arXiv:2107.02119v1 [physics.ins-det] 5 Jul 2021

Calorimetric Measurement of Multi-TeV Muons via Deep Regression

Jan Kieseler[1], Giles C. Strong[2,3], Filippo Chiandotto[2], Tommaso Dorigo[3], and Lukas Layer[4,3]

[1]CERN
[2]Università di Padova
[3]INFN, Sezione di Padova
[4]Università di Napoli "Federico II"

June 30, 2021

Abstract

The performance demands of future particle-physics experiments investigating the high-energy frontier pose a number of new challenges, forcing us to find improved solutions for the detection, identification, and measurement of final-state particles in subnuclear collisions. One such challenge is the precise measurement of muon momentum at very high energy, where an estimate of the curvature provided by conceivable magnetic fields in realistic detectors proves insufficient for achieving good momentum resolution when detecting, *e.g.*, a narrow, high mass resonance decaying to a muon pair.

In this work we show the feasibility of an entirely new avenue for the measurement of the energy of muons based on their radiative losses in a dense, finely segmented calorimeter. This is made possible by exploiting spatial information of the clusters of energy from radiated photons in a regression task. The use of a task-specific deep learning architecture based on convolutional layers allows us to treat the problem as one akin to image reconstruction, where images are constituted by the pattern of energy released in successive layers of the calorimeter. A measurement of muon energy with better than 20 % relative resolution is shown to be achievable for ultra-TeV muons.

1 Introduction

Muons have been used as clean probes of new phenomena in particle physics ever since their discovery in cosmic showers [1, 2]. Their detection and measurement enabled many groundbreaking discoveries, from those of heavy quarks [3, 4, 5] and weak bosons [6] to that of the Higgs boson [7, 8]; most recently, a first evidence for $H \to \mu\mu$ decays has also been reported by CMS [9], highlighting the importance of muons for searches as well as measurements of standard model parameters. The uniqueness of muons is due to their intrinsic physical properties, which produce a distinctive phenomenology of interactions with matter. Endowed with a mass 206.77 times higher than that of the electron, the muon loses little energy by electromagnetic radiation as it traverses dense media; it behaves as a minimum ionizing particle in a wide range of energies, where it is easily distinguishable from long-lived light hadrons such as charged pions and kaons.

In continuity with their glorious past, muons will remain valuable probes of new physics phenomena in future searches at high-energy colliders. A number of heavy particles predicted by new-physics models are accessible preferentially, and in some cases exclusively, by the detection of their decay to final states that include electrons or muons; in particular, the reconstruction of the resonant shape of dileptonic decays of new Z' gauge bosons resulting from the addition of an extra U(1) group or higher symmetry structures to the Standard Model [10, 11] constitutes a compelling reason for seeking the best possible energy resolution for electrons and muons of high energy.

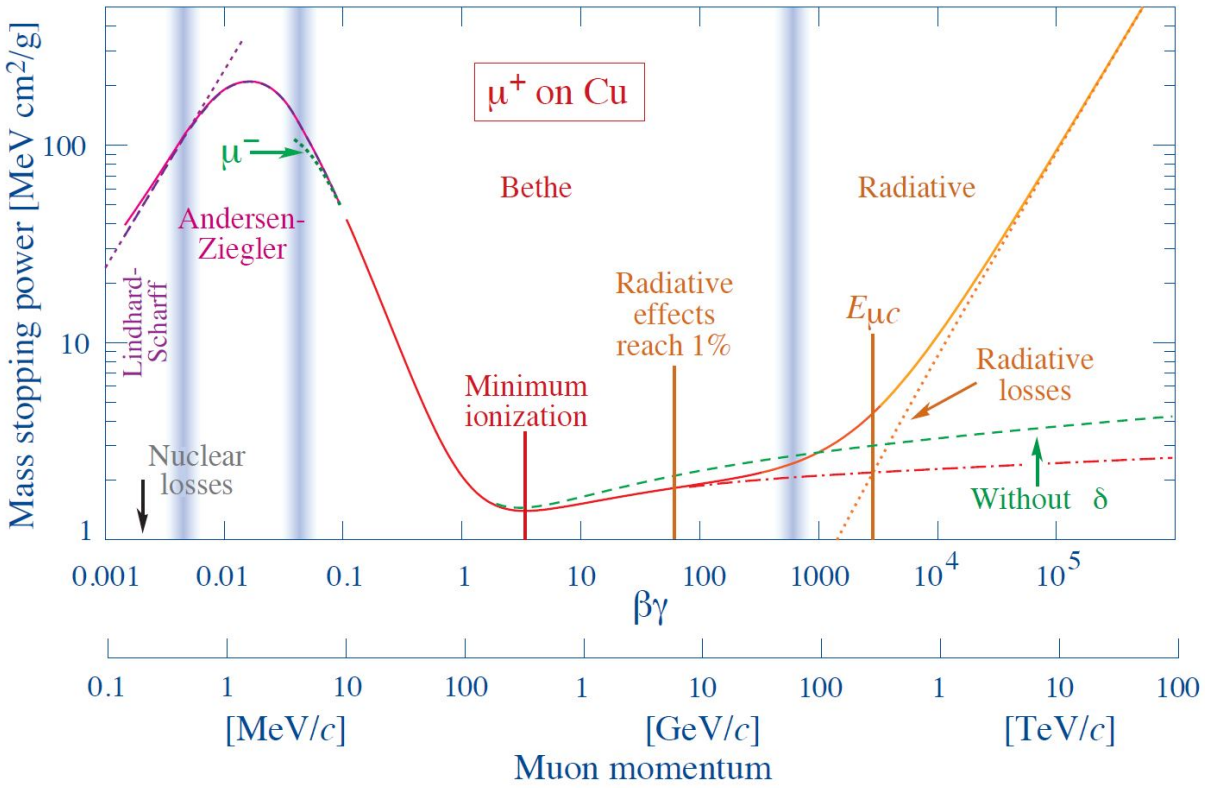


Figure 1: Mass stopping power for muons in the 0.1 MeV *to* 100 TeV *range, in* $\mathrm{MeV cm^2/g}$. *The rise in radiative loss becomes important above* 100 GeV. *Image source [12].*

Unfortunately, the very features that make muons special and easily distinguishable from backgrounds also hinder the precise measurement of their energy in the ultra-relativistic regime. While the energy

of electrons is effectively inferred from the electromagnetic showers they initiate in dense calorimeters, muon energy estimates rely solely on the determination of the curvature of their trajectory in a magnetic field. If we consider the ATLAS and CMS detectors as a reference, we observe that the relative resolution of muon transverse momentum achieved in those state-of-the-art instruments at 1 TeV ranges from 8 to 20% in ATLAS, and from 6 to 17% in CMS [13, 14], depending on detection and reconstruction details; by comparison, for electrons of the same energy the resolution ranges from 0.5 to 1.0% in ATLAS, and from 1 to 2% in CMS [15, 16]. Clearly, for non-minimum-ionizing particles, calorimetric measurements win over curvature determinations at high energy, due to the different scaling properties of the respective resolution functions: relative uncertainty of curvature-driven estimates grows linearly with energy, while the one of calorimetric estimates decreases with $\sqrt{E}$.

However, ultra-relativistic muons do not behave as minimum-ionizing particles; rather, they show a rise in their radiative energy loss [12] above roughly 100 GeV (see Fig. 1). The effect is clear, although undeniably very small in absolute terms; for example, a 1 TeV muon is expected to lose a mere 2.3 GeV in traversing the 25.8 X_0 of the CMS electromagnetic calorimeter [17]. For that reason, radiative losses have never been exploited to estimate muon energy in collider detectors[1]. It is the purpose of this work to show how, contrarily to the standing paradigm, low-energy photons radiated by TeV-energy muons and detected in a sufficiently thick and fine-grained calorimeter may be successfully exploited to estimate muon energy even in collider detector applications. Crucially, we will also demonstrate how the input of such a measurement is not only the magnitude, but also the pattern of the detected energy depositions in the calorimeter cells.

The spatial patterns of calorimeter deposits are a well known and heavily exploited feature for object identification purposes, *e.g.* to distinguish electromagnetic showers from hadronic showers by comparing the depth profile of the energy deposits [20, 15]. Recently, in the context of proposals for calorimeters endowed with fine grained lateral and longitudinal segmentation, it has been shown that this granularity not only improves the identification purity, but also allows for an accurate determination of the energy of hadronic showers, by identifying individual patterns of their electromagnetic and hadronic sub-components [21, 22, 23, 24, 25]. In parallel, machine learning techniques have proven to be very powerful for reconstructing individual showers [26, 27, 25] as well as multiple, even overlapping showers while at the same time being adaptable to the particularities of the involved detector geometries [28, 29, 30]. Also pattern recognition applications for quick identification of pointing and non-pointing showers at trigger level have been proposed [31, 32]. To some extent, the identification of low-energy photons that are emitted by the muons in the energy range we aim to investigate is an extreme case of a reconstruction-driven energy correction for hadronic showers. Therefore, we chose a deep learning approach to the problem, based on convolutional neural networks and loosely inspired by the techniques used for reconstructing hadronic showers in [26, 27].

The plan of this document is as follows. In Sec. 2 we describe the idealised calorimeter we have employed for this study. In Sec. 3 we discuss the architecture of the convolutional neural network we used for the regression of muon energy from the measured energy deposits. In Sec. 4 we detail our results. We offer some concluding remarks in Sec. 5. In Appendix A we describe the high-level features we constructed from energetic and spatial information of each muon interaction event; these features are used as additional input for the regression task. In Appendix B we offer an extensive ablation study of the model architecture and loss, the training schedule, and other technical aspects of our approach. Finally, in Appendix C we describe the hardware and time requirements of both

[1] To our knowledge, a measurement strategy has been demonstrated only in the IceCube experiment [18, 19], where the energy of muons of interest is still higher than that investigated in this work, and the thickness in radiation lengths of the traversed detector material is over an order of magnitude larger than that of present-day collider detectors.

the study and the regressor.

2 Detector geometry and simulation

2.1 Detector geometry

Since our goal in this work is to show the feasibility of muon-energy estimation from energy deposits in a calorimeter, we strip the problem of any complication from factors that are ancillary to the task. For that reason, we consider a homogeneous lead tungstate cuboid calorimeter with a total depth in z of 2032 mm and a spatial extent of 120 mm in x and y. The calorimeter is segmented into 50 layers in z, each with a thickness of 39.6 mm; this corresponds to 4.5 radiation lengths. Such a longitudinal segmentation allows for electromagnetic showers to be well resolvable. Each layer is further segmented in x and y in 32×32 cells, with a size of 3.73 mm $\times$ 3.73 mm. This results in 51 200 channels in total.

In order to make the study more realistic, as well as to provide a comparison with curvature-based momentum estimates, we assume that the calorimeter is embedded in a uniform 2-Tesla magnetic field, provided by an external solenoid or dipole magnet. The chosen magnet strength equals that of the ATLAS detector, and is in the range of what future collider detectors will likely be endowed with. We note that the magnetic bending of muon tracks inside the calorimeter volume is very small in the energy range of our interest (1 TeV and above), and its effect on the regression task is negligible there[2]. In the studies reported *infra* we will both compare the curvature-based momentum estimate provided by an ATLAS-like detector to the radiative losses-driven one, and combine the two to show their complementarity.

2.2 Data generation

We generate unpolarised muons of both charges with a momentum $P = P_z$ in the z direction, of magnitude ranging between 50 GeV and 8 TeV. This interval extends beyond the conceivable momentum range of muons produced by a future high-energy electron-positron collider such as CepC or FCC-ee [33], and it therefore enables an unbiased study of the measurement of that quantity in an experimentally interesting scenario.

The generated initial muon position in the z coordinate is set to $z = -50$ mm with respect to the calorimeter front face; its x and y coordinates are randomly chosen within $|x| \leq 20$ mm and $|y| \leq 20$ mm. The momentum components in x and y direction are set to zero. As mentioned *supra*, to compare the curvature-based and calorimetric measurement we assume that the calorimeter is immersed in a constant $B = 2T$ magnetic field, oriented along the positive y direction. The detector geometry and the radiation pattern of a muon entering the calorimeter are shown in Fig. 2. Even at a relatively low energy of 290 GeV, the produced pattern of radiation deposits is clearly visible. The interaction of the muons with the detector material is simulated using GEANT 4 [34, 35].

For the training and validation tasks of the regression problem a total of 886 716 events are generated, sampled from a Uniform distribution in the 0.05 to 8 TeV range. Additional muon samples, for a total of 429 750 events, are generated at fixed values of muon energy (E=100, 500, 900, 1300, 1700, 2100, 2500, 2900, 3300, 3700, 4100 GeV) in order to verify the posterior distributions in additional tests discussed *infra*, Sec. 4.

[2] A 1 TeV muon traversing a uniform 2-Tesla field for 2.032 m withstands a transverse displacement of 1.24 mm from its original trajectory, which is less than a third of a calorimeter cell.

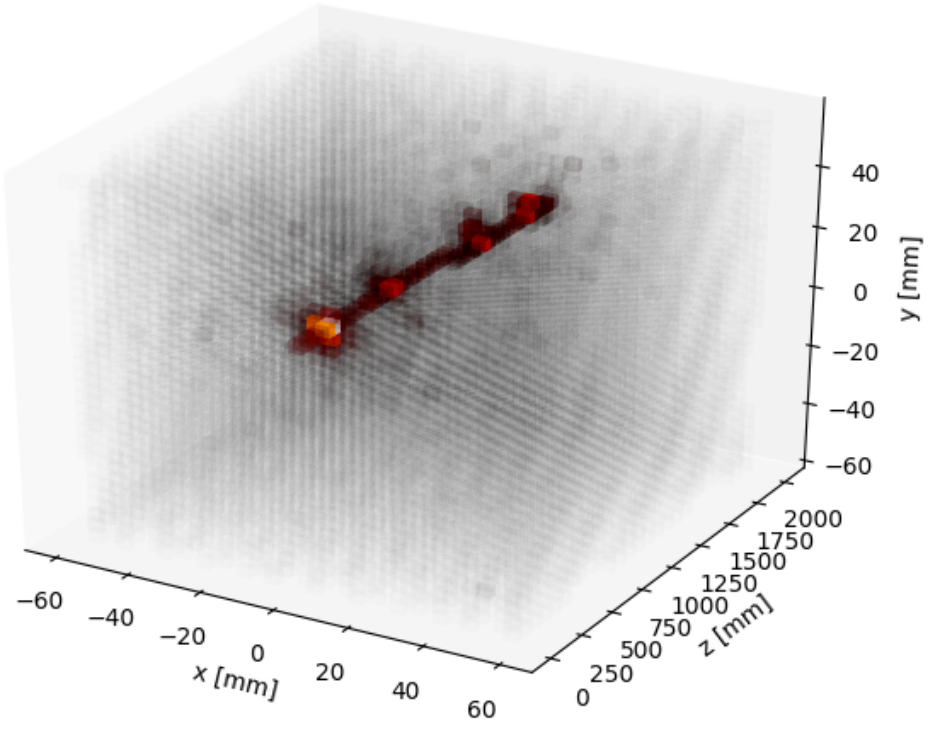


Figure 2: Muon entering the simulated calorimeter in the z direction. The colour palette indicates logarithmic energy deposits of a muon with an energy of approximately 290 GeV*. Black corresponds to zero, red to intermediate, and white to the maximum energy.*

3 The CNN regression task

Three regressor architectures are considered: regressors that only use continuous input-features (such as the energy sum and other high-level features) pass their inputs through a set of fully-connected layers (referred to as the network *body*), ending with a single-neuron output; when the 3D grid of energy deposits is considered, the body is prepended with a series of 3D convolutional layers (referred to as the *head*), which act to reduce the size of the grid, whilst learning high-level features of the data, prior to passing the outs to the body; a hybrid model is also used, in which the energy deposits are passed through the head, and the pre-computed high-level features are passed directly to the body. Layout diagrams for these three models are illustrated in Fig. 3, and a technical description of component is included in the following subsection. Models are implemented and trained using PyTorch [36] wrapped by Lumin [37].

