Logic Gate Networks for Front-End ML at a Muon Collider

Mila Bileska, Lino Gerlach, Kiley Kennedy, Isobel Ojalvo

Simons Kickoff

February 11, 2026

Petersen et al:
2210.08277, 2411.04732

Introduction to Logic Gate Networks (LGNs)

Principle of Hardware-Aware ML Training/Learning:

- CICADA (on FPGA, quantized weights) → better to train a quantized model than quantize a trained model
- Logic Gate Networks (on ASIC) → better to train the logic circuit directly than translate a NN into logic

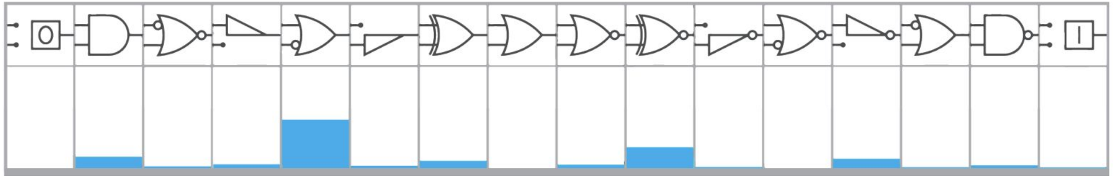
LGN Approach:

- Challenge: how to build and train network of discrete operators that are not inherently differentiable (e.g., no gradient-descent?)
- Solution: relax operators, train, then discretize

Map of Binary to Continuous Logic

ID	Operator	real-valued	00	01	10	11
0	False	0	0	0	0	0
1	$A \wedge B$	$A \cdot B$	0	0	0	1
2	$\neg(A \Rightarrow B)$	$A - AB$	0	0	1	0
3	A	A	0	0	1	1
4	$\neg(A \Leftarrow B)$	$B - AB$	0	1	0	0
5	B	B	0	1	0	1
6	$A \oplus B$	$A + B - 2AB$	0	1	1	0
7	$A \vee B$	$A + B - AB$	0	1	1	1
8	$\neg(A \vee B)$	$1 - (A + B - AB)$	1	0	0	0
9	$\neg(A \oplus B)$	$1 - (A + B - 2AB)$	1	0	0	1
10	$\neg B$	$1 - B$	1	0	1	0
11	$A \Leftarrow B$	$1 - B + AB$	1	0	1	1
12	$\neg A$	$1 - A$	1	1	0	0
13	$A \Rightarrow B$	$1 - A + AB$	1	1	0	1
14	$\neg(A \wedge B)$	$1 - AB$	1	1	1	0
15	True	1	1	1	1	1

During training, LGN nodes are a combo of logic gates

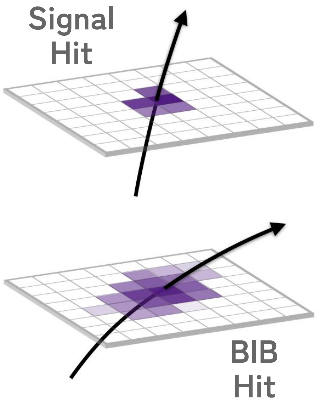


Petersen et al: 2210.08277, 2411.04732

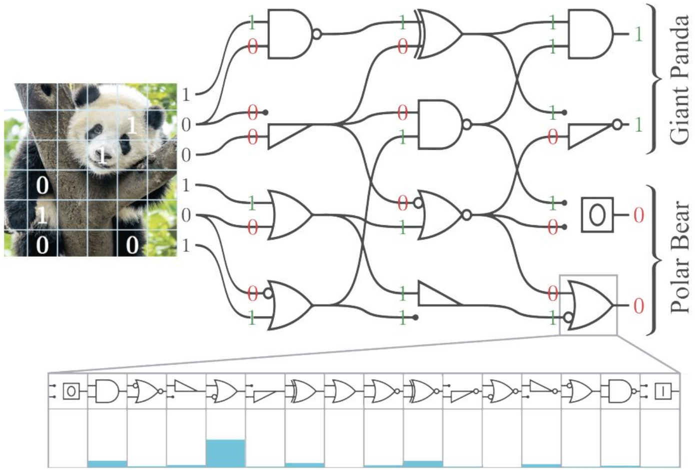
Introduction to Logic Gate Networks (LGNs)

