UC Berkeley

Title

Flow-enhanced transportation for anomaly detection

Permalink

Journal

ISSN

Authors

Golling, Tobias

Klein, Samuel

Mastandrea, Radha

et al.

Publication Date

DOI

Copyright Information

Peer reviewed

Flow-enhanced transportation for anomaly detection

Tobias Golling[1,*] Samuel Klein[1,†] Radha Mastandrea[2,3,‡] and Benjamin Nachman[3,4,§]

[1]*Département de Physique Nucléaire et Corpusculaire, Université de Genève,*
24, quai Ernest-Ansermet, CH-1205 Genève 4, Switzerland
[2]*Department of Physics, University of California, Berkeley, California 94720, USA*
[3]*Physics Division, Lawrence Berkeley National Laboratory, Berkeley, California 94720, USA*
[4]*Berkeley Institute for Data Science, University of California, Berkeley, California 94720, USA*

Resonant anomaly detection is a promising framework for model-independent searches for new particles. Weakly supervised resonant anomaly detection methods compare data with a potential signal against a template of the Standard Model (SM) background inferred from sideband regions. We propose a means to generate this background template that uses a flow-based model to create a mapping between high-fidelity SM simulations and the data. The flow is trained in sideband regions with the signal region blinded, and the flow is conditioned on the resonant feature (mass) such that it can be interpolated into the signal region. To illustrate this approach, we use simulated collisions from the Large Hadron Collider (LHC) Olympics dataset. We find that our flow-constructed background method has competitive sensitivity with other recent proposals and can therefore provide complementary information to improve future searches.

I. INTRODUCTION

While the Large Hadron Collider (LHC) has been operational for over a decade, no targeted search for physics beyond the Standard Model has found significant evidence for new particles or forces of nature. As a result, there has recently been a growing interest in *model-agnostic* searches: in such studies, the goal is to find anomalies while making few assumptions about the underlying physics model that could have produced them. This mode of analysis is highly general, and it has gained momentum from recent advances for modeling, classifying, and finding anomalies in data. For a broad review of the role of modern machine learning (ML) methods in searches for new physics, see [1]; for performance summaries of ML-inspired methods on LHC-like data, see the LHC Olympics [2] and Dark Machines Anomaly Score Challenge [3] reports.

One class of anomaly detection searches is for *resonant* anomalies. In this case, we assume that the beyond-the-Standard Model (BSM) physics signature would be resonant

in at least one feature, typically some sort of mass. Such a signal might correspond to the production of a new on-shell particle. These "bump hunts" search for an excess in the mass feature above the known Standard Model (SM) background in a well-defined signal region. However, such searches require the existence of accurate SM background templates within the signal region. While particle generators for SM physics exist, the resulting simulated data (even with detector effects added) is not accurate enough to be used as a background for bump hunts in hadronic final states: hard process generators carry out perturbative calculations only to NLO, showering simulators make use of heuristic nonperturbative models, and detector simulators make necessary simplifying assumptions about particle-detector interactions in order to speed up runtime.

In response to this problem of subaccurate particle generators, a number of studies have focused on constructing SM background templates for LHC-like detected data within a signal region. Broadly, methods can be categorized on two qualities:

(1) *Simulation-assisted vs data-driven.* For simulation assisted methods, the background template construction is informed by a set of simulated LHC-like collider data representing SM processes; for data-driven methods, data from sidebands mass regions is used, where the sidebands are far enough from the resonant process such that the data can be treated as a proxy for SM background.

(2) *Likelihood learning vs feature morphing.* For the former, methods learn the likelihood of a SM-only

[*]tobias.golling@unige.ch
[†]samuel.klein@unige.ch
[‡]rmastand@berkeley.edu
[§]bpnachman@lbl.gov

TABLE I. Broadly speaking, many methods for constructing SM background templates for resonant anomaly detection can be classified on two axes. In this study, we introduce the FETA method.

	Simulation-assisted	Data-driven
Likelihood learning	SALAD [8]	Overdensity searches [9], ANODE [10], (LA)CATHODE [11,12]
Feature morphing	FETA (this work)	CURTAINS [13]

dataset (such as simulation, or detected data in sidebands) where there are no signal events. This learned likelihood is then interpolated into the signal region to act as a SM background template above which signal events might be detected. Alternatively, methods can physically morph features from the SM-only dataset to the detected, signal-containing dataset.

A number of previous methods for SM background construction are classified visually in Table I. This two-axis scheme is not meant to be all-encompassing, as there exist many methods for anomaly detection that cannot be so neatly classified (see [4–7] for examples of such methods in practice).

In this paper, we propose the "flow-enhanced transportation for anomaly detection" (FETA) method to model SM-like background at the LHC. FETA is simulation-assisted and relies on feature morphing, therefore filling the previously empty section of Table I.

In FETA, we train a normalizing flow[1] [15] to learn a mapping between simulation and data in sidebands regions. We then apply the learned mapping to signal region simulation to create a simulation-informed template for signal region SM background. FETA benefits from being simulation-assisted since it can use simulated SM data as a physically-informed prior for the background template; the method further benefits from using feature morphing since it is robust to mapping between feature spaces with non-overlapping support.

The structure of this paper is as follows. In Sec. II, we provide a concise background of normalizing flows and outline how they will be applied to physics-specific datasets. In Sec. III, we illustrate the effectiveness of flow-based models for creating context-dependent mappings with a toy example of triangular datasets. In Sec. IV, we exchange the toy models for LHC-like data and use FETA to create a model for LHC-like detected SM data. In Sec. V, we test the performance of FETA in a series of realistic anomaly detection tasks. In Sec. VI, we conclude and suggest avenues for further study.

---

[1]A comprehensive review of flow-based models is given in [14].

II. METHODOLOGY

A. Normalizing flows as morphing functions

Normalizing flows are constructed from invertible neural networks between sets of variables sampled from different probability densities. Given a random variable X sampled from a *reference distribution* p_R, one can define a transformation T that produces another random variable Z, i.e., $Z = T(X)$. The density of Z is then given by $p_Z(Z) = p_R(X)|\det \frac{\partial T}{\partial X}|^{-1}$. By chaining together a number of transformations T_i, one can produce an arbitrarily complex mapping between the initial reference density p_R and a *target distribution* p_T. Typically, the target density is taken to be a standard normal distribution. (Hence the name "normalizing" flow.)

In this work, we use the normalizing flow both for its density estimation power and its ability to construct morphing functions between nontrivial reference $X_R \sim p_R$ and target $X_T \sim p_T$ distributions. Our reference density is derived from Standard Model simulation ($X_{\rm SIM} \sim p_{\rm SIM}$), and our target dataset is derived from detected data ($X_{\rm DAT} \sim p_{\rm DAT}$).

More explicitly, we define a set of N event observables such that events in the reference and target are N-dimensional vectors $X_{\rm SIM}$ and $X_{\rm DAT}$, which respectively are sampled from the N-dimensional feature densities $p_{\rm SIM}$ and $p_{\rm DAT}$. We then train a flow to learn the mapping T between $p_{\rm SIM}$ and $p_{\rm DAT}$. In training the flow, we must ensure that the learned mapping is between simulated Standard-Model background and LHC-detected *background*. For resonant anomalies, we assume that the signal will be localized in one feature $m_{\rm res}$. This allows us to define sidebands (SB) and a signal region (SR) in $m_{\rm res}$, where data from the former regions is assumed to follow the SM distribution. We then train the flow only on data from SB, using the resonant feature to *condition* the mapping $T(\cdot|m_{\rm res})$. The learned flow is then applied to simulation in the SR to produce an approximation of $X_{\rm DAT}$: $X_{\rm SIM}^* = T(X_{\rm SIM}|m_{\rm res})$ for LHC-detected background in the SR. A schematic of this method is shown in Fig. 1.

