Preliminary draft 15:56 28 August 2024

28 August 2024
nkosinathi.sangweni@cern.ch

Developing an MVA Machine Learning algorithm to Classify Prompt and Non-Prompt Muons using the CMS Weaver Framework

Author: Thobani Sangweni
Supervisor: David Walter, Sebastian Wuchterl
University of Cape Town

Keywords: Multivariate Analysis, Gradient Reversal, Prompt Lepton Identification, Domain Adversarial

Summary

This document presents the development of an extension to a Machine Learning algorithm for classifying prompt and non-prompt muons within the CMS experiment at CERN. Traditional methods for background estimation, such as the ABCD method, often suffer from inaccuracies due to variable correlations. To address this, we implemented a neural network model using domain adaptation techniques to reduce correlations to another discriminating variable often used in background estimation methods, the lepton isolation. The model was developed and trained via the CMS Weaver framework and includes two approaches: domain adversarial classification and domain adversarial regression. The model was evaluated by analysing the classification performance and the variable correlation. The correlation can be reduced with a trade-off in the classification performance, while the trade-off heavily depends on the model hyperparameters. It is shown that using machine learning for muon identification could enhance their identification efficiency while simultaneously allowing the use of robust methods for background estimation. This advancement could enhance muon identification in high-energy physics, contributing to more precise data analyses at the LHC.

Contents

1 Introduction

One of the aims of the CMS experiment at the LHC at CERN is to identify muons in proton-proton collisions accurately at the centre-of-mass energy of $\sqrt{s} = 13.6$ TeV in Run 3. This is very important for many analyses with muons as signatures for interesting physical phenomena and objects that can be calibrated with the best precision. The main challenge is identifying genuine muons from background sources, such as misidentified charged hadrons. Furthermore, discriminating between prompt and non-prompt muons is important for particular analyses such as those that are targeting decays of the W, Z, or Higgs bosons, or τ leptons [3]. There are many challenges when it comes to background estimation in high-energy physics experiments, especially in data from Hadron colliders.

These issues arise mostly because QCD multi-jet events have large cross-sections, thus it is computationally intensive to generate enough MC events and the modelling of non-prompt leptons is inaccurate because effects like hadron decays are happening in the non-perturbative QCD regime and are described by phenomenological models, e.g. parton showers via the Pythia programme. Because of these limitations in theoretical models, we resort to using Data-Driven Techniques which use data from control regions to estimate the background in the signal region, relying on the assumption that the background distribution is smooth and can be interpolated or extrapolated. These estimates are vital to estimate the background with good precision and accuracy. This can help to identify possible new physics on top of the background. Conventional background data-driven estimation methods, such as the ABCD method, often fall short of accounting for correlations between control variables, leading to inaccurate background estimations. In which case, we resort to using an extended version of the ABCD method which accounts for correlations, provided that they are "low enough" between the parameters used for the background estimation [2].

2 Lepton Identification

Lepton identification is one of the crucial steps in particle physics experiments. Leptons, specifically electrons, muons, and taus, belong to a family of fundamental particles that play an important role in a wide variety of physical processes. Their accurate identification is crucial for event analysis and meaningful physics results. Many processes produce particles that mimic muons, like hadrons misidentified as muons. Reducing these misidentifications using effective identification techniques improves the quality of the samples. Reconstructing muon tracks from data from different detector systems and employing selection criteria based on factors such as transverse momentum, impact parameter, and number of hits in the detector are steps in the identification process. Sophisticated methods such as multivariate analysis (MVA) improve the overall efficiency of the identification process by enhancing the distinction between real muons and background noise. Muon isolation is important for accurately identifying prompt muons that originate from the decay of heavy particles, such as W and Z bosons. Isolation helps reduce contamination