Muon Collider Application: convolutional differentiable LGNs → on-detector intelligence (SmartPixels)

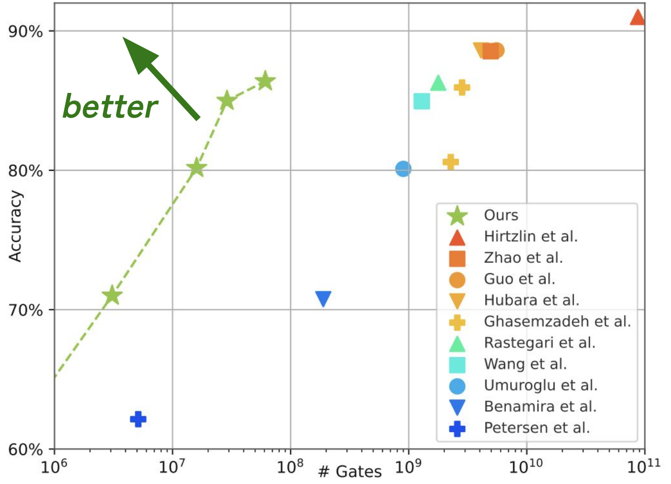
Input



Network (*dev in progress*)



Performance (*from LGN paper*)



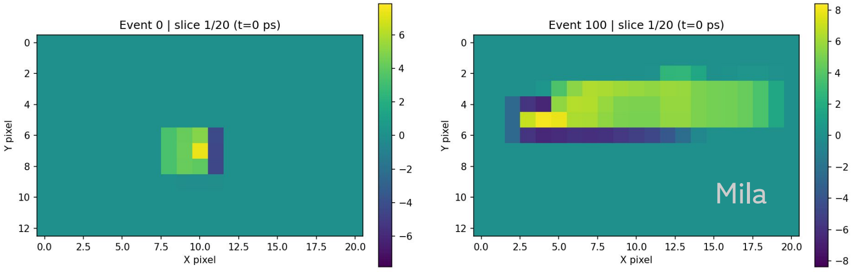
Mila Bileska, Lino Gerlach
SmartPixels reference: [2310.02474](2310.02474)

Current Status

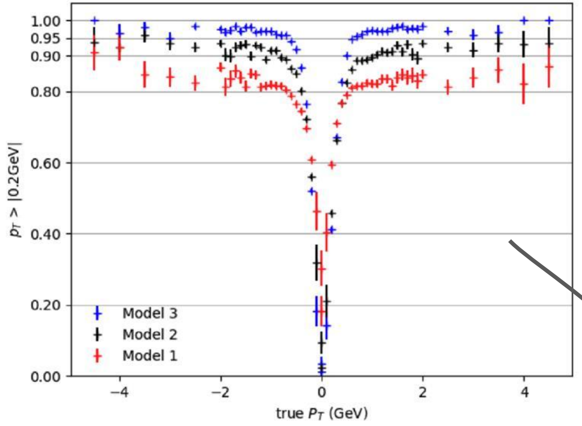
First Objective: Reproduce results with SmartPixels dataset

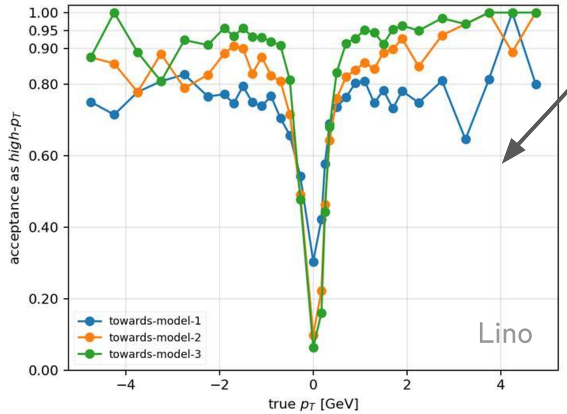
- Train a 3-class classifier (in GeV): (1) $|p_T| > 0.2$ GeV, (2) $|p_T| < 0.2$, negative charge, (3) $|p_T| < 0.2$, positive charge
- Three different models:
  - Model 1 – only takes 2 high level features
  - Model 2 – uses projected y-profile at the last time frame
  - Model 3 – takes a 2d array: y-profile at each time step