There are several advantages to using a flow-based architecture over other architectures such as GANs or VAEs. Normalizing flows are known to be more stable and achieve convergence during training faster, especially in higher dimensions. This property allows for the freedom to choose a larger feature space X, which may be desirable for a model-agnostic study. For GANs specifically, attempting to learn conditional mappings between datasets is not an easily done task. In addition, the density estimating power of the normalizing flow allows us to *oversample* from the reference distribution and reduce statistical uncertainties (explored in more detail in Appendix A). This is not possible with VAEs, which require the definition of explicit pairs to train the encoder-decoder architecture.
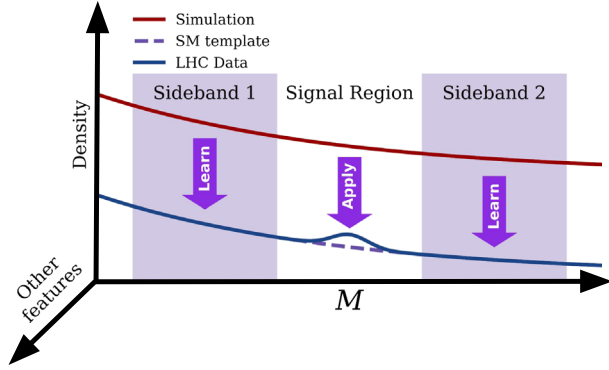
FIG. 1. A schematic of the FETA method. We train a flow to learn the mapping between simulation and data in sidebands regions, which are expected to be background-only. We then apply the learned flow to simulation in a signal region to produce an approximation for background in that region.

B. Flow construction

All flows were constructed using the nFlows package [16] and were trained using PyTorch [17].

As an important procedural note: to learn a flow that maps between two nontrivial densities, we use a two-step procedure. We first train a *base density* normalizing flow to learn the mapping between a normal distribution and the reference density p_R across all mass bands (i.e., using reference data from both the SB and SR). We then train a *transport* flow to learn the mapping between the *base density* distribution and the target density p_T, this time only using data from the SB. This specific method, which allows for the use of flows to map between nontrivial distributions, was proposed and implemented in the CURTAINS background construction method (and is further explored in [18]). It is thus more accurate to simply call FETA a flow-based method, rather than a normalizing flow-based one.

III. TESTING THE FETA METHOD WITH A TOY MODEL

To concretely illustrate the FETA method, we train a flow to map between two toy triangular datasets. Both datasets contain 100,000 samples in a two-dimensional feature space, with a feature X that is conditionally dependent on a feature M. This conditioning feature can be interpreted as a masslike feature, or one in which we expect an anomaly to be resonant.

A. Toy model datasets

To construct each dataset, we first generate samples of the conditioning (resonant) feature M, which we take to be uniformly distributed between (0, 1). On this feature, we define the signal region SR $\sim (0.34, 0.66)$. We also define two sidebands, SB1 $\sim (0.0, 0.34)$ on the low mass side of the SR, and SB2 $\sim (0.66, 1.0)$ on the high mass side.
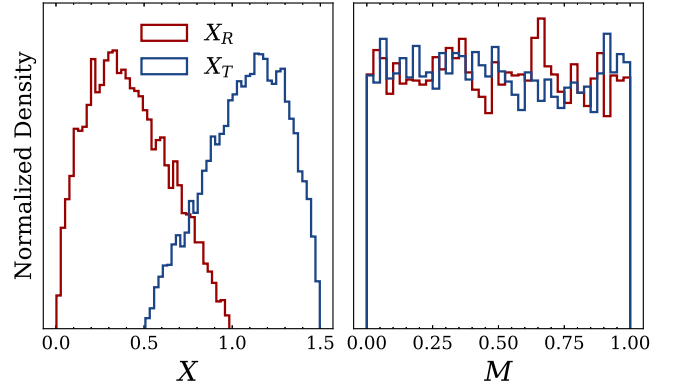

FIG. 2. Toy datasets X_R and X_T. The feature X is a function of the conditioning feature M.

For the corresponding nonresonant feature, we draw samples from a triangular dataset with endpoints at (0, 1). In order to condition this feature X on the mass, for each sample we define the midpoints m_R, m_T of the triangular distributions for reference and target to be linear functions of M, $m_R = 0.4995M$ and $m_T = 0.4995M + 0.5005$. Then the toy reference dataset X_R consists of left-triangular dataset samples in the range (0, 1), and the toy target dataset X_T consists of right-triangular samples. We further shift all the samples X_T of the target dataset by 0.5 such that the reference and the target sets have nonidentical support. Samples from the two datasets are shown in Fig. 2.

The flow architecture and hyperparameters are given in Table II. All settings were optimized via manual tuning and were chosen to give the best possible performance on the SB regions (as quantified by the ROC AUC, which is defined in Sec. III B). Flow training is optimized with AdamW [19], and the learning rate is annealed to zero following a cosine schedule [20]. Before training, all features are minmaxscaled to the range $(-3, 3)$, which was found to be optimal with respect to the flow training; further, the samples are split into training (80%) and

TABLE II. (RQ = rational quadratic; MADE = masked autoencoder for distribution estimation [21].) Flow architecture and training hyperparameters used for the toy (triangular) dataset. Both the *base density* and *transport* flow parameters were optimized through manual tuning.

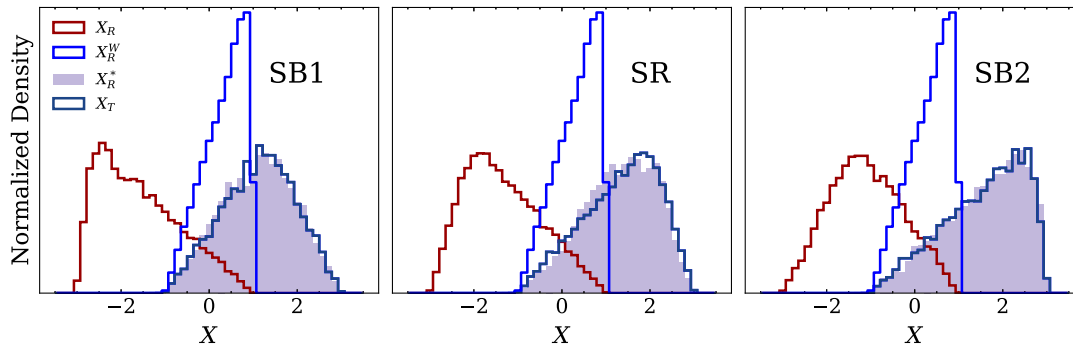| Parameter	*Base density* flow	*Transport* flow
Flow type	Autoregressive [22]	Coupling
Spline	Piecewise RQ	Piecewise RQ
Number MADE blocks	8	8
Number layers	2	2
Number hidden features	16	16
Epochs	20	20
Batch size	128	256
Learning rate	3×10^{-4}	3×10^{-4}
Weight decay	1×10^{-4}	1×10^{-5}

FIG. 3. Probability distributions for X_R, X_R^*, and X_T for the toy model. The X_R^* samples represent the trained flow acting upon X_R. The good agreement between X_R^* and X_T indicates that the flow has successfully learned to map between the two datasets. We also provide the distribution for reweighted reference X_R^W to illustrate its failure to provide an accurate model for X_T, as the reference and target datasets have different regions of support.

validation (20%) sets. The model from the epoch with the lowest validation loss is used for evaluation.

B. Toy model results

Plots of the reference X_R, transformed reference X_R^* as found by the FETA method, and target X_T distributions are shown in Fig. 3. We compare the distributions separately in SB1, SR, and SB2 (recall that the flow is trained only on data from SB1 and SB2 and is applied to the blinded SR data). Qualitatively, there is excellent agreement between the distributions for X_R^* and X_T in SB1 and SB2 and good agreement in the SR. We conclude that the flow has learned to map between the reference and the target data.

In Fig. 3, we also plot the reweighted reference X_R^W as found through the SALAD method. For this method, we train a binary classifier to discriminate between X_R and X_T. Such a classifier will learn the likelihood ratio between the reference and the target distributions. This likelihood ratio can then be interpolated into the SR and used to reweight X_R. Note that for this toy model, feature reweighting is expected to fail due to the fact that there are regions of nonoverlapping support between the reference and target.

