Classification uncertainty for transient gravitational-wave noise artifacts with optimized conformal prediction

Ann-Kristin Malz,[1,*] Gregory Ashton,[1] and Nicolo Colombo[2]

[1]*Department of Physics, Royal Holloway University of London, Egham TW20 0EX, United Kingdom*
[2]*Department of Computer Science, Royal Holloway University of London, Egham TW20 0EX, United Kingdom*

(Received 17 December 2024; accepted 7 April 2025; published 29 April 2025)

With the increasing use of machine learning (ML) algorithms in scientific research comes the need for reliable uncertainty quantification. When taking a measurement it is not enough to provide the result, we also have to declare how confident we are in the measurement. This is also true when the results are obtained from a ML algorithm, and arguably more so since the internal workings of ML algorithms are often less transparent compared to traditional statistical methods. Additionally, many ML algorithms do not provide uncertainty estimates, and auxiliary algorithms must be applied. Conformal prediction (*CP*) is a framework to provide such uncertainty quantifications for ML point predictors. In this paper, we explore the use and properties of *CP* applied in the context of glitch classification in gravitational wave astronomy. Specifically, we demonstrate the application of *CP* to the Gravity Spy glitch classification algorithm. *CP* makes use of a score function, a nonconformity measure, to convert an algorithm's heuristic notion of uncertainty to a rigorous uncertainty. We use the application on Gravity Spy to explore the performance of different nonconformity measures and optimize them for our application. Our results show that the optimal nonconformity measure depends on the specific application, as well as the metric used to quantify the performance.

DOI: 10.1103/PhysRevD.111.084078

I. INTRODUCTION

With the first detection of gravitational waves in 2015 [1], sourced by a pair of stellar-mass black holes colliding in another galaxy, a new way to explore the universe and a new field of astrophysics research has opened. Gravitational waves are observed using laser interferometers, such as LIGO (Laser Interferometer Gravitational-Wave Observatory) [2], Virgo [3], and KAGRA (Kamioka Gravitational Wave Detector) [4], sensitive to relative differences in the arms of the interferometers of less than 1 part in 10^{-21} caused by a passing gravitational wave. We have now observed around 100 such signals [5], but they are rare events, lasting a few seconds in year-long observing runs.

The sensitivity of the detectors is determined by background noise, which on short timescales can be approximated as quasistationary colored Gaussian noise in addition to non-Gaussian transient noise artifacts, known as "glitches" [6]. Glitches are troublesome, as they often have unknown physical origins (environmental or instrumental) or are difficult to mitigate in the detectors [6,7]. They can be mistaken for gravitational wave signals, reduce the significance of signal candidates, or bias the astrophysical parameter estimation results when occurring in temporal proximity to a signal [7,8]. Additionally, glitches occur at a rate of approximately 1 per minute [9], while we detect approximately 1 signal per week [10], depending on the thresholds used. Thus, to improve the detection of gravitational waves and the scientific research of astrophysical events, the causes of these nonastrophysical noise artifacts must be identified and minimized in the detectors, or, alternatively, the glitches must be mitigated in the data [9,11].

We generally expect to observe astrophysical signals in all detectors observing with the required sensitivity, while glitches are caused locally only. However, the high glitch rate [7] implies that accidental coincidence between detectors is possible. Furthermore, the detectors also independently record signal-free auxiliary data, which measure different aspects of the detector components and environment [12–14]. The auxiliary channels can thus witness disturbances and can be used to help distinguish astrophysical signals from noise, as well as correlate a glitch in the strain data with noise from auxiliary sensors [12,15].

Glitches come in a variety of different morphologies, with each class of glitches sharing similar features [12,16]. Accurate classification helps correlate glitch classes with auxiliary channels and subsequent identification of the

*Contact author: ann-kristin.malz@ligo.org

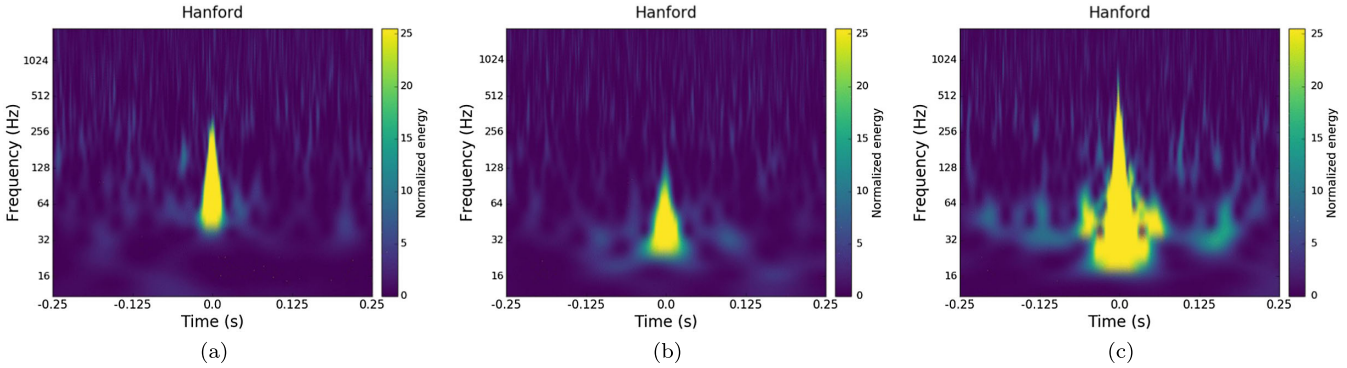

FIG. 1. Example plots of three different glitches, represented as time-frequency q-scans, as observed by the LIGO Hanford detector. The plots are the "golden images" from the Gravity Spy project, which are used to demonstrate what each glitch class looks like [20]. (a) `Blip`. (b) `Tomte`. (c) `Koi Fish`.

underlying cause [7]. Furthermore, identifying glitches with similar morphologies allows for prioritization by characteristics and quantity. Thus, classifying glitches correctly is an important first step for their mitigation, and hence is directly related to improved scientific results.

Even when the cause of a particular glitch class cannot be identified in the detectors, correct classifications can help model the glitches and hence subtract them from the data (see, for example, [17–19]). The classification of glitches depending on their features is a typical machine learning (ML) problem and is addressed by the Gravity Spy project for LIGO [20] and GWitchHunters for Virgo [21].

Gravity Spy [20] is a citizen-science and ML project to classify glitches in gravitational wave data. The ML algorithm consists of a convolutional neural network (`CNN`) [22] that is trained on human-classified time-frequency-energy plots (so-called omega scans or Q-transforms [23]) of glitches. Both citizen volunteers and the trained ML algorithm then provide classifications for new glitches.

The plots in Fig. 1 show examples of the omega scan of three different glitch classes named after their morphological features. This showcases the difficulty in correctly classifying each glitch, and the need to quantify the uncertainty of the classification, as the classes appear very similar. The similarity between some glitch classes could also suggest that they are, in fact, the same kind of glitch.

When applying the trained Gravity Spy ML algorithm to a particular glitch, the final layer in the `CNN` returns the classification score, the estimated probability, for each possible label. The predicted label is the one that received the highest score. Due to the importance of correctly classifying the glitches, it is also useful to consider the uncertainty of this classification. For example, it would be a major advantage to have a dataset of glitches and be sure that, say, 95% are classified correctly. The classification probabilities from the `CNN` can be imperfectly calibrated, as shown by the off-diagonal points in the reliability plot in Fig. 8 in the recent Gravity Spy paper [22]. Thus, no inherent well-calibrated uncertainty is associated with the predictions, creating an opportunity for an external algorithm to calibrate the uncertainty. Conformal prediction (*CP*) [24] is a framework, developed in the context of ML, to provide such uncertainty quantification for any point-prediction algorithm.

CP converts any heuristic notion of uncertainty from an algorithm to a rigorous uncertainty estimate [25]. The first step to achieving this is the definition of a score function, a so-called nonconformity measure. The nonconformity measure is built on the heuristic notion of uncertainty of the underlying pretrained ML model, and describes how well a sample conforms to other data. Smaller scores imply a better prediction by the underlying algorithm. This function can be defined arbitrarily as long as it returns a real-valued number that is representative of the nonconformity of a sample [26]. Consequently, different nonconformity measures could be used for the same application, and the question is which one to choose for optimal results. Furthermore, each nonconformity measure can be parametrized, and the performance can change significantly with different values of the parameters. Choosing which nonconformity measure to use thus becomes an optimization problem. The second step is to define a calibration dataset, containing correctly labeled data. This is used to calibrate the uncertainty for the underlying algorithm, and *CP* can thus be seen as learning the uncertainty of the algorithm from previous outputs.

In this paper, we apply *CP* to Gravity Spy, demonstrating the concept and its properties, and showing why such an uncertainty framework is beneficial. We then use this application to explore how nonconformity measures can be optimized, and we discuss how their performances on glitch classifications compare under different metrics.

We have chosen Gravity Spy as our example algorithm, as it is a well-established ML application in the gravitational wave community. It performs the task of image classification using a `CNN`, which, in the computer science literature, is a well-explored problem but lacks inherent well-calibrated uncertainty quantification. Thus, Gravity

Spy provides a straightforward application for proof of concept of *CP* and a good case study for future development.

Applying *CP* to gravitational wave research is a new approach that has (to the best of our knowledge) only been explored in Ref. [27], where a comprehensive introduction is provided, and the application of *CP* to binary classification in gravitational wave search pipelines (is the event a signal or noise?) is considered. In comparison, our work discusses multiclass classification, we apply it to a ML algorithm, and we extensively discuss the optimization problem of choosing a nonconformity measure. Optimizing nonconformity measures is relatively uncommon in the *CP* literature, but is considered, for example, in Ref. [28]. Reference [29] is one of few publications where different nonconformity measures have been compared under different metrics. Thus, our application of *CP* to Gravity Spy, as well as the comprehensive discussion of optimization and comparison of different nonconformity measures for this application, is novel.