3.1 Architecture components

3.1.1 Convolutional head

The head architecture is inspired by domain knowledge and is based on the fact that the sum of the energy deposits is related to the energy of the traversing muon, however accurate correspondence requires that the deposits receive small corrections based on the distribution of surrounding deposits. The convolutional architecture draws on both the DenseNet [38] and ResNet [39] architectures,

and is arranged in blocks of several layers. Within each block, new channels are computed based on incoming channels (which include the energy deposits) using a pair of 3D convolutional layers. The channels computed by the convolutional layers are weighted by a squeeze-excitation (SE) block [40]. The convolutional plus SE path is by-passable via a residual sum to an identity path. At the output of the block, the channel corresponding to the energy deposits is concatenated (channel-wise) to the output of the addition of the convolutional layers and the identity path [3]. In this way, convolutional layers always have direct access to the energy deposits, allowing their outputs to act as the "small corrections" required.

The architecture becomes slightly more complicated when the energy is downsampled; in such cases, convolutional shortcuts [41] are used on the identity path, and fixed, unit-weighted convolutional layers with strides equal to their kernel size are applied to the energy deposits. These fixed kernels act to sum up the energy deposited within each sub-cube of the detector, and are referred to here as the "E-sum layers". This approach is strongly inspired by [26, 27]. Additionally, for blocks after the very first one, a *pre-activation* layout [41] is adopted with regards to the placement of batch normalisation layers. Figure 4 illustrates and discusses the general configurations of the three types of blocks used. Sets of these convolutional blocks are used to construct the full convolutional head. In all cases, the grid is downsampled four times, each time with a reduction by a factor of two. However, non-downsampling blocks (Fig. 4b) may be inserted in between the downsampling blocks in order to build deeper networks. Figure 5 illustrates the layout of the full convolutional head.

Technical specification In the convolutional layers, the kernel sizes of all convolutional and average-pooling layers are set to three, with the exception of the first convolution in downsampling and initial blocks, which use a kernel size of four, to match the stride and padding of the E-sum layer. Zero-padding of size one is used (two when the kernel size is four). Swish activation-functions [42] are used with $\beta = 1$ (Swish-1). Weights are initialised using the Kaiming rule [43], with the exception of the E-Sum layers, which are initialised with ones. No biases are used.

The squeeze-excitation blocks feed the channel means into a fully connected layer of width $\max(2, N_c//4)$ (N_c = number of channels, // indicates integaer division, Kaiming weight initialisation and zero bias initialisation) and a Swish-1 activation, followed by a fully connected layer of width N_c (Glorot [44] weight initialisation and zero bias initialisation) and a sigmoid activation. This provides a set of multiplicative weights per channel which are used to rescale each channel prior to the residual sum.

Due to the sparse nature of the data, we found in necessary to use *running* batch-normalisation [45]. This modifies the batch normalisation layers to apply the same transformation during both training and inference, *i.e.* during training, the batch statistics are used to update the running averages of the transformation, and then the averaged transformation is applied to the batch (normally only batch-wise statistics are used to transform the training batches, causing potential differences between training and inference computations). Additionally, running averages of the sums and squared sums of the incoming data are tracked, rather than the mean and standard deviation, allowing the true standard deviation to be computed on the fly (normally the average standard deviation is used). Together, these provide greater stability during training, and enabled generalisation to unseen data. All batch normalisation layers use a momentum of 0.1, meaning that the running average of statistic θ is tracked according to $\bar{\theta} \leftarrow 0.9\bar{\theta} + 0.1\theta_{\text{batch}}$.

[3] By convention, the energy channel is kept as the zeroth channel.

3.1.2 Network body and output

The body of the network consists of three fully connected layers, each with 80 neurons. Weights are initialised using the Kaiming rule, and biases are initialised with zeros. Swish-1 activation functions are placed after every layer. No batch normalisation is used.

The output layer of the network consists of a single neuron. Weights are initialised using the Glorot rule, and the bias is initialised to zero. No activation function is used.

3.2 Training

3.2.1 Data

Models are trained on simulated data for the full considered range of muon true energy, 50 to 8000 GeV. The 3D grid of raw energy deposits does not undergo any preprocessing, nor do the target energies. When used, the measured energy extracted from the curvature fit (V[24], see *infra*, Appendix A) is clamped between 0 and 10 TeV [4]. All high-level features are then standardised and normalised by mean-subtraction and division by standard deviation.

The full training dataset consists of 886 716 muons. This is split into 36 *folds* of 24 631 muons; the zeroth fold is used to provide a hold-out validation dataset on which model performance is compared. During training a further fold is used to provide monitoring validation to evaluate the general performance of the network and catch the point of highest performance.

Prior to using the discrete-energy testing-data to compute the resolution, the continuous-energy validation dataset is finely binned in true energy, allowing us to compute an approximation of the resolution at the central energy of the bin (computed as the median true-energy of muons in the bin).

3.2.2 Loss

Models are trained to minimise a Huberised [46] version of the mean fractional squared error (MFSE):

$$\mathcal{L}(y, \hat{y}) = \frac{1}{N} \sum_{n=1}^{N} \frac{(y_n - \hat{y_n})^2}{y_n}, \tag{1}$$

where y is the true muon-energy, $\hat{y}$ is the predicted energy, and N is the batch size. The form of this loss function reflects the expectation of a linear scaling of the variance of the energy measurement with true energy, as is normally the case for calorimeter showers when the energy resolution is dominated by the stochastic term. In this study, the batch size used for training the models is 256.

Huber loss To prevent non-Gaussian tails of the regressed muon energy distribution from dominating the loss estimate, element-wise losses are first computed as the squared error, $(y_n - \hat{y_n})^2$, and high-loss predictions above a threshold are modified such that they correspond to a linear extrapolation of the loss at the threshold:

$$\mathcal{L}_{\text{Huber},i} = t + \left(2\sqrt{t}\left(|y_i - \hat{y}_i| - \sqrt{t}\right)\right), \tag{2}$$

where i are indices of the data-points with a squared-error loss greater than the threshold t. This Huberised element-wise loss is then divided by the true energy to obtain the fractional error, which

[4] The computation code provides a signed energy according to the direction of curvature as dictated by the charge of the muon, but the sign is dropped before using the feature.

is then multiplied by element-wise weights (discussed below) and averaged over the data points in the batch.

Since the loss values vary significantly across the true-energy spectrum, data points are grouped into five equally sized bins, each of which has its own threshold used to define the transition to the absolute error. The transition point used for a given bin is the 68^{th} percentile of the distribution of squared-error losses in that bin (allowing the threshold to always be relevant to the current scale of the loss as training progresses). However, since for a batch size of 256 one expects only 51 points per bin, the threshold can vary significantly from one batch to another. To provide greater stability, the bin-wise thresholds are actually running averages of the past 68^{th} percentiles, again with a momentum of 0.1, *i.e.* for bin j, the threshold is tracked as $t_j \leftarrow 0.9t_j + 0.1\,\mathcal{L}_{\text{SE},j,68^{\text{th}}}$, where $\mathcal{L}_{\text{SE},j,68^{\text{th}}}$ is the 68^{th} percentile of the squared errors in bin j.

Data weighting Models are trained on muons of true energy in the 50 to 8000 GeV range, but will only be evaluated in the range 100 to 4000 GeV in order to avoid biases due to edge effects; effectively the regressor can learn that no targets exist outside of the data range, and so it is more efficient to only predict well within the data-range. This leads to an overestimation of low-energy muons, and an underestimation of high-energy muons. By training on an extended range and then evaluating on the intended range, these edge-effects can be mitigated. Yet we still want the network to focus on the intended range; rather than generating data with a pre-defined PDF in true energy, we use a uniform PDF and down-weight data with true muon energy outside the range of interest. These weights are the element-wise weights mentioned in Sec. 3.2.2.

The weighting function used depends solely on the true energy of the muons and takes the form of:

$$w = \begin{cases} 1 - Sigmoid\left(\frac{E-5000}{300}\right) & E \leq 5000\,\text{GeV}, \\ 1 - Sigmoid\left(\frac{E-5000}{600}\right) & E > 5000\,\text{GeV}. \end{cases} \tag{3}$$

This provides both a quick drop-off above the intended range, and a slow tail out to the upper-limit of the training range. Figure 6 illustrates this weighting function. It should be noted that the above weights correspond to a comparatively smooth modification of the true energy prior; for specific applications where the physics puts hard boundaries on the energy spectrum (such as, *e.g.*, a symmetric electron-positron collider, where one may safely assume that muons cannot be produced with energy larger than the beam energy) a sharper prior may be used instead, and significantly improve the resolution at the high end of the spectrum.

3.2.3 Optimiser

The Adam optimiser [47] is used for updating the model weights. The ϵ and β_2 parameters are kept constant, at 1×10^{-8} and 0.999, respectively. The learning rate (LR) and β_1 (momentum coefficient) are adjusted during training in two stages. For the first 20 epochs of training, the 1cycle schedule [48, 49], with cosine interpolation [50], is used to train the network quickly at high learning rates; this is followed by up to 30 epochs of a step-decay annealing [39], which is used to refine the network at small learning rates. For the 1cycle schedule, training begins at an LR of 3×10^{-7} and $\beta_1 = 0.95$. Over the first two epochs of training the LR is increased to 3×10^{-5} and β_1 is decreased to 0.85. Over the next 18 epochs, the LR is decreased to 3×10^{-6}, and β_1 increased back to 0.95. Following this, the best performing model-state, and its associated optimiser state, is loaded and training continues at a fixed LR and β_1 until two epochs elapse with no improvement in validation loss. At this point, the best performing model-state is again reloaded, β_1 is set to 0.95, and the LR

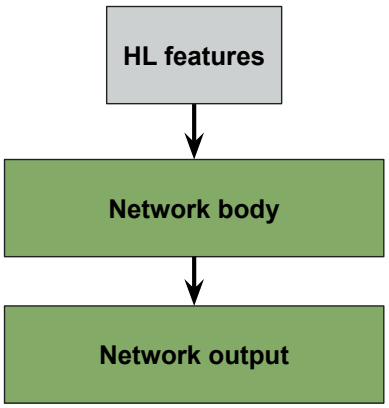
is halved. This process of training until no improvement, reloading, and halving the LR continues until either all 50 epochs have elapsed, or 10 epochs elapse with no improvement. At this point the best performing model-state is again loaded and saved as the final model. Figure 7 details a typical training with such a schedule.

Explicit, tunable regularisation was not found to be required during training. Instead, overtraining is prevented by continual monitoring of the model performance on a separate validation sample, and saving of the model parameters whenever the validation loss improves.

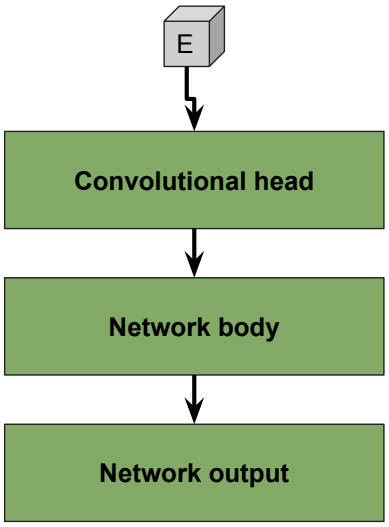
3.2.4 Ensemble training

As mentioned in Sec. 3.2.1, the training dataset is split into 36 folds, one of which is retained to provide a comparison between models, and another is used to monitor generalised performance during training. During development and the ablation study (discussed *infra*, Appendix B), it was useful to obtain an averaged performance of the model architecture from five repeated trainings. Since, however, one training on the full dataset takes about one day, we instead ran these trainings on unique folds of the full dataset, using different folds to monitor generalisation, *i.e.* each model is trained on seven folds and monitored on one fold, and no fold is used to train more than one model (but folds can be used to monitor performance for one model and also to train a different model). This allows us to train an ensemble of five models in just one day, and also to get average performance metrics over the unique validation folds, to compare architecture settings. This method of training is referred to as "unique-fold training".

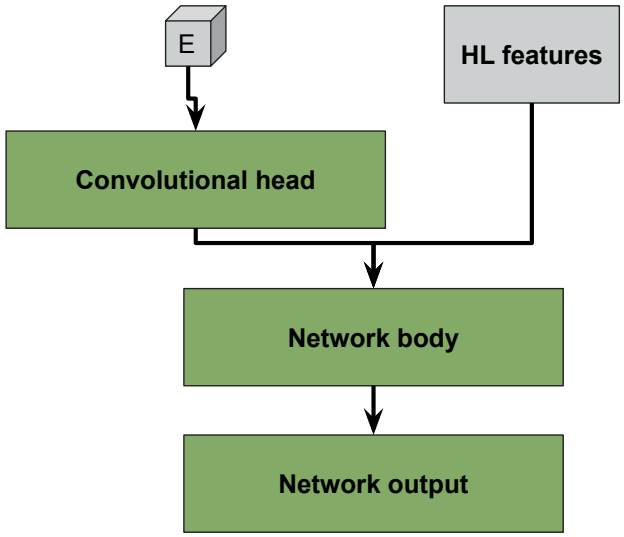
For the full, final ensemble, each model is trained on 34 folds and monitored on one fold, which is different for each model. Once trained, the ensemble is formed by weighting the contributions of each model according to the inverse of its validation performance during training. This method of training is referred to as "all-fold training".



(a) High-level model, in which flat, continuous high-level features are fed directly to the network body

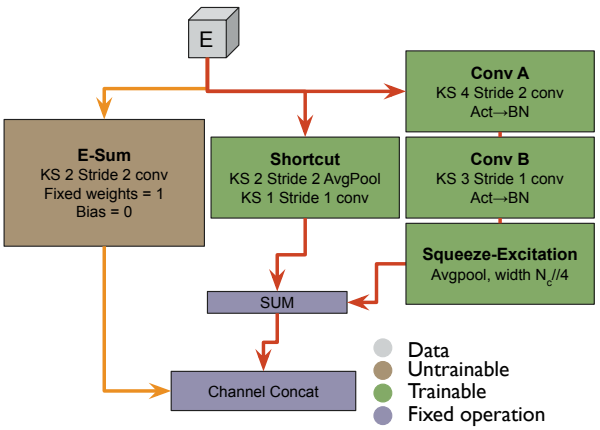


(b) Convolutional model, in which the 3D grid of raw energy deposits is fed through the convolutional head.

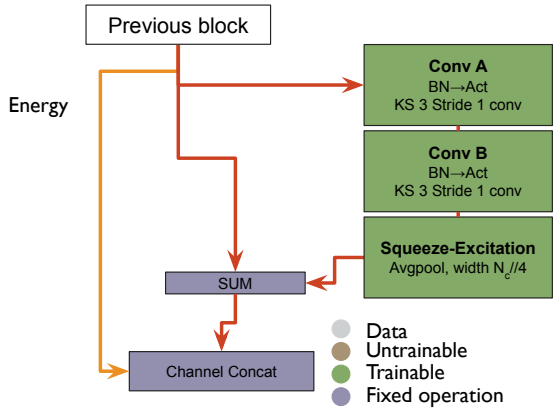


(c) The hybrid model, in which both the 3D grid of raw energy deposits and the high-level features are used.

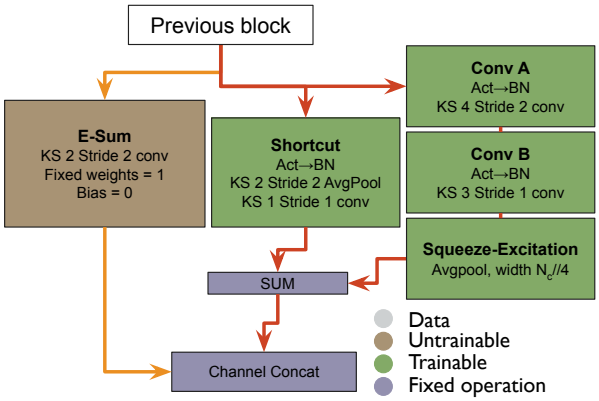
Figure 3: Diagrams illustrating the three types of models used.



