Eur. Phys. J. C manuscript No.
(will be inserted by the editor)

Punzi-loss

A non-differentiable metric approximation for sensitivity optimisation in the search for new particles

F. Abudinén[14], **M. Bertemes**[16], **S. Bilokin**[18], **M. Campajola**[4,9], **G. Casarosa**[3,11], **S. Cunliffe**[2], **L. Corona**[3,11], **G. De Pietro**[12], **S. Dey**[20], **M. Eliachevitch**[21], **P. Feichtinger**[16], **T. Ferber**[15], **J. Gemmler**[15], **P. Goldenzweig**[15], **A. Gottmann**[15], **E. Graziani**[12], **H. Haigh**[16], **M. Hohmann**[22], **T. Humair**[19], **G. Inguglia**[a,16], **J. Kahn**[7], **T. Keck**[15], **I. Komarov**[2], **J.-F. Krohn**[22], **T. Kuhr**[18], **S. Lacaprara**[10], **K. Lieret**[18], **R. Maiti**[16], **A. Martini**[2], **F. Meier**[5], **F. Metzner**[15], **M. Milesi**[22], **S.-H. Park**[8], **M. Prim**[21], **C. Pulvermacher**[15], **M. Ritter**[18], **Y. Sato**[8], **C. Schwanda**[16], **W. Sutcliffe**[21], **U. Tamponi**[13], **F. Tenchini**[11], **P. Urquijo**[22], **L. Zani**[1], **R. Žlebčík**[6], **A. Zupanc**[17]

[1]Aix Marseille Universite, CNRS/IN2P3, CPPM, 13288 Marseille, France
[2]Deutsches Elektronen-Synchrotron, Hamburg, Germany
[3]Dipartimento di Fisica, Università di Pisa, I-56127 Pisa, Italy
[4]Dipartimento di Scienze Fisiche, Università di Napoli Federico II, I-80126 Napoli, Italy
[5]Duke University, Durham, USA
[6]Faculty of Mathematics and Physics, Charles University, Prague, Czech Republic
[7]Helmholtz AI, Karlsruhe Institute of Technology, 76131, Karlsruhe, Germany
[8]High Energy Accelerator Research Organization (KEK), Tsukuba, Japan
[9]INFN Sezione di Napoli, I-80126 Napoli, Italy
[10]INFN - Sezione di Padova, Padova, Italy
[11]INFN Sezione di Pisa, I-56127 Pisa, Italy
[12]INFN - Sezione di Roma Tre, Roma, Italy
[13]INFN - Sezione di Torino, Torino, Italy
[14]INFN - Sezione di Trieste, Trieste, Italy
[15]Institut für Experimentelle Teilchenphysik, Karlsruher Institut für Technologie, Karlsruhe, Germany
[16]Institute of High Energy Physics, 1050, Vienna, Austria
[17]Jožef Stefan Institute, Ljubljana, Slovenia
[18]Ludwig Maximilians University, Munich, Germany
[19]Max-Planck-Institut für Physik, München, Germany
[20]Tel Aviv University, Tel Aviv, Israel
[21]University of Bonn, Bonn, Germany
[22]University of Melbourne, Melbourne, Australia

Received: date / Accepted: date

Abstract We present the novel implementation of a non-differentiable metric approximation and a corresponding loss-scheduling aimed at the search for new particles of unknown mass in high energy physics experiments. We call the loss-scheduling, based on the minimisation of a figure-of-merit related function typical of particle physics, a Punzi-loss function, and the neural network that utilises this loss function a Punzi-net. We show that the Punzi-net outperforms standard multivariate analysis techniques and generalises well to mass hypotheses for which it was not trained. Our result constitutes a step towards fully differentiable analyses in particle physics. This work is implemented using PyTorch, and we provide users full access to a public repository containing all the codes.

[a]e-mail: gianluca.inguglia@oeaw.ac.at (corresponding author)

