Journal of Cosmology and Astroparticle Physics
An IOP and SISSA journal

Probabilistic learning for pulsar classification

Sambatra Andrianomena

South African Radio Astronomy Observatory (SARAO),
Black River Park, Observatory, Cape Town, 7925, South Africa
Department of Physics & Astronomy, University of the Western Cape,
Bellville, Cape Town 7535, South Africa

E-mail: andrianomena@gmail.com

Received May 13, 2022
Revised August 19, 2022
Accepted September 18, 2022
Published October 6, 2022

Abstract. In this work, we explore the possibility of using probabilistic learning to identify pulsar candidates. We make use of Deep Gaussian Process (DGP) and Deep Kernel Learning (DKL). Trained on a balanced training set in order to avoid the effect of class imbalance, the performance of the models, achieving relatively high probability of differentiating the positive class from the negative one (*roc-auc* $\sim$ 0.98), is very promising overall. We estimate the predictive entropy of each model predictions and find that DKL is more confident than DGP in its predictions and provides better uncertainty calibration. Upon investigating the effect of training with imbalanced dataset on the models, results show that each model performance decreases with an increasing number of the majority class in the training set. Interestingly, with a number of negative class 10$\times$ that of positive class, the models still provide reasonably well calibrated uncertainty, i.e. an expected Uncertainty Calibration Error (UCE) less than 6%. We also show in this study how, in the case of relatively small amount of training dataset, a convolutional neural network based classifier trained via Bayesian Active Learning by Disagreement (BALD) performs. We find that, with an optimized number of training examples, the model — being the most confident in its predictions — generalizes relatively well and produces the best uncertainty calibration which corresponds to UCE = 3.118%.

Keywords: Bayesian reasoning, Machine learning , radio pulsars

ArXiv ePrint: 2205.05765

https://doi.org/10.1088/1475-7516/2022/10/016

JCAP10(2022)016

Contents

JCAP10(2022)016

1 Introduction

The detection of a pulsar PSR 1913+16 in a binary system by [1] and its subsequent monitoring [2] pointed toward the existence of gravitational radiation, an energy loss which is consistent with the decrease of the orbital period of the system. A couple of decades later, gravitational waves from a binary black hole merger were directly detected using the Laser Interferometer Gravitational-Wave Observatory (LIGO) experiment [3], confirming Einstein's prediction. PSR 1913+16, described as "*an accurate clock moving at high velocity in the strong gravitational field of its unseen companion*" in [2], was utilized as a laboratory test for gravity. Their results confirmed Einstein's General Relativity (GR) theory at 0.2% level. Further tests using PSR B1534+12 [4] and PSR J0737-3039A/B [5] put constraints on GR at 0.7% and 0.05% level respectively [6], using ten binary systems,[1] also investigated the viability of *Screened modified gravity* by constraining the Post-Keplerian Parameters of the theory. These examples highlight the important role that pulsars play when addressing the validity of Einstein's GR and other alternative theories in highly non-linear regime.

In nuclear physics, [7] demonstrated the correlation between the inner crust composition of neutron star and its observed spin period, placing some constraints on the latter. Ref. [8] listed some tests in fundamental physics where pulsars can be used as tools. Ref. [9] showed

[1] 5 neutron star - neutron star and 5 neutron star - white dwarf systems.

that Pulsar Timing Arrays (PTAs) will help further our understanding of dark matter, are great probes for the detecting cosmic superstrings, and will be used to place constraints on gravitational wave spectrum in the inflationary universe. NANOGrav Collaboration [10, 11] is spending a great amount of effort in an attempt to detect stochastic gravitational-wave background using PTAs.

With the advent of upcoming big survey like SKA, a considerable increase of the number of detected pulsars is expected. Ref. [12] estimated that SKA experiment would detect 14000 pulsars. At the time of writing, MeerKAT, a precursor of SKA-Mid, has observed 1005 pulsars [13]. The huge amount of data from experiments requires an automated way to identify pulsar candidates. Ref. [14], for instance, trained an Artificial Neural Network (ANN) by considering 12 predictors as inputs[2] to search for candidates and discovered a new pulsar using the method. Similar approaches, using 22 and 6 features as inputs to train an ANN, were adopted in [15] and [16] respectively. Following [16, 17] selected the same features (6 of them) to train neural network and tree based algorithms. The Pulsar Image-based Classification System (PICS) prescribed by [18] was more involved. Having highlighted the potential bias induced by the score based systems which use the predictor variables as inputs to the algorithms, [18] opted for methods that extracted the features directly from the diagnostic plots. In other words, they fed the summed pulse profile (1D), time vs phase plot (2D image), frequency vs phase plot (2D image) and Dispersion Mesure (DM) curve (1D) to PICS, yielding an instance comprising 4 different inputs. The innovative approach consisted of two stages. In the first stage, each type of inputs was fed to two different methods, giving a total of 8 output scores which were then passed through a logistic regression for predictions in the second stage. They used convolutional neural network (CNN) combined with SVM to learn the features from the images and ANN (with dense layers) combined with SVM to extract features from 1D inputs.

Deep neural network classifiers tend to be "*overconfident*" in their predictions which are point estimates by construction. It was shown in [19, 20] that neural network classifiers are likely to classify an input, regardless of the fact that the latter is drawn from an out-of-distribution sample, with high probability (Softmax output). By estimating uncertainties, it is possible to assess how confident the classifier predictions are. Given a model with good uncertainty calibration, predictions associated with high predictive uncertainty can be discarded as they indicate what the model doesn't know. To estimate uncertainties in classification task, one can resort to probabilistic learning such as Bayesian Neural Network and Monte-Carlo (MC) Dropout model [21].

In this work, we make use of Deep Gaussian Process (DGP) and Deep Kernel Learning (DKL) for pulsar classification. We also demonstrate the use of Bayesian Active Learning by Disagreement (BALD) to train a CNN classifier in the case where a relatively small amount of labeled data is available for training. Our aim is twofold: highlighting the predictive power of probabilistic models and estimating predictive uncertainty, which is done for the first time in pulsar classification, to the best of our knowledge. The paper is structured as follows: we present the dataset used in this work in section 2 and the methods we consider in section 3. The performance of the classifiers is presented in section 4. In section 5, we provide the details of the predictive uncertainty estimation and assess the calibration of the uncertainties produced by the models. We investigate the effect of the imbalanced training dataset on how the models perform and the resulting prediction uncertainties in section 6. We show the use of BALD to classify pulsars in section 7 and finally conclude in section 8.

[2] See their table 1.