The remainder of this paper is structured as follows. Section II defines *CP* and its properties and demonstrates the application of *CP* to the Gravity Spy ML algorithm. Section III introduces a few different nonconformity measures from the literature, which we modify and optimize, before comparing their performance when applied to Gravity Spy using different metrics. In Sec. IV, we discuss our results in the context of the literature. The code accompanying this paper is available from the Zenodo repository [30].

II. CONFORMAL PREDICTION

CP was first developed by Gammerman *et al.* [24,31] as a framework to quantify uncertainties in the context of ML. *CP* does not affect the underlying algorithm itself but uses past predictions from the algorithm to learn the uncertainty. Thus, there are no assumptions of the underlying model or data distributions, and no priors are needed. The only assumption required is that the data must be exchangeable. This makes *CP* universally applicable to any algorithm (ML or not) that returns a point prediction. In this paper, we will discuss the application to classification algorithms only, but the methods described are also applicable to regression algorithms.

CP can be applied to any classification algorithm that for each data point x produces a label y. Given a labeled dataset, *CP* generalizes the point prediction from the underlying algorithm to a prediction set Γ^α of possible labels, with a user-defined error rate $\alpha \in [0,1]$, with guaranteed validity. Validity means that for a given α, the true label, $\hat{y}$, is included in Γ^α with a probability of approximately $1-\alpha$ [26]. Specifically, this is known as marginal coverage and, more formally, it can be shown [25] that, for N calibration data points,

$$1-\alpha \leq \Pr(\hat{y} \in \Gamma^\alpha) \leq 1-\alpha+\frac{1}{N+1}. \tag{1}$$

As the number of calibration data points N increases, the approximate result $1-\alpha$ is recovered.

To apply *CP*, first, a nonconformity measure $A(x,y)$ is defined. This measures the heuristic uncertainty of the underlying algorithm so that smaller scores imply a better prediction, and can be tailored to the specific application [26]. A common example for classification is $A(x,y) = 1 - f_y(x)$, where $f_y(x)$ is the classification score of the algorithm for label y (later, we refer to this as the `baseline` nonconformity measure).

CP is applied in two steps, calibration and testing. This requires separate calibration and test datasets, each consisting of some data points x and corresponding labels y. In the calibration step, the nonconformity measure $A(x,y)$ is used to calculate a nonconformity score $s_i = A(x,y)$ for each data point x in the calibration dataset. Sorting the nonconformity scores s_i in ascending order, the $1-\alpha$ quantile $\hat{q}$ is calculated as

$$\hat{q} = s_{\lceil (N+1)(1-\alpha) \rceil}, \tag{2}$$

where N is the total number of data points in the calibration set and the ceiling function $\lceil x \rceil$ denotes the smallest integer $\geq x$. Thus, the quantile $\hat{q}$ is simply the jth element in the list of ordered scores, with $j = \lceil (N+1)(1-\alpha) \rceil$.

The nonconformity measure $A(x,y)$ can, as discussed, be any function. However, as it is used to calculate scores which are then ranked, only relative values, and their ranking, matter. Two nonconformity measures that are monotonic transforms of each other will thus result in exactly the same outcome under *CP* [26].

In the testing step, the aim is to form a prediction set Γ^α for a test data point x'. Nonconformity scores are calculated for all possible labels, and the labels y with scores less than $\hat{q}$ are included in the prediction set. The set of predicted labels is thus defined as

$$\Gamma^\alpha = \{y : A(x',y) < \hat{q}\}. \tag{3}$$

For example, an image that could be either a cat or a dog is classified by an algorithm as a cat with classification scores cat $= 0.8$ and dog $= 0.2$. Using nonconformity measure $A(x',y) = 1 - f_y(x')$, the nonconformity scores for this example image x' are $s(\text{cat}) = 0.2$ and $s(\text{dog}) = 0.8$. Assuming the quantile was previously calculated as $\hat{q} = 0.35$ from some calibration data, the prediction set becomes $\Gamma^\alpha = \{\text{cat}\}$, since $0.2 < 0.35$, but $0.8 \not< 0.35$.

Varying the error rate α will change the number of labels included in the prediction set, since a lower α, and thus a higher coverage, implies that more labels will be included in Γ^α to fulfil the validity condition in Eq. (1). As α goes to

zero, the prediction set must include the true label with 100% probability and will hence include all the labels. As α approaches one, the true label is included with 0% probability, which implies an empty prediction set. Hence, the average number of labels included in the prediction set increases as α decreases. The shape of this curve depends on the problem explored as well as the chosen nonconformity measure. The value of $1-\alpha$ when the prediction set size changes from a single label to include more than one label is known as the *CP* confidence [26]. A discussion of alternative confidence definitions can be found in [27].

A. Mondrian conformal prediction

We are also interested in conditional coverage for each class. For example, we might want a set of data points with a specific label that is certain to be at least 95% accurate, as would be critical, e.g., when using ML for medical diagnoses. To guarantee validity for each class individually, Mondrian (label conditional) *CP* can be applied [32,33]. In Mondrian *CP*, the data are split by class, and *CP* is applied for each class separately. Thus, the conditional, and by extension the marginal, labels are guaranteed to obey Eq. (1). The number of calibration data N in Eq. (1) now becomes the number of data points per label N_y, thus increasing the error and hence requiring a bigger dataset to allow for small error rates α.

For Mondrian *CP*, the $1-\alpha$ quantile in Eq. (2) becomes label conditional—the calibration step is performed separately for each class, and a different quantile $\hat{q}_y = s_{\lceil (N_y+1)(1-\alpha)\rceil}$ is obtained for each label y. In the testing step, the calculated nonconformity score for each possible label y for a test data point x' is compared to the corresponding quantile $\hat{q}_y$ and Eq. (3) becomes $\Gamma^\alpha = \{y : A(x', y) < \hat{q}_y\}$.

As we are interested in conditional coverage for each glitch class, we will use Mondrian *CP* throughout this paper.

B. Application to Gravity Spy

We now apply *CP* to the Gravity Spy glitch classification algorithm. Applying the trained Gravity Spy ML algorithm to a particular glitch outputs the most likely class label (`ml_label`) and its classification score (`ml_confidence`), as well as an array of the classification scores corresponding to each glitch class. These classification scores are the output of the final layer in the `CNN` and are used as the probability distribution of the classifier [20]. Hence, they provide the heuristic uncertainty for the algorithm, which we can use as input to our nonconformity measures.

1. The dataset

To apply *CP*, we need a dataset of glitches that contains both the true label and the predicted label for each glitch. There are multiple Gravity Spy datasets available. For the work in this paper, the "retired" dataset, available from Ref. [34], has been used, as it already contains all the information we need. This dataset contains the citizen scientist and ML classifications of glitches from the first three observing runs of the LIGO detectors. All glitches in the dataset have been classified by the ML algorithm, as well as received at least one citizen volunteer classification [35]. For each glitch, the dataset contains the ML classification scores for all classes, the ML predicted label (`ml_label`), as well as the `final_label`, which is the combined volunteer and ML classification. Thus, all the information we need is included in the dataset and there is no need for us to rerun the ML algorithm.

The "true label" of a glitch is required to calibrate the algorithm. Since there is no ground truth, human-only classifications would be the next obvious choice. However, there were no human-classified glitch datasets available to us that contained enough glitches of each class to calibrate *CP*. Thus, we have chosen to use the `final_label` in the "retired" dataset [34] as the "true label," which combines the human and ML classification scores [20]. Our "true label" thus might not always be accurate and, hence, the apparent performance of the algorithm according to the results in this dataset does not match the (significantly better) performance reported in Ref. [35]. Furthermore, the Gravity Spy ML algorithm [36] our dataset [34] is based upon has since been improved, see Ref. [22]. Nevertheless, our goal is to provide a proof of concept of how to apply *CP*.

The distribution of glitches within this dataset is far from uniform, for example, there are 327262 `Scattered Light` and 1167 `Wandering Line` glitches. We want to identify a set where there are enough glitches in each class to apply *CP*. Hence, we randomly choose 1500 glitches of each class (or all glitches of classes where fewer are available) to make up our dataset.

2. Application

First, we use the output array of classification scores to determine a simple nonconformity measure

$$A(x, y) = 1 - f_y(x), \tag{4}$$

where f_y is the classification score for label y, as given by Gravity Spy. The nonconformity measure defined in Eq. (4) is the most common for classification problems and is sometimes referred to as the hinge loss, see, e.g., Ref. [29]. In the remainder of this paper, we will refer to it as the `baseline` measure. However, this choice is not necessarily optimal. In Sec. III, we will discuss and compare alternative measures and introduce approaches to parametrize and optimize them.

Next, we use the curated Gravity Spy dataset, split equally into a calibration and test set, and follow the

calibration steps, as described above, to obtain a quantile $\hat{q}_y$ for each glitch class. We can then apply *CP* to the Gravity Spy output for a random test glitch (our test data point x'), calculating the nonconformity scores for each possible label and including all labels y with a nonconformity score smaller than $\hat{q}_y$ in the prediction set Γ^α.

To demonstrate the application, we consider the following examples, where we calculate the quantiles $\hat{q}_y$ using two different choices of error rate: $\alpha = 0.32$ (68% probability that the true label is included in the prediction set) and $\alpha = 0.1$ (90%). We then create prediction sets for a test glitch for each of the α values. Using a `Blip` glitch (randomly chosen from the test dataset) that was classified correctly by Gravity Spy with a classification score of 0.998 gives the following results:

`68%: Blip -> {Blip(0.998)},`

`90%: Blip -> {Blip(0.998), Tomte(0.0001)},`

where the glitch class on the left is the true label and {.} represents the prediction set. The numbers in parentheses represent the classification score from Gravity Spy for that glitch class.

As another example, we pick a case where Gravity Spy classifies the glitch incorrectly. Here, a `Tomte` glitch is classified as a `Koi Fish` with a classification score of 0.49. Applying *CP* gives prediction sets as follows:

`68%: Tomte -> {Koi_Fish(0.49), Tomte(0.35)},`

`90%: Tomte -> {Blip(0.0003), Koi_Fish(0.49),`
`Tomte(0.35)}.`

In this example, Gravity Spy makes an incorrect classification but *CP* still includes the correct label in the prediction set, and is guaranteed to do so approximately $9/10$ times for $\alpha = 0.1$.