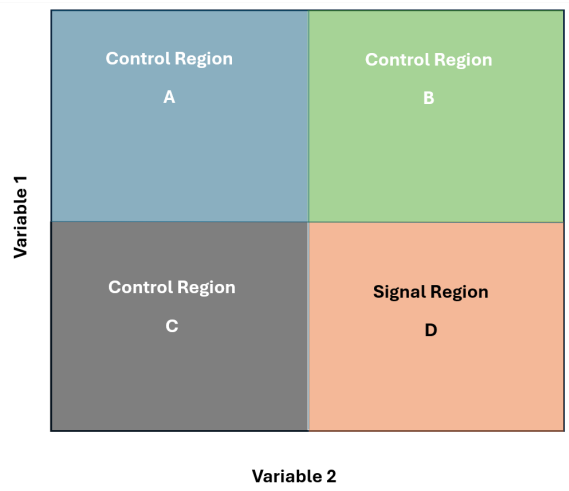

Figure 1: Visual showing the regions in the traditional ABCD background estimation Method.

from non-prompt muons from hadron decays, thereby enhancing sample purity. Isolation is quantified using variables that measure the energy of nearby particles within a defined cone around the muon, ensuring that identified muons are genuine signals rather than misidentified background events.

3 The Traditional and Extended ABCD Method

The traditional ABCD method relies on two control variables that are assumed to be uncorrelated. The two-dimensional phase space is divided into four regions, one of which is the signal region (SR), and the other three neighbouring regions are the control regions (CR), refer to Figure 1. The two variables need to be as uncorrelated as possible, and the choice of those variables depends on the physics process being studied. The information from the CRs (A, B and C) is then used to estimate the expected number of events in the SR (D), which can be expressed as:

$$\hat{F}_D = \frac{F_C F_B}{F_A}$$

where $\hat{F}_D$ is the expected number of background events in the SR and F_i denotes the number of background events in the i^{th} CR. The control variables are not entirely uncorrelated in most cases which might introduce biases in the background estimation. However, the extended ABCD method introduces additional CRs to capture more information about the correlations between control variables instead of assuming that they are completely uncorrelated [2]. But if the correlation becomes too large, the extended ABCD method also breaks. Therefore, it is beneficial to reduce the correlation between the control variables to avoid potential bias.

4 ML in Data-Driven Background Estimation Methods

The main goal of this study is to use Machine Learning (ML) to produce two (almost) uncorrelated parameters that will then be used for background estimation using data-driven methods. The ML model used in this study attempts to uncorrelate scores from a classification task which distinguishes between prompt and non-prompt leptons and the isolation of the leptons. These two parameters can be used to estimate the expected number of background events in the signal region of interest. The extended ABCD method is used to achieve this, but it is plausible that the simple ABCD method is good enough if the correlation can be reduced well enough. Massive amounts of labelled data are typically needed to train deep architectures to perform well. But in the case of the unavailability of labelled data for particular tasks, which is often the case in particle physics, domain adaptation is frequently used to mitigate this given that labelled data of a similar nature is available but originates from a separate and different domain. But for our specific case, make use of domain adaption to unlearn target domain-specific features.

Our main objective is to develop an architecture that can be trained on a large labelled dataset from the **source domain** and a large labelled dataset from the **target domain**. The training then promotes the emergence of discriminative features for the main classification task in the source domain that are invariant with respect to the shift between the source and target domains. This should essentially reduce the correlation between the scores of the main classification task and the target domains. This can be done on any feed-forward model by adding a few standard layers and a simple gradient reversal layer at the start of the target domain branch of layers [1].

5 The MVA Machine Learning Model

In this study, an existing neural network (NN) framework called **weaver** was used to train the model. This meant that the neural network architecture developed in the study needed to be compatible with this large framework and ultimately compatible with data samples used by the CMS collaboration. This framework had numerous moving parts that made the model work, however, we highlight only the most relevant for the study.

5.1 Basics of Neural Networks

A neural network model called **SimpleParticleNet** was developed to manage a range of particle physics features for domain adaptation and classification applications. It combines multiple feature types and uses several layers to process these features, ultimately outputting source and target domain classification predictions. This model consists of a gradient reversal layer, dropout layers, and fully connected layers.