(a) Initial block taking the grid of reconstructed energy, E, as input and downsampling. The E-Sum layer uses a kernel size equal to its stride and so reduces the size of the grid by two without missing any deposits. The identity and convolutional paths also use stride two filters in their first layers, in order to produce representations of the same dimensionality. Since downsampling is required, the identity path is a convolutional shortcut, consisting of a stride two average pooling, followed by a stride one convolution. "Act" = activation function, "BN" = batch normalisation, "AvgPool" = average-pooling layer, and "conv" = convolutional layer.



(b) A subsequent block in which no downsampling occurs. The Energy path refers only to the zeroth channel of the output of the previous block. The identity and convolutional blocks act on all channels. The identity path here is straightforward, as no downsampling or changes in number of channels is required. Note that the convolutional path now has the batch normalisation layers (BN) before the activation and convolutional layers, a la pre-activation ResNet.



(c) A subsequent block in which downsampling does occur. This is effectively a pre-activation version of the block illustrated in Fig. (a) , except that E-Sum layer acts only on the zeroth channel of the output of the previous block and the identity path includes activation and batch normalisation layers. The identity and convolutional blocks act on all channels.

Figure 4: Diagrams illustrating the three types of blocks used to construct the convolutional heads.

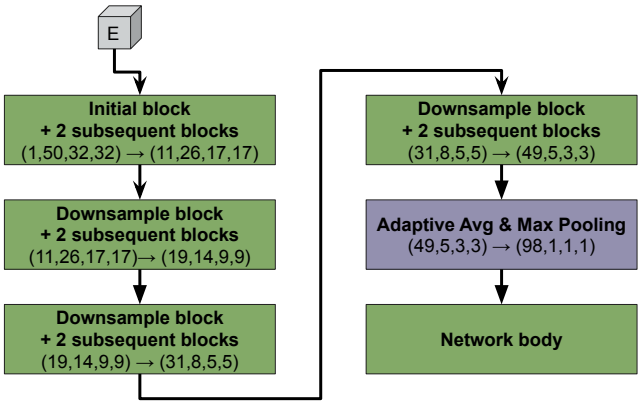


Figure 5: Block layout for the convolutional head. Tensor dimensions are indicated in the form $(channel, z, x, y)$*. The convention is to increase the number of channels to eight in the first downsample, and then increase the number of channels at each downsample by a factor of* 1.5*. The number of channels increases by one in each block due to the energy concatenation. Prior to being fed into the network body, the tensor is pooled by computing the maximum and mean of each channel. The data-batch dimension is not shown, for simplicity.*

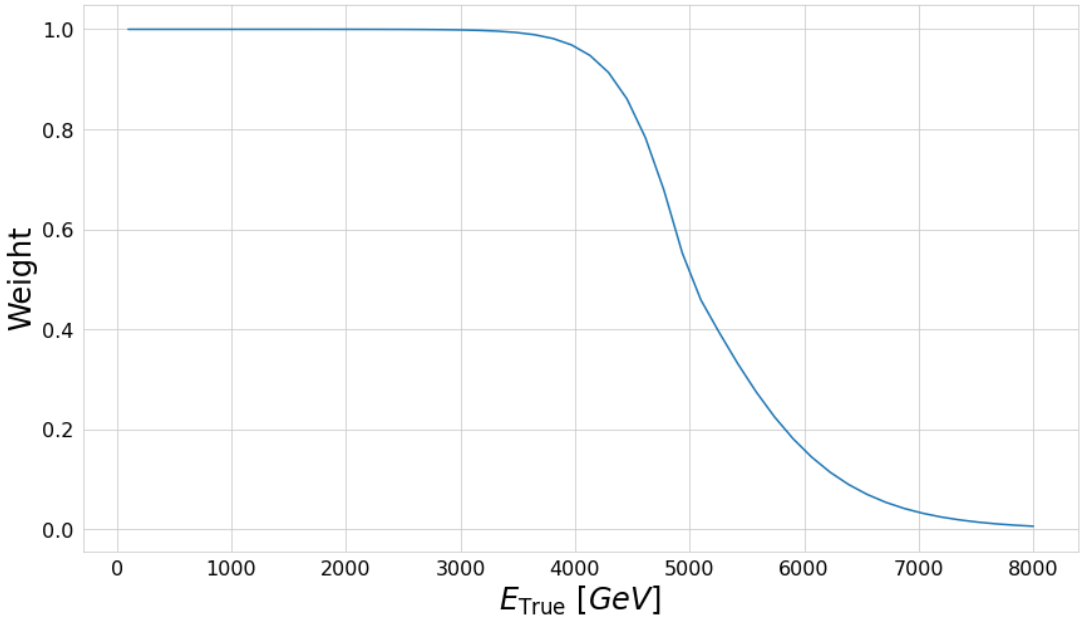
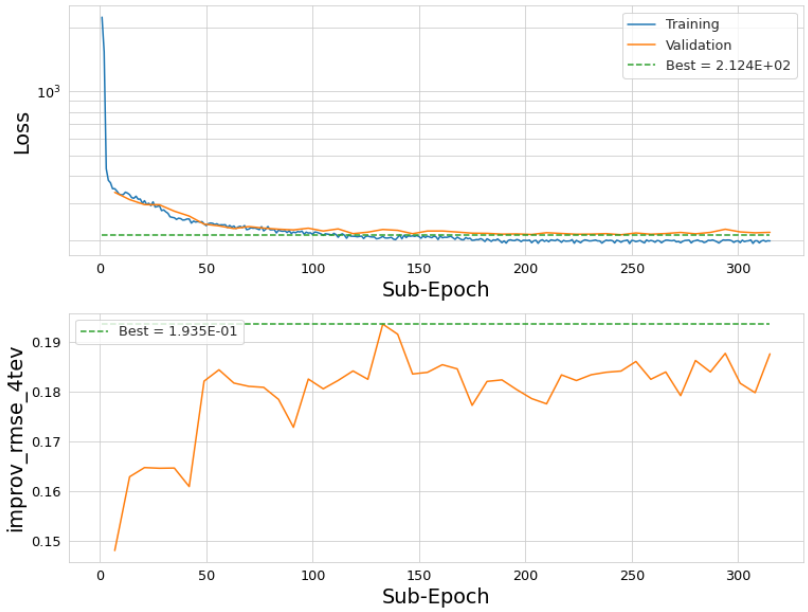
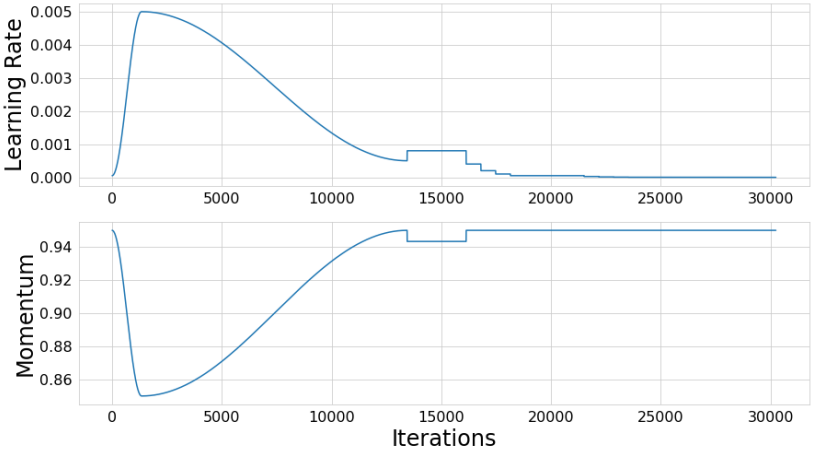


Figure 6: Data weight as a function of true muon energy.



(a) Typical loss evolution during the training of a single model. The evolution of the Mean Improvement is also shown (described in detail in Sec. 4.2). Whilst the MI is what we aim to maximise, it fluctuates too much during training to provide a reliable indication of the point of best performance, and instead the validation loss is used to select the best model.



(b) Learning rate and momentum schedule associated with the training shown in Fig. (a) . Initially, the parameters evolve as per 1cycle with cosine interpolation. Following this fixed period, the best performing models are continually reloaded, and the LR evolves as a step-decay whenever the model fails to improve.

Figure 7: Details of a typical training, showing the loss and metric evolution and the associated schedule of the optimiser hyper-parameters.

4 Results

4.1 Regressor response and bias correction

Figure 8 shows the predictions of the regression ensemble as a function of true energy for the holdout-validation dataset. Whilst the general trend is linear, we can see some dispersion about the central line. Figure 9 better details the fractional error as a function of true energy, along with the trends in the quantiles. From this we can see that regressor overestimates medium energies, and underestimates high energies. Low energies are predicted without significant bias.

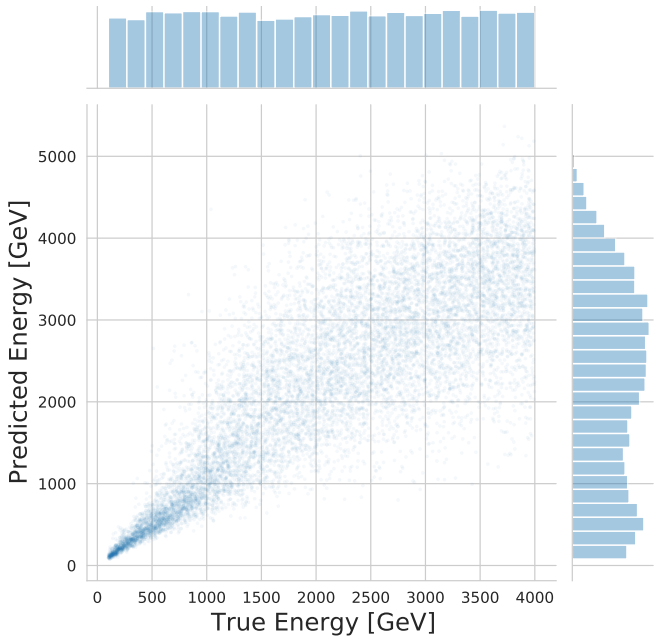


Figure 8: Raw predictions of the regressor ensemble as a function of true energy. The ideal response is for all points to lie on a straight line along $y = x$.

We can correct for the bias in the prediction, however we must do so in a way that does not assume knowledge of the true energy, such that the correction can also be applied to prediction in actual application. The method used is to fit a function (in this case a linear function) to the medians of the predictions in bins of true energy, considering the width of the central 68^{th} percentile in each bin as an estimate of the uncertainty. Having fitted the function, the inverse of the function can now be used to look up the true energy of a given prediction, resulting in a *corrected* prediction. Figure 10 illustrates the fit and subsequent inversion of the linear function to arrive at corrected predictions. Although the difference is only slight, as we will see later in Appendix B, the de-biased predictions allow for a better resolution once the residual biases in the predictions are accounted for. To best reproduce actual application, the debiasing correction is fixed using the validation data, and then applied as is to the testing data.

Figure 11 shows the distributions of the ratios of corrected predictions to true energies on the testing data.

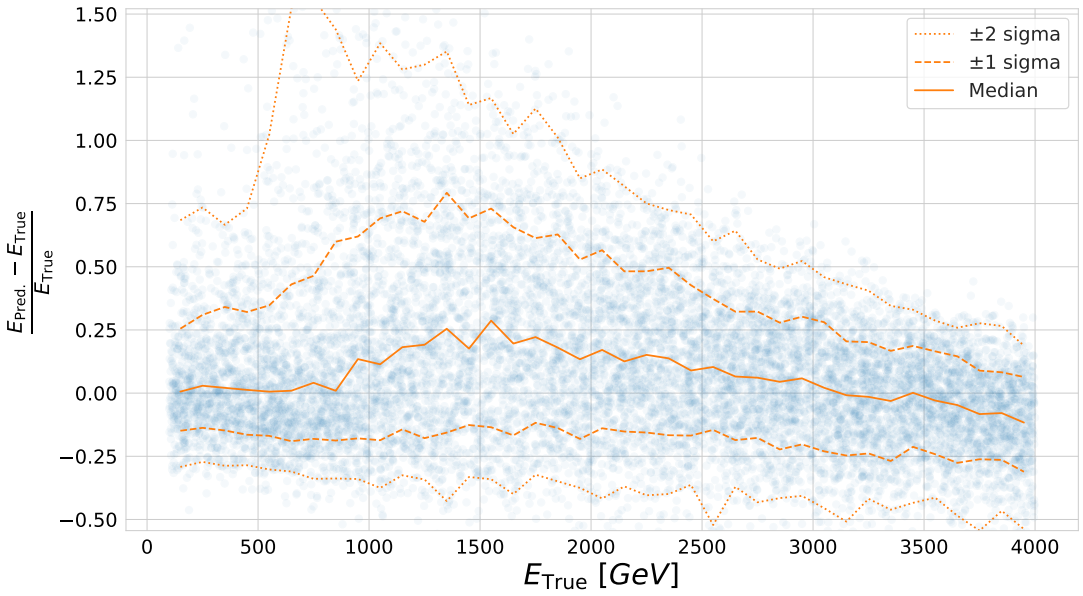


Figure 9: Fractional error of predictions as a function of true energy, along with quantile trends. The ideal response is for all points to lie on a straight line along $y = 0$.

4.2 Resolution and combination with curvature measurement

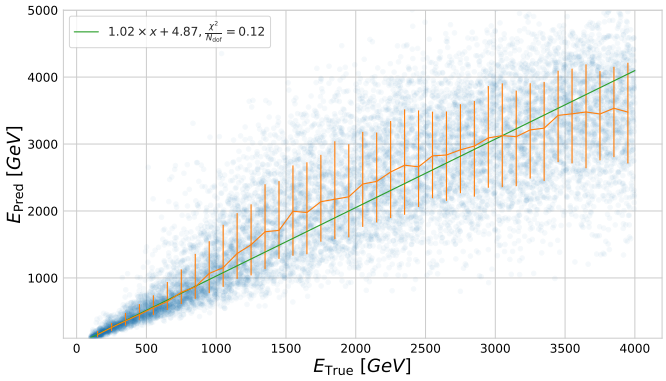
From the discussion in Sec. 1 we can expect that the relative resolution of the energy estimation from the calorimeter should improve as the energy increases, similarly we expect the resolution from magnetic-bending in the tracker will improve as the energy decreases. This difference in energy dependence means that the two measurements are complementary to one another and it would make sense in actual application to use both approaches in a weighted average.

Since our setup only includes a calorimeter, we assume that the resolution of a tracking measurement, performed independently by an upstream or downstream detector, scales linearly with energy, and equals 20 % resolution at 1 TeV. Figure 12 shows the resolution of both the regressor measurement and the simulated tracker measurement, along with the resolution of their weighted average. Resolution here is the fractional root median squared-error computed in bins of true energy according to:

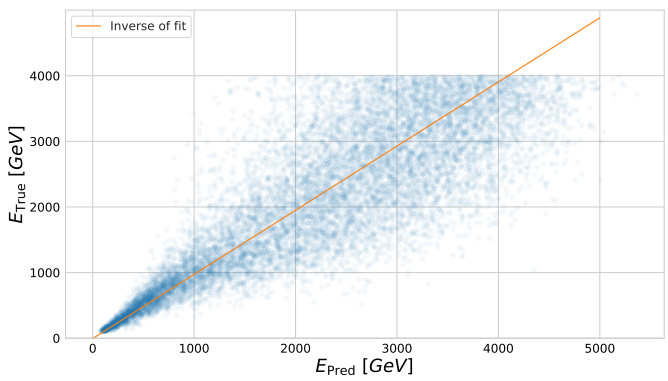
$$\text{Resolution} = \frac{\sqrt{\left(\tilde{E}_p - \tilde{E}_t\right)^2 + \Delta_{68}\left[E_p\right]^2}}{\tilde{E}_t}, \tag{4}$$

where $\tilde{E}_p$ and $\tilde{E}_t$ are the median predicted and true energies in a given bin of true energy (their difference being the residual bias after the correction via the linear fit), and $\Delta_{68}\left[E_p\right]$ is the difference between the 16^{th} and 84^{th} percentiles of the predicted energy in that bin (the central 68^{th} percentile width). When computing the resolution on the testing data (which are generated at fixed points of true energy), $\tilde{E}_t$ is instead the true energy at a given point.