We can quantify the performance of the learned morphing function by training a 5-fold binary classifier neural network to discriminate between X_R^* and X_T. Each classifier is a fully connected (dense) network, with linear layers of sizes (5, 64, 32, 1) and a dropout of 0.1 between each layer. Each classifier is trained for up to 100 epochs with a batch size of 128, learning rate of 10^{-3}, and patience of 5 epochs. We evaluate all test data on the classifier from the fold with the best (lowest) validation loss.

As our scoring metric, we use the area under the receiver operating characteristic (ROC) curve. These Area Under the Curve (AUC) scores for this classifier are shown in Table III. An ideal morphing function would result in the transformed reference and target datasets being indistinguishable from each other, which would correspond to a AUC close to 0.50. Indeed, the AUCs hit this performance

benchmark, and they are also far lower than those for a classifier trained to discriminate between untransformed X_R and X_T.

IV. APPLYING FETA TO LHC-LIKE DATA

We now move to a realistic example: training a flow to learn the morphing function between Standard Model simulation and detected data. For an ideal application of FETA, the reference dataset would consist of simulated Standard Model-like data ($X_R = X_{\text{SIM}}$), and the target dataset would consist of LHC-detected Standard Model data $X_T = X_{\text{DAT}}$). However, we do not have access to LHC data. Therefore in this study we use two distinct sets of simulated data for the reference and target datasets.

A. LHCO datasets

We focus on the LHC 2020 Olympics R&D dataset [2,23]. The full dataset consists of 1,000,000 background dijet events (Standard Model Quantum Chromodynamic (QCD) dijets) and 100,000 signal dijet events. The signal comes from the process $Z' \rightarrow X(\rightarrow q\bar{q})Y(\rightarrow q\bar{q})$, with three new resonances Z' (3.5 TeV), X (500 GeV), and Y (100 GeV). Events are required to have at least one large-radius jet ($R = 1$) trigger with a p_T threshold of 1.2 TeV. The events are generated with HERWIG++ [24], PYTHIA 8.219 [25,26], and

TABLE III. ROC AUCs for a binary classifier trained to discriminate the transformed reference X_R^* from the target dataset X_T. For comparison, we also provide the AUCs for a binary classifier trained to discriminate the untransformed reference X_R from the target. Uncertainties are the 1σ bounds from retraining the binary classifier 20 times, each with a different random seed.

Band	AUC for X_R^* vs X_T	AUC for X_R vs X_T
SB1	0.5144 ± 0.0095	0.9830 ± 0.0009
SR	0.5117 ± 0.0040	0.9806 ± 0.0000
SB2	0.5086 ± 0.0082	0.9813 ± 0.0009

TABLE IV. Band edge definitions in m_{JJ}-space for the LHC Olympics datasets and corresponding event counts.

Band	GeV Bounds	Herwig++ events	PYTHIA events
SB1	(2900, 3300)	210767	212115
SR	(3300, 3700)	121978	121339
SB2	(3700, 4100)	68609	66646
SB1 ∪ SB2		279376	278761

Delphes 3.4.1 [27]. Each event contains up to 700 particles with three degrees of freedom (DoF) p_T, η, ϕ.

For this study, we choose a feature space of six dijet observables m_{J_1}, Δm_{JJ}, $\tau_{J_1}^{21}$, $\tau_{J_2}^{21}$, ΔR_{JJ}, and m_{JJ}. Following the example of the CURTAINS analysis, we include the ΔR_{JJ} dijet observable as a feature that is highly correlated with m_{JJ}.

As with many anomaly detection searches, we assume that the anomaly is resonant in one feature, m_{JJ}. This would correspond to a new resonant particle that would produce the two jets. The assumption of resonance allows us to define a signal region (SR) in m_{JJ}-space, as well as two sidebands (SB1 and SB2) on the low- and high-mass ends of the SR. The region edges are defined in Table IV. While we have chosen to focus on one SR/SB setup, one could in practice perform a *sliding bump hunt* and scan the SR across a wide range of resonant masses.

The LHC Olympics dataset does not contain any LHC-detected data that would be the obvious choice for X_{SIM}. Therefore we take the LHC Olympics Herwig++ data as our simulation dataset X_{SIM} and the PYTHIA data as our target dataset X_{DAT}. Histograms of the six dijet observables for X_{SIM} and X_{DAT} are shown in Fig. 4.

B. Training method

The method for training a flow on the LHC data is the same as for the toy dataset. However, the flow architectures used for LHC-like data are significantly more expressive. Architectures and hyperparameters are outlined in Table V. Notably, the *base density* flow parameters were derived from the main architecture from CATHODE (which relies on faithful density estimation of detected collider data in SB),

TABLE V. Flow architecture and training hyperparameters used for SM background construction for the LHC Olympics dataset. The *base density* flow parameters were derived from the main architecture from CATHODE, but were confirmed to give the best performance through manual tuning. The *transport* flow parameters were optimized through manual tuning.

Parameter	*Base Density* flow	*Transport* flow
Flow type	Autoregressive	Coupling
Spline	Piecewise RQ	Piecewise RQ
Number MADE blocks	15	8
Number layers	1	2
Number hidden features	128	32
Epochs	100	50
Batch size	128	256
Learning rate	1×10^{-4}	5×10^{-4}
Weight decay	1×10^{-4}	1×10^{-5}

and they were confirmed to give the best performance through manual tuning. Data is minmaxscaled to the range $(-3, 3)$ before flow training, and a training-validation split of 80%–20% is used. All settings were manually optimized to give the best-performing flow possible, as quantified by the ROC AUCs in SB1 and SB2.

C. LHCO results

In Figs. 5(a) and 5(b), we plot the distributions for X_{SIM}, X_{SIM}^*, and X_{DAT} for the LHC-like data in SB1 and SB2. For each band, we also plot the ratio of untransformed and transformed simulation distributions to the target distribution. For all features, the transformed simulation X_{SIM}^* is visually much closer to the target X_{DAT} than the untransformed simulation.

In Fig. 5(c), we provide the same distributions for X_{SIM}, X_{SIM}^*, and X_{DAT} in SR. For these plots, we once again see good qualitative agreement between X_{SIM}^* and X_{DAT}, despite the fact that the flow was not explicitly trained to morph between SR datasets.

In Table VI, we quantify the performance of the flow through the ROC AUC of a binary classifier (with the same architecture as in Sec. III B) trained to discriminate X_{SIM}^*
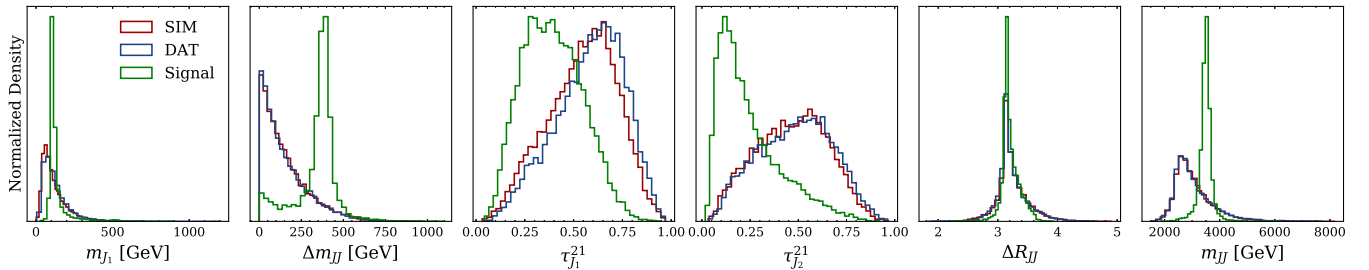

