

How to pick the best anomaly detector?

Marie Hein,^{1,*} Gregor Kasieczka,^{2,†} Michael Krämer,^{1,‡} Louis Moureaux,^{2,§} Alexander Mück,^{1,¶} and David Shih^{3,**}

¹*Institute for Theoretical Particle Physics and Cosmology,
RWTH Aachen University, D-52056 Aachen, Germany*

²*Institut für Experimentalphysik, Universität Hamburg, 22761 Hamburg, Germany*

³*NHETC, Dept. of Physics and Astronomy, Rutgers University, Piscataway, NJ 08854, USA*

Anomaly detection has the potential to discover new physics in unexplored regions of the data. However, choosing the best anomaly detector for a given data set in a model-agnostic way is an important challenge which has hitherto largely been neglected. In this paper, we introduce the data-driven ARGOS metric, which has a sound theoretical foundation and is empirically shown to robustly select the most sensitive anomaly detection model given the data. Focusing on weakly-supervised, classifier-based anomaly detection methods, we show that the ARGOS metric outperforms other model selection metrics previously used in the literature, in particular the binary cross-entropy loss. We explore several realistic applications, including hyperparameter tuning as well as architecture and feature selection, and in all cases we demonstrate that ARGOS is robust to the noisy conditions of anomaly detection.

I. INTRODUCTION

In recent years there has been an explosion of new methods for model-agnostic new physics searches at the LHC, leveraging powerful modern machine-learning (ML) anomaly detection methods such as autoencoders and weak supervision [1–4]. These methods are able to find anomalies in the data with varying degrees of model independence: either signal-model-independence, in that they make no use of any specific new physics models in their training, and/or background-model-independence in that they make no use of Standard Model (SM) simulations in their training. As such, these new model-agnostic search strategies offer significant new opportunities for discovery in the vast, unexplored phase space of the LHC data, beyond what model-specific search strategies have been able to achieve to date.

However, one important question has been left largely unaddressed to date: how does one pick the best anomaly detector from all the available options? In our case, we consider the “best” anomaly detector to be the *most sensitive to unknown signals*, as measured by the standard Significance Improvement Characteristic (SIC):

$$\text{SIC} = \frac{\epsilon_S}{\sqrt{\epsilon_B}}, \quad (1)$$

where $\epsilon_{S/B}$ is the signal/background efficiency after selecting the most anomalous events with the anomaly detector. The SIC is a multiplicative factor that quantifies the extent to which the anomaly detector enhances the significance of the new physics hiding in the data. Our

goal is to select an anomaly detector that can achieve a high SIC sensitivity to any signal that is actually present in the given dataset.

So far, when it comes to different approaches (e.g. autoencoders vs. weak supervision), the idea has been to just try them all, as seen in Ref. [5]. Within a given approach, there are myriad smaller but still vital choices to make: which model epoch to pick during training, which model architecture and hyperparameters to use, or even which features to use. Metrics traditionally used to guide these choices, such as the SIC and area under curve (AUC), are based on truth labels and require the use of benchmark signals. This is clearly not satisfactory for anomaly detection as the added signal dependence may, in the worst case, work against finding a signal actually present in the data.

Consequently, none of the existing experimental anomaly detection analyses [5–12] report systematic model optimization. Model architectures and training parameters are generally derived from the original publication introducing the corresponding method, with adjustments to network capacity to deal with the larger experimental datasets. In some cases [5, 8, 11], parameters have been retuned using a small set of benchmark signals. In Refs. [5, 11], anomaly scores have been selected from different classifier runs or epochs in certain approaches – CWoLa Hunting and TNT used the number of events selected in the signal region at a fixed background efficiency in the side bands as a metric; and CATHODE used the validation loss, following Ref. [13].

In this paper we introduce ARGOS (Above Random Gain Of SIC), a new *fully data-driven* metric for selecting the best anomaly detector. The key ingredient (besides the anomaly detector) that is required to calculate ARGOS is what we refer to as a background template (BT), which is a sample of events following the distribution of SM background in the signal region (SR). Then ARGOS is defined to be:

$$\text{ARGOS} = \frac{\epsilon_{\text{SR}}}{\sqrt{\epsilon_{\text{BT}}}} - \sqrt{\epsilon_{\text{BT}}}, \quad (2)$$

* marie.hein@rwth-aachen.de

† gregor.kasieczka@uni-hamburg.de

‡ mikraemer@physik.rwth-aachen.de

§ louis.moureaux@cern.ch

¶ mueck@physik.rwth-aachen.de

** shih@physics.rutgers.edu

where $\epsilon_{\text{SR/BT}}$ is the efficiency to select SR/BT events for a given anomaly threshold. In contrast to SIC, which cannot be calculated for real (unlabeled) data, ARGOS is accessible for any anomaly detector if there is a corresponding background template in addition to the (unlabeled) data set. In Section II, we show that ARGOS has a sound theoretical foundation, being monotonic with the actual signal-to-background SIC for any unknown signal at a fixed background efficiency when the background template is perfect. More generally, ARGOS is proportional to SIC to a very good approximation in any meaningful setting.

For concreteness, we demonstrate the utility of ARGOS within the context of weak supervision [14], where a classifier is trained to distinguish two mixed-label data sets, for resonant anomaly detection. However, nothing in principle limits ARGOS to this context. Using the LHC-O dataset [1], introduced in Section III, we demonstrate how ARGOS can consistently select the best hyperparameters, architecture (e.g. neural networks vs. different boosted decision trees) and model epoch in Section IV. As an outlook, we also briefly discuss the potential for ARGOS to select the best features in a fully data-driven way before we conclude in Section V. Technical aspects of our experiments are discussed in several Appendices.

II. THE ARGOS METRIC

In this Section, we show that ARGOS, as introduced in Sec. I, is the optimal data-driven metric for evaluating anomaly detectors when using an ideal BT. For this purpose, we rewrite the ARGOS metric as given in Eq. (2) in terms of the background and signal efficiencies using the probability density of the SR

$$p_{\text{SR}}(x) = f_S \cdot p_S(x) + (1 - f_S) \cdot p_B(x), \quad (3)$$

where f_S is the signal fraction in the signal region and $p_{S/B}(x)$ denote the probability density for the signal/background events. For an ideal BT, we have

$$p_{\text{BT}}^{\text{ideal}}(x) = p_B(x), \quad (4)$$

which we try to approximate in real-world applications as discussed in Sec. III. Using these densities, which imply $\epsilon_{\text{BT}} = \epsilon_B$, we find

$$\text{ARGOS} = \frac{f_S \cdot \epsilon_S + (1 - f_S) \cdot \epsilon_B}{\sqrt{\epsilon_B}} - \sqrt{\epsilon_B} \quad (5)$$

$$= f_S \cdot (\text{SIC} - \sqrt{\epsilon_B}), \quad (6)$$

Thus we see that ARGOS and SIC are closely related, at least in the idealized setting.