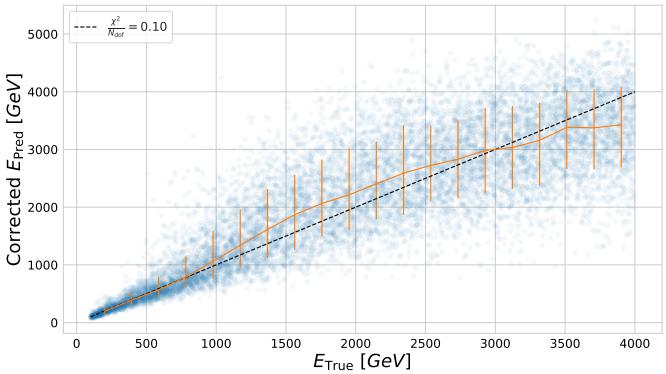
It is interesting to note that the regression resolution initially gets worse with energy, rather than starting poor and gradually improving. Good resolution at low energy was not observed in previous experiments without the magnetic field, therefore we assume that the CNNs are able to make use of the magnetic bending in the calorimeter to recover performance when there is reduced radiation. As expected, the regressor quickly improves in resolution once the energy reaches a certain threshold at



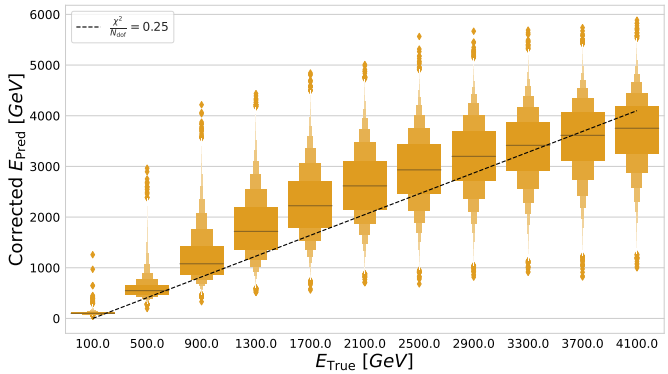
(a) Linear fit to predictions in bins of true energy (green line). The orange line indicates the median prediction in each bin, with asymmetric uncertainty lines indicating the 16^{th} and 84^{th} percentiles.



(b) Subsequent inversion of fit, to look up the true energy as a function of predicted energy.



(c) Corrected predictions on validation data resulting from the inversion of the fit as a function of the true energy. The black dashed line indicates the ideal response, the reduced χ^2 value indicates the agreement between the corrected predictions and the ideal response.



(d) Corrected predictions on testing data. The correction applied is not refitted, and has been fixed using the validation data

Figure 10: Details of the de-biasing fit process.

around 1.5 TeV.

Having established that both the calorimeter and tracker measurements are useful and complementary, for later studies it makes sense to compare models in terms of the performance of the combined measurement. One such metric is the poorest resolution achieved by the combined measurement for the studied energy range (in this case 29.5 % - lower=better). This however relies only on a single point of the response. A more general metric is to compute the improvement of the combined measurement over the tracker-only measurement in bins of true energy, and take the average or sum; this then characterises the improvement due to the regression across the whole spectrum. We will refer to this metric as the Mean Improvement (MI). Considering the 11 points in the range 100 to 4100 GeV, our mean improvement is 22.1 % (higher=better). Computation of the MI on the validation data instead uses 20 bins in the 100 to 4000 GeV.

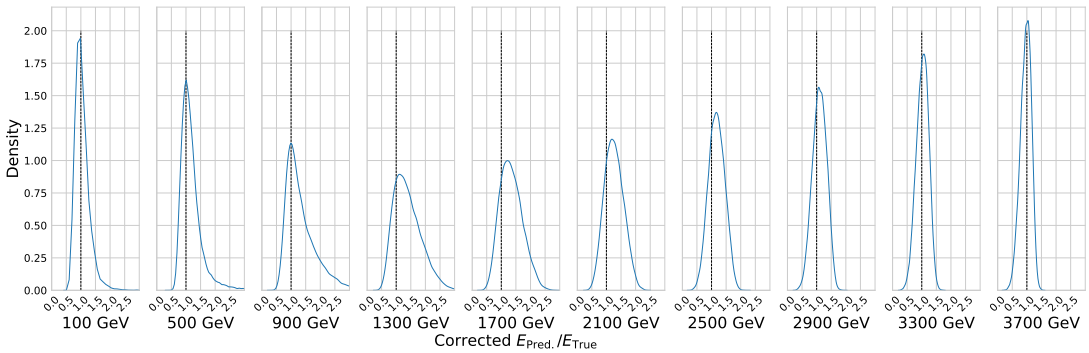


Figure 11: Distributions of the ratios of corrected predictions to true energies in bins of true energy. The ideal response here would be delta distributions centred at one. Distributions not centred at one are indicative of residual bias in the predictions for that energy range.

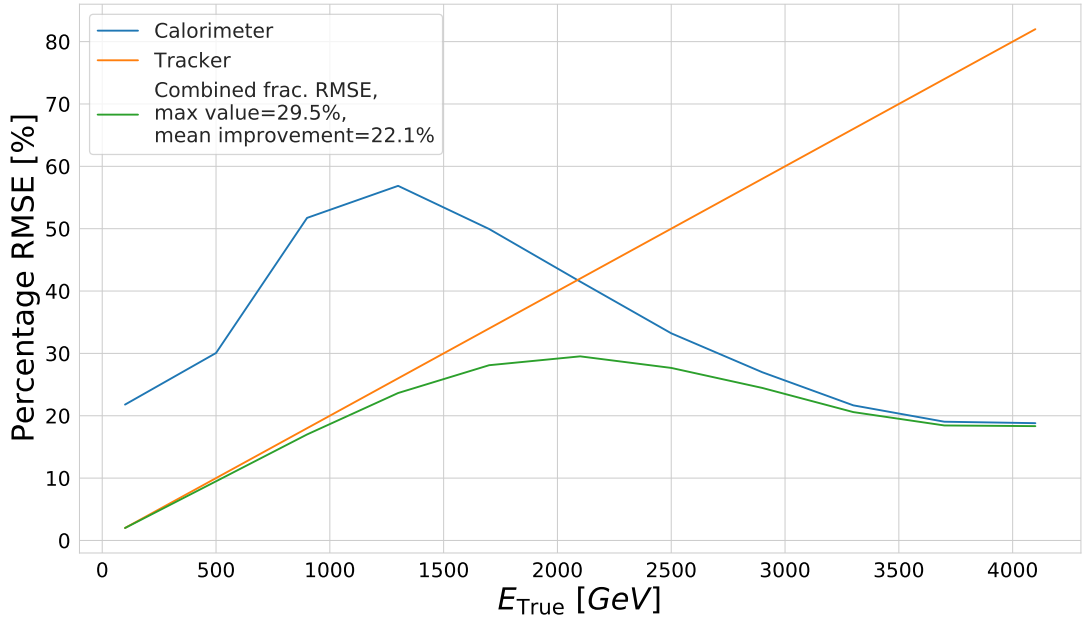


Figure 12: Resolutions of the energy regression (Calorimeter), the simulated tracker, and their weighted average in a combined measurement. Resolution is computed on testing data at fixed points of true energy. The tracker is assumed to provide a linearly scaling resolution with a relative value of 20 % *at* 1 TeV.

Input comparison: high-level features and raw inputs As discussed in Appendix A, alongside the recorded calorimeter deposits, a range of high-level (HL) features are also fed into the neural network. To better understand what, if anything, the CNN learns extra from the raw information, we can study what happens when the inputs are changed. In Tab. 1 we show the MI metric values for a range of different inputs. In cases when the raw inputs are not used, the neural network (NN) consists only of the fully connected layers. For this comparison, we use the MI computed during training on the monitoring-validation dataset and average over the five models trained per configuration.

Inputs	MI	Change in MI [%]
Raw inputs + HL feats.	20.30 ± 0.08	N/A
Raw inputs only	19.53 ± 0.06	-3.8 ± 0.4
HL-feats. only	17.60 ± 0.08	-13.3 ± 0.6
Energy-sum only	14.98 ± 0.05	-26.2 ± 0.5

Table 1: Mean Improvements for a range of different input configurations. The MI is computed on the monitoring-validation data and averaged over the training of five models per configuration. The change in MI is computed as the difference between configuration and the nominal model ("Raw inputs + HL feats.") as a fraction of the MI of the nominal model. Energy-sum features are the three features corresponding to the sums of energy in different threshold regions (V[0], V[26], & V[27]).

From these results we can see the CNN is able to extract more useful information from the raw data that our domain expertise provides, however we are still able to help the model perform better when we also leverage our knowledge. Moreover, we can see that the additionally computed HL-features provide a significant benefit to the energy-sum features. The importance of the top features as a function of true energy is illustrated in Fig. 13. From this, it is interesting to note the shift in importance between V[0] and V[26] (the sum of energy in cells above 0.1 GeV and below 0.01 GeV, respectively; see Appendix A, *infra*). The fact that the HL-features give access to a finer-grained summation of energy than the energy-pass-through connections in the CNN architecture (which sum all the energy in cells within the kernel-size, regardless of energy) is potentially why the HL-features are still useful; a further extension to the model, then, could be to also perform the binned summation during the energy pass-through.

Figure 14 shows the resolutions of the four different models on the holdout-validation data. From this we can clearly see the benefits of providing access to the raw-hit data. The benefits of the high-level features are most prominent in the low to medium energy range, where features V[0] and V[26] have very similar importance.

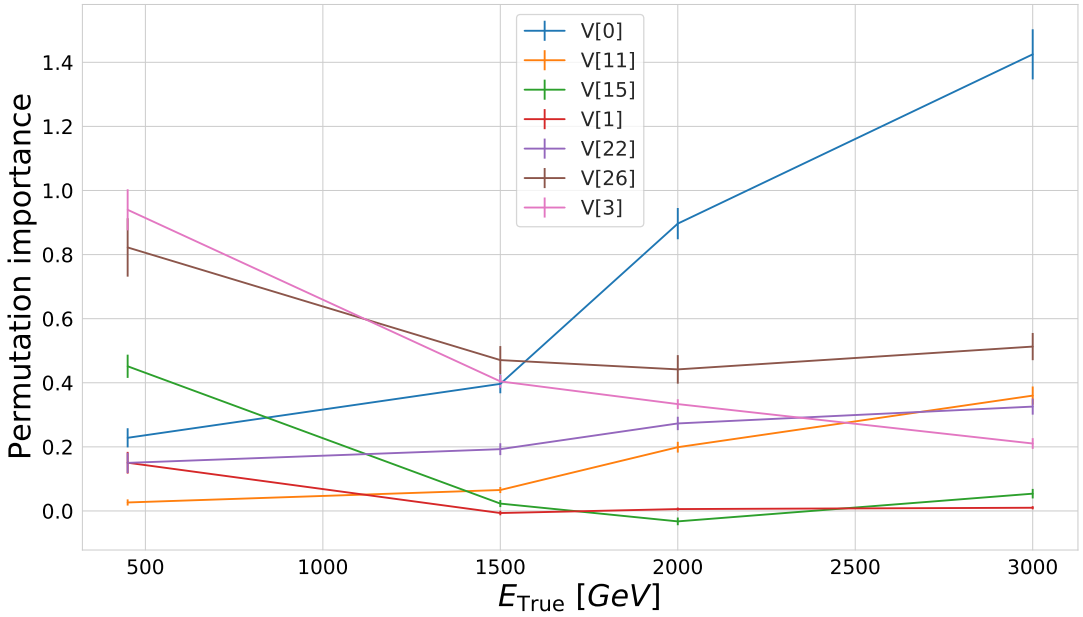


Figure 13: Feature importance of the most important features as evaluated using the "HL-feats. only" model in bins of true energy. The features, described further in Appendix A are: V[0] - E-sum in cells above 0.1 GeV*, V[1] - fractional MET, V[3] - overall* 2^{nd} *moment of transverse E distribution, V[11] - maximum total E in clustered deposits, V[15] - maximum energy of cells excluded from clustered deposits, V[22] - relative* 1^{st} *moment of E distribution along x-axis, V[26] - E-sum in cells below* 0.01 GeV*.*

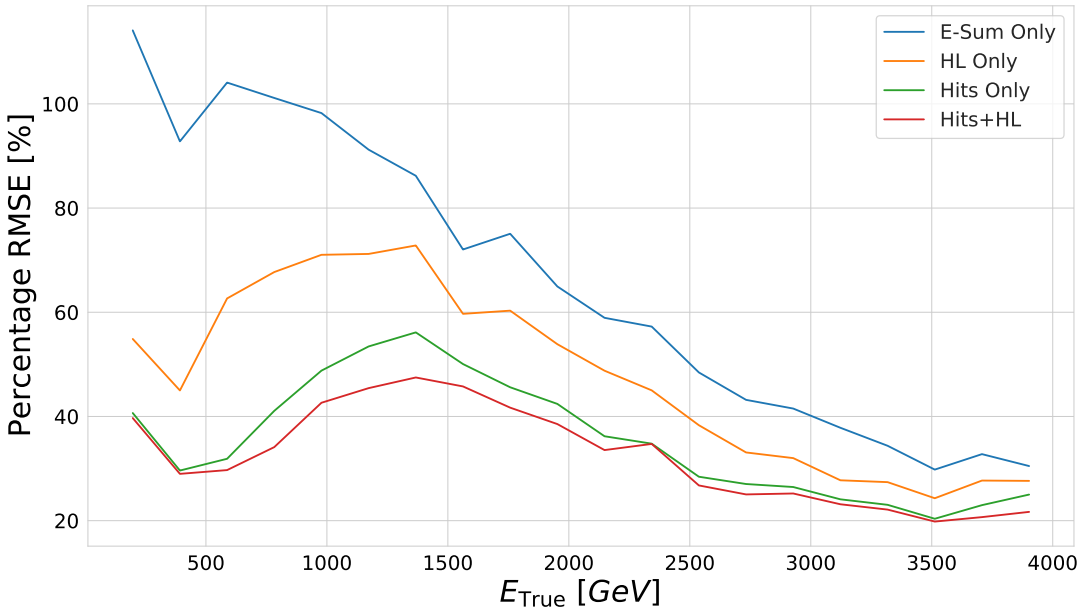


Figure 14: Resolutions of the models with varying input menus. Resolution is computed on the holdout-validation data in bins of true energy.

5 Conclusions

As we move towards the investigation of the potential of new accelerators envisioned by the recently published "2020 Update of the European Strategy for Particle Physics" [51], we need to ask ourselves how we plan to determine the energy of multi-TeV muons in the future detectors which those machines will be endowed with. As mentioned *supra* (Sec. 1), the CMS detector is able to achieve relative resolutions in the range of 6 to 17 % at 1 TeV, thanks to its very strong 4-Tesla solenoid. It is important to note that the choice of such a strong magnet for CMS imposed a compact design to the whole central detector; the result proved successful at the LHC, but might be sub-optimal in other experimental situations. Given the linear scaling with momentum of relative momentum resolution as determined with curvature fits, it is clear that complementary estimates of the energy of high-energy muons would be highly beneficial in future experiments.

