A Machine Learning Method for calorimeter signal processing in sPHENIX

M.Potekhin and T.Rinn for sPHENIX Collaboration

Brookhaven National Laboratory, Upton, NY11973, USA

E-mail: potekhin@bnl.gov

Abstract. sPHENIX is a new precision high energy nuclear physics experiment at RHIC and features both electromagnetic and hadronic calorimeters. The initial stage of the calorimeter data reconstruction is extraction of signal features such as the amplitude, timing of the signal peak and the pedestal, for each calorimeter channel. The baseline technique used for that is fitting the signal waveforms with a parameterized function which optimally represents the signal shape. Due to the calorimeter channel count, this procedure may consume a non-trivial fraction of the total reconstruction time in a given event. We present an alternative technique for signal feature extraction, based on a Machine Learning algorithm, as well as characterization of its performance and a few approaches for deployment in the production environment.

1. Introduction

The Relativistic Heavy Ion Collider (RHIC) has enabled many discoveries into the nature of the strong force. One of the most impactful discoveries is that of the formation of a deconfined state of nuclear matter called the Quark-Gluon Plasma which can be formed in relativistic heavy ion collisions. The sPHENIX experiment at RHIC[1], which will start taking data in 2023, is a state of the art detector designed to perform precision measurements of the Quark-Gluon Plasma characteristics complementary to those coming from the LHC experiments. One of the principal components of the sPHENIX detector is its Electromagnetic Calorimeter (EMCal)[2]. It features a novel design based on scintillating fibers embedded in a mix of tungsten powder and epoxy with optical readout utilizing silicon photomultipliers (SiPM). The total EMCal channel count is 24,576. The performance of the EMCal was evaluated using prototype devices, the most recent of which was tested in 2018[3] at the Fermi National Accelerator Laboratory's test beam facility. The EMCal energy resolution (measured in percent) was found to be around $\sigma(E)/(E) = 3.5 \oplus 13.3/\sqrt{E}$ utilizing a position-dependent correction to compensate for non-uniformities in response, determined using an external hodoscope detector.

2. Extracting the characteristics of the EMCal signal

2.1. The Signal

In the EMCal prototype, the silicon photomultiplier signals are processed using 14-bit digitizers operating at the nominal frequency of 60MHz, and digitized signals were recorded in 32 time samples. A typical signal shape recorded in the test beam experiment is shown in the left panel of figure 1. As the digitizer frequency was fixed in this exercise, the sample number (the horizontal axis) corresponds to the time axis with 1 sample being equivalent to approximately 16.7ns. All

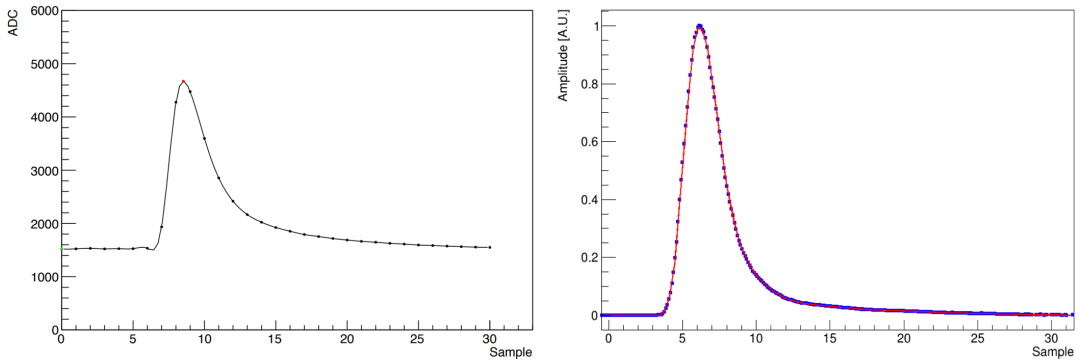

Figure 1. On the left, a typical example of the digitized signal shape observed with the EMCal prototype at the Fermilab Test Beam Facility. The right panel shows the extracted "signal template".

subsequent calculations discussed in this proceedings were performed using thus defined units of time.

By design the amplitude of the signal (i.e. ADC counts above the pedestal) scales linearly with the light output in the calorimeter tower, thus providing a linear measure of the energy deposition. Determination of the time when the signal reaches its peak value is necessary for rejection of signals not associated with a real beam-beam crossing. Therefore, three parameters of the signal must be determined as the initial step in the tower reconstruction – the pedestal, the absolute peak value and the time when the signal reaches the peak value. Since the signal is digitized in discrete time bins, it is necessary to utilize numerical calculations (such as fitting) to precisely infer the underlying signal's amplitude and timing characteristics. The data representing an individual signal is processed as an array (a vector) of 32 elements.