Zenodo Dataset (17180303) – September 2025



Acceptance vs. True pT





Ongoing Work + Next Steps

LGN Development (Lino)

- Actively working on an LGN implementation of the network, anticipate first results in O(days)

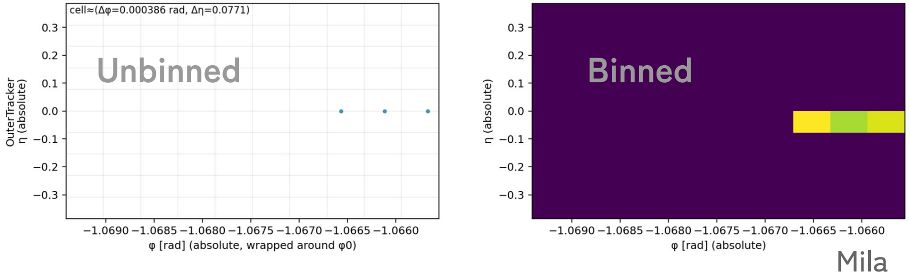
Applications to MAIA (Mila)

- Able to run MAIA simulations + modify digi step (from Angira) → hits still *very* sparse, but need to revisit running over digi hits instead of sim hits
- Training dataset: Signal (muon gun), background (neutrino gun w/ BIB)

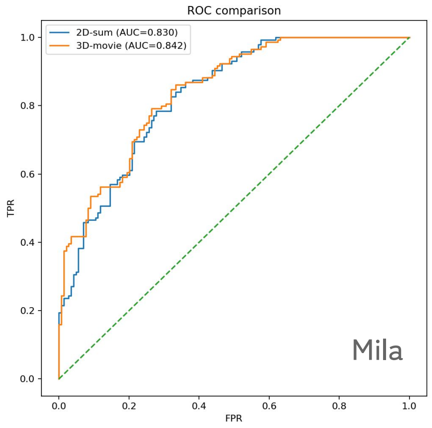
Future Plans + Applications

- Investigating additional funding opportunities, especially private funding
- MuC demonstrator beam monitoring applications
- Open to more suggestions, ideas and collaboration!