In more detail, there are multiple ways one can view this result:

- At fixed ϵ_B , ARGOS is monotonic with SIC, since the signal fraction f_S is a constant. Then the best

anomaly detector with respect to the ARGOS metric is also the best anomaly detector with respect to SIC. Additionally, ARGOS and the number of selected signal region events are also monotonic in this case and therefore also result in the same optimal decisions. The latter was used in Ref. [5, 11] to select epochs from multiple trainings of the CWoLa Hunting and TNT approaches.

- For ARGOS, however, there is no need to fix the background efficiency a priori. For a good anomaly detector, $\sqrt{\epsilon_B} \ll \text{SIC}$ also holds, such that the maximum value of ARGOS as a function of score threshold is also close to the maximum value of SIC. Then an added benefit of using the ARGOS metric in this way is that one can also select the optimal working point of the anomaly detector in a data-driven way.

In Section IV, we take the second approach and employ the maximum value of ARGOS for optimization. We will show that it retains good performance for realistic BTs.

III. SETUP

A. Data set

In Section IV, we perform experiments in a number of different scenarios, which test different metrics for optimizing the setup for anomaly detection. These experiments are performed using the LHC Olympics 2020 (LHC-O) [1] R&D data set [15], which is a common benchmark data set for LHC anomaly detection. It contains 10^6 QCD dijet background events and 10^5 signal events. The signal is a W' resonance at $m_{W'} = 3.5$ TeV with the decay $W' \rightarrow X(\rightarrow qq)Y(\rightarrow qq)$, where X and Y have masses $m_X = 100$ GeV and $m_Y = 500$ GeV, respectively. Additionally, we use 612858 SR background events from Ref. [16], which were generated for Ref. [13].

The simulation pipeline consists of **Pythia 8** [17, 18] and **Delphes 3.4.1** [19]. Using **Fastjet** [20], reconstructed particles are clustered with the anti- k_T algorithm [21] with a distance parameter of $R = 1$. All events must pass a leading-jet trigger using $p_T > 1.2$ TeV. For the classification, we use the features also used in Ref. [13]. We select the two highest p_T jets and use the mass of the lighter jet m_{J_1} , the mass differences of the two jets $\Delta m_J = m_{J_2} - m_{J_1}$ as well as both jets' 21-subjettiness ratios [22, 23]. In Sec. IV E, additional feature sets based on Ref. [24] are used. The dijet mass m_{JJ} is used as the resonant feature.

For the anomaly detection scenarios, we use all 1M background events and inject N_{sig} signal events. We use the same signal events in all runs in analogy to the fixed data set available in an analysis. We define our signal region using the dijet mass by requiring $3.3 \text{ TeV} \leq m_{JJ} \leq 3.7 \text{ TeV}$, i.e., a 0.4 TeV window around the $m_{W'}$ resonance, resulting in approximately 120 000

SR background events. The additional SR background events from Ref. [16] constitute the background template for the IAD (see Sec. III B 1), and also form part of the test set, for which a total of 340 000 SR background and 20 000 SR signal events are used.

B. Weakly Supervised Methods for Resonant Anomaly Detection

We focus on classifier-based resonant anomaly detection in this work, since these methods are predicated on the existence of an accurate BT that describes the smooth background in the SR. For resonant anomalies, the data-driven construction of the BT is usually based on adjacent sideband (SB) regions that facilitate smooth interpolation into the SR. There are a variety of different methods of constructing the background template [13, 14, 25–36], and we use the following three to illustrate the utility of the ARGOS metric:

1. Idealized Anomaly Detector

The idealized anomaly detector (IAD) [13] is used as a benchmark which is available for simulated data only. Instead of constructing the BT in a data-driven way, the IAD simply uses background events simulated with the same pipeline that generated the background data in the SR. Hence, by definition, $p_{\text{BT}}^{\text{IAD}}(x) = p_{\text{BT}}^{\text{ideal}}(x) = p_B(x)$. We use approximately 272 000 such events.

2. CWoLa Hunting

CWoLa Hunting [25, 26] directly uses data from short sidebands of 0.2 TeV on either side of the signal region as the background template, assuming there is little correlation of the features with the dijet invariant mass. Using short sidebands reduces correlations with the dijet mass. As these short sidebands are in total as wide as the SR, the resulting background template has a similar size to the SR.

3. CATHODE

CATHODE [13] uses conditional density estimation to interpolate the SB distribution into the SR: as the SB contains only very low amounts of signal events, learning $p(x|m \in \text{SB})$ is largely equivalent to learning the background distribution $p_B(x|m)$, which can then be sampled in the SR to obtain background events in the SR. The density estimator can be used to generate more BT events than there are SR events in order to obtain better background statistics. We use a fixed oversampling factor of four as in Ref. [5]. The implementation of the density estimation is discussed in App. A 1.

C. Classifiers

We use a total of three different classifier setups for our test cases:

1. A standard multi-layer perceptron (MLP) neural network (NN) with three hidden layers, which is trained using the binary cross-entropy (BCE) loss with the Adam optimizer for 100 epochs. We ensemble the best 10 epochs according to our different metrics, following the ensembling strategy used in Ref. [13], where the best 10 epochs based on the validation loss are used.
2. The `HistGradientBoostingClassifier`, which is a gradient boosted decision tree classifier implementation in `scikit-learn` [37]. It histograms the classification features before training. We ensemble 50 independent trainings with different 50-50 training-validation splits as was done in Ref. [24].
3. The `AdaBoostingClassifier`, which is an adaptive boosted decision tree classifier implementation in `scikit-learn` [37]. We build our implementation in analogy to Ref. [24] and perform an ensembling of 10 independently trained classifiers using random 50-50 splits of the training dataset.

A more detailed explanation of these classifiers as well as their hyperparameters can be found in App. A 2.

D. Performance Metrics

Throughout this work, we use the following performance metrics:

1. The maximum SIC value with a statistics cut-off at a relative statistical error on the background efficiency of 20%. This is a supervised metric for the anomaly detection performance calculated on the large held-out test set defined in Sec. III A. We refer to this metric as “max SIC” in the following.
2. The BCE loss calculated on the validation set, which is the standard metric used to perform model optimization or select the best epochs during training (see e.g. Refs. [5, 13]).
3. The maximum ARGOS calculated on the validation set as defined in Eq. (2). We refer to this metric as “max ARGOS” in the following.

