

Deep Learning approach to Cellular Automaton CALO reconstruction in LHCb

Núria Valls

DS4DS - La Salle URL Barcelona

On behalf of the LHCb Real Time Analysis Project

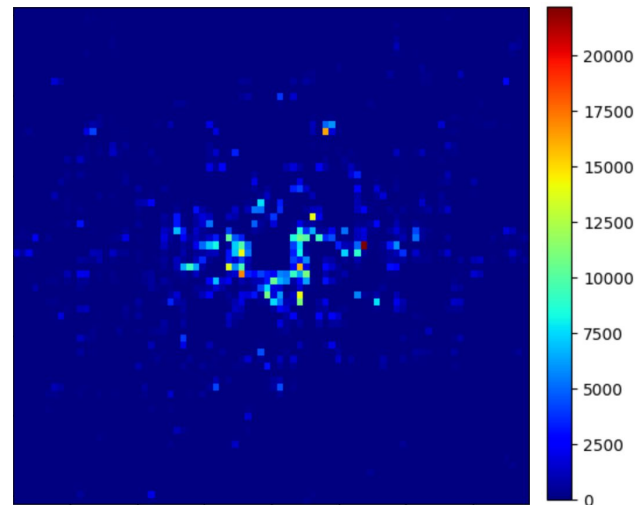
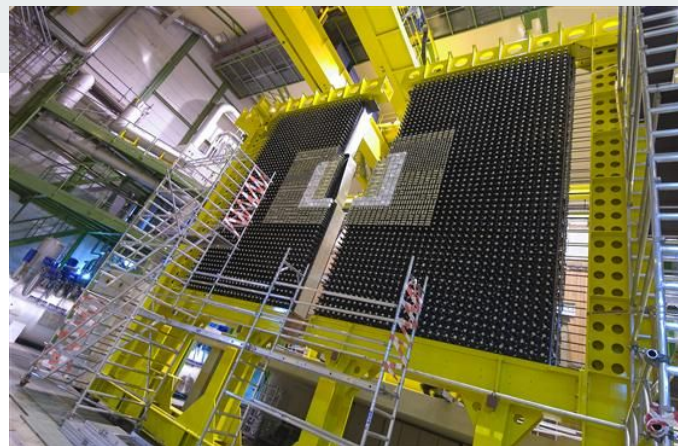
25th International Conference on Computing in High-Energy and Nuclear Physics

18th May 2021



Introduction: ECAL

- **2D grid of Shashlik** modules → absorber plates and scintillator tiles (active material)
- Three regions:
 - a. **Inner** → 4×4 cm² cell size
 - b. **Middle** → 6×6 cm² cell size
 - c. **Outer** → 12×12 cm² cell size
- 2D images as **raw data**: 12 bit precision
- **Cluster**: group of cells that contain the energy of a particle (3×3)