1 Introduction

The standard model (SM) of particle physics is the theoretical framework that describes fundamental interactions and the fundamental constituents of matter. Although successful in predicting phenomena, there is a general consensus that this framework is not a complete description of nature, and new physics (NP) has to exist. Searches for NP beyond the SM can be grouped into two main categories: searches for direct production and decays of new, unknown particles; and searches for deviations from the theoretical predictions in precision measurements. When searching for new particles, for example in a collider experiment, one of the main challenges is to correctly reconstruct and identify the new particles (the signal) and reject any (or most) contributions from potential background sources. This is a common problem referred to as event classification. A common approach to correctly classify a signal with respect to background uses Monte Carlo (MC) simulation to generate signal- and background-like event distributions. The use of MC simulation can help find underlying features or patterns in the signal and the background distributions that allow one to disentangle the two (possibly) unambiguously. In the last decade, the use of advanced data analysis methodologies, such as multivariate analysis (MVA) methods, has often largely improved analysis signal selection power, allowing for more precise analyses, often performed in a shorter time. Typical MVA methods in use in the field of particle physics include, but are not limited to, decision trees, boosted decision trees (BDT) [1], or shallow and deep artificial neural networks (ANNs) [2]. This paper focuses on the implementation of ANNs. We propose and describe how to implement a new loss function, called Punzi-loss, based on the so-called Punzi figure-of-merit (FOM) [3]. We henceforth refer to a neural network trained with the Punzi-loss function as a Punzi-net. As a benchmark study to test the performance of the Punzi-loss and compare it to other techniques, we consider the search for invisible decays of the hypothetical Z' boson produced in the reaction $e^+e^- \to \mu^+\mu^- Z$' at the Belle II experiment [4,5] at the SuperKEKB collider [6], based on MC simulations.

2 Neural networks

Artificial Neural Networks (ANNs) describe a broad class of neural network implementations (e.g. convolutional neural networks, transformers, etc.), used in a variety of applications ranging from image classification in the case of CNNs to natural language processing with transformers. In this work, we focus on a fully connected feed-forward neural network for our experiments. We nonetheless emphasise that the concepts outlined in this work apply to all neural network implementations that use backpropagation.

A Neural Network comprises a collection of connected neurons. In a fully connected neural network, these constitute a series of layers in which each neuron is connected to all those in both the previous and subsequent layers. Each neuron describes a mathematical function that produces an output dependent on those input connections and some unique bias, defined as

$$a^l_j = \sigma\left(\sum_k w^l_{jk} a^{l-1}_k + b^l_j\right). \tag{1}$$

Where w^l_{jk} is the weighting of the connection to the k^{th} neuron in the previous $(l-1)$ layer, b^l_j is the bias and σ is the *activation function*. A variety of different activation functions can be applied here, and most have specific traits that may be desirable depending on the application. Commonly used examples include sigmoid, rectified linear activation (ReLU), or hyperbolic tangent functions. The key requirements are that they are non-linear and have a derivative defined everywhere. Using Eq. 1, a network of individual neurons is able to map input variables to some desired output. For this to be possible, however, the weight and bias parameters must be optimised. In the implementation we present here, this is done via *supervised training*, whereby training data, x, is passed to the network along with the set of corresponding labels, y. The actual output of the network, $f(x) = \hat{y}$, can then be compared with this desired output to measure how well it maps input data. This comparison is quantified by way of a loss function, a commonly used example of which is the Binary Cross Entropy loss,

$$L = -y\ln\hat{y} + (1-y)\ln(1-\hat{y}), \tag{2}$$

where $y \in \{0,1\}$ and $\hat{y} \in [0,1]$. With this measure of the error, the training process becomes a minimisation problem: what set of weights and biases will minimise the loss and therefore provide the most effective network? This is solved by employing a method such as *stochastic gradient descent*, by which the parameters can be iteratively adjusted in the direction opposite that of the loss function's gradient,

$$w_{n+1} = w_n - \eta\frac{\delta L}{\delta w_n} \text{ and} \tag{3}$$

$$b_{n+1} = b_n - \eta\frac{\delta L}{\delta b_n}, \tag{4}$$

where η is the *learning rate*, the step size by which the parameters are adjusted at each iteration of the learning process. Each of these iteration steps constitutes a complete pass through a randomly sampled *batch* from the full training data set, a pass through the entirety of which is referred to as an *epoch*. The derivatives $\frac{\delta L}{\delta w_j}$ and $\frac{\delta L}{\delta b_j}$ are calculated through use of the *backpropagation* algorithm. This starts from the final layer and utilises the chain rule to calculate all the derivatives incrementally through one full backward pass to the first layer. The key is the careful selection of a loss function whose minimum solves the given task while remaining differentiable across all possible neural network outputs.

