IMPROVING VARIATIONAL AUTOENCODERS FOR NEW PHYSICS DETECTION AT THE LHC WITH NORMALIZING FLOWS

Pratik Jawahar, Thea Aarrestad, Maurizio Pierini
European Center for Nuclear Research (CERN)
CH 1211, Geneva 23, Switzerland

Kinga A. Wozniak
European Center for Nuclear Research (CERN)
CH 1211, Geneva 23, Switzerland
University of Vienna
1010 Wien, Austria

Jennifer Ngadiuba
California Institute of Technology
Pasadena, CA 91125, USA
Fermi National Accelerator Laboratory (FNAL)
Batavia, IL 60510, USA

Javier Duarte, Steven Tsan
University of California San Diego
La Jolla, CA 92093, USA

October 19, 2021

ABSTRACT

We investigate how to improve new physics detection strategies exploiting variational autoencoders and normalizing flows for anomaly detection at the Large Hadron Collider. As a working example, we consider the DarkMachines challenge dataset. We show how different design choices (e.g., event representations, anomaly score definitions, network architectures) affect the result on specific benchmark new physics models. Once a baseline is established, we discuss how to improve the anomaly detection accuracy by exploiting normalizing flow layers in the latent space of the variational autoencoder.

1 Introduction

Most searches for new physics at the CERN Large Hadron Collider (LHC) target specific experimental signatures. The underlying assumption of a specific new physics model could enter at various stages in the search design, e.g., when reducing the data rate from 40 M to 1000 collision events per second in real time [1–3], when designing the event selection, or when running the final hypothesis testing. When searching for preestablished and theoretically well-motivated particles (e.g., the Higgs boson), this strategy is extremely successful because the underlying assumption can be exploited to maximize the search sensitivity. On the other hand, the lack of a predefined target might turn this strength into a limitation.

To compensate for this potential problem, *model independent* searches are also carried out [4–8] at hadron colliders. These searches consist in an extensive set of comparisons between the data distribution and the expectation derived from Monte Carlo simulation. Many comparisons are carried out in parallel for multiple physics-motivated features while applying different event selections. However, when searching for new physics among many channels, the "global" significance of observing a particular discrepancy must take into account the probability of observing such a discrepancy anywhere. This so called look-elsewhere effect can be quantified in terms of a trial factor [9]. While the large trial factor typically reduces the statistical power of this strategy in terms of significance, model independent searches are valuable tools to identify possible regions of interest and provide data-driven motivations for traditional, more targeted searches to be performed on future data.

Recently, the use of machine learning (ML) techniques has been advocated as a means to reduce the model dependence [10–34]. In this context, the high energy physics (HEP) community engaged in two data challenges: the LHC

Olympics 2020 [35] and the DarkMachines challenge [36], where different approaches were explored to attempt to detect an unknown signal of new physics hidden in simulated data.

As part of our contribution to the DarkMachines challenge, we investigated the use of a particle-based variational autoencoder (VAE) and the possibility of enhancing its anomaly detection capability by using normalizing flows [37] in the latent space to improve the modeling of the decoding posterior. In this paper, we document those studies and expand that effort, investigating the impact of specific architecture choices (event representation, network architecture, usage of expert features, and definition of the anomaly score). This study is an update of Ref. [36], which benefits from the lessons learned by the DarkMachines challenge. Taking inspiration from solutions presented by other groups in the challenge (e.g., Refs. [38, 39]), we evaluate the impact of some of their findings on our specific setup. In some cases (but not always), these solutions translate in an improved performance, quantified using the same metrics presented in Ref. [36]. In this way, we establish an improved baseline model, on top of which we evaluate the impact of the normalizing flow layers in the latent space.

2 Data samples and event representation

This study is based on the datasets released on the Zenodo platform [40] in relation to the Dark Machines Anomaly Score Challenge [36]. It consists on a set of SM processes, mixed according to their production cross section at 13 TeV, and a set of benchmark signal samples. The datasets contains labels, identifying the process that generated each event. Labels are ignored during training and used to evaluate performance metrics.

For each sample, four datasets are provided, corresponding to four different event selections (called *channels* [36]):

- Channel 1: $H_{\mathrm{T}} \geq 600\,\mathrm{GeV}$, $p_{\mathrm{T}}^{\mathrm{miss}} \geq 200\,\mathrm{GeV}$, and $p_{\mathrm{T}}^{\mathrm{miss}}/H_{\mathrm{T}} \geq 0.2$.
- Channel 2a: $p_{\mathrm{T}}^{\mathrm{miss}} \geq 50\,\mathrm{GeV}$ and at least three light leptons (muons or electrons) with $p_{\mathrm{T}} > 15\,\mathrm{GeV}$.
- Channel 2b: $p_{\mathrm{T}}^{\mathrm{miss}} \geq 50\,\mathrm{GeV}$, $H_{\mathrm{T}} \geq 50\,\mathrm{GeV}$ and at least two light leptons (muons or electrons) with $p_{\mathrm{T}} > 15\,\mathrm{GeV}$.
- Channel 3: $H_{\mathrm{T}} \geq 600\,\mathrm{GeV}$, $p_{\mathrm{T}}^{\mathrm{miss}} \geq 100\,\mathrm{GeV}$.

where p_{T} is the magnitude of a particle's transverse momentum, H_{T} is the scalar sum of the jet p_{T} in the event, and $\vec{p}_{\mathrm{T}}^{\mathrm{miss}}$ is the vector equal and opposite to the vector sum of the transverse momenta of the reconstructed particles in the event, while $p_{\mathrm{T}}^{\mathrm{miss}}$ is its magnitude[1]. More details are provided in Ref. [36].

Table 1: Summary of the available dataset size.

Dataset	Channel 1	Channel 2a	Channel 2b	Channel 3
Training	193,800	13,425	238,450	7,100,934
Validation	10,200	707	12,550	373,733
Bkg. Test	10,000	5,868	89,000	1,025,333
Sig. Test	38,666	5,868	89,676	1,023,320

The input consists of the momenta of all the reconstructed physics objects in the event (jets, b jets, electrons e, muons μ, and photons), ordered by decreasing p_{T}. Each list of objects is zero-padded to force each event into a fixed-length matrix with the same order: up to 15 jets, and up to 4 each of b jets, $\mu^{\pm}$, $\mathrm{e}^{\pm}$, and photons. We preprocess the input by applying the `scikit-learn` standard scaling [41] and arranging the list of objects into a matrix of 39 particles times four momentum features $(E, p_{\mathrm{T}}, \eta, \phi)$, where E is the particle energy. This matrix is interpreted as an image or an unordered graph, depending on the underlying VAE architecture.

The training and validation dataset consists of background events from the SM mixture. The available dataset size is detailed in Table 1 for each of the channels. The background test samples are combined with the benchmark signal samples listed in Table 2 to form the labelled test dataset on which performance is evaluated.

