Smartpixels: Towards on-sensor inference of charged particle track parameters and uncertainties

Jennet Dickinson[1], **Rachel Kovach-Fuentes**[2], **Lindsey Gray**[1], **Morris Swartz**[3], **Giuseppe Di Guglielmo**[1,4], **Alice Bean**[5], **Doug Berry**[1], **Manuel Blanco Valentin**[4], **Karri DiPetrillo**[2], **Farah Fahim**[1,4], **James Hirschauer**[1], **Shruti R. Kulkarni**[6], **Ron Lipton**[1], **Petar Maksimovic**[3], **Corrinne Mills**[7], **Mark S. Neubauer**[8], **Benjamin Parpillon**[1,7], **Gauri Pradhan**[1], **Chinar Syal**[1], **Nhan Tran**[1,4], **Dahai Wen**[3], **Jieun Yoo**[7], **Aaron Young**[6]

[1] Fermi National Accelerator Laboratory, Batavia, IL 60510, USA
[2] The University of Chicago, Chicago, IL 60637, USA
[3] Johns Hopkins University, Baltimore, MD 21218, USA
[4] Northwestern University, Evanston, IL 60208, USA
[5] University of Kansas, Lawrence, KS 66045, USA
[6] Oak Ridge National Laboratory, Oak Ridge, TN 37831, USA
[7] University of Illinois Chicago, Chicago, IL, 60607, USA
[8] University of Illinois Urbana-Champaign, Champaign, IL 61801, USA

Abstract

The combinatorics of track seeding has long been a computational bottleneck for triggering and offline computing in High Energy Physics (HEP), and remains so for the HL-LHC. Next-generation pixel sensors will be sufficiently fine-grained to determine angular information of the charged particle passing through from pixel-cluster properties. This detector technology immediately improves the situation for offline tracking, but any major improvements in physics reach are unrealized since they are dominated by lowest-level hardware trigger acceptance. We will demonstrate track angle and hit position prediction, including errors, using a mixture density network within a single layer of silicon as well as the progress towards and status of implementing the neural network in hardware on both FPGAs and ASICs.

Machine Learning and the Physical Sciences Workshop, NeurIPS 2023.

1 Introduction

Tracking detectors are an important component of modern particle physics detectors that provide the majority of kinematic information describing the aftermath of a collision of fundamental particles. The tracking detector's basic function is the measurement of space points, i.e. the "hits" on a "track", along the trajectories of charged particles that are typically being deflected by a magnetic field, without significantly altering the particles' trajectories during these measurements. Those space points are processed later by pattern recognition algorithms to yield instances of track hypotheses, each with estimates of the true parameters of the charged particle's kinematics. Achieving these measurements often requires algorithms that scale with the number of possible combinations of all input data, and this causes a significant increase in an experiment's computational needs for future colliders such as HL-LHC, FCC-hh, or a muon collider. Any new technology that restricts the number of combinations improves the physics impact of these future colliders by reducing this compute burden and allowing for more detailed downstream analysis. If a large enough reduction is achieved, it makes real-time trigger systems possible that drastically alter the physics reach of detectors. Focusing on trackers designed using silicon sensors, this reduction is traditionally achieved using special double-layers with local pattern recognition logic to select interesting hit pairs yielding coarse track hypotheses that permit fewer valid combinations when considering the output of other layers. In this work we demonstrate the feasibility of making single-layer track trajectory measurements using a pixelated silicon device, including credible error estimates, by applying machine-learning based uncertainty prediction techniques in quantized neural networks that can be rendered into both FPGA and radiation-hard circuits on the pixel readout ASIC. Furthermore, given the general nature of the techniques used, this technology could be beneficially applied in other real-time, on-device pattern recognition tasks. This technique improves the information supplied by a single layer of silicon, while reducing the material budget necessary to make such measurements, to the point that a pixel-based trigger system becomes feasible for the challenging environments expected at the HL-LHC and beyond.

2 Related Work

Usage of neural networks in pattern recognition problems is commonplace in HEP, though typically as a discriminant or refinement to the output of traditional pattern recognition algorithms. Both the CMS [1, 2] and ATLAS [3, 4] collaborations employ neural networks in their current reconstructions to improve the processing efficiency by removing or correcting mis-reconstructed space points or trajectory hypotheses. Expanding on this theme, next-generation pattern recognition algorithms that are based mostly or wholly in neural architectures are merging these refinements with the higher-level pattern recognition [5] to reduce the combinatorial fake rate and corresponding computational load. Our contribution to this avenue of research is to refine the techniques discovered so far by providing a complete statistical interpretation of the space-point and trajectory angle information, and to synthesize the network predicting these data into a form that is implementable in detector front-end hardware. This creates the possibility for novel, intelligent, and low mass detectors that can significantly improve triggering capabilities by providing space points close to the interaction region that do not cause egregious combinatorial growth in the number of patterns to consider.