The basic units of the neural network are called neurons. Inputs are put into every neuron, usually processed via a weighted sum and an activation function, and then sent to the next layer. Neurons are organized into layers. These consist of the input layer, the hidden layer(s), and the output layer(s). While inputs are received by the former, the latter is where data gets transformed through a series of computations performed by

each hidden neuron. The last layer generates final outputs from this network, and it may be made up of more than one node. Each neuron in a fully connected layer connects to every neuron in the previous layer. This kind of layer can be implemented using matrix multiplication followed by an optional activation function. Activation functions introduce nonlinearity to the model so that very complex patterns can be learnt (e.g. ReLU – Rectified Linear Unit, GELU – Gaussian Error Linear Unit, Sigmoid, softmax, etc). A Dropout layer is added as a regularisation technique that randomly sets a fraction of input units to zero at each update during training. This prevents overfitting by making the network more robust.

Training involves adjusting the weights and biases of the network to minimize the deviations between the predicted outputs and the actual target values. This is done using optimization algorithms like gradient descent. The loss function measures these deviations from the predicted outputs and the target values. It is an important component in training, as the network is trained to minimize this loss. Backpropagation is the process of calculating the gradient of the loss function with respect to each weight by the chain rule. These gradients are then used to update the weights to minimise the loss. The optimisation algorithm (like stochastic gradient descent) updates the weights using the gradients obtained from backpropagation to minimise the loss function.

5.2 Domain Adversarial Neural Network Architecture

Using a python library *PyTorch*, two specialized neural network architectures were created. Both of these DANN models were designed to use adversarial training to incorporate domain adaptation while training for the main classification task. Fully connected layers processing input features from various sources, including particle flow features (**pf_features**), secondary vertex features (**sv_features**), and lepton features (**lep_features**), formed the foundation of the models. After flattening, concatenating, and going through several fully connected shared layers with dropout for regularisation and the RELU activation function, the training branched into two branches, i.e. the main classification task and the domain adversarial task. These features were finally produced as class logits via a final classification layer that was applied to both the source and target domains.

A key aspect and the difference between these architectures was the integration of domain adaptation, where one used classification and the other used regression. They were designed to make the network's feature representation domain-invariant. This was achieved using a gradient reversal layer (**GradientReverse**), which was implemented through a custom autograd function (**RevGrad**). This layer helps the feature extractor "unlearn" features that are specific to the target domain by effectively reversing and scaling the gradients by a factor $-\alpha$ during backpropagation.

The Loss function used to optimise the training for the domain classification architecture was **binary cross-entropy** which outputs a probability value between 0 and 1. The Loss function used to optimise the training for the domain regression architecture was the **mean square error** which calculates the average of the squared differences between predicted and actual values. The domain classification **ClassDomain** architecture had three nodes that classified the target domain into 3 categories (Low Isolation, Medium

Isolation, High Isolation) at the output layer and the domain regression **ClassReg** architecture had only one node at the output layer.

A **GradientAnalysisHook** is used to record the gradients before and after reversal in order to track and analyse the consequences of this gradient reversal. Particularly in adversarial circumstances where the network was taught to confuse the domain classifier while remaining efficient at the main classification task, the hook made it easier to see how the reversal affected the learning process. The network also included a domain classification branch, which, after applying gradient reversal, processed the feature representation through several additional fully connected layers. The output of these layers was then used to predict the target domain labels for both architectures, allowing the network to learn a domain-invariant feature space. During inference, the model applied the softmax function to the main classification and domain classification outputs, yielding probability distributions.

Overall, the architecture was designed to perform well on the main classification task while also being robust across different domains, making it suitable for tasks like domain adaptation where the goal was to generalise across varying data distributions. The gradient reversal mechanism was central to achieving this, as it ensured that the learnt features were not biased toward any specific domain.

5.2.1 Gradient Reversal Layer (GRL)

The primary function of the GRL is to enable adversarial training between the feature extractor and the domain classifier. It promotes the feature extractor's ability to develop representations that are indistinguishable across both source and target domains. During the forward pass, the GRL operates as an identity function, transmitting the input features from the preceding layer directly to the subsequent layer without any alterations. Consequently, the GRL does not modify the features during forward propagation; they are merely relayed to the target domain layers. The essential role of the GRL is activated during the backward pass (backpropagation), where it adjusts the loss gradient regarding the input features. In this phase, the GRL applies a negative scalar $-\alpha$, to the gradient. Thus, the feature extractor is effectively penalised when the target domain layers successfully differentiate between the source and target domains.

