MACHINE LEARNING
Science and Technology

PURPOSE-LED PUBLISHING™

Part of the MACHINE LEARNING Series

PAPER • OPEN ACCESS

It’s not a FAD: first demonstration of flows for unsupervised anomaly detection at 40 MHz for use at the Large Hadron Collider

To cite this article: Francesco Vaselli *et al* 2026 *Mach. Learn.: Sci. Technol.* **7** 025052

IOP Publishing Mach. Learn.: Sci. Technol. 7 (2026) 025052 https://doi.org/10.1088/2632-2153/ae51dd

MACHINE
LEARNING
Science and Technology

Check for updates

OPEN ACCESS

RECEIVED
20 November 2025

REVISED
16 February 2026

ACCEPTED FOR PUBLICATION
13 March 2026

PUBLISHED
9 April 2026

PAPER

It's not a FAD: first demonstration of flows for unsupervised anomaly detection at 40 MHz for use at the Large Hadron Collider

Francesco Vaselli[1,6,*], Chang Sun[4], Thea Aarrestad[3], Dimitrios Danopoulos[1], Roope Oskari Niemi[1], Maciej Mikolaj Glowacki[1], Katya Govorkova[2], Vladimir Loncar[5], Felice Pantaleo[1] and Maurizio Pierini[1]

[1] European Organization for Nuclear Research (CERN), CH-1211 Geneva 23, Switzerland
[2] Massachusetts Institute of Technology, Cambridge, MA, United States of America
[3] ETH, Federal Institute of Technology Zurich, Zurich CH, Switzerland
[4] California Institute of Technology, Pasadena, CA, United States of America
[5] Institute of Physics, Belgrade, Serbia
[6] Also at Scuola Normale Superiore, Pisa, Italy and Istituto Nazionale di Fisica Nucleare (INFN) sezione di Pisa, Italy.
[*] Author to whom any correspondence should be addressed.

E-mail: francesco.vaselli@cern.ch

Keywords: anomaly detection, normalizing flows, flow matching, machine learning for L1 triggers

Abstract

We present the first implementation of a continuous normalizing flow (CNF) model for unsupervised anomaly detection within the realistic, high-rate environment of the Large Hadron Collider's L1 trigger systems. While CNFs typically define an anomaly score via a probabilistic likelihood, calculating this score requires solving an ordinary differential equation, a procedure too complex for field programmable gate array (FPGA) deployment. To overcome this, we propose a novel, hardware-friendly anomaly score defined as the squared norm of the model's vector field output. This score is based on the intuition that anomalous events require a larger transformation by the flow, and it is shown to be physically interpretable as the norm of the input features for our specific training choice. Our model, trained via flow matching on standard model (SM) data, is synthesized for an FPGA using the hls4ml and da4ml libraries. We demonstrate that our approach effectively identifies a variety of beyond-the-SM signatures with performance comparable to existing machine learning-based triggers. The algorithm achieves a latency of a few hundred nanoseconds, or even less when using advanced quantization techniques, and requires minimal FPGA resources, establishing CNFs as a viable new tool for real-time, data-driven discovery at 40 MHz.

1. Introduction

The 40 million proton-proton collisions per second at the CERN Large Hadron Collider (LHC) [1] present a formidable data challenge. The sensors in the ATLAS [2] and CMS [3] detectors output hundreds of terabytes of data per second, which must be filtered in real-time to decide what to save on disk for further analysis. This is accomplished by a two-stage trigger system: a hardware-based Level-1 Trigger (L1T) [4–7] on *field programmable gate arrays* (FPGAs), which reduces the data rate by almost a factor of 1000 within microseconds, followed by a software-based high-level trigger (HLT) on a CPU farm.

The selection algorithms in these triggers are typically supervised, targeting specific physics signatures motivated by theory, as exemplified by the successful search for the Higgs boson [8, 9]. This paradigm, however, may fail to identify unforeseen new physics. Consequently, unsupervised anomaly detection (AD) has been increasingly investigated as a possible solution. The general strategy behind unsupervised AD is to define a metric to distinguish events based on their abnormality from the usual data streams. The abnormality is something not defined *a priori* but emergent directly from data. There is significant community effort to use unsupervised machine learning for this type of model-agnostic searches [10, 11]. These efforts have explored autoencoders [12] for offline processing and have proposed their integration into the HLT to capture rare events in a special data stream [13, 14], echoing earlier non-ML strategies [15].

More recently, a new approach to AD argued for deploying these algorithms earlier in the data processing chain: at the L1T. Placing an AD search at this stage avoids the selection biases introduced by the standard L1T algorithms, which often discard potentially interesting low-energy topologies. An unbiased, model-independent AD trigger can select events based on their abnormality, not a specific signature, thus maximizing discovery potential. Govorkova *et al* [16] pioneered this new type of strategy using autoencoders as the AD algorithm to be ported to the L1T.

Since 2023, the CMS Collaboration is operating autoencoder-based AD algorithms. At first, a VAE based on high-level features (named AXOL1TL [17]) was deployed, following the strategy highlighted in [16]. Then, a convolutional autoencoder (named CICADA [17]) was put in use, using knowledge distillation as a compression algorithm [18]. In 2025, ATLAS took the first steps towards the same strategy, with the deployment of GELATO [19]. All these algorithms consider autoencoders as the baseline tool for AD in real-time, as originally proposed in [16].

To the best of our knowledge, the present work is the first proof of concept for the application of a new type of ML model at the L1T for unsupervised AD: *continuous normalizing flows* (CNFs) [20].