3 Dataset and model architecture

The studies in this paper are based on a simulated dataset of silicon pixel clusters produced by charged particles (pions) [6]. The kinematic properties of the incident particles are taken from fitted tracks in CMS 13 TeV proton-proton collision data. To study a concrete sensor configuration, charge deposition is simulated in a 21×13 array of pixels described by coordinates $x \times y$, with the z direction normal to the sensor plane. The position (x, y) where the charged particle traverses the sensor mid-plane is assumed to be uniform across the central 3×3 pixel array. The sensor is taken to be 100 μm thick with pixel pitch 50 $\mu m \times$ 12.5 μm in $x \times y$. A bias voltage of -100V is applied and the detector is immersed in a 3.8 T magnetic field parallel to the x coordinate. The particle origin point is taken to be 30mm from the sensor plane.

The response of this detector is simulated using a time-sliced version of PixelAV [7]. This provides an accurate model of charge deposition by primary hadronic tracks, a realistic electric field map

resulting from the simultaneous solution of Poisson's Equation, carrier continuity equations, and various charge transport models, an established model of charge drift physics, a simulation of charge trapping and the signal induced from trapped charge, and a simulation of electronic noise, response, and threshold effects. PixelAV also provides the time evolution of the drift and induced currents in the pixel sensor, and the charge deposition is sampled at 20 time points each separated by 200ps. Figure 1 shows the time evolution of the cluster for an example particle, and the training dataset contains 3 million examples of such clusters.

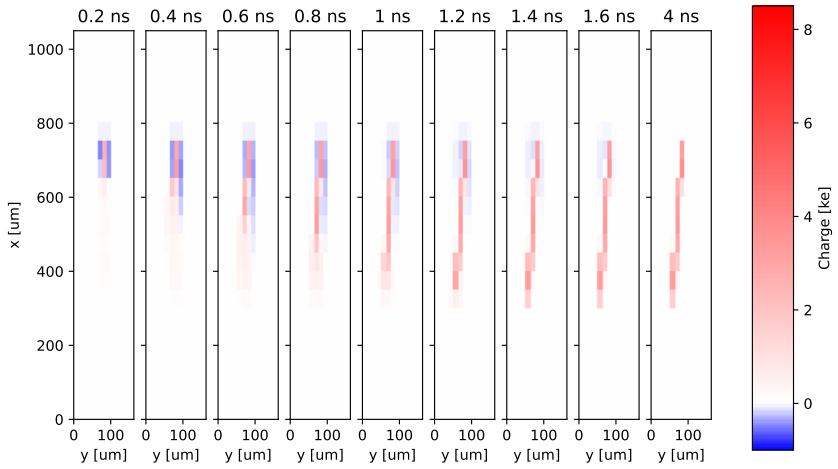


Figure 1: Charge deposition at different times during the cluster evolution for an example cluster. The color scale represents the collected charge in kilo-electrons, with blue representing negative induced charge.

The shape of the charge deposited in the pixel array is sensitive to the particle's impact position and angle of incidence. The collected charge is log-compressed, scaled to fit in the range $[-1, 1]$ and then quantized into 4 bits with a sign bit and 3 bits below the decimal point. The incident angle in the $x - z$ plane is denoted by α, and by β in the $y - z$ plane (the bending plane of the magnetic field).

In order to achieve a well-performing network that can reduce combinatorics in downstream reconstruction tasks, we employ a mixture density network (MDN) [8] with some specializations to the task at hand. In order to provide a trajectory state estimate at the mid-plane of the pixel sensor the model must predict the local x, y, α, and β of the cluster, as well as the associated covariance matrix. The response and resolution distributions are inherently multi-Gaussian, but safely approximated as Gaussian per-cluster, from the varying number of hits per cluster. Therefore, the network predicts the parameters of a single multi-dimensional Gaussian, and we construct the mixture of the MDN at the level of the likelihood over the training data. This likelihood is the loss function that is minimized for this machine learning task.

