A Neural-Network-defined Gaussian Mixture Model for particle identification applied to the LHCb fixed-target programme

Dina Dervisevic (ZNJ486) & Amalie T.B.Clausen (RMB582)

05/03-2025

1 Introduction

Particle physics is a diverse and ever-evolving area of quantum physics that time again proves to be indispensable not only for science, but also for the progress and development of our society.

The field of experimental particle physics relies heavily on software capability to differentiate between the particles involved in the experiments, to sort through the large amounts of data for physicists to analyse and interpret said data [2]. This is traditionally done through computer simulations, but the article "A Neural-Network-defined Gaussian Mixture Model for particle identification applied to the LHCb fixed-target programme" (Graziani et al., 2022), explores how to optimise the process of particle identification, using machine-learning techniques as an alternative to the standard simulatory methods [1].

The purpose of this proceeding is to review the article and the methods used.

2 Methodology

2.1 Mixture Model

In the article, a particle identification (PID) classifier is used to distinguish between different particle types. To model the distributions of the PID classifiers, a mixture model approach is applied, specifically a Gaussian Mixture Model (GMM).

A GMM is a method used to determine probabilities. It is used when points in a dataset can be separated into different clusters, which have their own probability distribution. Each cluster can therefore be assigned its own Gaussian distribution, where the weight, α, of each distribution describes the proportion of points in that cluster and determines how much the distribution dominates in the overall model. Weights themselves are probabilities that represent how likely it is that a specific data point comes from a given cluster, which means that they must satisfy [4]:

$$\sum_{j=1}^{N_g} \alpha_j(\theta) = 1. \tag{1}$$

In the article, the authors model the probability distribution of a PID classifier, x, for a particle of species, p, as a GMM:

$$x_p \sim \sum_{j=1}^{N_{g,p}} \alpha_{j,p}(\underline{\theta})\mathcal{G}(x, \mu_{j,p}(\underline{\theta}), \sigma_{j,p}(\underline{\theta})), \tag{2}$$

where $\mathcal{G}$ is a Gaussian distribution with α as the weight, μ as the mean, and σ as the standard deviation, N_g is the number of Gaussian functions, and $\underline{\theta}$ represents the physical input features, such as detector occupancy. The more accurately the parameters α, μ, and σ are determined, the better the GMM can describe the data. In the article presented in this write-up, a method using neural networks is implemented to best determine the parameters.

2.2 Neural Network

The article utilises a Multi-Layer-Perceptron (MLP) neural network, to determine the parameters of the GMM. As indicated by the name, MLPs are constructed by multiple layers, which all serve to process the input data step by step so that the network can learn to recognise more complex patterns. Each layer consists of several neurons, that take the output from the previous layer, perform simple calculations on it, and then pass the results on to the next layer [3][5].

An important tool, used in neural networks, are the loss functions. Loss functions are used to measure how far the model's predictions are from the true outcomes, when training the model. The training process then tweaks the parameters of the model to minimise the loss, thus improving the model.

The loss function is, here, the negative logarithm of the likelihood function, which, when including the weights, w_i, takes the form,

$$\mathcal{L} = -\sum_{i=1}^{n_p} w_i \ln \left[\sum_{j=1}^{N_g,p} \alpha_{j,p}(\underline{\theta_i}) \mathcal{G}(x_i, \mu_{j,p}(\underline{\theta_i}), \sigma_{j,p}(\underline{\theta_i})) \right], \tag{3}$$

where the sum inside the parenthesis is the right-hand-side of Eq. (2). These weights indicate how likely it is that a training sample is an accurate representation of the specific calibration process being used to identify a particular particle. Larger values of w_i will thereby affect the loss function more and nudge it towards a more accurate model of the probability distribution.

It makes great sense to have the loss function be the negative logarithm of the likelihood function, as the negative log-likelihood (LLH), is directly used to determine the maximum likelihood estimator (MLE) 'best-fit' values, for the parameters of the likelihood function, Eq. (2). When training the model, the MLE 'best fit' values for the parameters α, μ, and σ are thus repeatedly adjusted to minimise the loss, which then ensures that the probability distribution of the PID classifier matches that of the real data. The MLP thereby takes in some experimental features, and learns the relation between these and the behaviour of the PID classifier. The output is then the MLE 'best fit' values of the parameters α, μ, and σ, which together describe the probability distribution of the PID classifier.