Raw data of an example event from LHCb simulation

The current approach

Peak detector

Cluster reconstructor

Overlap solver

Cluster corrections

Find local maxima

Tag energy cells around maxima

Fraction energy values for overlapping cells

Can be done with reconstructed data

One iteration for all cells at least

One iteration through all the cells at least

Needs more than local information

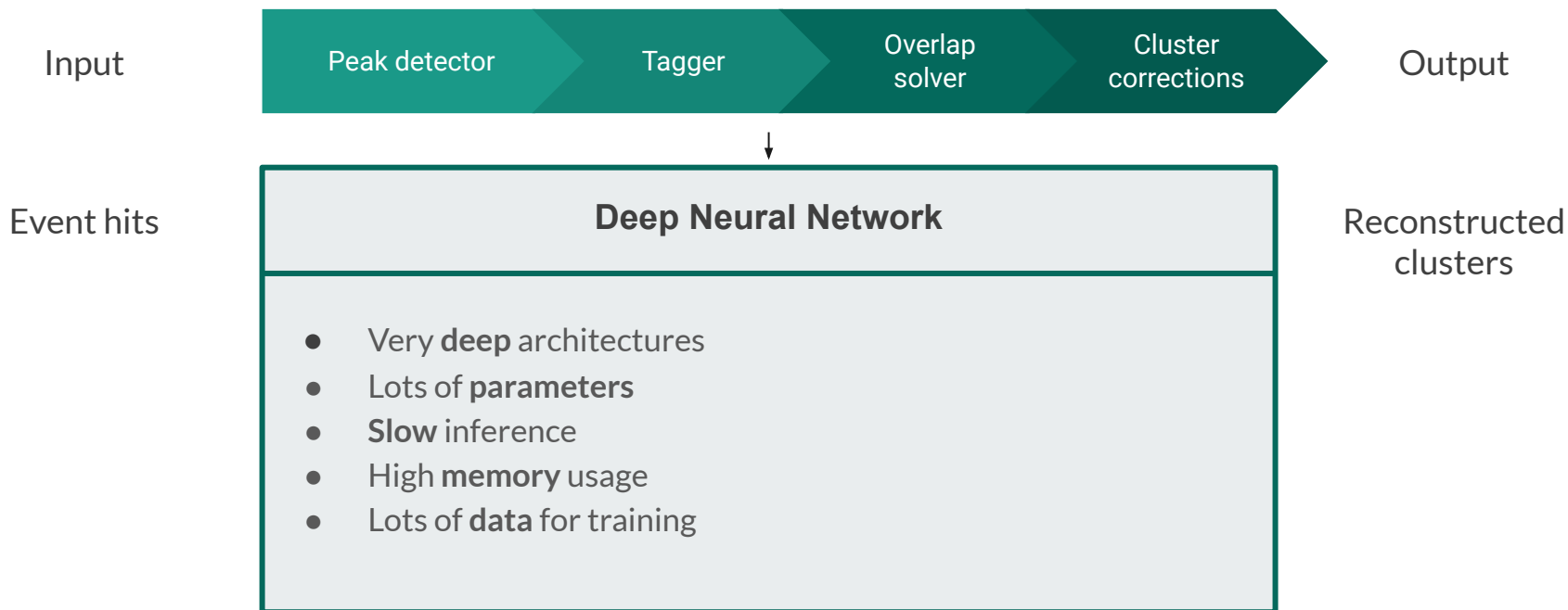
Is “Cellular Automata (CA)-like”