In this work we investigated, using an idealised calorimeter layout, how spatial and energy information on emitted electromagnetic radiation may be exploited to obtain an estimate of muon energy. Given the regularity of the detector configuration, processing of the raw data was possible using 3D convolutional neural networks. This allowed us to exploit the granular information of the deposited energy pattern to learn high-level representations of the detector readout, which we could also combine with high-level information produced by physics-inspired statistical summaries. We found the use of deep learning and domain-driven feature engineering to both be beneficial. In Appendix B we further explore the CNN architecture and training loss, finding there too, that using knowledge of the physical task can help inspire more performant solutions.

Our studies show that the fine-grained information on the radiation patterns allows for a significant improvement of the precision of muon energy estimates. *E.g.* for muons in the 1 to 3 TeV range, which are the ones of higher interest for future applications, the relative resolution improves approximately by a factor of two with respect to what can be achieved by only using the total energy release (see Fig. 14). A combination of such information with that offered by a curvature measurement, such as a resolution term of the form $\delta P = 0.2P$ (with P in TeV) which can typically be enabled by tracking in a $B = 2\,\mathrm{T}$ magnetic field, may keep the overall relative resolution of multi-TeV muons below 30 % across the spectrum, and achieve values below 20 % at 4 TeV (see Fig. 12).

Acknowledgements

A significant fraction of computational and storage resources used in this investigation were provided by CloudVeneto; we thank the them and their support team not only for the compute offered, but the high up-time.

Appendices

A High-level event features

The regression task we set up in Sec. 3 uses 28 global features extracted by combining spatial and energy information collected in the calorimeter cells. In this section we describe how those features are calculated.

Some of the features describe general properties of the energy deposition (*e.g.*, the sum of the signal in all cells recording energy above or below a $E_{thr} = 0.1\,\text{GeV}$ threshold), while others are fully reliant on fine-grained information (moments of the energy distribution, in five regions of detector depth: $z < 400\,\text{mm}$, $400 < z < 800\,\text{mm}$, $800 < z < 1200\,\text{mm}$, $1200 < z < 1600\,\text{mm}$, and $z > 1600\,\text{mm}$; and imbalance of the deposited energy in the transverse plane). A few more variables describe the result of a clustering of the energy deposits, which is briefly described in Sec. A.1 *infra*. A final set of features described in Sec. A.2 are specifically constructed to leverage the magnetic field and estimate the curvature of muons by detecting the spread of the radiation pattern along the x coordinate caused by the small bending along x that muons of sub-TeV energy follow as they penetrate in the calorimeter. Below we discuss in detail how the features are computed.

A.1 Clustering of calorimeter cells

The small size of calorimeter cells (which span 0.24 radiation lengths in x and y, and 4.5 radiation lengths in z) implies that photons of energy large enough to produce showers by pair production will produce a signal in multiple cells, especially if they are emitted with non-null angles with respect to the z direction. Given that all the information on radiation emission by the muon is possessed by primary photons, it seems reasonable to try and decipher the pattern of emitted radiation by aggregating the granular cell-based information into clusters, whose properties may constitute useful statistical summaries to complement the full resolution of the calorimeter.

We set a minimum threshold $E_{thr} = 0.1\,\text{GeV}$ for the energy recorded in cells elected as seeds for the clustering procedure. The search for clusters starts with seed cells belonging to the column of same transverse coordinates x and y of the incident muon[5], and performs the following calculations:

1. The highest-energy cell is selected as a seed if it has $E > E_{\text{thr}}$;
2. The six calorimeter cells adjacent in either x, y, or z to the seed cell are added to the cluster if they recorded a non-null energy deposition;
3. Cells with non-null energy deposition are progressively added to the cluster if they are adjacent to already included cells;
4. The final cluster is formed when there are no more cells passing the above criteria; at that point, features such as the number of included cells and the total cluster energy are computed (see below).
5. All cells belonging to the cluster are removed from the list of unassigned cells;
6. The algorithm returns to step 1 to form other clusters.

[5]The impact position of muons is well determined as that of the xy position whose z-integrated recorded energy is the highest, but we assume here that we know it from a tracking detector located upstream, with no loss of generality.

Once clusters seeded by the column of cells along the muon trajectory are formed by the above procedure, a second set of clusters is constructed using cells yet unassigned to any cluster:

1. The highest-energy cell above $E_{\rm thr}$ is considered, irrespective of its x, y coordinates;
2. The six calorimeter towers adjacent in either x, y, or z to the seed cell are added to the cluster if they recorded a non-null energy deposition;
3. Cells with non-null energy deposition are progressively added to the cluster if they are adjacent to cells already included;
4. The final cluster is formed when there are no more cells passing the above criteria; features are then computed for the identified cluster;
5. All cells belonging to the cluster are removed from the list of unassigned cells;
6. The algorithm returns to step 1) to search for additional clusters.

Using the results of the above two-step clustering procedure, we define the following high-level features:

- V[9]: The number of muon trajectory-seeded clusters (type-1 clusters);
- V[10]: The maximum number of cells among type-1 clusters;
- V[11]: The maximum total energy among type-1 clusters;
- V[12]: The maximum extension along x of type-1 clusters;
- V[13]: The maximum extension along y of type-1 clusters;
- V[14]: The maximum extension along z of type-1 clusters;
- V[16]: Average number of cells included in type-1 clusters.
- V[17]: The number of clusters seeded by a cell not belonging to the muon trajectory (type-2 clusters);
- V[18]: The maximum number of cells among type-2 clusters;
- V[19]: The maximum total energy among type-2 clusters;
- V[20]: Ratio between maximum energy and maximum number of cells of type-2 clusters;
- V[21]: Average number of cells included in type-2 clusters.

Finally, some cells may remain non associated to any type-1 or type-2 clusters. To extract further information from them, we search for the $3\times3\times3$ cube of 27 cells in x,y,z which captures the highest total energy among cells still not included in clusters (V[25]), and the second-highest total energy (V[15]). These two features are listed *infra*. Standardised distribution of these features, along with the others defined in this Appendix, are shown in Figs. 15-17; correlations with muon energy are shown in Figs. 18-20.

A.2 Measuring curvature with energy deposits

Muons entering our simulated calorimeter do so with an initial trajectory orthogonal to the calorimeter front face [6]. From that point on, they undergo interactions with the material, as well as a bending Lorentz force. If we ignore all physical effects except the magnetic bending, which we wish to estimate, we may model the muon track as an arc of circumference in the xz plane. At the back face of the calorimeter, the expected deviation of such a circumference from a straight line oriented along z is very small in absolute terms: for a muon of momentum P in GeV in a magnetic field B in Tesla, the curvature of the trajectory is $R = P/(0.3B)$ meters, hence the estimated deviation is $\Delta_x = R - \sqrt{R^2 - \Delta_z^2}$, where $\Delta_z = 2\,\mathrm{m}$ is the calorimeter depth along z. Assuming, *e.g.*, $P = 600\,\mathrm{GeV}$ we find a curvature $R = 1000\,\mathrm{m}$ and from it a displacement $\Delta_x = 2\,\mathrm{mm}$, which is already smaller than the calorimeter granularity.

In constructing a variable sensitive to curvature, we observe that circular trajectories that start orthogonal at the front face of the calorimeter may in principle be determined by measuring any two points along their path in the lead tungstate material. We further notice that while calorimeter cells traversed by the muon track usually collect a detectable amount of energy from ionization processes, they are not the only ones carrying information on the muon trajectory. In fact, for muons that bend very little in the magnetic field, the process of muon radiation in a homogeneous medium is dominated by brehmsstrahlung originating by multiple scattering processes. In the plane orthogonal to the muon trajectory the direction of the emitted photons is thus largely random, but these photons do not travel very far before depositing their energy in calorimeter cells. Hence the position of additional cells lit up by photons traveling away from the muon trajectory contains a good deal of extra information on the position of the radiating particle.

We construct a statistical estimator of the muon curvature by determining two separate points in the xz plane, using separately the first 25 and the second 25 layers of crystals in z. We compute the following weighted averages:

$$\begin{aligned}
\hat{x_1} &= \frac{\sum_{i_1} E_i x_i w_i}{\sum_{i_1} E_i w_i} \\
\hat{z_1} &= \frac{\sum_{i_1} E_i z_i w_i}{\sum_{i_1} E_i w_i} \\
\hat{x_2} &= \frac{\sum_{i_2} E_i x_i w_i}{\sum_{i_2} E_i w_i} \\
\hat{z_2} &= \frac{\sum_{i_2} E_i z_i w_i}{\sum_{i_2} E_i w_i}
\end{aligned}$$

where the sums over indices i_1 run on calorimeter cells in the first 25 layers, and sums over indices i_2 run on calorimeter cells in the second 25 layers along z, and where weights w_i are defined as follows:

$$w_i = exp(-(|y_i - y_\mu|)/50)$$

with mm units, and where y_μ is the center of the towers in the y plane containing the highest amount of measured energy. In other words, calorimeter cells are assumed to contain information on the xz position of the radiating particle in proportion to their detected energy, and inversely proportional to the distance of the cell to the y coordinate at which the particle track lays.

The two points (x_1, z_1), (x_2, z_2) in the xz plane allow the construction of an estimator for the radius of the muon trajectory: we first specify the equation of a circumference as

[6] A very small initial bending is produced for low-energy muons in traversing the first 50mm from the point of origin to the calorimeter front face.

$$(x - x_0)^2 + z^2 = R^2$$

from which we get

$$x_0 = \frac{{z_2}^2 - {z_1}^2 + {x_2}^2 - {x_1}^2}{2(x_2 - x_1)}$$

and from it the radius estimator as

$$R = \sqrt{{x_1}^2 - 2x_1x_0 + {x_0}^2 + {z_1}^2}$$

Variable $V[24]$ is then defined as $V[24] = 0.3BR$. It provides useful information for muon momenta below about 500 GeV, as can be seen in Fig. 20.

A.3 Description of other global features

We list below the other features we compute for each event:

- V[0]: The total energy recorded in the calorimeter in cells above the $E_{\text{thr}} > 0.1\,\text{GeV}$ threshold;
- V[1]: the ratio between the missing transverse energy deposition in the xy plane and the total energy, computed as $V[2] = \frac{\sqrt{E_x^2 + E_y^2}}{\sum_i E_i}$, where $E_x = \sum E_i \Delta x_i$, $E_y = \sum E_i \Delta y_i$, and Δx_i, Δy_i are the spatial distances in the transverse plane to the center of the cell hit by the muon at the calorimeter front face; in this calculation, all cells are used;
- V[2]: This variable results from the same calculation extracting $V[1]$, but it is performed using in all sums only towers exceeding the $E_{\text{thr}} = 0.1\,\text{GeV}$ threshold;
- V[3]: The second moment of the energy distribution around the muon direction in the transverse plane, computed with all towers as $V[3] = \sum_i [E_i(\Delta x_i^2 + \Delta y_i^2)] / \sum_i E_i$, where indices run on all towers and the distances are computed in the transverse plane, as above;
- V[4]: The same as V[3], but computed only using towers located in the first 400 mm-thick longitudinal section of the detector along z;
- V[5]: The same as V[3], but computed only using towers in the $400 < z_i < 800\,\text{mm}$ region;
- V[6]: The same as V[3], but computed only using towers in the $800 < z_i < 1200\,\text{mm}$ region;
- V[7]: The same as V[3], but computed only using towers in the $1200 < z_i < 1600\,\text{mm}$ region;
- V[8]: The same as V[3], but computed only using towers in the $z_i \geq 1600\,\text{mm}$ region;
- V[9]-V[14] and V[16]-V[21]: See *supra* (Sec. A.1);
- V[15]: Second-highest maximum energy in a $3 \times 3 \times 3$ cubic box from cells not included in type-1 or type-2 clusters;
- V[22]: The first moment of the energy distribution along the x axis, relative to the x position of the incoming muon track;
- V[23]: The first moment of the energy distribution along the y axis, relative to the y position of the incoming muon track;

- V[24]: See *supra* (Sec. A.2);
- V[25]: Maximum energy in a $3 \times 3 \times 3$ cubic box from cells not included in type-1 or type-2 clusters;
- V[26]: Sum of energy recorded in cells with energy below 0.01 GeV;
- V[27]: Sum of energy recorded in cells with energy between 0.01 and $E_{thr} = 0.1$ GeV.

The correlation matrix of the 28 variables is shown in Fig. 21.

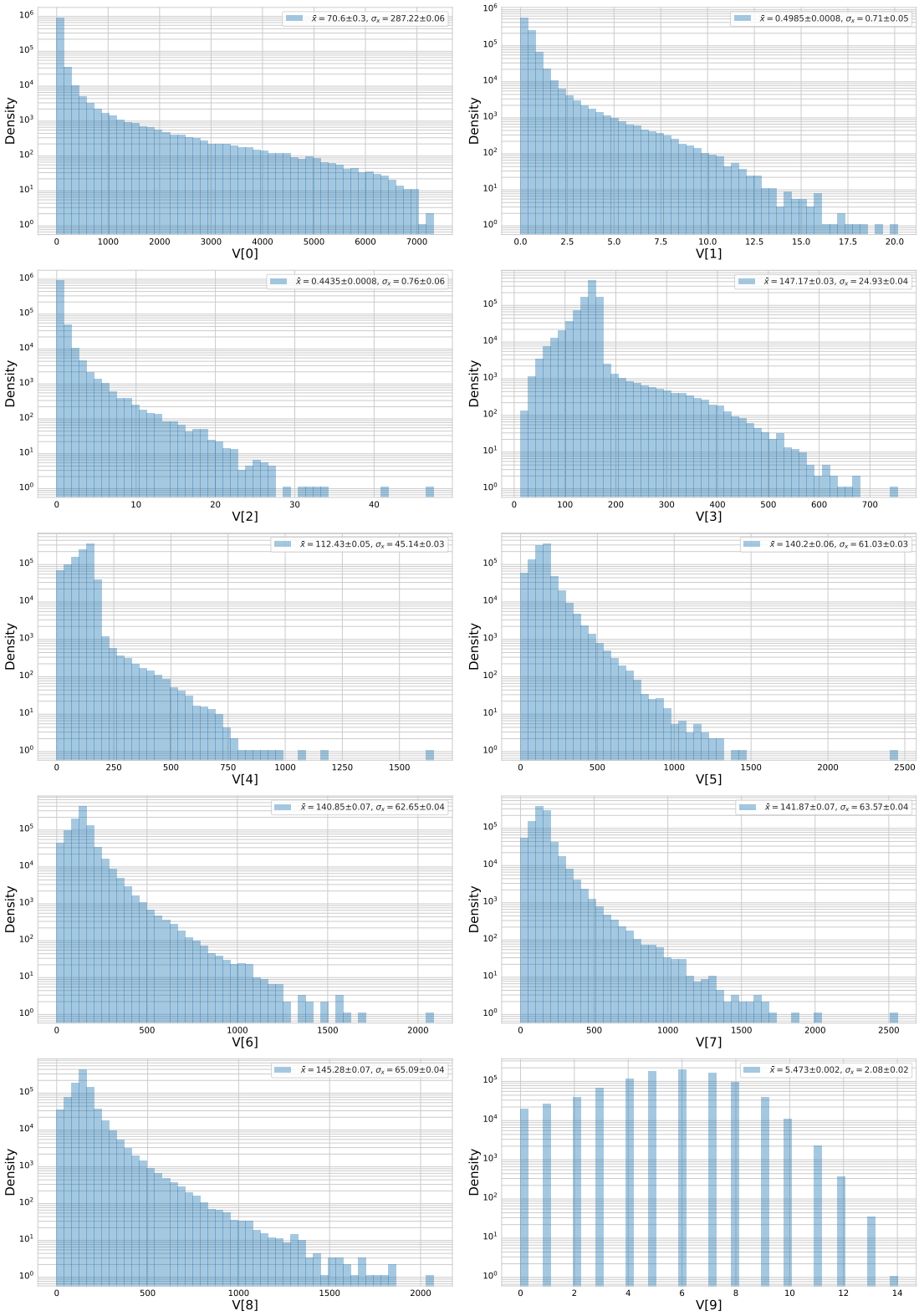


Figure 15: Marginals of features V[0] to V[9]. Features are defined in Section A.3.