JCAP10(2022)016

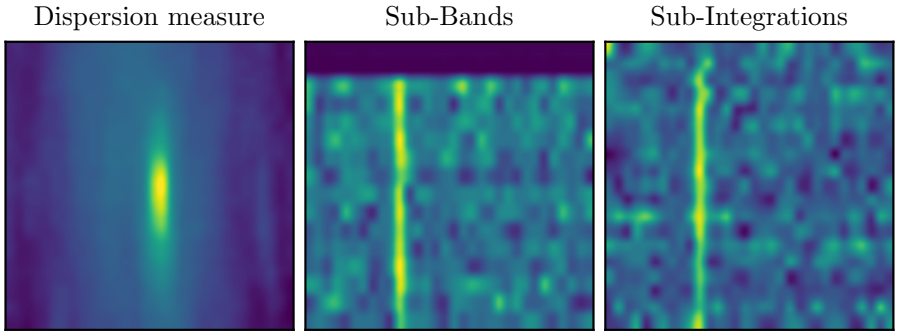

Figure 1. Example of pulsar in the dataset we consider in this study. Left panel shows the 2D dispersion measure (DM), middle panel represents Sub-band and Sub-integration is shown on the right panel. Each input is composed of these three channels, mimicking RGB channels.

2 Data

We consider pulsar data, named HTRU Medlat Data, which were collected by [22] and [16]. The HTRU1 batched data, a subset of HTRU Data, contains 50000 labeled images for training and 10000 images for testing.[3] The imbalance ratio in both the training and testing dataset is about 1:49. Each input has three channels which are Dispersion Measure (channel 0), Sub-band (channel 1) and Sub-integration (channel 2). Each channel, which is a 2D image, has a resolution of 32×32 pixels. In our approach, we select all the channels — Dispersion Measure, Sub-bands, Sub-integrations. As an example, we present in figure 1 the 2D map of each channel of a pulsar. Our task is a binary classification in which the positive class is pulsar and the negative class is non-pulsar (rfi). The target is categorical; **0** for non-pulsar and **1** for pulsar.

3 Algorithms

In this section, we give an overview of the methods that are used in our analyses. We first present the approach used to find an optimal solution for each algorithm. In a probabilistic model inference, one can consider a sampling strategy such as Metropolis-Hastings algorithm [23–25] which is more accurate but time consuming as it fully explores the parameter space or a faster method known as variational inference [26] but less accurate. We adopt the latter in our analyses.

3.1 Stochastic Variational Inference (SVI)

The approximation of the posterior probability emerges out of attempting to maximize the log marginal likelihood

$$\log p(\mathbf{y}) = \log \int \mathrm{d}\boldsymbol{\theta}\; p(\mathbf{y}, \boldsymbol{\theta}), \tag{3.1}$$

which, in general intractable, consists of averaging out all parameters in a model. Introducing a known variational distribution q_γ in eq. (3.1) gives

$$\log p(\mathbf{y}) = \log \int \mathrm{d}\boldsymbol{\theta}\; q_\gamma(\boldsymbol{\theta}) \frac{p(\mathbf{y}, \boldsymbol{\theta})}{q_\gamma(\boldsymbol{\theta})}. \tag{3.2}$$

[3]Data can downloaded from https://github.com/as595/HTRU1.

JCAP10(2022)016

By using Jensen's inequality which states that $\log \mathbb{E}[f(x)] \geq \mathbb{E}[\log f(x)]$ and introducing the posterior distribution yield

$$\log p(\mathbf{y}) \geq \int \mathrm{d}\boldsymbol{\theta}\, q_\gamma(\boldsymbol{\theta}) \log \frac{p(\mathbf{y}, \boldsymbol{\theta})}{q_\gamma(\boldsymbol{\theta})}, \tag{3.3}$$

where the right-hand side is known as the evidence lower bound ($\mathbb{ELBO}(q_\gamma)$) [27]. We have that

$$\log p(\mathbf{y}) - \mathbb{ELBO}(q_\gamma(\boldsymbol{\theta})) = \log p(\mathbf{y}) - \int \mathrm{d}\boldsymbol{\theta}\, q_\gamma(\boldsymbol{\theta}) \log \frac{p(\boldsymbol{\theta}|\mathbf{y}) p(\mathbf{y})}{q_\gamma(\boldsymbol{\theta})}. \tag{3.4}$$

Rearranging the integral term gives

$$\log p(\mathbf{y}) - \mathbb{ELBO}(q_\gamma(\boldsymbol{\theta})) = \log p(\mathbf{y}) - \int \mathrm{d}\boldsymbol{\theta}\, q_\gamma(\boldsymbol{\theta}) \log \frac{p(\boldsymbol{\theta}|\mathbf{y})}{q_\gamma(\boldsymbol{\theta})} - \int \mathrm{d}\boldsymbol{\theta}\, q_\gamma(\boldsymbol{\theta}) \log p(\mathbf{y}). \tag{3.5}$$

Exploiting the fact that q_γ is a probability density function (i.e. $\int \mathrm{d}\boldsymbol{\theta} q_\gamma(\boldsymbol{\theta}) = 1$) and simplifying the right-hand side of eq. (3.5) finally give

$$\log p(\mathbf{y}) - \mathbb{ELBO}(q_\gamma(\boldsymbol{\theta})) = \int \mathrm{d}\boldsymbol{\theta}\, q_\gamma(\boldsymbol{\theta}) \log \frac{q_\gamma(\boldsymbol{\theta})}{p(\boldsymbol{\theta}|\mathbf{y})}, \tag{3.6}$$

where the right-hand side is the Kullback-Leibler divergence ($\mathbb{KL}(q(\boldsymbol{\theta})||p(\boldsymbol{\theta}|\mathbf{y}))$, which measures the dissimilarity between two distributions. KL-divergence is both asymmetric[4] and positive, and as it gets closer to zero, the variational distribution ($q_\gamma(\boldsymbol{\theta})$) approaches the true posterior distribution. To approximate the latter, one chooses to either minimize the KL-divergence or maximize the evidence lower bound by varying the variational parameters γ in a stochastic gradient descent (or ascent) manner. In all our analyses, we opt for maximizing the evidence lower bound.

3.2 Deep Gaussian Process (DGP)

As prescribed in [28], DGP consists of chaining up GP layers such that the outputs (latent spaces) of an intermediate layer are the inputs of the following one. Assuming we have two GP layers for simplicity, we have that

$$\mathbf{y} = \boldsymbol{f}^y(\mathbf{t}) + \boldsymbol{\epsilon_o}, \tag{3.7}$$

$$\mathbf{t} = \boldsymbol{f}^t(\mathbf{X}) + \boldsymbol{\epsilon_i}, \tag{3.8}$$

where $\boldsymbol{\epsilon_i}$ and $\boldsymbol{\epsilon_o}$ are Gaussian noise at the hidden layer and output layer respectively, $\mathbf{t}$ is a noisy realization of the intermediate latent function $\boldsymbol{f}^t(\mathbf{X}) \sim \mathcal{GP}(\mathbf{0}, K^t(\mathbf{X}, \mathbf{X}'))$ and the observation $\mathbf{y}$ is also a noisy realization of the latent function $\boldsymbol{f}^y(\mathbf{t}) \sim \mathcal{GP}(\mathbf{0}, K^y(\mathbf{t}, \mathbf{t}'))$. We refer the interested reader to [28] for full details of the theory. In practice, we consider a DGP composed of two layers of variational sparse GPs (see appendix A for more details), each with a radial basis function (RBF) kernel. For the training we select: Adam optimizer, learning rate = 0.02, batch size = 256 and number of inducing points = 64. The algorithm is trained by maximizing the $\mathbb{ELBO}$ via variational inference for 31 epochs. It is noted that due to the stochastic nature of the prediction, owing to the hidden layer, we sample 100

[4] $\mathbb{KL}(q||p) \neq \mathbb{KL}(p||q)$.