6 ML Model Evaluation

To quantify the performance of the domain classification (**ClassDomain**) architecture, we looked at 3 figures of merit. The first is the average accuracy of the main classification task which was denoted as **AvgAccCat**. The second is the average accuracy of the target domain classification, which is denoted as **AvgAccDomain**. The last is the **Pearson correlation coefficient** which gauged the strength of the correlation between the output of the main classification task and the target domain. The outputs of the main classification task were scores between 0 and 1 for four classes (**Prompt**, **Lightfake**, **Heavy** and **Tau**) and the variable that is used as the target domain is the isolation of charged leptons. The architecture is trained with the goal of maximising the AvgAccCat while making the correlation as close to 0 as possible. The value of α determines the strength of the gradient reversal and therefore a large value of α resulted

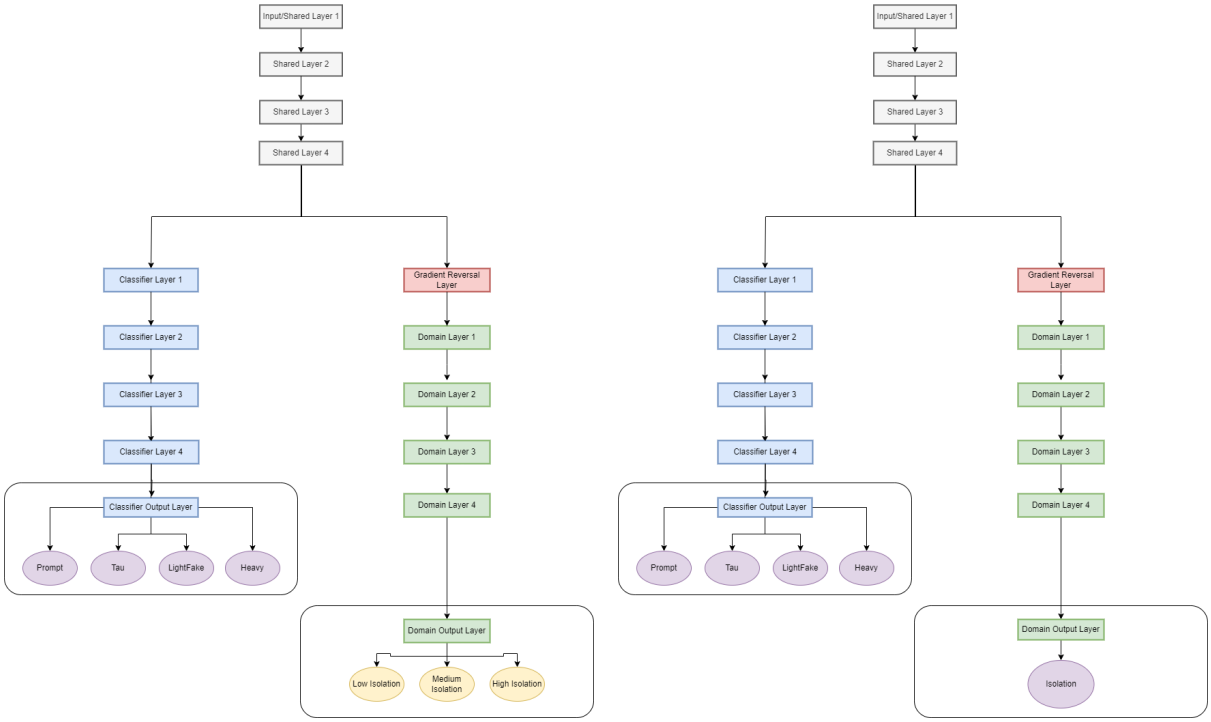

Figure 2: A schematic that shows the architecture of the two models, domain adversarial classification (left) and domain adversarial regression (right).

in a reduced correlation coefficient. The value $\alpha = 0$ corresponds to no gradient reversal applied in the target domain branch. Naturally, there is a trade-off between these goals since when the model does better in one, it does worse in the other, refer to Table 1. Low values of α because the accuracy of the main classification task reduces to zero very quickly; thus, the scores for every event are the same, causing errors when computing the correlation.