Extending the example to multiple test glitches, we now consider one test data point from each glitch class (randomly chosen from our test dataset) and show the results of applying *CP* (with $\alpha = 0.1$) in Fig. 2. The figure shows one example for each glitch class to demonstrate *CP* and is not indicative of the general behavior of the respective glitch classes. For each test glitch on the x-axis, the predicted label by Gravity Spy, as well as all labels included in the prediction set, are shown on the y-axis. Correct predictions by Gravity Spy are shown on the diagonal, where the predicted label matches the true label. We observe that the *CP* set varies in size, and the true label is included in the set for all but two of the test glitches shown in the plot. This illustrates the validity property, as the true label is only guaranteed to be in the prediction set 90% of the time. The plot in Fig. 2 shows the varying prediction set size and thus the varying uncertainty of each prediction. Furthermore, some glitch classes, such as `Violin Mode` and

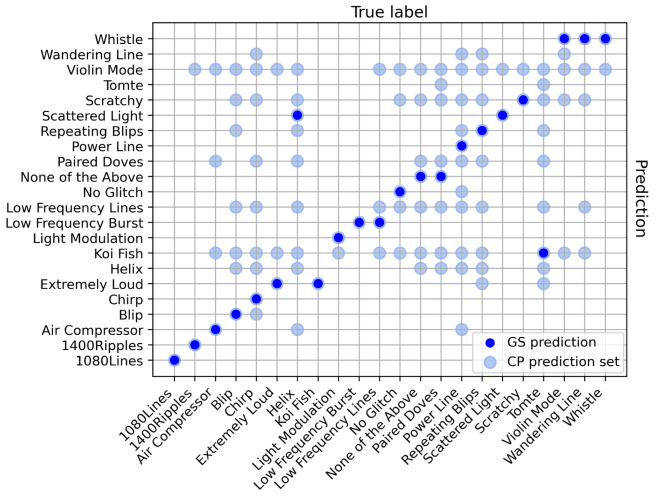

FIG. 2. Scatter plot for Gravity Spy (GS) predictions on a few chosen test glitches (dark blue points). Points on the diagonal represent correct predictions, as the true label on the x-axis and the predicted label on the y-axis agree. The plot also contains the *CP* set (using $\alpha = 0.1$) for each example glitch, represented as bigger light blue points. For each true label, there are one or several labels included in the prediction set.

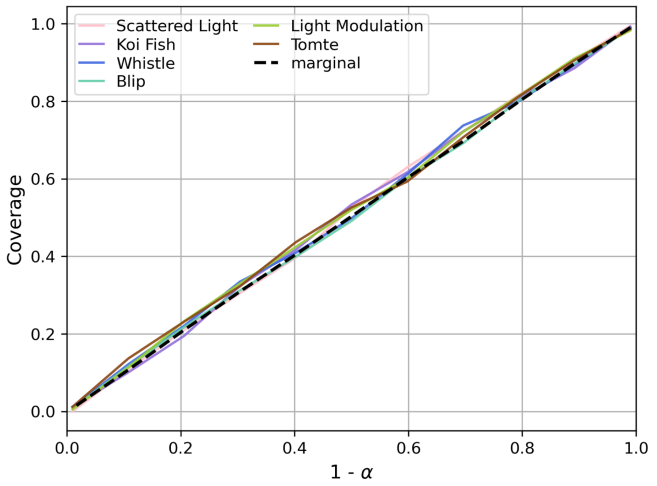

FIG. 3. *CP* conditional and marginal coverage, $\Pr(\hat{y} \in \Gamma^\alpha)$, for varying error rate α. The colored lines represent the conditional cases for a few different glitch classes, and the black dashed line represents the marginal case. This plot illustrates the *CP* validity property from Eq. (1).

`Koi Fish`, are often incorrectly included in the prediction set, implying that they are more likely to be confused with other glitch classes in this specific example. To demonstrate the concept of validity from Eq. (1) on a larger test set, we calculate the marginal and label-conditional coverage for varying α (choosing a few representative example glitches for the conditional cases). The result is shown in Fig. 3, where the diagonal line confirms the statement of validity. This is similar in nature to the probability-probability

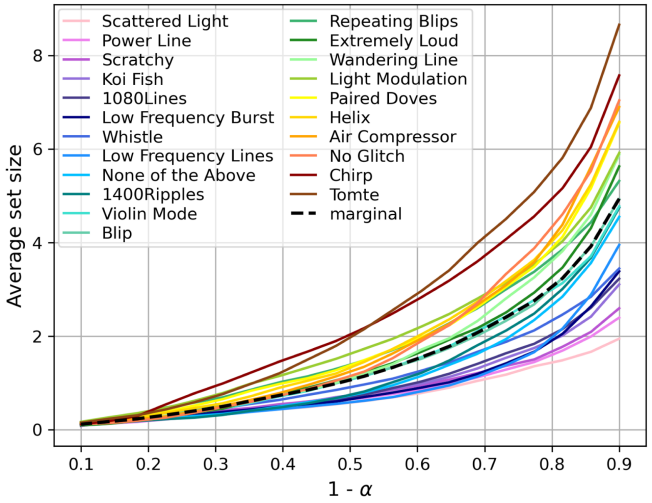

FIG. 4. Average number of labels in the *CP* set Γ^α, for varying error rate α. The colored lines represent the conditional cases for each glitch class (defined by the true labels) and the black dashed line represents the marginal case.

plots [37], which test the agreement of datasets or if a model fits the data, and are commonly used to verify the performance of various parameter estimation algorithms. It is also similar to reliability plots [38], which can be used to evaluate probabilistic predictions.

We note that the marginal coverage is on the diagonal within the expected Poisson error due to the finite sample size, while the label-conditional lines deviate. This is explained by the smaller datasets giving a larger Poisson counting error and is included by the last term in the validity guarantee in Eq. (1), where the smaller label-conditional datasets (fewer data points N) give a larger interval and thus are allowed to deviate further from the diagonal in Fig. 3. Finally, we investigate how the average prediction set size varies with the chosen error rate. One might be tempted to choose a low error rate α to get a high probability of the true label being included in the prediction set, but the cost of a lower error rate is a larger uncertainty in the form of a larger prediction set, as shown in Fig. 4.

The average number of labels can also be computed for each class separately, thus showing how some glitch classes are easier to classify and have higher average *CP* confidence than others. The confidence can be read from the plot by investigating at what $1-\alpha$ value each line reaches an average set size of two. For example, from Fig. 4 we can see that a `Tomte` glitch (the brown, upper-most line) has significantly lower confidence (50%) than a `Scattered Light` glitch (the pink, bottom-most line) with confidence of 90%. This illustrates how the similarity of, for example, `Tomte` and `Blip` glitches (see Fig. 1) often results in large prediction sets compared to the more easily uniquely classified `Scattered Light` glitch. The plot in Fig. 4 is cut off at $1-\alpha = 0.9$ for clarity, as the set size is equal to the total number of glitches (22) when $\alpha = 0$. Having demonstrated how to apply *CP*, we can now discuss the benefits. For example, if a certain glitch has a prediction set of size one, we can guarantee that this label is the true label with probability $1-\alpha$. Another objective might be to create a set of glitches of a certain class, with known uncertainty. For example, we can collect a set of glitches that have all been classified as `Tomte`, apply *CP*, and determine the *CP* confidence. We could then choose to include only those glitches that are above a certain confidence threshold, say $1-\alpha = 0.9$, thus creating a set where each glitch is guaranteed to be a `Tomte` with 90% certainty.

III. OPTIMIZING NONCONFORMITY MEASURES

There are many different nonconformity measures in the literature that could be applied instead of the `baseline` measure in Eq. (4). In this section, we review common examples and explore their performances on our Gravity Spy dataset. Furthermore, to find the optimal versions of each nonconformity measure for our application, we parametrize each function to then be optimized. An overview of the parametrized nonconformity measures is given in Table I, where β, γ, and ν are the tunable parameters. Note that we have generalized several of the nonconformity measures compared to the papers referenced by adding additional tunable parameters.

TABLE I. Nonconformity measures.

Name	Definition $A(x,y)$	Reference
`Baseline`	$1-f_y(x)$	[39]
`Softmax`	$1-f_{\beta y}, f_{\beta y} = [\mathrm{softmax}(\beta f)]_y = \frac{e^{\beta f_y}}{\sum_{y'} e^{\beta f_{y'}}}$	[40]
`Entropy`	$(1-f_y(x))\lvert 1+\gamma h(f)\rvert^\nu, \quad h(f) = -\sum_{y'} f_{y'} \log(f_{y'})$	[40]
`Entropy softmax`	$(1-f_y(x))\lvert 1+\gamma h(f_\beta)\rvert^\nu$	[40]
`Margin2`	$\max_{y'\neq y}(f_{y'})\lvert f_y+\gamma\rvert^{\lvert\nu\rvert}, \quad \gamma \geq 0$	[41]
`Maxscore2`	$\frac{1-f_y(x)}{\lvert 1+\gamma_1 f_{\max}+\gamma_2 f_{2\mathrm{ndmax}}\rvert}$	This work
`CNN`	$1-\gamma f_y + (1-\gamma)\max_{y'\neq y}(f_{y'}), \quad \gamma \in [0,1]$	[28]
`Brier`	$\frac{1}{\lvert\mathcal{Y}\rvert}\sum_{y'} \lvert \mathbf{1}(y'=y) - f_{y'}\rvert^\nu$	[29]

A. Nonconformity measures

All of the nonconformity measures are based solely on the classification scores f_y output from Gravity Spy for each label y, as this encodes the heuristic uncertainty, which *CP* transforms into a rigorous one [25]. However, if additional information was available, this could also be used to inform the nonconformity measure. The first four nonconformity measures in Table I are adaptions on the `baseline`, making use of the `softmax` function, adding weighting in the form of a cross-entropy term and a combination of both. Meanwhile, the `maxscore2` measure uses the largest and second-largest classification scores as a weight to the `baseline` measure.

