Rapid identification of strongly lensed gravitational-wave events with machine learning

Srashti Goyal,[1] Harikrishnan D.,[1,2] Shasvath J. Kapadia,[1] and Parameswaran Ajith[1,3]

[1]*International Centre for Theoretical Science, Tata Institute of Fundamental Research, Bangalore 560089, India*

[2]*Plaksha Tech Leaders Fellowship, Plot 17, Sector 18, Udyog Vihar, Gurugram, Haryana 122022, India*

[3]*Canadian Institute for Advanced Research, CIFAR Azrieli Global Scholar, MaRS Centre, West Tower, 661 University Ave, Toronto, Ontario M5G 1M1, Canada*

(Received 15 July 2021; accepted 19 November 2021; published 20 December 2021)

A small fraction of the gravitational-wave signals that will be detected by second and third generation detectors are expected to be strongly lensed by galaxies and clusters, producing multiple observable copies. While optimal Bayesian model selection methods are developed to identify lensed signals, processing tens of thousands (billions) of possible pairs of events detected with second (third) generation detectors is both computationally intensive and time consuming. To mitigate this problem, we propose to use machine learning to rapidly rule out a vast majority of candidate lensed pairs. As a proof of principle, we simulate nonspinning binary black hole events added to Gaussian noise, and train the machine on their time-frequency maps (Q transforms) and localization skymaps (using Bayestar), both of which can be generated in seconds. We show that the trained machine is able to accurately identify lensed pairs with efficiencies comparable to existing Bayesian methods.

DOI: 10.1103/PhysRevD.104.124057

I. INTRODUCTION

With the tens of gravitational-wave (GW) signals detected by the LIGO-Virgo network of detectors [1,2] during its first three observing runs [3–8], there is no doubt that GW astronomy has well and truly arrived. While these detections have enabled stringent tests of Einstein's general relativity [9,10], future observing runs are likely to provide a number of additional tests. Among them is the highly anticipated observation of gravitationally lensed GWs [11–15], akin to the gravitational lensing of electromagnetic waves where the deflection of light from a source due to large agglomerations of matter (such as galaxies and galaxy clusters) residing along the line of sight of the observer produces multiple magnified (or demagnified) copies of the source. Apart from being a unique probe of general relativity's prediction of gravitational lensing with a different messenger [16], lensing of GWs could afford unique constraints on various aspects of astrophysics and cosmology, including models of the populations of galaxies [17], as well as models that probe the distribution and composition of dark matter [18].

Gravitational lensing observations are typically divided into three categories: strong lensing, weak lensing and microlensing (see, e.g., [19]). This classification is based on the properties of the lens, in particular the density of the lens projected along the plane perpendicular to the line of sight of the observer. In this work, we concern ourselves exclusively with strong lensing, where the projected density exceeds a critical density, resulting in the production of multiple resolvable images.

Note that this classification is done in the *geometric optics* limit, where the wavelength (of light or GWs) is much smaller than the characteristic gravitational radius of the lens. While this is almost always true in the lensing of light, this is not always the case for the lensing of GWs. In this work, we assume that the wavelength of the GWs is much smaller than the Schwarzchild radius of the lenses, as is the case when GWs from coalescing stellar-mass binary black holes are lensed by galaxies or galaxy clusters. In this limit, strongly lensed GWs will result in the production of potentially resolvable images.

The n of images in the sky is ultimately dependent on the resolution of the telescopes that observe these images. GW detectors typically have very poor angular resolution [20,21] (at least in comparison to optical telescopes); the localization skyarea for GW events detected by the LIGO-Virgo network in the second and third observing runs spanned tens of square degrees at best [3]. As a result, even strongly lensed GW events typically have images whose skyareas almost completely overlap each other. Indeed, one of the signatures that two GW events are lensed copies is that their skymaps overlap (see, e.g., [22,23]).

While strongly lensed GW events are completely unresolvable in the sky with current GW detectors, they are typically very well resolved in time. Indeed, the temporal resolution of GW events (miliseconds) is in general orders

of magnitude smaller than the expected time delay (minutes to weeks) between strongly lensed GW images. In the geometric optics limit, these GW images would have different amplitudes, but their phase evolution would be identical [14,15,24–27]. Thus, in principle, determining whether two nonoverlapping GW events are lensed copies comes down to comparing the shapes of these signals with respect to each other.

In practice, however, such a comparison is nontrivial. Firstly, the observed GW signals are projections of the true GW signals onto the detectors; this projection depends on the location and orientation of the detector relative to the source, and would therefore be different for each of the temporally separated GW images. Furthermore, these images would be buried in detector noise. Even if the noise is assumed to be Gaussian and the corresponding power spectral density (PSD) is assumed to be time invariant, each of the images would be buried in different realizations of this noise.

A robust alternative to such a direct comparison of the GW signals is to work in the space of the inferred source parameters. Using optimal matched-filter based parameter inference techniques [28], Bayesian posterior distributions on the intrinsic parameters of the source (the masses and spins of the binary) and its extrinsic parameters (the skylocation of the binary) can be constructed. As mentioned earlier, the phase evolution of the GW images are expected to be identical, and therefore comparing the inferred posteriors on the intrinsic parameters (which completely govern the phase evolution) of pairs of GW events should enable us to discriminate between lensed and unlensed pairs. This discriminatibility can be further enhanced by comparing the localization skymaps which are expected to overlap almost entirely for lensed GW pairs [22].

Quantitatively, such a comparison can be achieved using Bayesian model selection [22,29]. A Bayes factor derived from the overlap between the posteriors of pairs of events can be constructed and used to segregate these pairs as either lensed or unlensed. However, evaluating this discriminator is computationally expensive and time consuming. Bayesian parameter inference of binary black-hole (BBH) events can take hours to days. Additionally, constructing the Bayes factor can take up to a few minutes per event, and the number of such evaluations will grow as the square of the number GW events. This makes the estimation of the Bayes factor computationally challenging when large numbers of BBH events are expected to be detected in future observing runs.

Current estimates of the rate of stellar-mass BBH mergers [30] suggest that hundreds of BBH events are expected to be detected in LIGO-Virgo-Kagra's next observing run (O4). Among these GW detections, up to a percent could be lensed copies of each other [31,32], suggesting that there is a non-trivial chance that the first confirmed detection of a lensed GW pair could occur in O4. However, identifying such lensed pairs would require constructing $\mathcal{O}(10^2)$ posteriors on the GW events' source-parameters and $\mathcal{O}(10^4)$ Bayes factors.

These numbers will get significantly larger with observing runs beyond O4, and astronomically large by the time the third generation (3G) network of ground-based detectors [33–35] completes its observations. The 3G network is expected to observe $\mathcal{O}(10^5$–$10^6)$ events, of which $\sim 0.3\%$ could be strongly lensed [32]. Therefore, $\mathcal{O}(10^5$–$10^6)$ event posteriors, and $\mathcal{O}(10^{10}$–$10^{12})$ Bayes factors, would need to be evaluated.

This motivates the need to come up with a method to conduct a preliminary segregation of pairs of GW events to rapidly "weed out" the vast majority of unlensed pairs. In this work, we propose to use machine learning algorithms, trained on time-frequency maps of the detector strain time series [36] and the (rapidly estimated) localization skymaps [37], from both lensed and unlensed pairs of GW events, to construct a statistic to discriminate between lensed and unlensed pairs. Using synthetic, nonspinning BBH signals—both lensed and unlensed—injected in Gaussian noise, we show that our machine-learning-based statistic, performs almost as well as the optimal Bayes factor statistic described above, while reducing the computation time by orders of magnitude. The significant reduction in evaluation time is a direct consequence of the fact that time-frequency maps and localization skymaps can be constructed in seconds, in contrast to GW inference posteriors which take hours to days to sample.