α	**AvgAccCat**	**AvgAccDomain**	**Correlation**
0	0.66908	0.91258	−0.40443
1	0.77396	0.91258	−0.39856
2	0.74410	0.91322	−0.43719
3	0.66169	0.91354	−0.49675

Table 1: Table showing the performance of the class-domain architecture for varying values of α

Similarly, to quantify the performance of the domain regression (**ClassReg**) architecture, we looked at 3 figures of merit, refer to Table 2.

Another metric that was used to quantify the performance of the two models was the **Receiver Operator Characteristic** (ROC) curve. The ROC curve essentially indicates the trade-off between the efficiency of the True Positive Rate (TPR) at which the model correctly identifies prompt leptons and the False Positive Rate (FPR) at which it incorrectly identifies non-prompt leptons as prompt. A good measure of how well the model performed could be captured using another numerical metric computed from the

α	**AvgAcc**	**AvgMSE**	**Correlation**
0	0.68789	0.00159	−0.49771
100	0.57870	0.00269	−0.45192
1000	0.35424	0.00394	−0.26113
10000	0.22232	0.00217	−0.21514

Table 2: Table showing the performance of the class-regression architecture for varying values of α

ROC curve was the area under the curve (AUC).

In Figure 3, The x-axis was the prompt efficiency and the y-axis was the non-prompt efficiency which meant that a good model would bend toward the lower right so that the non-prompt efficiency would be as low as possible. This meant that the lower the auc, the better the model performed. The ROC curves indicated that the ClassDomain model could distinguish between prompt and heavy leptons the best with an auc = 0.02 followed closely by lightfake with an auc = 0.03, which was followed by the combination of all 3 non-prompt flavours with an auc = 0.16 and the worst was the Tau with an auc = 0.24. In Figure 3, the ROC curves indicated that the ClassReg model could distinguish between prompt and heavy leptons the best with an auc = 0.06 followed by lightfake with an auc = 0.11, which was followed by the combination of all 3 non-prompt flavours with an auc = 0.24 and the worst was the Tau again with an auc = 0.36.

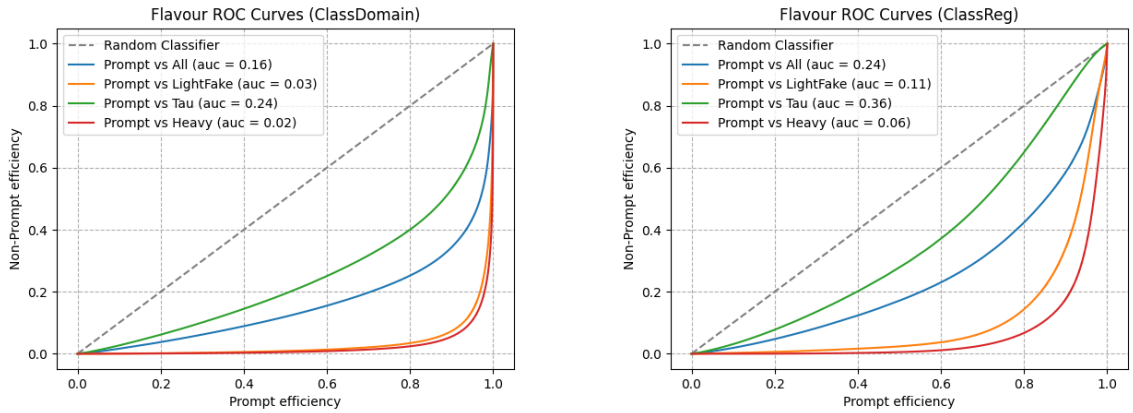

Figure 3: Plots that show the efficiency of the ClassDomain (left) and the ClassReg (right) models to distinguish between prompt and the other 3 non-prompt flavours and their efficiency to distinguish between prompt and all 3 non-prompt flavours combined with corresponding AUC metrics. The model was trained with $\alpha = 1000$.