The model itself is a 6 layer quantized network with 3 convolutional layers and 3 dense layers built using the QKeras package [9], which enables a quantization-aware training. The model is then trained on the previously described dataset for 300 epochs. The architecture of the model is shown in Figure 2. The convolutional layers are implemented as depthwise-separable convolutions to reduce the number of model parameters and total operations needed and the averaging pooling layer is introduced to reduce the number of dense-layer parameters by nearly a factor of 4, to minimize on-device resource usage. The optimization of the parameter bitwidths was performed by hand, focusing on reducing the bitwidth in the computationally expensive convolutional layers as much as possible while still retaining good regression performance. This has the particular benefit of keeping the necessary bitwidth for accumulators, registers that contain total results of multiply-accumulate operations, beneath the threshold of requiring complicated multipliers to be used or implemented during synthesis for target devices. To translate the algorithm from a quantized graph representation into an efficient hardware implementation, we use `hls4ml`, an open-source Python framework for co-design [10, 11].

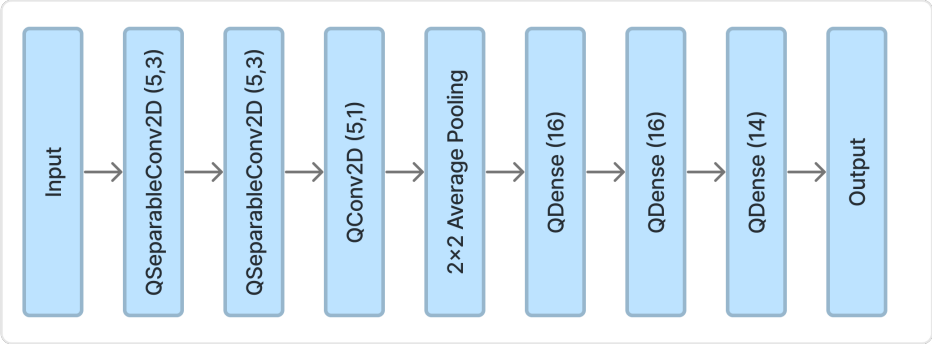
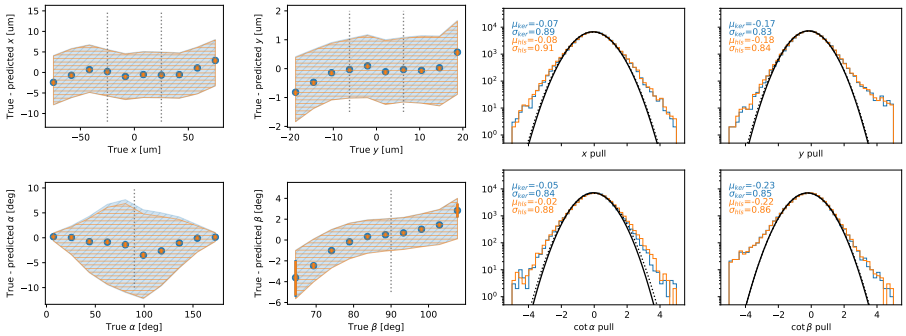


Figure 2: A block diagram of the model architecture used for the mixture-density network based regression of track parameters. The first two separable convolution layers output five filters have have a 3x3 kernel, and the final pointwise convolution outputs 5 filters. The first two dense layers have 16 output units and the last dense layer has 14. The convolutional layers have 4-bit quantized weights with 3 bits below the decimal point and a sign bit, and the dense layers have 8-bit quantized weights with 7 bits below the decimal point and a sign bit. All layers except for the average pooling are activated with a "hard-tanh" function that has the same quantization as for the respective layer weights. The average pooling is 8-bit quantized with 7 bits after the decimal point and a sign bit.

4 Results



(a) (b)

Figure 3: A: The mean of the residual distribution and average predicted error band for QKeras (blue, cross-hatched) and `hls4ml` synthesized (orange, horizontal lines) regression networks. There are slight biases at the endpoints of the training data in the raw network output. Average predicted local coordinate errors of 5.7 microns in x, 1.1 microns in y, 3.8 degrees in the polar angle, and 1.7 degrees in the azimuthal angle are achieved. B: The distribution of the pull test-statistic for each predicted variable demonstrating that the predicted errors are 3-18% larger than the width of the residual distributions. Excellent, but not bitwise perfect, agreement between the QKeras and `hls4ml` models is observed.

After training the network and determining quantization parameters that achieve good agreement between the QKeras and `hls4ml` implementations of the network, results are attained as summarized in Figure 3. To expound upon these results and demonstrate their utility, we shall focus on the predictions of angles. Using the average predicted errors from the network on the test data sample of 600,000 clusters and using a tolerance of a factor of four on the error, the average size of a solid angle patch with high efficiency for a match is smaller than 0.02 steradians, 0.2% of the solid angle, or a 4 cm^2 search region for hits in a subsequent pixel layer 2.2 cm away. This implies that with such a 'singlet' track seed, we can retain high track seeding efficiency without any constraints on

a charged track's transverse momentum while also reducing the initial combinatorics by orders of magnitude. Similarly, this relaxes the design requirements for track triggers that include pixel data by significantly reducing the number of initial matches to search through when, for instance, matching tracks from an outer tracker to pixel hits. In addition, this reduces needs for data replication and sharing across multiple trigger regions in the same layer.