IV. EXPERIMENTS

There are a number of different applications for a metric that can analyze the performance of weakly supervised anomaly detection. Here, we discuss four examples, namely: selecting the best epochs to ensemble

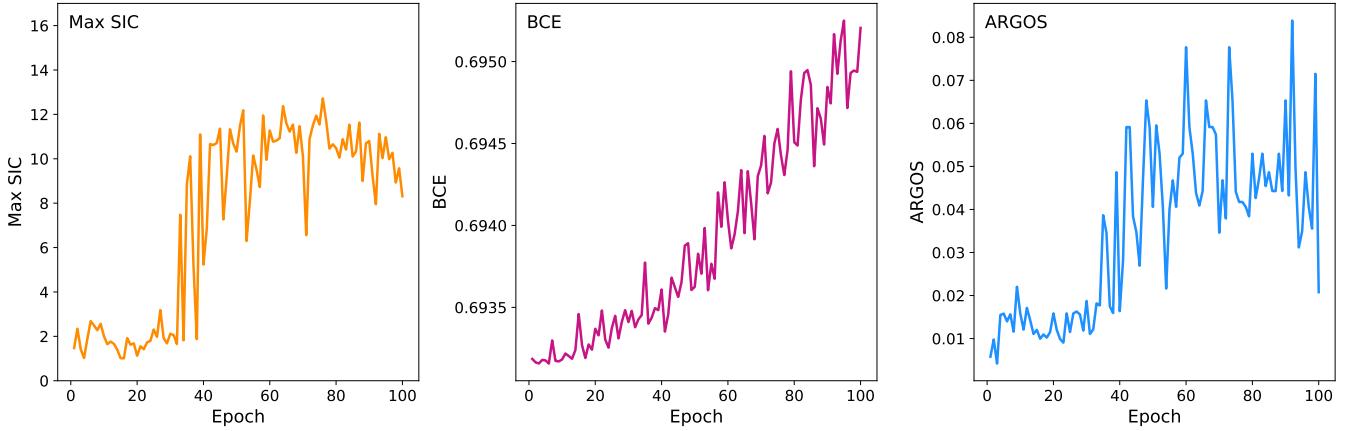


FIG. 1. Example of metrics tracked throughout a NN training with $N_{sig} = 400$ signal events using the default hyperparameters. We show the supervised max SIC metric (left) evaluated on the test set as well as BCE (middle) and max ARGOS (right), both evaluated on the validation set.

from a neural network training; optimizing hyperparameters; selecting an architecture; and selecting features for anomaly detection. However, we first highlight the correlation of ARGOS and SIC.

A. Correlation of ARGOS and SIC

In Fig. 1, we show our three performance metrics (see Section III D) tracked throughout a NN training for the IAD setup (see Sec. III B 1).

Comparing the standard BCE validation loss to the max SIC, there is no visible correlation. Instead, the validation loss very quickly shows overtraining, where the classifier tries to tell apart indistinguishable background events on the basis of statistical fluctuations. As the BCE loss takes the average over all events and most events are background, they dominate the metric.

ARGOS instead focuses on the events with the highest anomaly score, since max ARGOS is found for large background rejection $1/\epsilon_B$. Therefore it is not as sensitive to background overtraining. While we expect an almost perfect correlation between max SIC and max ARGOS according to Eq. (6), the validation-set ARGOS is severely statistics limited by the small number of signal events in the data. Thus the max ARGOS has a slightly noisy correlation with the max SIC but shows the same underlying trend.

In this particular example, the signal is not found at the beginning of the training but instead after about 30 epochs, where the BCE already shows significant overtraining. This is not the case for every network initialization but it is common for the max SIC to show a delayed onset of the signal identification for lower signal injections, which the BCE is unable to identify. Hence, using early stopping could be counter productive.

B. NN epoch selection

As noted in Sec. III C, we ensemble the ten best epochs for the NN to obtain a lower variance and to stabilize the anomaly-detection performance. Which epochs are selected strongly depends on the metric used, as can be seen in Fig. 1 for example. The impact of ensembling based on our three tested metrics for the IAD, CWoLa Hunting and CATHODE is shown in Fig. 2. We show the median anomaly detection performance of ten trainings as well as an error band representing the 16 to 84 percent quantiles.

By definition, selecting the epochs based on our supervised target metric, the max SIC, results in the optimal performance in all cases. Selecting using the max ARGOS results in a slightly lower and somewhat noisier performance but clearly outperforms the selection based on the BCE loss, which especially falls short for low signal injections.

This difference is particularly pronounced for CWoLa Hunting: when using BCE for epoch selection, the max SIC drops below 10 already at $N_{sig} = 900$, whereas with ARGOS this drop only occurs around $N_{sig} = 600$. In our setup, CWoLa Hunting generally has the worst quality background template of the three methods considered.¹ Therefore, the NN can learn to distinguish background template and signal region based on the background events only, which results in a lower validation BCE loss but does not transfer onto a better performance on the anomaly detection test set. Instead using the ARGOS metric, and in particular max ARGOS, allows the

¹ The background template of the IAD contains no mismodeling by definition and the flow-matching based density estimate used for CATHODE is highly accurate for the small feature space used here.

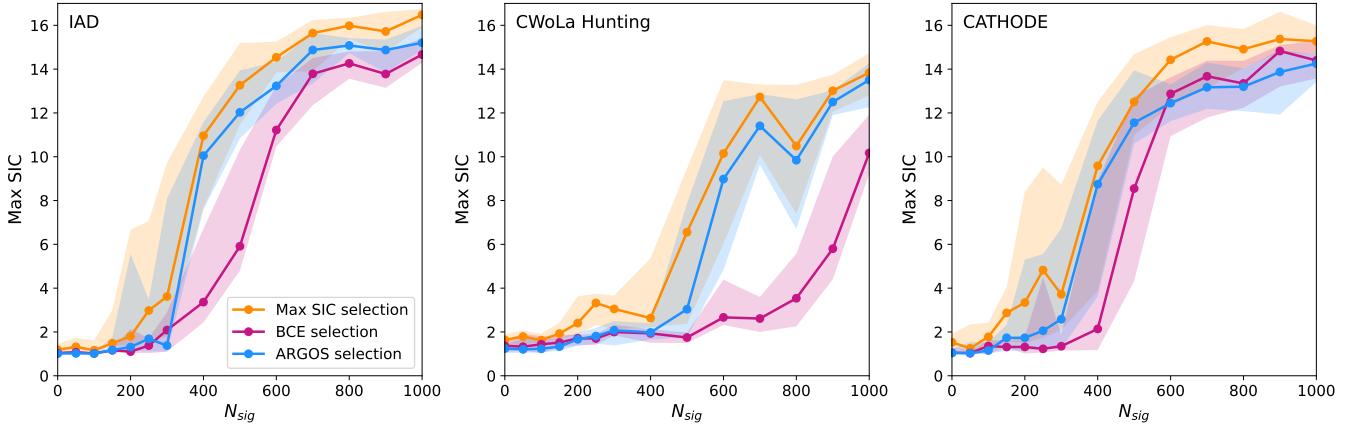


FIG. 2. Anomaly detection performance (max SIC) as a function of the number of signal events N_{sig} after epoch selection. The epoch selection is performed using the supervised benchmark metric max SIC and the two data-driven metrics (max ARGOS and BCE), shown for IAD (left), CWoLa Hunting (middle) and CATHODE (right).