The `margin2` measure makes use of the largest classification score, excluding the score of the label y currently considered. The parameter $\gamma \geq 0$ makes the measure sensitive to small changes in f_y, since decreasing γ increases the importance of f_y compared to the scores of the other labels [41].

The `CNN` measure was developed specifically for use on a convolutional neural network and is, like the `margin2` measure, constructed from the classification score of the considered label, f_y, and the largest score when excluding f_y. The parameter γ provides a trade-off between the two terms [28].

The `Brier` measure includes the classification scores of all possible labels so that the final nonconformity score is affected even if there is minor confusion about the true label [29]. The normalization factor $\mathcal{Y}$ represents the total number of labels.

It is worth noting that most of the nonconformity measures reduce to the `baseline`, Eq. (4), for specific choices of values for the tunable parameters β, γ, and ν.

B. Metrics

To optimize the nonconformity measures and compare them, we first need to define what we mean by "optimal" performance. This is not necessarily straightforward, as different applications of *CP* have different purposes and thus varying definitions of what is optimal. In this section, we describe three different metrics that can be used for such optimizations and comparisons: the average prediction set size [29], the number of correct predictions of set size one, so-called singletons [29], and the F_1 score [22,42]. We have chosen these metrics because they are commonly used in the context of machine learning and *CP* in the literature, but other metrics could also be defined and considered. Each metric has different advantages, and the choice of metric depends on what the user considers an optimal outcome.

We define the average prediction set size as

$$\texttt{set\_size} = \frac{1}{N}\sum_{n}^{N} |\Gamma_n^\alpha|, \tag{5}$$

where $|.|$ defines the set size and N is the number of test data points. The size of the prediction set Γ^α is an inherent feature of *CP* that is linked to the certainty of predictions. Minimizing the average set size implies more certain predictions and can hence be seen as minimizing the uncertainty.

The average number of singletons is defined as

$$\texttt{singleton} = \frac{1}{N}\sum_{n}^{N} \mathbf{1}(|\Gamma_n^\alpha| = 1), \tag{6}$$

where we make use of the same notation as in Eq. (5) and $\mathbf{1}$ is an indicator function. Singletons are often useful when we want to classify something uniquely. For example, when classifying a set of gravitational wave events as signals or noise, one might want to maximize the number of events that are uniquely classified as signals. In general, singletons are the metric to choose if the purity of a dataset is valued, such as in population studies.

The F_1 score is the harmonic mean of precision and recall, defined as

$$F_1 = \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} = \frac{2\text{TP}}{2\text{TP} + \text{FP} + \text{FN}}, \tag{7}$$

where true positive (TP) is the number of data points correctly predicted as "positive," false positive (FP) is the number incorrectly predicted as "positive," and false negative (FN) is the number incorrectly predicted as "negative." The number of data points correctly predicted as "negative" is the true negative (TN). The $F_1 \in [0, 1]$ score is thus defined such that a higher score implies a better prediction, as given by the trade-off between the TP rate and the FP and FN rates.

The F_1 score is generally defined for binary classification and does not directly map to *CP*, since *CP* returns a prediction set rather than a single prediction. Therefore, we extend the F_1 score so that all classes are considered as "positive" in turn, and the F_1 score is calculated for each label separately. Let us demonstrate this with an example, as follows:

(i) Blip $\rightarrow$ {Blip}: TP(Blip) $+= 1$.
(ii) Blip $\rightarrow$ {Blip, Tomte}: TP(Blip) $+= 1$, FP(Tomte) $+= 1$.
(iii) Blip $\rightarrow$ {Tomte, Koi_Fish}: FN(Blip) $+= 1$, FP(Tomte) $+= 1$, FP(Koi_Fish) $+= 1$.
(iv) Blip $\rightarrow$ {}: FN(Blip) $+= 1$.

Here, the first test data point, consisting of a true label, `Blip`, and a prediction set, {Blip}, gives TP(Blip) $= 1$, and all other counts are zero.

Taking all four test data points above and considering `Blip` glitches only, this set of examples has TP(Blip) $= 2$, FP(Blip) $= 0$, and FN(Blip) $= 2$, thus giving

$$F_1(\texttt{Blip}) = \frac{2 \cdot 2}{2 \cdot 2 + 0 + 2} = 0.67, \tag{8}$$

when applying Eq. (7).

After calculating the F_1 scores per label, they can be combined by taking the average, also known as macro F_1 [43], over all classes. There are several alternative ways to combine the individual scores, for example, using the geometric mean, but these are not considered here.

1. ROC curve

The measured TP, FP, FN, and TN values can also be used to determine the true positive rate, defined as $\mathrm{TP}/(\mathrm{TP}+\mathrm{FN})$, and the false positive rate, defined as $\mathrm{FP}/(\mathrm{FP}+\mathrm{TN})$. By varying the error rate α, the trade-off between the true positive rate and the false positive rate can be shown with a receiver operating characteristic (ROC) curve [44] (see the red-yellow curve in Fig. 5). Here, we have used the standard definition of a ROC curve, as defined for binary classification, with the caveat that the TPs, FPs, FNs, and TNs are the values summed over all glitch classes. This demonstrates how a small error rate, which is generally desirable, achieves a high true positive rate, but at the cost of also increasing the false positive rate. The ideal case is in the top-left corner of the plot, where the true positive rate is maximized and the false positive rate is minimized.

The blue-green line in Fig. 5 is the ROC curve for our Gravity Spy dataset, where the true positive and false positive rates are calculated for multiclass classification (an illustrative multiclass confusion matrix demonstrating this can be found, for example, in Fig. 3 in Ref. [45]). We observe that for higher false positive rates, the *CP* curve has higher true positive rates than the Gravity Spy curve. However, interestingly, the two curves intersect, and the Gravity Spy curve has a higher true positive rate at low false positive rates. Furthermore, we can compare the area under the ROC curve, which has a value of 0.929 for *CP* and 0.861 for Gravity Spy. With these two results, we show that using *CP* to calibrate the Gravity Spy predictions can improve the overall classification performance.

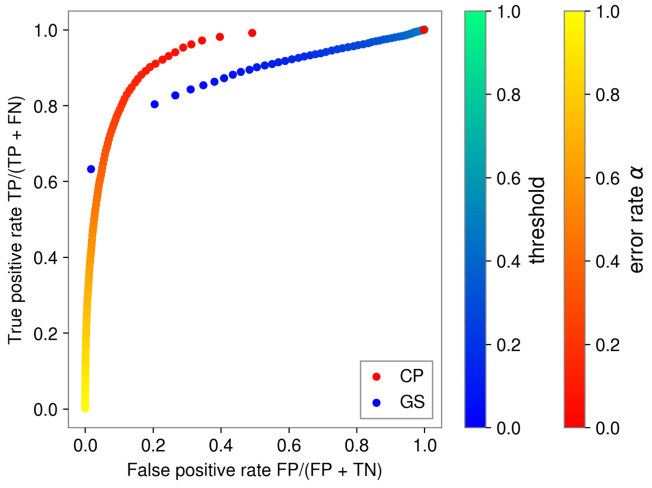

FIG. 5. ROC curves for *CP* for varying error rate α and GS for varying the threshold.

C. Optimizing nonconformity measures

To optimize the nonconformity measures in Table I, we use both a grid search and SciPy optimize [46] with the `L-BFGS-B` method [47,48]. From the grid search, we can obtain an understanding of the topology of the chosen parameter space, while using the `L-BFGS-B` method in SciPy is more efficient to solve the optimization problem. We perform the optimization separately for each metric, and find the best parameters by maximizing the F_1 score, maximizing the number of singletons, and minimizing the average set size, respectively. Note that the prediction sets will not be affected if the nonconformity measure is changed monotonically [26].

Due to the discrete nature of the underlying function, optimization algorithms that use autodifferentiation can struggle to find the global maxima. On large scales, our metrics appear to vary smoothly with the parameters in the nonconformity measures. However, if the scale over which the parameters are varied becomes comparable to $1/N$, where N is the size of the calibration dataset, it is no longer approximately smooth but becomes visibly discreet. The optimization can be improved by increasing the step size of the algorithm, to ensure it continues on from local maxima. Alternatively, applying a differentiable approximation of the F_1 score could be considered if the objective is improved optimization. For our purpose of demonstrating that nonconformity measures can be optimized, using grid search and SciPy is sufficiently accurate.

For the `entropy` and `entropy softmax` nonconformity measures, there is degeneracy for all metrics along both axes; where one of the parameters is zero, the measure reduces to the `baseline` and we obtain unchanged scores. The degeneracy makes the optimization difficult, as no one optimum exists, and unchanged scores along an axis implies that any parameter value gives an equally good nonconformity measure so no improvements are obtained. To avoid this, we add a regularization, $\mathcal{R}$, in the optimization, such that we instead maximize $F_1 - \mathcal{R}$ and $\texttt{singleton} - \mathcal{R}$, and minimize $\texttt{set\_size} + \mathcal{R}$. We use $\mathcal{R} = \rho(\gamma^2 + \nu^2)$ and $\mathcal{R} = \rho(\nu^2 + \beta^2)$, respectively, for the two nonconformity measures, where $\rho = 0.00001$. Large parameter values are thus penalized and the degeneracy is removed. The small factor ρ ensures that although we break the degeneracy along the axes, the regularization does not affect the overall results.

To perform the optimization, we first split our dataset (as described in Sec. II B 1) equally into an optimization and an evaluation set. The optimization set is used to find the optimal parameters and the evaluation set is used to

calculate the scores of each metric for the obtained best parameters. Each of the optimization and evaluation datasets are then split equally into calibration and test sets to apply *CP*. We repeat the optimization five times, using different calibration/test data splits of the optimization dataset, and can thus obtain standard deviations on the optimized parameters.

In the following three subsections, we will discuss and show the optimization results for each of the three metrics for some of our nonconformity measures. All the optimizations in these sections use an error rate of $\alpha = 0.1$.