The rest of this paper is organized as follows. Section II summarizes the evaluation of the optimal Bayes factor statistic, introduces the machine learning algorithms we use, and delineates their training and validation. Section III describes our results in distinguishing between lensed and unlensed GW event pairs and compares them with the performance of the posterior overlap statistic. Section IV summarizes this work and discusses its potential benefits.

II. METHOD

A. The posterior overlap statistic

Let $d(t)$ be the detector strain time series which is known to contain a gravitational wave signal $h(t,\vec{\theta})$ with shape (intrinsic and extrinsic) parameters $\vec{\theta}$, as well as one realization of stochastic Gaussian noise as characterized by its power spectral density $S_n(f)$. A Bayesian inference of $\vec{\theta}$ from $d(t)$ can be achieved by sampling the posterior distribution on $\vec{\theta}$:

$$p(\vec{\theta}|d) = \frac{p(\vec{\theta})p(d|\vec{\theta})}{p(d)}, \tag{2.1}$$

where [38]

$$p(d|\vec{\theta}) \propto \exp\left[-(d-h|d-h)/2\right] \tag{2.2}$$

is the Gaussian likelihood, $p(\vec{\theta})$ is the prior distribution on the source parameters, $p(d)$ is the evidence and $(\cdot|\cdot)$ symbolizes the noise-weighted inner product:

$$(a|b) \equiv 2\int_{f_{\min}}^{f_{\max}} \frac{\tilde{a}(f)\tilde{b}^*(f)}{S_n(f)}df. \tag{2.3}$$

Here, $\tilde{a}, \tilde{b}$ represent the Fourier transform of the time series $a(t)$, $b(t)$; $[f_{\min}, f_{\max}]$ is the frequency range over which the inner product is evaluated, and * represents complex conjugation.

Now consider two segments of data, $d_1(t)$ and $d_2(t)$, both of which are known to contain one GW signal each, $h_1(t)$ and $h_2(t)$, respectively. We now wish to determine which of the two hypotheses, $\mathcal{H}_L$ and $\mathcal{H}_U$, is preferred by the data at hand.

$\mathcal{H}_L$ is the hypothesis that $h_1(t)$ and $h_2(t)$ are lensed copies of a GW signal originating from a single source. On the other hand, $\mathcal{H}_U$ is the hypothesis that $h_1(t)$ and $h_2(t)$ are signals originating from two distinct, unrelated, sources.

As shown in [22] (in the absence of any prior knowledge of which of the hypotheses is preferred), the optimal Bayesian statistic to quantitatively determine the preferred hypothesis is the Bayes factor $\mathcal{B}^L_U$, defined as the ratio of the evidences of the joint dataset $\{d_1, d_2\}$ given each of the hypotheses.

$$\mathcal{B}^L_U \equiv \frac{p(\{d_1,d_2\}|\mathcal{H}_L)}{p(\{d_1,d_2\}|\mathcal{H}_U)} = \int \frac{p(\vec{\theta}|d_1)p(\vec{\theta}|d_2)}{p(\vec{\theta})}d\vec{\theta}. \tag{2.4}$$

This Bayes factor can be evaluated making use of the posteriors $p(\vec{\theta}|d_1)$ and $p(\vec{\theta}|d_2)$ estimated from the two datasets d_1 and d_2, as well as the prior $p(\vec{\theta})$ employed in the parameter estimation.

B. Classification with machine learning

In the language of machine learning (ML), determining whether a pair of GW events are lensed copies of a single GW event, or unrelated (unlensed) to each other, is a binary classification problem. Using features derived from the data surrounding pairs of GW signals, we can in principle train an ML algorithm to classify them as either lensed or unlensed. In this subsection we first describe the construction of the features we use, the ML algorithms we employ, along with their training, testing and optimization.

1. Data representation

The posterior overlap statistic crucially relies on a time-consuming way of representing the detector data, viz., the posterior distributions of source parameters inferred from the data surrounding the confirmed GW detections. To bypass this issue, we construct and train a machine learning model which takes as inputs time-frequency maps (*Q transforms* of the GW event), as well as localization skymaps (*Bayestar Skymaps*). Both of these can be produced within seconds, in contrast to sampling the full posterior on the source parameters which can take anywhere from several hours to several days.

Q transforms.—Q transforms [36] are a means by which time-frequency maps of generic transient signals can be produced. This is achieved by first representing the time-frequency plane as a collection of tiles (bins), and then reconstructing these generic signals as a combination of sine Gaussians defined by their quality factor "Q." The choice of "Q" in each tile is determined from a matched-filter search across multiple "Q" templates, and the template that produces the largest SNR is selected. Using the corresponding optimal sine Gaussian, a spectrogram is generated. The time-frequency map is then plotted as colored tiles, where the color represents the so-called normalized signal energy, which is proportional to the Q-transform magnitude (and related to the SNR).

As shown in Fig. 1, lensed events will have time-frequency maps whose shapes are similar, but whose signal energies across time-frequency tiles will differ in magnitude. This is a direct consequence of the fact that the phase evolution of strongly lensed pairs are expected to be identical, but the amplitudes will differ by a constant factor. On the other hand, unlensed signals will have distinct time-frequency maps with dissimilar shapes in general.[1]

Bayestar Skymaps.—"Bayestar" [37] is the flagship low-latency skylocalization software of the LIGO-Virgo-Kagra collaboration, used during the LIGO-Virgo-Kagra's third observing run (O3) to disseminate skymaps in real time for the electromagnetic follow-up of GW events [40]. These skymaps are produced in seconds, and are found to be comparable to those estimated from a full sampling of the joint posterior distribution of the source parameters. Bayestar exploits the fact that errors in sky localization and the errors in the inference of the source masses are semi-independent. Given that this software is exclusively focused on providing localization skyareas, it exploits this semi-independence to drastically reduce the dimensionality of the parameter estimation problem by fixing the intrinsic parameter values to those of the maximum likelihood template in the matched filter search that identified the event. It is thus able to evaluate the (dimensionally reduced)

[1]A constant (additive) phase factor called the Morse phase, which is an integral multiple of $\pi/2$ depending on image type, will in general change the coalescence phase of the dominant GW mode [26,39]. Note that Q transforms are independent of coalescence phase, and are therefore unaffected by the Morse phase.

