ISBN: 978-3-95450-262-2 ISSN: 2673-5350 doi: 10.18429/JACoW-IBIC2025-MOPCO36

APPLICATION OF CONVOLUTIONAL NEURAL NETWORKS FOR PILE UP CORRECTION IN SINGLE PARTICLE COUNTING

T. Habermann*,[1], M. Kumm[1], R. Singh[2]
[1]Fulda University of Applied Sciences, Fulda, Germany
[2]GSI Helmholtzzentrum fuer Schwerionenforschung, Darmstadt, Germany

Abstract

An exploration into the application of machine learning (ML) approaches for real time identification and correction of pile-ups in single particle counters at the GSI Helmholtz Centre for Heavy Ion Research is presented. About 26,000 particle pulse data were manually labelled and a convolutional neural network (CNN) was developed to accurately locate the particles without domain-specific knowledge. This contribution represents proof-of-work for a fast error-free automated particle counting system. The identified algorithm was developed with a perspective of implementation into FPGA.

INTRODUCTION

Accurate particle counting along with their arrival time determination is fundamental to experimental nuclear and particle physics. Similarly in accelerator beam diagnostics, real-time and precise measurements of particle fluxes and their temporal distribution is critical for providing usable particle beams (spill) to the user experiments. Scintillator-type single particle counters face significant challenges from pile-up events [1], e.g., where multiple particles arrive within a short time interval causing superimposed pulse shapes at the output of the sensor. Typically the number of pile-ups in extracted beams from particle accelerators is much more than one expects from a random discrete process or Poisson process [2,3]. Figure 1 shows examples where pulses pile-up in the time series data.

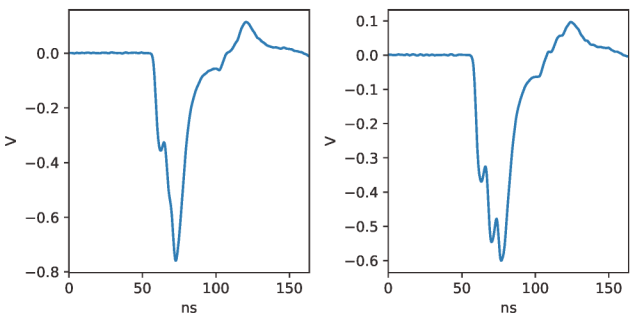

Figure 1: Examples of pile-ups of multiple particles where pulses overlap in the time series data of the scintillator sensor.

Traditional real-time methods struggle to decompose this complex composite signals. This can lead to significant undercounting which can impact precision experiments. Most traditional methods rely on analytical models and expert-designed algorithms that require extensive calibration and often fail with complex pile-up scenarios involving multiple overlapping particles [4]. The most used algorithm for high rate real-time particle counting is still single or double threshold crossing algorithms. Other post-processing algorithms relying on analytical models or heuristics like template matching with deconvolution, or peak property–based detection have also been used [5–7].

Recent advances in machine learning, particularly convolutional neural networks (CNNs), have demonstrated remarkable success in pattern recognition tasks involving complex time series data [8, 9]. CNNs can automatically learn hierarchical representations without requiring explicit domain knowledge, making them well-suited for pile-up identification where relevant features may be subtle and difficult to capture analytically.

In this work, we present a CNN-based approach for particle counting and pile-up identification in scintillator counters. We trained our network on approximately 26,000 manually labeled particle pulse events, treating the time series data as one-dimensional signals suitable for CNN analysis. Our approach achieves accurate particle counting without relying on detailed analytical models, with the CNN model developed specifically for their real-time implementation on field programmable gate arrays (FPGAs).

DATA

Single particle induced pulse data was collected at the HTP beamline at GSI Helmholtz Centre for Heavy Ion Research. Average beam extraction rate from synchrotron SIS-18 is controlled with the spill optimization system [10]. A plastic (BC400) scintillator coupled with a photo multiplier tube via a light guide was used to collect the single particle pulse data. The scintillator is irradiated with Carbon (C^{6+}) ion beam with an extraction kinetic energy of 300 MeV/u.