Example Sparse MAIA Hit Pattern



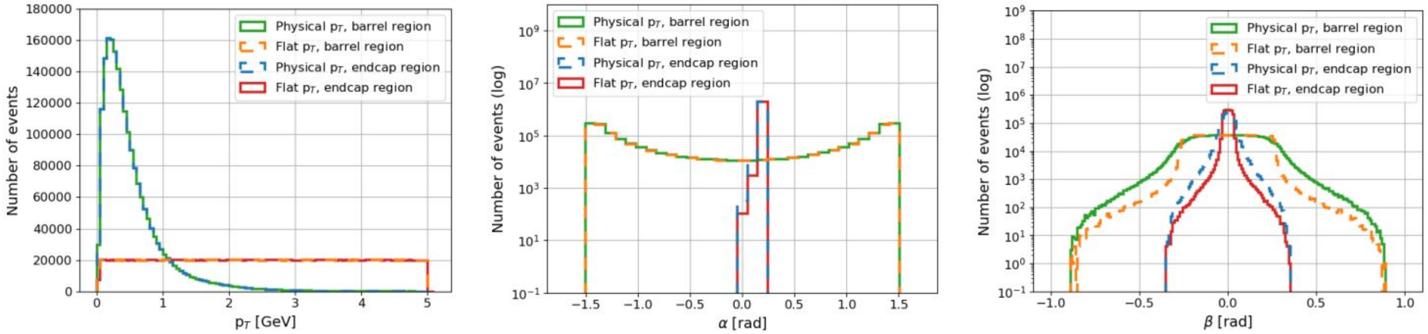
Preliminary ROCs using MAIA Hits



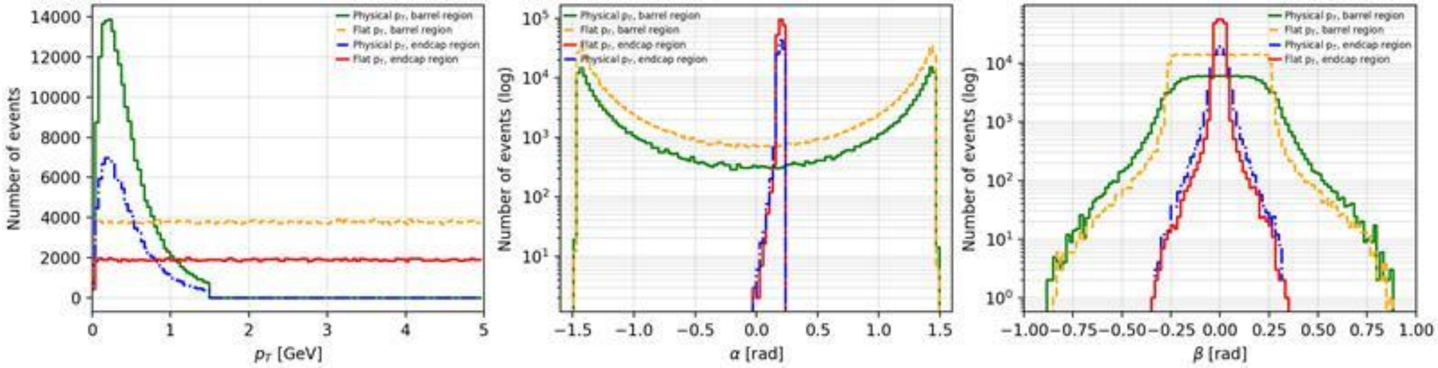
Backup

Comparison: Zenodo/17180303 Plots

Figure 3: Shekar et al., y-size distribution



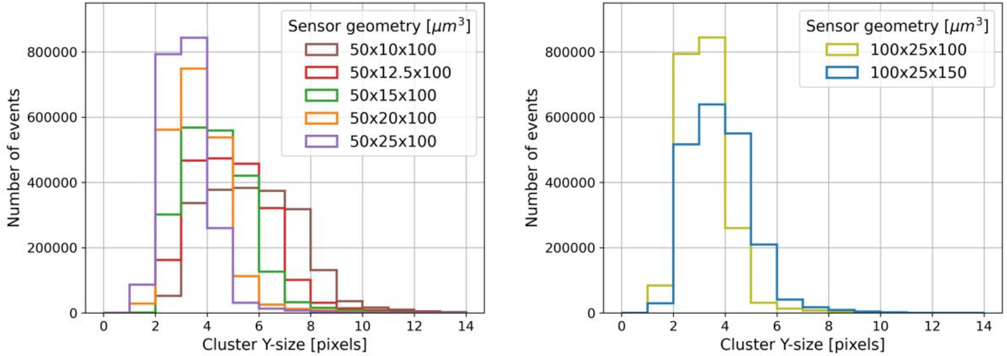
Recreation of y-size distribution, partial dataset



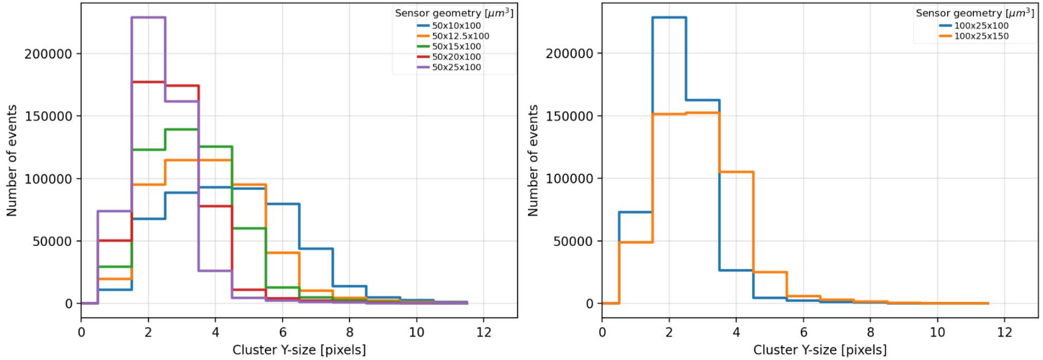
Mila Bileska

Comparison: Y-Size Plots

Figure 3: Shekar et al., y-size distribution



Recreation of y-size distribution, partial dataset



Y-size is computed by taking the final time frame (4000 ps) and counting the number of active Y-rows.