1. Results: F_1 score

To demonstrate the optimization process, Fig. 6 shows the parameter space grid plots of the F_1 scores, as calculated for the two varying parameters in the respective nonconformity measures, overlaid with the results from several SciPy optimization runs. The plots visualize the complicated topology of the parameter space for the different nonconformity measures and show that some regions of the parameter space are greatly preferred over others.

Studying the individual nonconformity measure equations in Table I, it is clear that, for all the measures but `softmax`, `margin2`, and `Brier`, certain parameter choices reduce the measure to the `baseline`. For example, $\gamma = 0$ or $\nu = 0$ in the `entropy` measure recover the `baseline`. When applying the F_1 score optimization to these nonconformity measures, they all optimize to the `baseline`. As discussed, setting either of the parameters to zero for the `entropy` and `entropy softmax` measures reduces them to the `baseline` measure and results in the highest F_1 scores. Applying the regularization, this behavior is still visible, as shown in Figs. 6(a) and 6(b), but the degeneracy is removed and the optimization improved. The `maxscore2` nonconformity measure also optimizes to the `baseline` measure, for $\gamma_1 = \gamma_2 = 0$, as shown in Fig. 6(d). The `margin2` measure in Fig. 6(c) does not reduce to the `baseline` for any parameters. It is optimized with $\gamma = 0$ and $\nu = -8.8 \pm 1.5$, which gives F_1 scores comparable to the `baseline`, see the discussion in Sec. III D. Despite the appearance in the plot, there is no degeneracy along $\gamma = 0$ when inspecting the values.

2. Results: Singletons

When maximizing the number of singletons, we observe that the nonconformity measures no longer optimize to values that reduce them to the `baseline`, as can be seen in Fig. 7. In fact, all nonconformity measures shown in Fig. 7 can be optimized to give better results than the `baseline` measure when using singletons as the comparison metric.

We also note that there is partial degeneracy for the singleton plots, however, this is mainly due to the scores being very close together and thus not distinguishable on the color scheme of the plots.

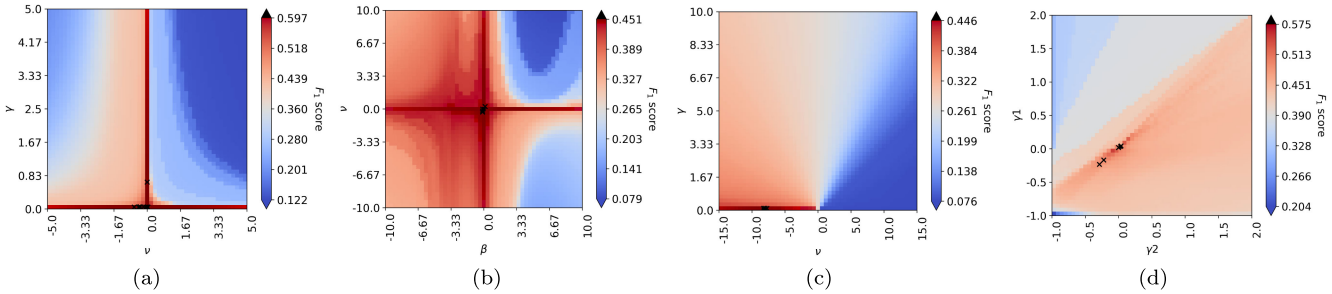

FIG. 6. Grid plots with SciPy optimizations (black cross) using the F_1 score for four example nonconformity measures. The definition of each nonconformity measure is given in Table I. (a) `Entropy`. (b) `Entropy softmax`. (c) `Margin2`. (d) `Maxscore2`.

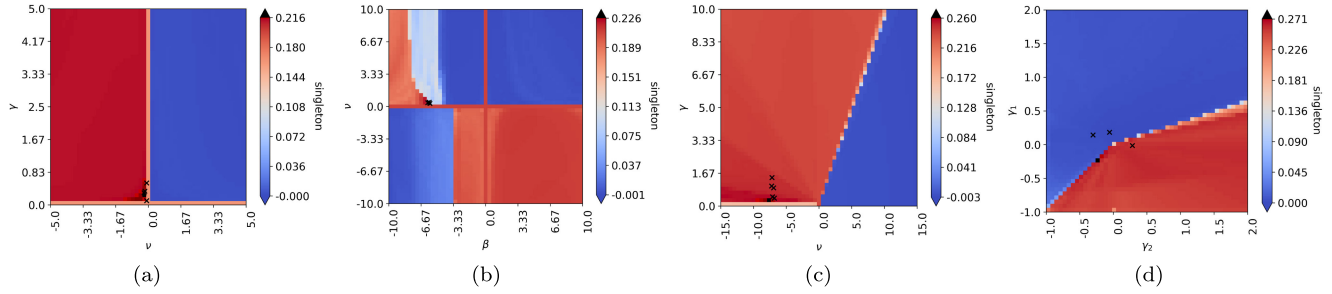

FIG. 7. Example grid plots for four different nonconformity measures with SciPy optimizations (black cross), maximizing the number of singletons. The definition of each nonconformity measure is given in Table I. (a) `Entropy`. (b) `Entropy softmax`. (c) `Margin2`. (d) `Maxscore2`.

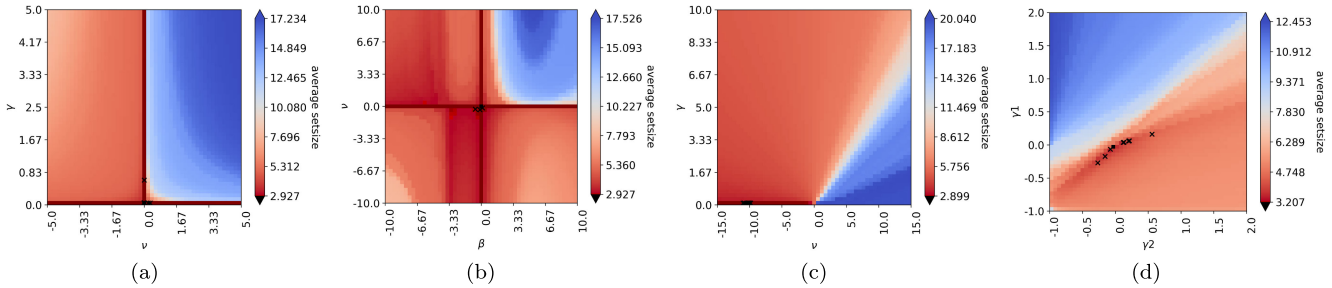

FIG. 8. Example grid plots for four different nonconformity measures with SciPy optimizations (black cross), minimizing the average set size. The definition of each nonconformity measure is given in Table I. (a) `Entropy`. (b) `Entropy softmax`. (c) `Margin2`. (d) `Maxscore2`.

3. Results: Average set size

Repeating the optimization procedure for minimizing the average set size, the results are shown in Fig. 8. Similar to the F_1 score, we again find that the `entropy`, `entropy softmax`, and `maxscore2` nonconformity measures optimize to the `baseline` measure, and that the `entropy` and `entropy softmax` measures are degenerate along the zero-axes when no regularization is used. The `margin2` measure optimizes to $\gamma = 0$, $\nu = -11.9 \pm 3.5$, for which the average set size is slightly larger but comparable to the `baseline` measure.

Although not equivalent, the results from using the F_1 score and the average set size metrics are very similar, suggesting that the metrics are strongly correlated.

D. Comparison of nonconformity measures

Having optimized each nonconformity measure, we can now compare their performance on the evaluation dataset. The results from optimizing all of our nonconformity measures for the three chosen metrics are summarized in Table II. The optimal parameters obtained are the mean values over five optimization runs with different calibration/test data splits of the optimization dataset. The uncertainties are the standard deviations over these runs. The plots shown and discussed in Sec. III C each represent the first of these five optimization runs. The evaluation scores are calculated using the optimized parameters and the evaluation dataset. The uncertainties in the evaluation scores stem from the uncertainty of the parameter optimization.

For the F_1 score, as noted in the previous section, most of the nonconformity measures optimize to parameters that reduce them to the baseline. The exceptions are the `softmax`, `margin2`, and `Brier` nonconformity measures, which do not reduce to the `baseline` measure but achieve F_1 scores comparable to the `baseline`.

Using singletons, we find that the baseline is no longer the best and that all other nonconformity measures perform slightly better, with `maxscore2` giving the highest number. However, the evaluation scores are all very similar, and the nonzero standard deviations on the parameters indicate that there is still some statistical uncertainty.

Using the average set size, the behavior of the nonconformity measures is similar to using F_1 scores.

TABLE II. Optimization results at $\alpha = 0.1$. To estimate uncertainties, we calculate the standard deviations in the fitted parameters over multiple runs on different data splits. However, in some cases, all runs produce identical results for the fitted parameters; in these cases, we instead take a conservative estimate of the uncertainty by giving the bin size from gridding.