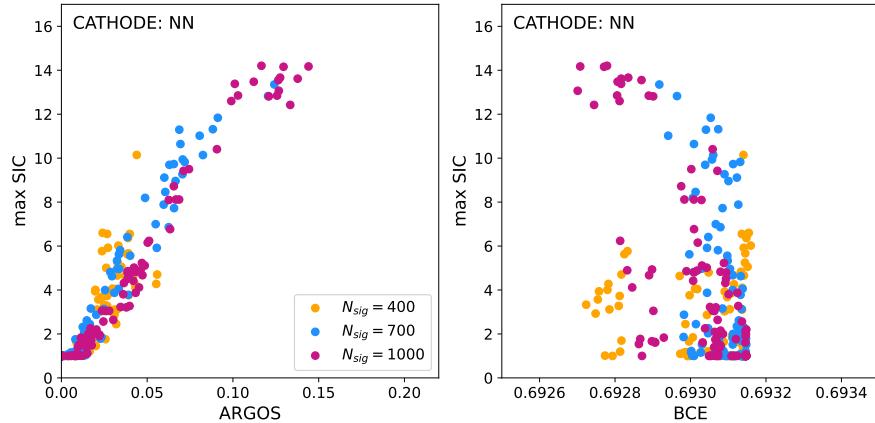


FIG. 3. Correlation between median anomaly detection performance max SIC and two data-driven metrics max ARGOS (left) and BCE (right) for all 100 hyperparameter sets for CATHODE using the NN classifier at three example signal injections.

epoch selection to focus on the events with the highest anomaly score.

An interesting feature can be seen in the performance of the max SIC based epoch selection at $N_{sig} = 0$, where the performance is slightly better than random. The fluctuations of the classifier weights in the absence of signal still lead to some epochs with non-zero anomaly-detection performance, which the max SIC identifies. Selecting the epochs using signal information can therefore enhance the false positive rate. Using ARGOS, which does not know anything about a potential signal, this issue is not present.

In the rest of the paper, we select epochs using the max ARGOS metric, as this is the best-performing data-driven metric for this application.

C. Hyperparameter optimization

An especially important application of a data-driven metric for weak supervision is data-driven hyperparameter optimization. As an illustration, we perform this optimization using a random-search strategy, where we randomly sample different hyperparameters $N_{hp} = 100$ times from a range of possible values as given in App. A 2. Using each hyperparameter set, the classifier is trained and all three metrics are evaluated; BCE and ARGOS on the validation set and max SIC on the test set. Then the best hyperparameter set based on each metric is selected. This procedure is performed ten times for each signal injection N_{sig} to obtain a stable median as well as 16 to 84 percent quantile error bands. A more detailed explanation of the procedure can be found in App. B.

In Fig. 3, we show the correlation between our data-driven metrics and the anomaly detection performance (max SIC) for the NN based CATHODE optimization

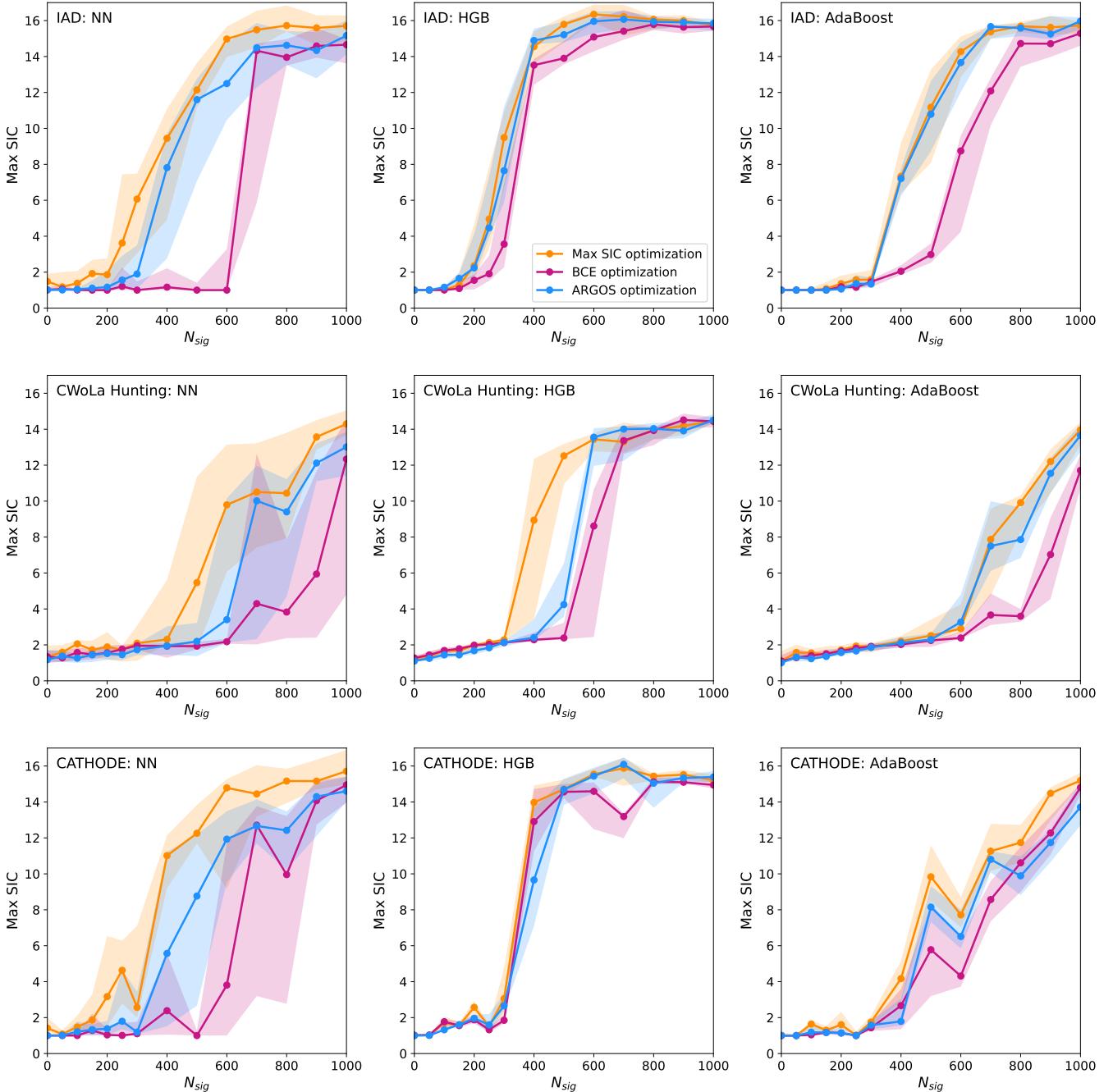


FIG. 4. Anomaly detection performance (max SIC) after hyperparameter optimization for the NN (left), HGB (middle) and AdaBoost (right) classifiers. The optimization is performed using the supervised benchmark metric max SIC and the two data-driven metrics (max ARGOS and BCE), shown for IAD (top), CWoLa Hunting (middle), and CATHODE (bottom).

