COMPARISON STUDY BETWEEN CONVOLUTIONAL AND RECURRENT NEURAL NETWORKS FOR MICROMEGAS DETECTORS' TRIGGERING

I.S. TRANDAFIR[1,2], I.-M. DINU[1,2b], C. ALEXA[1]

[1]Horia Hulubei National Institute for Physics and Nuclear Engineering (IFIN-HH),
Elementary Particle Physics Department, Reactorului 30, 077125 Măgurele, Romania
[2]University of Bucharest, Faculty of Physics, Atomiștilor 405, 077125 Măgurele, Romania
Corresponding author[b]: ioan.dinu@nipne.ro

Received March 7, 2023

Abstract. Compounded by the current leaps in hardware acceleration, Neural Networks are becoming viable for high data-rate applications such as data acquisition trigger algorithms. Using data from a simulated assembly of eight Micromegas detector planes interacting with muons originating from $p-p$ collisions, two neural network approaches are compared in terms of resource requirements and track reconstruction performance. Within our trigger use-case, the track's slope is used as part of the discriminant variable for rejecting background muons that can't be traced back to the interaction point. Convolutional and Recurrent Neural Networks are trained on the same dataset and the results are compared in terms of the angular resolution and the number of parameters. The Convolutional Neural Network is found to be more efficient in terms of the number of parameters, but the Recurrent Neural Network is more robust to noise and has a better angular resolution.

Key words: Machine learning, FPGA, trigger.

1. GOALS AND MOTIVATION

The aim of this work is to compare the performance of two different machine learning approaches for the muon trigger algorithm previously proposed in [1]. Specifically, we will compare the performance of a Convolutional Neural Network (CNN) with that of a Recurrent Neural Network (RNN) on simulated muon tracks. While the CNN approach was successful in detecting muon tracks, we seek to determine whether the RNN approach is a more suitable alternative for the same problem. We will evaluate the performance of the two approaches in terms of accuracy and efficiency, taking into account the limited computing and memory resources of FPGAs relative to modern CPU-based systems.

In order to ensure a fair comparison between the two approaches, we will use the same dataset and training procedure as in [1]. Specifically, we will train the RNN to compute the same difference between the local and global track angles as the CNN: $\Delta\theta = |\theta_{\text{global}} - \theta_{\text{local}}|$. We will pay close attention to the size of the RNN model in terms of number of parameters, with the aim of keeping it comparable to that of the CNN, in order to ensure fair comparison of the two approaches.

This work has significant implications for High Energy Particle Physics, where the amount of experimental data continues to grow, and where Micromegas-type detectors are becoming increasingly popular. The development of a highly efficient and accurate muon trigger algorithm, based on machine learning and FPGA deployment, has the potential to revolutionize data acquisition and analysis in the field. By comparing the performance of two different machine learning approaches to the same problem, this work will provide valuable insights into the effectiveness of different neural network architectures for high-speed, high-resolution particle detection in harsh, high pile-up environments.

2. EXPERIMENTAL SETUP

In this research, the feasibility of a muon trigger system based on Micromegas detectors is investigated. The Micromegas detectors are gas-based detectors that operate based on the ionization of gas by charged particles. The primary electrons generated by the ionization process drift towards the amplification region where they produce an electron avalanche, and the resulting signals are read out by anode strips. Figure 1 shows an illustration of this process. The advantages of this detector technology, including excellent spatial and time resolutions, make it suitable for various particle physics applications.