3 Figure of merit

As highlighted in Section 1, one of the main challenges when performing a precision test of the SM, or in the search for NP, is the fact that some background processes may mimic the signal and therefore contaminate the results. In the search for a new particle, for example, one is often performing a counting experiment which is described by the Poisson distribution. As discussed in [3], the number of events n in a counting experiment in the case of a background (B) only hypothesis (H_B), and in the case of a signal (S) in the presence of the same background (H_{S+B}) follows the Poisson distributions

$$p(n \mid H_B) = \frac{B^n e^{-B}}{n!} \tag{5}$$

and

$$p(n \mid H_{S+B}) = \frac{(S+B)^n e^{-(S+B)}}{n!}. \tag{6}$$

When MC simulations for both the signal and the background are available, it is possible to identify quantities or features in the data to separate and classify them correctly by applying some specific selection criteria. This would eventually enable one to choose between the (null) background only and the signal plus background hypotheses. In general, however, applying some selection criteria to reduce the background contamination will also remove some of the signal. It is, therefore, fundamental to define some additional criteria that would indicate the best balance between reducing the background without compromising the signal. This is done via the implementation of a figure-of-merit. One can define $S(t)$ and $B(t)$ as the number of signal and background events that pass some selection criteria (e.g. particles having a momentum or energy larger than a specified threshold t). In that case, standard FOMs used in particle physics are:

$$FOM = \frac{S(t)}{\sqrt{B(t)}} \text{ and} \tag{7}$$

$$FOM = \frac{S(t)}{\sqrt{S(t)+B(t)}}. \tag{8}$$

Neither of the above is usable in the search for new particles since the number of expected signal events depends on the cross-section of the process, and this is not known *a priori*. An alternative FOM for this specific case was proposed in [3], often referred to as the Punzi FOM after the author, and is now in widespread use. The Punzi FOM to maximize is the inverse of the minimal detectable cross-section σ_{min}, defined as [3]

$$\sigma_{\text{min}}(t) = \frac{\frac{b^2}{2} + a\sqrt{B(t)} + \frac{b}{2}\sqrt{b^2 + 4a\sqrt{B(t)} + 4B(t)}}{\varepsilon(t)\cdot L}, \tag{9}$$

where a and b are the number of sigmas corresponding to one-sided Gaussian tests at some predefined significance level (α and β), L is the target luminosity, $\varepsilon(t)$ is the signal efficiency and $B(t)$ is the number of background events after the selection defined by t [3].

4 Punzi-loss

We propose here a quantity approximating the Punzi FOM, appropriate for optimising neural networks for physics selections.

This loss function is based on the equation for the Punzi sensitivity region (Eq. 9). However, Eq. 9 can not be used directly because the number of background events B and the signal efficiency ε are discrete functions of the network parameters for any given fixed cut on the classifier output, whereas the loss function must be differentiable. We can build a differentiable function by replacing the fixed cut on the output with a sum over all events, weighted with the respective value of the output. If events classified as signal cluster around an output of 1 and events classified as background at 0, this quantity will closely approximate the original function. In Eq. 9 this weighting can be captured by performing the replacements

$$\varepsilon(t) \rightarrow \varepsilon(\boldsymbol{w},\boldsymbol{b}) = \sum_{\boldsymbol{x}} \frac{y_i \cdot \hat{y}_i(\boldsymbol{w},\boldsymbol{b}) \cdot s_{\text{sig}}}{N_{\text{gen}}} \quad \text{and} \tag{10}$$

$$B(t) \rightarrow B(\boldsymbol{w},\boldsymbol{b}) = \sum_{\boldsymbol{x}} (1-y_i)\cdot \hat{y}_i(\boldsymbol{w},\boldsymbol{b}) \cdot s^i_{\text{bkg}}, \tag{11}$$

where the sum is over all training inputs $\boldsymbol{x}$ and the index i denotes the i^{th} training event. The collection of weights and biases that constitute the free parameters of the network are denoted as $\boldsymbol{w}$ and $\boldsymbol{b}$. N_{gen} is the total number of generated signal events, s_{sig} is a scale factor for the signal and s^i_{bkg} is a scale factor for the background, which can include a weight factor to scale the luminosity for the individual simulated background samples to the target luminosity. The scale factors can also include correction factors such as trigger efficiencies and should account for the sample size when only a subset of the generated data is used to compute the loss. A similar approach of building a differentiable metric based on a FOM was taken by Elwood and Krücker [7], with a loss function based on the discovery significance.

The Punzi-loss function is given by the arithmetic mean of this continuous Punzi sensitivity calculated for all signal hypotheses ($m_{Z'}$) that are used in training,

$$C_{\text{Punzi}} = \frac{1}{N_{Z'}} \sum_{m_{Z'}} \sigma_{\text{min}}(\boldsymbol{w},\boldsymbol{b}), \tag{12}$$

with $N_{Z'}$ being the total number of hypotheses that were considered. Note that this loss function can no longer be calculated using single training events but is instead based on a set of training data.