Is a CA

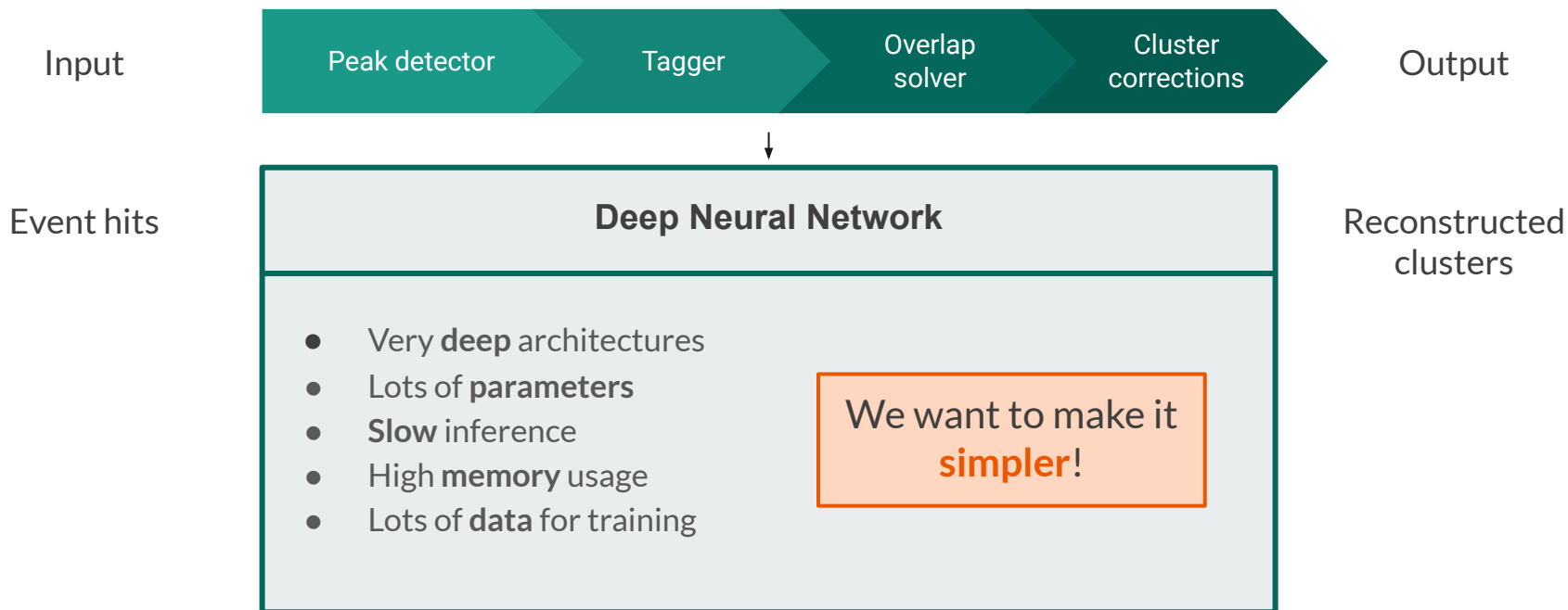
Calorimeter reconstruction takes 25% of High Level Trigger 2 sequence time

[LHCb Public Results](#)

The current approach in Deep Learning



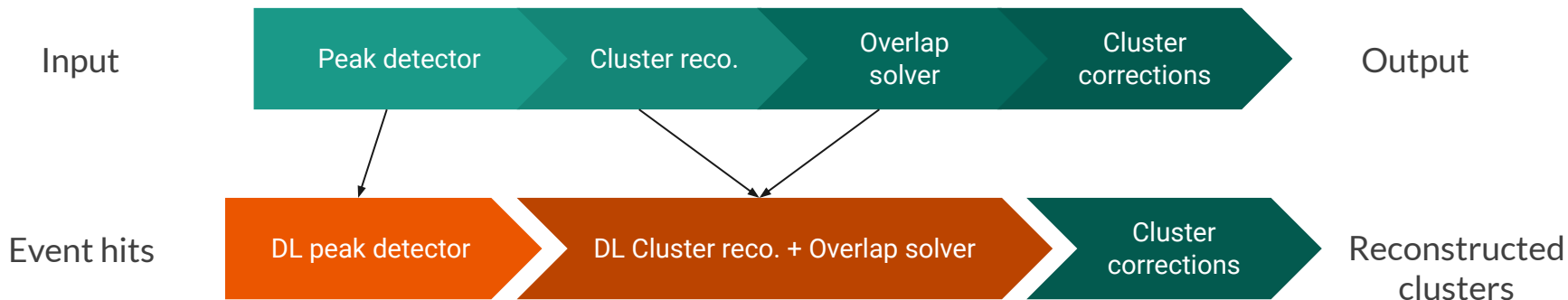
The current approach in Deep Learning



Our approach in Deep Learning

Base idea → The rules of a Cellular Automata can be learned by a deep convolutional architecture

Gilpin, W. (2019). Cellular automata as convolutional neural networks. *Physical Review E*, 100(3), 032402



Peak finder

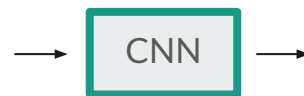
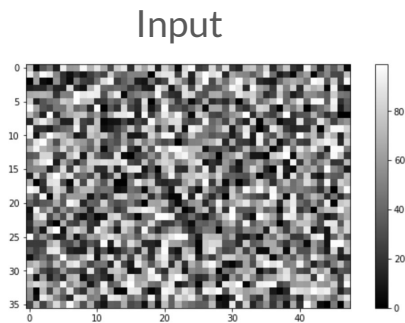
DL peak detector