[1] We use a Cartesian coordinate system with the z axis oriented along the beam axis, the x axis on the horizontal plane, and the y axis oriented upward. The x and y axes define the transverse plane, while the z axis identifies the longitudinal direction. The azimuth angle ϕ is computed with respect to the x axis. The polar angle θ is used to compute the pseudorapidity $\eta = -\log(\tan(\theta/2))$. The transverse momentum (p_{T}) is the projection of the particle momentum on the (x, y) plane. We fix units such that $c = \hbar = 1$.

Table 2: BSM processes contributing to the signal dataset in each channel. The process code, adopted in this study, is taken from Ref. [36].

BSM process	Code	Ch.1	Ch.2a	Ch.2b	Ch.3
Z' + jet	`monojet_Zp2000.0_DM_50.0`	×	×		×
Z' + W/Z	`monoV_Zp2000.0_DM_50.0`				×
Z' + t	`monotop_200_A`	×			×
Z' in LFV $U(1)_{L_\mu - L_\tau}$	`pp23mt_50`		×	×	
	`pp24mt_50`		×	×	
$\not{R}$-SUSY $\widetilde{t}\widetilde{t}$	`stlp_st1000`	×		×	×
$\not{R}$-SUSY $\widetilde{q}\widetilde{q}$	`sqsq1_sq1400_neut800`	×			×
SUSY $\widetilde{g}\widetilde{g}$	`glgl1400_neutralino1100`	×	×	×	×
	`glgl1600_neutralino800`	×	×	×	×
SUSY $\widetilde{t}\widetilde{t}$	`stop2b1000_neutralino300`	×			×
SUSY $\widetilde{q}\widetilde{q}$	`sqsq_sq1800_neut800`	×			×
SUSY $\widetilde{\chi}^{\pm}\widetilde{\chi}^{0}$	`chaneut_cha200_neut50`		×	×	
	`chaneut_cha250_neut150`		×	×	
SUSY $\widetilde{\chi}^{\pm}\widetilde{\chi}^{\pm}$	`chacha_cha300_neut140`			×	
	`chacha_cha400_neut60`			×	
	`chacha_cha600_neut200`			×	

3 Training setup and evaluation metrics

The VAE models are trained on the training and validation datasets, minimizing the loss function:

$$L_{\text{total}} = \beta D_{\text{KL}} + (1-\beta) L_{\text{C}} \,, \tag{1}$$

where L_{C} is an L_1-type permutation-invariant Chamfer [42] loss:

$$L_{\text{C}} = \sum_{\vec{x} \in S_{\text{input}}} \min_{\vec{y} \in S_{\text{output}}} \left|\vec{x} - \vec{y}\right| + \sum_{\vec{y} \in S_{\text{output}}} \min_{\vec{x} \in S_{\text{input}}} \left|\vec{x} - \vec{y}\right| \,, \tag{2}$$

similar to the L_2-type Chamfer distance used in Refs. [43, 44]. In eq. (2), D_{KL} is Kullback–Liebler (KL) divergence term usually employed to force the data distribution in the latent space to a multidimensional Gaussian with unitary covariance matrix [45], and β is a parameter that controls the relative importance of the two terms [46].

All of our models are optimized using the Adam minimizer [47]. A learning rate of 10^{-4} is applied along with a brute force early stopping strategy used on an ad-hoc basis. A batch size of 32 is chosen to train models. All models are implemented with the `PyTorch` [48] deep learning framework and are hosted on GitHub [49].

We train and test all our models on the WPI Turing Research Cluster[2]. We use 8 CPU nodes and 1 GPU node to train our models on the cluster. NVIDIA Tesla V100 and Tesla P100 GPUs were used for acceleration.

At inference time, L_{C} is used as an anomaly detection score, to quantify the distance between the input and the output. By applying a lower-bound threshold on L_{C}, we identify every event with an L_{C} value larger than the threshold as an anomaly. By comparing this prediction with the ground truth, we can assess the performance of the VAE on specific signal benchmark models.

To evaluate model performance we follow the same strategy and code used in Ref. [36] to enable comparison with other models tested on this dataset. As explained in Ref. [36], we extract four main performance parameters from the receiver operating characteristic (ROC) curves based on the chosen anomaly metric for each model, namely the area under the curve (AUC) and the signal efficiency (ϵ_{S}) or true positive rate at three different, fixed background efficiencies (ϵ_{B}) or false positive rates. We then combine these scores from all models on all available signal regions across all channels of the dataset to form box-and-whisker plots, using 6 different combination and comparison strategies namely, the highest mean score method, highest median score method, average rank method, top scorer method, top-5 scorer method, and highest minimum scorer method. A box is drawn spanning the inner half (50% quantile centered at the median) of the data as shown in Fig. 1. A line through the box marks the median. Whiskers extend from the box to either the maximum and minimum unless these are further away from the edge of the box than 1.5 box lengths. The outlier points are shown as circles.

[2] `https://arc.wpi.edu/computing/hpc-clusters/`

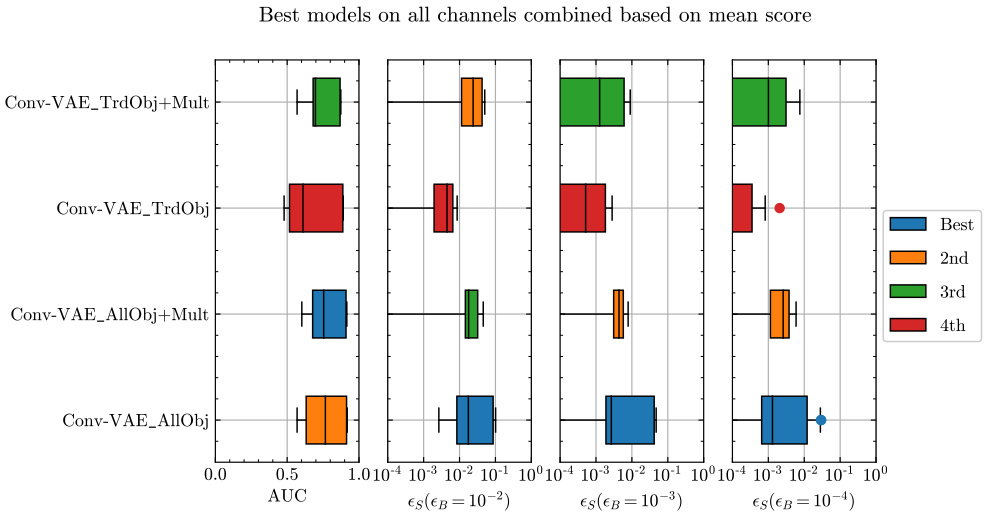

Figure 1: Anomaly detection performance for the Conv-VAE with different inputs given (see text for more details): all physics objects in the event (AllObj); truncated input object list (TrdObj); all objects and array of object multiplicity (AllObj+Mult); truncated input object list and array of object multiplicity (TrdObj+Mult).

For Fig. 1 and the other figures, the representative ranking as denoted by the legend corresponds to the performance based on the highest mean score method. However, to choose the best model for each experiment described in this paper, we consider all six comparison methods to arrive at a consensus. The code to perform these comparisons and to generate the corresponding plots is available in Ref. [36].

