ISBN: 978-3-95450-214-1 ISSN: 2673-5490 doi:10.18429/JACoW-IPAC2021-THPAB181

AI-ML DEVELOPMENTS FOR THE ATLAS ION LINAC FACILITY*

B. Mustapha, J. Martinez, B. Blomberg, C. Peters, C. Dickerson
Physics Division, Argonne National Laboratory, Lemont, USA

Abstract

ATLAS is a DOE/NP User Facility for the study of low-energy nuclear physics with heavy ions. It operates ~6000 hours per year. In addition to delivering any stable beam from proton to uranium, the facility also provides radioactive beams from the CARIBU source or via the in-flight radioactive ion separator, RAISOR. The facility uses 3 ion sources and services 6 target areas at energies from ~1-15 MeV/u. To accommodate the large number and variety of approved experiments, ATLAS reconfigures once or twice per week over 40 weeks of operation per year. The start-up time varies from ~12-48 hours depending on the complexity of the tuning, which will increase with the upcoming Multi-User Upgrade to deliver beams to two experimental stations simultaneously. DOE/NP has recently approved a project to use AI/ML to support ATLAS operations. The project aim is to significantly reduce the accelerator tuning time and improve machine performance by developing and deploying artificial intelligence methods. These improvements will increase the scientific throughput of the facility and the quality of the data collected. Our recent developments and plans will be presented and discussed.

INTRODUCTION

The use of machine learning (ML) and artificial intelligence (AL) has the potential of significantly reducing the time needed to tune an accelerator. This is very important for ATLAS [1] due to the frequency of machine tuning, where a new beam is tuned once or twice a week. In addition to the existing multiple sub-systems of the ATLAS facility, such as CARIBU [2] and RAISOR [3], the upcoming Multi-User Upgrade [4] will further complicate machine operations. The timely development and implementation of AI/ML techniques will be very beneficial to the whole facility. By reducing the time for beam tuning, more beam time will be available to help relieve the over-booked experimental nuclear physics program at ATLAS. In addition to beam tuning, AI/ML models can be used to improve beam quality with the installation of new diagnostics and real-time data acquisition. These improvements have the potential of increasing the scientific throughput of the facility and the quality of the data collected.

To support these developments, DOE/NP has recently approved a project to use AI/ML for ATLAS operations. Following a description of the project objectives and future plans, the results from some recent developments will be presented and discussed.

* This work was supported by the U.S. Department of Energy, under Contract No. DE-AC02-06CH11357. This research used the ATLAS facility, which is a DOE Office of Nuclear Physics User Facility. The authors would like to also thank Nathaniel Bowden of Wheaton College, Illinois.

PROJECT OBJECTIVES & PLANS

The main project goal is to use AI/ML techniques to streamline beam tuning and help improve machine performance. The project objectives are three-fold:

- Establish data collection, organization and classification, towards a fully automatic and electronic data collection for both machine and beam data
- Develop an online tuning model to optimize operations, shorten beam tuning time and make more beam time available for the experimental program
- Develop a virtual machine model to enhance our understanding of the machine behavior, improve machine performance and optimize particular aspects and help develop new operating modes

Data Collection

Figure 1 below shows a sample of historic machine settings and beam parameters data recorded at ATLAS. For every beam tune, this information is stored and can be re-used or re-scaled to start a new tune for a different ion beam. Currently, some of the beam data is collected in paper only form and had to be entered manually in the shown spreadsheet. As part of this project, we plan to move to fully automatic and electronic data collection.

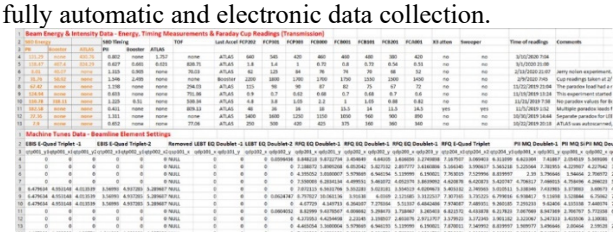

Figure 1: A sample of historic machine settings and beam data for the ATLAS linac.

Online Tuning Model

The main goal of the online tuning model is to optimize operations and shorten the beam tuning time in order to make more beam time available for the experimental program. A first version of this model consists of an initial tune model based on the existing machine tunes database and a set of optimization procedures and feedback loops fed by online data. The model can be further enhanced with new data from additional real and virtual beam diagnostics.