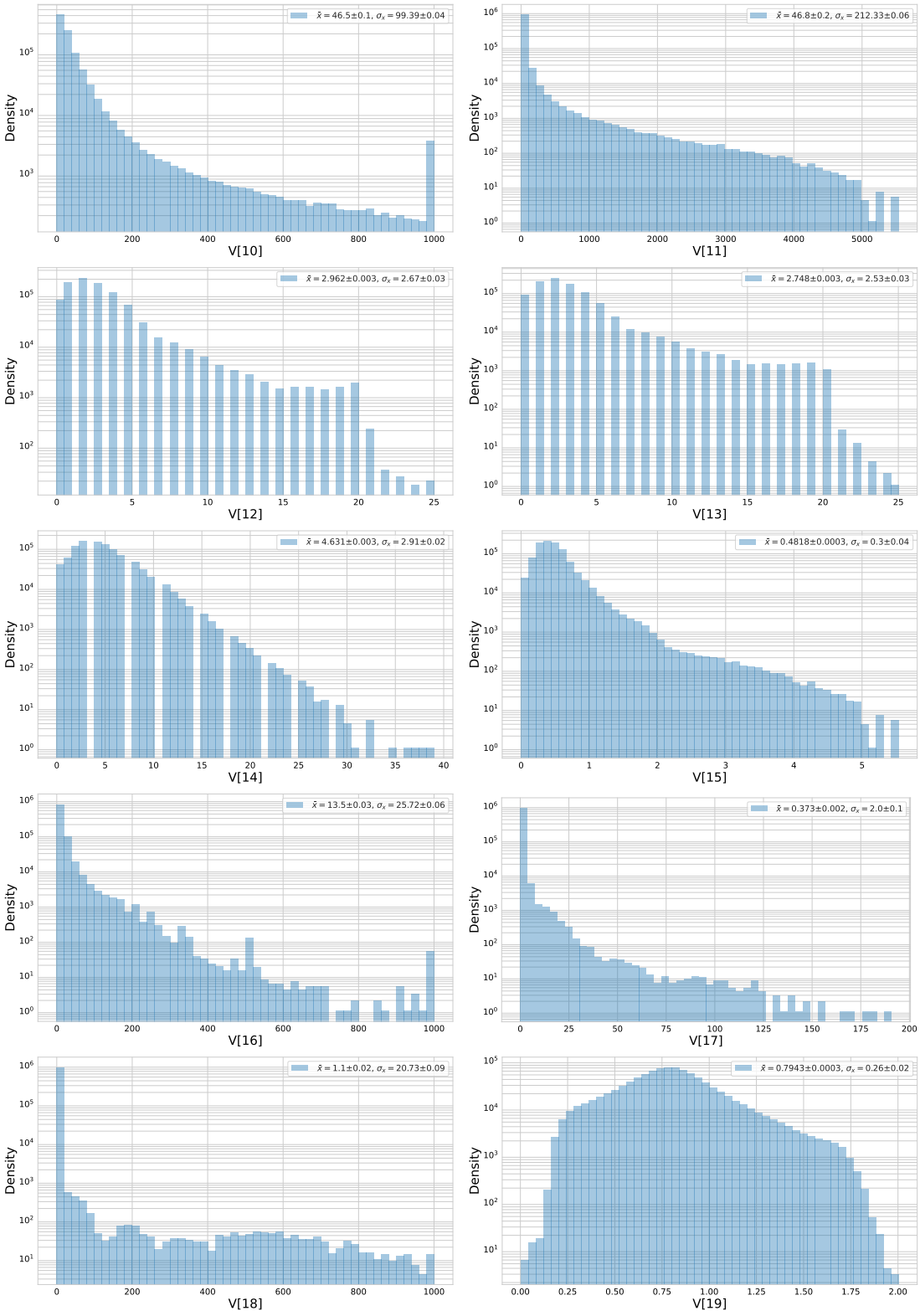


Figure 16: Marginals of features V[10] to V[19]. Features are defined in Section A.3.

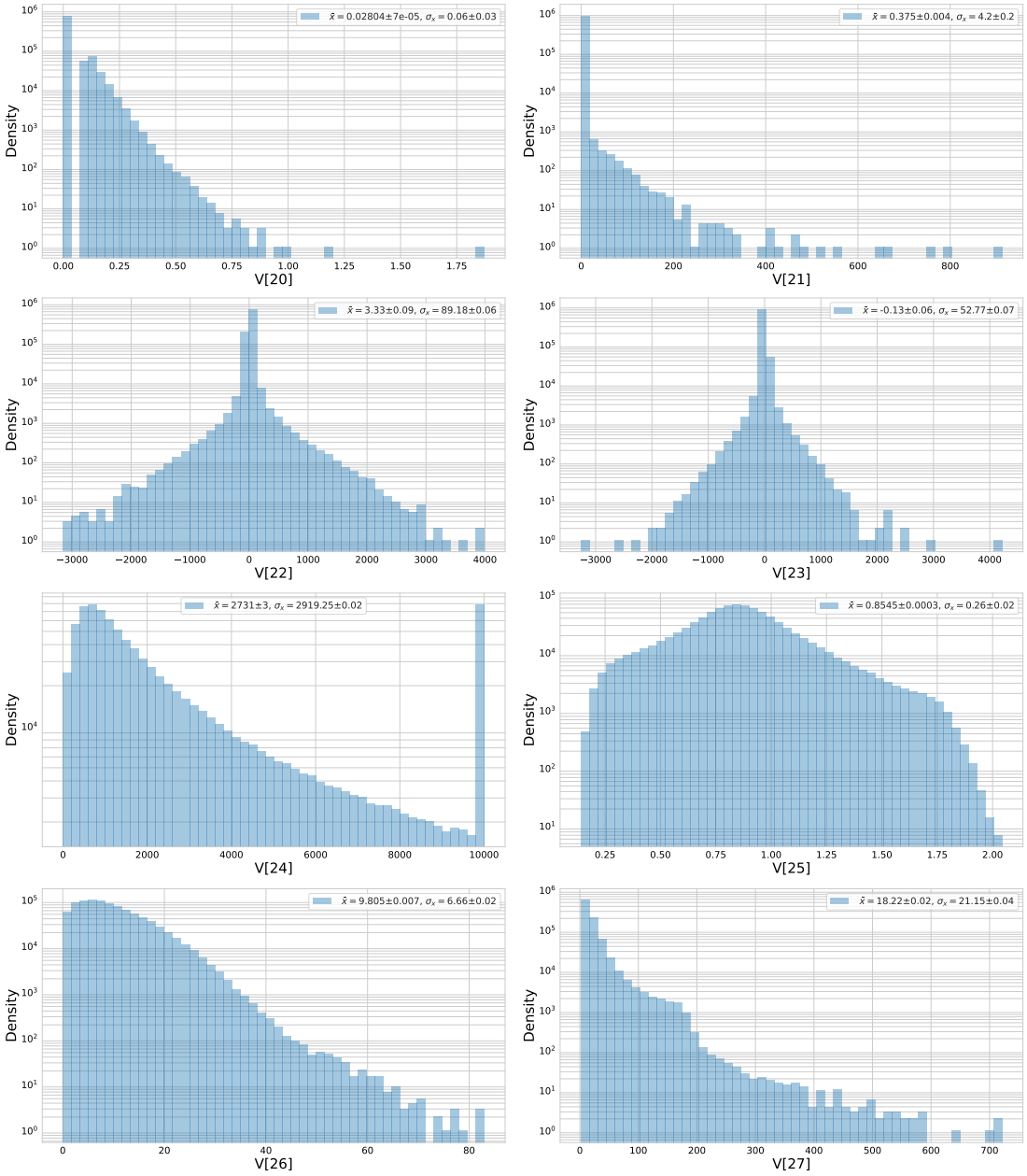


Figure 17: Marginals of features V[20] to V[27]. Features are defined in Section A.3.

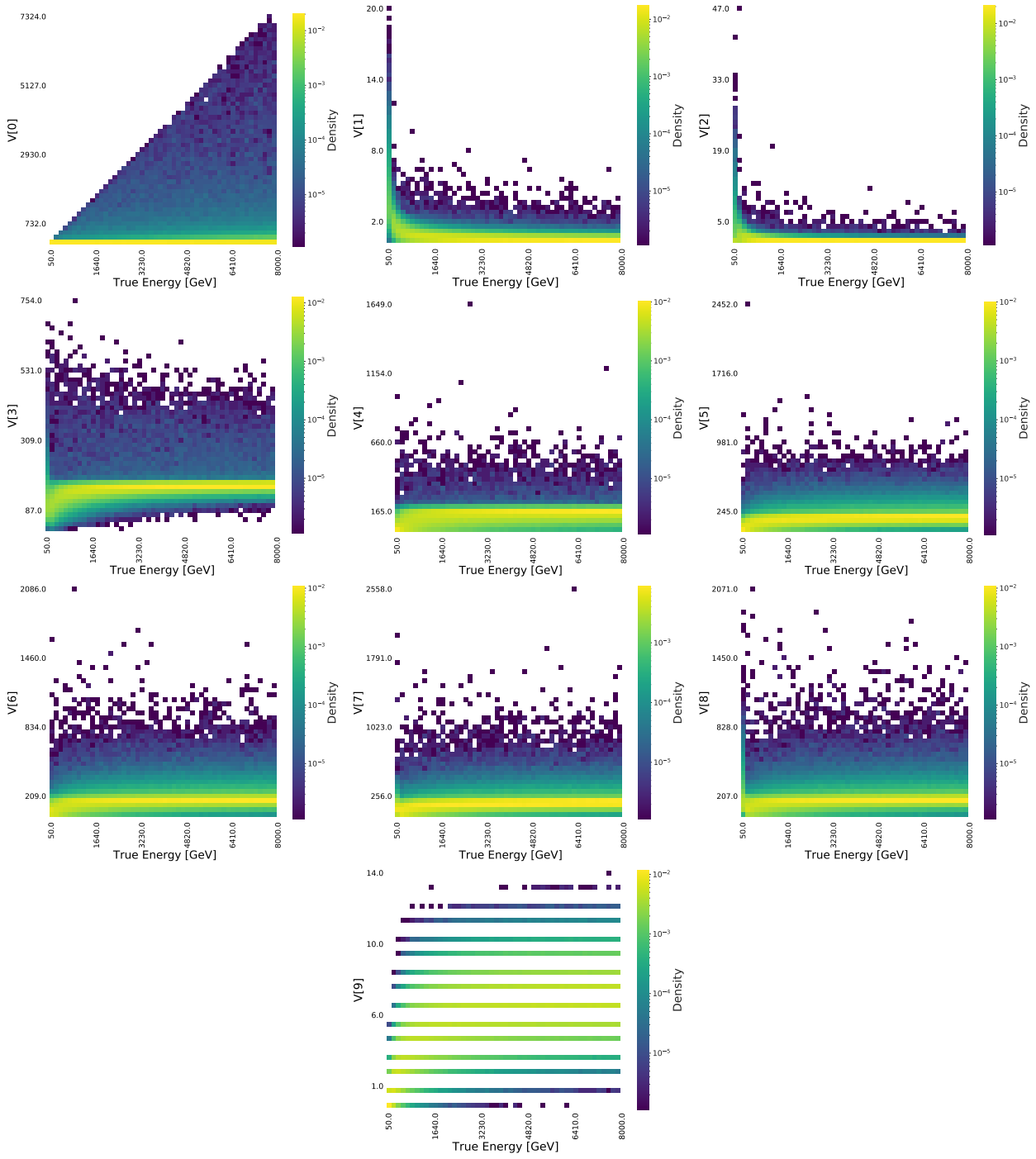


Figure 18: 2D histograms showing the dependence of features V[0] to V[9] (on the y axes) on true muon energy (on the x axes). Features are defined in Section A.3.

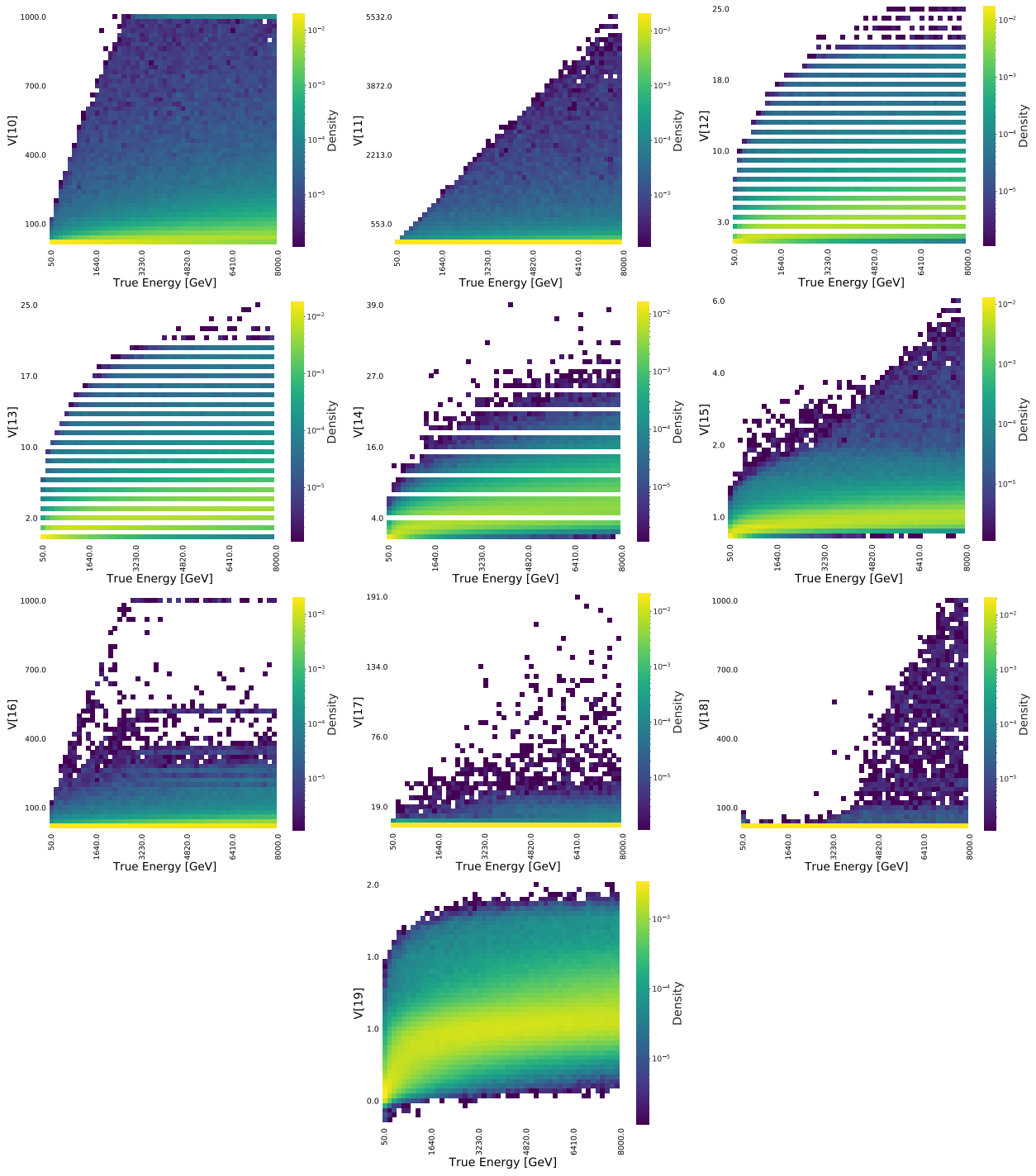


Figure 19: 2D histograms showing the dependence of features V[10] to V[19] (on the y axes) on true muon energy (on the x axes). Features are defined in Section A.3.