JCAP10(2022)016

	Layer	(in channel, out channel, kernel, stride)
1	Convolutional Layer	(3, 16, 3×3 , 2)
2	ReLU Activation	–
3	Convolutional Layer	(16, 16, 3×3 , 1)
4	ReLU Activation	–
5	Convolutional Layer	(16, 32, 3×3 , 2)
6	ReLU Activation	–
7	Convolutional Layer	(32, 32, 3×3 , 1)
8	ReLU Activation	–
9	Convolutional Layer	(32, 32, 3×3 , 2)
10	ReLU Activation	–
11	Convolutional Layer	(32, 32, 3×3 , 1)
12	ReLU Activation	–
13	Flatten	–
14	Fully Connected Layer	(512, 1024, –, –)
15	ReLU Activation	–

Table 1. The architecture of the feature extractor that is considered in this study.

predictions in each forward pass. Both the predictive mean $\hat{\boldsymbol{f}}$ and standard deviation $\hat{\boldsymbol{\sigma}}$ of the latent function (logit) are defined as

$$\hat{\boldsymbol{f}} = \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{f}_i, \tag{3.9}$$

$$\hat{\boldsymbol{\sigma}} = \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{\sigma}_i, \tag{3.10}$$

where $\boldsymbol{f}_i$ and $\boldsymbol{\sigma}_i$ are the mean and standard deviation in each sample respectively and N is the sample size in each forward pass. The implementation of this method is achieved with Pyro library [29].

3.3 Deep Kernel Learning (DKL)

For inputs with high dimensional features, e.g. 2D image with 256×256 pixels, using simple kernel based classifier such as GP might be a challenge. Ref. [30] exploited the capacity of a deep neural network and the flexibility of a GP to arrive at a probabilistic deep network which they named Stochastic Variational Deep Kernel Learning. The salient features from the image (input) are first extracted via a neural network model. The extracted features are fed into a Gaussian process which in turn outputs the predictive mean $\hat{\boldsymbol{f}}$ and standard deviation $\hat{\boldsymbol{\sigma}}$ of the latent function. Predicting a class label[5] is done by feeding a sample $\boldsymbol{f} \sim \mathcal{N}(\hat{\boldsymbol{f}}, \hat{\boldsymbol{\sigma}})$ to a sigmoid function. The weights of the network together with the GP

[5]In the case of classification.

	f_1-*score*	*recall*	*precision*	*roc-auc*	*specificity*
DGP	0.955	0.924	0.989	0.992	0.989
DKL	0.969	0.954	0.984	0.992	0.984
DCGAN-L2-SVM-2 [31]	0.964	0.963	0.965	—	—
DCNN-S [32]	0.962	0.962	0.963	—	—
BALD	0.947	0.914	0.983	0.987	0.984

Table 2. Performance metrics used to assess the performance of the classifiers.

hyperparameters are learnt by maximizing $\mathbb{ELBO}$ using variational inference. In this study, we consider the architecture presented in table 1 as our feature extractor and the base kernel of the GP layer is also RBF. For the training we choose: Adam optimizer, learning rate = 0.001, batch size = 256 and number of inducing points = 64. The training converges over 200 epochs. We also use Pyro for the model implementation.

4 Model performance

As a way to mitigate the effect of imbalance on the training, we consider a representative sample which is composed of all positive instances in the original training dataset[6] and the same number of randomly drawn negative instances, giving a balanced dataset of 1990 instances in total. That latter is split into training set (80%) and validation set (20%). The original testing dataset[7] has 199 positive instances (pulsar) which are combined with randomly drawn negative instances (rfi) from the same testing dataset to get a balanced test set with 398 examples in total. It is worth noting that, although we investigate the effect of imbalanced training dataset on the predictions in the following section, we defer a thorough investigation on dealing with imbalance classification for future work. To assess the performance of each method, we use various metrics

- *recall* (also known as *sensitivity* or *completeness*) encodes the minimization of the number of false negatives which are positive instances misidentified as negative ones.
- *specificity* indicates how well the number of false positives, which are negative instances incorrectly classified as positive instances, is minimized.
- *precision* (also known as *purity*) denotes how well the positive class is identified.
- f_1-*score* is simply the harmonic mean of the *recall* and *precision*.
- *roc-auc* is the degree of separability which indicates how good a classifier performs in terms of making the distinction between the two classes in our case which is a binary classification.

We refer the interested reader to [33] for a more explicit summary of the metrics mentioned above. Table 2 shows the results corresponding to each algorithm. Overall, the values of the metrics are > 0.92 indicating a relatively good performance of all the methods. DKL, with

[6] Which has 50000 examples in total.
[7] Which has 10000 examples.

JCAP10(2022)016

its higher *recall*, is more sensitive than DGP. However comparing the values of *recall* with those of *specificity* suggests that all learners are more likely to misclassify pulsars. This can be explained by the fact that the salient features of pulsar candidates can be challenging to extract due to the strength of the signal for instance. Results also show that the models exhibit similar performance in terms of identifying the positive class, as evidenced by the values of their *precision*. In pulsar candidate search, provided the relative scarcity of the object, the idealized scenario is a classifier that has high *sensitivity* and a great capability of detecting the positive class (high *precision*). However, the *precision/recall* tradeoff which is known as the effect of optimizing one to the detriment of the other, may be challenging to overcome. Therefore, even though optimizing *recall* is the main objective, it can only be done at a fixed value of tolerable *precision*.[8] A very useful metric that encodes the effect of that tradeoff is f_1-*score* which is in favour of a method that optimizes both *precision* and *recall*.