	Optimized parameters			Evaluation scores		
Nonconformity measure	F_1 score	Singletons	Set size	F_1 score	Singletons	Set size
`Baseline`	...	...	...	0.456	0.179	4.263
`Softmax`	$\beta = 0.001 \pm 0.001$	$\beta = 0.51 \pm 0.39$	$\beta = 0.001 \pm 0.001$	0.455 ± 10^{-4}	0.261 ± 0.006	4.324 ± 0.026
`Entropy`	$\gamma = 0 \pm 0.01$	$\gamma = 0.14 \pm 0.05$	$\gamma = 0 \pm 0.01$	0.456 ± 0.003	0.267 ± 0.003	4.263 ± 0.04
	$\nu = 0 \pm 0.01$	$\nu = -0.2 \pm 0.01$	$\nu = 0 \pm 0.01$			
`Entropy softmax`	$\nu = 0 \pm 0.01$	$\nu = 1.8 \pm 0.2$	$\nu = 0 \pm 0.01$	0.456 ± 10^{-6}	0.237 ± 0.007	4.263 ± 10^{-4}
	$\beta = 0 \pm 0.01$	$\beta = -6.5 \pm 0.4$	$\beta = 0 \pm 0.01$			
`Maxscore2`	$\gamma_1 = 0 \pm 0.01$	$\gamma_1 = -0.4 \pm 0.2$	$\gamma_1 = 0 \pm 0.01$	0.456 ± 0.006	0.271 ± 0.002	4.263 ± 0.074
	$\gamma_2 = 0 \pm 0.01$	$\gamma_2 = -0.4 \pm 0.2$	$\gamma_2 = 0 \pm 0.01$			
`Margin2`	$\gamma = 0 \pm 0.01$	$\gamma = 0.2 \pm 0.01$	$\gamma = 0 \pm 0.01$	0.455 ± 10^{-6}	0.264 ± 0.002	4.276 ± 0.002
	$\nu = -8.8 \pm 1.5$	$\nu = -7.4 \pm 3.8$	$\nu = -11.9 \pm 3.5$			
`CNN`	$\gamma = 1.0 \pm 0.001$	$\gamma = 0.96 \pm 0.04$	$\gamma = 1.0 \pm 0.001$	0.456 ± 0.006	0.266 ± 0.006	4.263 ± 0.075
`Brier`	$\nu = 1.001 \pm 0.002$	$\nu = 1.012 \pm 0.015$	$\nu = 1.002 \pm 0.001$	0.448 ± 0.005	0.263 ± 0.006	4.671 ± 0.011

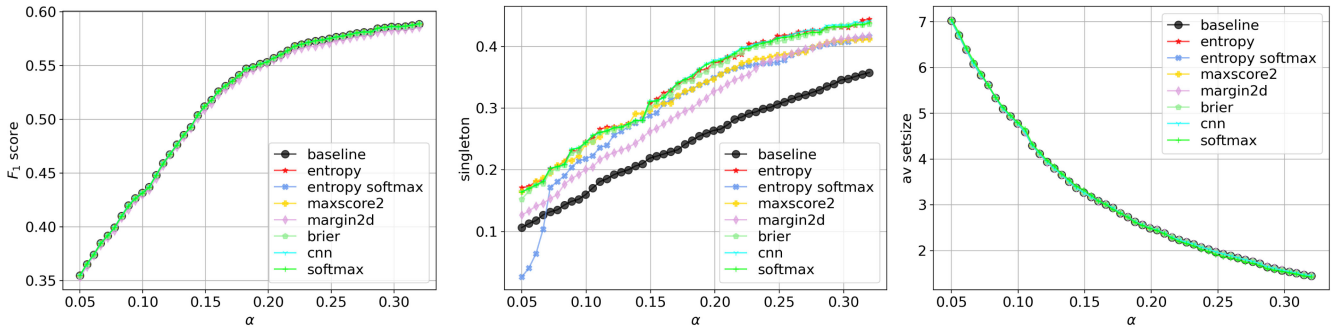

FIG. 9. Comparison of all the discussed nonconformity measures, making use of different metrics. (a) F_1 score. (b) Singletons. (c) Average set size.

All nonconformity measures either reduce to the baseline for optimal parameters or return comparable average set sizes.

To explore if the optimization depends on the error rate α, we run repeated tests for varying α. The plots in Fig. 9 show the optimized nonconformity measures together with the `baseline` over varying error rates α. We note that, as expected, the results from all metrics vary with varying α. Furthermore, the relative performance of the different nonconformity measures depends on the value of α for the singleton metric only.

For the F_1 score and average set size metrics, all optimized nonconformity measures either reduce to or are comparable to the baseline measure. This is true regardless of the value of α, as shown in Figs. 9(a) and 9(c). The optimal parameters for each nonconformity measure were found for $\alpha = 0.1$, and these optimized parameters were then applied for other values of α. However, a brief investigation shows that optimizing at other values of α does not change the results significantly and that the baseline measure is still preferred.

As shown in both Fig. 9 and Table II, all optimized nonconformity measures considered return a higher average number of singletons than the baseline measures for all α. For this metric, the relative performance of nonconformity measures varies slightly with varying α, and most notably the `entropy softmax` measure performs significantly worse at lower α.

Our results thus show that the same nonconformity measures optimize and perform differently when considering different metrics. Hence, when applying *CP* to a new problem, it is worth considering not only which nonconformity measure will perform best but also which metric we are most interested in. For example, for our application of *CP* to Gravity Spy, if we value overall smaller uncertainties we should minimize the average set size to find our optimal nonconformity measure, but if we value uniquely classifying the glitches we should maximize the number of singletons. The F_1 score is a metric considering various aspects of the performance of *CP*. For our application, we find that the F_1 score results are similar to the average set size and thus the size of the prediction sets appears to have a greater impact than how often glitches are uniquely classified correctly. From the definition of the F_1 score in Eq. (7), it is evident that the number of incorrect classifications (FP and FN) increases with larger prediction sets. Therefore, we see that the F_1 score and average set size metric will be strongly correlated, explaining the similarity of results when used as a loss function.

E. Individual glitch classes

It is important to note that the results in the previous section are calculated for Gravity Spy and the ensemble of all glitches in our dataset and that other algorithms or other datasets for the same algorithm may perform differently.

As an example, we can treat each glitch class as an individual dataset and perform the same optimization. We find that for most of the glitch classes, the results agree with the results in Sec. III C, as expected. However, for some of the classes, other nonconformity measures return higher F_1 scores than the `baseline`. In Fig. 10, a few example plots where the F_1 score does not optimize to the `baseline` are shown.

We further note that the same nonconformity measure can behave very differently for different glitch datasets, as seen, for example, in comparing the plots in Figs. 10(c) and 10(d).

This demonstrates how the choice of nonconformity measure is problem-specific. Applying the analysis from this section to another dataset or problem setup may give different results.

The different glitch classes have varying Gravity Spy classification accuracies, with some classes being classified correctly more often than others. To investigate how the performance changes when applying *CP* to the individual glitch datasets, we optimize our nonconformity measures for each of these glitch-specific datasets individually. We can then calculate evaluation scores for each optimized nonconformity measure and glitch-specific dataset and plot them against the error rate for the respective glitch class (see Fig. 11). We find that a higher classification accuracy

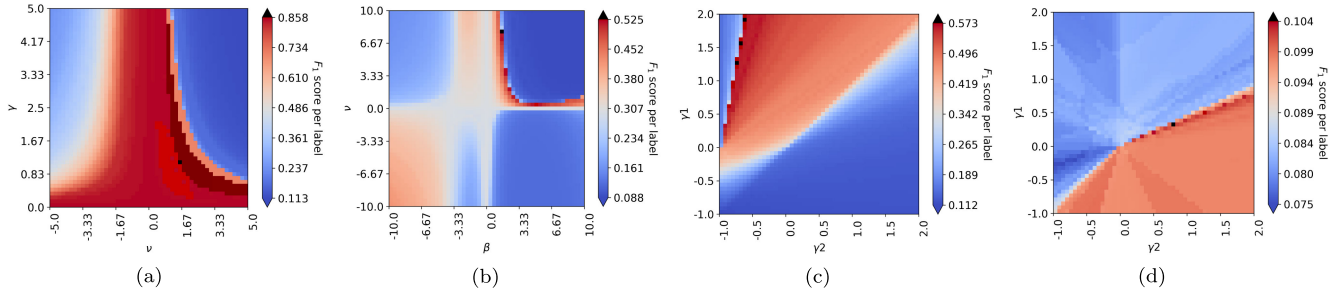

FIG. 10. Example grid plots of individual glitch classes using the F_1 score. The plots show that in these cases, the nonconformity measures do not optimize to the `baseline`. (a) `Entropy`, `Koi Fish`. (b) `Entropy softmax`, `Low Frequency Burst`. (c) `Maxscore2`, `Low Frequency Burst`. (d) `Maxscore2`, `Wandering Line`.

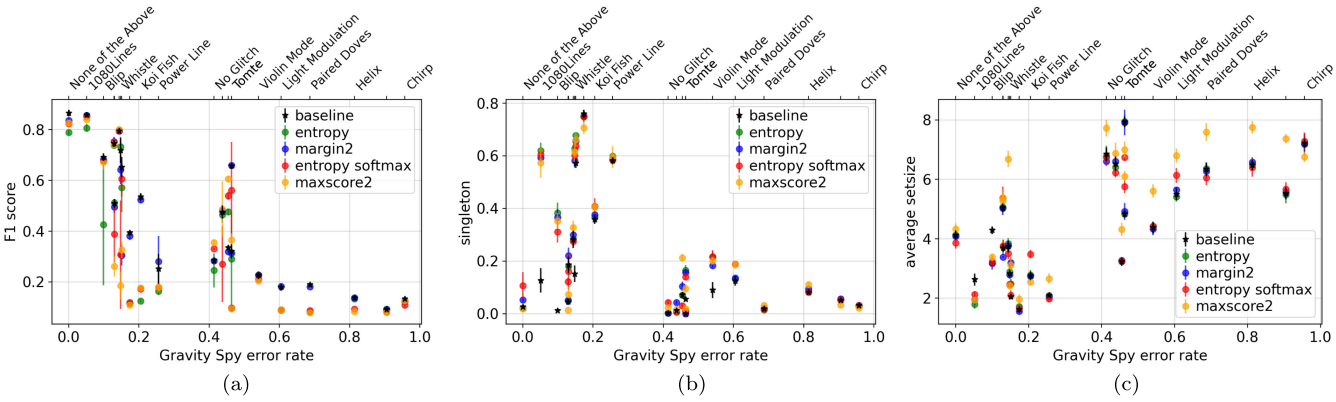

FIG. 11. The plots show the GS error rate of individual glitch class datasets versus label-specific scores for all metrics for a few of the discussed nonconformity measures. Each error rate point on the x-axis corresponds to one glitch class, and a few of the classes are named in the top axis (we do not show all for clarity). The nonconformity measures have been optimized for each glitch class individually. The error bars represent the standard deviation obtained from repeating the evaluation calculation for different splits of each dataset. (a) F_1 score. (b) Singletons. (c) Average set size.

(lower Gravity Spy error rate) for a glitch class implies that the dataset will generally achieve higher overall F_1 scores, see Fig. 11(a), and smaller average prediction set sizes, see Fig. 11(c). The correlation for the singleton metric is less clear, as shown in Fig. 11(b).

