PROCEEDINGS OF SCIENCE

Machine Learning for Real-Time Processing of ATLAS Liquid Argon Calorimeter Signals with FPGAs

Nairit Sur[a,*] **on behalf of the ATLAS Liquid Argon Calorimeter Group**

[a] *Centre de Physique des Particules de Marseille, Centre national de la recherche scientifique, France*

E-mail: nairit.sur@cern.ch

The high luminosity upgrade of the LHC (HL-LHC) will see a massive increase in the instantaneous luminosity leading to up to 200 proton-proton collisions in each bunch crossing (pileup) demanding higher performance from the LHC detectors' electronics and real-time data processing. The ATLAS Liquid Argon (LAr) calorimeter, which measures the energy of particles from LHC collisions, employs dedicated data acquisition electronic boards based on FPGAs, to process large data volumes with low latency. The optimal filtering algorithm currently used for the energy reconstruction has been found to suffer significant performance degradation under high pileup conditions. We show that small recurrent or convolutional neural networks can surpass the performance of the optimal filter. Prototype implementations of the inference code in VHDL indicate that deploying these networks on FPGAs is feasible, with the resulting firmware fitting onto the planned Intel Agilex devices. The complete design can process 384 detector cells per FPGA by integrating parallel instances of the firmware with a latency smaller than 125 ns.

ATL-LARG-PROC-2024-003
09 October 2024

42nd International Conference on High Energy Physics (ICHEP2024)
18-24 July 2024
Prague, Czech Republic

*Speaker

1. Introduction

The ATLAS [1] detector is a multipurpose detector designed to study a wide variety of phenomena produced in high energy particle collisions at the LHC [2], taking place every 25 ns (40 MHz). The high luminosity phase of the LHC (HL-LHC) starting from 2029 will result in an increase of the pileup leading to 140–200 simultaneous proton-proton collisions $\langle\mu\rangle$ observed by ATLAS on average. The ATLAS liquid argon (LAr) calorimeter measures the energy of photons, electrons, and positrons as they pass through the LAr cells causing ionization. The generated pulse in each cell is shaped, and the resulting bi-polar pulse is digitized and used by a linear optimal filtering (OF) algorithm [3] to infer the deposited energy. The pulse shape spanning about 25 bunch crossings (BC) is susceptible to distortions caused by energy deposits from a previous BC in the high-pileup scenario of the HL-LHC. Most of the LAr readout electronics will be updated to accommodate for increased trigger rate and acceptance of events from consecutive BCs. Each of the 278 off-detector electronic boards connected to the front-end by optical fibers will be fitted with two Intel Agilex FPGAs to process signals from all the 182468 LAr cells at 40 MHz. Artificial neural network (ANN) based algorithms are being developed to replace the OF to mitigate the effects of overlapping pulses in LAr energy reconstruction.

2. Energy reconstruction

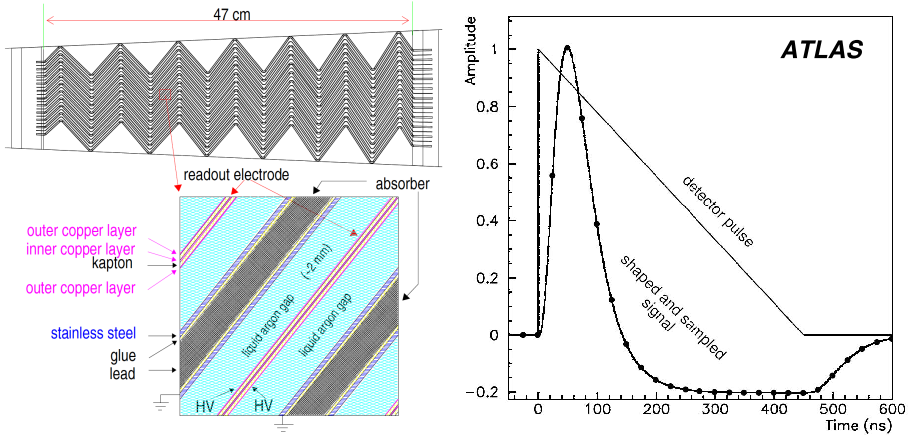

Figure 1: *(left)* Schematic view of a small sector of the barrel calorimeter in a plane transverse to the LHC beams. *(right)* Shaped and digitized LAr pulse. The dots indicate an ideal position of samples separated by 25 ns [4].

The LAr calorimeters are comprised of accordion-shaped copper-kapton electrodes positioned between lead absorber plates and the system is immersed in liquid argon (Figure 1). The induced pulse height due to ionization is proportional to the energy deposited in each calorimeter cell, while the pulse peaking time is used to measure the arrival time of the incident particle by OF. A previous study [5] has established that both convolutional neural networks (CNNs) and recurrent neural networks (RNNs) outperform OF, especially when there are overlapping pulses. This article further explores the applicability of these two architectures as viable replacement of OF on the FPGAs.