Apart from *roc-auc* score, the other metrics considered in this work are all based on a single value of a threshold which can be a score (e.g. logit) or a probability. The predicted class label is positive or negative whether the predicted probability/score is above or below the threshold respectively. In our case, the value of that threshold, which is a probability, is 0.5.[9] Therefore, depending on the problem, the value of the threshold can be adjusted to meet a target value of a metric particularly chosen for the task. In general, the *roc-auc* score, which is the area under the curve of the *true positive rate*[10] against the *false positive rate*,[11] is commonly used to compare how well different methods generalize in a given classification task. The reason is that the curve highlights the tradeoff between *recall* and *false positive rate*, in other words the value of the former that corresponds to that of the latter at all possible values of the threshold. Therefore *roc-auc* can be viewed as the mean value of the *recall* at different values of the threshold. High values of *roc-auc* (~ 0.99) indicate that the capability of the models to distinguish pulsars from non-pulsars is promising.

In table 2, we show some results from previous studies. Ref. [31] made use of the features extracted from a generative model (Deep Convolutional Generative Adversarial Network) trained on the HTRU dataset as inputs to a support vector machine (SVM). They fed the Sub-bands and Sub-Integrations maps into their model for the learning process and achieved good performance as suggested by their chosen metric values all above 0.96. Ref. [32], who also selected Sub-bands and Sub-Integrations maps as inputs, resorted to data augmentation to train their CNN and obtained a performance similar to the model used in [31]. Overall our methods, although slightly less sensitive (especially DGP), perform comparably to those of [31] and [32]. However, since we consider different inputs for our models and use slightly different dataset, the idea behind this comparison is to check that our results are consistent with those of previous studies.

5 Quantifying uncertainty

Uncertainty in deep learning was shown to have two main components [34, 35]. Epistemic uncertainty, also referred to as model uncertainty, describes the uncertainty in the model parameters and is propagated through to the model predictions. The model ignorance is encoded in this type of uncertainty and decreases with more data for the training. The other component

[8] And vice versa.
[9] It is the default value in most cases.
[10] Another name for *recall*.
[11] This is defined as *false positive*/(*false positive* + *true positive*).

JCAP10(2022)016

is aleatoric uncertainty which encodes the inherent noise in the observed data. The aleatoric uncertainty which cannot be mitigated with more data, can be dependent (heteroscedastic) or independent (homoscedastic) of the inputs. In our analyses, we estimate the predictive entropy (total uncertainty) which is the sum of the epistemic and aleatoric uncertainties.

5.1 Predictive uncertainty

We make use of the prescription described in [34, 35] and also followed by [36] in their in-depth investigation of deep learning uncertainty in radio galaxy classification. At a test point, the predictive entropy is given by

$$\mathcal{H}(y_*|X_*, \mathcal{D}) = -\sum_{c=0}^{C-1} p(y_* = c|X_*, \mathcal{D}) \log p(y_* = c|X_*, \mathcal{D}), \tag{5.1}$$

where C is the number of classes and the predictive probability is given by

$$p(y_* = c|X_*, \mathcal{D}) = \frac{1}{N}\sum_{i=1}^{N} \text{Softmax}(\boldsymbol{f}_i(X_*)) \tag{5.2}$$

where $\boldsymbol{f}_i(X_*)$ is a sample that can be obtained from a stochastic forward pass in the case of MC Dropout model [21]. In the case of Bayesian Neural Network, $\boldsymbol{f}_i(X_*)$ is a prediction obtained by sampling from the posterior distribution of the model weights. In our case,[12] provided that our models output posterior mean and uncertainty of the logit ($\hat{\boldsymbol{f}}$, $\hat{\boldsymbol{\sigma}}$), $\boldsymbol{f}_i$ can be sampled from the distribution $\mathcal{N}(\hat{\boldsymbol{f}}, \hat{\boldsymbol{\sigma}})$. In information theory, the unit of entropy depends on the base of logarithm which is used in equation (5.1). We use natural logarithm throughout, therefore the unit is natural unit (nat). The entropy is maximum (corresponding to maximum uncertainty) if predicting rfi and predicting pulsar are equally likely.[13]

We show the distributions of predictive uncertainty obtained from each model in figure 2. The red and blue boxes correspond to the uncertainty distribution of DKL and DGP respectively. Each box indicates the first quartile (Q_1), the median, the third quartile (Q_3). The minimum and maximum[14] are given as a function of the interquartile range (IQR), and the dots denote the outliers. The results suggest that DKL model has more confidence in its predictions compared to DGP. This is evidenced by both its lower median value of entropy (DKL: 0.046 nats, DGP: 0.114 nats) and its smaller IQR. This can be explained by the capacity of DKL model which uses convolutional layers to capture the relevant features of the input, unlike the DGP model whose kernel is built from high dimensional data.[15]

5.2 Uncertainty calibration

The uncertainty estimated from our models which are trained via variational inference can be miscalibrated. As defined in [37], the uncertainty is perfectly calibrated if

$$\mathbb{P}(\hat{y} \neq y|\mathcal{H}(y|X, \mathcal{D}) = q) = q, \ \forall q \in [0, 1]. \tag{5.3}$$

In other words, the test error[16] in a test set (X, y) is strongly correlated with the predictive uncertainty in the case of perfect calibration. To assess the calibration of the

[12] Both DGP and DKL.
[13] In that case, $p(y_* = 0|X_*) = p(y_* = 1|X_*) = 0.5$, hence $\mathcal{H} = 2 \times \log(0.5) \times 0.5 = 0.693147$.
[14] Which are denoted by the whiskers.
[15] Each input has to be flattened to get a one dimensional array with length $3 \times 32 \times 32$.
[16] Or top-1 error in the case of multi-class.

JCAP10(2022)016

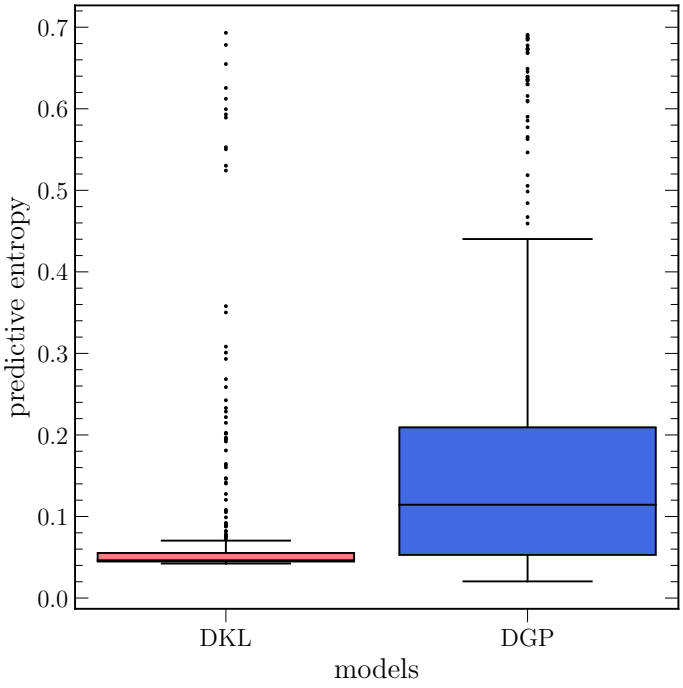

Figure 2. Distribution of the predictive uncertainty. Red denotes the uncertainty distribution corresponding to DKL model, whereas blue corresponds to that of DGP model.