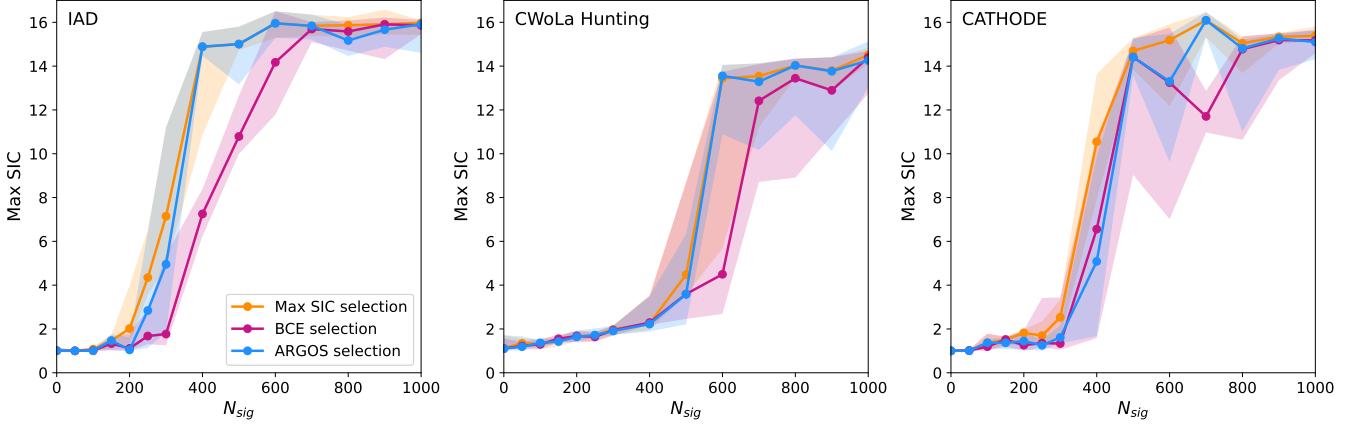


FIG. 5. Anomaly detection performance (max SIC) after architecture selection based on the supervised metric max SIC as a benchmark and the two data-driven metrics max ARGOS and BCE for IAD (left), CWoLa Hunting (middle) and CATHODE (right).

at three different signal injections. The range of different hyperparameter configurations result in a large spectrum of performance. We observe a much stronger correlation with max SIC for max ARGOS than for BCE, particularly for lower signal injections. For example, we find relatively low BCE values for low max SIC values at $N_{sig} = 400$. These low BCE values are likely achieved by optimizing on the differences between BT and SR background. We do not see a similar effect for the ARGOS metric. Analogous results are also obtained for the IAD and CWoLa Hunting and the other classifier architectures. Hence, we expect the optimization based on ARGOS to be more stable and to yield better results, especially at lower signal injections.

This is demonstrated in Fig. 4, which shows the significance improvement after hyperparameter optimization with the different metrics. Starting with the IAD results of the hyperparameter optimization in the top row of Fig. 4, we can once more see that an optimization using the supervised max SIC metric of course leads to ideal results. Second best in all cases is optimizing based on ARGOS, which in particular for the NN leads to a slightly noisier but still similar performance compared to optimizing on the max SIC. Optimizing on the BCE loss results in a worse anomaly detection performance, which is especially significant for the NN and AdaBoost. This effect is less prominent for the HGB, likely because its histogramming of the input data limits overtraining. These trends generally persist for CWoLa Hunting and CATHODE in the bottom two rows of Fig. 4, although the imperfect background templates cause the results to be slightly noisier, especially for CATHODE where the individual density estimation runs cause the BT quality to slightly vary between different signal injections.

D. Architecture selection

In addition to optimizing the hyperparameters of a given architecture, one may also be interested in selecting the architecture itself. We train our three architectures (with hyperparameters optimized using ARGOS) on 50% of our standard training and validation data. On the other half, we calculate BCE and ARGOS once the trainings are complete. We can then select the best architecture based on these two metrics as well as on the max SIC of the half-statistics training. The selected architecture is then trained on the full statistics and evaluated on the test set. We perform a separate architecture selection for each of the ten optimized hyperparameter sets obtained in Sec. IV C at each signal injection. We plot these ten runs as a median and an error band corresponding to the 16 to 84 percent quantiles. A more detailed description of the architecture-selection procedure can be found in App. C.

Fig. 5 shows the results of the architecture selection for IAD, CWoLa Hunting and CATHODE. Again, selecting based on max SIC or ARGOS results in very similar performance, while a worse performance is observed when selecting based on the BCE. Interestingly, the effect is not the same for the IAD as it is for CATHODE or CWoLa Hunting. For the IAD, BCE results in a performance drop at higher signal injections compared to ARGOS and max SIC, whereas for CWoLa Hunting and CATHODE the dominant effect is a larger error band. This large error band originates from the same metric selecting different classifier architectures across the ten runs. For example, in the case of CATHODE, the BCE metric largely chooses the HGB architecture, which leads to the median largely matching the optimal performance; but sometimes (three times out of ten) it chooses AdaBoost, which leads to the large downward shift of the error band. For CWoLa Hunting the selection of worse

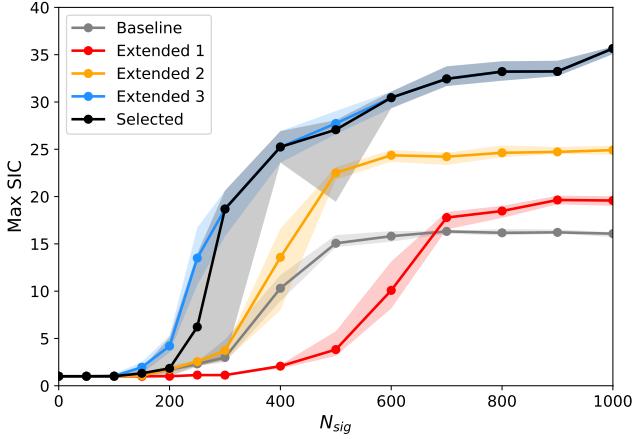


FIG. 6. Anomaly detection performance max SIC for different subjettness-based feature sets and for the feature set selection using max ARGOS in black. We use the IAD setup with the HGB classifier. The baseline set is defined in Section III. The extended sets are introduced in Ref. [24].

models becomes common enough to also lead to a downward shift of the median.