The ANNs are trained using data generated with AREUS [6] which combines electronic noise and low-energy deposits from an average pileup of 140 with signals of energy up to 5 GeV injected randomly with a mean interval of 30 BCs. It simulates the response of a LAr calorimeter cell in the middle layer of the barrel at pseudorapidity $\eta = 0.5125$ and azimuthal angle $\phi = 0.0125$.

2.1 Convolutional neural networks

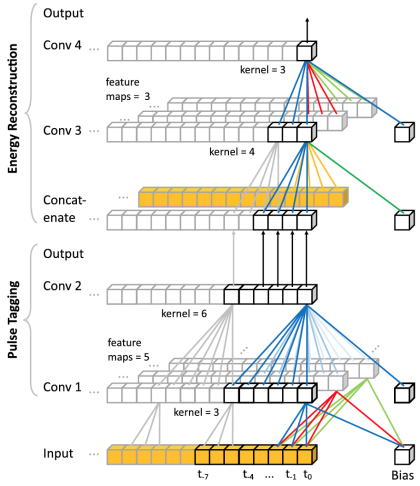

Figure 2: The CNN architecture used for LAr energy reconstruction in [5].

The CNN operates on the continuous time-series data coming from the ADC samples of one detector cell. It uses one-dimensional filters and consists of a two-staged architecture shown in Figure 2 where the first two layers are trained to tag energy deposits above the noise threshold. After a concatenation layer, the tag output and the input sequence are processed by the second stage of the CNN for energy reconstruction. A rectified linear unit (ReLU) [7] is used as the activation function. The input sequence lengths are 28 ADC samples and 13 ADC samples, respectively, for the three-layer and four-layer CNN variants.

2.2 Recurrent neural networks

RNNs are a family of neural networks adapted for processing sequential data. They contain simple or complex internal structures to control the flow of information to the next step in the sequence using NN layers. Figure 3 illustrates the operation of RNN in the sliding window mode, where the ADC sample sequence from the calorimeter is split into overlapping sub-sequences of length of five. Each of the five RNN cells of the network takes a sample value of the sub-sequences as input. The network is computed at every bunch crossing on the current input window, and the energy is extracted through a dense neuron connected to the last cell. Two types of RNN, the complex Long Short Term Memory (LSTM) and the simple Vanilla RNN, have been tested.

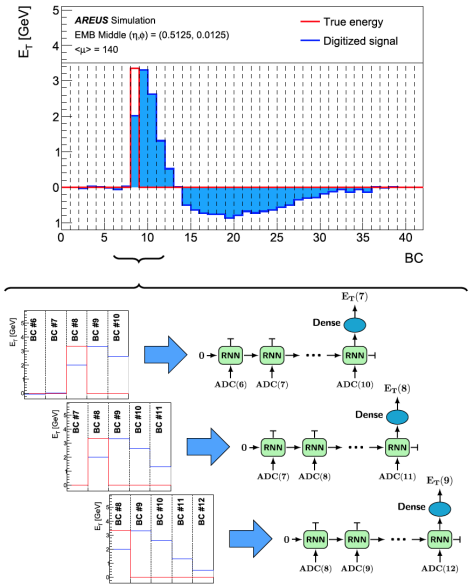

Figure 3: The RNN architecture in sliding window mode [5].

2.3 Performance

The transverse energy (E_T) resolution from various ANNs for energy deposits 3σ above the noise level (> 240 MeV) is shown in Figure 4. The ANNs perform better than the OF with MaxFinder as evident from the smaller bias on the mean and the better standard deviation. The main improvement happens in areas with overlapping events, as shown in Figure 5. Networks using more samples from past events yields a better correction for overlapping events.

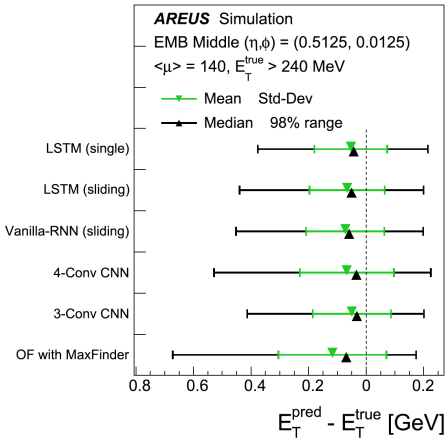

Figure 4: E_T resolution for E_T above 3σ noise level [5].