FIG. 4. Feature distributions for the six dijet observables used in the LHC Olympics analysis. SIM represents Herwig++ simulation, and DAT represents PYTHIA simulation. The last feature, m_{JJ}, is the feature in which we expect the anomaly to be resonant and conditions the flow mapping.
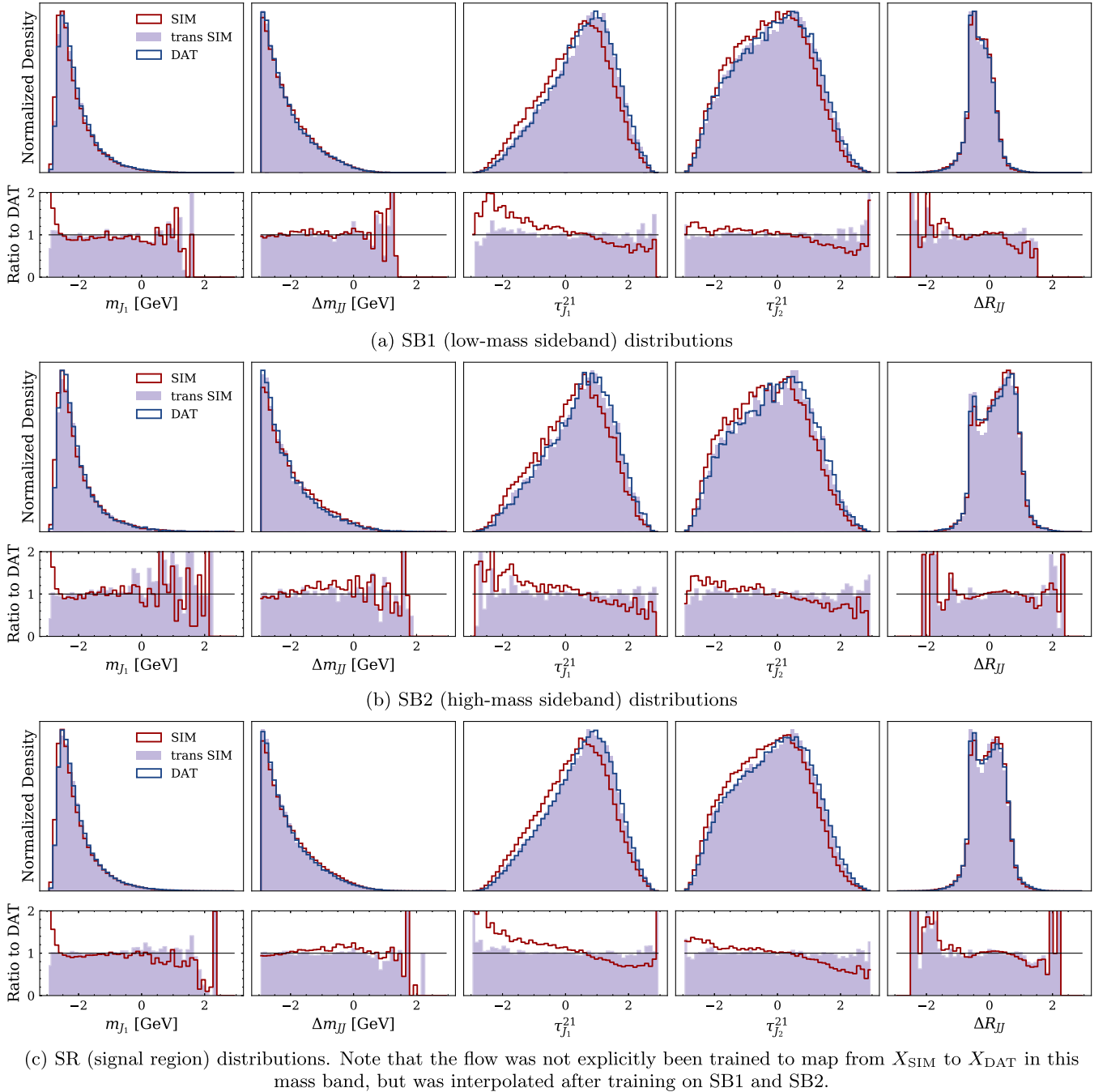
(a) SB1 (low-mass sideband) distributions

(b) SB2 (high-mass sideband) distributions

(c) SR (signal region) distributions. Note that the flow was not explicitly been trained to map from X_{SIM} to X_{DAT} in this mass band, but was interpolated after training on SB1 and SB2.

FIG. 5. Probability distributions and ratios for X_{SIM}, X^*_{SIM}, and X_{DAT} for the LHC Olympics model. The X^*_{SIM} samples represent the data that results from the trained flow acting upon X_{SIM}. The good agreement between X^*_{SIM} and X_{DAT} indicates that the flow has successfully learned to map between the two datasets.

from X_{DAT} in each band. In all bands, the AUC is consistent with below 0.51, so our benchmark for indistinguishability of transformed simulation from the target is achieved.

V. USING ꜰᴇᴛᴀ FOR ANOMALY DETECTION

For a well-trained flow, X^*_{SIM} should faithfully model the SM background as would be detected at the LHC.

However, if the LHC-detected data did contain some anomalous events, this would cause X^*_{SIM} to differ from X_{DAT}. In this case, an avenue for resonant anomaly detection emerges. We can train a binary classifier to discriminate X^*_{SIM} in the SR from data in the SR, and a significant deviation found between the distributions might provide evidence of anomalous events in the detected data. This method for anomaly detection relies on the fact that a

TABLE VI. ROC AUCs for a binary classifier trained to discriminate the transformed simulation X^*_{SIM} from the target dataset X_{DAT}. For comparison, we also provide the AUCs for a binary classifier trained to discriminate the untransformed simulation X_{SIM} from the target.

Band	AUC for X^*_{SIM} vs X_{DAT}	AUC for X_{SIM} vs X_{DAT}
SB1	0.5128 ± 0.0032	0.5970 ± 0.0022
SR	0.5034 ± 0.0010	0.5706 ± 0.0006
SB2	0.5153 ± 0.0054	0.5938 ± 0.0031

binary classifier trained to discriminate between two mixed samples (such as a background-only SM template and a detected mixture of SM background and resonant signal) is in fact the optimal classifier for distinguishing pure signal from pure background [28–30].

A. Signal injection procedure

To explore the capability of FETA for anomaly detection, we repeat the flow training method as outlined in Sec. IV B. We inject a known number of signal events into the X_{DAT} (PYTHIA) dataset. Since our chosen SR extends across the m_{JJ} range [3300, 3700] GeV, this allows for possible detection of the Z' resonance centered at 3500 GeV. We test a range of signal injections, scanning over $n_S \in$ [300, 500, 750, 1000, 1200, 1500, 2000, 2500, 3000] (corresponding to $S/B \approx [0.30\%, 0.49\%, 0.74\%, 0.99\%, 1.18\%, 1.48\%, 1.97\%, 2.47\%, 2.96\%]$). As a word of caution: despite the relatively wide SR window chosen, about 20% of the injected events go into the SB regions.

For each signal injection, we rerun the full FETA pipeline and train a flow to learn the mapping between simulation and data with the injected signal. (This retraining is necessary due to the signal contamination in the sidebands.) We then train a binary classifier (with the same architecture as in Sec. III B) to discriminate transformed simulation from data.

We compare the results of the FETA method with those from the CATHODE, CURTAINS, and SALAD methods. For each alternative method, we use the architecture cited in the respective paper.[2] To place all methods on an equal footing, we use the same set of training and validation dijet events for all methods. Such event breakdowns are given in Table VII. Note that all methods use the validation loss to select the best-performing model to be used for further analysis. The FETA, CATHODE, and CURTAINS methods all make use of *oversampling* of the SM background template to achieve better performance, which is a mechanism that allows for the reduction of statistical uncertainties.

---

[2]Further optimization certainly may yield even better result for the other methods considered, but this optimization task is nontrivial, especially to retain model-agnosticity.

We investigate the effects of oversampling on the FETA method performance in Appendix A.

All methods are evaluated by training a binary classifier to discriminate the SM background template samples from a set of 100 k dijet data events from the SR. All classifiers are tested on the same set of 20 k signal and 20 k background SR dijet events, which were not used at any point during the training or validation procedures.

B. Anomaly detection performance summary

In Fig. 6, we show a selection of summary plots from this final binary classifier corresponding to 2500 injected signal events ($S/B = 1.97\%$, $S/\sqrt{B} \approx 7.9$). Each curve represents the mean performance of 20 different random classifiers trained to discriminate the given SM background template from the "detected" events in the SR. Error bands represent the spread across the (16, 84) percentiles across these 20 runs. Note that the error bands are comparable for FETA, CATHODE, CURTAINS, and SALAD.