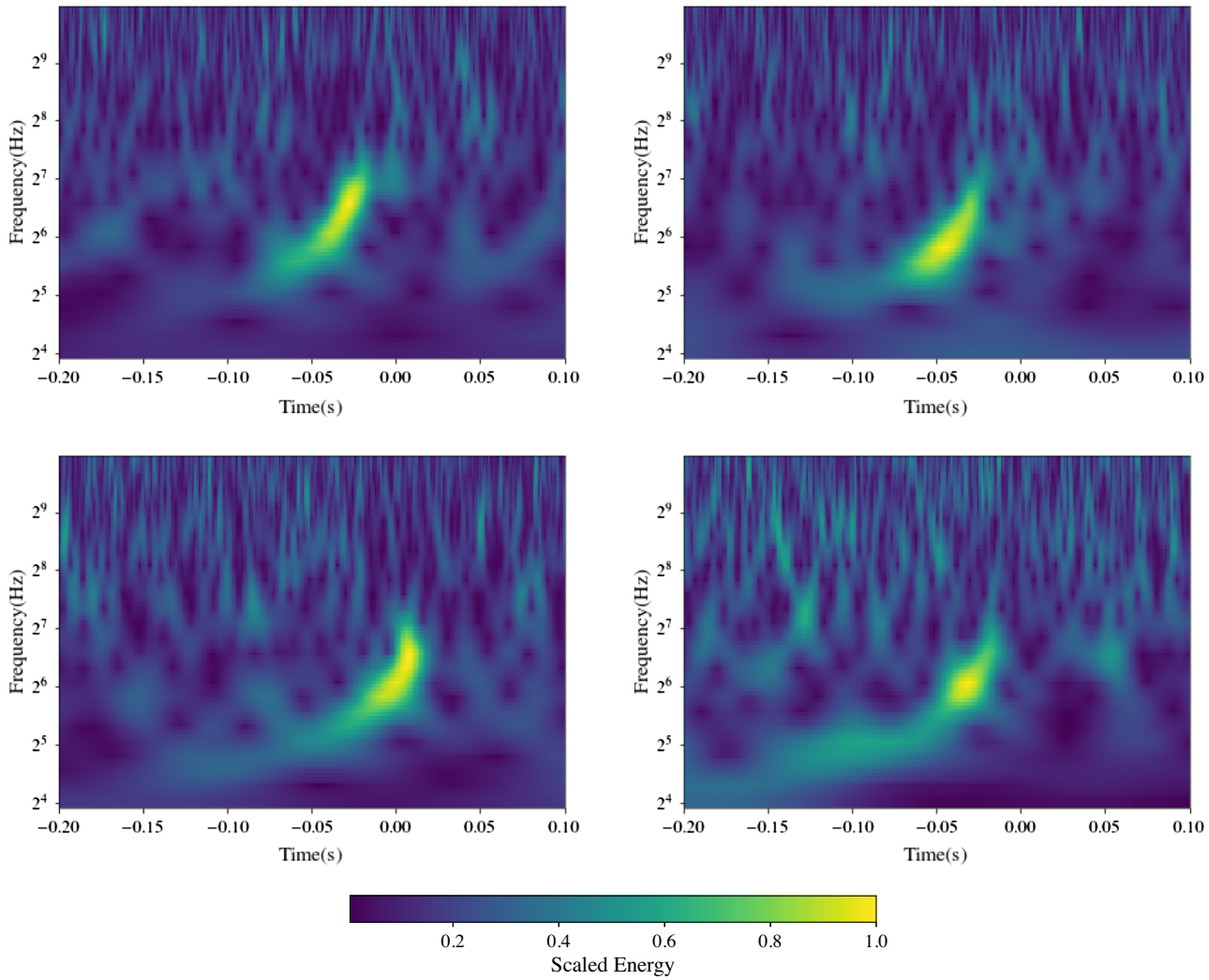

FIG. 1. Top panels: a pair of lensed GW events detected by the H1 (Hanford) inteferometer at design sensitvity. These events have time-frequency tracks with similar shapes. However, the signal energy in different time frequency bins along their tracks differ with respect to each other. Bottom panels: a pair of unlensed GW events projected detected by the H1 inteferometer at design sensitivity. These events have time-frequency tracks whose shapes are significantly different.

posterior on the extrinsic parameters rapidly, without significant loss in precision.

As shown in Fig. 2, lensed events are expected to have overlapping localization skyareas, by virtue of the poor [$\mathcal{O}(10)$ sq. deg.] angular resolution of ground based GW detectors with respect to the typical angular separation of the images [$\mathcal{O}(1'')$]. On the other hand, unlensed signals will generally have nonoverlapping skymaps.

2. Data preparation

In order to train, optimize and test our machine learning models, we simulate the lensed and unlensed GW signals and inject them in Gaussian noise. Our events consist of nonspinning binary black hole mergers detectable by the LIGO-Virgo network at design sensitivity, where detectability is defined by setting a threshold of 8 on the network SNR.

We follow [22] to generate a set of strongly lensed pairs of GW events, where the source BBH mergers follow a well-motivated distribution of masses and redshifts, and the lenses are assumed to be galaxies that can be modeled as singular isothermal ellipsoids whose parameters are drawn from the SDSS galaxy population catalog [41]. We generated ≈2800 detectable lensed event pairs and ≈1000 unrelated events, which corresponds to half a million unlensed pairs. We subdivide this set into two sets; we use one for training, and the other for validation. For testing, we use another, distinct, set, although the general prescription still follows [22].[2] This set consists of ≈300

[2]This dataset is chosen for testing because the posterior overlap statistic was already evaluated for the candidate pairs in this set (and reported in [22]), which allows for a ready comparison with the ML statistic.

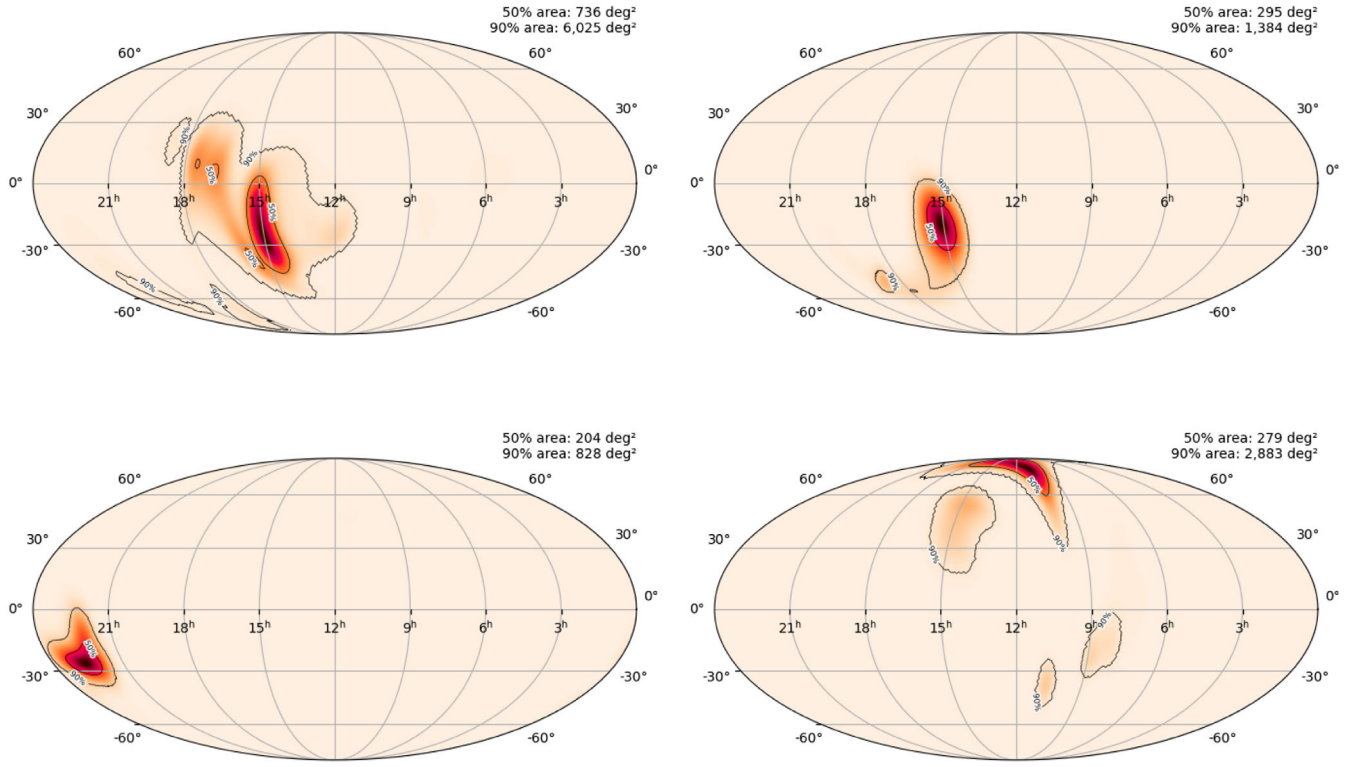

FIG. 2. Top panels: Bayestar skymaps of a pair of lensed events detected by the H1 (Hanford), L1 (Livingston), V1 (Virgo) network at design sensitivity. The skymaps of these events overlap. *Bottom panels*: Bayestar skymaps of a pair of unlensed events detected by the H1, L1, V1 network at design sensitivity. The skymaps of these events do not overlap.

lensed pairs and ≈1000 unrelated events (half a million unlensed pairs). From here on out, we refer to the training and validation dataset as "DSTV" and the testing set as "DST."