Virtual Machine Model

The main goal of the virtual machine model is to enhance our understanding of the machine behaviour in order to improve the performance and optimize particular aspects and new operating modes. It will be particularly useful for multi-beam transport and acceleration as part of the upcoming ATLAS multi-user upgrade, as well as for high-intensity beams. Since full beam physics models, which

ISBN: 978-3-95450-214-1 ISSN: 2673-5490 doi:10.18429/JACoW-IPAC2021-THPAB181

usually include particle tracking in 3D fields, are slow and not very useful to support online accelerator operations, we are developing surrogate AI models for different sections of the linac. A surrogate model can be trained on beam simulation data to reliably reproduce the physics results in very short time, then be enhanced with experimental data. A preliminary surrogate model developed for the ATLAS RFQ is presented in the following section.

SURROGATE MODELS FOR ATLAS RFQ

We have developed a surrogate model for the ATLAS RFQ, which is the first accelerating section of the linac and often the most time consuming when simulated with 3D fields and particle tracking using the beam dynamics code TRACK. The goal is to be able to reliably reproduce the physics results in the shortest possible time. To generate the data to train the model, we performed 7000 TRACK simulations. Both the input and output beam parameters were recorded for each simulation. The main focus was on beam transmission and output Twiss parameters as functions of input beam parameters which include the beam emittances and input Twiss parameters. At first, a neural network with three hidden layers was used. Although it was promising the error was not satisfactory (objective was not optimally met). Then we tried a neural network with two hidden layers and two residual blocks directly connecting the input to the output [5], and the results were significantly improved. A schematic of the model is shown in Fig. 2 below.

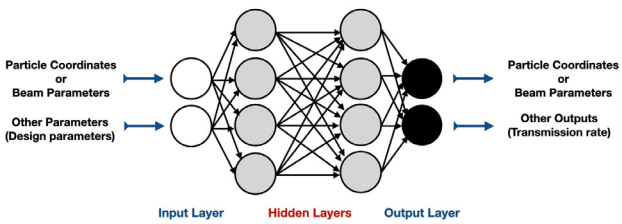

Figure 2: Schematic of a neural network surrogate model for the ATLAS RFQ.

Two models were developed using the same data set. The first was trained only to reproduce the beam transmission rate, while transverse Twiss parameters were added to the second one. Figure 3 shows the loss function/error and convergence for both models. We can clearly see the deterioration in the model performance as the dimension of the problem increased. This also means that by increasing the size of the data set, we should be able to recover the original model accuracy. Since the same data set was used in both cases, the degrees of freedom for the model have decreased by increasing the dimensionality of the problem.

In addition to more data, the accuracy of the model can also be improved by exploring more advanced and deeper NN architectures, which we didn't pursue at this time because the results were satisfactory as shown in Fig. 4, comparing the results predicted by the surrogate model to the actual results from the physics model. The agreement is very good, it's consistent with the comparison of two beam dynamics codes, TRACK vs. IMPACT for example [6]. Therefore, the surrogate model can be considered reliable and capable of reproducing the physical results, with the big advantage of being ~ 30,000 faster than the 3D model in this case. This is exactly what is needed for an online machine model to be useful for fast optimization during beam tuning and routine operations. The speed-up factor is from 90 seconds for tracking 10,000 particles through the RFQ to 0.003 second using the RFQ surrogate model.

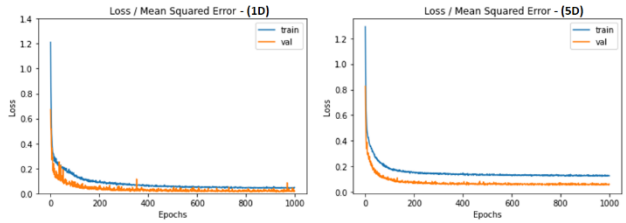

Figure 3: Loss function, in this case the mean squared error ($\chi 2$), showing the convergence (training to actual value) for two models. The one on the left was trained only for beam transmission rate (1D), while the one on the right also included the transverse Twiss parameters of the beam (5D).

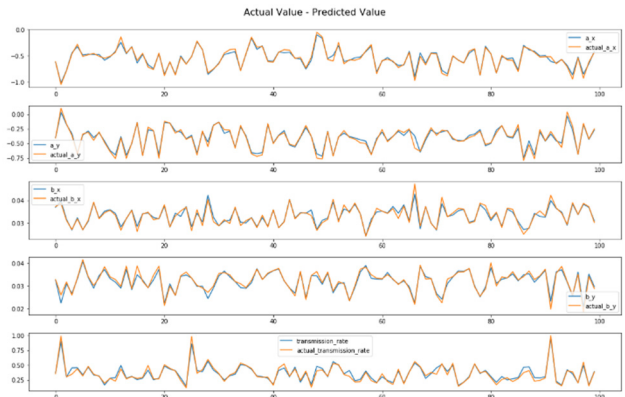

Figure 4: Comparison of the RFQ surrogate to the physical model results for 100 simulations. Included are output Twiss parameters and beam transmission rate.