Pulse Characteristics

All single pulses are caused by production of many photons (100-10000s depending on energy deposition) per ion incidence in the scintillation medium which results in a linearly scaled number of charges at the PMT output. Therefore, ions within the same spill (which are of the same energy and charge state) induce PMT pulses which share the same characteristics. Since the area under all pulses is comparable, the integral can be used to estimate the number of pulses present in any user defined time window. The integral-indicator is a

* tobias.habermann@informatik.hs-fulda.de

ISBN: 978-3-95450-262-2 ISSN: 2673-5350 doi: 10.18429/JACoW-IBIC2025-MOPCO36

sum over all discrete data points (y) given by

$$I(y) = \frac{1}{|y|}\sum_{i=1}^{|y|} y_i - \delta\,, \tag{1}$$

where δ describes the average of the dataset and is used to center the signal data around zero. The histogram in Fig. 2 shows the distribution of the integral-indicator for all single pulses and pile-ups. The indicator has a tendency to

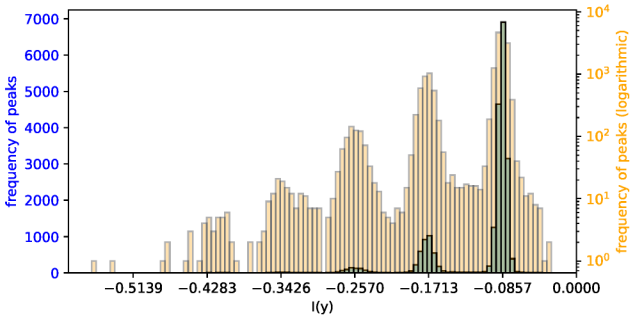

Figure 2: A histogram of the integral-indicator for all single pulses (at $\kappa = -0.0857$) and pile-ups (multiples of κ).

be some multiple of $\kappa = -0.0857$. This can be expected as each indicator either represents a single pulse or a pile-up containing multiple pulses. In general, there are more examples of single pulses than pile-ups and pile-ups with many pulses occur less frequently.

Preprocessing

The data was labeled in a multistep process. At first, the data was labeled using a simple thresholding approach and manual inspection. This allowed to analyse the data to find κ. With κ, all pile-ups could be labeled with the correct number of pulses. After this step, the imbalance between single pulses and pile-ups became more apparent.

Identifying the peak of a single pulse is a straightforward process. However, accurately localizing the peaks of individual pulses within a superimposed signal presents a significantly greater challenge. To allow for a more accurate localization in pile-ups, and to fix the imbalance between the number of single pulse and pile-up examples, we created superimposed single pulses. By taking multiple single pulse examples from the dataset and superimposing them, we create a pile-up example where all peak locations are exactly known. This simple method fixes the imbalance between single pulses and pile-up examples. By training the proposed CNN on the superimposed single pulses, where all exact peak locations are known exactly, the model was able to learn how to locate peaks in pile-ups. The trained model was then used to refine the dataset further to allow a more accurate peak localization in pile-ups.

CNN ARCHITECTURE

The model architecture was designed without fully connected layers. This design choice enables training and inference on inputs of almost arbitrary size and reduces the number of weights, as fully connected layers scale quadratically with input size.

Our model follows ideas of the YOLO CNN [11] adapted to 1D signals. The input is divided into equal-sized cells (16 samples each), and each cell can predict up to m peaks. For each of the m peaks per cell, the model outputs a relative position and a confidence score. The relative position encodes the peak's location within the cell, whereas the confidence score reflects the estimated likelihood that the predicted position corresponds to an actual peak.

Following this design approach, the model is processing the continuous data stream window after window where the window size has to be some multiple of the cell size.

The CNN architecture is shown in Table 1. The first two conv. layers shrink the input width down by a factor of 16. This defines how big each cell is. Each following layer is then processing each of the n cells individually. The biggest kernel size of the following layers is 3, which constrains each cell to only consider its direct neighbours.

Table 1: Model Architecture for n Cells Where Each Cell Can Detect up to m Pulses

Layer	Kernel	Stride	Filters	Output shape
input	-	-	-	$(16n, 1)$
conv1	5	4	8	$(4n, 8)$
conv2	5	4	16	$(n, 16)$
conv3	3	1	16	$(n, 16)$
conv4	1	1	64	$(n, 64)$
conv5	3	1	32	$(n, 32)$
conv6	1	1	$2m$	$(n, 2m)$
reshape	-	-	-	$(n, m, 2)$

This constraint mitigates edge effects in pulse detection. After each inference, the last cell is discarded as it is lacking a neighbour, and the next window includes the final 32 inputs from the previous one. This preserves edge precision, at the cost of some data reprocessing, which can be reduced by enlarging the window size. The last convolutional layer produces the final output features. For each cell, all m pulse features are predicted. Most layers use a simple ReLu activation. Only the last layer uses a sigmoid activation.

An example output of the model for data containing 6 pulses is shown in Fig. 3. The window size is 256, so there are 16 cells. Each cell is highlighted with an alternating white and grey background. The figure shows all predicted peak locations where the confidence score is greater than 0.5. It can be seen that all predictions match closely with the ground truth marked in green.