DL Cluster reco. + Overlap solver

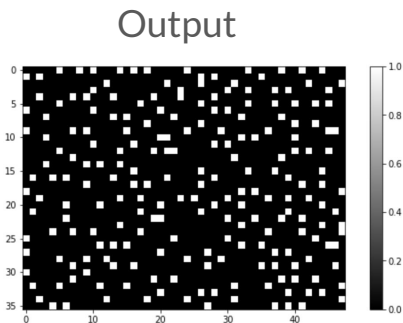
Cluster corrections

The ruleset defines **comparisons**: Identify local maxima **independently of the numerical value scale**.

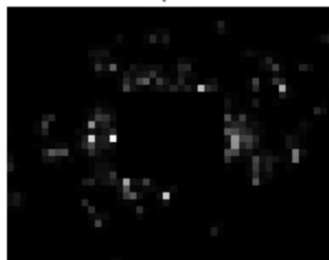
Training:
1k images of
random values
(0-99)



2D Conv. + Dense layers
1272 parameters



Testing:
with ECAL
simulation data
0.9973 acc



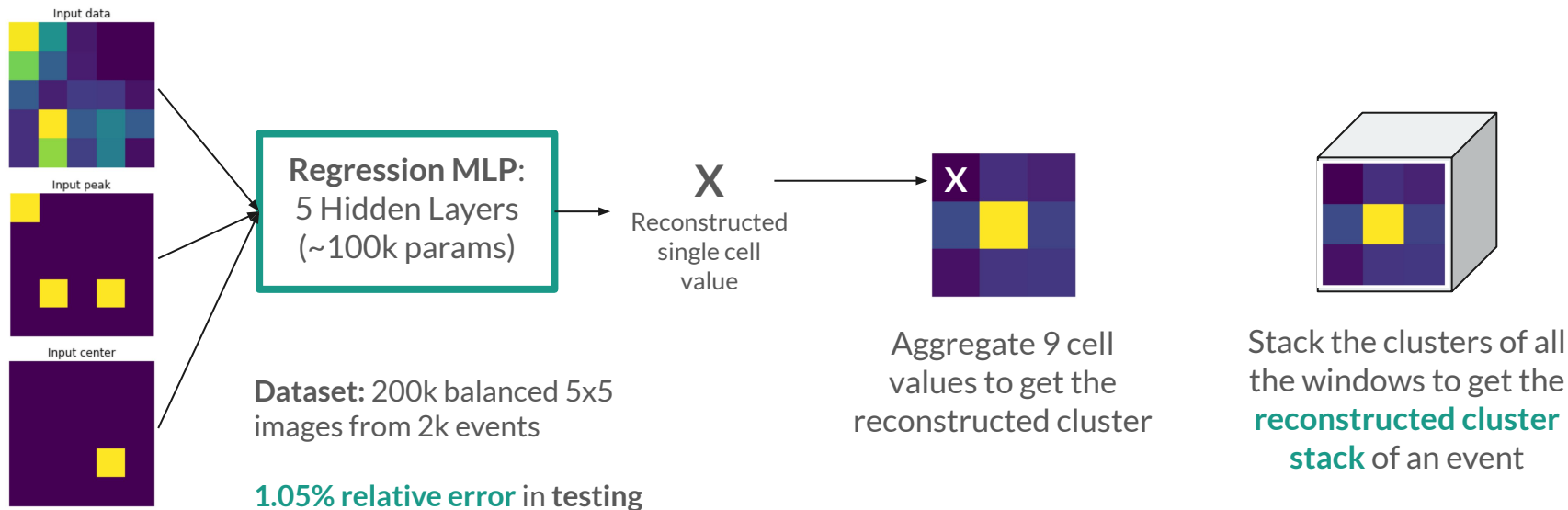
Cluster reco + overlap

DL peak detector

DL Cluster reco. + Overlap solver

Cluster corrections

The ruleset defines **fractions**: Need to model **non-linearities** → 2D Conv. Layers don't work
We need a **MLP** to be the kernel of the convolution