Unlike VAEs, which optimize a lower bound (ELBO) on the data likelihood and compress inputs into a lower-dimensional latent space—potentially losing information—NFs learn an exact, bijective mapping between the data and the latent distribution. This fundamental difference in the training objective suggests that CNFs may exhibit complementary sensitivity to anomalous signatures compared to autoencoder-based approaches. Furthermore, the vector field learned by the flow offers a direct physical interpretation of the anomaly score, which, as we discuss in section 4, can correlate with robust physical quantities such as the input norm.

Discrete NF have already been considered as AD tools for high energy physics, but only in the context of offline data analysis, see [11, 21, 22]. We are not aware of any work in the community leveraging CNFs for offline AD. In such a case the AD score would likely employ the capacity of these algorithms to model the distribution of the training data, which is an idea already discussed in the context of discrete flows and which predates this work (see e.g. [11]) using scores similar to the flow ordinary differential equation (ODE) score introduced in section 4.

Implementing these algorithms at the L1T, with its severe latency ($<1\,\mu$s) and resource constraint, is made possible by open-source `hls4ml` [23–26] and `da4ml` [27] libraries, providing a pathway to deploy neural networks and other ML models on FPGAs. By generating highly optimized, fully on-chip firmware, these tools can meet the L1T latency and throughput requirements (initiation interval <150 ns, related to the bunch-crossing time for the LHC operations). Moreover, they support quantization-aware training (QAT) [28] that potentially allows extreme model compression and a further reduction of FPGA resource consumption.

The main contributions of this work are the following:

- We train a continuous NF on a realistic dataset of low-level features of physics objects standard model (SM) signatures, and from there we define an *AD score* suitable for inference on FPGA;
- We evaluate results on different benchmarks for new physics scenarios at floating point precision, comparing it to a similar architecture proposed by Govorkova *et al* [16], and observing similar performances;
- We compress the model through different strategies, namely *post training quantization* (PTQ) or *QAT* using *high granularity quantization* (HGQ). We then port the results to FPGA, showing that the resource usage is well within the requirements of a L1T trigger system.

2. Data samples

We use the datasets employed by [16], published on Zenodo [29–32] and discussed in detail in [33].

The SM data sample represents a typical proton–proton collision dataset that has been pre-filtered by requiring the presence of an electron or a muon with a transverse momentum $p_T > 23$ GeV and a pseudo-rapidity $|\eta| < 3$ (electron) and $|\eta| < 2.1$ (muon). This is representative of a typical L1T selection algorithm of a multipurpose LHC experiment. In addition to this, we consider the four benchmark new physics scenarios discussed in Cerri *et al* [13]:

- A leptoquark (LQ) with a mass of 80 GeV, decaying to a b quark and a τ lepton: LQ $\to b\tau$, 340 544 events [30],

- A neutral scalar boson (A) with a mass of 50 GeV, decaying to two off-shell Z bosons, each forced to decay to two leptons: $A \to 4\ell$, 55 969 events [29],
- A scalar boson with a mass of 60 GeV, decaying to two tau leptons: $h^0 \to \tau\tau$, 691 283 events [31],
- A charged scalar boson with a mass of 60 GeV, decaying to a tau lepton and a neutrino: $h^{\pm} \to \tau\nu$, 760 272 events [32].

These four processes are used to evaluate the accuracy of the trained models.

In total, we use 3.5 million events from the background sample for training. The new physics benchmark samples are only used for evaluating the performance of the models, along with an additional 2 million background events not used in training.

2.1. Preprocessing

We apply a simple standard-scaling operation in training and inference, with each feature transformed independently to have zero sample-mean and unit-sample-variance. This operation is usually replicated on FPGA with a subtraction and division, and, while it implies the use of a limited amount of extra resources, it could theoretically be applied upstream in the board and be used for running multiple algorithms. Thus, it is not included in the resources estimate of the flow algorithm.

We note that the performance of the algorithm is very much dependent on the specific preprocessing operation being applied, with standard-scaling being one of the best performing strategy for our new-physics benchmarks.

3. NF models

NFs are a class of generative models that learn an explicit representation of an unknown data probability density function, $p(\mathbf{x})$. They achieve this by defining an invertible and differentiable mapping, f, between the complex data space and a simple base distribution (typically a standard Gaussian with mean 0 and standard deviation 1; this is also the choice made in this work), $p(\mathbf{z})$:

$$\mathbf{x} = f(\mathbf{z}) \quad \text{and} \quad \mathbf{z} = f^{-1}(\mathbf{x}). \tag{1}$$

We refer to [20] for a comprehensive review of existing NF algorithms.

3.1. CNFs

A *CNF* defines the invertible mapping f not as a single function, but as the solution to a differential equation parameterized by a continuous variable $t \in [0,1]$. This establishes a 'probability path' that smoothly transforms the base distribution p_0 into the target data distribution p_1. The dynamics of this transformation are governed by a time-dependent vector field $\mathbf{v}_t$, which is parameterized by a neural network with parameters ϕ:

$$\frac{\mathrm{d}\mathbf{z}(t)}{\mathrm{d}t} = \mathbf{v}(\mathbf{z}(t), t|\phi), \tag{2}$$

with the initial condition $\mathbf{z}(0)$ being a sample from the base distribution, $\mathbf{z}(0) \sim p(\mathbf{z})$. A data-like sample $\mathbf{x}$ is generated by solving this *ODE* from $t = 0$ to $t = 1$, yielding $\mathbf{x} = \mathbf{z}(1)$. Note that the dimensionality of the vector field $\mathbf{v}_t$ is the same as the input features $\mathbf{x}$.

3.2. Flow matching

A powerful and stable method for training CNFs is *flow matching* [34, 35]. This approach recasts the training objective as a simple regression problem. The goal is to make the model's vector field $\mathbf{v}_t$ match a predefined target vector field $\mathbf{u}_t$. This target field $\mathbf{u}_t$ is constructed to transport samples from the base distribution towards samples from the data distribution along a specific 'probability path' p_t.