4 Baseline VAE model

The main goal of this study is to evaluate the impact of normalizing flow layers in the latent space on the anomaly detection capability of a reference VAE model. This and the following sections describe how this reference model is built, starting from the VAE based on convolutional layers (Conv-VAE) presented in Ref. [36] and modifying its architecture based on some of the lessons learned during the DarkMachine challenge.

The encoder of the initial Conv-VAE consists of three convolutional layers, with 32, 16, and 8 kernels of size $(3, 4)$, $(5, 1)$, and $(7, 1)$, respectively. For all layers, the stride is set to 1 and zero padding to "same". The output of the convolutional layers is flattened and passed to 2 fully-connected neural network (FCN) layers that output the mean and variance for the latent space. The cardinality of the latent space is fixed to 15. The decoder mirrors the encoder architecture, returning an output of the same size as the input.

In order to define the reference model, the architecture of the starting model is modified in different ways, each time evaluating the impact of a given choice on the test dataset. Several possibilities are considered: how to embed the event in the 2D array (see section 4.1); how to interpret the array, e.g., as an image or a graph (see section 4.2); whether to extend the event representation beyond the particle momenta, adding domain-specific high level features as an additional input (see section 4.3); and which anomaly score to use (see section 4.4). We study various options for each of these points, following this order. Doing so, we establish a candidate model, on which we evaluate the benefit of using normalizing flow layers in the latent space (see section 5) to improve the anomaly detection accuracy.

4.1 Data representation

By their nature, events consist of a variable number of objects. To some extent, this conflicts with most neural network architectures, which assume a fixed-size input. As a baseline, we adopt the simplest solution, i.e., to zero-pad all events to standardized event sizes for all available samples. To get a better idea of how padding affects results, we study performance across alternative input encodings. We consider two main types of encodings, listed as AllObj and TrdObj in Fig. 1. The former involves considering the entire event which implies allowing for a large enough padding such that every object per event is taken into consideration across the entire dataset. The latter involves cutting down the padding and the input sequence by considering only up to four leading jets and three objects each of the other types per event.

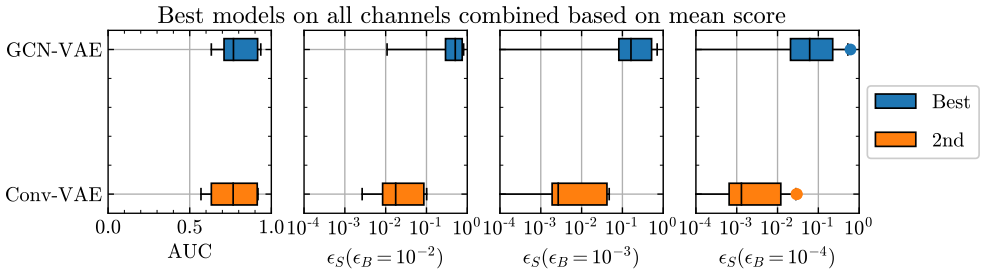

Figure 2: Comparison of the GCN-VAE and Conv-VAE performances, in terms of the benchmark figures of merit adopted in the paper.

When using the truncated sequence, the model loses information regarding the number of objects of each type per event, which is implicitly learned when the whole sequence is considered. To compensate for this loss, one can explicitly add this information passing a second input to the model, consisting of a vector containing the multiplicities of each object type. This input is concatenated to the flattened output received from the convolutional layers in the encoder before passing them to the fully connected layers. For the sake of comparison, we also do the same for the AllObj case (labelled as "+Mult" in Fig. 1).

The results in Fig. 1 show that the truncated sequence does worse than the full sequence. We also see little improvement in performance with the addition of multiplicity information per event in both the AUC as well as performance at lower background efficiencies. As a result, we fix the input encoding that considers the complete sequence per event as described in section 4.

4.2 VAE architecture

The convolutional architecture used for the baseline VAE is not the only option to handle the input considered in this study. The ensemble of reconstructed particles in an event can be represented as a point cloud and process it with a graph network. The main advantage of this choice stands with the permutation invariance of the graph processing, which pairs that of the loss in Eq. 2 and complies with the unordered nature of the input list of particles. Graph-based architectures have already been shown to perform better with sparse, non-Euclidean data representations in general [50, 51] and particle physics in particular [52, 53].

To this end, we consider a GCN-VAE model composed of multilayer graph convolutional network layers (GCNs) [54] and FCN layers in both the encoder and the decoder. As for the VAE, the input graphs are built from the input list described in section 2, each particle representing one vertex of the graph in the space identified by five particle features: $E, p_{\mathrm{T}}, \eta, \phi$, and object type. The object type is a label-encoded integer that signifies the object type. The input is structured as a fully connected, undirected graph which is passed to the GCN layers of the encoder, defined as [54]:

$$H_{(l+1)} = \sigma(\widetilde{D}^{-\frac{1}{2}}\widetilde{A}\widetilde{D}^{-\frac{1}{2}}H_{(l)}W_{(l)}) \,, \tag{3}$$

where $H_{(l)}$ is the input to the $(l+1)$th GCN layer with $H_{(0)} = X$ where X represents the node feature matrix. $H_{(l+1)}$ is the layer output, $\widetilde{A} = A + I$ is the adjacency of the graph A with I being the identity matrix which implies added self connections for each node. $\widetilde{D}_{ii} = \sum_j A_{ij}$ is defined for the normalized adjacency based message passing regime, $W_{(l)}$ is the layer weights matrix and $\sigma(\bullet)$ is a suitable nonlinear activation function. The output of the last GCN layer is flattened and passed to an FCN layer which populates the latent space. The encoder has 3 GCN layers that scale the 5 node features to 32, 16, and 2 respectively, followed by a single FCN layer which generates a 15-dimensional latent space. The decoder has a symmetrically inverted structure with the sampled point being upscaled through an FCN layer first and the resulting output is unflattened and passed to GCN layers that reconstruct back the node features.

Considering all comparison metrics along with the representative results shown in Fig. 2, we see a definitive improvement in performance compared to the Conv-VAE. The improvement is seen not only in the AUC metric, but more significantly in the signal efficiencies at lower background efficiencies. Because of this, the GCN-VAE is used as the reference architecture in the rest of this section and in section 5.