The waveforms are generated using the approximant IMRPhenomPv2 [42–44], as implemented in the LALSimulation module of the LALSuite software package [45]. The waveforms are then projected onto the LIGO and Virgo detectors using their antenna pattern functions, as implemented in the PYCBC [46] software package.

The detector noise is assumed to be Gaussian, and is generated using the zero-detuned high-power PSDs of Advanced LIGO and Advanced Virgo at their design sensitivities [47,48], as implemented in PYCBC. The projected waveforms are then added to the detector noise strain to produce the total detector strain time series.

From the time series surrounding each GW event, we generate Q-transform images for each detector. For events whose primary mass $m_1 > 60\ M_\odot$, we set the range of quality factors to (3, 7); otherwise, we set the range to (4, 10). Further, using the same time series, we use Bayestar to generate the localization skymaps for all the events.

3. Feature construction

Comparing the shapes of two time-frequency maps can be interpreted as a problem of image recognition, and therefore lends itself nicely to a ML analysis designed for such problems. Motivated by the fact that the Q-transform-based time-frequency maps of lensed pairs will have similar shapes (though different signal energies across time-frequency tiles), while unlensed pairs have dissimilar shapes in general, we superimpose the time-frequency maps of candidate pairs by aligning them along the time axis, which we pass to our ML algorithm.

On the other hand, while lensed pairs will have overlapping skymaps and unlensed pairs will not, the *shapes* of these maps are not in general expected to be the same, since the relative position of the two images with respect to the detectors are, in general, different (due to the rotation of the earth). However, GW events' localization skymaps are probability density functions in the space of right ascension (α) and declination (δ). Thus, a skymap can be thought of as a two-dimensional matrix where each element gives the probability density evaluated at a given pixel in the skymap's image grid spanning the space of (α, δ). The products of simple operations involving the matrices of candidate pairs can then be used as features that ML algorithms can employ to identify lensed events.

The Bayestar localization skymaps are usually generated in FITS format, which contains the skylocalization posterior information sampled over an adaptive HEALPix grid [49]. We project them to Cartesian coordinates using the HEALPY PYTHON library [50,51], which gives us the

localization posterior evaluated over a 400×800 rectangular grid of pixels corresponding to (α, δ) pairs. Denoting the skylocalization posteriors of each of the events pertaining to a candidate lensed pair as $P^1_{ij} = P(\alpha_i, \delta_j | d_1)$ and $P^2_{ij} = P(\alpha_i, \delta_j | d_2)$, we can construct the following metrics which can serve as features using which we can train an ML algorithm:

$$k_1 = \sum_i \sum_j P^1_{ij} P^2_{ij}, \qquad k_2 = \sum_i \sum_j |P^1_{ij} - P^2_{ij}|,$$
$$k_3 = \sqrt{\langle (P^1_{ij} P^2_{ij}))^2 \rangle - \langle k_1 \rangle^2}, \tag{2.5}$$

where k_1 is motivated by the posterior overlap statistic [22], k_2 is the absolute difference between the elements of the matrices and k_3 is a metric of the overlap between the skymaps, similar to the standard deviation. Note that angular brackets signify averaging over the total number of elements in each matrix.

4. Overall flow

For simplicity, we build two sets of ML models—one that learns from Q-transforms and another that is fed with skymaps—to classify the event pairs as either lensed and unlensed. The models employ two different ML algorithms—DenseNet201 [52] and XGBoost [53] (see Sec. II B 5).

The first set consists of three DenseNet201 ML models trained on superimposed QT (Q-transform) images of the event pairs for each of the three detectors: H1 (Hanford), L1 (Livingston) and V1 (Virgo), operating at their design sensitivities. We further construct an XGBoost model trained on the output of the DenseNet201 models. The output of this XGBoost model gives us the probability of the lensing hypothesis, given the Q-transform images: $P(\mathcal{H}_L|\text{QT1, QT2})$.[3]

We construct another XGBoost model trained on the metrics derived from pairs of lensed and unlensed Bayestar skymaps. The output of this XGBoost model gives us the probability of the lensing hypothesis, given the Bayestar skymaps: $P(\mathcal{H}_L|\text{SM1, SM2})$.

The final output of our ML classifier is then given by

$$P(\mathcal{H}_L|\{\text{QT1, QT2}\};\{\text{SM1, SM2}\}) = P(\mathcal{H}_L|\text{QT1, QT2}) \cdot P(\mathcal{H}_L|\text{SM1, SM2}). \tag{2.6}$$

We summarize the overall flow of our classification scheme in Fig. 3.

[3]A more complete notation for this probability would be as follows: $P(\mathcal{H}_L|\{\text{QT1-H1, QT2-H1}\};\{\text{QT1-L1, QT2-L1}\};\{\text{QT1-V1, QT2-V1}\})$. However, for notational simplicity, we omit the reference to the interferometers.

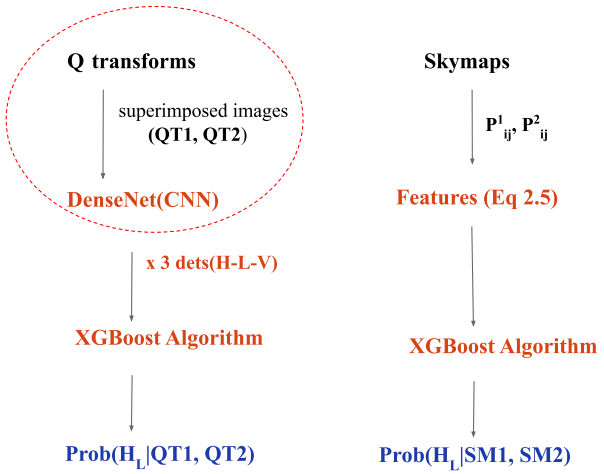

FIG. 3. A visual representation of the overall flow of our ML classification scheme. Note that, in principle, one could have avoided the step that trains a second XGBoost algorithm on features derived exclusively from the skymaps, and instead just used one XGBoost that jointly trains on features from the skymaps and the outputs of the DenseNet algorithms. We found that both methods give similar results. We therefore choose to include the additional XGBoost because it facilitates a stepwise analysis of the outputs of the individual components of the overall flow, trained separately on intrinsic and extrinsic parameters of the candidate pairs.

5. Machine-learning models

In this subsection, we briefly summarize the ML algorithms we useDenseNet201 and XGBoost.

DenseNet201.—A number of supervised machine learning algorithms exist for binary classification problems. However, only a relatively small subset of these are particularly suited for image recognition. Among them is the DenseNet ML [52] algorithm, which is a kind of convolutional neural network (CNN) with important improvements to mitigate problems that typically plague CNNs. A CNN, in turn, is a category of artificial neural networks (see, e.g., [54]) often used for classification problems that involve images, image recognition and computer vision (see, for example, [55]).

The basic architecture of a neural network consists of input/output layers of neurons, and a set of hidden layers in between [56]. Each neuron holds a number between in the range [0, 1]. An image passed to a neural network would fill the neurons of the input layer with values corresponding to the pixels of the image grid. The classification prediction of the neural network is recorded in the neurons of the output layer; specifically, in a binary classification problem such as ours, the output layer has one neuron representing the probability that the pair of superimposed Q transforms corresponds to the "lensed" case.