Crucially, the path p_t and its corresponding vector field $\mathbf{u}_t$ can be constructed conditionally for each training sample $\mathbf{x}$. This simplifies the training into a regression of the model's output $\mathbf{v}_t$ onto the target $\mathbf{u}_t$ at random points along these paths. The loss function for the model parameters ϕ is then:

$$\mathcal{L}_{\mathrm{FM}}(\phi) = \mathbb{E}_{t,\mathbf{x},\mathbf{z}_t}\left[\|\mathbf{v}(\mathbf{z}_t, t|\phi) - \mathbf{u}(\mathbf{z}_t, t|\mathbf{x})\|^2\right], \tag{3}$$

where $t \sim \mathrm{Unif}[0,1]$, $\mathbf{x} \sim p(\mathbf{x})$, and $\mathbf{z}_t$ is a point sampled from the conditional path distribution $p_t(\mathbf{z}|\mathbf{x})$. This is a straightforward regression loss, making the model relatively easy to train. In this work, we draw from recent developments reviewed in [35]. We refer the reader there for a detailed discussion on the construction of various probability paths and their associated vector fields.

Specifically, for this work the training dynamics governed by:

$$\mathbf{z}_t(\mathbf{x}, \mathbf{z}_0) = t\mathbf{x} + (1-t)\,\mathbf{z}_0, \quad p_0(\mathbf{z}_0 \mid \mathbf{x}) = p_0(\mathbf{z}_0).$$

which corresponds to a target field $\mathbf{u}_t(\mathbf{z}_t, t|\mathbf{x}) = \mathbf{x} - \mathbf{z}_0$. As we will discuss in the next section, this has a profound impact on the anomaly scores which we can define from our model.

3.3. Model architecture and training

The input to our model consists of the kinematic variables—transverse momentum (p_T), pseudorapidity (η), and azimuthal angle (ϕ)—for 18 reconstructed physics objects. These objects are ordered by type: 4 muons, 4 electrons, and 10 jets. The list is augmented with the missing transverse energy, for which the magnitude and ϕ are used, while its η component is set to zero. This creates a fixed-size input tensor of shape (19, 3). For events with fewer than the maximum number of objects of a given type, the corresponding input slots are zero-padded, a standard practice in L1T algorithm design.

Before being processed by the network, this (19, 3) tensor is flattened into a single 57-dimensional vector. As detailed previously, a standard scaling transformation is applied to these input features. This preprocessing step is a simple affine transformation that can be readily implemented on an FPGA as a bit-shift operation; since its resource usage is minimal, it is omitted from the following discussion.

The core of our model is the neural network that parameterizes the time-dependent vector field $\mathbf{v}_t(\mathbf{z}_t, t|\phi)$ of the CNF. After a basic hyperparameter scan, we settle for simple multilayer perceptron (MLP) architecture consisting of two hidden layers, each with 16 nodes, and using the rectified linear unit as the activation function. The network takes a 58-dimensional flattened input vector, i.e. the 57 physics features plus an additional input for the current timestamp t, and outputs a 57-dimensional vector, which represents the vector field $\mathbf{v}_t$ required for the flow transformation. This simple architecture amounts to 1913 trainable parameters.

The model was trained for 100 epochs using the Adam optimizer with a learning rate of 10^{-3} and a batch size of 1024. The network parameters were optimized by minimizing the flow matching loss described in section 3 (with the scheduler called `CondOTScheduler()` and the `AffineProbPath()` defined in [35] to build the loss function), using the `PyTorch` [36] and `flow-matching` [35] libraries.

3.4. VAE model for comparison

In order to compare to the fully-connected vae architecture of [16], we decide to retrain the same DNN VAE architecture. We refer the reader to section 3 of that work for details about the implementation; we train with the same conditions and the hyperparameter $\beta = 0.8$, plus the same preprocessing used for the flow architecture.

We could not exactly reproduce the results obtained in [16]. In addition to possible differences in the underlying training software, a possible cause of discrepancy could originate from undocumented preprocessing steps, with respect to the code published by the authors, see [37, 38]. In our experiments, we do observe a large performance dependence on the applied processing. For consistency reasons, we take our trained version of the VAE as the reference in our plots, but we still report the original results of [16] in table 1.

4. AD scores

Once trained, CNFs map an input data point $\mathbf{x}$ to a latent variable $\mathbf{z}$ by solving the ODE that defines the model, see equation (2).

We denote as $\mathbf{v}_t(\mathbf{z}(t), t|\phi)$ the neural network with parameters ϕ that defines a time-dependent vector field. For this application, the integration runs from $t = 1$, where $\mathbf{z}(1) = \mathbf{x}$, to $t = 0$, where $\mathbf{z}(0)$ is the corresponding point in the latent Gaussian space.

4.1. Flow ODE

The canonical approach to define an anomaly score for flows is the negative Gaussian log-likelihood of the latent point $\mathbf{z}(0)$. This score has a direct probabilistic interpretation, making it an elegant definition. However, calculating it requires integrating the ODE, a procedure too complex to implement efficiently on an FPGA. This would involve:

1. Starting from the data point $\mathbf{x}$ at timestep $t = 1$.
2. Executing a forward pass of the model to get the vector field $\mathbf{v}_t$.
3. Solving the current step of the ODE integrator using this vector field.
4. Repeating steps 2 and 3 until the integration reaches $t = 0$ to find the latent point $\mathbf{z}(0)$.
5. Evaluating the Gaussian log-likelihood of $\mathbf{z}(0)$ to get the final score.