In our case, it was necessary to perform multiple independent architecture selections in order to obtain error bands to help compare the different metrics. In an actual analysis, however, it may be more effective to select architectures based on the mean or median metric obtained from multiple trainings of the same hyperparameter configuration, which could further stabilize the performance and reduce downward fluctuations of the selection.

E. Feature selection

For anomaly detection to be truly signal model agnostic, feature sets should not be tuned on specific signals. To test the concept of choosing the best feature set based on data-driven metrics, we apply the procedure used for the architecture selection (see Sec. IV D) to the feature sets used in Ref. [24] for the IAD in Fig. 6. The largest feature set, extended set 3, is in this case most sensitive and at high signal injections always chosen by the ARGOS metric. At lower signal injections ARGOS sometimes also selects less performing feature sets as can be seen from the increased error band and the slightly earlier performance drop-off.

This IAD proof-of-concept study shows the potential viability of data-driven feature selection in general.

V. CONCLUSION AND OUTLOOK

In this paper, we have introduced the fully data-driven ARGOS metric for choosing the best anomaly detector. Given an accurate template of background events, we

show that the ARGOS metric has a sound theoretical grounding – it is monotonic with the significance improvement characteristic for any unknown signal in the data. This guarantees the effectiveness of the ARGOS metric for comparing arbitrary anomaly detectors.

We demonstrated the power of the ARGOS metric with a number of realistic use cases in the context of weak supervision. For two well-known methods (CWoLa Hunting and CATHODE), applications to epoch selection, hyperparameter optimization, and architecture selection showed that the ARGOS metric is a better proxy for the true anomaly detection performance than the binary cross-entropy loss, which is the default standard currently used in classifier-based weakly-supervised anomaly detection.

In this work, we employ the maximum of the ARGOS metric as a function of ϵ_{BT} , i.e., we optimize the working point ϵ_{BT} and the anomaly detector at the same time. One could just as easily evaluate ARGOS at a specific background template efficiency ϵ_{BT} , if ϵ_{BT} is fixed in a given analysis.

We stress that this is not the only context where the ARGOS metric can be used. For example, it is directly applicable to ANODE [27] and R-ANODE [38], which are density-estimator-based resonant anomaly detection methods. Beyond resonant anomaly detection, different outlier detectors (e.g. autoencoders) could also be compared to one another if a background template is available. In contrast to the binary cross-entropy loss for classifier-based anomaly detection, ARGOS does not assume predictions to represent a class probability. ARGOS is a cut-based metric, which is only affected by the relative ordering of different events and invariant to rescaling or shifting of the entire anomaly score. Therefore, ARGOS can be applied to any anomaly score.

As an illustration of the potential for data-driven feature selection, we carried out a proof-of-concept study using the IAD setup, where ARGOS reliably identifies the best-performing feature sets at high signal injections. However, for a realistic background template, ARGOS will exhibit a bias towards worse background templates. The mitigation of this issue needs to be studied in future work in order for data-driven feature selection to be viable on real data.

ACKNOWLEDGEMENTS

The authors thank Tobias Quadfasel for his contributions in the early stages of this project. MH is supported by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under grant 400140256 - GRK 2497: The physics of the heaviest particles at the Large Hadron Collider. The research of MK and AM is supported by the DFG under grant 396021762 - TRR 257: Particle Physics Phenomenology after the Higgs Discovery. GK and LM acknowledge support by the DFG under Germany's Excellence Strategy 390833306 – EXC

2121: Quantum Universe. DS is supported by DOE grant DOE-SC0010008. Computations were performed with computing resources granted by RWTH Aachen University under project rwth0934.

CODE

The code for this paper can be found in Ref. [39].

Appendix A: Architectures and training

1. Implementation of the CATHODE Density Estimation

In order to obtain state of the art results, we use a conditional flow matching (CFM) density estimator based on Ref. [34]. In this implementation [40], the conditional vector field needed for CFM is learned using a ResNet-style [41] architecture from `nflows` [42]. The hyperparameters from Ref. [34] are used for both the model and the training. For each signal injection, ten density estimators are trained and ensembled by combining their samples in order to limit fluctuations of the density estimation quality between the different signal injections.

2. Classifier implementations and hyperparameter settings

a. NN

For the NN, we use a standard MLP based on the architecture used in Ref. [13] with an implementation in PyTorch [43] based on Ref. [44]. We choose a fixed number of three hidden layers with 64 nodes each and ReLU activation. We train on the binary cross-entropy loss using the ADAM optimizer [45] for 100 epochs. Other hyperparameters such as batch size, dropout, learning rate and other ADAM settings are optimized later on. Their default values can be found in Tab. I. For the NN, we draw hyperparameters from a set of choices, which are also included in the table.

We choose a 50-50 training and validation split, calculating BCE and max ARGOS on the validation set and max SIC on the test set after every epoch. After training, we ensemble the best 10 epochs as decided by the different metrics. This follows the ensembling strategy used in Ref. [13], where the best 10 epochs based on the validation loss are used. The impact of choosing different metrics can be seen in Sec. IV B. As a default, we pick the epochs on max ARGOS.

Hyperparameter	Default	Choices
Learning rate	10^{-3}	[0.01, 0.005, 0.001, 0.0005, 0.0001]
Batchsize	128	[64, 128, 256, 512, 1024, 2048, 5096]
Dropout	0	[0, 0.1, 0.2, 0.3, 0.4, 0.5]
Weight Decay	0	[0, 10^{-5} , 10^{-4} , 10^{-3} , 10^{-2}]
Momentum	0.9	[0.8, 0.9, 0.95, 0.99]

TABLE I. Default NN hyperparameter settings and discrete choices sampled from for hyperparameter optimization

Hyperparameter	Default	Sampling	Min	Max
Learning rate	0.1	Log10Uniform	10^{-4}	$10^{-0.1}$
Max. Iterations	200	UniformInt	2	200
Max. Leaf Nodes	31	UniformInt	2	100
Max. Depth	None	UniformInt	2	15
Max. Bins	255	UniformInt	31	255
L2 Regularization	0	Uniform	0	10

TABLE II. Default HGB hyperparameter settings and distributions sampled from for hyperparameter optimization

b. HGB

`HistGradientBoostingClassifier` is a gradient boosted decision tree classifier implementation in `scikit-learn` [37], which histograms the classification features before training. It is currently state of the art for weakly supervised anomaly detection [24, 46] with high-level feature. Following Ref. [24], we perform 50 independent trainings with different 50-50 training-validation splits, which we ensemble by taking the mean over the predictions. Our default hyperparameters are largely equivalent to the default of the `scikit-learn` package and can be found in Tab. II. For hyperparameters optimization, we sample hyperparameters from the distributions also shown in the table.