To test the Punzi-loss function, we implemented a simple fully-connected network in `PyTorch` [8] with four input neurons, one output neuron, and two hidden layers with 8 and 4 neurons, respectively. The size of the net was determined empirically to give good results while keeping the network relatively small.[1]

5 Training strategy

For the Punzi-loss training to converge, we found that the parameters of the network should already be initialised in a way that defines some separation between signal and background (similar to the loss scheduling scheme described in [9]). This can be achieved by pretraining the network using a conventional loss function and subsequently fine-tuning this through the use of the Punzi-loss function.

For the activation function of the neurons in the hidden layers, a hyperbolic tangent is used while the output neuron uses a sigmoid function. Before training, the input variables were scaled to lie between 0 and 1, and the network parameters were randomly initialised. For the pretraining, a weighted binary cross-entropy (BCE) loss function was used. A weighting was attributed to the signal events such that their weighted sum was equal to the weighted sum of all background events. An outline of the network architecture is given in Figure 1.

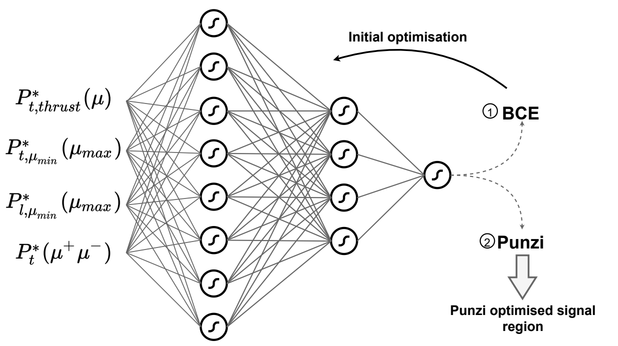
$P^*_{t,thrust}(\mu)$
$P^*_{t,\mu_{min}}(\mu_{max})$
$P^*_{l,\mu_{min}}(\mu_{max})$
$P^*_t(\mu^+\mu^-)$
Initial optimisation
① BCE
② Punzi
Punzi optimised signal region

Fig. 1 An outline of the network architecture. The first training with the BCE loss function was used to set the weights and biases of the net for the second training with the custom loss function based on the Punzi figure of merit.

Initially, using the BCE loss function, the network was trained with a batch size of 2048 and a learning rate (LR) of 1. When the loss did not decrease for 10 epochs, the LR was reduced by a factor of 0.5. The pretraining was stopped after 200 epochs. This network is then once again trained using the Punzi-loss function for which a learning rate of 0.0005 was used, and again this was reduced upon plateauing. This training was stopped after 2000 epochs. For both of these trainings, the stochastic gradient descent algorithm was used for optimisation. All hyperparameters were optimised to give the best results for the respective training methods. One particularly important hyperparameter is the batch size, the variation of which presents some unique aspects of the Punzi-loss function that must be considered.

[1]The network size and architecture is not relevant for our approach.

Due to the nature of the Punzi-loss function concerning the optimisation for a desired luminosity, utilising batches requires the addition of weightings in the loss calculation. The background data used for training contained 1000 fb^{-1} worth of events; however, in this study, we wish to optimise the classifier for just 50 fb^{-1} of real-world data. Naturally, it is preferred that all background data is utilised, and thus we introduce a background scaling factor of 0.05. Additionally, dividing the training data into batches brings about the additional requirement of multiplying both $\varepsilon(\boldsymbol{w},\boldsymbol{b})$ and $B(\boldsymbol{w},\boldsymbol{b})$ by the number of batches used.

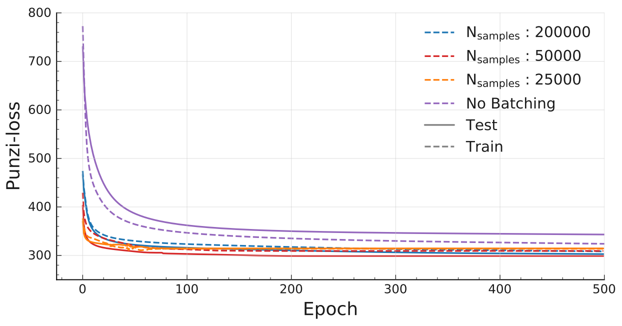

Fig. 2 Evolution of Punzi-loss during training with batch sizes of 2×10^5, 5×10^4 and 2.5×10^4 and without batching.