2.2. The Template Fitting Method

The raw SiPM signals are processed through a preamplifier and shaped before being driven into the digitizer. This electronics chain largely drives the shape of the signal waveform, which does not follow a trivial analytical form. To address this, the following "template" method was developed for the fitting procedure:

- A large number of signal events were recorded during the test beam experiment. Due to the aforementioned linear scaling of the signal amplitude vs the energy in the corresponding tower, it is possible to combine and normalize signals (after pedestal subtraction) along both time and amplitude axes in a way that effectively provides oversampling, i.e. the average normalized signal shape (the "template") can be tabulated with a much greater resolution than the 32 original time bins. For example, the signal template used in this study featured 486 data points. A typical template shape is presented in the right hand panel of figure 1. For convenience, normalization is done in a way such that the pedestal-subtracted peak amplitude is 1.0.
- Using the template produced as described above, a fit function was defined based on linear interpolation of the tabulated template data. Due to oversampling, the linear interpolation was quite precise and shown to match other interpolation methods such as spline, to within six decimal places precision. In the resulting function the pedestal was treated a constant offset term, while the amplitude of the signal peak and its coordinate on the time axis comprised the remaining two fit parameters. The fit was performed utilizing the GSL Multidimensional minimizer implemented in the ROOT[4] software package.

This fitting algorithm was benchmarked for performance, and it was estimated that the signal feature extraction done with this method takes 1.4s on average for a sPHENIX Au+Au event with nominal channel occupancy. This is just the first, most basic step in processing the calorimeter data, and yet it takes a non-negligible fraction of the overall CPU budget available for full event reconstruction (approx. 20s). There is therefore a motivation to consider alternative methods of signal feature extraction which would hopefully be faster. We investigated application of Machine Learning (ML) techniques for this purpose. The study was done in three phases:

- Experimentation with simulated signals, in order to ascertain applicability of this approach in controlled conditions and get an initial insight into its accuracy and performance.
- Using the test beam data to train and evaluate ML models and collect performance metrics.
- Investigating the available deployment options to be used in the production environment.

3. Using a Neural Network for the signal parameter extraction

A simple Neural Network (NN) was created for the signal parameter extraction. We used the Keras[5] open-source software library because of its easy-to-learn Python interface and rich functionality. In this application, the network contains one hidden layer with the same dimension as the input layer, which is 32 as defined by the input data. The output layer contains three neurons; all layers are of the type "Dense". This is illustrated in figure 2.

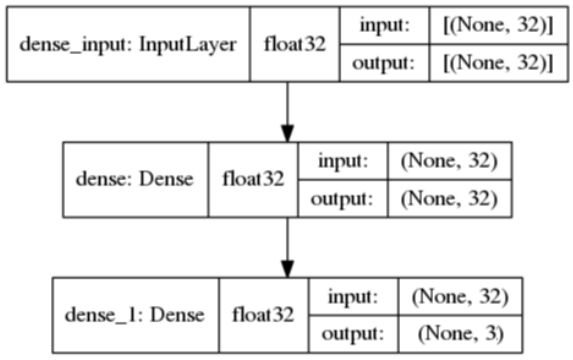

Figure 2. Diagram of the Neural Network for signal feature extraction, constructed in Keras.

The Neural Network is implemented with just a few lines of Python code utilizing Keras:

```
model = Sequential()
model.add(Dense(32, input_dim=L, activation='relu'))
model.add(Dense(3, activation='linear'))
model.compile(loss='mse', optimizer='adam', metrics=['accuracy'])
```

The parameters given in the code sample above are one example of a few sets that were investigated. To get a general idea of this model characteristics and performance, simulated signals were used first. The modified Landau distribution function[6] was determined to be a close match to the shape of the *leading edge* of the EMCal signals recorded in the test beam experiment. A few batches of simulated signals were generated, covering a range of amplitudes, peak times, noise levels and pedestals. Parameters of the simulated signals effectively represented the "Monte Carlo Truth", which was used to train the Neural Network. The Keras software [5] was used for both model training and inference. Inference was tested with varying batch sizes of the input data. The following results were obtained

- Relative accuracy of inference for both the peak amplitude and time was found to be $O(10^{-3})$, which is quite satisfactory considering the estimated energy resolution of the calorimeter

- Inference performance in terms of speed was found to depend on the batch size (as was expected), with larger batches resulting in a better per-vector performance. When using batch sizes of a few tens of thousand of input vectors the inference time per individual input vector was as low as $3\mu s$, which was roughly 50 times faster than the conventional fit.