This iterative process is clearly infeasible within the strict latency constraints of the L1T. In what follows, we still compute this score on a CPU with floating-point precision as a comparison, and we refer to it as *Flow ODE*. We show results obtained using the `euler` solver with just 2 timesteps from 1 to 0. From a preliminary investigation, when increasing the timesteps of integration performances are anyways similar to those showed in section 5 for the current choice of timesteps.

4.2. Flow $\mathbf{v}_t$

To create an FPGA-compatible alternative, we need an anomaly score that avoids this integration. We propose a new score based on the intuition that anomalous points, being further from the training distribution, require a larger 'push' from the vector field to be mapped toward the latent prior. Therefore, the magnitude of the vector field itself, i.e. the output of the model, can serve as a proxy for anomaly. We define our score as the squared norm of the vector field evaluated at the initial point $\mathbf{x}$:

$$\mathcal{AS}(\mathbf{x}) = \|\mathbf{v}_t(\mathbf{x}, t|\phi)\|^2 = \sum_i \left(v_t(\mathbf{x}, t|\phi)_i\right)^2. \tag{4}$$

This strategy has the key advantage of requiring only a single forward pass of the model, making it suitable for fast inference. We refer to this approach as *Flow* $\boldsymbol{v}_t$.

The value of this score depends on the timestep t at which the model is evaluated. For this work, we choose to evaluate the field at the beginning of the trajectory, $t = 1$, as this is where the model first encounters the input data. We note that other choices are possible, and recent developments in single-step or straight-path Flow models may offer other optimal evaluation points.

It is important to note that, for the specific case of our choice of target, $\mathbf{u}_t(\mathbf{z}_t, t|\mathbf{x}) = \mathbf{x} - \mathbf{z}_0$, since $\mathbf{z}_0$ is sampled from a distribution with mean 0 our model output will converge to the following:

$$\mathbb{E}_{\mathbf{z}_0 \sim p_0}[\mathbf{v}_1(\mathbf{x})] = \mathbb{E}_{\mathbf{z}_0 \sim p_0}[\mathbf{x} - \mathbf{z}_0] = \mathbf{x}.$$

This provides an intuitive physical interpretation: the anomaly score Flow $\mathbf{v}_t$ functions similarly to a distance metric (or norm) $\|\mathbf{x}\|^2$ in the feature space. While a simple norm calculation would be computationally cheaper, the Flow model retains the flexibility to learn more complex, non-linear manifold structures if the data distribution requires it, and generalizes to other time-slices where this simplification does not hold. In the results, we also display the performance of the norm operation as anomaly score in order to check whether this interpretation is actually verified.

4.3. VAE score

In order to present a comparison with a similar architecture as the one used in [16], we use the Kullback–Leibler score D_{KL} used in the loss function as the anomaly score for this model:

$$D_{\mathrm{KL}} = \frac{1}{2}\sum_{j=1}^{2}\left(\sigma_j^2 + \mu_j^2 - \log\left(\sigma_j^2\right) - 1\right) \tag{5}$$

and we refer to it as *VAE* or D_{KL} in the following.

4.4. Further considerations and outlook

Finally, we would like to note that a vector-field-based approach also opens avenues for other potential scores, such as the *divergence* of the field (at the cost of multiple model evaluations), and provides a conceptual link to other ODE-based generative models like *diffusion models* [39]. In this paper, however, we focus on the first two approaches described, leaving the investigation of these alternatives to future work.