4.3 Physics-motivated high-level features

We also experiment with adding high-level features (HLFs) that are physics motivated, as explicit inputs to the model, similar to what was done with object multiplicities in section 4.1. Doing so, we intend to check if domain knowledge

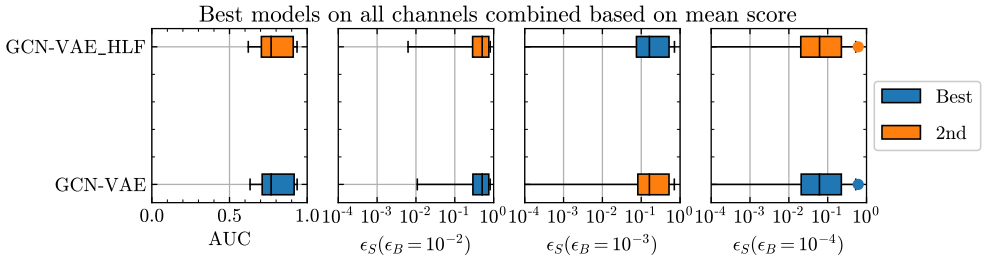

Figure 3: Comparison of the GCN-VAE performance with and without HLFs added as a separate input.

helps in improving anomaly detection capability. We pass event information such as the missing transverse momentum in the event ($p_{\mathrm{T}}^{\mathrm{miss}}$), the scalar sum of the jet p_{T} (H_{T}) and $m_{\mathrm{Eff}} = H_{\mathrm{T}} + p_{\mathrm{T}}^{\mathrm{miss}}$ to the model, by concatenating these with the output of the convolutional layers of the encoder. The concatenated output is then passed to the fully connected layers in the encoder to form the latent space. After the point sampled from the latent space passes through the fully connected layers of the decoder, the reconstructed $p_{\mathrm{T}}^{\mathrm{miss}}$, H_{T} and m_{Eff} are extracted and the rest of the layer output is re-shaped and further passed to the subsequent layers of the decoder.

To include the reconstruction of these features in the loss, we add to Eq. (1) a mean-squared error (MSE) term, computed from the reconstructed and input HLFs and weighted by a coefficient. This coefficient is treated as a hyperparameter that is scanned until the best performance is found.

Figure 3 shows that adding HLFs shows no definitive improvement in performance, thereby leading us to conclude that the baseline model with marginally lower number of trainable parameters is a good choice.

4.4 Anomaly scores

While so far the Chamfer loss has been used as the anomaly score, this is not the only possibility. We consider two alternative metrics: the D_{KL} term in Eq. (1) and [36]:

$$R_z = \sum_i \left(\frac{\mu_i}{\sigma_i}\right)^2 \tag{4}$$

where μ and σ are the mean and RMS returned by the encoder and the index i runs across the latent-space dimensions.

The use of different anomaly scores requires a tuning of the β hyperparameter. Since β determines the relative importance of the D_{KL} and Chamfer loss terms in the loss, the use of one or the other as anomaly score is certainly related to the choice of the optimal β value. Similarly, the use of R_z (i.e., anomaly detection in the latent space) might not be optimal when using a β value that was tuned to emphasize the reconstruction accuracy (i.e., the minimization of the Chamfer term in the loss). On the other hand, the study in Ref. [36] shows that an excessive tuning of the hyperparameters affects generalization of performance negatively beyond the available dataset.

In order to address this point, we compare three weights for the β term. In the first case represented as $\beta = 1$ in Fig. 4, the weight is chosen such that the contribution of the reconstruction loss is negligible to the total loss. The second case involves equal contribution of both terms to the loss represented as $\beta = 0.5$, and the final case corresponds to negligible contribution of the KL divergence term, represented as $\beta = 10^{-6}$.

Figure 4 shows that all three anomaly scores underperform in the $\beta = 10^{-6}$ case. The best performing models overall are the $\beta = 1$ and $\beta = 0.5$ cases. On comparing across the three different anomaly scores as well, we see that the $\beta = 1$ model that uses KL divergence and radius metrics, as well as the $\beta = 0.5$ model that uses the reconstruction metric perform the best. All three cases also show very similar performance across all comparison metrics as well as methods, implying that either model-anomaly score combination is equally suitable. We also find that the $\beta = 1$ KL score and the $\beta = 0.5$ reconstruction score are positively correlated. This implies that combining the two metrics would perform similarly as either metric used individually, thereby eliminating the need for a combination strategy as used in Ref. [38].

4.5 Baseline discrimination

As a result of the tests presented so far, the baseline VAE model is established as a GCN-VAE taking as input the whole set of reconstructed physics object but no domain-specific high level features. The Chamfer loss function is used as the anomaly score. The GCN-VAE is trained and tested only with data available within a given channel and

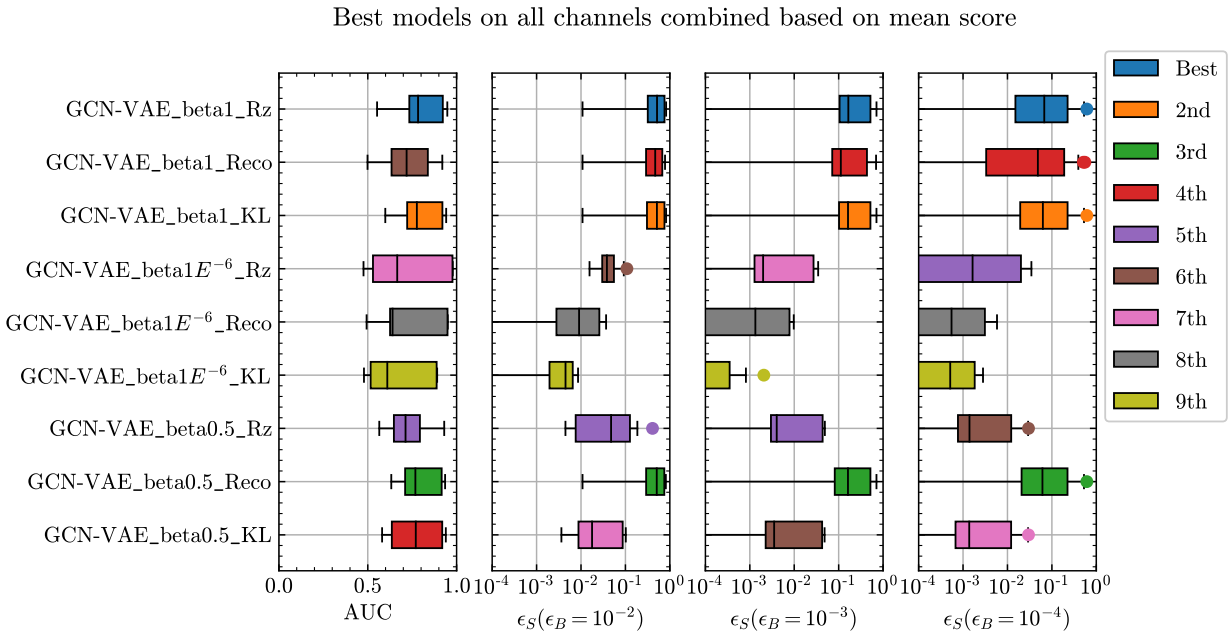

Figure 4: Comparison of anomaly detection performance from different anomaly score definitions, applied to the GCN-VAE.

the dataset sizes per channel are described in Table 1. Figure 5 shows the ROC curves for the baseline VAE model on benchmark signals in the four channels. It is evident that we suffer from a shortage of events for some signal models at very low FPRs (ϵ_{B}) specially in the smaller channels. We still depict ROC curves down to $\epsilon_{\mathrm{B}} = 10^{-4}$ because this was a convention chosen in Ref. [36] to make a fair comparison. Apart from a few signal models however, we see a definitive overall improvement in ϵ_{S} at very low ϵ_{B} for the GCN-VAE compared to our Conv-VAE submission in [36].