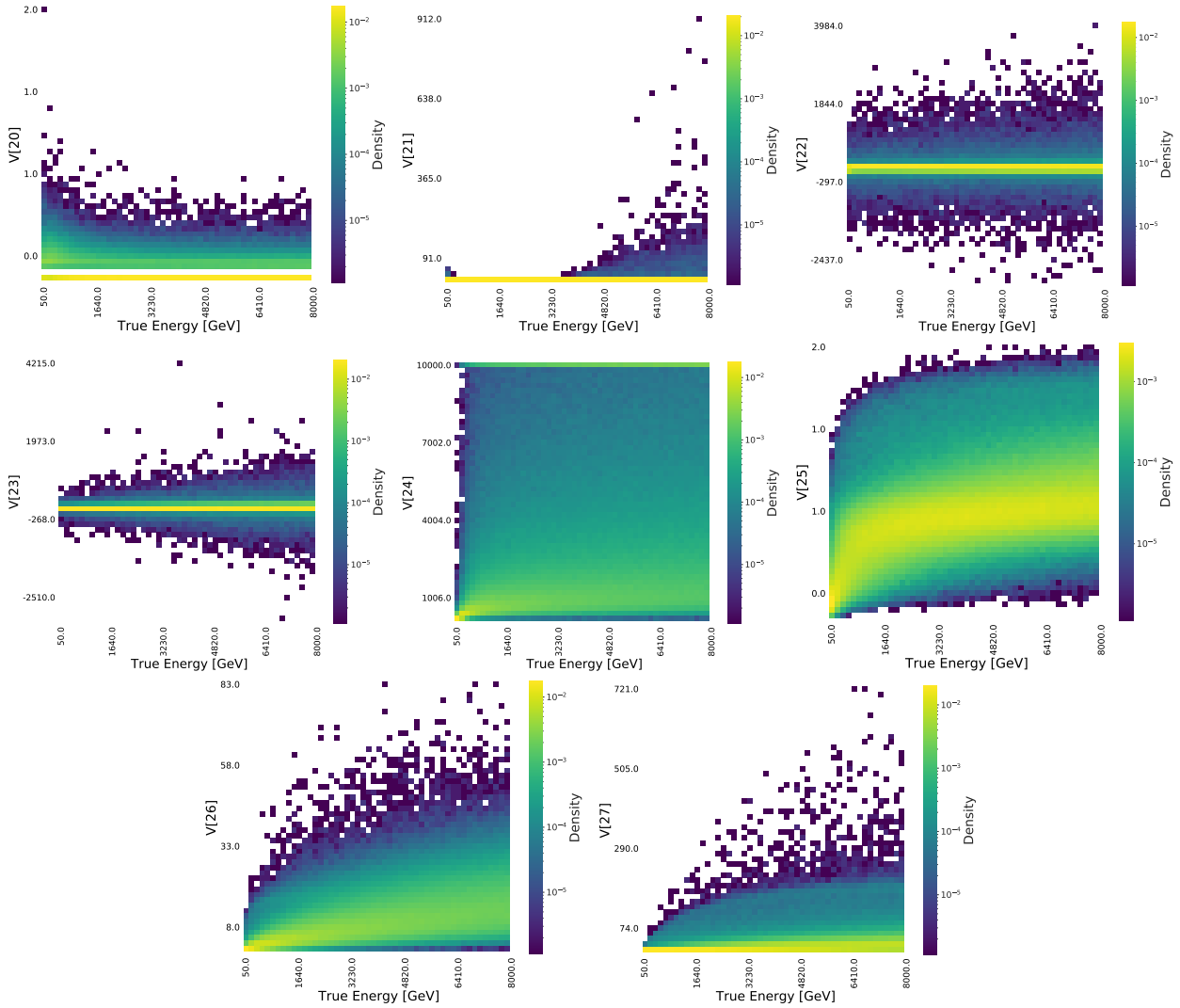


Figure 20: 2D histograms showing the dependence of features V[20] to V[27] (on the y axes) on true muon energy (on the x axes). Features are defined in Section A.3.

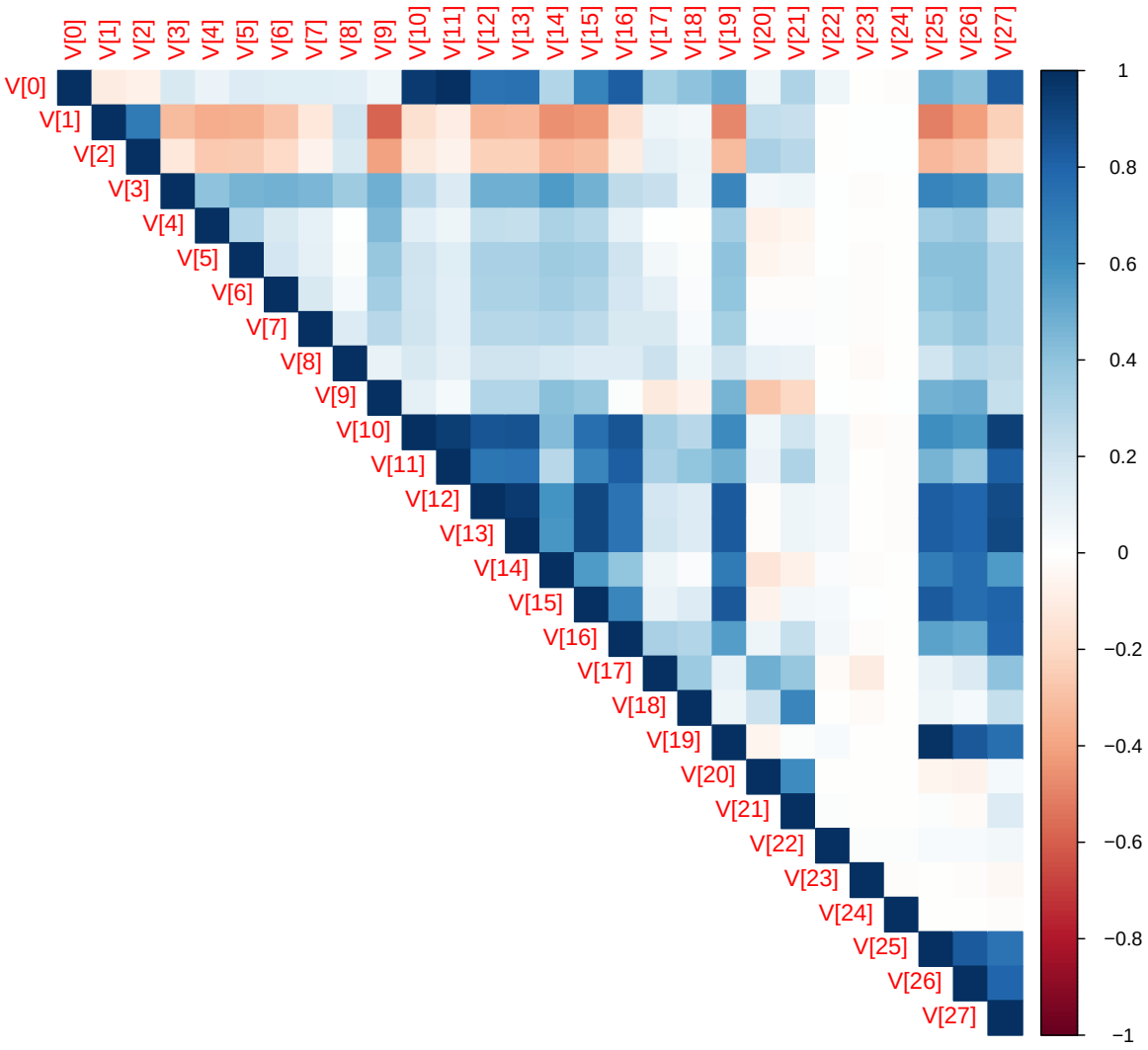


Figure 21: Correlation matrix between the high-level features. See the text for details.

B Ablation study

The architecture of the network described in Sec. 3.1, and training methodology detailed in Sec. 3.2, are both reasonably complex, leveraging a range of recently published, or otherwise unusual, techniques, as well as other aspects that are specific to the task at hand. Similar to the study of the input features in Sec. 4.2, it is worth quantifying the actual benefits of each of these items in the hope of simplifying the model, or to help inform future studies in similar task regimes.

This ablation study takes the strategy of inspecting separately: the loss, the ensembling, the training, the architecture, and the bias correction. In each inspection, particular aspects of the model or method will be removed, or replaced with more standard approaches, individually (*i.e.* only one aspect is ever different) to quantify the benefit of each particular aspect in the presence of the others. Finally, the whole section of the model or method will be replaced with a standard approach, to quantify the overall benefit of that part of the model or method. Unless otherwise stated, the results shown are computed on the monitoring-validation dataset and averaged over the five models trained per test via unique-fold training (see Sec. 3.2.4).

B.1 Loss function

As a reminder, the regressor is trained to minimise a Huberised version of a domain-informed loss function, which tracks running averages of its thresholds in bins of true energy, and element-wise losses are weighted according to a function of the true energy. We can simplify this loss function by: using just a single bin of true energy; using a fixed threshold per bin computed solely on the first batch of data; using thresholds per bin computed entirely on the current batch of data without averaging; using unweighted loss elements; or using the mean fraction squared error without Huberisation (*i.e.* as in Eq. 1). Finally we can replace the domain-inspired function with a more standard mean squared-error loss, however we will still include the down-weighting of the data. The results of these studies are summarised in Tab. 2.

Ablation	MI	Change in MI [%]
Default	19.42 ± 0.08	N/A
Single bin	19.14 ± 0.08	-1.5 ± 0.6
Batchwise thresholds	19.25 ± 0.04	-0.9 ± 0.5
Non-Huberised loss	19.36 ± 0.06	-0.4 ± 0.5
Fixed thresholds	19.39 ± 0.05	-0.2 ± 0.5
MSE loss	18.43 ± 0.06	-5.1 ± 0.5
No down-weighting	16.5 ± 0.2	-15.12 ± 1.03

Table 2: Ablation study of the loss. The change in MI is computed as the fractional difference with respect to the default model. The "MSE" test uses the down-weighting of the data, and the "No down-weighting" test uses the full adaptive Huberised MFSE loss.

From these results we can confirm that the domain-inspired loss we have adopted is beneficial to the training, and that the down-weighting is also very important. The Huberisation of the loss is potentially useful, however when it is used one should compute separate thresholds in bins of true energy, and either fix these for the whole training, or track their running average.

B.2 Ensembling and dataset size

Whilst in this study the focus is mainly on improving performance, in application one may also be concerned by retraining time, inference time, and dataset-size requirements. Due to these potential concerns it is worth checking the benefits of ensembling and training on larger datasets. Table 3 summarises two ensemble trainings, one via full-fold, and the other via unique-fold. Each training is then interpreted in two ways: one assumes that all five models were trained and applied as an ensemble (full/unique ensemble); the other assumes that only one model was trained and computes the average MI across the five models that were actually trained (full/unique singles).

Ablation	Dataset size	Times		MI
		Training [h]	Inference [second per batch]	
Full ensemble	862 085	113.4	0.47	20.72
Full singles	862 085	22.7	0.091	20.29 ± 0.04
Unique ensemble	862 085	23.3	0.47	19.83
Unique singles	197 048	4.7	0.0.091	19.37 ± 0.08

Table 3: Ablation study of the ensembling and dataset size and usage. Inference time is per batch of 256 muons and excludes the disk-to-RAM time. MI is computed on the holdout-validation data. "Full" indicates the model was trained on 34/36 folds of data and monitored on one fold of data. "Unique" indicates the model was trained on a unique set of seven folds of data, and monitored on one fold of data. In the case of the single models, the MI is averaged across five individual models.

From the above results we can see that both ensembling and using a larger dataset provide performance improvements. We also see that if training with a larger dataset, then it is better to train a single model on the whole of it, than an ensemble on unique subsamples of it, both in terms of inference time and MI. Since the training time of the full models, the disk-space-size per data point, and the data generation times are all relatively low compared to many other algorithms used in HEP and trainings can be used for an entire data-taking run, our recommendation would be to use as much training data as possible. The choice between single model or ensembling depends mostly on the time-budget available during application (since other reconstruction algorithms will be being run during processing), and whether the regression is performed online during data-taking for triggering, or during offline reconstruction.

B.3 Training

The nominal training scheme involves changing the learning rate and the momentum of the optimiser during training: first via a 1cycle schedule, to quickly train the model; and second via a step decay of the LR. To check the advantage of this, we can retrain keeping the LR and momentum constant. The LR is set to 1×10^{-4}, slightly lower than the maximum LR used for nominal training, to account for the fact that it has no possibility to decrease, other than through ADAM's scaling parameters, and that the momentum will not be able to stabilise the higher LR. The number of epochs and early-stopping criteria are kept the same. Such a training results in a $(5.2 \pm 0.6)\,\%$ decrease in MI, and an increase in the required training time due to the nominal scheme triggering the early-stopping criterion earlier.

B.4 CNN architecture

The CNN, although inspired by established architectures, is by no means standard, and includes a task-specific component in the form of the energy-pass-through connections. The studies performed are: removal of the squeeze-excitation blocks; removal of the max-average pooling layer, instead flattening the hidden state and feeding all inputs to the fully connected layers; replacing the running batchnorm layers with standard BN layers; removal of BN entirely; removal of the identity paths, *i.e.* the paths through the trainable convolutional layers are no longer residual (in this case the positions of the BN and activation layers are changed to always be convolution into activation into BN); and removal of the energy-pass-through connections (in this case the number of channels added at each downsampling stage is increased to maintain a similar number of trainable parameters). Finally, we can remove the CNN head entirely and flatten all 51 200 cell values into a vector to be fed directly to the full connected layers. Table 4 details the results of these studies.

Ablation	MI	Change in MI [%]
Default	19.42 ± 0.08	N/A
No BN	18.5 ± 0.3	-5 ± 1
No identity path	18.72 ± 0.08	-3.6 ± 0.6
Nominal BN	19.2 ± 0.2	-1.1 ± 0.9
No E-pass	19.30 ± 0.05	-0.6 ± 0.5
No SE	19.33 ± 0.09	-0.5 ± 0.6
No pooling	19.4 ± 0.1	-0.4 ± 0.7
No CNN	17.45 ± 0.09	-10.2 ± 0.6

Table 4: Ablation study of the architecture. The change in MI is computed as the fractional difference with respect to the default model.

As expected, the CNN head is essential to avoid over-parameterising the model. Additionally, the use of running batchnorm is necessary to avoid instabilities in the validation performance of the network (running without any BN at all also produces instabilities in the training loss). The identity paths also provide a large improvement to the model. It is interesting to note that the energy pass-through connections provide an improvement, since the model should be able to learn this itself, however similar to DenseNet, the fact that we explicitly retain a part of the previous representation of the data throughout the model, allows a slightly more direct flow of gradient update to the trainable layers. Additionally we are implicitly suggesting that the trainable layers act as small corrections to the recorded energy, rather than allowing the model to learn this approach.

B.5 Bias correction

Whilst not strictly part of the architecture, we can also check whether the minor correction to the predictions that we apply post-training is useful in improving the resolution. Without the bias correction, the change in MI is $-2.1\,\%$ on the holdout-validation dataset when using the nominal model and all-fold training (see Sec. 3.2.4), so the correction is worth applying.