Results

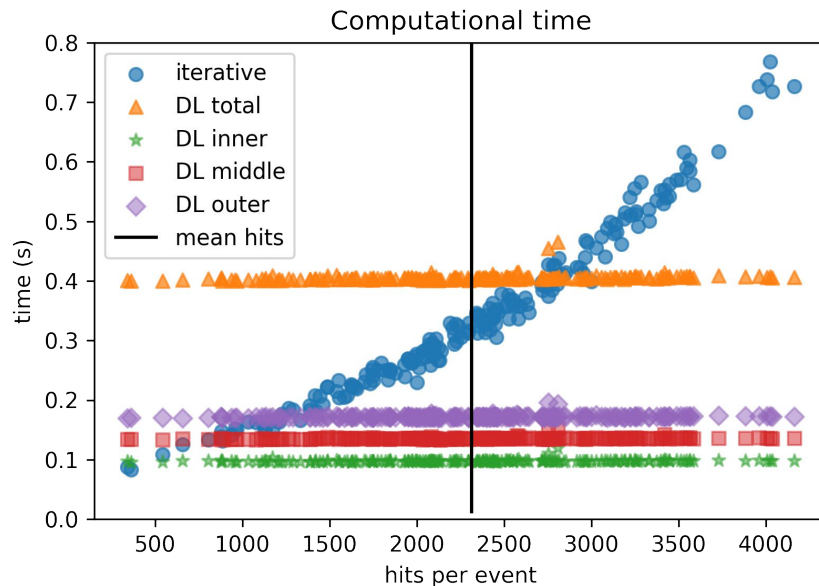
Local comparison between an **iterative method** (same complexity as the current algorithm) and the proposed **DL method**

Relative **difference of reconstructed energy per cluster** with simulated events:
Mean 0.083 ± 0.153 and 0.055 ± 0.104

→ Nearly **constant behavior** with independence of the events complexity

→ Only faster in **25%** of the events

→ If **region-parallelized** it is faster in **89%** of the events



- Coding: Python 3.8
- Networks: Tensorflow 2.3.0
- Test environment: Intel CPU, 4 cores

Conclusions

- We achieved to **reproduce** the steps of the current reconstruction in a **linear complexity algorithm** that once trained, the **reconstruction cost is independent from the event complexity**
- The **volume of data** needed for the network to **generalize knowledge** is much less than training with the whole calo images and can be **artificially generated** in some cases to avoid depending on simulation data
- Each step can be tuned and retrained **adding new features** (e.g. train directly with montecarlo data)
- Region-independence and neural network architectures could **benefit from parallel architectures**



Thank you!

Any questions?