These results provided the motivation to progress to training models with the test beam data.

4. Implementation of a ML model with the test beam data

During the test beam experiment with the EMCal prototype [3] samples of events were recorded with a wide range of electron beam energies: $2, 3, 4, 6, 8, 16, 20$ and 28 GeV. Datasets were processed in the ROOT format and then imported into Python applications using the *uproot* package[7]. Signals contained in the recorded data were processed using the template fit procedure, as described in section 2.2. The arrays of the input data combined with the fit results formed the training sample for the Neural Network constructed as described in Section 3. In other words, the Neural Network was trained to reproduce the conventional (template-based) fitting procedure, with expectation of better performance (speed) in the inference stage, but no superior precision or other characteristics.

A few training datasets (representing different parts of the dynamic range) were used, and a few models were thus created. The trained models were preserved, and inference performed on data samples distinct from the training samples. Residuals of the ML fit vs the conventional (template-based) fit were calculated. The training process was optimized to minimize the residuals, using the following procedures:

- Input datasets were normalized, such that numerical values of the signal amplitude parameters (both pedestal and the absolute peak value) and the time coordinate of the peak were on a close (same order of magnitude) scale. This was achieved by reversably scaling down the ADC count values of the input by a fixed factor of 1000.
- Composition of the training sample was designed to cover all of the dynamic range of the signals in the available data, from 2 to 28GeV, by mixing data from the data samples collected at the respective beam energies. A number of mixing ratios were investigated, and best performing candidates were picked for further study.

5. Deployment and Integration

As noted above, construction, training and optimization of the ML models in this study was performed in the Python/Keras environment, which simplified the development process and allowed using efficient tools like Jupyter notebooks. One the other hand, the sPHENIX computing environment is based on C++ so there is an issue of integration of the Keras-derived models into C++ applications in order to perform the inference. The following deployment options were explored:

- An inference service, based on the Django platform[8]. The service receives input from C++ or other clients via HTTP, performs inference and ships back the result as a HTTP response.
- Another HTTP-based approach that was tested comprised a microservice deployed on worker nodes, based on the Gunicorn server software[9]
- ONNX – the Open Neural Network Exchange. The ONNX-based runtime libraries[10] provide remarkable cross-platform compatibility.

Testing of both serverized versions (Django and Gunicorn) of inference software was successful, however the network latency and bandwidth did introduce a degree of performance penalty, depending on the inference batch size. Ultimately, the ONNX-based deployment model proved

the most flexible and straightforward. The ONNX runtime libraries are available in precompiled form for download for a number of platforms, and can also be built from source with reasonable effort. There are build versions optimized for size, and in this case these runtime libraries are remarkably compact at less than 20MB in size. Three steps were necessary in order to utilize ONNX in the context of this project:

- Conversion of the Keras models to ONNX format, with an existing Python package.
- A software adapter layer, converting ROOT data structures into tensor format appropriate for the ONNX API.
- Augmenting the preexisting fitting function with the additional capability to use inference to extract the signal parameters.

All these steps were completed and the code was incorporated into the sPHENIX software suite. This allowed us to proceed to the testing stage in a realistic production environment, using the raw test beam data.

6. Performance of ML inference vis-a-vis conventional fit

6.1. Inference speed

Similar to the results obtained with the Python/Keras software, the speed with which inference is performed within the sPHENIX environment (compiled C++ code) depends on the size of the data batches being processed. For example, if all of the 24,576 calorimeter channels are included in a batch for an inference run, the inference takes 12ms. Thus, processing the raw data for one complete sPHENIX EMCal event to extract signal parameters in every channel takes only 12ms compared to approximately 1.4 seconds for the template fitting method. This is an improvement of approximately two orders of magnitude over the conventional fit method previously used. Of course, to achieve this level of performance the processing chain must be designed correspondingly, to make such batching of inputs possible.