5 Normalizing flows

With the GCN-VAE serving as the baseline, we investigate how the use of normalizing flows impacts the anomaly-detection performance. Normalizing flow layers are inserted between the Gaussian sampling and the decoder. They provide additional complexity to learn better posterior distributions [45] by morphing the multivariate prior of the latent space to a more suitable, learned function. In other words, we use normalizing flows to go beyond the Gaussian prior approximation of our baseline VAE model.

A normalizing flow can be generalized as any invertible transformation that can be applied to a given distribution. In order to be compatible with variational inference, it is desirable for the transformations to have an efficient mechanism for computing the determinant of the Jacobian, while being invertible [45]. The normalizing flows are trained sequentially, together with the baseline VAE model.

We utilize four major families of flow models:

- **Planar flows** (PFs) are invertible transformations whose Jacobian determinant can be computed rather efficiently, making them suitable candidates for variational inference [45]. PF transformations are defined as:

$$\mathbf{z}' = \mathbf{z} + \mathrm{u}h(\mathrm{w}^T\mathbf{z} + b) \ , \tag{5}$$

  where $\mathrm{u}, \mathrm{w} \in \mathbb{R}^D$, $b \in \mathbb{R}$ and h is a suitable smooth activation function.

- **Sylvester normalizing flows** (SNFs) [55] build on the planar flow formulation and extend it to be analogous to a multilayer perceptron with one hidden layer of M units and a residual connection as:

$$\mathbf{z}' = \mathbf{z} + \mathbf{A}h(\mathbf{B}\mathbf{z} + b) \ , \tag{6}$$

  where $\mathrm{A} \in \mathbb{R}^{D\times M}, \mathrm{B} \in \mathbb{R}^{M\times D}$, $b \in \mathbb{R}^M$ and $M \leq D$. Computing the Jacobian determinant for such a formulation is made more efficient by utilizing the Sylvester determinant identity [55]. Depending on the

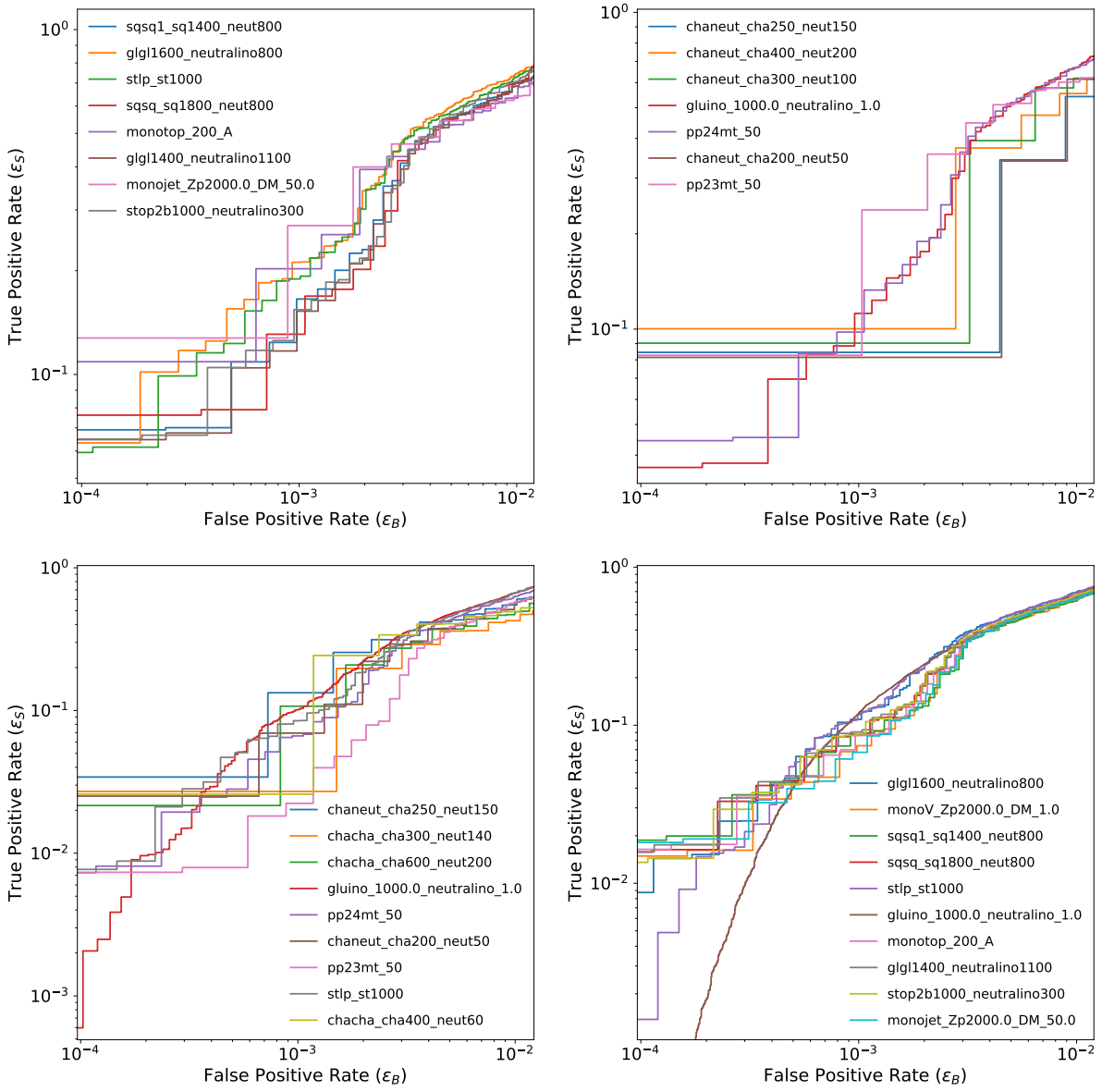

Figure 5: ROC curves for the baseline GCN-VAE model in channel 1 (top left), channel 2a (top right), channel 2b (bottom left), and channel 3 (bottom right), computed from the ϵ_{S} and ϵ_{B} values obtained on the background sample and the benchmark signal samples. Most of the ROC curves are not smooth, due to the small dataset size for some of the channels.

way A and B are parametrized, we get different types of Sylvester normalizing flows. In this paper we use orthogonal, Householder, and triangular SNFs, as described in Ref. [55].

- **Inverse autoregressive flows** (IAFs) [56] are computation-efficient normalizing flows based on autoregressive models. Autoregressive transformations are invertible, making them suitable candidates for our case. However, computing the transformation requires multiple sequential steps [55]. The inverse transformation however, leads to certain simplifications as described in Ref. [55], allowing more efficient parallel computing, thereby making it a more desirable transformation for our case. We use the IAFs formulated as:

$$z_i^t = \mu_i^t(z_{1:i-1}^{t-1}) + \sigma_i^t(z_{1:i-1}^{t-1}) \cdot z_i^{t-1} \quad , \quad i = 1, 2, ..., D \, . \tag{7}$$

Such a formulation allows one to stack multiple transformations to achieve more flexibility in producing target distributions.