The neurons in each hidden layer are derived using a two step process. The first step involves a linear operation

between the vector of neurons $\vec{a}$ in the previous layer, and a matrix of weights $\overleftrightarrow{W}$, and the second a nonlinear operation that maps the output of the linear operation to numbers in the range [0, 1]:

$$\vec{a}_{n+1} = f(\overleftrightarrow{W}_n \cdot \vec{a}_n + \vec{b}_n). \tag{2.7}$$

Here, the nonlinear function f is referred to as the "activation function"; common choices include the "sigmoid function" and "the rectified linear unit" function (see, e.g., [57]). Further, the vector $\vec{b}$ is called the "bias." This process is applied iteratively until the output layer is filled.

Training the neural network ultimately comes down to determining an optimal choice of weights matrices and bias vectors. This can be achieved by feeding the neural network with labeled data, and penalizing the network's incorrect predictions using an appropriately defined cost function. The popular choice of cost function for binary classification is the binary cross entropy:

$$L_{\mathrm{CE}} = -[y \log(p) + (1-y)\log(1-p)], \tag{2.8}$$

where y is the ground-truth ("lensed" $= 1$ or "unlensed" $= 0$) of the labeled data, and p is the neural network's predicted value for a given choice of weights and biases. Minimizing the loss function averaged over multiple training instances with distinct labeled data, using gradient descent, provides the required weights and biases.

In CNNs, some of the hidden layers perform convolution operations between the previous layer, and appropriately chosen filters, in place of the operation described in Eq. (2.7). The filter can be thought of as a matrix whose size is usually smaller than the matrix of pixels input to the CNN. The convolution operation then involves "sliding" the filter across the pixel grid matrix, which mathematically amounts to taking the product of the filter with each of the submatrices of the pixel grid matrix. The resulting output is sometimes referred to as a "feature map."

A DenseNet is a type of deep CNN. In addition, its architecture has a few modifications to alleviate some of the problems commonly faced when using CNNs. DenseNet's are based on the observation that CNNs can be substantially deeper, more accurate, and computationally efficient to train if there are shorter connections between the layers close to the input and those close to the output. Thus, in a typical DenseNet model, for each layer, the feature maps of all preceding layers are used as inputs. Furthermore, the current layer's own feature map is used as input to all the subsequent layers. Because of this type of architecture, DenseNet models have several advantages compared to other CNN models. They greatly reduce the number of parameters that define the architecture of the neural network, mitigate the vanishing-gradient problem, encourage feature reuse and strengthen the feature propagation through the network.

XGBoost.—Extreme gradient boosting (XGBoost) [53] is a type of ensemble classifier that uses the combined output of a collection of trained decision trees to provide a probabilistic prediction of class membership to data that needs to be segregated into discrete categories. A decision tree, in turn, learns from training data by iteratively placing linear cuts in feature space which minimizes an appropriately chosen loss function. The repeated splits result in the segregated data being pushed down two separate branches at each leaf node in the tree, starting from the root node where the first split in the training data takes place, and ending at leaf nodes where a terminating criterion (e.g., minimum number of samples in a leaf) has been satisfied.

"Bagging" (see, e.g., [58]) and "boosting" (see, e.g., [59]) are two ways in which the outputs of decision trees can be combined. In bagging, bootstrapped copies of the training data are passed to a collection of decision trees. The trees are then fitted, in parallel, to the training data they receive, and the final prediction of the classifier is an average over all the outputs across the ensemble of trees [60]. In contrast, boosting algorithms such as XGBoost, fit decision trees to training data sequentially, where each subsequent tree improves on the errors in the predictions of class probability of the preceding tree.

In extreme gradient boosting, the iterative process of incrementally improving the prediction of the classifier with every fitted decision tree, reduces to minimizing the following objective function [53]:

$$\mathcal{L}^{\mathrm{obj}}_{t+1} = \sum_i \mathcal{L}(y^i, p^i_{t+1} = p^i_t + O_t) + \gamma T + \frac{1}{2}\lambda O_t^2, \tag{2.9}$$

where, as before, y^i is the ground truth of training data point i, $p^i_t(p^i_{t+1})$ is the classifier's predicted probability of class membership after the sequential fitting of t $(t+1)$ trees. For binary classification problems such as the one we are trying to tackle, the loss function $\mathcal{L}$ is simply the binary cross entropy defined in Eq. (2.8) (summed over the entire training set), and O_t is the output of the decision tree t with respect to which the objective function is to be minimized. The piece $\gamma T + \frac{1}{2}\lambda O_t^2$ in the objective function is a regularization term that controls the classifier's tendency towards overfitting by reducing its sensitivity to individual training data points. Here, T is the total number of leaves in a tree, and λ, γ are hyperparameters that can be appropriately set depending on the data at hand.

Minimizing $\mathcal{L}^{\mathrm{obj}}$ for each decision tree (which can have a vast variety of structures) is in general highly complicated. XGBoost thus simplifies the minimization process in two ways. The first is that the loss function is approximated by a second-degree Taylor polynomial in O_t. The second is that within each tree, the objective function is repeatedly

minimized at each leaf node. As a result, the process of fitting a decision tree reduces to maximizing the gain when splitting the training data at each leaf node. The gain is defined as the difference between the sum of the similarity scores of the two daughter nodes post the split and the similarity score of the parent leaf. The similiarity score at a leaf node l (containing N_l training samples) in tree $t+1$ is defined as [53]

$$S^l_{t+1} = \frac{(\sum_i^{N_l} R^i_t)^2}{\sum_i p^i_t(1-p^i_t)+\lambda}, \tag{2.10}$$

where the sum is taken over all the samples in the leaf node, and $R^i_t \equiv p^i_t - y^i$ is the residual of the ith training data point in the leaf node. The output of each tree, defined as $S^{\rm term}_{t+1}/\sum_i R^i_t$ for the terminal leaf node, is then rescaled by a user defined learning rate η and then added to the log of the odds ratio corresponding to p^i_t, from which the probability estimate of tree $t+1$ can be trivially computed.

As mentioned earlier, λ, γ are user defined regularization parameters that control overfitting. Specifically, γ sets a threshold on the gain; leaves along branches whose gains do not exceed γ are pruned. Thus, since positive values of λ tend to reduce the gain, λ effectively encourages pruning, which in turn reduces the sensitivity of the decision tree to individual training data points.[4]

6. Training and optimization

DenseNet201.—We use a DenseNet pretrained on the "imagenet dataset" [61], which allows it to pick up features common to most images. We then add fully connected layers to it, along with the final layer of just one neuron, for our binary classification, and then retrain it with data specific to our problem (to wit, the superimposed Q transforms). This method of pretraining with a generic dataset and then retraining with a more specific one, is called "transfer learning." The most significant benefit of this method is that it reduces the size of the dataset required for training and solving the problem at hand.

For each of the three detectors H1, L1 and V1, we train three individual DenseNet201 models using superimposed Q-transform pairs, where each image corresponds to a three-dimensional array $(128 \times 128 \times 3)$ of pixels.[5] The DenseNet model is loaded with the imagenet weights using the neural network package [62]. To make it suitable for our binary classification task, its top layer is removed and a dense layer of 256 neurons with the rectified linear unit activation function is added along with the final output layer of a single neuron with a sigmoid activation function. Each of the three models is trained on an equal number (1400) of lensed and unlensed Q-transform image pairs subselected from the DSTV dataset using tensor processing unit hardware, which is available in a KAGGLE notebook [63]. In the top fully connected layer of the network, we use the sigmoid activation function (see. e.g., [64]) and we employ the Adam optimizer [65] for efficient gradient calculations. The model prediction is validated using a validation set subselected from the total training set.

XGBoost.—As described in the previous section, XGBoost has a number of tunable hyperparameters that need to be set based on the problem at hand.

The hyperparameter "n_estimators" sets the number of decision trees in the ensemble classifier that are to be fit to the training data sequentially. It can equivalently be thought of as the number of fitting iterations the model goes through as it sequentially improves the prediction of the ensemble classifier. We set n_estimators to 110. The learning rate, regularization parameter and tree complexity parameter are set to their default values of 0.3,1,0, respectively. The maximum depth of each decision tree is set using max_depth $= 6$.