Furthermore, the plots in Fig. 11 confirm the previous discussion stating that different metrics are preferred for different glitch datasets. However, there does not seem to be any correlation between the classification accuracy of a glitch class and which nonconformity measure is preferred for that dataset.

IV. DISCUSSION AND CONCLUSION

In this work, we have demonstrated the application of *CP* to the Gravity Spy glitch classification ML algorithm. We have discussed properties of *CP*, such as its guaranteed validity, and particularly focused on the nonconformity measure.

The nonconformity measure is a key element of *CP* that transforms the heuristic output of the underlying algorithm into a rigorous uncertainty. Since the optimal choice of nonconformity measure is not theoretically predicted, we parametrized families of possible measures and then investigated the choice of three metrics, the F_1 score, average prediction set size, and number of singletons, in optimizing the nonconformity parameters. We have applied this to our Gravity Spy test case, but the methodology could be applied in general to identify optimal nonconformity measures.

For the F_1 score and average set size metrics, the results showed that the simple, and most common, `baseline` nonconformity measure returned the best scores and that, even after optimizing, none of the other nonconformity measures were better. In fact, all nonconformity measures that reduced to the `baseline` measure for certain parameter values were optimized for these parameters. Meanwhile, all optimized nonconformity measures we have considered returned a higher number of singletons than the `baseline` measure. Hence, our results showed that the choice of nonconformity measure should depend on the metric of interest. For example, if the overall

certainty of predictions (small prediction set sizes) is important, we should choose the `baseline` measure. However, if creating a set of correctly classified glitches of a certain class is the aim (maximized singletons), the `maxscore2` measure will return the best result for our Gravity Spy test application.

Furthermore, considering the F_1 scores for individual glitch class datasets, the `baseline` measure is not always the best, and some of the nonconformity measures optimize to parameters that do not reduce them to the `baseline`. We can thus conclude that choosing the optimal nonconformity measure is complicated and depends both on the algorithm *CP* is applied to as well as the dataset used and the metric of interest.

Knowing that different nonconformity measures are preferred for the different glitch class datasets, one option to further improve performance for the full dataset would be to mix different measures for different classes, such that each class uses the preferred nonconformity measure. Alternatively, adopting the measure that works best overall is a solution. As we have shown in this work, the `baseline` measure gave the best results for the F_1 score for the overall dataset and was also most often preferred for the class-specific datasets for this metric. This confirmed the intuition that if a nonconformity score is good for the majority of smaller datasets, it will also be good for the full dataset.

Comparing our results for the `baseline`, `CNN`, and `Brier` nonconformity measures to similar work in the literature, we confirmed that the optimal nonconformity measure is dependent on the application. In Ref. [28], the authors found that the `CNN` nonconformity measure returned the highest number of singletons for $\gamma = 0$ while the smallest average set size was found for $\gamma = 1$ for their application of a `CNN` to face and object recognition databases. For our Gravity Spy application, we also found that the `CNN` nonconformity measure was optimized for $\gamma = 1$ for the average set size metric, however, for the singleton metric, we found $\gamma = 0.95$, which differs from the result in [28]. Similar to our results, the authors of Ref. [29] found that the `baseline` nonconformity measure produced a smaller average set size than the `brier` measure, while the `brier` measures returned a higher number of singletons than the `baseline` for their application, confirming again that the optimal choice of nonconformity measure depends on the preferred metric.

To address the significance of our results, we first observed that the optimization we applied improved the individual performance of all our nonconformity measures for all metrics, as seen from the plots in Sec. III C. Secondly, the differences between the evaluation scores for the optimized nonconformity measures were small for all metrics, and one could thus argue that choosing one above another would only minimally affect the outcome. Hence, our results on the full Gravity Spy dataset could be taken to justify the use of the `baseline` measure for all metrics. However, investigating the example plots for individual glitch class datasets in Fig. 10 showed that the difference in F_1 scores between where the measures were optimized and where they reduced to the `baseline` were no longer as small [for example, the difference in $F_1 \in [0, 1]$ score was ≈ 0.2 in Fig. 10(b)]. In these cases, other nonconformity measures returned notably better results, and using the `baseline` would be less justified.

While the optimized values found in this paper are only applicable to the Gravity Spy glitch classification algorithm and the dataset we have used, the methods we have described in this paper are applicable to any point-prediction algorithm. *CP* can be used to add uncertainty to classification algorithms in the manner described in this paper, but can also be used to add confidence intervals to ML regression algorithms [26], where it is otherwise difficult to obtain uncertainty quantifications. The method for the regression case is almost identical to the classification case, with the main difference being a different choice of nonconformity measure.

While uncertainties are important in themselves, *CP* can also be used to compare algorithm performances quantitatively, or to optimally combine the output from multiple algorithms. For example, *CP* together with the quantification metrics discussed in this paper could be used to investigate which algorithm performs a given task better, or if a combination of algorithms gives the optimal result. By defining a metric based on the prediction sets (such as those discussed in this paper), *CP* can be applied to each algorithm, and the most optimal one for the given task and chosen metric can be determined (similarly to how we have found optimal nonconformity measures in this work). It is also possible to use *CP* to improve the underlying algorithm itself [49].

To continue this work, *CP* could be built into or around the Gravity Spy algorithm for future analysis, providing point predictions with uncertainties. This idea also extends to other gravitational wave applications, for example, searches for compact binary coalescence signals [27], or could be applied to newly evolving ML methods [50,51]. *CP* can be used to combine search pipelines to improve performance or provide uncertainties for a ML-based parameter estimation algorithm. In summary, there are a multitude of applications, for gravitational wave science and otherwise, where *CP* can be useful.

The code for Sec. II is openly available from the Zenodo repository [30].

ACKNOWLEDGMENTS

We want to thank Christopher Berry, Zoheyr Doctor, Ryan Fisher, Siddharth Soni, and Michael Zevin from the

Gravity Spy team for their help using the Gravity Spy code and datasets and for early discussions of this work. This material is based upon work supported by NSF's LIGO Laboratory, which is a major facility fully funded by the National Science Foundation. The authors are grateful for computational resources provided by the LIGO Laboratory and supported by National Science Foundation Grants No. PHY-0757058 and No. PHY-0823459.

DATA AVAILABILITY

The data that support the findings of this article are openly available [30,34].

[1] Benjamin P. Abbott, Richard Abbott, T. D. Abbott, M. R. Abernathy, Fausto Acernese, Kendall Ackley, Carl Adams, Thomas Adams, Paolo Addesso, Rana X. Adhikari *et al.*, Observation of gravitational waves from a binary black hole merger, Phys. Rev. Lett. **116**, 061102 (2016).

[2] Junaid Aasi, B. P. Abbott, Richard Abbott, Thomas Abbott, M. R. Abernathy, Kendall Ackley, Carl Adams, Thomas Adams, Paolo Addesso, R. X. Adhikari *et al.*, Advanced LIGO, Classical Quantum Gravity **32**, 074001 (2015).

[3] F. Acernese, M. Agathos, K. Agatsuma, D. Aisa, N. Allemandou, A. Allocca, J. Amarni, P. Astone, G. Balestri, G. Ballardin *et al.*, Advanced Virgo: A second-generation interferometric gravitational wave detector, Classical Quantum Gravity **32**, 024001 (2014).

[4] Yoichi Aso, Yuta Michimura, Kentaro Somiya, Masaki Ando, Osamu Miyakawa, Takanori Sekiguchi, Daisuke Tatsumi, Hiroaki Yamamoto (KAGRA Collaboration), Interferometer design of the KAGRA gravitational wave detector, Phys. Rev. D **88**, 043007 (2013).

[5] R. Abbott, H. Abe, F. Acernese, K. Ackley, S. Adhicary, N. Adhikari *et al.*, Open data from the third observing run of LIGO, Virgo, KAGRA, and GEO, Astrophys. J. Suppl. Ser. **267**, 29 (2023).

[6] Benjamin P. Abbott, Rich Abbott, Thomas D. Abbott, Sheelu Abraham, Fausto Acernese, Kendall Ackley, Carl Adams, Vaishali B. Adya, Christoph Affeldt, Michalis Agathos *et al.*, A guide to LIGO–Virgo detector noise and extraction of transient gravitational-wave signals, Classical Quantum Gravity **37**, 055002 (2020).

[7] Derek Davis and Marissa Walker, Detector characterization and mitigation of noise in ground-based gravitational-wave interferometers, Galaxies **10**, 12 (2022).

[8] Chris Pankow, Katerina Chatziioannou, Eve A. Chase, Tyson B. Littenberg, Matthew Evans, Jessica McIver, Neil J. Cornish, Carl-Johan Haster, Jonah Kanner, Vivien Raymond, Salvatore Vitale, and Aaron Zimmerman, Mitigation of the instrumental noise transient in gravitational-wave data surrounding GW170817, Phys. Rev. D **98**, 084016 (2018).

[9] Richard Abbott, T. D. Abbott, F. Acernese, K. Ackley, C. Adams, N. Adhikari, R. X. Adhikari, V. B. Adya, C. Affeldt, D. Agarwal *et al.*, GWTC-3: Compact binary coalescences observed by LIGO and Virgo during the second part of the third observing run, Phys. Rev. X **13**, 041039 (2023).

[10] Benjamin P. Abbott, R. Abbott, T. D. Abbott, S. Abraham, Fausto Acernese, K. Ackley, C. Adams, V. B. Adya, C. Affeldt, M. Agathos *et al.*, Prospects for observing and localizing gravitational-wave transients with Advanced LIGO, Advanced Virgo and KAGRA, Living Rev. Relativity **23**, 1 (2020).

[11] Derek Davis, T. B. Littenberg, I. M. Romero-Shaw, M. Millhouse, J. McIver, F. Di Renzo, and G. Ashton, Subtracting glitches from gravitational-wave detector data during the third LIGO-Virgo observing run, Classical Quantum Gravity **39**, 245013 (2022).