Concerning technical aspects, it is important to note that while bit-wise equivalence is not attained with these parameters, the performance of the two implementations of the network is nearly equivalent. There is also room for additional optimization of the QKeras implementation, particularly the bit-widths. When synthesized into firmware with the Xilinx Vivado toolkit for the Alveo U250 accelerator card, this network requires no digital signal processors (DSP) or heavy processing resources to implement and so stands as a promising candidate for synthesis into the digital part of a front-end ASIC in a future pixel detector as seen in Table 1. Justifying that, the ASIC synthesis is achieved in 45 nm CMOS using Catapult HLS [12] for a 200 MHz clock, and the logic synthesis completed successfully and can now be used to guide place and route for an ASIC design. The differences in latency between the FPGA and ASIC syntheses can be attributed to different compiler-specific optimizations being applied, due to the different maturity of backends for Vivado and Catapult in `hls4ml`, and are likely not representative of inescapable flaws in the ASIC synthesis. In both cases, the next major improvement in inference latency and initiation interval will come from a completely parallel implementation, as opposed to the current streaming implementation, in the `hls4ml` backends. This implementation is not used in this work because the parallel SeparableConv2D layer is missing from `hls4ml` and is currently being implemented. This change will remove the predominant source of latency, which is loading of the data pixel-by-pixel, though it may be replaced by additional resource usage in the spatial domain. The next steps for this network are to fully validate its current performance in a real FPGA, and take the preliminary ASIC synthesis further including full place-and-route, understanding power usage, implementation in our target 28 nm process, and optimize input data requirements as well as implementation size.

QKeras Model Analysis		Alveo U250 Synthesis		ASIC Synthesis (45nm)	
4-bit ops.	50,140	Clock Period	5 ns	Clock Period	5 ns
8-bit ops.	3,040	BRAM_18K	12.5	Area Estimate	1.4 mm^2
1^{st} Layer N_0	17	DSP48E	0	Buffer Area	0.0017 mm^2
2^{nd} Layer N_0	9	FF	14,289	Inverter Area	0.055 mm^2
3^{rd} Layer N_0	2	LUT	57,398	Logic Area	0.80 mm^2
4^{th} Layer N_0	117	URAM	0	Sequential Area	0.52 mm^2
5^{th} Layer N_0	15	Latency	1.46 μs	Latency	27 μs
6^{th} Layer N_0	51	Interval (II)	1.38 μs	-	-

Table 1: QKeras Model Analysis (left), FPGA Firmware (middle) and ASIC (right) complete synthesis figures of merit including resource usage results. The QKeras Model Analysis provides an operation-wise breakdown of how many N-bit ops occur in the model. The number of sparse parameters, N_0, per layer, is also summarized, indicating that further model compression is possible using pruning techniques. FIFO-depth optimization is included in the FPGA synthesis workflow. The ASIC synthesis is successful and represents the first attempt at using Catapult HLS for this design. The long latency and II for FPGA and ASIC syntheses are largely due to the implementation of the architecture as streaming 20 4 bit words per pixel rather than presenting all data in parallel. The amount and nature of data being fed to the model is under heavy study to achieve a realistic and low-power implementation, and this should not be taken as a fundamental limitation on the model's performance or suitability for implementation.

5 Conclusions

In this work we have demonstrated a highly compressed, quantized, and well performing regression neural network for predicting track states from a possible future silicon pixel sensor that is implementable in modern FPGAs and silicon lithography processes. This network is inspired by and improves upon techniques used in modern tracking detector pattern recognition research. In addition to mean positions and angles, the covariance matrix for these values is predicted using a

mixture density network technique, which provides the novel capability to predict efficient search windows for additional hits with a single silicon sensor. The first estimate of the average size of this window indicates that a reduction of two orders of magnitude in search space for track seeds is possible without making assumptions on the track momentum parameters, which indicates that this network could be used as part of the the readout scheme directly on the pixel sensor. Given the general nature of the technique used here, this sort of on-device, real time, data-compressing feature extraction with error prediction could be developed for a variety of sensing hardware that can be deployed in high data-rate experiments across the physical sciences. The next steps of research are to start investigating network power usage and implementation as an ASIC in 28 nm CMOS, and to verify the network performance in hardware.