c. AdaBoost

`AdaBoostingClassifier` is an adaptive boosted decision-tree classifier implementation in `scikit-learn`, which was investigated as an alternative to HGB in [24] but did not achieve the performance or robustness of HGB. We build our implementation in analogy to [24] and perform an ensembling of 10 independently trained classifiers using random 50-50 splits of the training dataset. AdaBoost does not make use of a validation set. However, as training is deterministic, the random data split results in variations between different trainings, which can be used to obtain a better performance through ensembling. Our default hyperparameters reflect

Hyperparameter	Default	Sampling	Min	Max
Learning rate	1	Log10Uniform	10^{-4}	$10^{-0.1}$
Number Estimators	50	UniformInt	2	100
Max. Leaf Nodes	31	UniformInt	2	100
Max. Depth	None	UniformInt	2	15
Max. Samples per Leaf	20	UniformInt	1	100

TABLE III. Default AdaBoost hyperparameter settings and distributions sampled from for hyperparameter optimization

the default used for AdaBoost in [24] and can be found in Tab. III. The distributions used to randomly sample hyperparameters for optimization can be found there, as well.

Appendix B: Hyperparameter optimization procedure

As explained in Sec. IV C, the hyperparameter optimization in this work follows a random search strategy with a random sampling of $N_{hp} = 100$ hyperparameter configurations. The distributions these hyperparameters are sampled from can be found in App. A 2. A classifier is trained with each of these configurations, BCE and max ARGOS are evaluated on the validation set and max SIC on the test set. We use slightly different strategies of stabilising the metrics and speeding up the computation time for each of our architectures:

- For the NN, each network is trained for 20 epochs, with the epoch number set to 100 after optimization. All three metrics are evaluated after each epoch and the best value for each (minimum for BCE, maximum for max SIC and max ARGOS) is selected.
- For AdaBoost and HGB, we do not ensemble during the optimization, instead the metrics evaluated after training are averaged over ten runs to obtain more stable results.

Based on the metric values obtained in this way, the best hyperparameter configuration based on each metric is selected. This entire procedure is performed ten times for each signal injection N_{sig} .

Appendix C: Details of the architecture selection

In this appendix, we explain the procedure of the architecture selection in more detail using the example of picking the best architecture for the IAD using ARGOS in Fig. 7. In Fig. 5 of Sec. IV D this is shown as a single line with an error band but in Fig. 7 the entire procedure can be seen.

The first step of our architecture selection is to train our architecture options – NN, HGB and AdaBoost, on 50% of our normal training and validation set. The anomaly detection performance characterized by the max SIC of these runs is shown in the middle panel of Fig. 7. On the other half of the data, we then calculate the ARGOS metric, the result of which is shown in the left panel of Fig. 7. For both of these panels we show the median and 16 to 84 percent quantiles of ten runs as lines and error bands respectively. These ten runs contain not only ten independent trainings but also ten different hyperparameter configurations per model as we use the results from Sec. IV C. For each of these sets of ten runs, we then pick the best-performing architecture between NN, HGB and AdaBoost based on the obtained max ARGOS values and plot the resulting set of ten again as median and quantile-error band. We perform this 50-50 split of the data set in order to have the metric values used for architecture selection be calculated on a held-out evaluation set. As all classifiers use the validation set in one form or another, this is not the case when directly selecting on the validation set.

For this set of selected architectures, we then run another training using the full statistics, which leads to the black line and error band in the right panel of Fig. 7. For comparison, we also show the full statistics performance of the different architectures. One can see that the final performance of the selected runs largely matches that of the best performing architecture, namely the HGB, with only some downward fluctuations.

-
- [1] G. Kasieczka *et al.*, *Rept. Prog. Phys.* **84**, 124201 (2021), arXiv:2101.08320 [hep-ph].
- [2] T. Aarrestad *et al.*, *SciPost Phys.* **12**, 043 (2021), arXiv:2105.14027 [hep-ph].
- [3] V. Belis, P. Odagiu, and T. K. Aarrestad, *Rev. Phys.* **12**, 100091 (2023), arXiv:2312.14190 [physics.data-an].
- [4] HEP ML Community, “A Living Review of Machine Learning for Particle Physics,” .
- [5] CMS Collaboration, *Rept. Prog. Phys.* **88**, 067802 (2024), arXiv:2412.03747 [hep-ex].
- [6] ATLAS Collaboration, *Phys. Rev. Lett.* **125**, 131801 (2020), arXiv:2005.02983 [hep-ex].
- [7] ATLAS Collaboration, *Phys. Rev. D* **108**, 052009 (2023), arXiv:2306.03637 [hep-ex].
- [8] ATLAS Collaboration, *Phys. Rev. Lett.* **132**, 081801 (2024), arXiv:2307.01612 [hep-ex].
- [9] G. Aad *et al.* (ATLAS), *Phys. Rev. D* **112**, 072009 (2025), arXiv:2502.09770 [hep-ex].
- [10] ATLAS Collaboration, *Phys. Rev. D* **112**, 012021 (2025), arXiv:2505.01634 [hep-ex].
- [11] CMS Collaboration, *Machine-learning techniques for model-independent searches in dijet final states*, Tech.

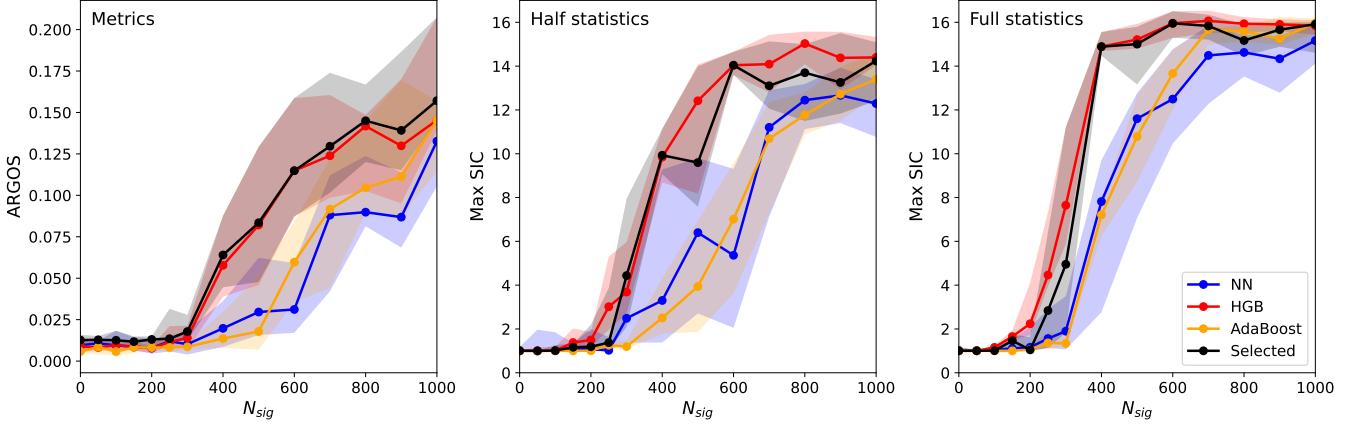


FIG. 7. Architecture selection for IAD using max ARGOS. Left: Max ARGOS calculated on half the data when trained on the other half. Middle: Max SIC of the trainings of half the data. Right: Max SIC of full statistics training. In each case the selected runs are shown in black. Note that the x-axis shows the full statistics signal injection values even for the half statistics runs.