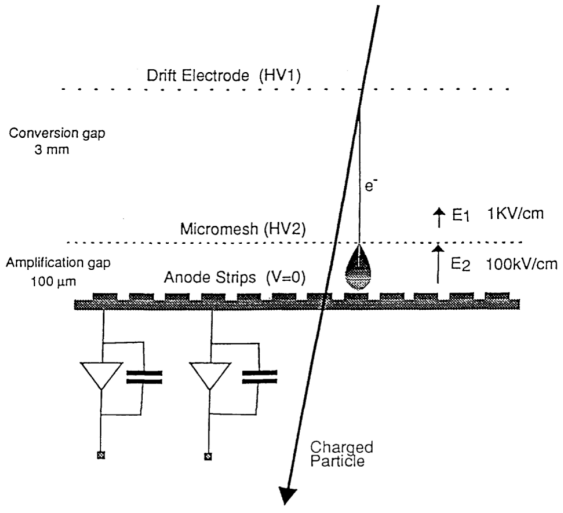


Fig. 1 – Sketch of a charged particle passing through the Micromegas detector I. Giomataris *et al.* [2].

The setup used in this research consists of eight consecutive Micromegas detector planes, with rectangular dimensions of 2200 mm in width and 3500 mm in

height. Each plane has 8800 strips with a pitch of 0.4 mm between them. To determine the azimuthal angle of the tracks, two layers have their strips inclined at 1.5° relative to the x-axis, and two others with -1.5° inclination. Figure 2 shows a 3D representation of the track.

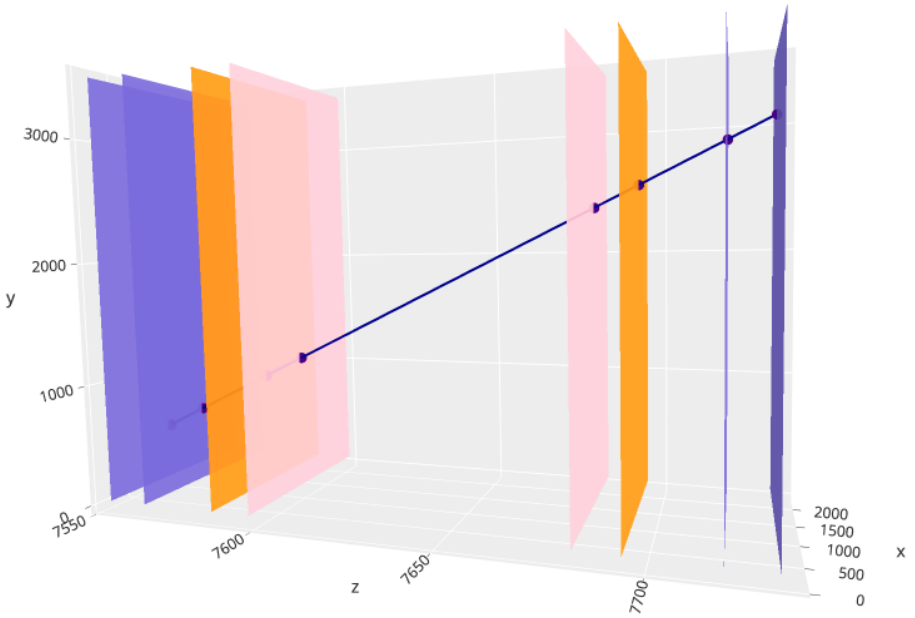


Fig. 2 – Muon track intersecting eight consecutive Micromegas detector planes [1].

The simulation of one million muon tracks and the responses of the eight Micromegas detector planes were generated using the configuration described above. The simulated data contains strip number, plane number, and the $\Delta\theta$ value, and was interpreted using the `ROOT` framework [3].

For the CNN model to take advantage of the structure of the data, the muon track was represented as a binary matrix $A_{8800\times 8}$, where $A_{i,j} = 1$ only if the strip number i from plane j registered a hit. This representation allows for the use of convolutional approaches to neural networks that can be easily transferred to FPGAs using the `hls4ml` tool [4].