Loss Function

As described in the previous section, the model outputs predictions for each cell. The output tensor is split into the confidence $\hat{p}_c$ and position $\hat{p}_x$ predictions. With the corresponding ground truth data p_c and p_x, the loss function

ISBN: 978-3-95450-262-2 ISSN: 2673-5350 doi: 10.18429/JACoW-IBIC2025-MOPCO36

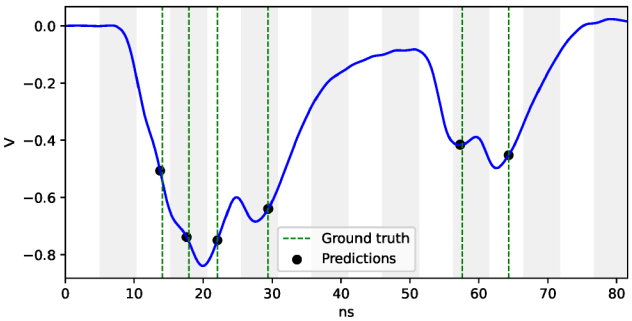

Figure 3: An example containing 6 pulses with all true peak locations and predicted locations shown in the figure. Also, each cell is highlighted.

is defined as the sum of the confidence and position loss:

$$\mathscr{L}(\hat{p}_c, p_c, \hat{p}_x, p_x) = \mathscr{L}_{\text{conf}} + \mathscr{L}_{\text{pos}} \,. \tag{2}$$

In most cases, a cell will not contain any pulse rather than m pulses. To fix this imbalance, an α-balanced focal loss [12] is used. This introduces two hyperparameters α and γ to dampen the loss for empty sections and raise the loss for sections with actual pulses that were misclassified. As described in [12], we can define $\hat{p}_t$ as

$$\hat{p}_t = \begin{cases} \hat{p}_c, & \text{if } p_c = 1 \\ 1 - \hat{p}_c, & \text{otherwise} \end{cases} \tag{3}$$

and the weighting for each confidence prediction α_t as

$$\alpha_t = \begin{cases} \alpha, & \text{if } p_c = 1 \\ 1 - \alpha, & \text{otherwise} \end{cases} . \tag{4}$$

The confidence loss can be defined as the average loss for each pulse prediction using focal loss

$$\mathscr{L}_{\text{conf}} = \frac{\lambda_{\text{conf}}}{nm} \sum_i -\alpha_{t,i}(1 - \hat{p}_{t,i})^{\gamma} \log(\hat{p}_{t,i}) \,, \tag{5}$$

where λ_{conf} is used to weight the whole loss term to allow further fine-tuning of the loss function. The position loss is based on the average MSE between $\hat{p}_x$ and p_x for all pulses. Thereby, it can be defined as

$$\mathscr{L}_{\text{pos}} = \frac{\lambda_{\text{pos}}}{N + \varepsilon} \sum_i p_{c,i}(\hat{p}_{x,i} - p_{x,i})^2 \,, \tag{6}$$

if there is no pulse present at i, $p_{c,i}$ is zero, penalizing only the relevant position predictions similar to [11]. The sum is then divided by the number of pulses present in the ground truth data N with a small ϵ added to prevent division by zero.

EXPERIMENTAL RESULTS

The model was trained on superimposed single pulses generated from a small subset of the dataset. The rest of the dataset was used for validation without any balancing or augmentations to ensure realistic validation results and contained 25,678 pulses in total. Table 2 shows how the model performs compared to triple thresholding (with Thresholds set to −0.3, −0.5, and −0.8, counting one, two or three pulses with respect to the threshold) and a local maxima search [13] (`scipy.signal.find_peaks` function with distance=9 and height=−0.2) on the validation data. The CNN model outperforms both methods with a difference of just 101 pulses between the total predicted number and the actual number of pulses.

Table 2: The Number of Predicted Pulses for Each Method Compared to the True Number of Pulses

	T. thr.	**Loc. max. search**	**Ours**
Pred. pulses	25,461	25,083	25,779
Difference	217	595	101

We created superimposed single pulses using all single pulses in the validation data, to test the localization performance. The data consists of 1500 pile-ups: 500 with 2 pulses, 500 with 3 pulses, and 500 with 4 pulses. Leading to a total of 4500 pulses. Each of the 1500 pile-ups is contained in a single window. For each method, we measured the absolute localization error, in units of sampling steps, for each pulse prediction for each of the 1500 windows. When the predicted number of pulses was miscounted, matching them becomes difficult, so the predicted location data was disregarded. Table 3 shows the results. It can be seen that most of the pulses are disregarded when using the triple thresholding, and the local maximum search method because both methods often fail to predict the right number of peaks. But, if the predictions can be matched, both methods show a low average and median error. However, the CNN model was able to predict most pulses correctly and shows the lowest average, median and maximum localization error.