C Resource requirements

C.1 Regressor

The models used for this study were trained on Nvidia V100S GPUs. Training the nominal architecture at a batch size of 256 requires 5 GB of VRAM, 23 GB RAM, and 100 % of both a single (virtual) CPU core (Intel Xeon Gold 6248 CPU @ 2.5 GHz in our case) and the GPU. The training time per model is about 23 h, and about five days for the full ensemble when trained serially; however, with sufficient resources, ensemble training would be trivially parallelisable.

Application of the ensemble takes 61 s for a dataset of 24 631 muons computed in batches of 256, of which 15 s are spent loading the dataset into RAM. Excluding the disk-to-RAM time, inference is about 0.5 s for a batch of 256 muons (including RAM-to-VRAM time) for the ensemble (0.1 s per batch per model).

Although some steps are taken to reduce data-loading times (LZF-compression and sparse hit-representation), disk-to-RAM loading time is still significant and training/inference time depends highly on the disk read speed and access latency; whilst production and development was mainly performed in the cloud on powerful and expensive GPUs, local runs on a much cheaper Nvidia 1080 Ti GPU with a solid-state hard-drive were actually just as quick.

Whilst the loading time from RAM to GPU is minor compared to the load-time from disk to RAM, further improvements would be to retain the sparse representation of the data, however sparse tensors in PyTorch are still experimental, and sparse CNN implementations are limited in functionality, let alone implemented for 3D convolutions.

C.2 Datasets and preparation

The time to generation the data via Geant 4 heavily depends on the muon energy, however by running the generation as 7000 simultaneous jobs on a batch system, the dataset was processed in about one day. The raw ROOT files require 183 GB of storage space.

Computation of the high-level features is performed in C++ and is also run as 7000 jobs on a batch system, taking a few hours to complete. The resulting uncompressed CSV files require 246 MB of space.

Processing of the raw hits from ROOT into the HDF5 files required by Lumin, and combination with the high-level features is a three-step process:

1. Each ROOT file is processed into an LZF-compressed HDF5 file containing only the raw hits and the muon energy (the ROOT files also contain additional information which is no longer required). This takes about six hours and requires about a further 44 GB of space.

2. Meta data required to pre-process the HL-features is computed via a loop over the CSV files, which takes a few seconds.

3. The individual HDF5 files are combined with the CSV files into two LZF-compressed HDF5 files with the training and validation data being split into 36 folds, and the testing data split into 18 folds. At this point the HL-features are pre-processed based on the meta data computed beforehand, and the raw energy deposits are transformed into a sparse format (which reduces loading time). This requires several hours; the final training file has a size of 32 GB and the testing file of 12 GB.

D Software

The investigation performed in this project depended on many open-source software packages. These are summarised in Tab. 5.

Software	Version	References	Use/Notes
Lumin	0.8	[37]	Wrapping PyTorch to implement networks
PyTorch	1.8	[36]	Implementing neural networks
Seaborn	0.9	[52]	Plot production
Matplotlib	3.2	[53]	Plot production
Pandas	1.2	[54]	Data analysis and computation
NumPy	1.21	[55]	Data analysis and computation
Scikit-Learn	0.22.0	[56]	Data shuffling & splitting
Geant	4	[34, 35]	Detector simulation
Root	6	[57]	Processing of data
Uproot	3.11	[58]	Processing of data

Table 5: Software used for the investigation

References

[1] C. D. Anderson and S. H. Neddermeyer, *Cloud Chamber Observations of Cosmic Rays at 4300 Meters Elevation and Near Sea-Level*, Phys. Rev. **50** (Aug, 1936)263–271. https://link.aps.org/doi/10.1103/PhysRev.50.263.

[2] S. H. Neddermeyer and C. D. Anderson, *Note on the Nature of Cosmic-Ray Particles*, Phys. Rev. **51** (May, 1937)884–886. https://link.aps.org/doi/10.1103/PhysRev.51.884.

[3] J. E. Augustin *et al.*, *Discovery of a Narrow Resonance in e^+e^- Annihilation*, Phys. Rev. Lett. **33** (Dec, 1974)1406–1408. https://link.aps.org/doi/10.1103/PhysRevLett.33.1406.

[4] S. W. Herb *et al.*, *Observation of a Dimuon Resonance at 9.5 GeV in 400-GeV Proton-Nucleus Collisions*, Phys. Rev. Lett. **39** (Aug, 1977)252–255. https://link.aps.org/doi/10.1103/PhysRevLett.39.252.

[5] **D0** Collaboration, S. Abachi *et al.*, *Observation of the Top Quark*, Phys. Rev. Lett. **74** (Apr, 1995)2632–2637. https://link.aps.org/doi/10.1103/PhysRevLett.74.2632.

[6] G. Arnison *et al.*, *Experimental observation of isolated large transverse energy electrons with associated missing energy at s=540 GeV*, Physics Letters B **122** no. 1, (1983)103–116. https://www.sciencedirect.com/science/article/pii/0370269383911772.

[7] G. Aad *et al.*, *Observation of a new particle in the search for the Standard Model Higgs boson with the ATLAS detector at the LHC*, Physics Letters B **716** no. 1, (2012)1–29. https://www.sciencedirect.com/science/article/pii/S037026931200857X.

[8] S. Chatrchyan *et al.*, *Observation of a new boson at a mass of 125 GeV with the CMS experiment at the LHC*, Physics Letters B **716** no. 1, (2012)30–61. https://www.sciencedirect.com/science/article/pii/S0370269312008581.

[9] **CMS** Collaboration, *Measurement of Higgs boson decay to a pair of muons in proton-proton collisions at $\sqrt{s} = 13\,\text{TeV}$*, tech. rep., CERN, Geneva, 2020. https://cds.cern.ch/record/2725423.

[10] P. Fayet, *Extra U(1)'s and new forces*, Nuclear Physics B **347** no. 3, (1990)743–768. https://www.sciencedirect.com/science/article/pii/055032139090381M.

[11] P. Langacker, *The physics of heavy Z' gauge bosons*, Rev. Mod. Phys. **81** (Aug, 2009)1199–1228. https://link.aps.org/doi/10.1103/RevModPhys.81.1199.

[12] **Particle Data Group** Collaboration, M. Tanabashi *et al.*, *Review of Particle Physics*, Phys. Rev. D **98** (Aug, 2018) 030001. https://link.aps.org/doi/10.1103/PhysRevD.98.030001.

[13] **ATLAS** Collaboration, G. Aad *et al.*, *Muon reconstruction performance of the ATLAS detector in proton–proton collision data at $\sqrt{s}$ =13 TeV*, Eur. Phys. J. C **76** no. 5, (2016) 292, arXiv:1603.05598 [hep-ex].

[14] **CMS** Collaboration, A. M. Sirunyan *et al.*, *Performance of the reconstruction and identification of high-momentum muons in proton-proton collisions at $\sqrt{s} = 13$ TeV*, JINST **15** no. 02, (2020) P02027, arXiv:1912.03516 [physics.ins-det].

[15] **ATLAS** Collaboration, *Electron and photon performance measurements with the ATLAS detector using the 2015–2017 LHC proton-proton collision data*, Journal of Instrumentation **14** no. 12, (Dec, 2019) P12006–P12006. http://dx.doi.org/10.1088/1748-0221/14/12/P12006.

[16] **CMS** Collaboration, *Search for resonant and nonresonant new phenomena in high-mass dilepton final states at $\sqrt{s} = 13$ TeV*, 2021.

[17] **CMS** Collaboration, *The CMS experiment at the CERN LHC*, JINST **3** (2008) S08004.

[18] R. Abbasi *et al.*, *An improved method for measuring muon energy using the truncated mean of dE/dx*, Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment **703** (Mar, 2013) 190–198. http://dx.doi.org/10.1016/j.nima.2012.11.081.

[19] M. G. Aartsen *et al.*, *Energy reconstruction methods in the IceCube neutrino telescope*, Journal of Instrumentation **9** no. 03, (Mar, 2014) P03009–P03009. http://dx.doi.org/10.1088/1748-0221/9/03/P03009.

[20] **CMS** Collaboration, *Electron and photon reconstruction and identification with the CMS experiment at the CERN LHC*, Journal of Instrumentation **16** no. 05, (May, 2021) P05014. http://dx.doi.org/10.1088/1748-0221/16/05/P05014.

[21] D. Belayneh *et al.*, *Calorimetry with deep learning: particle simulation and reconstruction for collider physics*, The European Physical Journal C **80** no. 7, (Jul, 2020). http://dx.doi.org/10.1140/epjc/s10052-020-8251-9.

[22] **CALICE** Collaboration, *Hadronic energy resolution of a highly granular scintillator-steel hadron calorimeter using software compensation techniques*, JINST **7** (2012) P09017, arXiv:1207.4210 [physics.ins-det].

[23] **CALICE** Collaboration, *Shower development of particles with momenta from 15 GeV to 150 GeV in the CALICE scintillator-tungsten hadronic calorimeter*, JINST **10** (2015) P12006, arXiv:1509.00617 [physics.ins-det].

[24] **CALICE** Collaboration, Y. Israeli, *Energy Reconstruction of hadrons in highly granular combined ECAL and HCAL systems*, JINST **13** no. 05, (2018) C05002, arXiv:1803.05232 [physics.ins-det].

[25] **CMS** Collaboration, *The Phase-2 Upgrade of the CMS Endcap Calorimeter*, Tech. Rep. CERN-LHCC-2017-023. CMS-TDR-019, 2017. https://cds.cern.ch/record/2293646.

[26] C. Neubüser *et al.*, *Calorimeters for the FCC-hh*, FCC Document CERN-FCC-PHYS-2019-0003, CERN, 2019. arXiv:1912.09962 [physics.ins-det]. https://cds.cern.ch/record/2705432.

[27] C. Neubüser, J. Kieseler, and P. Lujan, *Optimising longitudinal and lateral calorimeter granularity for software compensation in hadronic showers using deep neural networks*, 2021.

[28] S. R. Qasim, J. Kieseler, Y. Iiyama, and M. Pierini, *Learning representations of irregular particle-detector geometry with distance-weighted graph networks*, The European Physical Journal C **79** no. 7, (Jul, 2019). http://dx.doi.org/10.1140/epjc/s10052-019-7113-9.

[29] X. Ju *et al.*, *Graph Neural Networks for Particle Reconstruction in High Energy Physics detectors*, 2020.

[30] S. R. Qasim, K. Long, J. Kieseler, M. Pierini, and R. Nawaz, *Multi-particle reconstruction in the High Granularity Calorimeter using object condensation and graph neural networks*, 2021.

[31] J. Alimena, Y. Iiyama, and J. Kieseler, *Fast convolutional neural networks for identifying long-lived particles in a high-granularity calorimeter*, Journal of Instrumentation **15** no. 12, (Dec, 2020) P12006–P12006. http://dx.doi.org/10.1088/1748-0221/15/12/P12006.

[32] Y. Iiyama *et al.*, *Distance-Weighted Graph Neural Networks on FPGAs for Real-Time Particle Reconstruction in High Energy Physics*, Frontiers in Big Data **3** (Jan, 2021). http://dx.doi.org/10.3389/fdata.2020.598927.

[33] A. Abada *et al.*, *FCC-ee: The Lepton Collider*, Eur. Phys. J. Spec. Top. **228** (2019) 261–623. https://doi.org/10.1140/epjst/e2019-900045-4.

[34] **GEANT4** Collaboration, S. Agostinelli *et al.*, *GEANT4–a simulation toolkit*, Nucl. Instrum. Meth. A **506** (2003)250–303.

[35] J. Allison *et al.*, *Geant4 developments and applications*, IEEE Transactions on Nuclear Science **53** no. 1, (Feb, 2006)270–278.

[36] A. Paszke *et al.*, *PyTorch: An Imperative Style, High-Performance Deep Learning Library*, arXiv:1912.01703 [cs.LG].

[37] G. C. Strong, *LUMIN*, Mar., 2019. https://doi.org/10.5281/zenodo.2601857. Please check https://github.com/GilesStrong/lumin/graphs/contributors for the full list of contributors.

[38] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, *Densely Connected Convolutional Networks*, in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2261–2269. 2017. arXiv:1608.06993.

[39] K. He, X. Zhang, S. Ren, and J. Sun, *Deep Residual Learning for Image Recognition*, in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778. 2016. arXiv:1512.03385. http://arxiv.org/abs/1512.03385.

[40] J. Hu, L. Shen, and G. Sun, *Squeeze-and-Excitation Networks*, in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7132–7141. 2018.

[41] K. He, X. Zhang, S. Ren, and J. Sun, *Identity Mappings in Deep Residual Networks*, CoRR **abs/1603.05027** (2016), arXiv:1603.05027. http://arxiv.org/abs/1603.05027.

[42] P. Ramachandran, B. Zoph, and Q. V. Le, *Searching for Activation Functions*, CoRR **abs/1710.05941** (2017), arXiv:1710.05941. http://arxiv.org/abs/1710.05941.

[43] K. He, X. Zhang, S. Ren, and J. Sun, *Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification*, in *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*, ICCV '15, p. 1026–1034. IEEE Computer Society, USA, 2015. arXiv:1502.01852 [cs.CV]. https://doi.org/10.1109/ICCV.2015.123.

[44] X. Glorot and Y. Bengio, *Understanding the difficulty of training deep feedforward neural networks*, JMLR Workshop and Conference Proceedings **9** (2010). http://jmlr.org/proceedings/papers/v9/glorot10a/glorot10a.pdf.

[45] fast.ai, *fastai course V3, lesson 10*, 2019. https://course19.fast.ai/videos/?lesson=10. Accessed 2021/05/25.

[46] P. J. Huber, *Robust Estimation of a Location Parameter*, The Annals of Mathematical Statistics **35** no. 1, (1964)73 – 101. https://doi.org/10.1214/aoms/1177703732.

[47] D. P. Kingma and J. Ba, *Adam: A Method for Stochastic Optimization*, arXiv:1412.6980 [cs.LG]. arXiv:1412.6980.

[48] L. N. Smith and N. Topin, *Super-convergence: very fast training of neural networks using large learning rates*, in *Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications*, T. Pham, ed., vol. 11006, pp. 369 – 386, International Society for Optics and Photonics. SPIE, 2019. arXiv:1708.07120. https://doi.org/10.1117/12.2520589.

[49] L. N. Smith, *A disciplined approach to neural network hyper-parameters: Part 1 - learning rate, batch size, momentum, and weight decay*, CoRR **abs/1803.09820** (2018), arXiv:1803.09820. http://arxiv.org/abs/1803.09820.

[50] fast.ai, *fastai library documentation*, 2019. https://docs.fast.ai. Accessed 2019/05/13.

[51] *2020 Update of the European Strategy for Particle Physics (Brochure)*, tech. rep., Geneva, 2020. https://cds.cern.ch/record/2721370.

[52] M. L. Waskom, *seaborn: statistical data visualization*, Journal of Open Source Software **6** no. 60, (2021) 3021. https://doi.org/10.21105/joss.03021.

[53] J. D. Hunter, *Matplotlib: A 2D graphics environment*, Computing In Science & Engineering **9** no. 3, (2007)90–95.

[54] Wes McKinney, *Data Structures for Statistical Computing in Python*, in *Proceedings of the 9th Python in Science Conference*, Stéfan van der Walt and Jarrod Millman, eds., pp. 56 – 61. 2010.

[55] S. van der Walt, S. C. Colbert, and G. Varoquaux, *The NumPy Array: A Structure for Efficient Numerical Computation*, Computing in Science Engineering **13** no. 2, (March, 2011)22–30.

[56] F. Pedregosa *et al.*, *Scikit-learn: Machine Learning in Python*, JMLR **12** no. 85, (2011)2825–2830. http://jmlr.org/papers/v12/pedregosa11a.html.

[57] R. Brun and F. Rademakers, *ROOT: An object oriented data analysis framework*, Nucl. Instrum. Meth. **A389** (1997)81–86.

[58] J. Pivarski *et al.*, *scikit-hep/uproot: 3.11.7*, June, 2020. https://doi.org/10.5281/zenodo.3877289.