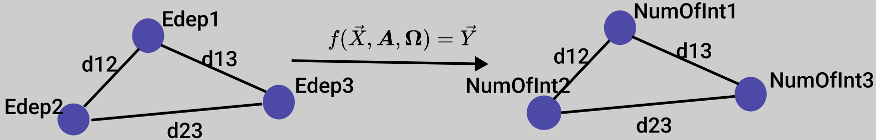
The second approach

```
Global_array[0][1]

array([[0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       ...,
       [0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.]])

Global_array[0][1].shape

(165, 56)
```

$$f(\vec{X}, \mathbf{\Omega}) = \vec{\mathbf{Y}}$$

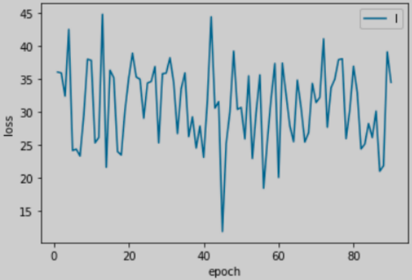
```
Global_array[1][1]

array([[0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       ...,
       [0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.]])

Global_array[1][1].shape

(165, 56)
```

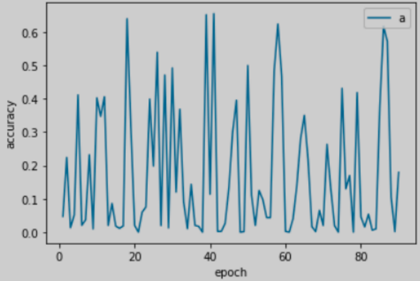
The third approach



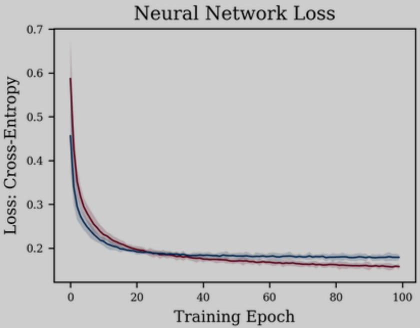
$$\vec{X} = [Edep_i] \qquad \mathbf{A} = [\mathbf{Distance_{ij}}]$$

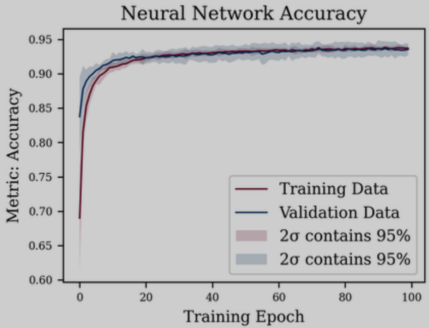
Impossibility to train on small set of data