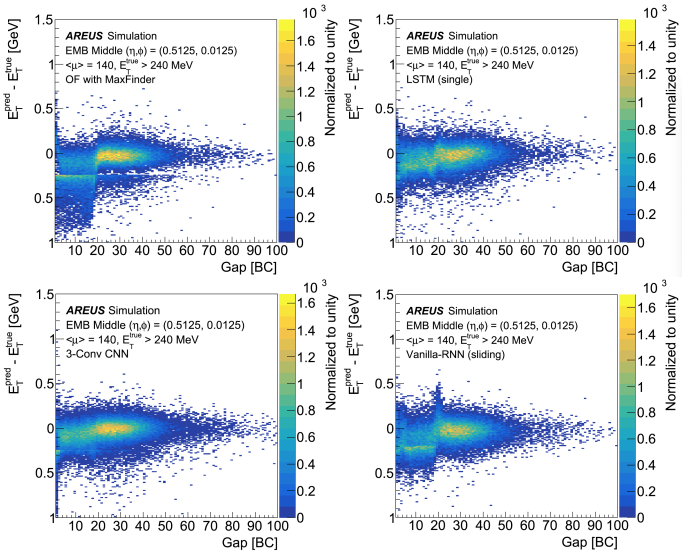

Figure 5: Resolution as a function of the time interval between two high energy deposits (Gap) [5].

3. Implementation on FPGA

The CNNs are implemented using Very High-speed integrated circuit hardware Description Language (VHDL). The firmware has a modular structure, where the number of layers and the parameters for each layer can be configured at compile time. Model architecture parameters are automatically extracted from Keras output. It is designed to support pipelining and time-division multiplexing, running at twelve times the sampling frequency and processing twelve detector cells cyclically.

FPGA implementation for the RNNs was prototyped with the Intel high-level synthesis (HLS) [8] compiler which enabled a detailed study of the effects of rounding in different parts of the network. FPGAs works most efficiently with 18 bit fixed point numbers, while the training happens in 32 bit floating point numbers. Different rounding or truncation strategies have been evaluated, as can be seen in Figure 6 *(left)* to minimize the effect of quantization of the arithmetic operations.

Several instances of the neural network are needed to process all the channels. However, for every compilation, each instance is placed differently on the FPGA due to randomization (Figure 6 *(right)*). This complicates the optimization of the timing critical paths needed to reach higher frequencies, and thus, higher multiplexing. A direct VHDL implementation of the RNN was created based on the initial HLS-inspired design to facilitate low-level optimizations such as

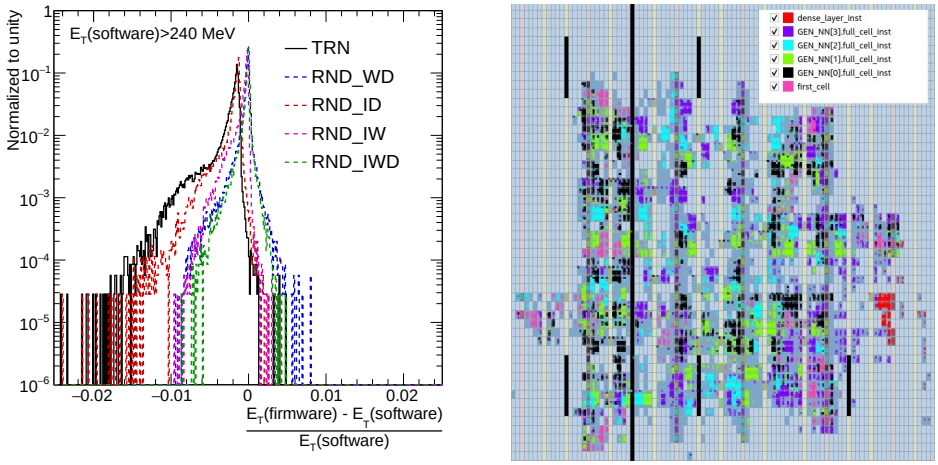

Figure 6: *(left)* Resolution of the transverse energy (E_T) computed in firmware with respect to the one computed in software. For each test, the letters I, W, and D indicate that rounding (RND) is applied for the internal data category, the weights, and the input/output data, respectively, while the truncation (TRN) mode is applied by default in all other categories [9]. *(right)* Random placement of network components on the FPGA before VHDL optimization.