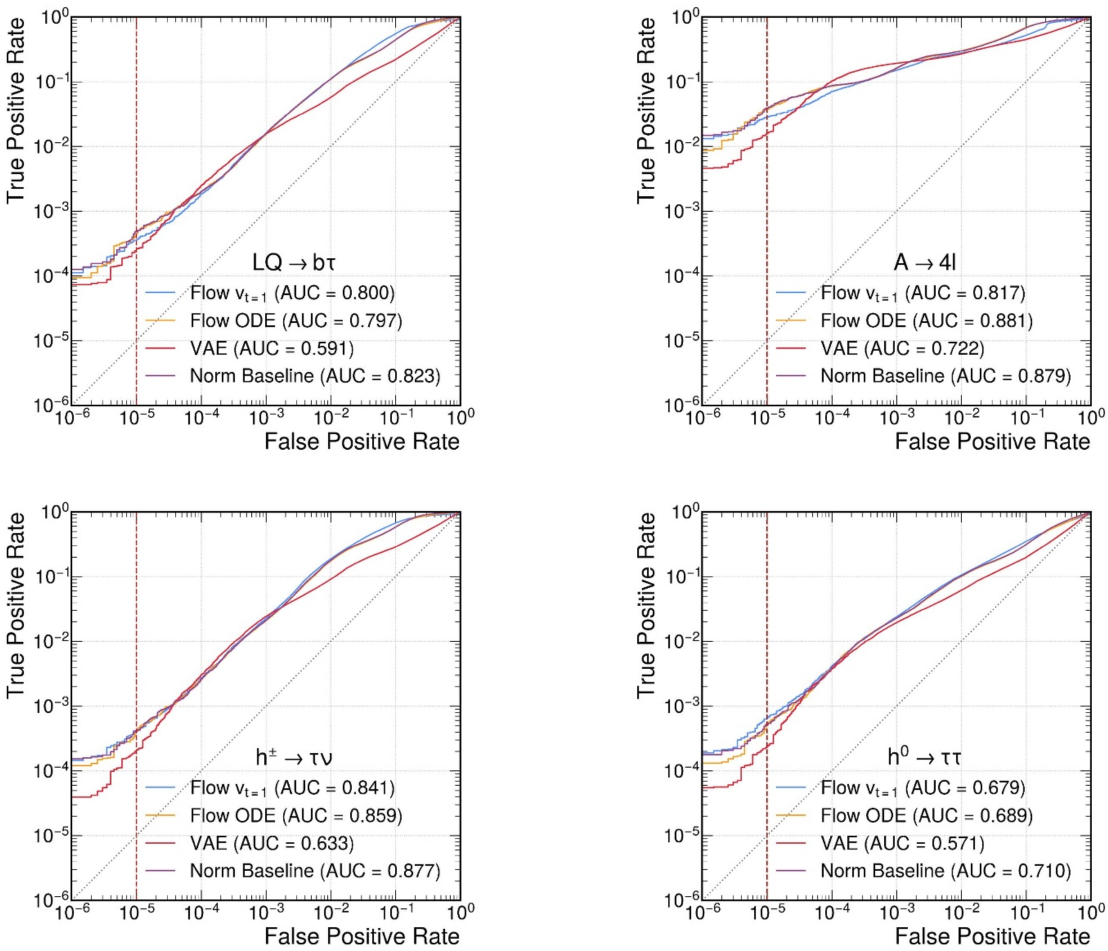

Figure 1. ROC curves for the 4 anomalous samples. For each sample we show the results for the flow, for both the *Flow* $\mathbf{v}_t$ and *Flow ODE* anomaly scores, and for the VAE. We show as a dashed red line the threshold of 10^{-5} FPR at which we evaluate the fraction of TPR for each model; and as a dotted gray line the baseline performance of a random classifier. The flow is capable of detecting the anomalies, with both ROC AUC and TPR better than a VAE trained under the same condition.

Table 1. Physics performance comparison of different anomaly detection models and scores. Performance is measured by TPR at a fixed FPR of 10^{-5} and the area Under the ROC curve (AUC).

		TPR @ FPR 10^{-5} (%)				AUC (%)			
Model	AD score	$LQ \to b\tau$	$A \to 4\ell$	$h^{\pm} \to \tau\nu$	$h^0 \to \tau\tau$	$LQ \to b\tau$	$A \to 4\ell$	$h^{\pm} \to \tau\nu$	$h^0 \to \tau\tau$
VAE from [16]	D_{KL}	0.07	5.27	0.08	0.11	92	94	94	81
Flow	$\mathbf{v}_t$	0.04	2.8	0.04	0.06	80	82	84	68
	ODE	0.04	3.8	0.04	0.05	80	88	86	69
Norm	$\|\|\mathbf{x}\|\|^2$	0.05	3.9	0.04	0.05	82	88	88	71
VAE (Ours)	D_{KL}	0.02	2.4	0.02	0.04	59	72	63	57

5. Results at floating point precision

The model performance is assessed using the four new physics benchmark models. The AD scores considered are the *Flow* $\mathbf{v}_t$ and *Flow ODE* (as offline comparison) for the Flow model, and the D_{KL} for the VAE model. The model is first validated at floating point precision, also know as single precision or 32-bit precision.

The receiver operating characteristic (ROC) curves in figure 1 show the true positive rate (TPR) as a function of the false positive rate (FPR), computed by changing the lower threshold applied on the different anomaly scores. We further quantify the AD performance quoting the area under the ROC curve (AUC) and the TPR corresponding to a FPR working point of 10^{-5} (see table 1), which on this dataset corresponds to the reduction of the background rate to approximately 1000 events per month. We show as a dashed red line the working point of 10^{-5} FPR; and as a dotted gray line the baseline performance of a random classifier.

Table 1 summarizes the results numerically for both ROC AUC score and TPR, reporting a comparison with the D_{KL} anomaly score from [16] as well.

Looking at the results, we see that the flow is capable of detecting anomalies when using either the $\mathbf{v}_t$ or the *ODE* anomaly scores. The *ODE* score has marginally better ROC AUC scores, but similar performances when comparing the TPR values.

We attribute the slight performance advantage of the *Flow ODE* score to the fact that it evaluates the exact negative log-likelihood of the data point. This calculation incorporates information accumulated along the entire integration path, including both the final latent position and the deformation of the probability density (captured by the divergence of the vector field). In contrast, the Flow $\mathbf{v}_t$ score captures only the instantaneous velocity at $t = 1$. While this serves as an excellent proxy for the transport distance (and thus the input norm), it misses the fine-grained density information contained in the full likelihood path. This suggests that while 'distance from the mean' is the dominant factor in AD on our anomaly datasets, the subtle correlations captured by the full ODE integration do add discrimination power in some instances.

As expected from our discussion in section 4, we also observe that the Flow $\mathbf{v}_t$ score is equivalent to the norm $||\mathbf{x}||^2$ for our specific choice of training target. Furthermore, we observe that TPR and AUC scores for the norm are actually compatible or better than those of the Flow ODE score. We explain this behavior with the fact that, for small Flow models such as this, the ODE path learned by the model may be quite simple and capture essentially the average displacement of the point.

Interestingly, the VAE model achieves lower performance than the flow on all processes. As already noticed, the VAE performance does not match those shown in [16]. This could be due to differences in preprocessing with respect to the published dataset. This does not make our comparison less meaningful, since we used the same preprocessing for both our VAE implementation and the Flow model.

We attribute the differences between the model results in this work and the ones of [16] to:

1. A difference in the data partitions used for training;
2. A difference in the exact preprocessing operations used on the data (in which case it would be interesting to assess the performance of the flow under the same preprocessing);
3. Any remaining difference in the hyperparameters of the NN/training.

6. Model compression and HLS C simulation results

As mentioned in section 1, we perform two different compression strategies for porting the model to FPGA.

6.1. PTQ

We perform some tests and find that a simple, brute force casting of the precision to 18 bits *post-training* is enough for synthesizing the algorithm while having a good tradeoff between performance and resource consumption. The actual weights of the network can be casted down to 12 bits, leaving the 18 bits casting for biases and other operations.