MODELS FOR PARTICLE TRACKING

The stated goal of creating an ML surrogate model for particle tracking is divided into two problems: classification and regression. For the classification problem, a model is trained to predict whether particles are accepted into the RFQ by supplying samples of input particle coordinates and their integer acceptance flag. The regression problem refers to predicting output coordinates from input coordinates, requiring training on pairs of input and output coordinates for already accepted particles. The two models can then chain into one process, first predicting acceptance or loss, then finding the output coordinates if the particle was accepted. The task starts by generating the data.

Data Generation

Particle coordinate data, namely, the input and output coordinates of particles, as well as an integer flag indicating acceptance (1) or loss (0), was generated for the ATLAS RFQ using the beam dynamics code TRACK. A continuous (DC) beam of 10,001 uranium-238 atoms (q = +34) was

ISBN: 978-3-95450-214-1 ISSN: 2673-5490 doi:10.18429/JACoW-IPAC2021-THPAB181

simulated as it passed through the RFQ. This was repeated for 1,000 different combinations where the beam parameters: x and y emittances and energy spread, were varied. The results of all simulation runs were aggregated into a single dataset of ~10 million particles. Of these ~ 27% were accepted. The distributions of lost and accepted particles in phase space are shown in Fig. 5.

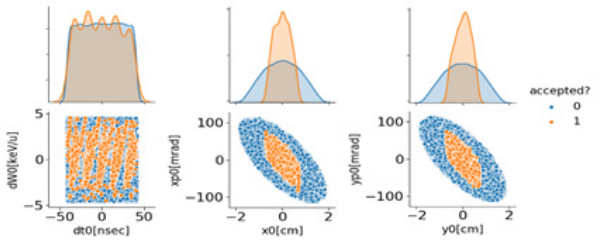

Figure 5: Distributions of accepted (orange) and lost particles (blue) in the phase-space of initial (u_0) coordinates. Top plots are probability densities for acceptance and loss with respect to dt_0, x_0, and y_0, bottom plots display the acceptance of the RFQ in the 2D phase planes.

Models & Training Process

For the classification and regression problems, we used the PYTHON libraries SCI-KIT LEARN [7] (SL) and TENSORFLOW KERAS [8, 9] (TF) to implement Machine Learning (ML) models for this study. SL provides a selection of frequently used ML algorithms which package the model, loss, and optimizer into one entity. TF allows the construction of custom Deep Learning (DL) Neural Network (NN) topologies which can be compiled with various loss functions and optimizers.

For each problem, the data was split into training, validation, and test sets. For classification, the split was in the ratio 60:20:20. Since output coordinates for lost particles cannot be defined, only data for accepted particles (2.7 million) was used for regression and was split in the ratio 70:10:20. The training split was presented for the model to learn, while the validation split was used to track model performance during training. Final evaluation and comparison of the models was performed with the test split. Trained models were saved for later use in applications.

Classification Models for Particle Acceptance

Figure 6 compares the performance of different ML and DL classification model architectures based on the Accuracy, Precision and Recall criteria defined below:

$$Accuracy = (\text{true predictions})/(total\ samples)$$
$$Precision = (true\ positives)/(predicted\ positives)$$
$$Recall = (true\ positives)/(real\ positives)$$

The first two ML models, 'freq' and 'strat' are 'most frequent' and 'random stratified' guessing strategies, respectively. They provide a baseline for comparing model performance. LR, PAC, Ridge, and LDA are linear models, while the rest are nonlinear models. The DL models to the right (TFCn) are NN architectures. TFC1, 2, and 3 include L2 regularization and all models use dropout layers. Binary cross-entropy was chosen as a loss function and was minimized using the Adam [10] optimizer for the DL classification models. The figure clearly shows that TFC0, TFC2 and Tree are the best models for particle acceptance.

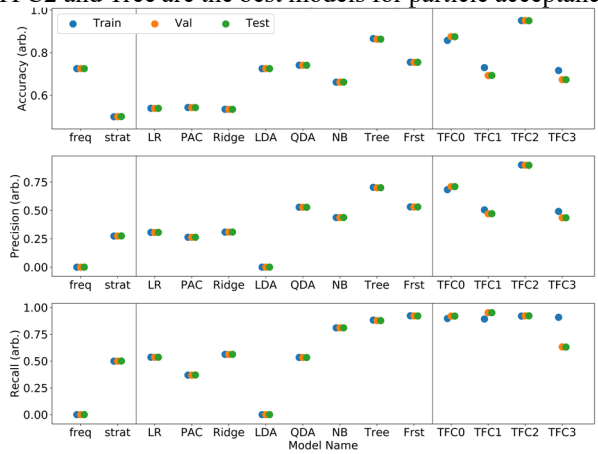

Figure 6: Comparison of classification models using Accuracy (top), Precision (middle), and Recall (bottom) plots. Higher scores indicate better performance. Partitions delineate dummy guessers, ML, DL models, respectively.