In Fig. 6(a), we provide the significance improvement characteristic curves, given by SIC $= \frac{\text{true positive rate}}{\sqrt{\text{false positive rate}}}$. The SIC can be interpreted in the limit of large S and B as the gain in signal significance (i.e., the multiplicative factor) over the initial significance that can be achieved by making a well-motivated cut on the dataset. Therefore for an optimally performing classifier, we expect to see the SIC $\gg 1$. In Fig. 6(b), we provide the classifier rejection curves, given by the reciprocal of the false positive rate. An optimally performing classifier should see this rejection also be $\gg 1$. Finally, in Fig. 6(c), we plot the SIC curves against the rejection curves.

In addition to the curves for the four methods considered, we also provide the curves corresponding to a fully supervised classifier, i.e., a classifier trained on perfectly labeled signal and background events. This curve demonstrates the maximum possible performance to discriminate SM physics from anomalous physics. Importantly, the fully supervised classifier should *not* be interpreted as the limiting case of an idealized anomaly detection study, which would come from a classifier trained to discriminate detected data from perfectly simulated background.

We find that the FETA method is competitive with the performance of a fully supervised classifier at signal efficiencies of around 0.3 and lower. This performance is encouraging as in practice, we would expect FETA to be used primarily in this range (for higher signal-efficiencies, the amount of background is sufficiently large that nonfully supervised methods are unlikely to effectively find the signal). Indeed, the performance curves of FETA for all three metrics, (SIC, rejection, and SIC vs rejection) very closely align with those of CATHODE, which was demonstrated to be state-of-the-art, and CURTAINS.

In Fig. 7, for each signal injection, we plot the maximum of the SIC, where the maximum is taken across all signal

TABLE VII. Number of dijet events used in training and validation, as well as number of events taken as the SR background template for each SM background template construction method. For the CURTAINS method, SR template samples are generated from transporting both the training and the validation SB samples into the SR. Note that oversampling (the reduction of statistical uncertainties by drawing more SM samples) is not possible with the SALAD method, which does not use a generative model for the reference dataset. The oversampling factor is the number of SM background template samples generated over the number of dijet events used to construct the SM background template samples.

Method	Train	Validation (Model selection)	Template samples (Evaluation)	Oversampling factor
FETA	224 k SB simulation 186 k SB data	56 k SB simulation 47 k SB data	720 k SR	×6
CATHODE	186 k SB data	47 k SB data	400 k SR	×3
CURTAINS	186 k SB data	47 k SB data	1120 k SR	×4
SALAD	224 k SB simulation 186 k SB data	56 k SB simulation 47 k SB data	120 k SR	Not applicable

efficiencies. Since the SIC relates to an increase in signal significance from a well-motivated cut, the max(SIC) is then the best possible cut for a given signal. Across the board, FETA, CATHODE, CURTAINS, and SALAD achieve similar performances, all of them becoming just about consistent with a fully supervised classifier at $\sim 1.97\%$ signal injection. The S/B corresponding to a minimum detectable signal (set to be ~ 3 for "observation" of BSM

(a) Significant improvement characteristic (SIC)

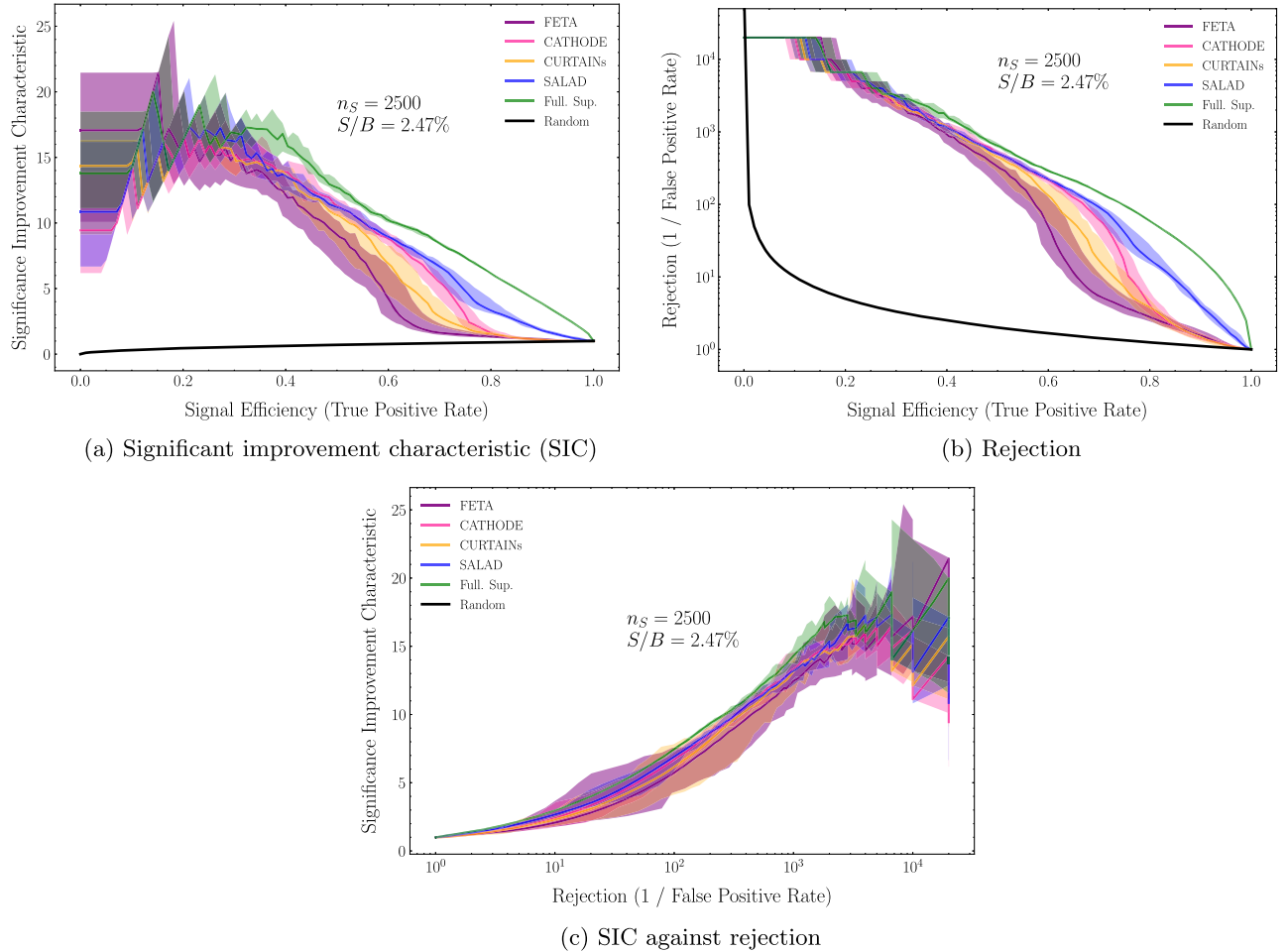(b) Rejection

(c) SIC against rejection

FIG. 6. Various performance metrics for a binary classifier trained to discriminate a constructed SM background template from detected SR data. We retrain the binary classifier 20 times, each with a different random seed. Curves illustrate the median of these classifier runs, and bands represent the spread across the (16, 84) percentiles. "Full. Sup." corresponds to a fully supervised classifier trained on pure signal and pure background.
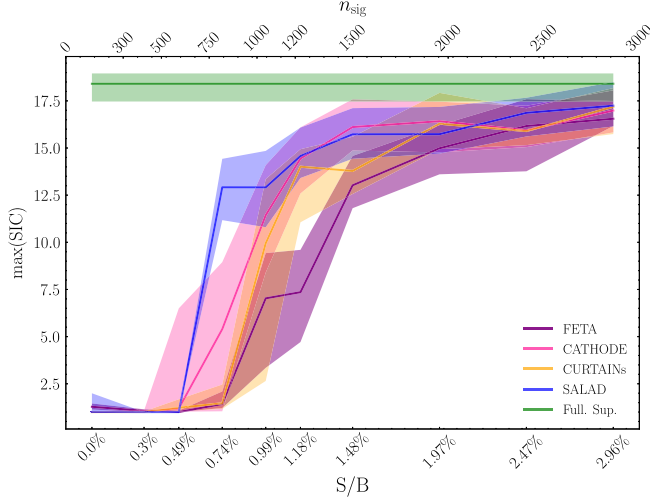
FIG. 7. Max significance improvement characteristic (SIC) as a function of the fraction of injected signal events.

physics) lies between 0.49% and 0.74%, as calculated by $\max(\text{SIC}) \times (S/\sqrt{B})$.