Núria Valls

DS4DS - La Salle URL Barcelona

On behalf of the LHCb Real Time Analysis Project

25th International Conference on Computing in High-Energy and Nuclear Physics

18th May 2021





Backup

Núria Valls

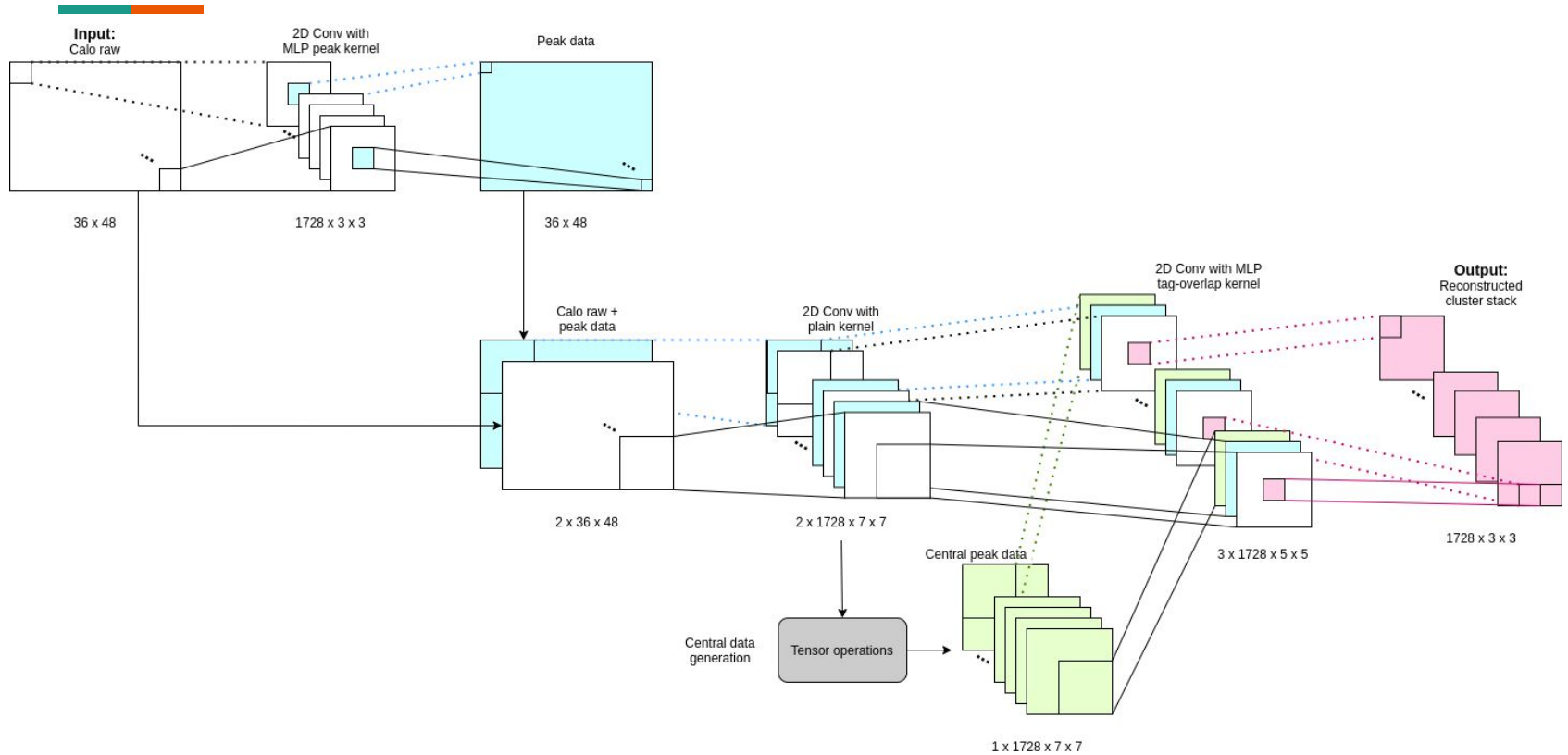
DS4DS - La Salle URL Barcelona

On behalf of the LHCb Real Time Analysis Project

25th International Conference on Computing in High-Energy and Nuclear Physics

18th May 2021

Overall structure



Formulation of the peak finder

CA-like formulation:

- Output states: 2
- Neighbourhood: 8 cells
- Ruleset:

$$f(c_{i,j}^{t+1}) = \begin{cases} 1, & \text{if } c_{i,j}^t > c_{i+M,j+N}^t \text{ for } M \in \{-1, 0, 1\}, N \in \{-1, 0, 1\} \\ 0, & \text{otherwise} \end{cases}$$

Where the energy value of each cell is defined as $c_{i,j}^t$ and the case $M=0, N=0$ is excluded

→ Identify local maxima **independently of the numerical value scale:**

- Generate data for the training where the **ruleset is uniformly represented**
- Dataset: **1k** images with random values from **0 to 99**
- Network architecture: 2D Conv. + Dense layers → **1272 parameters** → **0.9973 acc**