References

[1] "The CMS experiment at the CERN LHC. The Compact Muon Solenoid experiment". In: *JINST* 3 (2008). Also published by CERN Geneva in 2010, S08004. DOI: 10.1088/1748-0221/3/08/S08004. URL: https://cds.cern.ch/record/1129810.

[2] Walaa Elmetenawee. *CMS track reconstruction performance during Run 2 and developments for Run 3*. 2020. arXiv: 2012.07035 [physics.ins-det].

[3] "The ATLAS Experiment at the CERN Large Hadron Collider". In: *JINST* 3 (2008). Also published by CERN Geneva in 2010, S08003. DOI: 10.1088/1748-0221/3/08/S08003. URL: https://cds.cern.ch/record/1129811.

[4] Elham E. Khoda. "ATLAS pixel cluster splitting using Mixture Density Networks". In: *PoS* LHCP2019 (2019), p. 009. DOI: 10.22323/1.350.0009.

[5] P.J. Fox et al. "Beyond 4D tracking: using cluster shapes for track seeding". In: *Journal of Instrumentation* 16.05 (May 2021), P05001. DOI: 10.1088/1748-0221/16/05/P05001. URL: https://dx.doi.org/10.1088/1748-0221/16/05/P05001.

[6] Morris Swartz and Jennet Dickinson. *Smart pixel dataset*. Version 1. Zenodo, Nov. 2022. DOI: 10.5281/zenodo.7331128. URL: https://doi.org/10.5281/zenodo.7331128.

[7] Morris Swartz. *A Detailed Simulation of the CMS Pixel Sensor*. Tech. rep. Geneva: CERN, 2002. URL: https://cds.cern.ch/record/687440.

[8] Christopher M. Bishop. *Mixture density networks*. English. WorkingPaper. Aston University, 1994.

[9] Claudionor N. Coelho et al. "Automatic heterogeneous quantization of deep neural networks for low-latency inference on the edge for particle detectors". In: *Nature Machine Intelligence* 3.8 (June 2021), pp. 675–686. DOI: 10.1038/s42256-021-00356-5. URL: https://doi.org/10.1038%2Fs42256-021-00356-5.

[10] FastML Team. *fastmachinelearning/hls4ml*. 2021. DOI: 10.5281/zenodo.1201549. URL: https://github.com/fastmachinelearning/hls4ml.

[11] Javier Duarte et al. "Fast inference of deep neural networks in FPGAs for particle physics". In: *JINST* 13.07 (2018), P07027. DOI: 10.1088/1748-0221/13/07/P07027. arXiv: 1804.06913 [physics.ins-det].

[12] Siemens. *Catapult HLS*. https://eda.sw.siemens.com/en-US/ic/ic-design/high-level-synthesis-and-verification-platform.

Appendix: Acknowledgements

This work was completed using computing resources at the Fermilab Elastic Analysis Facility (EAF). We thank Burt Holzman for computing support.

We acknowledge the Fast Machine Learning collective as an open community of multi-domain experts and collaborators. This community, Javier Duarte and Vladimir Loncar in particular, were important for the development of this project.

We would like to extend our sincere gratitude to Harish Jamakhandi and David Burnette from Siemens EDA for their assistance and expertise with Catapult HLS.

DB, JD, GDG, FF, LG, JH, RL, BP, GP, CS and NT are supported by Fermi Research Alliance, LLC under Contract No. DE-AC02-07CH11359 with the Department of Energy (DOE), Office of Science, Office of High Energy Physics. JD, FF, GDG, BP, GP, and NT are also supported by the DOE Early Career Research Program. NT is also supported by the DOE Office of Science, Office of Advanced Scientific Computing Research under the "Real-time Data Reduction Codesign at the Extreme Edge for Science" Project (DE-FOA-0002501).

AB is supported through NSF-PHY award 2013007. MS is supported by NSF-PHY award 2012584. CM is supported by NSF-PHY award 2208803. KD is supported in part by the Neubauer Family Foundation Program for Assistant Professors and the University of Chicago. RK is supported by the Metcalf Fellowship program of the University of Chicago. AY and SK are supported by the DOE Office of Science Research Program for Microelectronics Codesign (sponsored by ASCR, BES, HEP, NP, and FES) through the Abisko Project. MSN is supported through NSF cooperative agreement OAC-2117997, the DOE Office of Science, Office of High Energy Physics, under Contract No. DE-SC0023365, REFERENCES 24 and the Discovery Partners Institute under the "Democratizing AI Hardware with an Open-Source AI-Chip Design Toolkit" Project.