https://github.com/Ctahncpiac/CMS_RNN

Use of machine learning techniques to discriminate single particle clusters within the CMS silicon strip detector from various background physical and instrumental contributions.

Sacha Rejai and Stanislas Lambert
Université de Strasbourg, IPHC, 23 rue du Loess 67037 Strasbourg, France

(Dated: February 25, 2024)

In this research, we conduct a detailed comparison between a simple algorithm SimpleAlgo and Recurrent Neural Networks (RNN) for the classification of signal and background data in the context of particle physics, employing the ROOT framework for comprehensive data analysis. Our evaluation criteria include hypothesis test, ROC curve and the corresponding AUC. The comparative study reveals that while SimpleAlgo provides a straightforward and effective approach for initial data classification, RNN exhibits significantly higher performance, particularly evidenced by superior ROC curve metrics, thereby suggesting RNN's enhanced capability in complex data interpretations within simulated datasets.

DATA SIMULATION

To simulate the passage of particles through a detector, we created the *ClusterSimulator* class to emulate the response of a particle detector to MIPs (Minimum Ionizing Particle). We introduced a configurable framework that incorporates various physical and electronic factors affecting particle detection, including noise, crosstalk and signal digitization.

The *ClusterSimulator* class encapsulates the entire simulation framework via a JSON-based configuration file. Simulation parameters include digitization (b), signal cutting (r), detector width (w), thickness (t) and the noise (n).

The heart of the simulation lies in the generation of particle trajectories and the resulting charge distribution inside the detector. The initial position of each simulated particle is randomly determined across the width of the detector, while its angle of incidence is derived from a Gaussian distribution, as shown in Figure 1, simulating realistic particle interactions.

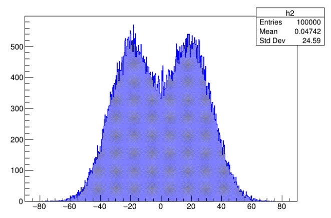

Figure 1. angle theta distribution

The simulation calculates the path of the particle through the detector and the resulting charge distribution across the detector's channels. It accounts for physical phenomena such as the angular spread due to scattering and the variation in charge deposition along the particle's path.

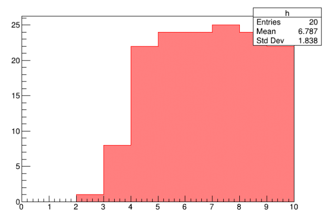

Figure 2. Example of a MIP cluster

The cluster simulator can simulate both single MIP events, as shown in Figure 2, and double MIP events (2MIP) as shown in Figure 3. This function enables us to study the detector's response to single particle and coincident particle events, so that we can compare situations later.

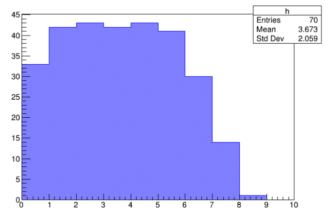

Figure 3. Example of a double MIP cluster

By adjusting the simulation parameters, users can study the impact of detector size, resolution and electronic noise on particle detection efficiency and accuracy. These elements will be used with both the simple algorithm and the RNN to provide baseline data for comparison.

RNN

The simulation framework incorporates a class for generating synthetic data sets, catering to signal (MIP) and background (2MIP) scenarios. The data generation process involves the *ClusterSimulator* class to simulate the physical interaction of particles with the detector and the resulting charge distribution. This procedure is encapsulated within the *file* class, which automates the creation of ROOT files containing the simulated data. These files are structured into two TTree objects, *tsgn* and *tbkg*, corresponding to signal and background data sets, respectively. Each TTree is filled with arrays of floating-point numbers representing the charge detected in each channel of the detector, mimicking the output from an actual particle detection event.

Firstly to distinguish signal and background events, the framework introduces a *SimpleAlgo* class, implementing basic analysis algorithms based on various criteria such as integral, charge, cluster width, signal position and ratio (charge over cluster width). This class demonstrates the application of simple heuristics in particle physics experiments.

The implementation uses ROOT's TMVA (Toolkit for Multivariate Data Analysis) and TMVAGuide [1], allowing the integration of deep learning models. The *Classification* class facilitates the creation, training, and evaluation of RNN models, automating the entire process from data pre-processing to model inference. Additionally, the integration with Keras and TensorFlow provides a flexible environment for experimenting with different neural network architectures and parameters.

The RNN models are trained using the simulated data sets, where the performance metrics ROC curve, AUC (and more from TMVAGui) are monitored to assess the model's capability to distinguish between signal and background events. The framework allows for the adjustment of training parameters, including batch size and the number of epochs, enabling fine-tuning of the model to achieve optimal performance.

RESULTS

The following results are obtain with the configuration : $\{"b" : 255, "r" : 10, "t" : 200, "w" : 100, "noise" : 10\}$ inside the config1.json file and 1000 samples of signal and background respectively. Figure 4is a hypothesis test with the width variable. Width variable is define as the size of a cluster. Figure 5 show the ROC curves obtain for different TMVA methods and for the Width hypothesis along with the area under curve (AUC). The best performance is obtain for the TMVA RNN method. We used a batch size of 100 and an epoch of 50. Finally, Figure 6 check if overtraining occured. Since the Kolmogorov-Smirnov test is above 0.01, there was no overtraining.

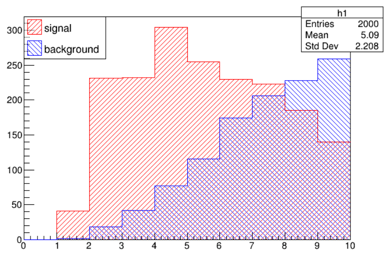

Figure 4. Hypothesis test Width

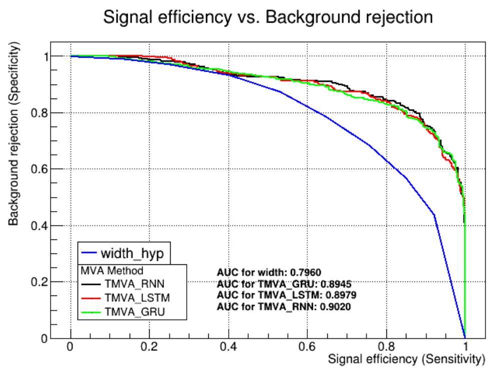

Figure 5. ROC Curves

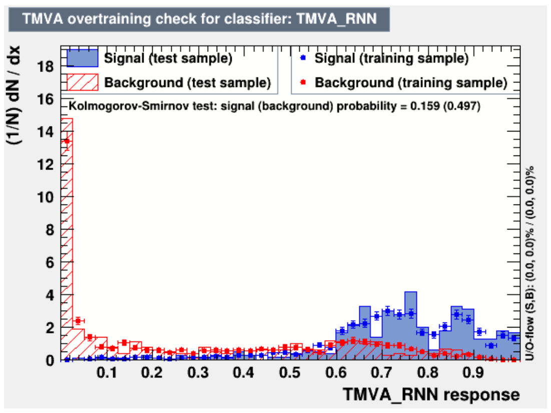

Figure 6. Overtraining check

[1] TMVA User Guide `https://root.cern.ch/download/doc/tmva/TMVAUsersGuide.pdf`