- Rep. (CERN, 2025) CMS Physics Analysis Summary CMS-PAS-MLG-23-002.
- [12] CMS Collaboration, (2025), submitted to *Eur. Phys. J. C*, arXiv:2509.13635 [hep-ex].
 - [13] A. Hallin, J. Isaacson, G. Kasieczka, C. Krause, B. Nachman, T. Quadfasel, M. Schlaffer, D. Shih, and M. Sommerhalder, *Phys.Rev.D* **106**, 055006 (2021), arXiv:2109.00546 [hep-ph].
 - [14] E. M. Metodiev, B. Nachman, and J. Thaler, *JHEP* **10**, 174 (2017), arXiv:1708.02949 [hep-ph].
 - [15] G. Kasieczka, B. Nachman, and D. Shih, “R&d dataset for lhc olympics 2020 anomaly detection challenge,” <https://zenodo.org/record/6466204> (2019).
 - [16] D. Shih, “Additional qcd background events for lhco2020 r&d (signal region only),” <https://zenodo.org/record/5759086> (2021).
 - [17] T. Sjöstrand, S. Mrenna, and P. Z. Skands, *JHEP* **05**, 026 (2006), arXiv:hep-ph/0603175 [hep-ph].
 - [18] T. Sjöstrand, S. Mrenna, and P. Z. Skands, *Comput. Phys. Commun.* **178**, 852 (2008), arXiv:0710.3820 [hep-ph].
 - [19] J. de Favereau, C. Delaere, P. Demin, A. Giannanco, V. Lemaitre, A. Mertens, and M. Selvaggi (DELPHES 3), *JHEP* **02**, 057 (2014), arXiv:1307.6346 [hep-ex].
 - [20] M. Cacciari, G. P. Salam, and G. Soyez, *Eur. Phys. J. C* **72**, 1896 (2012), arXiv:1111.6097 [hep-ph].
 - [21] M. Cacciari and G. P. Salam, *Phys. Lett.* **B641**, 57 (2006), arXiv:hep-ph/0512210 [hep-ph].
 - [22] J. Thaler and K. Van Tilburg, *JHEP* **03**, 015 (2011), arXiv:1011.2268 [hep-ph].
 - [23] J. Thaler and K. Van Tilburg, *JHEP* **02**, 093 (2012), arXiv:1108.2701 [hep-ph].
 - [24] T. Finke, M. Hein, G. Kasieczka, M. Krämer, A. Mück, P. Prangchaikul, T. Quadfasel, D. Shih, and M. Sommerhalder, *Phys.Rev.D* **109**, 034033 (2023), arXiv:2309.13111 [hep-ph].
 - [25] J. H. Collins, K. Howe, and B. Nachman, *Phys. Rev. Lett.* **121**, 241803 (2018), arXiv:1805.02664 [hep-ph].
 - [26] J. H. Collins, K. Howe, and B. Nachman, *Phys. Rev.* **D99**, 014038 (2019), arXiv:1902.02634 [hep-ph].
 - [27] B. Nachman and D. Shih, *Phys. Rev. D* **101**, 075042 (2020), arXiv:2001.04990 [hep-ph].
 - [28] A. Andreassen, B. Nachman, and D. Shih, *Phys. Rev. D* **101**, 095004 (2020), arXiv:2001.05001 [hep-ph].
 - [29] K. Benkendorfer, L. L. Pottier, and B. Nachman, *Phys.Rev.D* **104**, 035003 (2020), arXiv:2009.02205 [hep-ph].
 - [30] J. A. Raine, S. Klein, D. Sengupta, and T. Golling, *Front.Big Data* **6**, 899345 (2022), arXiv:2203.09470 [hep-ph].
 - [31] A. Hallin, G. Kasieczka, T. Quadfasel, D. Shih, and M. Sommerhalder, *Phys.Rev.D* **107**, 114012 (2022), arXiv:2210.14924 [hep-ph].
 - [32] T. Golling, S. Klein, R. Mastandrea, and B. Nachman, *Phys. Rev. D* **107**, 096025 (2023), arXiv:2212.11285 [hep-ph].
 - [33] T. Golling, G. Kasieczka, C. Krause, R. Mastandrea, B. Nachman, J. A. Raine, D. Sengupta, D. Shih, and M. Sommerhalder, *Eur.Phys.J.C* **84**, 241 (2023), arXiv:2307.11157 [hep-ph].
 - [34] R. Das and D. Shih, *Phys. Rev. D* **112**, 074040 (2025), arXiv:2410.20537 [hep-ph].
 - [35] M. Leigh, D. Sengupta, B. Nachman, and T. Golling, (2024), arXiv:2407.19818 [hep-ph].
 - [36] I. Oleksiyuk, S. Voloshynovskiy, and T. Golling, *JHEP* **07**, 177 (2025), arXiv:2503.04342 [hep-ph].
 - [37] F. Pedregosa *et al.*, *Journal of Machine Learning Research* **12**, 2825 (2011).
 - [38] R. Das, G. Kasieczka, and D. Shih, (2023), arXiv:2312.11629 [hep-ph].
 - [39] M. Hein, “ARGOS metric for anomaly detection,” <https://github.com/mariehein/ARGOS-metric-for-anomaly-detection> (2025).
 - [40] R. Das, “Density Estimation for “Accurate and robust methods for direct background estimation in resonant anomaly detection,” https://github.com/rd804/cut_and_count_FM (2024).
 - [41] K. He, X. Zhang, S. Ren, and J. Sun, arXiv:1512.03385 [cs.CV].
 - [42] C. Durkan, A. Bekasov, I. Murray, and G. Papamakarios,

- “nflows: normalizing flows in PyTorch,” (2020).
- [43] J. Ansel *et al.*, in *29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 2 (ASPLOS ’24)* (ACM, 2024).
- [44] O. Amram, J. Birk, and M. Sommerhalder, “sk_cathode,” https://github.com/uhh-pd-ml/sk_cathode (2024).
- [45] D. P. Kingma and J. Ba, arXiv:1412.6980 [cs.LG].
- [46] M. Freytsis, M. Perelstein, and Y. C. San, *JHEP* **02**, 220 (2023), arXiv:2310.13057 [hep-ph].