However, we observe that significantly less compression is achievable by default on the `einsum` operation, responsible for computing the norm squared of the model's output. We find that at least 23 bits are required for casting this operation and retain a good performance, meaning that the majority of resources spent and of the latency will be due to this operation alone.

Performance estimates from the HLS C Simulation of the PTQ model are reported in table 2. For the same quantization, we report in table 3 the resource usage, latency, and initiation interval for the PTQ model deployed on a Xilinx Virtex UltraScale+ FPGA. Resources are based on the Vivado estimates from Vivado HLS.

We observe a good retention of performance, with comparable results to the un-casted model on the TPR and minimal drops of performance on the ROC AUC. The resources being used are a fraction of the available ones on the board, around 7%, making it possible to host this algorithm on FPGA along with other trigger algorithms, as is usually the case for these applications. The latency of 230 ns and the initiation interval of 5 ns make the algorithm fully compatible with the constraints from the trigger rates of a large LHC experiment and the LHC operational parameters of Run 3.

6.2. QAT

The model trained with QAT is trained using the HGQ method [40].

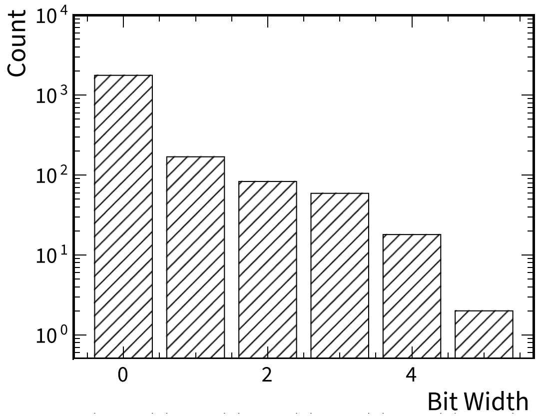

Figure 2. Distribution of bit-widths for the weights of the HGQ-trained model. 84.2% of the weights are quantized to 0 bits (pruned), while the remaining weights are distributed between 1 and 5 bits.

Table 2. Physics performance comparison for the PTQ and HGQ model. Performance is measured by the TPR at a fixed FPR of 10^{-5} and the area under the ROC curve (AUC). For the AUC we also report the ratio with the AUC score of the offline Flow $\mathbf{v}_t$ model.

	TPR @ FPR 10^{-5} (%)				AUC (%)			
Model	$\mathrm{LQ} \to b\tau$	$A \to 4\ell$	$h^{\pm} \to \tau\nu$	$h^0 \to \tau\tau$	$\mathrm{LQ} \to b\tau$	$A \to 4\ell$	$h^{\pm} \to \tau\nu$	$h^0 \to \tau\tau$
Flow PTQ	0.03	3.6	0.04	0.06	75 (0.94)	81 (0.92)	81 (0.94)	65 (0.94)
Flow HGQ	0.04	3.4	0.05	0.06	77 (0.96)	86 (0.97)	82 (0.95)	66 (0.96)

Table 3. Post-routing resource usage (as total estimates and as percentages over the whole board resources), latency, and initiation interval for the PTQ and HGQ model deployed on a Xilinx Virtex UltraScale+ xcu250-figd2104-2L-e FPGA. The designs are implemented Vivado 2025.1, with a target clock period of 5 ns out-of-context.

Model	DSP	LUT	FF	BRAM	Latency (ns)	II (clk)
Flow PTQ	916 (7.45%)	40, 835 (2.36%)	11, 397 (0.33%)	0	230	1
Flow HGQ	28 (<0.01%)	5,978 (0.34%)	1,683 (0.05%)	0	35	1

We initialize the model with 6-fractional bits for weights and activations. We then train the model for 2000 epochs with a cosine decay restarting learning schedule to map out the Pareto front between performance and bit-width with increasing β from 5×10^{-7} to 5×10^{-6}. Here, β is a hyperparameter that controls the trade-off between model performance and bit-width reduction during training, see [40] for details. We show one model obtained with this method in tables 2 and 3. Since HGQ quantizes the weights at a per-weight level and includes zero-bit (pruning), the final model is simultaneously unstructured pruned and quantized. The distribution of bit-widths for the weights of the model is shown in figure 2.

We use the da4ml [27] library to convert the model to RTL design and optimize the constant-matrix-vector multiplication operations (i.e. the dense layers of the MLP) using distributed arithmetic for lower resource usage and latency. Specifically, we convert directly to Verilog, and synthesize with Vivado 2025.1 with a target clock period of 5 ns.

We list in table 2 the physics performance of this approach, while in table 3 we report the usage of resources. We again observe a good retention of performance and a marked improvement in resource usage over the PTQ model, with a latency of just 35 ns. All results reported are from RTL behavioral simulation of the RTL designs via Verilator [41], and all resource usages are from Vivado post place and route reports.

6.3. Norm

The calculation of the input norm as the anomaly score $||\mathbf{x}||^2$ is not quantized as a standalone component. However, the norm operation on the output of the NN (same dimensionality as the input data) is synthesized as the final step of the complete neural network pipeline. From the PTQ synthesis estimates we can thus estimate that the norm operation represents a non-negligible fraction of the hardware footprint: the norm computation alone accounts for approximately 45% of the total flip-flops (FFs) and 20% of the look-up tables (LUTs) required by the full algorithm. This means that, for our specific choice of training target, we may obtain an even more performant AD score by simply synthesizing the norm operation.