Fig. 2 shows the evolution of loss during training with batch sizes of 2×10^5, 5×10^4 and 2.5×10^4, and also no batching (where the whole data set is passed as a single batch). The validation results for each of these all settle at similar values around 300 to 350. Batches of 50 000 appear to provide the best final loss value. While a batch size of 200 000 does reach the same value, it does so in a greater number of epochs and thus, due to its shorter training time, a batch size of 50 000, along with the unbatched model, is used in the following experiment.

We note that small batch sizes bring a degree of instability to the loss. We found that batches smaller than those shown in Fig. 2 led to increasing loss values over the training. This can be understood as a result of the limited number of signal events present in any given batch of a small size, leading to large statistical fluctuations in the calculated loss values. This lower limit is, of course, study dependent.

6 Results

In this section, we present the results of utilising a Punzi-net in a search for $e^+e^- \to \mu^+\mu^-Z$' signals amongst various common backgrounds found in e^+e^- collider experiments. At the Belle II experiment, this search was performed with the commissioning data for the specific case of invisible decays of the Z' boson [10], a final state in which only the two muons produced by the electron-positron annihilation can be reconstructed. All information about the production and decay of the Z' boson is therefore to be inferred by the two-muon system. The signal events are generated with `MadGraph 5` [11] for a range of candidate Z' masses, spanning 0.1 GeV/c^2 to 8.7 GeV/c^2 in steps of 0.1 GeV/c^2 with 20 000 events produced at each. Additionally, MC samples for the background process $e^+e^- \to e^+e^-\mu^+\mu^-$, $e^+e^- \to \tau^+\tau^-$ and $e^+e^- \to \mu^+\mu^-(\gamma)$ corresponding to 1000 fb^{-1} were used, since these can mimic the signal. The simulation and reconstruction of the events was done using `GEANT4` [12] and the Belle II Analysis Software Framework [13]. The analysis is conducted via the search for a peak in the distribution of the squared mass recoiling against the two-muon system. An excess of entries beyond that of the expected background at a given mass would indicate the presence of such a Z' particle of that mass. This distribution is divided in (potentially overlapping) bins with bin widths corresponding to $\pm 2\sigma$ of the fitted Z' signal distributions.

During both the initial BCE and subsequent Punzi-loss training, only every second generated Z' mass was used. For the calculation of σ_{min} in Eq. 12 only signal and background events that lie within the respective $\pm 2\sigma$ mass windows are considered, using only signal events that were generated for the corresponding mass. Thus, events that are not contained in any of the mass windows of the used signal samples are not taken for the training. This results in a data set of approximately 3.25 million total events, of which $\sim$10% are signal and the rest background. This is then split 80%/20% for training/validation. The unused signal hypotheses are utilised for validation and to check the trained networks ability to generalise to signals unseen in training. The network was trained with four carefully selected features related to the event kinematics that showed a good discrimination power when using a boosted decision tree classifier. These features are described in Tab. 1. A more detailed description of these features and the analysis can be found in [14].

The resultant maximum achievable Punzi figure of merit and corresponding sensitivity spanning the range of generated Z' signals are shown in Figs. 3 and 4, respectively. Included in each of these figures are the Punzi trained networks, both without batching and with a batch size of 50 000, along with the BCE pretrained network. These values are calculated using the background data contained within the $\pm 2\sigma$ bin around each generated mass point. The maximum achievable Punzi FOM and subsequent sensitivity values in each bin are found using the cut to the network output that provides the highest FOM for that respective bin. The plots show the average result found over 25 independently trained networks, along with the associated standard error. This serves to demonstrate that not only can the Punzi-loss function produce better FOMs, but can do so consistently. In contrast, BCE trained networks appear to achieve varying results after their respective training, as indicated by the greater spread in the standard error.

variable	description
$p^*_{t,\text{thrust}}(\mu)$	The transverse momentum component of the muons with respect to the thrust axis in the CMS.
$p^*_{t,\mu_{\text{min}}}(\mu_{\text{max}})$	The transverse momentum component of the higher energetic muon with respect to the lower energetic muon in the CMS.
$p^*_{l,\mu_{\text{min}}}(\mu_{\text{max}})$	The longitudinal momentum component of the higher energetic muon with respect to the lower energetic muon in the CMS.
$p^*_t(\mu^+\mu^-)$	The transverse momentum of the dimuon system in the CMS.

Table 1 The most important features found after training BDTs with many variables. These features are used for training the ANN.