Formulation of the cluster reco + overlap



CA-like formulation:

- Output states: **4096** (2^{12}) \rightarrow ECAL precision. This is “forcing” the network to be regressive

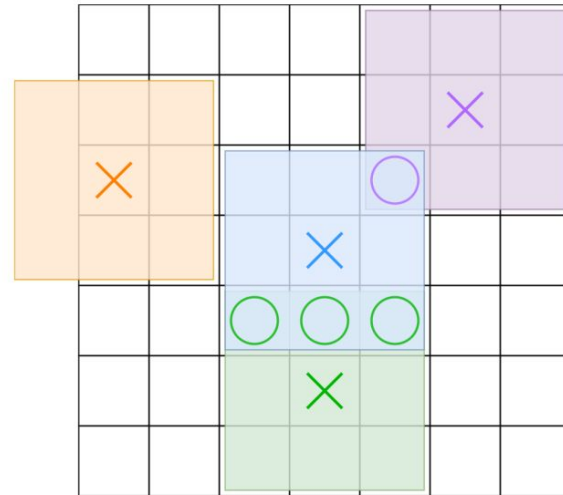
Formulation of the cluster reco + overlap

CA-like formulation:

- Output states: **4096** (2^{12})
- Neighbourhood: 24 cells

1. Tag cells around the peaks (X)
2. Fraction the energy of overlapping cells (O)

→ We need a **7x7 window** around each peak that outputs a 3x3 image (reconstructed cluster)



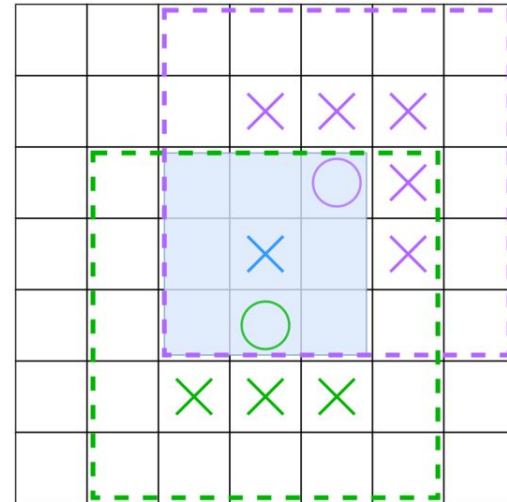
Formulation of the cluster reco + overlap

CA-like formulation:

- Output states: **4096** (2^{12})
- Neighbourhood: **24** cells (5x5 window)

1. Tag cells around the peaks (X)
2. Fraction the energy of overlapping cells (O)

→ We need a **5x5 window** around the central peak that **generates the energy value of the central cell in the reconstructed cluster**



Formulation of the cluster reco + overlap

CA-like formulation:

- Output states: **4096** (2^{12})
- Neighbourhood: **24** cells
- Ruleset:

$$f(c_{i,j}^{t+1}) = \begin{cases} c_{i,j}^t, & \text{if is central seed} \\ \frac{c_{i,j}^t C_0}{\sum_{k=0}^K C_k}, & \text{for } K = \text{num_seeds} \text{ if } c_{i,j}^t > 0 \text{ and is not central seed} \\ 0, & \text{otherwise} \end{cases}$$

The rules to learn become **non-linear**:

- 2D convolutional layer: kernel multiplied through the image
→ The kernel **cannot model non-linearities**
- Dense layer: can model non-linearities but the **neighbouring information is already lost**

→ Train a **MLP** to be the **kernel** of the convolution

Next steps

- Test time performance inside the **LHCb software framework**
 - How can we make an **efficient inference** of the models in **c++**?
- Add **MC information** to the cluster reco+ overlap model
- Adapt the kernel for **irregular geometry** using Graph Neural Networks
- Check if a **parallelized** version of this algorithm could fit in **HLT1** time requirements