Our result

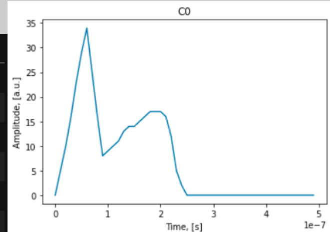


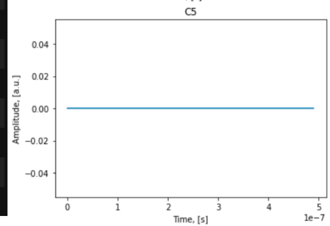


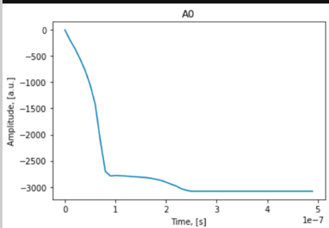
How it should look

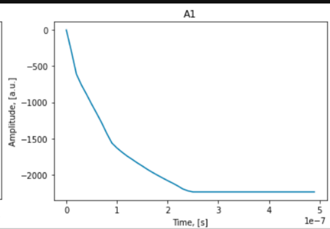
The fourth approach

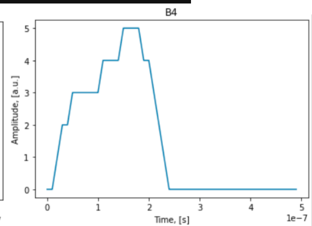
	time	ampl	A0	A1	A2	A3	A4	A5	B0	B1	...	E2	E3	E4	E5	F0	F1	F2	F3	F4	F5
0	0.000000e+00	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
1	1.000000e-08	235	-192	-296	176	12	1	0	34	11	...	3	1	0	0	-130	37	18	6	1	0
2	2.000000e-08	487	-359	-612	323	25	2	0	71	26	...	6	2	0	0	-248	64	39	12	2	0
3	3.000000e-08	742	-553	-763	378	33	4	0	100	41	...	9	3	0	0	-352	86	54	15	2	0
4	4.000000e-08	1032	-774	-891	413	39	5	1	115	51	...	10	4	1	0	-445	103	71	19	3	0
5	5.000000e-08	1364	-1048	-1026	440	46	6	1	137	60	...	11	5	1	0	-518	119	80	22	3	0
6	6.000000e-08	1759	-1421	-1154	447	51	7	1	157	67	...	13	5	1	0	-432	129	85	24	3	1
7	7.000000e-08	2259	-2106	-1286	435	53	7	1	144	73	...	10	4	1	0	-121	116	77	22	3	1
8	8.000000e-08	2919	-2701	-1432	407	51	6	1	98	66	...	6	3	0	0	140	93	66	19	3	0
9	9.000000e-08	3394	-2789	-1563	381	52	6	1	65	64	...	4	2	0	0	63	74	63	19	3	0
10	1.000000e-07	3433	-2780	-1632	371	55	7	1	71	67	...	5	3	1	0	69	77	67	20	3	0
11	1.100000e-07	3478	-2784	-1690	348	58	7	1	79	71	...	5	3	1	0	74	80	69	21	3	1
12	1.200000e-07	3527	-2791	-1743	327	59	7	1	85	73	...	6	3	1	0	77	84	72	22	4	1











What is already done.

Joa and I simulated the calibration source - Co60
inside half sphere AGATA.

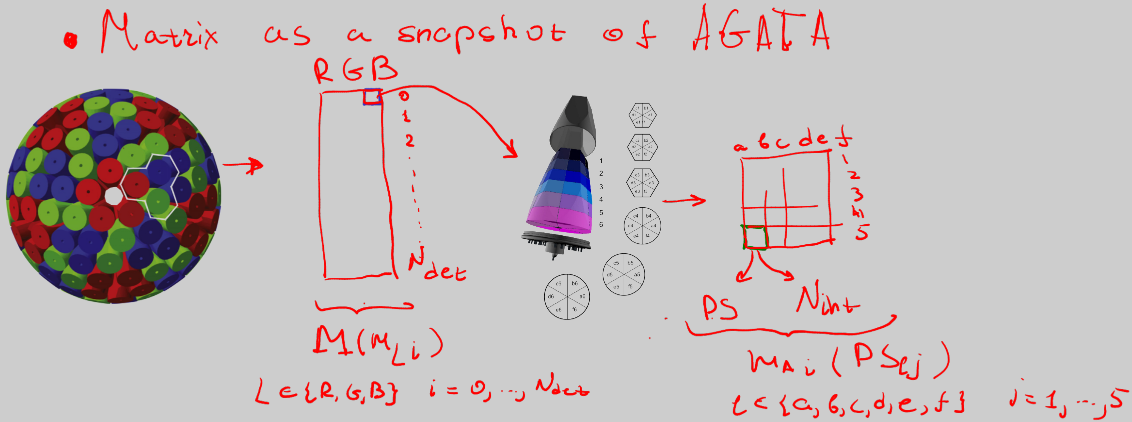
The output contain all gamma-ray interactions
inside detectors.

Than Joa made the pulse shapes.

The next step is to combine the pulse shapes with
GEANT4 output and create the proper data set (ps <-> Nint).

Data set creation

The idea is the same as for second approach



Data set creation

- Training data : $M(P.S) \Leftrightarrow M(N_{int})$

For different slice of time we will have:

Δt_1 : $M(PS_1) \Leftrightarrow M(N_{int\,1})$

⋮

Δt_k : $M(PS_k) \Leftrightarrow M(N_{int_k})$

$$f(M(PS_k), \vec{\Omega}) = M(N_{int})$$

- How to present the pulse shape PS_{ej}

* $PS_{ej} = [[A_0, t_0], [A_1, t_1], \ldots, [A_{13}, t_{13}]]$

* $PS_{ej} = [\vec{A}, \vec{t}]$

The complexity of fourth approach

1. Each crystal has isolated data storage - the only connection is time.
2. Size of vectors with whole pulse shapes much bigger than sizes of previous input data sets.

That means that data set creation and NN training computation needs much more time than before. Regardless, with PS we can use all possible information from detector.

In compare with previous attempts I can use more productive equipment:

- Intel® Core™ i7-10870H × 16 for simulation.
- GTX 1650 for NN training.

So now I can handle bigger data sets in limited time range.

Thanks for attention