[12] Derek Davis, Joseph S. Areeda, Beverly K. Berger, R. Bruntz, Anamaria Effler, R. C. Essick, R. P. Fisher, Patrick Godwin, Evan Goetz, A. F. Helmling-Cornell *et al.*, LIGO detector characterization in the second and third observing runs, Classical Quantum Gravity **38**, 135014 (2021).

[13] S. Soni, B. K. Berger, D. Davis, F. Di Renzo, A. Effler, T. A. Ferreira, J. Glanzer, E. Goetz, G. González, A. Helmling-Cornell *et al.*, LIGO detector characterization in the first half of the fourth observing run, arXiv:2409.02831.

[14] Philippe Nguyen, R. M. S. Schofield, Anamaria Effler, Corey Austin, Vaishali Adya, Matthew Ball, Sharan Banagiri, Katherine Banowetz, C. Billman, C. D. Blair *et al.*, Environmental noise in Advanced LIGO detectors, Classical Quantum Gravity **38**, 145001 (2021).

[15] Robert E. Colgan, K. Rainer Corley, Yenson Lau, Imre Bartos, John N. Wright, Zsuzsa Márka, and Szabolcs Márka, Efficient gravitational-wave glitch identification from environmental data through machine learning, Phys. Rev. D **101**, 102003 (2020).

[16] J. Glanzer, S. Banagiri, S. B. Coughlin, S. Soni, M. Zevin, Christopher Philip Luke Berry, O. Patane, S. Bahaadini, N. Rohani, K. Crowston *et al.*, Data quality up to the third observing run of Advanced LIGO: Gravity Spy glitch classifications, Classical Quantum Gravity **40**, 065004 (2023).

[17] Ruxandra Bondarescu, Andrew Lundgren, and Ronaldas Macas, Antiglitch: A quasi-physical model for removing short glitches from LIGO and Virgo data, arXiv:2309.06594.

[18] Melissa Lopez, Vincent Boudart, Kerwin Buijsman, Amit Reza, and Sarah Caudill, Simulating transient noise bursts in LIGO with generative adversarial networks, Phys. Rev. D **106**, 023027 (2022).

[19] Tiago Fernandes, Samuel Vieira, Antonio Onofre, Juan Calderón Bustillo, Alejandro Torres-Forné, and José A. Font, Convolutional neural networks for the classification of glitches in gravitational-wave data streams, Classical Quantum Gravity **40**, 195018 (2023).

[20] Michael Zevin, Scott Coughlin, Sara Bahaadini, Emre Besler, Neda Rohani, Sarah Allen, Miriam Cabero, Kevin

Crowston, Aggelos K. Katsaggelos, Shane L. Larson *et al.*, Gravity Spy: Integrating Advanced LIGO detector characterization, machine learning, and citizen science, Classical Quantum Gravity **34**, 064003 (2017).

[21] Massimiliano Razzano, Francesco Di Renzo, Francesco Fidecaro, Gary Hemming, and Stavros Katsanevas, GWitchHunters: Machine learning and citizen science to improve the performance of gravitational wave detector, Nucl. Instrum. Methods Phys. Res., Sect. A **1048**, 167959 (2023).

[22] Yunan Wu, Michael Zevin, Christopher P. L. Berry, Kevin Crowston, Carsten Østerlund, Zoheyr Doctor, Sharan Banagiri, Corey B. Jackson, Vicky Kalogera, and Aggelos K. Katsaggelos, Advancing glitch classification in Gravity Spy: Multi-view fusion with attention-based machine learning for Advanced LIGO's fourth observing run, arXiv:2401.12913.

[23] Shourov Chatterji, Lindy Blackburn, Gregory Martin, and Erik Katsavounidis, Multiresolution techniques for the detection of gravitational-wave bursts, Classical Quantum Gravity **21**, S1809 (2004).

[24] Vladimir Vovk, Alexander Gammerman, and Glenn Shafer, *Algorithmic Learning in a Random World* (Springer, New York, 2005), Vol. 29.

[25] Anastasios N. Angelopoulos and Stephen Bates, A gentle introduction to conformal prediction and distribution-free uncertainty quantification, arXiv:2107.07511.

[26] Glenn Shafer and Vladimir Vovk, A tutorial on conformal prediction, J. Mach. Learn. Res. **9**, 371 (2008), http://www.jmlr.org/papers/v9/shafer08a.html.

[27] Gregory Ashton, Nicolo Colombo, Ian Harry, and Surabhi Sachdev, Calibrating gravitational-wave search algorithms with conformal prediction, Phys. Rev. D **109**, 123027 (2024).

[28] Sergio Matiz and Kenneth E. Barner, Inductive conformal predictor for convolutional neural networks: Applications to active learning for image classification, Pattern Recogn. **90**, 172 (2019).

[29] Ulf Johansson, Henrik Linusson, Tuve Löfström, and Henrik Boström, Model-agnostic nonconformity functions for conformal classification, in *2017 International Joint Conference on Neural Networks (IJCNN)* (IEEE, 2017), pp. 2072–2079, 10.1109/IJCNN.2017.7966105.

[30] Ann-Kristin Malz, Gregory Ashton, and Nicolo Colombo, Example code for: "Classification uncertainty for transient gravitational-wave noise artefacts with optimised conformal prediction", 10.5281/zenodo.14066270 (2024).

[31] A. Gammerman, V. Vovk, and V. Vapnik, Learning by transduction, in *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence*, UAI'98 (Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1998), pp. 148–155, ISBN 155860555X.

[32] Tiffany Ding, Anastasios Angelopoulos, Stephen Bates, Michael Jordan, and Ryan J. Tibshirani, *Class-Conditional Conformal Prediction with Many Classes* (Curran Associates, Inc., New York, 2024), Vol. 36.

[33] Vladimir Vovk, Conditional validity of inductive conformal predictors, in *Asian Conference on Machine Learning* (Singapore Management University, Singapore, 2012), pp. 475–490.

[34] Michael Zevin, Scott Coughlin, Eve Chase, Sara Allen, Sara Bahaadini, Christopher Berry *et al.*, Gravity Spy volunteer classifications of LIGO glitches from observing runs O1, O2, O3a, and O3b, https://zenodo.org/records/5911227 (2022).

[35] Michael Zevin, Corey B. Jackson, Zoheyr Doctor, Yunan Wu, Carsten Østerlund, L. Clifton Johnson, Christopher P. L. Berry, Kevin Crowston, Scott B. Coughlin, Vicky Kalogera *et al.*, Gravity Spy: Lessons learned and a path forward, Eur. Phys. J. Plus **139**, 100 (2024).

[36] J. Glanzer, S. Banagiri, S. B. Coughlin, S. Soni, M. Zevin, Christopher Philip Luke Berry, O. Patane, S. Bahaadini, N. Rohani, K. Crowston *et al.*, Data quality up to the third observing run of Advanced LIGO: Gravity Spy glitch classifications, Classical Quantum Gravity **40**, 065004 (2023).

[37] H. C. Thode, *Testing for Normality*, Statistics, Textbooks and Monographs (CRC Press, 2002), ISBN 9780203910894, https://books.google.se/books?id=gbegXB4SdosC.

[38] Jochen Bröcker and Leonard A. Smith, Increasing the reliability of reliability diagrams, Weather Forecast. **22**, 651 (2007).

[39] Volodya Vovk, Alex Gammerman, and Craig Saunders, Machine-learning applications of algorithmic randomness, In *International Conference on Machine Learning* (Morgan Kaufmann Publishers Inc., San Francisco, CA, 1999), pp. 444–453.

[40] Rui Luo and Nicolo Colombo, Entropy reweighted conformal classification, arXiv:2407.17377.

[41] Harris Papadopoulos, Volodya Vovk, and Alex Gammerman, Conformal prediction with neural networks, in *19th IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2007)* (IEEE, Patras, Greece, 2007), Vol. 2, pp. 388–395, 10.1109/ICTAI.2007.47.

[42] A. Humphrey, W. Kuberski, J. Bialek, N. Perrakis, W. Cools, N. Nuyttens, H. Elakhrass, and P. A. C. Cunha, Machine-learning classification of astronomical sources: Estimating F1-score in the absence of ground truth, Mon. Not. R. Astron. Soc. **517**, L116 (2022).

[43] Zachary C. Lipton, Charles Elkan, and Balakrishnan Naryanaswamy, Optimal thresholding of classifiers to maximize f1 measure, In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2014, Nancy, France, 2014. Proceedings, Part II 14* (Springer, New York, 2014), pp. 225–239, 10.1007/978-3-662-44851-9-15.

[44] Glen Cowan, *Statistical Data Analysis* (Oxford University Press, New York, 1998).

[45] Daniel Bonilla Betancourth, Manuela Bravo, Stephany Bonilla, Angela Iragorri, Diego Mendez, Iván Mondragón, Catalina Alvarado-Rojas, and Julian Colorado, Progressive rehabilitation based on EMG gesture classification and an MPC-driven exoskeleton, Bioengineering **10**, 770 (2023).

[46] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Reddy *et al.*, SciPy 1.0: Fundamental algorithms for scientific computing in Python, Nat. Methods **17**, 261 (2020).

[47] Richard H. Byrd, Peihuang Lu, Jorge Nocedal, and Ciyou Zhu, A limited memory algorithm for bound constrained optimization, SIAM J. Sci. Comput. **16**, 1190 (1995).

[48] Ciyou Zhu, Richard H. Byrd, Peihuang Lu, and Jorge Nocedal, Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound-constrained optimization, ACM Trans. Math. Softw. **23**, 550 (1997).

[49] David Stutz, Ali Taylan Cemgil, Arnaud Doucet *et al.*, Learning optimal conformal classifiers, arXiv:2110.09192.

[50] Vasileios Skliris, Michael R. K. Norman, and Patrick J. Sutton, Real-time detection of unmodelled gravitational-wave transients using convolutional neural networks, arXiv:2009.14611.

[51] Ethan Marx, William Benoit, Alec Gunny, Rafia Omer, Deep Chatterjee, Ricco C. Venterea, Lauren Wills, Muhammed Saleem, Eric Moreno, Ryan Raikman *et al.*, A machine-learning pipeline for real-time detection of gravitational waves from compact binary coalescences, arXiv:2403.18661.