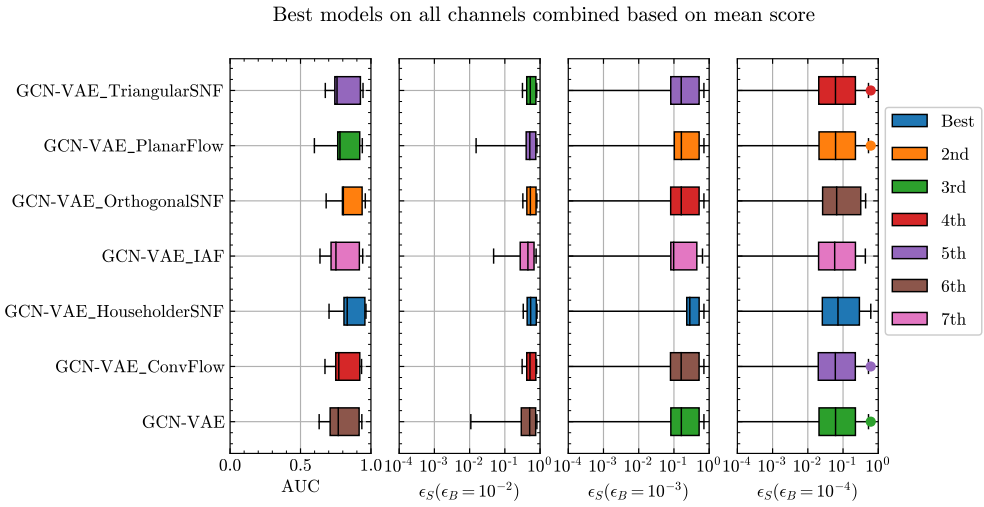

Figure 6: Comparison of anomaly detection performance for GCN-VAE models with different normalizing flow architectures in the latent space

- **Convolutional normalizing flows** (ConvFs) [57] are an extension of single-hidden-unit planar flows [56] to the case of multiple hidden units, further enhanced by replacing the fully connected network operation with a 1D convolution, to achieve bijectivity. They are defined by the following transformation:

$$\mathbf{z}' = \mathbf{z} + \mathbf{u} \odot h(\text{conv}(\mathbf{z}, \mathbf{w})) \ , \tag{8}$$

  where $w \in R^k$ is the parameter of the 1D convolution filter with k-sized kernel, h is a monotonic nonlinear activation function and $\odot$ denotes pointwise multiplication.

The hyperparameters for each normalizing flow architecture are chosen arbitrarily to avoid overtuning on the available dataset as learned from Ref. [36]. The PF model consists of a stack of six flows, each made of three dense layers with 90 neurons each. SNFs are defined by stacking six flows with 8 orthogonal, Householder and triangular transformations for each of the respective types of SNF. IAFs are constructed with four masked autoencoder for distribution estimation (MADE) [58] layers as described in [56], each containing 330 neurons. ConvFs include four flow layers with kernel size $k = 7$ and applying kernel dilation as described in [57].

Figure 6 shows the results of all GCN-VAE models combined with all the different types of flows as described in section 5. Based on results from all data channels combined, it is evident that using normalizing flows improves not only the AUC metric but also the signal efficiencies at low background efficiencies. We find that the Householder variant of SNFs produces the best improvement with respect to the baseline GCN-VAE model. The exercise was also repeated with a Conv-VAE model and similar trends were observed. There, the normalizing flows showed a larger improvement from the baseline Conv-VAE than for the GCN-VAE model but the overall results are less accurate than that of GCN-VAE with normalizing flows.

Figure 7 shows the ROC curves for the best presented model, GCN-VAE_HouseholderSNF across all available signal samples in all data channels. For some of the samples, the small dataset size translates in a discontinuous curve and larger uncertainties.

6 Conclusions

We constructed a graph-based anomaly detection model to identify new physics events in the DarkMachines challenge dataset, building its architecture by a set of tests aiming at optimizing performance with respect to specific design choices (data representation, use of physics-motivated high-level features, and anomaly score definition), inspired by the outcome of the DarkMachines challenge. As for many other deep learning applications to HEP data, the graph architecture better captures the point-cloud nature of HEP data, resulting in an enhanced performance.

On this baseline, we investigate the impact of using a stack of normalizing flows in the latent space of the variational autoencoder (VAE), between the Gaussian sampling and the decoding, in order to improve the accuracy of the posterior learning process by morphing the Gaussian prior to a more suitable prior, learned during training.