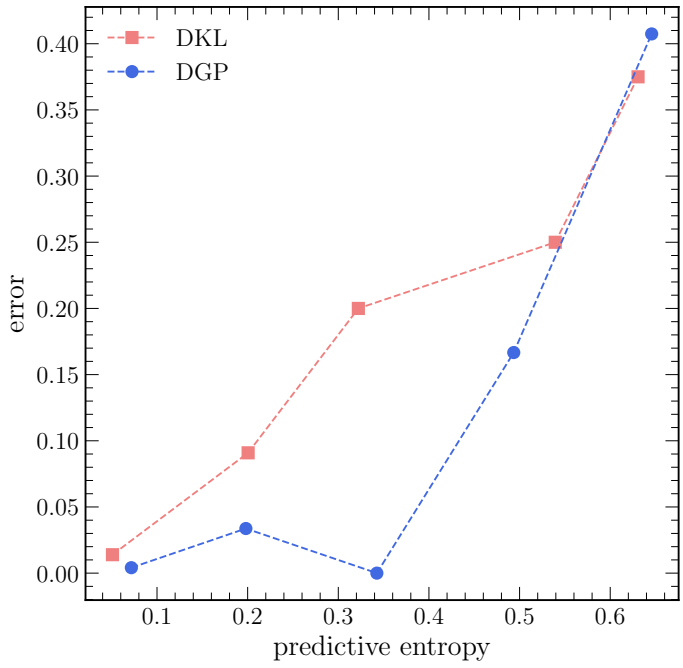

Figure 3. Test error in each bin as a function of average of predictive uncertainties in the same bin.

uncertainty obtained from our model, we compute the expected Uncertainty Calibration Error (UCE), which was prescribed by [37], and is given by

$$\text{UCE} = \sum_{m=1}^{M} \frac{|B_m|}{n} |\text{err}(B_m) - \text{uncert}(B_m)|, \tag{5.4}$$

where M is the number of bins, B_m denotes all the instances in a given bin, n is the total number of instances, $\text{err}(B_m)$ indicates the average error in each bin which is defined as

$$\text{err}(B_m) = \frac{1}{|B_m|} \sum_{i \in B_m} \mathbf{1}(\hat{y}_i \neq y), \tag{5.5}$$

and the average uncertainty in each bin $\text{uncert}(B_m)$ is given by

$$\text{uncert}(B_m) = \frac{1}{|B_m|} \sum_{i \in B_m} \text{uncert}_i. \tag{5.6}$$

uncert_i is the predictive entropy in our case. Due to the relatively small number of examples in the test set, we set the number of bins to 7. We present in figure 3 the variation of test error as a function of the average of predictive entropy. Red squares and blue circles are the errors (see equation (5.5)) obtained respectively from DKL and DGP in each bin. It is noticed that there are only 5 data points for each model. This is due to the fact that some bins are empty. Using equation (5.4), we obtain UCE = 4.897% and UCE = 12.728% with DKL and DGP respectively. This suggests that DKL produces good uncertainty calibration and the uncertainty from DGP model is not well calibrated. Ref. [37] obtained UCE $<$ 10% on classifying CIFAR dataset [38] after calibrating the uncertainties generated from deep network with MC Dropout. Despite the relatively small test set, to some extent it can be inferred that the uncertainties produced by DKL are good enough for our purpose. To quantify the miscalibration of the uncertainty, it is also possible to assess the strength of the correlation between the error and the predictive entropy (shown in figure 3) by computing the Pearson's correlation coefficient but due to the relatively small amount of data, that quantity is highly dependent on the number of bins selected and hence will not be considered in our analyses.

Another approach, which was considered by [39] to check whether a model can produce reliable uncertainty, is to compare the variance of the ratio of true positive/true negative ($\sigma^2_{TP/TN}$) with that of the ratio of false positive/false negative ($\sigma^2_{FP/FN}$). A model produces good uncertainty if $\sigma^2_{FP/FN} \gg \sigma^2_{TP/TN}$. The ratios TP/TN and FP/FN are computed for one prediction on the balanced test set. To compute the variances $\sigma^2_{FP/FN}$ and $\sigma^2_{TP/TN}$ for each model, 200 predictions are sampled. As presented in table 3, $\sigma^2_{FP/FN}$ is about two order of magnitude greater than $\sigma^2_{TP/TN}$ for all models, suggesting that the classifiers give good uncertainty estimation. The results in table 3 also indicate that uncertainty estimation from DKL model is more reliable.

6 Imbalance classification

In this section, although dealing with imbalance is outside the scope of this study, we address the effect of the imbalanced training dataset on both the performance of the classifiers and the uncertainty estimation. To this end, we consider three different imbalance ratios of the training set ρ = 1:1, 1:10, 1:49.[17]

[17] This is the imbalance ratio of the original training set.

JCAP10(2022)016

	$\sigma^2_{TP/TN}$	$\sigma^2_{FP/FN}$
DGP	9.110×10^{-5}	7.189×10^{-3}
DKL	5.107×10^{-5}	4.626×10^{-3}

Table 3. Variance of the ratio of true positive/true negative, and that of false positive/false negative for each model.

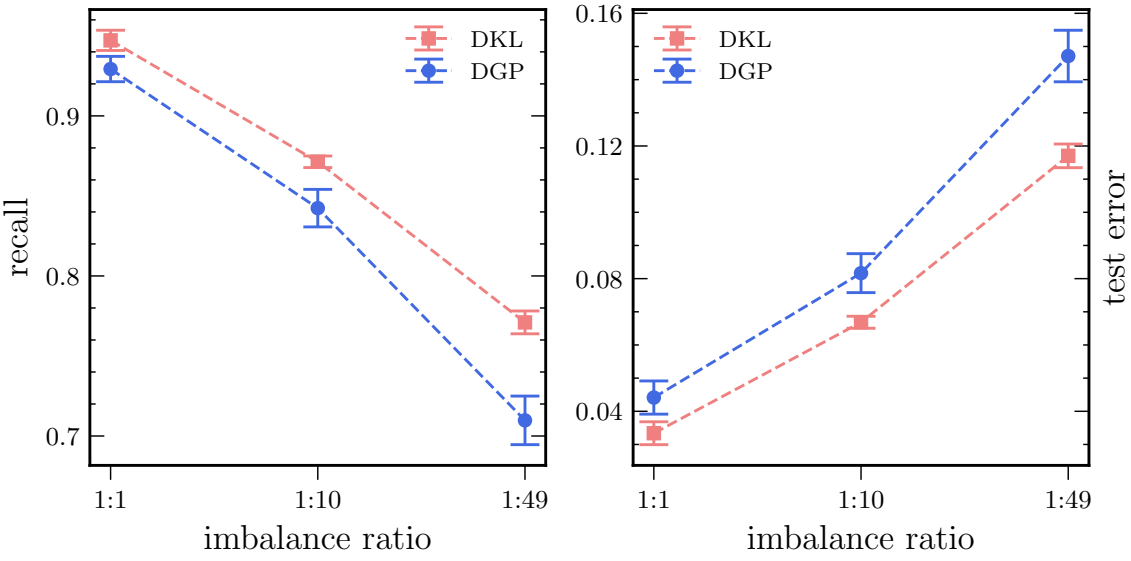

Figure 4. Effect of training with imbalanced dataset on *sensitivity*, and test error.