In addition to the CNN approach, a recurrent neural network approach was also explored for this research. The binary matrix representation of the muon track was modified to fit the input requirements of the recurrent neural network. Instead of representing the strip number and plane number as pixels in an image, they were embedded as features in the input sequence of the network. This modified input representation allows the recurrent neural network to capture the temporal dependencies

Table 1
Convolutional Neural Network Architecture

Type	Shape	Parameters
InputLayer	[(None, 8800, 8, 1)]	0
BatchNormalization	(None, 8800, 8, 1)	4
Conv2D	(None, 8800, 8, 8)	80
MaxPooling2D	(None, 2200, 8, 8)	0
ReLU	(None, 2200, 8, 8)	0
Conv2D	(None, 2200, 8, 16)	1168
MaxPooling2D	(None, 550, 8, 16)	0
ReLU	(None, 550, 8, 16)	0
Conv2D	(None, 548, 6, 32)	4640
ReLU	(None, 548, 6, 32)	0
MaxPooling2D	(None, 34, 3, 32)	0
Flatten	(None, 3264)	0
Dropout	(None, 3264)	0
Dense	(None, 1)	3265
Rescaling	(None, 1)	0
Total params: 9,157		
Trainable params: 9,155		
Non-trainable params: 2		

of the input data and potentially improve the accuracy of the muon trigger system.

3. NEURAL NETWORK ARCHITECTURE

The CNN model is designed to take in the high-dimensional input data from the particle detector and output an estimate of the particle's angle of incidence. The architecture of the CNN model is summarized in Table 1.

The model consists of three convolutional blocks, each consisting of a convolutional layer, max pooling layer, and ReLU activation [5]. The input is first batch normalized [6] before being fed into the convolutional layers. The output of the final convolutional block is then flattened and passed through a dropout layer [7] before being connected to a dense linear layer with a single output. The output value is then rescaled to cover the interval of angles representative of our data $[-0.0326\,\text{rad}\,;0.0326\,\text{rad}]$.

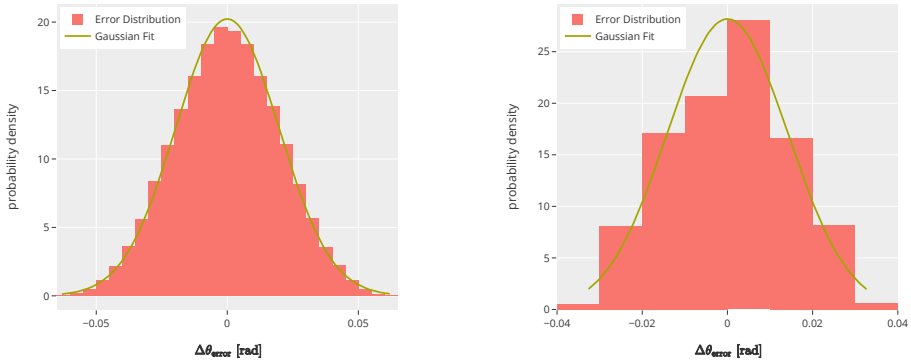
The RNN model is designed to take in the sequence of low-level detector readings and output an estimate of the particle's angle of incidence. The architecture of the RNN model is summarized in Table 2. It consists of an LSTM layer [8], followed by two dense layers with ReLU activation functions [5]. Likewise to the CNN

Table 2
Recurrent Neural Network Architecture

Type	Shape	Parameters
InputLayer	[(None, 16, 1)]	0
LSTM	(None, 128)	66560
Dense	(None, 32)	4128
Dense	(None, 1)	33
Rescaling	(None, 1)	0
Total params: 70,721		
Trainable params: 70,721		
Non-trainable params: 0		

approach, the output of the final dense layer is a single scalar value which is being rescaled to cover the $[-0.0326\,\text{rad}\,;0.0326\,\text{rad}]$ interval of angles.

For this recurrent model 70721 trainable parameters are required, an order of magnitude more than the CNN. 80% of the one million simulated events [9] were involved in the training process using the Adam optimizer [10] with default parameters in TensorFlow [11]. The mean squared error loss converged after 10 epochs of training with a batch size of 500.



(a) Distribution of test $\Delta\theta$ absolute prediction error for the CNN model

(b) Distribution of test $\Delta\theta$ absolute prediction error for the RNN model

Fig. 3 – The distributions of test $\Delta\theta$ absolute prediction error for the CNN and RNN model. A Gaussian fit is performed in order to compute the standard deviation of the distribution.