VI. CONCLUSIONS AND OUTLOOK

In this study, we have proposed a new method, FETA, for SM background construction that can be used for resonant anomaly detection. The method is simulation-assisted and relies on feature morphing (rather than reweighting) between a reference set of simulated SM data and detected data.

FETA can be seen as a hybrid of CURTAINs and SALAD: like CURTAINs, FETA uses a flow-based architecture that allows for feature morphing from a reference dataset to construct a SM template. This morphing property performs well in low-density regions of feature space where reweighting methods fail. Like SALAD, FETA uses simulated data as the reference dataset. This provides an advantage over data-driven methods as the reference dataset can act as a prior that is free of signal contamination.

For such simulation-assisted methods of background construction, optimality of transport is desirable as the simulated data provides a physics-informed prior that is expected to be close to the detected data. Therefore an efficient reweighting or morphing function should ideally do as little as possible and reduce to the identity when the simulation is exactly correct. One might ask if the *Transport* flow used in FETA executes the optimal transport between the reference and target datasets, especially as an out-of-the-box normalizing flow contains no loss terms that penalize non-optimal transport. In fact, for the scope of this problem (i.e. morphing between sets of Herwig++ and PYTHIA simulated LHC data), we found that modifications to the flow training method that enforce optimal transport did not significantly change the performance of FETA [31]. However, these modifications might become important if

FETA were applied to morph between a less similar reference and target.

Future avenues for exploration include monitoring the effects of the SB and SR widths on the performance of FETA adding rigorous uncertainty estimates for the performances of all four background construction methods considered in this study, and testing the four methods on a resonant anomaly other than the LHC Olympics one.

Code availability statement
The code can be found at [32].

ACKNOWLEDGMENTS

APPENDIX A: OVERSAMPLING

In this section, we investigate the effects of oversampling on the performance of the FETA background template. Oversampling was found to greatly improve the performance of the CATHODE and CURTAINS background templates, and similar improvements were found for FETA.

Oversampling refers to the reduction of statistical uncertainties by using a larger number of events. This method is only possible when the background template construction makes use of a density estimator for the reference dataset that can be sampled from multiple times. With FETA, the *base density* flow (defined in Sec. II B) plays this role: without oversampling, FETA transports from X_{SIM} to X_{DAT}; with oversampling, FETA draws samples X_{BD} from the *base density* and transforms these to X_{DAT}.

In Figs. 8(a) and 8(b), we plot the SIC and rejection (respectively) curves corresponding to oversampling factors from one (120 k SR events) through six (720 k SR events) for a signal injection of 1000 ($S/B = 0.99\%$). We find that the classifier performance generally increases to a peak at the oversampling factor of $\times 5$, after which performance saturates. However, for signal injections of

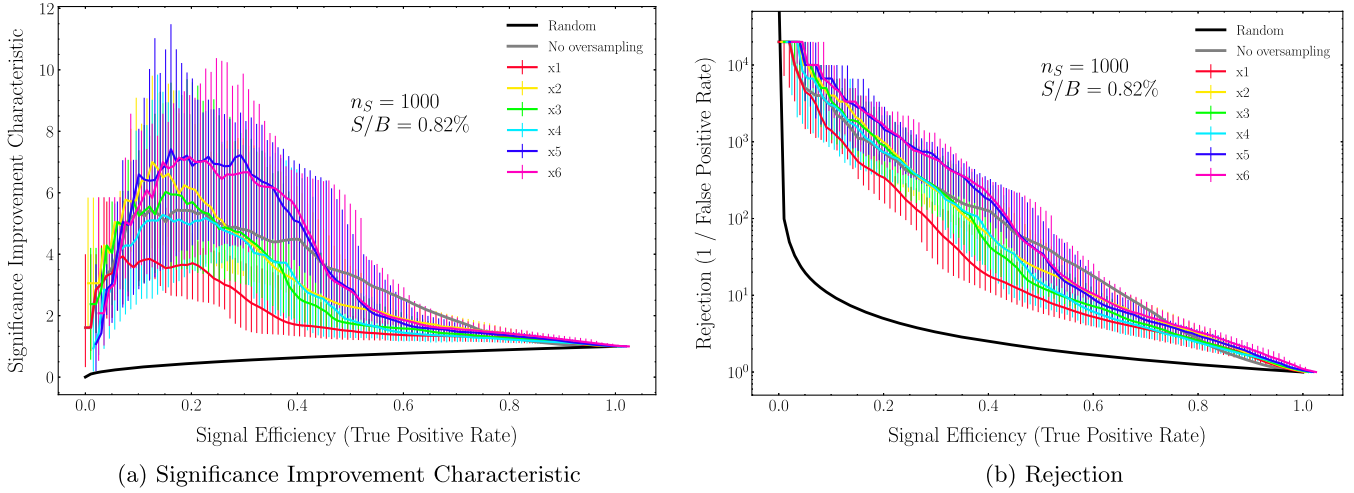(a) Significance Improvement Characteristic

(b) Rejection

FIG. 8. Performance metrics for a binary classifier trained to discriminate a constructed SM background template from detected SR data (with 1000 injected signal events) at various factors of oversampling. We retrain the binary classifier 20 times, each with a different random seed. Curves illustrate the median of these classifier runs, and bands represent the spread across the (16, 84) percentiles.

≥ 2000 ($S/B = 1.97\%$), all oversampling factors perform similarly.

APPENDIX B: SIGNAL VS BACKGROUND CORRELATION

In this section, we compare in more detail the events produced by the various SM background template construction methods (FETA, CATHODE, CURTAINS, and SALAD).

We expect all of the methods to produce background samples that are significantly different from anomalous events. This is what allows us to detect resonant anomalies when training a classifier to discriminate between SR background samples and SR detected data. However, it is not obvious (or expected) that the background samples will be similar between methods. In fact, we might expect classifier scores to be uncorrelated for background events.

To probe how different the phase spaces are between construction methods: we train a binary classifier (with the architecture as described in Sec. III B) to discriminate between SR background samples from SR detected data with 0 injected signal events. We evaluate the trained classifier on the test set of 20 k signal and 20 k background dijet events to get a score between 0 and 1 for each event. (Note that these scores are different than the AUC scores considered in the main analysis.) We finally plot these scores (standardized to zero mean and unit variance) for different methods as a function of each other, for FETA vs CATHODE in Fig. 9(a), for FETA vs CURTAINS in Fig. 9(b), and for FETA vs SALAD in Fig. 9(c).

For both such cases, we find that the scores assigned to background events derived from classifiers trained on SM background samples from FETA are broadly uncorrelated with those from classifiers trained on SM background samples from the other three methods. There is perhaps a mild degree of correlation between scores for signal events, but the correlation would not be expected to be strong as the classifiers saw no signal events during training.

We then repeat the analysis, this time training the classifiers to discriminate between SR background samples from SR detected data with 2500 injected signal events. We plot the same standardized scores for FETA vs CATHODE in Fig. 10(a), for FETA vs CURTAINS in Fig. 10(b), and for FETA vs SALAD in Fig. 10(c).

In these cases, there is a large degree of correlation between the classifier scores assigned to signal events between FETA and all of the other construction methods. This result indicates that all of the SM background construction methods are (roughly) equally sensitive to this particular LHCO anomaly.

(a) FETA vs. CATHODE.