6.1 Effect on prediction

In order to check the effect of the imbalance on the performance of the classifiers, they are tested on the same balanced test set as in section 4 after being trained on three different datasets, each with a specific ρ. To obtain the results in figure 4, we sample 200 predictions for each instance of the balanced test set (396 instances) in each scenario.[18] We first investigate the influence of ρ on the *recall* of the learners. The *sensitivity* of the models decreases with an increasing imbalance ratio as shown in the left panel in figure 4. This can be accounted for by the fact that the larger number of negative classes in the training dataset causes the models to be biased against the positive ones at test time. The bias gets stronger with larger number of the majority class. For DGP model, the standard deviation of the predictions increases with ρ. Whereas the standard deviation of the DKL predictions is less sensitive to the increasing number of the majority class in the training sample. Similar trend is also observed with the test error as a function of the imbalance ratio (see right panel in figure 4). Both models are more prone to error in their predictions with higher imbalance ratio. The standard deviation of the test error from DGP predictions is more impacted by the bias of the training set.

[18]Training with a dataset with a specific imbalance ratio.

JCAP10(2022)016

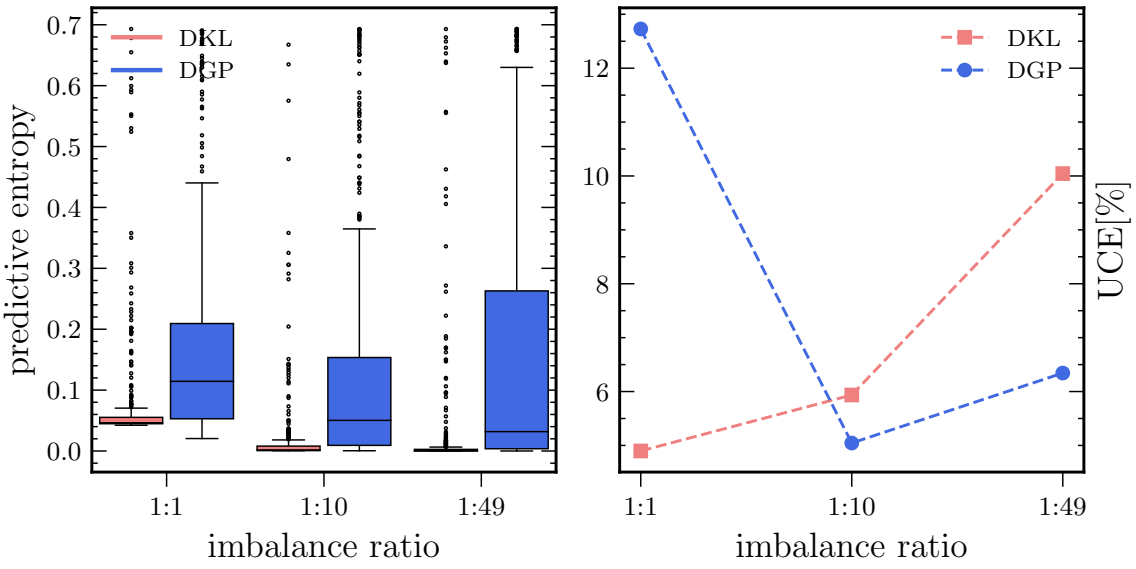

Figure 5. Effect of training with imbalanced dataset on the predictive entropy. *Left*: distribution of the predictive entropy related to each model for each imbalance ratio. *Right*: uncertainty calibration UCE corresponding to each model as a function of the imbalance ratio.

6.2 Effect on uncertainty

In a similar way, we also analyze the effect of the bias on the predictive uncertainty by considering the three cases[19] with different imbalance ratios. The left panel of figure 5 shows the variation of the distribution of the uncertainty as a function of ρ, and the resulting uncertainty calibration for each case is presented on the right panel of figure 5. Using the same balanced test set, it appears that the uncertainty produced by DKL model improves as the number of majority class increases in the training data. This is indicated by smaller *IQR* and lower median value as ρ increases. However, the uncertainty calibration obtained from DKL model degrades as a function of an increasing imbalance ratio of the training dataset. This implies that although the uncertainty seemingly gets better with the increasing number of majority class in the training data, its calibration gets worse. When considering the whole dataset (50000 instances, with ρ = 1:49) to train DKL model, the resulting UCE on the balanced test set reaches 10%.

For the DGP model, the median value of the distribution of its corresponding predictive entropy decreases as the number of the negative classes in the training set increases but the *IQR* gets smaller when the imbalance ratio of the training data is 1:10 and is the largest when the model is trained on the whole dataset (1:49) (see left panel in figure 5). Interestingly, the uncertainty produced by DGP is well calibrated (UCE = 5.044%) when the bias ρ = 1:10, but the calibration relatively degrades (UCE = 6.342%) when the DGP model is trained on the entire dataset.

[19]Training set with ρ = 1:1, 1:10, 1:49.

JCAP10(2022)016

7 BALD

In general, in order to achieve good generalization in the case of image classification, neural network needs to be trained with a substantial amount of the data, e.g. MNIST dataset [40] has 60000 instances. In production, labeling examples that will be used to train neural networks for a given task, may be expensive and active learning can be used to train the models with an optimized number of instances while achieving good generalization. In this section, we demonstrate the use of active learning approach to classify pulsar candidates, given the relatively small size of the training set.

Assuming there is a relatively large dataset with only a small number of labeled data points. In active learning, the training starts with that small amount of data. At the end of one step, which amounts to training a model for a specific number of epochs, an acquisition function selects a number of unlabeled data points from the pool[20] and asks an "*oracle*" to label them. The newly labeled data points are added to the training set used in the previous active learning step and a new step starts with the updated training set. The amount of training set increases at each step until convergence. There are many active learning approaches but we make use of Bayesian active learning [41, 42] which is scalable to both high dimensional data and large dataset.