In addition, we also set the "scale_pos_weight" parameter to 0.01. This hyperparameter serves as a weight to account for training data being biased towards one class—in our case, the unlensed class, for which we had about 100 times more data points than for the lensed class.

The first XGBoost model is trained on the features derived from lensed and unlensed pairs of skymaps, described in Sec. II B 3, using the "DSTV" dataset. Additionally, a second XGBoost model is trained on the outputs of each of the three DenseNet models. The outputs of the two XGBoost models are then combined [cf. Eq. (2.6)] to provide a ranking statistic for candidate lensed pairs.

III. RESULTS

A. Testing and cross validation

We assess the performance of the trained ML models on the "DST" dataset. This allows us to compare their performance with the posterior overlap statistic, which is already computed for this dataset [22]. We summarize the performance of the ML models and the posterior overlap statistic with ROC (receiver operating characteristic) plots of efficiency vs false positive probability (FPP), where efficiency is the ratio of accurately classified lensed events to the total number of lensed events, and FPP is the ratio of wrongly classified unlensed events to the total number of unlensed events.

To check the robustness of the outputs of the machine learning models to changing training sets, we use stratified k-fold cross validation. We implement cross validation by doing a round robin of dividing our dataset into $k = 3$

[4]In ML literature, λ is often referred to as a "regularization parameter" and γ is referred to as a "tree complexity parameter."

[5]Each pixel contains RGB values that correspond to the normalized signal energy at discrete time-frequency coordinates in the Q-transform image.

($k = 10$) parts for the DenseNet (XGBoost) models, using one part for validation and the rest for training. We test the k trained machines with the DST dataset.

B. ROC plots

We evaluate the performance of the overall classifier and its different components using ROCs. For comparison, we also plot the ROCs for the posterior overlap statistic. We first test the performance of the individual DenseNet models trained on Q transforms pertaining to each of the three detectors: H1, L1 and V1. We then test the XGBoost model trained on the outputs of the DenseNet models. Since we used cross validation to assess the robustness of the models, we trained and validated each of the models on the different cross validation subsets of the DSTV dataset, and tested the differently trained models on the DST data set. This gives us an estimate of the variation of the ROCs due to differences in the training set.

Figure 4 plots ROCs for the outputs of these models trained on Q transforms. The ROC for the posterior overlap statistic constructed using parameter estimation posteriors on the component masses (m_1, m_2), is also plotted for comparison. The ROCs pertaining to the individual DenseNet H1, L1, V1 models perform similarly to the ROC for the posterior overlap statistic, both at low and high false positive probabilities. The mean ROC corresponding to the XGBoost model trained on the outputs of the individual DenseNet models performs comparably to the posterior overlap statistic. At very low FPPs, ML seems to perform about 1.5–2 times better than the posterior overlap statistic. However, there is some variation in the XGBoost model's ROC due to the changing training set. These improvements must therefore be interpreted with some caution. As the variation in the ROCs at these FPPs suggests, low-number statistics are likely causing the ROC to be sensitive to changes in the training set.

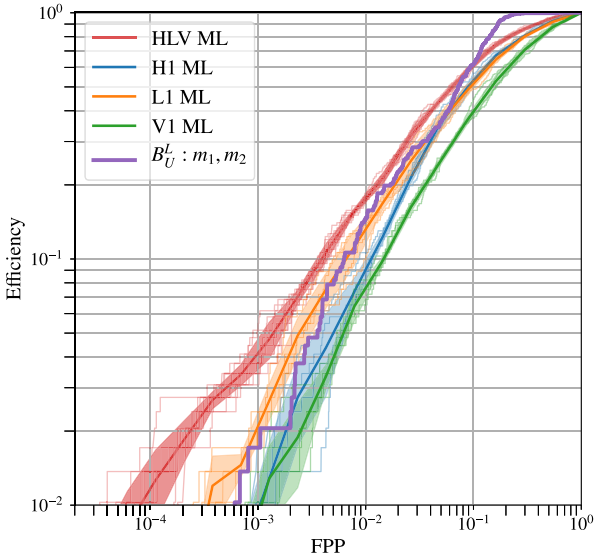

FIG. 4. ROCs for DenseNet models trained on lensed and unlensed pairs of superimposed Q transforms, for different cross-validation subsets of the DSTV training set. ROCs for models trained on Q transforms corresponding to individual detectors are evaluated, in addition to ROCs pertaining to the XGBoost model trained on the outputs of the individual DenseNet models. For comparison, the ROC for the posterior overlap statistic that uses parameter estimation posteriors on the component masses, m_1, m_2, is also plotted. At low false positive probabilities, the individual DenseNet models perform comparably to the posterior-overlap statistic. On the other hand, the XGBoost model produces efficiencies that are 1.5–2 times better than the posterior overlap statistic at low FPPs, although there is some variation in the ROCs when the training set is changed, caused by small-number statistics. These improvements at low FPPs must therefore be interpreted with some caution.

Figure 5 plots ROCs for the XGBoost model trained on the features (metrics) derived from pairs of Bayestar skymaps. Each ROC pertains to a different cross-validation subset of the DSTV dataset. The ROC for the posterior overlap statistic evaluated using only the right ascension (α) and declination (δ) is plotted for comparison. The XGBoost performs as well as the posterior overlap statistic at low false positive probabilities, although at higher false positive probabilities the latter performs marginally better. As with the DenseNet models, there is some variation in the ROCs when the training set is varied.

Figure 6 plots ROCs for the overall classifier, which is an XGBoost model trained on the outputs of the DenseNet models

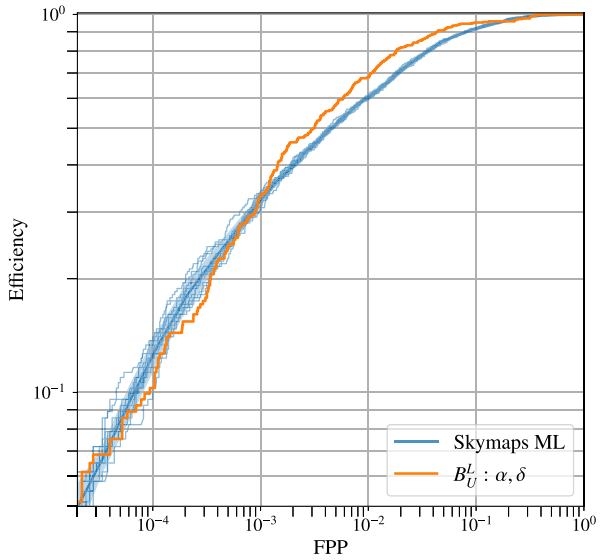

FIG. 5. ROCs for the XGBoost model trained on metrics derived from pairs of Bayestar localization skymaps, for different cross-validation subsets of the DSTV training set. For comparison, the ROC for the posterior overlap statistic that uses parameter estimation posteriors on the skylocation coordinates, α, δ, is also plotted. The XGBoost performs almost as well as the posterior overlap statistic at low false positive probabilities.