7. Conclusions

We have presented the first application of a CNF for unsupervised AD within the demanding real-time environment of an LHC L1T system. The central innovation of this work is an FPGA-friendly anomaly score, $\mathcal{AS} = \|\mathbf{v}_t\|^2$, derived directly from the model's vector field output, which circumvents the need for computationally expensive ODE integration. We demonstrated that this approach successfully identifies a variety of new physics signatures, achieving performance comparable to a VAE trained under identical conditions. Furthermore, we have shown used advanced quantization techniques to prove that the synthesized algorithm meets the stringent L1T constraints, with a latency ranging from 230 to 35 ns and minimal resource utilization.

Our results demonstrate that the Flow architecture is capable of identifying simple, robust features, such as the L2 norm of the input, directly from the data distribution. This may offer a distinct advantage over VAEs: the Flow's vector field provides an interpretable, transparent anomaly score that is robust against training instabilities often found in VAE latent spaces, while the FPGA implementation remains flexible enough to learn more complex topologies if required.

This work opens several avenues for future investigation:

- Enhancing the discriminating power of the vector field by incorporating physics-motivated or agnostic constraints into the flow matching loss function (equation (3.2)).
- Exploring alternative anomaly scores derived from the vector field, such as its divergence, and extending the methodology to other ODE-based generative models like diffusion models.
- Performing a comprehensive scan over model architectures and flow probability paths, to further optimize performance and resource efficiency.
- Investigating model pruning and knowledge distillation to potentially simplify the model for inference while preserving its performance.
- Try other architectures for implementing the flow matching schema, such as boosted decision trees as done in [42].

We see NFs as not just a useful tool for offline analysis, but also a practical one for deployment in real-time discovery. This study paves the way for integrating this new class of powerful, model-agnostic algorithms into the next generation of trigger systems, expanding the potential for model-agnostic searches at 40 MHz.

Acknowledgments

We thank Dr Prasanth Shyamsundar for his valuable insight regarding the interpretation of the vector field at the $t = 1$ time slice.

Francesco Vaselli acknowledges the support and guidance received to kick-off this work in the context of the first CERN Next Generation Triggers hackathon, 7–11 April 2025, CERN Geneva.

The work done by Dimitrios Danopoulos, Roope Oskari Niemi and Felice Pantaleo is supported by the Eric & Wendy Schmidt Fund for Strategic Innovation through the CERN Next Generation Triggers project (Grant Agreement SIF- 2023-004).

Chang Sun is partially supported by the U.S. Department of Energy (DOE), Office of Science, Office of High Energy Physics Grant DE-SC0011925, NSF ACCESS Grant PHY240298.

This work was supported by the Open Access Publishing Fund of the Scuola Normale Superiore.

Data availability statement

The code used for this work is made publicly available under MIT License at the following URL: https://github.com/francesco-vaselli/fAD.

The datasets used are hosted on Zenodo as explained in section 2. The data that support the findings of this study are openly available at the following URL/DOI:. https://doi.org/10.5281/zenodo.5046389 [43].

ORCID iDs

Francesco Vaselli 0009-0008-8227-0755
Chang Sun 0000-0003-2774-175X
Thea Aarrestad 0000-0002-7671-243X
Dimitrios Danopoulos 0000-0001-9327-5983
Roope Oskari Niemi 0009-0006-1753-1248
Maciej Mikolaj Glowacki 0009-0005-7170-7024
Katya Govorkova 0000-0003-1920-6618
Vladimir Loncar 0000-0003-3651-0232
Felice Pantaleo 0000-0003-3266-4357
Maurizio Pierini 0000-0003-1939-4268