In general, the "*oracle*" that is requested to label the queried data points from the pool is a human expert, but in our work the data points are already labeled such that they are retrieved from the pool using the indices which are selected by the acquisition function. To query new data points from the pool, we consider Bayesian Active Learning by Disagreement (BALD) [43] as acquisition function. The new points that are selected are those for which the mutual information between predictions and model posterior are maximized, according to

$$X^* = \text{argmax}_{X \in \mathcal{D}_{\text{pool}}} \mathcal{I}[y, \boldsymbol{\theta} | X, \mathcal{D}_{\text{train}}], \tag{7.1}$$

where $\mathcal{D}_{\text{pool}}$ and $\mathcal{D}_{\text{train}}$ are the sets of points in the pool and a training set at a given step respectively. The mutual information is given by [43]

$$\mathcal{I}[y, \boldsymbol{\theta} | X, \mathcal{D}_{\text{train}}] = \mathcal{H}[y | X, \mathcal{D}_{\text{train}}] - \mathbb{E}_{\boldsymbol{\theta} \sim p(\boldsymbol{\theta} | \mathcal{D}_{\text{train}})}[\mathcal{H}[y | X, \boldsymbol{\theta}]]. \tag{7.2}$$

The mutual information in equation (7.2) can also be used to estimate the epistemic uncertainty [34], such that the selection criterion in equation (7.1) can be interpreted as a way to seek for X^* that maximize the epistemic uncertainty.

In our setup, we split the balanced data (1990 examples) into 80% and 20% which constitute the pool and validation set respectively. The initial dataset for the first active learning step is composed of 100 instances from the pool and at each subsequent step 30 points with the maximum mutual information are queried by the acquisition function from the pool and added to the training set for the next step. At each step, the model is trained for 20 epochs and we consider a batch size of 16. The number of active learning steps is 5, such that the total number of data points used for the training is 220[21] which amounts to only about 14% of the pool data. The model considered for the active learning process is similar to the feature extractor used in DKL model, but batch normalization is added after each convolutional layer before ReLU activation and another dense layer is added to output the predictions. Another key component is the dropout layer with a probability of 0.2 before

[20] The large set of unlabeled data.

[21] This is given by $100 + 4 \times 30$.

JCAP10(2022)016

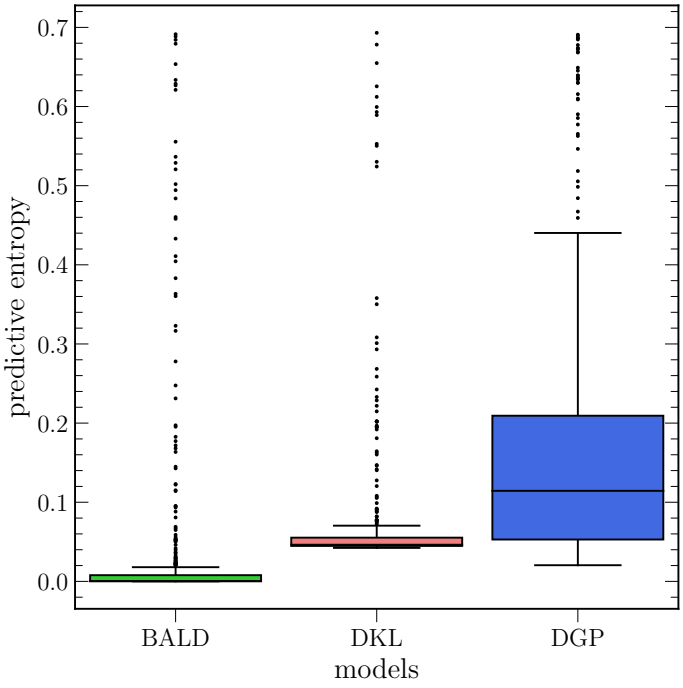

Figure 6. Comparing the predictive entropy obtained from all models.

	UCE [%]
DGP	12.728
DKL	4.897
BALD	3.118

Table 4. Uncertainty Calibration Error obtained from all models.

the output layer as it plays a crucial role in computing the uncertainty during the training (MC Dropout model [21]). It is worth mentioning that, for illustration, the choice of the dropout rate is based on the value used in [21] and is likely to be suboptimal for this specific task. However, for future work, we will optimize the hypeparameters for the active learning by using Bayesian optimization. For the implementation, we use BAAL library [42].

The results obtained from using the BALD method to train the neural network based classifier is shown in table 2. Overall the performance of the BALD method is comparable with those of DGP and DKL models, although its *sensitivity* is relatively lower. However, provided the amount of data used to train the model, the results look very promising. By construction, the uncertainty can be estimated using the dropout layer during test time. To obtain the predictive entropy distribution, shown in figure 6, we also run 200 forward passes. As indicated in figure 6, the model trained with BALD is more confident about its predictions, i.e. lower median value and smaller *IQR* compared to DGP and DKL, and its corresponding UCE, which is the lowest (see table 4), indicates that the model produces well calibrated uncertainty.

JCAP10(2022)016

8 Conclusion

This work has demonstrated the use of deep probabilistic learning to identify pulsars based on its features composed of 2D dispersion measure, sub-bands and sub-integrations. To this end, we have made use of a Deep Gaussian Process (DGP) which comprises two layers of Gaussian Process, a Deep Kernel Learning (DKL) which utilizes a neural network capacity to extract the features to be fed to the kernel of a Gaussian Process layer. Apart from highlighting the predictive power of the probabilistic methods used in this study, we have also estimated the uncertainty in their predictions and shown how good the calibration of the uncertainty is. The original dataset is strongly biased against the positive class with an imbalance ratio of $\rho = 1{:}49$. To deal with the potential impact of the bias on the models, we have performed undersampling on the majority class (via random sampling) and the models have been trained using well balanced dataset. Nevertheless, we have analyzed the effect of the imbalance ratio on the performance of the classifiers and the resulting predictive uncertainty. Provided that the number of pulsars available for training is relatively small in general, which can pose an issue for deep network to generalize well, we have explored the possibility of using a convolutional neural network based classifier which is trained via active learning whose acquisition function is Bayesian Active Learning by Disagreement (BALD).

The two models, DGP and DKL, exhibit great capability of differentiating pulsars from non-pulsars as indicated by their *roc-auc* > 0.98. The small difference in their *recall* suggests that DGP is more likely to misclassify the positive class but overall the classifiers generalize well. The performance of our methods is comparable to that achieved in previous studies. The lower median value of its predictive uncertainty distribution (0.046 nats) and the smaller interquartile range (*IQR*) indicate that DKL shows more confidence in its predictions compared to DGP model which has larger *IQR* and higher median value of predictive entropy distribution (0.114 nats). This can be attributed to DKL's higher capacity, which is achieved by utilizing convolutional neural network (CNN) based feature extractor to encode the salient features from the inputs. To assess the calibration of the uncertainty produced by the models, the expected Uncertainty Calibration Error (UCE) has been used. Results show that the uncertainty produced by DKL model, achieving UCE = 4.897%, is better calibrated than that estimated by DGP with UCE = 12.738%. The comparison between the variance of the ratio of true positive/true negative $\sigma^2_{FP/FN}$ with that of the ratio of false positive/false negative $\sigma^2_{FP/FN}$ is also considered to evaluate the quality of the uncertainty. For all models, $\sigma^2_{FP/FN}$ is larger than $\sigma^2_{TP/TN}$ by about two orders of magnitude overall, indicating that reliable uncertainty can be obtained from the models.