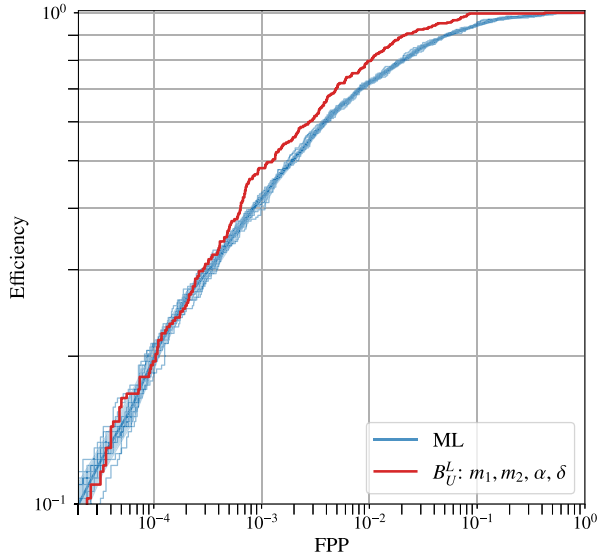

FIG. 6. ROCs for the overall classifier, for different cross-validation subsets of the DSTV training set. Note that the output of the overall classifier is the output of the XGBoost model trained on the ouputs of the three DenseNet models pertaining to H1, L1 and V1, as well as the ouput of the first XGBoost model trained on Bayestar skymaps. At low false positive probabilities, the classifier performs almost identically to the posterior overlap statistic, with mild variation in the ROCs when the training dataset is varied.

and the first XGBoost model. For comparison, the ROC for the posterior overlap statistic evaluated using the parameter estimation posterior on m_1, m_2, α, δ is also plotted. The mean ROC for the overall classifier performs almost identically to the posterior overlap statistic at low false positive probabilities, although at higher false positive probabilities the posterior overlap statistic performs marginally better.

IV. SUMMARY AND OUTLOOK

GW observations of BBH events is expected to increase significantly in future observing runs, with $\mathcal{O}(10^2)$ events during O4 and $\mathcal{O}(10^5$–$10^6)$ during the 3G era. The number of candidate lensed pairs to classifiy could therefore be as high as $\mathcal{O}(10^4)$ and $\mathcal{O}(10^{10}$–$10^{12})$, respectively. Current optimal Bayesian methods, such as the posterior overlap statistic, rely on the parameter estimation posterior on the source parameters, which could take anywhere from several hours to several days to sample.

This therefore motivates the need to come up with a preliminary classification scheme, that can rapidly rule out the vast majority of unlensed candidates. To that end, as a proof of principle, we construct a machine learning based classifier that can classify pairs of nonspinning BBH events in seconds. We use two ML algorithms, DenseNet201 and XGBoost, to build models trained on time frequency maps and Bayestar skymaps of pairs of events. We construct three DenseNet models trained on GW events projected onto each of the three detectors in the LIGO-Virgo network at design sensitivity. The outputs of these models are fed to an XGBoost classifier to construct a corresponding model. The output of this model is then combined with the output of another XGBoost model trained on pairs of lensed and unlensed Bayestar skymaps, to produce the final ranking statistic of our overall ML classifier [cf. Fig. 3 and Eq. (2.6)].

We train and validate the classifier on cross-validation subsets of the DSTV dataset, and test the performance of the ML classifier (including its different components) on the DST dataset. We find that the overall ML classifier performs comparably to the posterior overlap statistic evaluated from the parameter estimation posterior on m_1, m_2, α, δ. More specifically, the performance of the ML classifier, as captured by ROC plots, shows that at low false positive probabilities, the classifier performs almost identically to the posterior overlap statistic, although at high false positive probabilities, the performance of the latter is marginally better.

Simple benchmarking tests suggest that our trained ML classifier is able to classify each event within 2–3 seconds.[6] Including the time to produce the Q-transform images and Bayestar skymaps, the total classification time is still less than a minute. This is significantly faster than the posterior overlap statistic, which takes several minutes to classify once the parameter estimation posteriors are available. Since, in addition, these posteriors themselves can take hours to days to produce, per event, the benefit of using ML to perform a preliminary sweep of lensed candidate pairs to rule out the vast majority of them as unlensed becomes manifestly evident.

Additionally, rapid ranking of candidate pairs makes estimating a background distribution computationally feasible. Such a distribution enables assigning statistics such as p values/false positive probabilities, which are often the preferred statistics since they can be interpreted independently of the models used to analyze the pairs. Another potentially useful application of the rapid identification (and dissemination) of lensed GW events is in multi-messenger astronomy, since the joint GW-electromagnetic detection of lensed events could enable important tests of general relativity.

It might be worth mentioning that in addition to the posterior overlap statistic, there are more comprehensive Bayesian classification methods that take even longer to run. A fully Bayesian, joint parameter estimation scheme to identify lensed pairs by evaluating a coherence ratio that

[6]Note that this time is largely taken up in loading the necessary files for classification. The classification step itself takes less than a second.

accounts for correlations between parameters of lensed events, and selection effects, currently takes of the order of weeks to complete, per candidate pair [66,67]. A more approximate joint parameter estimation method that neglects selection effects, is found to identify lensed pairs with similar efficiencies as the full joint parameter estimation method, but within hours instead of weeks [68]. Thus, identifying lensed pairs from the enormous number of candidate pairs in future observing runs, can follow a stepwise procedure, where an ML classification method such as ours can rapidly rule out most of the candidate pairs as unlensed. The surviving pairs can then be followed up by the posterior overlap statistic and then by joint parameter estimation methods.

Note that our work assumed stationary Gaussian noise, and that the candidate pairs consist of confirmed, high-significance nonspinning BBH events. We plan to systematically relax these assumptions in future work. Specifically, we are currently looking at the possibility of classifying confident GW events in real noise. We plan to train the machine on events injected in real noise, whiten the data so that the Q transforms are less sensitive to varying PSDs, and investigate the possibility of using additional features. We are also working towards the classification of marginal BBH events, with an ML scheme similar to what was presented in this work. We hope to report the results of these investigations in the near future.

ACKNOWLEDGMENTS

We would like to thank Anupreeta More, Deep Chatterjee, Jean-Rene Cudell and Otto Hannuksela for useful discussions. We thank K. Haris for providing the DST dataset used in [22]. S. G. also thanks Pinak Mandal and Shashank Roy for helping her learn various machine learning techniques. S. G.'s, S. J. K.'s, and P. A.'s research was supported by the Department of Atomic Energy, Government of India. In addition, S. J. K.'s work was supported by a grant from the Simons Foundation (Grant No. 677895, R. G.). P. A.'s research was supported by the Max Planck Society through a Max Planck Partner Group at ICTS and by the Canadian Institute for Advanced Research through the CIFAR Azrieli Global Scholars program. This research has made use of data, software and/or web tools obtained from the Gravitational Wave Open Science Center ([69]), a service of LIGO Laboratory, the LIGO Scientific Collaboration and the Virgo Collaboration. LIGO is funded by the U.S. National Science Foundation. Virgo is funded by the French Centre National de Recherche Scientifique (CNRS), the Italian Istituto Nazionale della Fisica Nucleare (INFN) and the Dutch Nikhef, with contributions by Polish and Hungarian institutes. Some of the computations were performed with the aid of the Alice computing cluster at ICTS-TIFR, and rest on KAGGLE [63] with tensor processing unit hardware acceleration.