References

[1] Evans L and Bryant P 2008 Lhc machine *J. Instrum.* **3** S08001
[2] The ATLAS Collaboration 2008 The atlas experiment at the CERN Large Hadron Collider *J. Instrum.* **3** S08003
[3] The CMS Collaboration 2008 The CMS experiment at the CERN LHC *J. Instrum.* **3** S08004
[4] Evans L and Bryant P 2020 Performance of the CMS Level-1 Trigger in proton-proton collisions at sqrt(s) = 13 TeV *J. Instrum.* **15** 10017
[5] CMS 2020 The phase-2 upgrade of the CMS Level-1 Trigger *Technical Report* (CERN) (available at: https://cds.cern.ch/record/2714892)
[6] collaboration T A 2020 Operation of the atlas trigger system in run 2 *J. Instrum.* **15** 10004
[7] ATLAS 2017 Technical design report for the phase-II upgrade of the ATLAS TDAQ system *Technical Report* (CERN) (available at: https://cds.cern.ch/record/2285584)
[8] The ATLAS Collaboration 2012 Observation of a new particle in the search for the standard model Higgs boson with the atlas detector at the lhc *Phys. Lett. B* **716** 1–29
[9] The CMS Collaboration 2012 Observation of a new boson at a mass of 125 GeV with the CMS experiment at the LHC *Phys. Lett. B* **716** 30–61
[10] Aarrestad T *et al* 2022 The dark machines anomaly score challenge: benchmark data and model independent event classification for the large hadron collider *SciPost Phys.* **12** 043
[11] Kasieczka G *et al* 2021 The LHC Olympics 2020 a community challenge for anomaly detection in high energy physics *Rep. Prog. Phys.* **84** 124201
[12] Kingma D P and Welling M 2022 Auto-encoding variational bayes (arXiv:1312.6114)
[13] Cerri O, Nguyen T Q, Pierini M, Spiropulu M and Vlimant J-R 2019 Variational autoencoders for new physics mining at the Large Hadron Collider *J. High Energy Phys.* 1–29
[14] Knapp O, Dissertori G, Cerri O, Nguyen T Q, Vlimant J-R and Pierini M 2020 Adversarially learned anomaly detection on CMS open data: re-discovering the top quark (arXiv:2005.01598)
[15] Poppi F 2010 Is the bell ringing? p 14 46/2010 (available at: https://cds.cern.ch/record/1306501)
[16] Govorkova E *et al* 2022 Autoencoders on field-programmable gate arrays for real-time, unsupervised new physics detection at 40 MHz at the Large Hadron Collider *Nat. Mach. Intell.* **4** 154–61
[17] Gandrakota A 2024 Real-time anomaly detection at the L1 trigger of CMS experiment (arXiv:2411.19506)
[18] Hinton G, Vinyals O and Dean J 2015 Distilling the knowledge in a neural network (arXiv:1503.02531)
[19] Cohen M M, Addepalli S, Gonski J L, Gugel R, Jia K, Rankin D S, Shahinian J, Suarez P M and Sugizaki K 2025 GELATO: a generic event-level anomalous trigger option for ATLAS - slides (available at: https://cds.cern.ch/record/2938881)
[20] Papamakarios G, Nalisnick E, Rezende D J, Mohamed S and Lakshminarayanan B 2021 Normalizing flows for probabilistic modeling and inference (arXiv:1912.02762)
[21] Krause C, Nachman B, Pang I, Shih D and Zhu Y 2024 Anomaly detection with flow-based fast calorimeter simulators *Phys. Rev. D* **110** 035036
[22] Jawahar P, Aarrestad T, Chernyavskaya N, Pierini M, Wozniak K A, Ngadiuba J, Duarte J and Tsan S 2022 Improving variational autoencoders for new physics detection at the LHC with normalizing flows *Front. Big Data* **5** 803685
[23] FastML Team 2023 fastmachinelearning/hls4ml (available at: https://github.com/fastmachinelearning/hls4ml)
[24] Duarte J *et al* 2018 Fast inference of deep neural networks in FPGAs for particle physics *JINST* **13** 07027
[25] Aarrestad T *et al* 2021 Fast convolutional neural networks on FPGAs with hls4ml *Mach. Learn. Sci. Tech.* **2** 045015
[26] Ngadiuba J *et al* 2021 Compressing deep neural networks on FPGAs to binary and ternary precision with HLS4ML *Mach. Learn. Sci. Tech.* **2** 015001
[27] Sun C, Que Z, Loncar V, Luk W and Spiropulu M 2025 da4ml: distributed arithmetic for real-time neural networks on fpgas (arXiv:2507.04535)
[28] Coelho C N, Kuusela A, Li S, Zhuang H, Ngadiuba J, Aarrestad T K, Loncar V, Pierini M, Pol A A and Summers S 2021 Automatic heterogeneous quantization of deep neural networks for low-latency inference on the edge for particle detectors *Nat. Mach. Intell.* **3** 675–86
[29] Aarrestad T, Govorkova E, Ngadiuba J, Puljak E, Pierini M and Wozniak K A 2021 Unsupervised new physics detection at 40 MHz: a -textgreater4 leptons signal benchmark dataset Zenodo https://doi.org/10.5281/zenodo.5046446

[30] Aarrestad T, Govorkova E, Ngadiuba J, Puljak E, Pierini M and Wozniak K A 2021 Unsupervised new physics detection at 40 mhz: Lq -textgreater b tau signal benchmark dataset Zenodo https://doi.org/10.5281/zenodo.5055454
[31] Aarrestad T, Govorkova E, Ngadiuba J, Puljak E, Pierini M and Wozniak K A 2021 Unsupervised new physics detection at 40 MHz: h0 -textgreater tau tau signal benchmark dataset zenodo https://doi.org/10.5281/zenodo.5061633
[32] Aarrestad T, Govorkova E, Ngadiuba J, Puljak E, Pierini M and Wozniak K A 2021 Unsupervised new physics detection at 40 mhz: h+-textgreater tau nu signal benchmark dataset zenodo https://doi.org/10.5281/zenodo.5061688
[33] Govorkova E, Puljak E, Aarrestad T, Pierini M, Woźniak K A and Ngadiuba J 2021 Lhc physics dataset for unsupervised new physics detection at 40 MHz (arXiv:2107.02157)
[34] Lipman Y, Chen R T Q, Ben-Hamu H, Nickel M and Le M 2023 Flow matching for generative modeling (arXiv:2210.02747)
[35] Lipman Y, Havasi M, Holderrieth P, Shaul N, Le M, Karrer B, Chen R T Q, Lopez-Paz D, Ben-Hamu H and Gat I 2024 Flow matching guide and code (arXiv:2412.06264)
[36] Paszke A *et al* 2019 Pytorch: an imperative style, high-performance deep learning library (arXiv:1912.01703)
[37] Govorkova E *et al* 2021 fastml ae_l1_paper (available at: https://github.com/fastmachinelearning/hls4ml/tree/AE_L1_paper)
[38] Govorkova E *et al* 2025 cl-orca vae ad retraining (available at: https://github.com/katyagovorkova/cl-orca/tree/main/paper/vae)
[39] Ho J, Jain A and Abbeel P 2020 Denoising diffusion probabilistic models (arXiv:2006.11239)
[40] Chang S, Årrestad T, Lončar V, Ngadiuba J and Spiropulu M 2024 Gradient-based automatic per-weight mixed precision quantization for neural networks on-chip (available at: https://authors.library.caltech.edu/doi/10.7907/hq8jd-rhg30)
[41] Snyder W *et al* Verilator if you use this software, please cite it using the metadata from this file (available at: https://verilator.org)
[42] Jiang C, Qian S and Qu H 2024 Buff: boosted decision tree based ultra-fast flow matching (arXiv:2404.18219)
[43] Aarrestad T, Govorkova E, Ngadiuba J, Puljak E, Pierini M and Wozniak K 2021 Unsupervised New Physics detection at 40 MHz: Training Dataset *Zenodo* https://doi.org/10.5281/zenodo.5046389