Table 3: The Absolute Localization Error for Each Method and the Number of Disregarded Pulses

	T. thr.	**Loc. max. search**	**Ours**
Disre. pulses	4,070	3,545	98
Avg. loc. err.	2.044	1.809	0.385
Med. loc. err.	1.0	1.0	0.0
Max. loc. err.	211.0	18.0	9.0

CONCLUSION AND OUTLOOK

We presented a CNN-based approach for pile-up correction in single particle counting that outperforms traditional thresholding and local maxima methods in both pulse counting and localization. The lightweight architecture is well-suited for FPGA deployment, enabling real-time integration into accelerator diagnostics. Future work will focus on generating precise ground truth datasets, checking the generalization of CNN inference to data taken at different particle energy and extraction rates as well as implementing and validating the model on FPGAs.

ISBN: 978-3-95450-262-2 ISSN: 2673-5350 doi: 10.18429/JACoW-IBIC2025-MOPCO36

REFERENCES

[1] W.-H. Wong and H. Li, "A scintillation detector signal processing technique with active pileup prevention for extending scintillation count rates", *IEEE Trans. Nucl. Sci.*, vol. 45, no. 3, pp. 838–842, Jun. 1998. doi:10.1109/23.682647

[2] R.Singh, P. Forck, and S. Sorge, "Reducing Fluctuations in Slow-Extraction Beam Spill Using Transit-Time-Dependent Tune Modulation", *Phys. Rev. Appl.*, vol. 13, p. 044076, Apr. 2020. doi:10.1103/PhysRevApplied.13.044076

[3] P. Niedermayer and R. Singh, "Excitation Signal Optimization for Minimizing Fluctuations in Knock out Slow Extraction", *Sci. Rep.*, vol. 14, p. 10310, May 2024. doi:10.1038/s41598-024-60966-y

[4] D. Bertolini, P. Harris, M. Low, and N. Tran, "Pileup per particle identification", *J. High Energy Phys.*, vol. 2014, p. 59, Oct. 2014. doi:10.1007/JHEP10(2014)059

[5] S. Engel, P. Boutachkov, and R. Singh, "Application of Machine Learning towards Particle Counting and Identification", in *Proc. IBIC'22*, Kraków, Poland, Sep. 2022, pp. 508–511. doi:10.18429/JACoW-IBIC2022-WEP42

[6] V. Niennattrakul, D. Srisai, and C. A. Ratanamahatana, "Shape-based template matching for time series data", *Knowledge-Based Syst.*, vol. 26, pp. 1–8, Feb. 2012. doi:10.1016/j.knosys.2011.04.015

[7] G. Zhao, L. Wu, F. Grancagnolo, N. D. Filippis, M. Dong, and S. Sun, "Peak finding algorithm for cluster counting with domain adaptation", *Comput. Phys. Commun.*, vol. 300, p. 109208, Jul. 2024. doi:10.1016/j.cpc.2024.109208

[8] N. M. Foumani, L. Miller, C. W. Tan, G. I. Webb, G. Forestier, and M. Salehi, "Deep Learning for Time Series Classification and Extrinsic Regression: A Current Survey", *ACM Comput. Surv.*, vol. 56, no. 9, pp. 1–45, Apr. 2024. doi:10.1145/3649448

[9] Z. Zamanzadeh Darban, G. I. Webb, S. Pan, C. Aggarwal, and M. Salehi, "Deep Learning for Time Series Anomaly Detection: A Survey", *ACM Comput. Surv.*, vol. 57, no. 1, pp. 1–42, Oct. 2024. doi:10.1145/3691338

[10] P. Niedermayer, H. Bräuning, T. Milosic, and R. Singh, "Spill Optimization System Improving Slow Extraction at GSI", in *Proc. IPAC'25*, Taipei, Taiwan, Jun. 2025, to be published. doi:10.18429/JACoW-IPAC2025-TUPB008

[11] J. Redmon, S. Divvala, R. Girshick, A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection", in *Proc. IEEE Conf. Comput. Vision Pattern Recognit. (CVPR)*, Las Vegas, USA, Jun. 2016, pp. 779–788. doi:10.1109/CVPR.2016.91

[12] T. Lin, P. Goyal, R. Girshick, K. He, P. Dollár, "Focal Loss for Dense Object Detection", in *Proc. IEEE Int. Conf. Computer Vision (ICCV'17)*, Venice, Italy, Oct. 2017, pp. 2980–2988. doi:10.1109/ICCV.2017.324

[13] P. Virtanen *et al.*, "SciPy 1.0: fundamental algorithms for scientific computing in Python", *Nat. Methods*, vol. 17, pp. 261–272, 2020. doi:10.1038/s41592-019-0686-2