The Punzi-loss function shows greater effectiveness through the lower half of the recoil mass spectrum, providing clear improvements to the FOM below approximately 4 GeV/c^2. Furthermore, we note that introducing batching not only speeds up the training but also greatly improves the results; while the unbatched training appears to improve the FOM in the lower masses at the cost of decreasing in the higher masses, the batched training makes the same improvements while maintaining performance similar to that of the BCE trained networks at higher masses. This may be a result of the batching introducing stability to the training and reducing sensitivity to initial hyperparameters due to the additional stochastic component in the training.

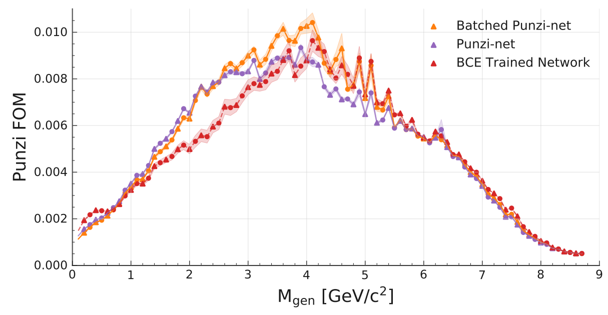

Fig. 3 The average maximum Punzi FOM achievable in each bin across range of generated Z' signals, with standard error spread taken from 25 independently trained networks. Triangles indicates those masses which were left out of training while circles indicates those used.

The generated Z' masses used for training of the network are shown with circles, and those not used in training are shown with triangles. The figures show little to no difference in the network's ability between these training and

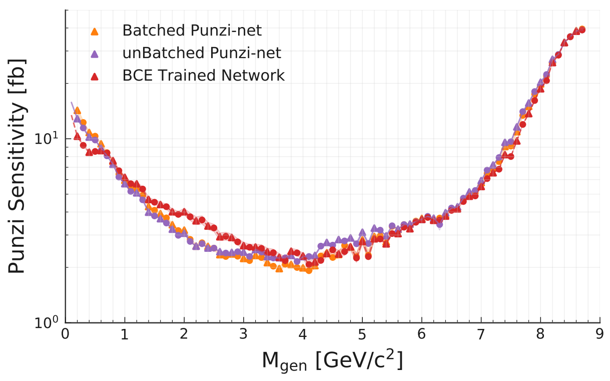

Fig. 4 The average maximum sensitivity achievable in each bin across range of generated *Z*' signals, with standard error spread taken from 25 independently trained networks. Triangles indicate the masses which were left out of training while circles indicate those used.

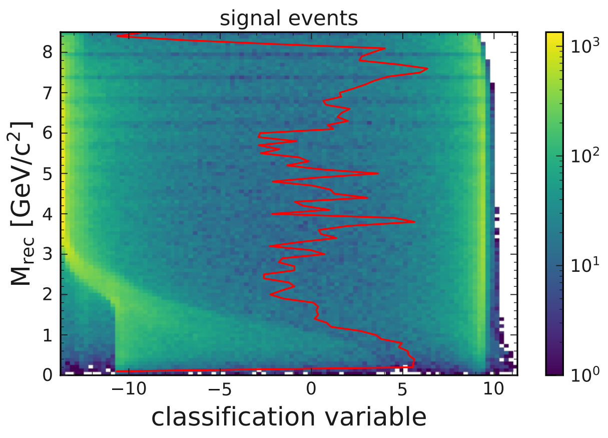

Fig. 5 The output distribution of all signal events using the Punzi-loss trained NN, overlaid with the cut value that would give the best Punzi FOM for each signal hypothesis. The classification variable shows the NN output before applying the last sigmoid function in order to better see the separation. The optimal cut value can be replaced by a uniform cut without any significant difference in the resulting selection.

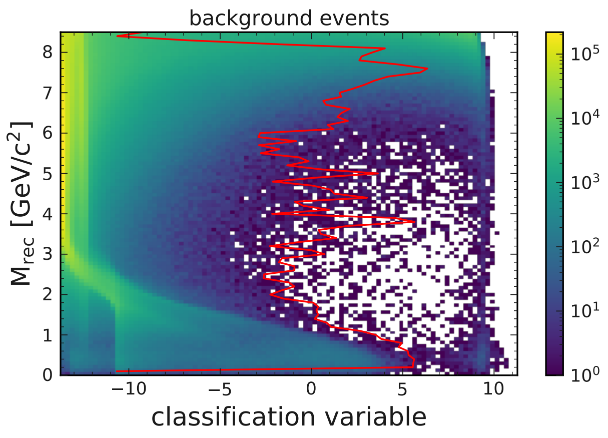

Fig. 6 The output distribution of all background events using the Punzi-net, overlaid with the cut value that would give the best Punzi FOM for each signal hypothesis.

validation masses, indicating that the model generalises well to unseen signals. In the region between approximately 4.5 GeV/c^2 and 5.5 GeV/c^2 some dependence on whether or not a mass was used in training does appear, this could be combated by generating a larger set of *Z*' signals covering more mass points in that region.