4. RESULTS

The performance of the CNN and RNN models was evaluated in terms of the angular resolution, which was measured as the standard deviation of the Gaussian fit of the distribution of test $\Delta\theta$ absolute prediction error.

As shown in Figure 3, the distribution of test $\Delta\theta$ absolute prediction error for the RNN model (Figure 3b) had a lower standard deviation (14 mrad) compared to that of the CNN model (Figure 3a), which had a standard deviation of 19 mrad.

This result shows that the RNN model was able to predict the angle of incidence of the particles with higher accuracy than the CNN.

It is worth noting that the RNN model required a significantly higher number of parameters than the CNN, with 70,721 parameters for the RNN and only 9,157 parameters for the CNN. This difference in the number of parameters suggests that the RNN model may be more prone to overfitting and require more training data than the CNN.

5. CONCLUSIONS

In this study, we have compared the performance of two different neural network architectures, Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs), for a muon trigger algorithm based on Micromegas detectors.

We used the same dataset and training procedure for both approaches to ensure a fair comparison. Our results indicate that while the CNN approach is more efficient in terms of the number of parameters, the RNN approach can produce better angular resolution albeit at a higher computational cost.

Future work could involve exploring different neural network architectures, such as hybrid architectures combining CNN and RNN components, for the muon trigger problem.

Additionally, the performance of these neural network architectures could be evaluated on real data from experiments, and further improvements could be made to optimize the neural network models for FPGA deployment.

Overall, this study highlights the potential of machine learning approaches for improving particle detection and data analysis in High Energy Particle Physics.

Acknowledgements. This study was supported by PN23210104 and ATLAS CERN-RO projects.

REFERENCES

1. I.-M. Dinu, I. S. Trandafir, C. Alexa, "*A Machine Learning Based Muon Trigger Algorithm for an Assembly of Micromegas Detectors*", Romanian Journal of Physics **67**, 401 (2022)
2. Giomataris, Ioanis and Rebourgeard, P C and Robert, J P and Charpak, Georges, "*Micromegas: a high-granularity position-sensitive gaseous detector for high particle-flux environments*", Nucl. Instrum. Methods Phys. Res., A, **376** (1996) 29-35, 10.1016/0168-9002(96)00175-1 (1995)
3. R. Brun, F. Rademakers, "*ROOT - An Object Oriented Data Analysis Framework*", Nucl. Inst. & Meth. in Phys. Res. A **389**, 81-86 (1997).
4. J. Duarte et al., "*Fast inference of deep neural networks in FPGAs for particle physics*", JINST **13** P07027 (2018), arXiv:1804.06913.
5. A. F. Agarap "*Deep learning using rectified linear units (relu)*". ArXiv Preprint ArXiv:1803.08375 (2018)
6. S. Ioffe, C. Szegedy, "*Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift*", ArXiv Preprint arXiv:1502.03167 (2015)
7. N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, "*Dropout: a simple way to prevent neural networks from overfitting*", Journal of Machine Learning Research **15**(1) 1929-1958, (2014)
8. Klaus Greff, Rupesh Kumar Srivastava, Jan Koutník, Bas R. Steunebrink, Jürgen Schmidhuber "*LSTM: A Search Space Odyssey*", EEE Transactions on Neural Networks and Learning Systems, **28**(10), 2222-2232 (2017).
9. A. Wang, "*Quick simulation for the MMTP trigger*", GitHub Repository, `https://github.com/annmwang/oct_sim`
10. D. P. Kingma, J. Ba, "*Adam: A Method for Stochastic Optimization*", 3rd International Conference for Learning Representations, arxiv:1412.6980, (2015)
11. Abadi, Martin and Barham, Paul and Chen, Jianmin and Chen, Zhifeng and Davis, Andy and Dean, Jeffrey and Devin, Matthieu and Ghemawat, Sanjay and Irving, Geoffrey and Isard, Michael and others "*TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*, `https://www.tensorflow.org` (2015)