The combined auc for ClassDomain of 0.24 was better than ClassReg of 0.36. This meant that the ClassDomain model performed the main classification task significantly better than ClassReg. Although not all results are yet in place, we observed that the increase in the value α slightly decreased the correlation in the ClassReg architecture, refer to Table 2, confirming that the training in the adversarial domain worked. However, we observed that this was not the case for the ClassDomain architecture since only $\alpha = 1$ slightly reduced the correlation, refer to Table 1. Values of α in the range between 0 and 1 may yield better results.

7 Conclusion

In this study, we explore the development and implementation of a DANN to reduce the correlation between the output scores of the main classification task and the target domain. By utilizing domain adversarial techniques within the CMS Weaver framework, we introduced a model that effectively reduces the correlation of the output scores of the main (lepton) classifier with the lepton isolation variable. Reducing correlations between variables is a common challenge in data-driven background estimation methods. Two architectures were developed and evaluated: the ClassDomain and ClassReg models. These architectures were designed to unlearn domain-specific features by making use of a gradient-reversal layer, allowing the feature extractor to develop representations that are invariant across both source and target domains.

The performance of both architectures was evaluated. The ClassDomain model aimed to maximise the accuracy of the lepton classification task while simultaneously minimising the correlation between the classifier scores and the target domain variable. However, the results indicated that as the gradient reversal strength (α) increased, the expected reduction in correlation was not clearly observed. While the accuracy of the lepton classification task varied with different values of α, the correlation between the classifier outputs and the target domain remained relatively constant, highlighting a limitation in the ClassDomain approach. On the other hand, the ClassReg model demonstrated a clearer trade-off between the reduction in correlation and the accuracy of the lepton classifier. As the gradient reversal strength α increased, the correlation between the main classification task and the target domain decreased, confirming that adversarial training effectively unlearned domain-specific characteristics. However, this reduction in correlation came at the cost of reduced accuracy in the lepton classification task, underscoring the inherent trade-off in the adversarial training process. The ClassReg model, therefore, performed as intended but also highlighted the challenge of balancing accuracy and correlation reduction.

The findings from this study suggest that while the ClassReg architecture effectively reduces correlations, careful consideration must be given to the value of α to maintain a balance between classification accuracy and domain invariance. The ClassDomain architecture, despite its potential, did not exhibit the expected decrease in correlation with increasing α, indicating that further refinement and experimentation with different hyperparameters or architectural adjustments may be necessary.

In conclusion, this work represents a significant step forward in the application of ML techniques to high-energy physics, particularly in the context of improving data-driven background estimation methods. By implementing domain adversarial approaches, we have demonstrated the potential to reduce correlations that traditionally impact data-driven background estimation. Future work should focus on optimizing these architectures, exploring alternative adversarial techniques, and applying these methods to larger and more diverse datasets to further enhance the accuracy and reliability of lepton classification in the CMS experiment.

References

[1] Yaroslav Ganin and Victor Lempitsky. *Unsupervised Domain Adaptation by Backpropagation*. 2015. arXiv: 1409.7495 [stat.ML]. URL: https://arxiv.org/abs/1409.7495.

[2] Suyong Choi and Hayoung Oh. *Improved Extrapolation Methods of Data-driven Background Estimation in High-Energy Physics*. 2021. arXiv: 1906.10831 [hep-ph]. URL: https://arxiv.org/abs/1906.10831.

[3] A. Hayrapetyan et al. "Muon identification using multivariate techniques in the CMS experiment in proton-proton collisions at $\sqrt{s} = 13$TeV". In: *Journal of Instrumentation* 19.02 (Feb. 2024), P02031. ISSN: 1748-0221. DOI: 10.1088/1748-0221/19/02/p02031. URL: http://dx.doi.org/10.1088/1748-0221/19/02/P02031.