The article used calibration data as training data, meaning, the particle types in the dataset are already correctly identified. By using calibration data, it is therefore possible to evaluate how accurately the neural-network-defined GMM (data-based method) represents the results.

3 Conclusion

Compared with the standard simulation-based method, the article found the data-based method to do better in predicting the PID classifiers' distribution. The distributions of the PID classifiers, $DLL_{p,\pi}$ and $DLL_{p,K}$ in proton-argon collision data, can be seen in Figure 1, where $DLL_{p,\pi}$ represents how much more likely it is that the particle in question is a proton than a pion, and $DLL_{p,K}$ does so for a proton versus a kaon. The subplots in the figure are over different ranges of momentum, p, and transverse momentum p_T, with the data represented by a black template, and the simulation-based, and the data-based methods represented by a violet and blue template, respectively. In the Figure, it is clear that the data-based method outperforms the simulation-based method, and manages to more accurately predict the PID classifiers' distributions.

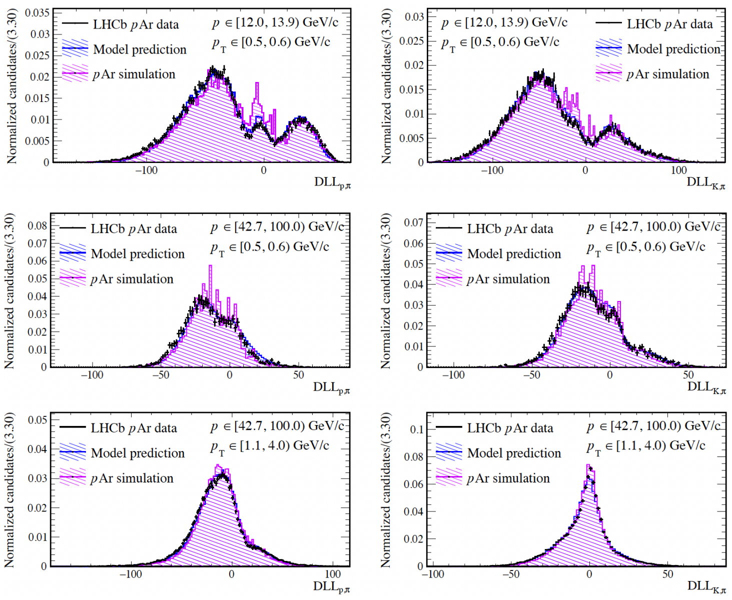

Figure 1: Comparison of the (left) $DLL_{p,\pi}$ and (right) $DLL_{p,K}$ distributions in pAr data modelled with simulation (violet) and data-based (blue) templates. Adapted from Ref. [1]

These results demonstrate the potential of advanced statistical methods in data-based modeling to estimate probability distributions more accurately than the standard simulation-based techniques.

4 References

[1] Giacomo Graziani et al. "A Neural-Network-defined Gaussian Mixture Model for particle identification applied to the LHCb fixed-target programme". In: *Journal of Instrumentation* 17.02 (Feb. 2022). arXiv:2110.10259 [hep-ex], P02018. ISSN: 1748-0221. DOI: 10.1088/1748-0221/17/02/P02018. URL: http://arxiv.org/abs/2110.10259 (visited on 02/28/2025).

[2] *The LHCb data flow — LHCb Starterkit Lessons documentation.* URL: https://lhcb.github.io/starterkit-lessons/first-analysis-steps/dataflow.html (visited on 02/28/2025).

[3] *Multi-Layer Perceptron Learning in Tensorflow.* en-US. Section: Deep Learning. URL: https://www.geeksforgeeks.org/multi-layer-perceptron-learning-in-tensorflow/ (visited on 02/28/2025).

[4] *Gaussian Mixture Model Explained.* en. URL: https://builtin.com/articles/gaussian-mixture-model (visited on 03/03/2025).

[5] Carolina Bento. *Multilayer Perceptron Explained with a Real-Life Example and Python Code: Sentiment Analysis.* en. Sept. 2021. URL: https://medium.com/towards-data-science/multilayer-perceptron-explained-with-a-real-life-example-and-python-code-sentiment-analysis-cb408ee93141 (visited on 03/03/2025).