6.2. Effect on energy resolution

Energy resolution of the EMCal prototype was calculated based on the test beam data, as described in [3]. The measured energy comes from the summation of a 5×5 cluster of calorimeter towers, centered around the tower hit by the incident particle (determined using an external hodoscope detector). In order to mitigate effects caused by non-uniformities in the detector response, the particles were selected as to be incident on the center of the tower. This results in a small improvment in the measured resolution relative to that quoted in reference [3]. The waveforms from each channel are processed independently, using both conventional (template-based) fit and ML inference. The extracted ADC values are summed, and a calibration procedure is performed to match the summed ADC counts to the nominal beam energy. Energy resolution values can then be calculated for each incident energy point, and a fit with an appropriate function can be performed. The graphic representing the data points and the fit is shown in figure 3, for both conventional and ML-based methods of waveform processing. Due to characteristics of the beam line there is a 2% energy spread among incoming particles. This is accounted for by the inclusion of a constant 2% term to the fit of the resolution function, as is indicated in the expression found in figure 3. It is evident from these studies that the ML inference implemented is able to accurately reproduce the results from the template fittng method, which demonstrates the validity of the machine learning approach in the EMCal signal processing.

7. Conclusion and plans

In this study we demonstrated a practical example of applying a simple neural network to signal processing, resulting in a substantial gain in speed while retaining the accuracy when compared

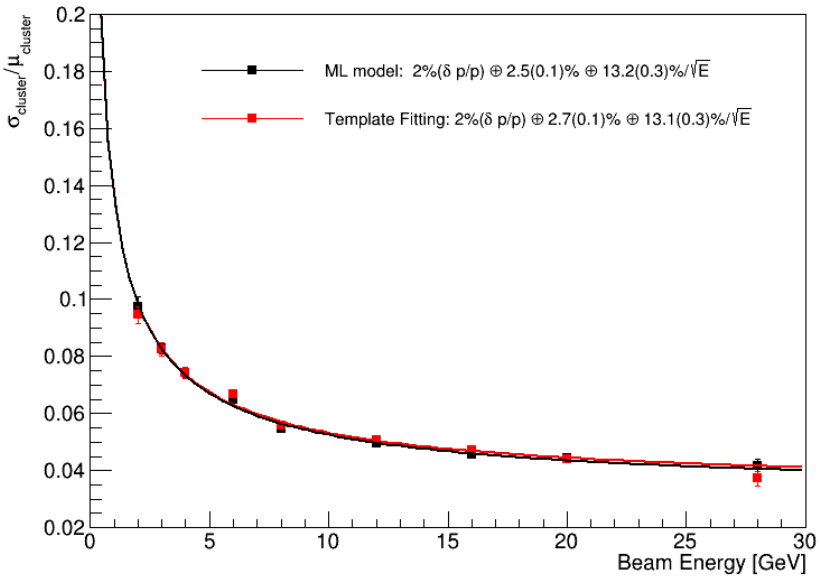

Figure 3. Comparison of energy resolution calculated with the test beam data, obtained with template fitting and the ML (ONNX) model

to other methods. The Keras Python package was used as the tool to create and evaluate ML models in this study. A few deployment options for the inference software were evaluated. The ONNX runtime library was chosen due to its small footprint and good performance characteristics. The results are promising and create potential for extending applications of Machine Learning in the sPHENIX calorimeter data processing chain.

Acknowledgments

The authors are grateful to D.Lis for the development of the concept of the template-based signal fitting. This work has been supported by the Office of Science of the Department of Energy under contract No. DE-SC0012704.

References

[1] C.Dean et al., "The sPHENIX experiment at RHIC", *Proceedings of Science*, 2021, ICHEP2020

[2] C.Woody at al., "Design Studies of the Calorimeter Systems for the sPHENIX Experiment at RHIC and Future Upgrade Plans", *J. Phys.: Conf. Ser.*, **587** (2015), 012054

[3] C. A. Aidala et al., "Design and Beam Test Results for the 2-D Projective sPHENIX Electromagnetic Calorimeter Prototype", *IEEE Transactions on Nuclear Science*, vol. 68, no. 2, pp. 173-181, Feb. 2021, doi: 10.1109/TNS.2020.3034643.

[4] R.Brun and F.Rademakers, "ROOT - An Object Oriented Data Analysis Framework", *Nucl. Inst. & Meth. in Phys. Res.* **A389** (1997) 81-86.

[5] `https://keras.io/`

[6] S. Behrens et al., Univ. of Rochester Preprint **UR-776** (1981)

[7] `https://pypi.org/project/uproot/`

[8] `https://www.djangoproject.com/`

[9] `https://gunicorn.org/`

[10] `https://onnx.ai/`