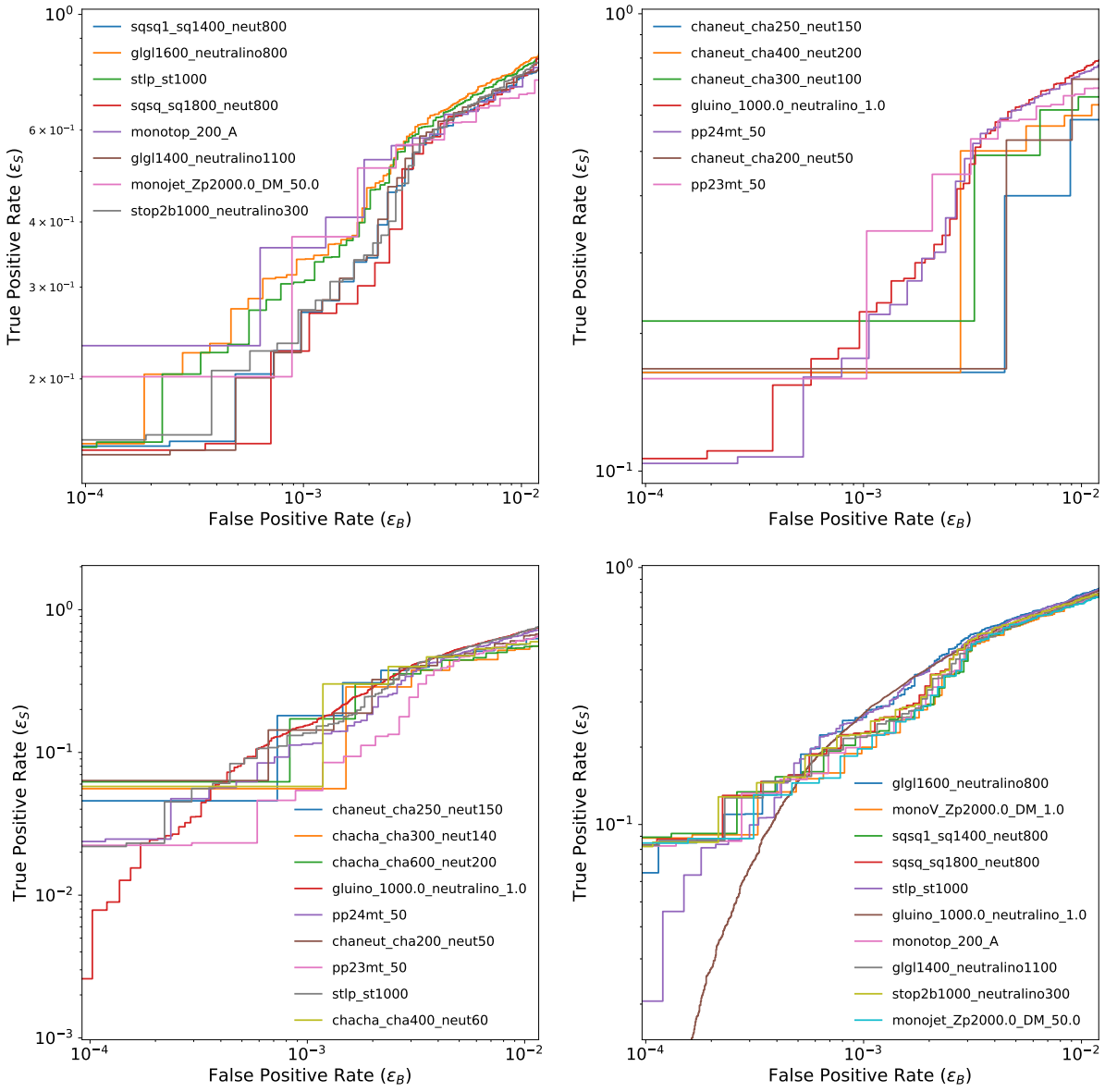

Figure 7: ROC curves of GCN-VAE_HouseholderSNF for all signals in each of channel 1 (top left), channel 2a (top right), channel 2b (bottom left), and channel 3 (bottom right).

Testing the trained model on a set of benchmark signal samples, we observe a general improvement when normalizing flows are used, with the Householder variant of the Sylvester normalizing flow model giving the best results. With that, we reach a median anomaly identification probability of 72% (34%) for an ϵ_{B} of 1% (0.1%) across all signal samples over all available channels. The median anomaly identification probability increases to 95% (96%) for an ϵ_{B} of 30% (60%).

This work represents an improvement over our convolutional neural network VAE model, submitted to the DarkMachines challenge [36].

7 Acknowledgement

P. J., T. A., M. P., and K. A. W. are supported by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation program (Grant Agreement No. 772369). J. D. is supported by the U.S.

Department of Energy (DOE), Office of Science, Office of High Energy Physics Early Career Research program under Award No. DE-SC0021187. S. T. was supported by the University of California San Diego Triton Research and Experiential Learning Scholars (TRELS) program. J. N. is supported by Fermi Research Alliance, LLC under Contract No. DE-AC02-07CH11359 with the U.S. Department of Energy, Office of Science, Office of High Energy Physics.

References

[1] CMS Collaboration, "Performance of the CMS Level-1 trigger in proton-proton collisions at $\sqrt{s} = 13$ TeV", *JINST* **15** (2020) P10017, doi:10.1088/1748-0221/15/10/P10017, arXiv:2006.10165.

[2] ATLAS Collaboration, "Operation of the ATLAS trigger system in Run 2", *JINST* **15** (2020) P10004, doi:10.1088/1748-0221/15/10/P10004, arXiv:2007.12539.

[3] D. Trocino, "The CMS High Level Trigger", *J. Phys. Conf. Ser.* **513** (2014) 012036, doi:10.1088/1742-6596/513/1/012036.

[4] H1 Collaboration, "A General Search for New Phenomena at HERA", *Phys. Lett. B* **674** (2009) 257–268, doi:10.1016/j.physletb.2009.03.034, arXiv:0901.0507.

[5] CDF Collaboration, "Global Search for New Physics with 2.0 fb^{-1} at CDF", *Phys. Rev. D* **79** (2009) 011101, doi:10.1103/PhysRevD.79.011101, arXiv:0809.3781.

[6] D0 Collaboration, "Model independent search for new phenomena in $p\bar{p}$ collisions at $\sqrt{s} = 1.96$ TeV", *Phys. Rev. D* **85** (2012) 092015, doi:10.1103/PhysRevD.85.092015, arXiv:1108.5362.

[7] CMS Collaboration, "MUSiC, a Model Unspecific Search for New Physics, in pp Collisions at $\sqrt{s} = 8\,\mathrm{TeV}$", 2017. CMS-PAS-EXO-14-016.

[8] ATLAS Collaboration, "A strategy for a general search for new phenomena using data-derived signal regions and its application within the ATLAS experiment", *Eur. Phys. J. C* **79** (2019), no. 2, 120, doi:10.1140/epjc/s10052-019-6540-y, arXiv:1807.07447.

[9] E. Gross and O. Vitells, "Trial factors for the look elsewhere effect in high energy physics", *Eur. Phys. J. C* **70** (2010) 525, doi:10.1140/epjc/s10052-010-1470-8, arXiv:1005.1891.

[10] C. Weisser and M. Williams, "Machine learning and multivariate goodness of fit", arXiv:1612.07186.

[11] O. Cerri et al., "Variational Autoencoders for New Physics Mining at the Large Hadron Collider", *JHEP* **05** (2019) 036, doi:10.1007/JHEP05(2019)036, arXiv:1811.10276.

[12] R. T. D'Agnolo and A. Wulzer, "Learning New Physics from a Machine", *Phys. Rev. D* **99** (2019), no. 1, 015014, doi:10.1103/PhysRevD.99.015014, arXiv:1806.02350.

[13] A. De Simone and T. Jacques, "Guiding New Physics Searches with Unsupervised Learning", *Eur. Phys. J. C* **79** (2019), no. 4, 289, doi:10.1140/epjc/s10052-019-6787-3, arXiv:1807.06038.

[14] M. Farina, Y. Nakai, and D. Shih, "Searching for New Physics with Deep Autoencoders", *Phys. Rev. D* **101** (2020), no. 7, 075021, doi:10.1103/PhysRevD.101.075021, arXiv:1808.08992.

[15] J. H. Collins, K. Howe, and B. Nachman, "Anomaly Detection for Resonant New Physics with Machine Learning", *Phys. Rev. Lett.* **121** (2018), no. 24, 241803, doi:10.1103/PhysRevLett.121.241803, arXiv:1805.02664.

[16] A. Blance, M. Spannowsky, and P. Waite, "Adversarially-trained autoencoders for robust unsupervised new physics searches", *JHEP* **10** (2019) 047, doi:10.1007/JHEP10(2019)047, arXiv:1905.10384.

[17] J. Hajer, Y.-Y. Li, T. Liu, and H. Wang, "Novelty Detection Meets Collider Physics", *Phys. Rev. D* **101** (2020), no. 7, 076015, doi:10.1103/PhysRevD.101.076015, arXiv:1807.10261.

[18] T. Heimel, G. Kasieczka, T. Plehn, and J. M. Thompson, "QCD or What?", *SciPost Phys.* **6** (2019), no. 3, 030, doi:10.21468/SciPostPhys.6.3.030, arXiv:1808.08979.

[19] J. H. Collins, K. Howe, and B. Nachman, "Extending the search for new resonances with machine learning", *Phys. Rev. D* **99** (2019), no. 1, 014038, doi:10.1103/PhysRevD.99.014038, arXiv:1902.02634.

[20] R. T. D'Agnolo et al., "Learning multivariate new physics", *Eur. Phys. J. C* **81** (2021), no. 1, 89, doi:10.1140/epjc/s10052-021-08853-y, arXiv:1912.12155.