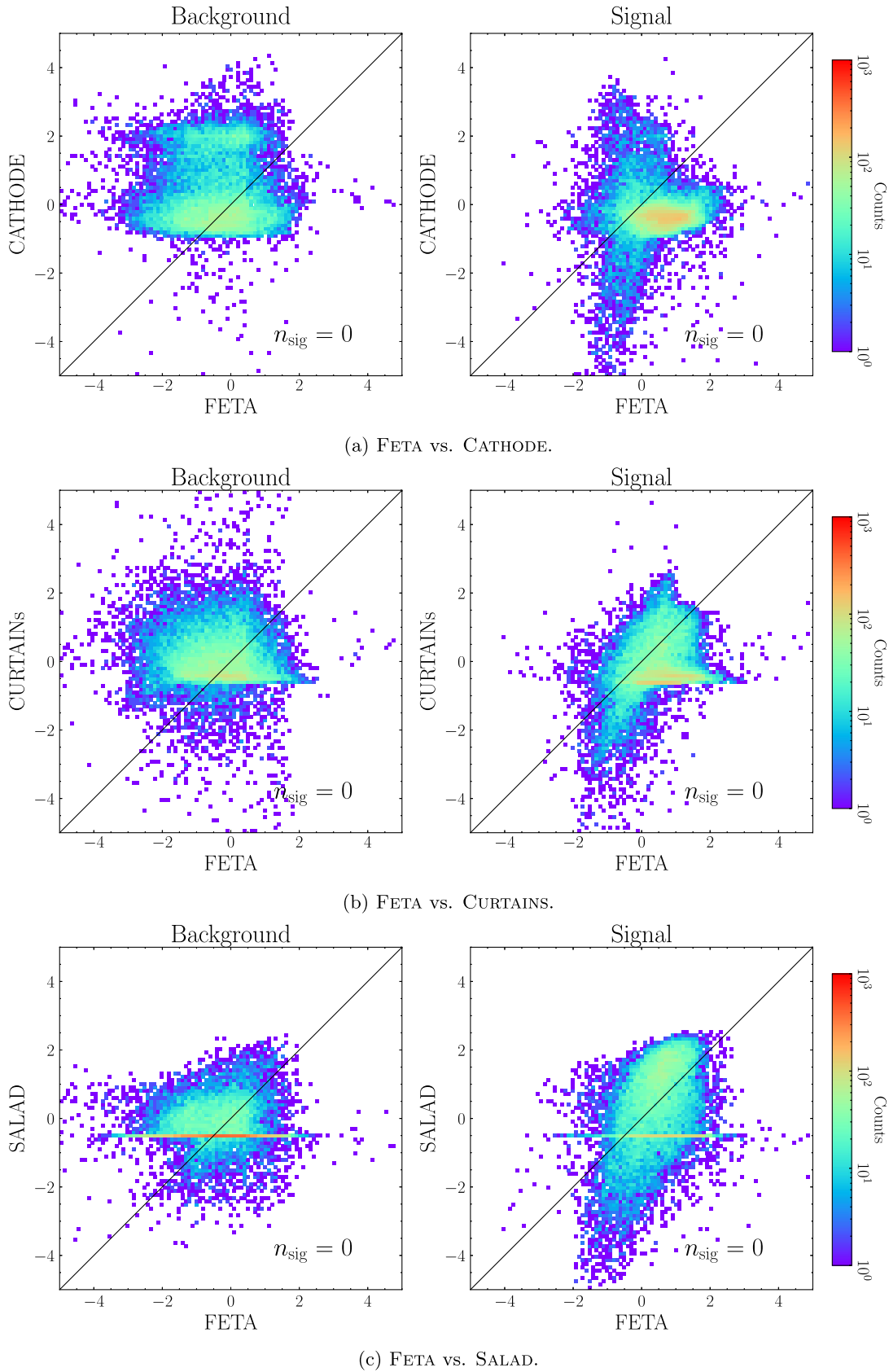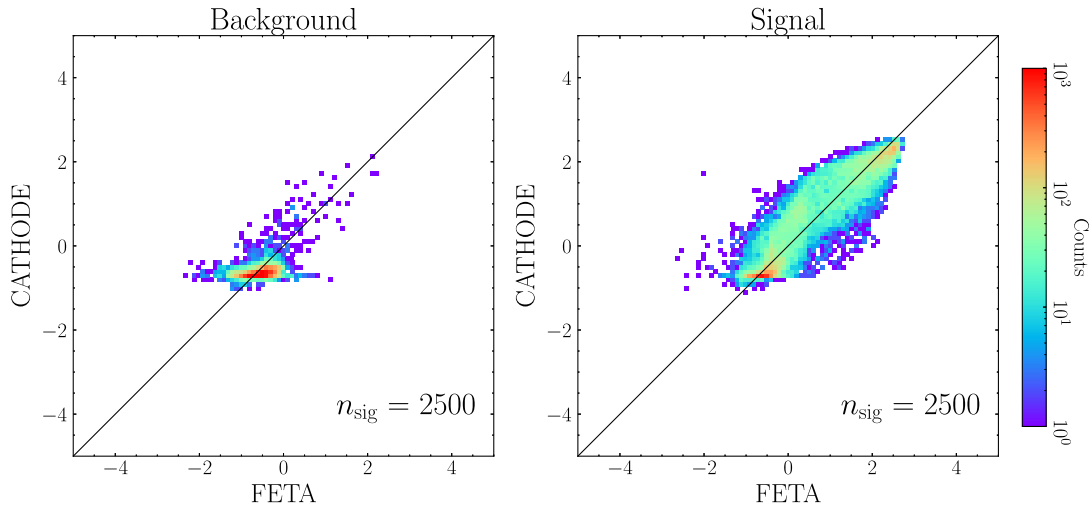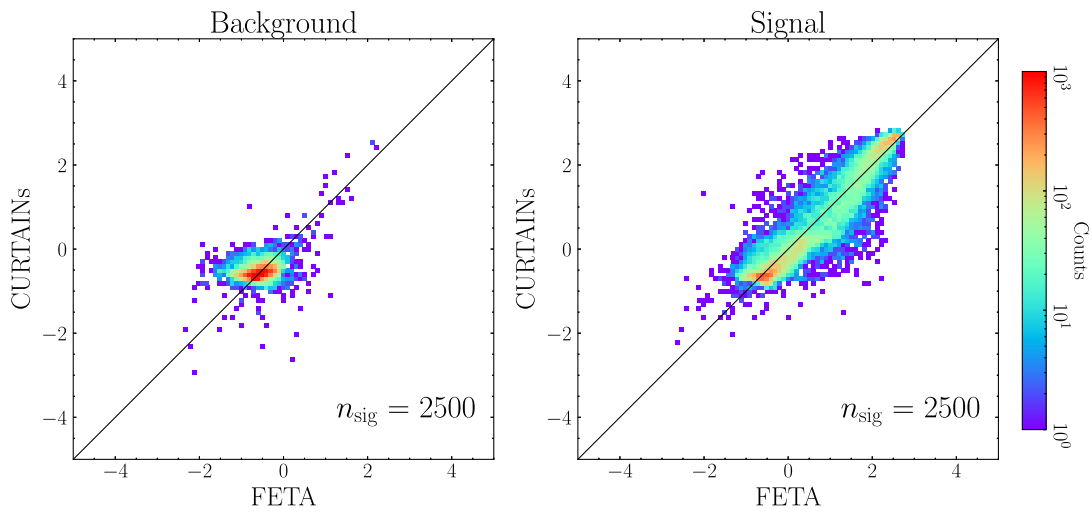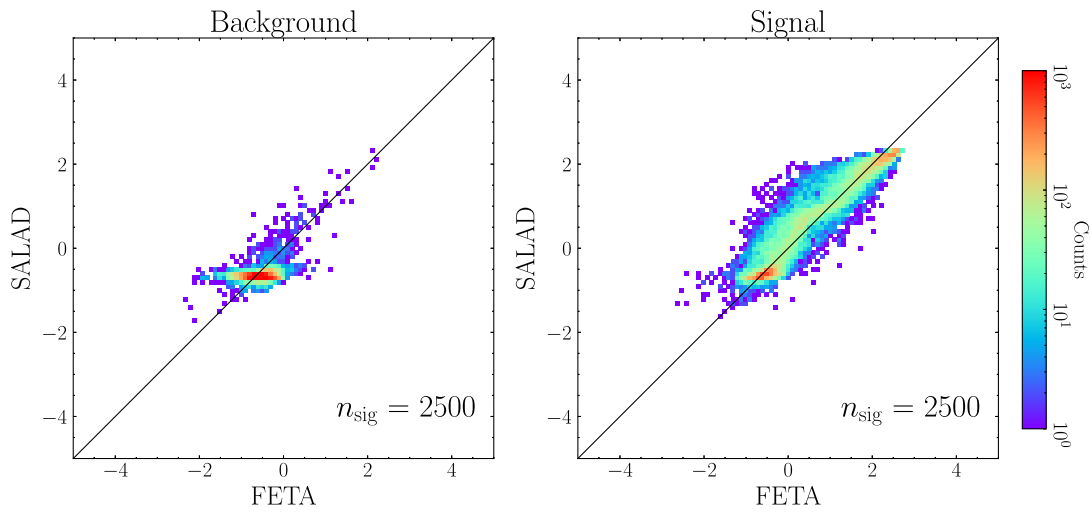(b) FETA vs. CURTAINS.