Regression Models for Output Coordinates

Similarly, a variety of models were used for the regression problem. As for the classification case, the 'Dummy' regressor provides a baseline for model performance. OLSR Ridge are linear models and the rest are nonlinear. The DL models TF0, 1, 3 and 4, share the same architecture as those for the classification problem, except that the last layer was altered to obtain six outputs. A new architecture (TF2) based on beam physics was added, where the transformation is decomposed into a linear and nonlinear terms.

Figure 7 compares the performance of the different ML and DL models based on the average Mean Square Error (aMSE) and average Mean Absolute Error (aMAE). These metrics were evaluated on the training, validation, and test data sets. The first two were used to indicate overfitting during the training process, because an overfit model will perform significantly better on the training split than the validation split. The value of the metrics from the test split was used for final model evaluation to select the best models for each problem as good candidates for future work.

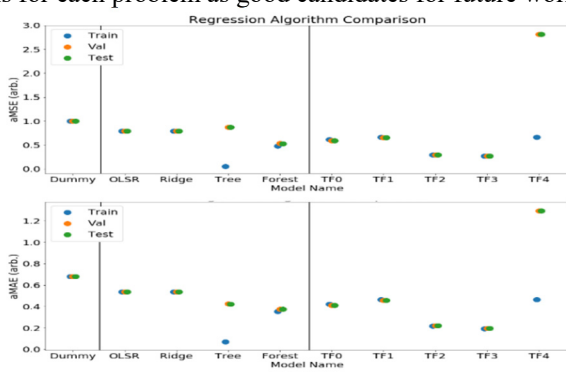

Figure 7: Comparison of regression models using the error metrics aMSE and aMAE. Smaller errors indicate better performance. Partitions delineate the dummy regressor, ML, and DL models, respectively.

ISBN: 978-3-95450-214-1 ISSN: 2673-5490 doi:10.18429/JACoW-IPAC2021-THPAB181

REFERENCES

[1] P. N. Ostroumov *et al.*, "Completion of Efficiency and Intensity Upgrade of the ATLAS Facility", in *Proc. LINAC'14*, Geneva, Switzerland, Aug-Sep. 2014, paper TUPP005, pp. 449-451.

[2] G. Savard *et al.*, "Radioactive beams from gas catchers: The CARIBU facility", *Nucl. Instrum. Meth. B*, vol. 266, pp. 4086-4091, Oct. 2008. doi:10.1016/j.nimb.2008.05.091

[3] B. Mustapha *et al.*, "An In-Flight Radioactive Ion Separator Design for the ATLAS Facility", in *Proc. LINAC'14*, Geneva, Switzerland, Aug-Sep. 2014, paper TUPP004, pp. 446-448.

[4] B. Mustapha *et al.*, "The ATLAS multi-user upgrade and potential applications", *J. Instrum.*, vol. 12, p. T12002, Dec. 2017. doi:10.1088/1748-0221/12/12/t12002

[5] K. He, X. Zhang, S. Ren, and J. Sun. "Deep residual learning for image recognition", Dec. 2015. arXiv:1512.03385

[6] B. Mustapha *et al.*, "RIA Beam Dynamics: Comparing TRACK to IMPACT", in *Proc. IPAC'05*, Knoxville, TN, USA, May 2005, paper TPAT029. pp. 2095-2097.

[7] A. Scheinker *et al.*, "Applying Artificial Intelligence to Accelerators", in *Proc. IPAC'18*, Vancouver, Canada, Apr.-May 2018, pp. 2925-2928. doi:10.18429/JACoW-IPAC2018-THYGBE1

[8] Y. Li *et al.*, "Genetic Algorithm enhanced by machine learning in dynamic aperture optimization", *Phys. Rev. Accel. Beams*, vol. 21, p. 054601, May 2018. doi:10.1103/physrevaccelbeams.21.054601

[9] A. Edelen *et al.*, "Machine learning for orders of magnitude speedup in multiobjective optimization of particle accelerator systems". *Phys. Rev. Accel. Beams*, vol. 23, p. 44601, Apr. 2020. doi:10.1103/physrevaccelbeams.23.044601

[10] P. Diederik and J. L. B. Kingma, "Adam: a method for stochastic optimization", Dec. 2014. arXiv:1412.6980