[21] B. Nachman and D. Shih, "Anomaly Detection with Density Estimation", *Phys. Rev. D* **101** (2020) 075042, doi:10.1103/PhysRevD.101.075042, arXiv:2001.04990.

[22] A. Andreassen, B. Nachman, and D. Shih, "Simulation assisted likelihood-free anomaly detection", *Physical Review D* **101** (May, 2020) doi:10.1103/physrevd.101.095004.

[23] O. Amram and C. M. Suarez, "Tag N' Train: a technique to train improved classifiers on unlabeled data", *JHEP* **01** (2021) 153, doi:10.1007/JHEP01(2021)153, arXiv:2002.12376.

[24] B. M. Dillon, D. A. Faroughy, J. F. Kamenik, and M. Szewc, "Learning the latent structure of collider events", *JHEP* **10** (2020) 206, doi:10.1007/JHEP10(2020)206, arXiv:2005.12319.

[25] T. Cheng et al., "Variational Autoencoders for Anomalous Jet Tagging", arXiv:2007.01850.

[26] C. K. Khosa and V. Sanz, "Anomaly Awareness", arXiv:2007.14462.

[27] B. Nachman, "Anomaly Detection for Physics Analysis and Less than Supervised Learning", arXiv:2010.14554.

[28] S. E. Park et al., "Quasi Anomalous Knowledge: Searching for new physics with embedded knowledge", *JHEP* **21** (2020) 030, doi:10.1007/JHEP06(2021)030, arXiv:2011.03550.

[29] B. Bortolato, B. M. Dillon, J. F. Kamenik, and A. Smolkovič, "Bump Hunting in Latent Space", arXiv:2103.06595.

[30] J. H. Collins, P. Martín-Ramiro, B. Nachman, and D. Shih, "Comparing weak- and unsupervised methods for resonant anomaly detection", *Eur. Phys. J. C* **81** (2021), no. 7, 617, doi:10.1140/epjc/s10052-021-09389-x, arXiv:2104.02092.

[31] T. Finke et al., "Autoencoders for unsupervised anomaly detection in high energy physics", *JHEP* **06** (2021) 161, doi:10.1007/JHEP06(2021)161, arXiv:2104.09051.

[32] J. Gonski, J. Lai, B. Nachman, and I. Ochoa, "High-dimensional Anomaly Detection with Radiative Return in e^+e^- Collisions", arXiv:2108.13451.

[33] A. Hallin et al., "Classifying Anomalies THrough Outer Density Estimation (CATHODE)", arXiv:2109.00546.

[34] B. Ostdiek, "Deep Set Auto Encoders for Anomaly Detection in Particle Physics", arXiv:2109.01695.

[35] G. Kasieczka et al., "The LHC Olympics 2020: A Community Challenge for Anomaly Detection in High Energy Physics", arXiv:2101.08320.

[36] T. Aarrestad et al., "The Dark Machines Anomaly Score Challenge: Benchmark Data and Model Independent Event Classification for the Large Hadron Collider", arXiv:2105.14027.

[37] G. Papamakarios et al., "Normalizing flows for probabilistic modeling and inference", *J. Mach. Learn. Res.* **22** (2021), no. 57, 1, arXiv:1912.02762.

[38] S. Caron, L. Hendriks, and R. Verheyen, "Rare and Different: Anomaly Scores from a combination of likelihood and out-of-distribution models to detect new physics at the LHC", arXiv:2106.10164.

[39] B. Ostdiek, "Deep Set Auto Encoders for Anomaly Detection in Particle Physics", arXiv:2109.01695.

[40] DarkMachines Community, "Unsupervised-hackathon", 2020. doi:10.5281/zenodo.3961917.

[41] F. Pedregosa et al., "Scikit-learn: Machine learning in Python", *J. Mach. Learn. Res.* **12** (2011) 2825.

[42] H. G. Barrow, J. M. Tenenbaum, R. C. Bolles, and H. C. Wolf, "Parametric correspondence and Chamfer matching: Two new techniques for image matching", in *Proceedings of the 5th International Joint Conference on Artificial Intelligence (KJCAI)*, volume 2, p. 659. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1977.

[43] H. Fan, H. Su, and L. J. Guibas, "A point set generation network for 3D object reconstruction from a single image", in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, p. 2463. 6, 2017. arXiv:1612.00603. doi:10.1109/CVPR.2017.264.

[44] Y. Zhang, J. Hare, and A. Prügel-Bennett, "FSPool: Learning set representations with featurewise sort pooling", in *8th International Conference on Learning Representations*. 2020. arXiv:1906.02795.

[45] D. Rezende and S. Mohamed, "Variational inference with normalizing flows", arXiv:1505.05770.

[46] I. Higgins et al., "beta-VAE: Learning basic visual concepts with a constrained variational framework", in *5th International Conference on Learning Representations*. 2017.

[47] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization", in *3rd International Conference for Learning Representations*. 2015. arXiv:1412.6980.

[48] A. Paszke et al., "PyTorch: An imperative style, high-performance deep learning library", in *Advances in Neural Information Processing Systems*, H. Wallach et al., eds., volume 32. Curran Associates, Inc., 2019. arXiv:1912.01703.

[49] P. Jawahar and M. Pierini, "mpp-hep/DarkFlow repository", 2021.

[50] M. M. Bronstein et al., "Geometric deep learning: Going beyond Euclidean data", *IEEE Signal Processing Magazine* **34** (2017), no. 4, 18, doi:10.1109/MSP.2017.2693418.

[51] J. Zhou et al., "Graph neural networks: A review of methods and applications", arXiv:1812.08434.

[52] J. Shlomi, P. Battaglia, and J.-R. Vlimant, "Graph neural networks in particle physics", *Mach. Learn.: Sci. Tech.* **2** (7, 2020) 021001, doi:10.1088/2632-2153/abbf9a, arXiv:2007.13681.

[53] J. Duarte and J.-R. Vlimant, "Graph neural networks for particle tracking and reconstruction", in *Artificial Intelligence for High Energy Physics*, P. Calafiura, D. Rousseau, and K. Terao, eds. World Scientific Publishing, 12, 2020. arXiv:2012.01249. Submitted to *Int. J. Mod. Phys. A*. doi:10.1142/12200.

[54] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks", in *5th International Conference on Learning Representations*. 2017. arXiv:1609.02907.

[55] R. v. d. Berg, L. Hasenclever, J. M. Tomczak, and M. Welling, "Sylvester normalizing flows for variational inference", *arXiv preprint arXiv:1803.05649* (2018).

[56] D. P. Kingma et al., "Improving variational inference with inverse autoregressive flow", in *Advances in Neural Information Processing Systems*, D. Lee et al., eds., volume 29. Curran Associates, Inc., 2016. arXiv:1606.04934.

[57] G. Zheng, Y. Yang, and J. Carbonell, "Convolutional normalizing flows", in *ICML 2018 Workshop on Theoretical Foundations and Applications of Deep Generative Models*. 2018. arXiv:1711.02255.

[58] M. Germain, K. Gregor, I. Murray, and H. Larochelle, "MADE: Masked autoencoder for distribution estimation", arXiv:1502.03509.