(c) FETA vs. SALAD.

FIG. 9.   Classifier scores for a binary classifier trained to discriminate a given constructed SM background template from detected SR data with no injected signal events and evaluated on pure signal and pure background. The scores have been normalized to have zero mean and unit variance.

(a) FETA vs. CATHODE.

(b) FETA vs. CURTAINS.

(c) FETA vs. SALAD.

FIG. 10.   Classifier scores for a binary classifier trained to discriminate a given constructed SM background template from detected SR data with 2500 injected signal events and evaluated on pure signal and pure background. The scores have been normalized to have zero mean and unit variance.

[1] Georgia Karagiorgi, Gregor Kasieczka, Scott Kravitz, Benjamin Nachman, and David Shih, Machine learning in the search for new fundamental physics, Nat. Rev. Phys. **4,** 399 (2022).

[2] Gregor Kasieczka *et al.*, The LHC Olympics 2020 a community challenge for anomaly detection in high energy physics, Rep. Prog. Phys. **84,** 124201 (2021).

[3] Thea Aarrestad *et al.*, The dark machines anomaly score challenge: Benchmark data and model independent event classification for the Large Hadron Collider, SciPost Phys. **12,** 043 (2022).

[4] Kees Benkendorfer, Luc Le Pottier, and Benjamin Nachman, Simulation-assisted decorrelation for resonant anomaly detection, Phys. Rev. D **104,** 035003 (2021).

[5] Oz Amram and Cristina Mantilla Suarez, Tag N' Train: A technique to train improved classifiers on unlabeled data, J. High Energy Phys. 01 (**2021**) 153.

[6] Suyong Choi, Jaehoon Lim, and Hayoung Oh, Data-driven estimation of background distribution through neural autoregressive flows, arXiv:2008.03636.

[7] CMS Collaboration, Evidence for the simultaneous production of four top quarks in proton-proton collisions at $\sqrt{s} = 13$ TeV, Report No. CMS-PAS-TOP-21-005, 2022.

[8] Anders Andreassen, Benjamin Nachman, and David Shih, Simulation assisted likelihood-free anomaly detection, Phys. Rev. D **101,** 095004 (2020).

[9] George Stein, Uros Seljak, and Biwei Dai, Unsupervised in-distribution anomaly detection of new physics through conditional density estimation, in *34th Conference on Neural Information Processing Systems* (2020), arXiv:2012.11638.

[10] Benjamin Nachman and David Shih, Anomaly detection with density estimation, Phys. Rev. D **101,** 075042 (2020).

[11] Anna Hallin, Joshua Isaacson, Gregor Kasieczka, Claudius Krause, Benjamin Nachman, Tobias Quadfasel, Matthias Schlaffer, David Shih, and Manuel Sommerhalder, Classifying anomalies through outer density estimation, Phys. Rev. D **106,** 055006 (2022).

[12] Anna Hallin, Gregor Kasieczka, Tobias Quadfasel, David Shih, and Manuel Sommerhalder, Resonant anomaly detection without background sculpting, arXiv:2210.14924.

[13] John Andrew Raine, Samuel Klein, Debajyoti Sengupta, and Tobias Golling, CURTAINS for your sliding window: Constructing unobserved regions by transforming adjacent intervals, Front. Big Data **6,** 899345 (2023).

[14] Ivan Kobyzev, Simon J. D. Prince, and Marcus A. Brubaker, Normalizing flows: An introduction and review of current methods, IEEE Trans. Pattern Anal. Mach. Intell. **43,** 3964 (2021).

[15] E. G. Tabak and Cristina V. Turner, A family of nonparametric density estimation algorithms, Commun. Pure Appl. Math. **66,** 145 (2013).

[16] Conor Durkan, Artur Bekasov, Iain Murray, and George Papamakarios, nFlows: Normalizing flows in PyTorch, 10.5281/zenodo.4296287 (2020).

[17] Adam Paszke *et al.*, PyTorch: An imperative style, high-performance deep learning library, in *Advances in Neural Information Processing Systems 32* (Curran Associates, Inc., Vancouver, Canada, 2019), pp. 8024–8035.

[18] Samuel Klein, John Andrew Raine, and Tobias Golling, Flows for flows: Training normalizing flows between arbitrary distributions with maximum likelihood estimation, arXiv:2211.02487.

[19] Diederik P. Kingma and Jimmy Ba, Adam: A method for stochastic optimization, arXiv:1412.6980.

[20] Ilya Loshchilov and Frank Hutter, SGDR: Stochastic gradient descent with restarts, arXiv:1608.03983.

[21] Mathieu Germain, Karol Gregor, Iain Murray, and Hugo Larochelle, Made: Masked autoencoder for distribution estimation, arXiv:1502.03509.

[22] Chin-Wei Huang, David Krueger, Alexandre Lacoste, and Aaron Courville, Neural autoregressive flows, arXiv:1804.00779.

[23] Gregor Kasieczka, Benjamin Nachman, and David Shih, Official Datasets for LHC Olympics 2020 Anomaly Detection Challenge (Version v6) [Data set]. 10.5281/zenodo.4536624 (2019).

[24] Manuel Bähr, Stefan Gieseke, Martyn A. Gigg, David Grellscheid, Keith Hamilton, Oluseyi Latunde-Dada, Simon Plätzer, Peter Richardson, Michael H. Seymour, Alexander Sherstnev, and Bryan R. Webber, Herwig++ physics and manual, Eur. Phys. J. C **58,** 639 (2008).

[25] Torbjorn Sjostrand, Stephen Mrenna, and Peter Z. Skands, PYTHIA 6.4 physics and manual, J. High Energy Phys. 05 (2006) 026.

[26] Torbjörn Sjöstrand, Stefan Ask, Jesper R. Christiansen, Richard Corke, Nishita Desai, Philip Ilten, Stephen Mrenna, Stefan Prestel, Christine O. Rasmussen, and Peter Z. Skands, An introduction to PYTHIA 8.2, Comput. Phys. Commun. **191,** 159 (2015).

[27] J. de Favereau, C. Delaere, P. Demin, A. Giammanco, V. Lemaître, A. Mertens, and M. Selvaggi (DELPHES 3 Collaboration), Delphes 3, A modular framework for fast simulation of a generic collider experiment, J. High Energy Phys. 02 (2014) 057.

[28] Eric M. Metodiev, Benjamin Nachman, and Jesse Thaler, Classification without labels: Learning from mixed samples in high energy physics, J. High Energy Phys. 10 (2017) 174.

[29] Jack H. Collins, Kiel Howe, and Benjamin Nachman, Extending the search for new resonances with machine learning, Phys. Rev. D **99,** 014038 (2019).

[30] Jack H. Collins, Kiel Howe, and Benjamin Nachman, Anomaly Detection for Resonant New Physics with Machine Learning, Phys. Rev. Lett. **121,** 241803 (2018).

[31] Radha Mastandrea and Benjamin Nachman, Efficiently moving instead of reweighting collider events with machine learning, in *36th Conference on Neural Information Processing Systems* (2022), arXiv:2212.06155.

[32] R. Mastandrea, rmastand/FETA: v1.0.0 (2023), 10.5281/zenodo.7927352.