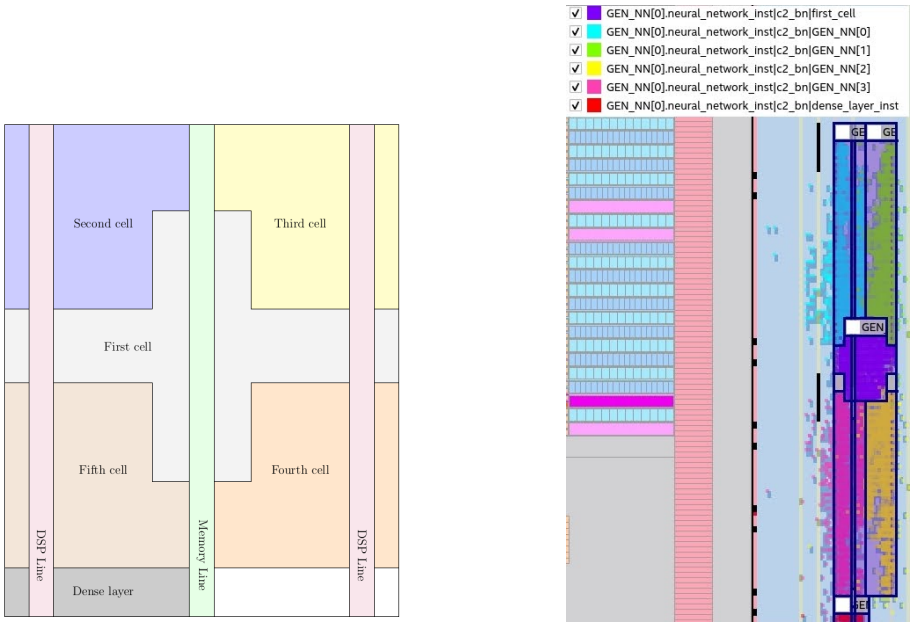

Figure 7: *(left)* Schematic view of the optimized placement of the Vanilla RNN showing the 5 RNN cells and the dense layer with respect to the memory and DSP lines inside the FPGA. *(right)* Constrained placement of network components on the FPGA after VHDL optimization. Each colour represents a cell in the RNN sequence.

reuse of common results between RNN cells or custom placement constraints, as can be seen in Figure 7. Thus, all instances of the neural network can be forced to have the same hardware implementation, which simplifies the optimization of the critical paths and minimizes the distance between connected cells. The network can be compiled multiple times incrementally so that for

Table 1: FPGA resource estimates for the Intel Stratix 10 and Agilex FPGAs.

FPGA	Network	Multiplexing	Channels	F_{max} [MHz]	ALMs	DSPs
Stratix-10	RNN (HLS)	10	370	393	90%	100%
	RNN (VHDL)	14	392	561	18%	66%
	2-Conv CNN	12	396	415	8%	28%
	4-Conv CNN	12	396	481	18%	27%
Agilex	2-Conv CNN	12	396	539	4%	13%
	4-Conv CNN	12	396	549	9%	12%

each iteration, the parts that meet the timing constraints are retained and only the violating regions are recompiled. Table 1 shows the FPGA resource estimates as a percentage of total available resources for the Intel Stratix-10 and Agilex devices. The RNN firmware with 28 network instances has been shown to be able to run at 560 MHz with a multiplexing of 14 and a latency of 116 ns [9].

4. Summary

Both CNN and RNN have been found to outperform the OF algorithm for energy reconstruction in the ATLAS LAr Calorimeter, particularly in the regions of overlapping pulses. Studies to quantify the effect on object (electrons, photons) reconstruction and physics performance are underway. The placement of the full firmware required to process 384 cells per FPGA has been shown to be feasible for both architectures through VHDL implementation.

5. Acknowledgments

This work was in part supported by the German Federal Ministry of Education and Research within the project 05H19ODCA9. The project has received funding from Excellence Initiative of Aix-Marseille Université - A*MIDEX, a French *Investissements d'Avenir* programme, AMX-18-INT-006 and from the French *Agence National de la Recherche*, ANR-20-CE31-0013. It was also supported by the French government under the France 2030 investment plan, as part of the Excellence Initiative of Aix-Marseille University - A*MIDEX (AMX-19-IET-008 - IPhU).

References

[1] The ATLAS Collaboration, *JINST* (2008) **3** S08003

[2] Lyndon Evans and Philip Bryant, *JINST* (2008) **3** S08001

[3] W. E. Cleland and E. G. Stern, Nucl. Instrum. Meth. A **338** (1994) no.2-3, 467-497

[4] ATLAS Collaboration, (2017) CERN-LHCC-2017-018, ATLAS-TDR-027

[5] G. Aad, et al., Comput. Softw. Big Sci. **5** (2021) no.1, 19

[6] N. Madysa, EPJ Web Conf. **214** (2019), 02006

[7] K. Fukushima, IEEE Trans. Syst. Sci. Cybern., **5**, no. 4, 322-333 (1969)

[8] Intel Corporation, *Intel High Level Synthesis Compiler*

[9] G. Aad, et al., JINST **18** (2023) no.05, P05017