[1] J. Aasi *et al.*, Classical Quantum Gravity **32**, 074001 (2015).
[2] F. Acernese *et al.*, Classical Quantum Gravity **32**, 024001 (2015).
[3] R. Abbott *et al.*, Phys. Rev. X **11**, 021053 (2021).
[4] B. P. Abbott *et al.*, Phys. Rev. X **9**, 031040 (2019).
[5] B. Zackay, L. Dai, T. Venumadhav, J. Roulet, and M. Zaldarriaga, Phys. Rev. D **104**, 063030 (2021).
[6] T. Venumadhav, B. Zackay, J. Roulet, L. Dai, and M. Zaldarriaga, Phys. Rev. D **101**, 083030 (2020).
[7] T. Venumadhav, B. Zackay, J. Roulet, L. Dai, and M. Zaldarriaga, Phys. Rev. D **100**, 023011 (2019).
[8] B. Zackay, T. Venumadhav, L. Dai, J. Roulet, and M. Zaldarriaga, Phys. Rev. D **100**, 023007 (2019).
[9] B. P. Abbott *et al.*, Phys. Rev. D **100**, 104036 (2019).
[10] R. Abbott *et al.*, Phys. Rev. D **103**, 122002 (2021).
[11] H. C. Ohanian, Int. J. Theor. Phys. **9**, 425 (1974).
[12] P. V. Bliokh and A. A. Minakov, Astrophys. Space Sci. **34**, L7 (1975).
[13] R. J. Bontz and M. P. Haugan, Astrophys. Space Sci. **78**, 199 (1981).
[14] S. Deguchi and W. D. Watson, Astrophys. J. **307**, 30 (1986).
[15] T. T. Nakamura, Phys. Rev. Lett. **80**, 1138 (1998).
[16] X.-L. Fan, K. Liao, M. Biesiada, A. Piórkowska-Kurpas, and Z.-H. Zhu, Phys. Rev. Lett. **118**, 091102 (2017).
[17] G. P. Smith, M. Bianconi, M. Jauzac, J. Richard, A. Robertson, C. P. L. Berry, R. Massey, K. Sharon, W. M. Farr, and J. Veitch, Mon. Not. R. Astron. Soc. **485**, 5180 (2019).
[18] S. Jung and C. S. Shin, Phys. Rev. Lett. **122**, 041103 (2019).
[19] S. Dodelson, *Gravitational Lensing* (Cambridge University Press, Cambridge, England, 2017), https://www.cambridge.org/us/catalogue/catalogue.asp?isbn=9781107129764.
[20] S. Fairhurst, New J. Phys. **11**, 123006 (2009).
[21] S. Fairhurst, Classical Quantum Gravity **35**, 105002 (2018).
[22] K. Haris, A. K. Mehta, S. Kumar, T. Venumadhav, and P. Ajith, arXiv:1807.07062.
[23] L. P. Singer, D. A. Goldstein, and J. S. Bloom, arXiv:1910.03601.
[24] Y. Wang, A. Stebbins, and E. L. Turner, Phys. Rev. Lett. **77**, 2875 (1996).
[25] R. Takahashi and T. Nakamura, Astrophys. J. **595**, 1039 (2003).
[26] L. Dai and T. Venumadhav, arXiv:1702.04724.
[27] J. M. Ezquiaga, D. E. Holz, W. Hu, M. Lagos, and R. M. Wald, Phys. Rev. D **103**, 064047 (2021).
[28] J. Veitch *et al.*, Phys. Rev. D **91**, 042003 (2015).
[29] O. A. Hannuksela, K. Haris, K. K. Y. Ng, S. Kumar, A. K. Mehta, D. Keitel, T. G. F. Li, and P. Ajith, Astrophys. J. Lett. **874**, L2 (2019).

[30] R. Abbott *et al.*, Astrophys. J. Lett. **913**, L7 (2021).
[31] K. K. Y. Ng, K. W. K. Wong, T. Broadhurst, and T. G. F. Li, Phys. Rev. D **97**, 023012 (2018).
[32] F. Xu, J. M. Ezquiaga, and D. E. Holz, arXiv:2105.14390.
[33] D. Reitze *et al.*, Bull. Am. Astron. Soc. **51**, 35 (2019), arXiv:1907.04833.
[34] S. Hild, S. Chelkowski, and A. Freise, arXiv:0810.0604.
[35] S. Hild, M. Abernathy, F. Acernese, P. Amaro-Seoane, N. Andersson, K. Arun, F. Barone, B. Barr, M. Barsuglia, M. Beker *et al.*, Classical Quantum Gravity **28**, 094013 (2011).
[36] S. Chatterji, L. Blackburn, G. Martin, and E. Katsavounidis, Classical Quantum Gravity **21**, S1809 (2004).
[37] L. P. Singer and L. R. Price, Phys. Rev. D **93**, 024013 (2016).
[38] C. Cutler and É. E. Flanagan, Phys. Rev. D **49**, 2658 (1994).
[39] L. Dai, B. Zackay, T. Venumadhav, J. Roulet, and M. Zaldarriaga, arXiv:2007.12709.
[40] R. Magee *et al.*, Astrophys. J. Lett. **910**, L21 (2021).
[41] T. E. Collett, Astrophys. J. **811**, 20 (2015).
[42] M. Hannam, P. Schmidt, A. Bohé, L. Haegel, S. Husa, F. Ohme, G. Pratten, and M. Pürrer, Phys. Rev. Lett. **113**, 151101 (2014).
[43] S. Husa, S. Khan, M. Hannam, M. Pürrer, F. Ohme, X. J. Forteza, and A. Bohé, Phys. Rev. D **93**, 044006 (2016).
[44] S. Khan, S. Husa, M. Hannam, F. Ohme, M. Pürrer, X. J. Forteza, and A. Bohé, Phys. Rev. D **93**, 044007 (2016).
[45] LIGO Scientific Collaboration, LIGO Algorithm Library—LALSuite, free software (GPL) (2018).
[46] A. Nitz *et al.*, gwastro/pycbc: Pycbc release 1.16.4 (2020).
[47] The Virgo Collaboration, Advanced Virgo sensitivity curve study, Technical Report No. VIR-0073D-12 (Virgo Collaboration, 2012).
[48] The updated Advanced LIGO design curve, Technical Report No. LIGO-T1800044-v5 (LIGO Document Control Center, 2018).
[49] M. R. Calabretta and B. F. Roukema, Mon. Not. R. Astron. Soc. **381**, 865 (2007).
[50] A. Zonca, L. Singer, D. Lenz, M. Reinecke, C. Rosset, E. Hivon, and K. Gorski, J. Open Source Software **4**, 1298 (2019).
[51] K. M. Górski, E. Hivon, A. J. Banday, B. D. Wandelt, F. K. Hansen, M. Reinecke, and M. Bartelmann, Astrophys. J. **622**, 759 (2005).
[52] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, arXiv:1608.06993.
[53] T. Chen and C. Guestrin, KDD '16, *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 785–794, arXiv:1603.02754.
[54] O. I. Abiodun, A. Jantan, A. E. Omolara, K. V. Dada, A. M. Umar, O. U. Linus, H. Arshad, A. A. Kazaure, U. Gana, and M. U. Kiru, IEEE Access **7**, 158820 (2019).
[55] J. Schmidhuber, Neural Netw. **61**, 85 (2015).
[56] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction* (Springer Science & Business Media, New York, 2009).
[57] C. Nwankpa, W. Ijomah, A. Gachagan, and S. Marshall, arXiv:1811.03378.
[58] L. Breiman, Mach. Learn. **24**, 123 (1996).
[59] Y. Freund and R. E. Schapire, in *In Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence* (Morgan Kaufmann Publishers Inc., CA, USA, 1999), pp. 1401–1406, http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.41.5846.
[60] L. Breiman, Mach. Learn. **45**, 5 (2001).
[61] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, in *2009 IEEE Conference on Computer Vision and Pattern Recognition* (IEEE, 2009), pp. 248–255, https://ieeexplore.ieee.org/document/5206848.
[62] F. Chollet *et al.*, Keras, https://github.com/fchollet/keras (2015).
[63] Kaggle, https://www.kaggle.com.
[64] C. Nwankpa, W. Ijomah, A. Gachagan, and S. Marshall, arXiv:1811.03378.
[65] D. P. Kingma and J. Ba, arXiv:1412.6980.
[66] R. K. L. Lo and I. M. Hernandez, arXiv:2104.09339.
[67] X. Liu, I. M. Hernandez, and J. Creighton, Astrophys. J. **908**, 97 (2021).
[68] J. Janquart, O. A. Hannuksela, K. Haris, and C. Van Den Broeck, Mon. Not. R. Astron. Soc. **506**, 5430 (2021).
[69] R. Abbott *et al.* (LIGO Scientific Collaboration and Virgo Collaboration), SoftwareX **13**, 100658 (2021).