Fig. 5 shows the output of the Punzi-trained network for all signal events and Fig. 6 shows the same for all background events. Here the variable on the x-axis shows the NN output before applying the last sigmoid activation function to better resolve the distribution of events. The classified signal and background events are separated into two clusters, corresponding to an output of 0 and 1. The overlaid line shows the cut value that would give the maximum achievable FOM for each *Z*' mass. The line separates the two clusters, showing that the training using the approximations in Eq. 10 and 11 worked as expected.

The events are separated so that when only the signal classified events are selected (for example, by applying a cut at a NN output of 0.5), this gives the optimal Punzi FOM for the whole mass range. This is a big advantage for an analysis since no additional interpolation between output values is required which can introduce discontinuities in the final recoil mass distribution. Additionally, since the selection generalises to all signal hypotheses, it gives also the best possible FOM for a signal in-between trained masses, which would otherwise have non-optimal results.

The values of the Punzi FOM obtained with the same cut on the network output applied to all mass values is shown in Fig. 7, together with the maximum achievable Punzi FOM values for the BCE-trained network. We note that, for the Punzi-net, applying a single cut to the output maintains performance very close to that of the maximum achievable in any given bin. This means that even when compared to an optimal varied cut applied to the BCE network output, interpolated across the recoil mass spectrum, the Punzi-nets show strong performance. As discussed previously, this cut interpolation can lead to discontinuities in the final recoil mass distribution, and so the ability to achieve comparable results with a single cut to the Punzi-net output is much preferable.

An understanding of why the model successfully generalises, and one network can be utilised for the full squared recoil mass spectrum, can be inferred from Fig. 8, which shows a 3D scatter plot of the $p^*_{t,\mathrm{thrust}}(\mu)$, $p^*_{t,\mu_{\min}}(\mu_{\max})$ and $p^*_{l,\mu_{\min}}(\mu_{\max})$ variables (after being normalised to values between 0 and 1) for three of the mass bins at a region of $p^*_t(\mu^+\mu^-) = (2.2 \pm 0.5)$ GeV/c. The green plane is the chosen signal/background classification boundary obtained with a single cut. One can see the masses describing three respective planes in the parameter space which occupy distinct regions. This partitioning allows the network to adapt between

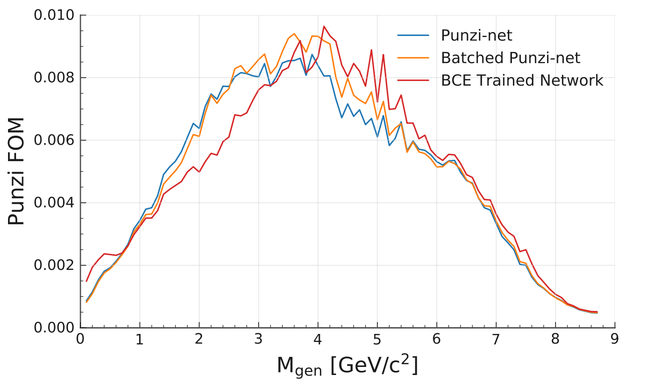

Fig. 7 The average FOM achieved with the best single cut applied to the Punzi-net and average maximum Punzi FOM achievable with the optimal varying cut to the BCE trained network in each bin across range of generated Z' signals.

the different mass regions and so negates any need for multiple classifiers for different regions.

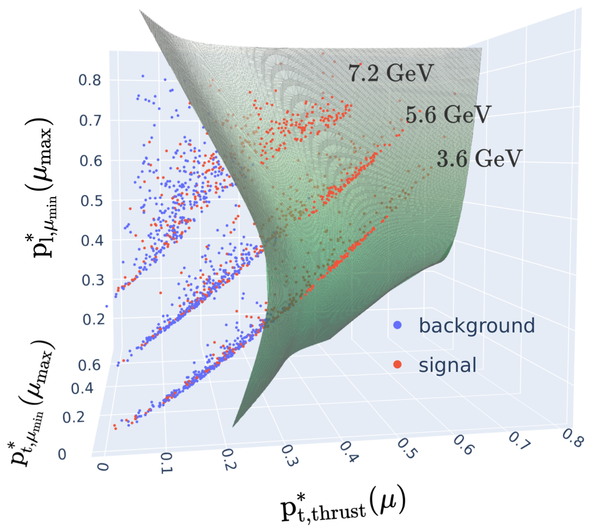

Fig. 8 A 3D scatter plot showing the input space of the ANN with $p_t^*(\mu^+\mu^-)$ fixed around 2.2 GeV/c. The separation boundary defined by the final selection (green sheet) separates the planes corresponding to different recoil masses in a way that optimises the selection for all signal hypotheses.

7 Conclusions

We have demonstrated that it is possible to implement a non-differentiable metric approximation and a corresponding loss-scheduling, combining both the approach of particle physics and that of machine learning. We have provided details on how this can be done, along with a publicly available code implementation in PyTorch [15]. We designed a new loss function directly related to a precise figure of merit and implemented it in the training of a neural network. We called the new loss function associated with the loss-scheduling a Punzi-loss function and the neural network implementing it a Punzi-net. Our proposed method applied to the search of new particles of unknown mass in high energy physics experiments achieves better performance than standard methods and simplifies the subsequent analysis since a common selection for all signal hypotheses can be applied on the classifier output. Our results are general and represent a further step towards a fully differentiable analysis framework in which the optimisation of the signal selection will also account for the size of systematic effects.

Acknowledgements The authors would like to thank the Belle II Collaboration and Belle II software group for useful discussions and suggestions on how to improve this work. P. Feichtinger, H. Haigh and G. Inguglia would like to acknowledge funding received under the Horizon 2020 framework of the European Research Council, ERC StG Nr. 947006 *InterLeptons*, and under the FWF standalone framework with grant Nr. P31361 *Searches for dark matter and dark forces at Belle II*. James Kahn's work is supported by the Helmholtz Association Initiative and Networking Fund under the Helmholtz AI platform grant.

References

1. T. Keck, FastBDT: A Speed-Optimized Multivariate Classification Algorithm for the Belle II Experiment, Comput. Softw. Big Sci. **1** (2017) no.1, 2 doi:10.1007/s41781-017-0002-8
2. K. Albertsson *et al.*, Machine Learning in High Energy Physics Community White Paper, arXiv: 1807.02876, 2019.
3. G. Punzi, Statistical problems in particle physics, astrophysics and cosmology. Proceedings, Conference, PHYSTAT 2003, Stanford, USA, September 8-11, eConf C030908, ArXiv: physics/0308063, 2003.
4. Abe, T., Belle II Technical Design Report, Belle II Collaboration, KEK-REPORT-2010-1, arXiv:1011.0352, 2010.
5. E. Kou *et al.*, The Belle II Physics Book, Progress of Theoretical and Experimental Physics **12**, 2050-3911, 2019.
6. T. Abe *et al.*, Accelerator design at SuperKEKB, Progress of Theoretical and Experimental Physics **3** 03, 2050-3911, 10.193/ptep/pts083, 2013.
7. Adam Elwood and Dirk Krücker, Direct optimisation of the discovery significance when training neural networks to search for new physics in particle colliders, arXiv:1806.00322, 2018.
8. A. Paszke *et al.*, An Imperative Style, High-Performance Deep Learning Library, Advances in Neural Information Processing Systems 32, 8024–8035, 2019.
9. O. Taubert *et al.*, Loss Scheduling for Class-Imbalanced Image Segmentation Problems, 2020 19th IEEE International Conference on Machine Learning and Applications (ICMLA)
10. I. Adachi *et al.*, Belle II Collaboration, Search for an Invisibly Decaying Z' Boson at Belle II in $e^+e^- \to \mu^+\mu^-(e^\pm\mu^\mp)$ Plus Missing Energy Final States, Phys. Rev. Lett. **124** 14, 141801, 2020.
11. J. Alwall *et al.*, The automated computation of tree-level and next-to-leading order differential cross sections, and their matching to parton shower simulations, JHEP 07, 079, 2014.
12. Agostinelli *et al.*, GEANT4–a simulation toolkit, Nucl. Instrum. Meth. A 506, 250–303, 2003.
13. T. Kuhr, The Belle II Core Software, Comput. Softw. Big Sci. **3** (2018) no.1, 1 doi:10.1007/s41781-018-0017-9

14. P. Feichtinger, Search for an invisibly decaying Z' boson and study of particle identification at the Belle II experiment, Master's Thesis, TU Wien, 2021 doi:10.34726/hss.2021.84843
15. Public Punzi-loss implementation, github.com/feichtip/punzinet