By considering three scenarios, each with a given bias $\rho = 1{:}1, 1{:}10, 1{:}49$ of the training dataset, we have found that the predictive power of the models (DKL and DGP) degrades with an increasing imbalance ratio of the training dataset, as shown by a decreasing *recall* and an increasing test error with an increase in the imbalance ratio. It has been found that as the number of majority class in the training dataset increases, DKL model becomes more confident in its predictions (lower median value of predictive entropy distribution and smaller *IQR*), whereas the resulting uncertainty calibration gets poorer (increasing UCE). Interestingly, DGP model appears to produce well calibrated uncertainty with a relatively small imbalance $\rho = 1{:}10$ in the training dataset. However with $\rho = 1{:}49$, the *IQR* of the resulting predictive uncertainty distribution is the largest, and the corresponding UCE reaches 6.324%. It can be argued that both DKL and DGP can still perform reasonably well when trained on dataset in which the number of majority class is less than $10\times$ that of the minority class.

JCAP10(2022)016

Using BALD method, a CNN based classifier with a dropout layer is trained with only using 220 instances in order to achieve generalization. This optimized number of training data points results from the learner requesting, via the acquisition function, data points with maximum mutual information (or epistemic uncertainty) to be included in the training dataset for the next active learning step. As a whole, the performance of the CNN model obtained via the BALD approach is comparable to that of the other two models (DKL and DGP). Its *recall*, which is relatively lower (0.914), denotes that it is more likely to misclassify the positive class. Compared to the other models, the CNN model trained via BALD is the most confident in its predictions, as shown by the median value (the lowest) and *IQR* (the smallest) of its predictive entropy distribution, and its lowest UCE (3.118%) suggests that its uncertainty prediction is best calibrated.

Despite the good performance of DGP to identify pulsars in this work, a potential constraint that it may suffer from, within the context of image classification, is the dimension of the inputs. In our case for instance, the dimension is 32×32, as opposed to 64×64 in [31] and [32]. As the resolution of an image gets smaller, more information is lost. It would then be interesting to investigate in a future work whether the results presented here can be recovered with an image resolution of 64×64. It should give us some insights into the impact of the resolution of the inputs on the performance of DGP. However, as shown by [30], Deep Kernel Learning, apart from its scalability, can outperform CNN in classification on dataset like MNIST and CIFAR-10. This is owing to the capacity of the feature extractor (CNN based) it uses, combined with a GP layer. Depending on a given task, the capacity of the model can be adjusted by simply adding or removing convolutional layers.

Acknowledgments

SA acknowledges financial support from the *South African Radio Astronomy Observatory* (SARAO). SA is very grateful to Anna Scaife for the recommendations during those fruitful discussions.

A Gaussian Process (GP)

Known as a non-parametric model, the main assumption in a Gaussian Process is that a finite set of real-valued function $\boldsymbol{f}$ — random variables — evaluated at inputs $\mathbf{X}$ has a joint Gaussian distribution. A Gaussian Process is fully described by the mean $\mu(\mathbf{X})$ and the covariance $K(\mathbf{X}, \mathbf{X}')$ such that [44]

$$\begin{aligned} \mu(\mathbf{X}) &= \mathbb{E}[\boldsymbol{f}(\mathbf{X})] \\ K(\mathbf{X}, \mathbf{X}') &= \mathbb{E}[(\boldsymbol{f}(\mathbf{X}) - \mu(\mathbf{X}))(\boldsymbol{f}(\mathbf{X}') - \mu(\mathbf{X}'))]. \end{aligned} \tag{A.1}$$

A GP prior is then defined as

$$\boldsymbol{f}(\mathbf{X}) \sim \mathcal{GP}(\mu(\mathbf{X}), K(\mathbf{X}, \mathbf{X}')), \tag{A.2}$$

and setting the mean function to zero for convenience, we have that $p(\boldsymbol{f}) = \mathcal{N}(\mathbf{0}, K(\mathbf{X}, \mathbf{X}'))$. In reality, the observed functions $\mathbf{y}$ are noisy realizations of the latent functions $\boldsymbol{f}$ such that

$$\mathbf{y} = \boldsymbol{f} + \boldsymbol{\epsilon}, \tag{A.3}$$

JCAP10(2022)016

where $\boldsymbol{\epsilon}$ is a Gaussian noise with a variance σ^2, therefore by adding the variance in quadrature the distribution of the observed functions reads

$$p(\mathbf{y}) = \mathcal{N}(\mathbf{0}, K(\mathbf{X}, \mathbf{X}') + \sigma^2 \mathbf{I}). \tag{A.4}$$

Given that any subset of a collection of random variables which is jointly Gaussian distributed is also a Gaussian distribution, we have the joint distribution of the observations and some test functions $\boldsymbol{f}_*$ at some test inputs $\mathbf{X}_*$ [44]

$$\begin{bmatrix} \mathbf{y} \\ \boldsymbol{f}_* \end{bmatrix} = \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} K(\mathbf{X}, \mathbf{X}) + \sigma^2 \mathbf{I}) & K(\mathbf{X}, \mathbf{X}_*) \\ K(\mathbf{X}_*, \mathbf{X}) & K(\mathbf{X}_*, \mathbf{X}_*) \end{bmatrix}\right), \tag{A.5}$$

where $K(\mathbf{X}, \mathbf{X}_*)$ and $K(\mathbf{X}_*, \mathbf{X}_*)$ are the cross-covariance matrix between training points and test points and the covariance matrix of the test points respectively. Exploiting the property of Gaussian conditionals, we obtain the predictive distribution conditioned on the training data and the test points

$$p(\boldsymbol{f}_*|\mathbf{y}, \mathbf{X}, \mathbf{X}_*) = \mathcal{N}(\mu_*, \tilde{K}), \tag{A.6}$$

where

$$\mu_* = K(\mathbf{X}_*, \mathbf{X})[K(\mathbf{X}, \mathbf{X}) + \sigma^2 \mathbf{I}]^{-1}\, \mathbf{y}, \tag{A.7}$$

and

$$\tilde{K} = K(\mathbf{X}_*, \mathbf{X}_*) - K(\mathbf{X}_*, \mathbf{X})[K(\mathbf{X}, \mathbf{X}) + \sigma^2 \mathbf{I}]^{-1} K(\mathbf{X}, \mathbf{X}_*). \tag{A.8}$$

There is a variety of kernel functions that can be chosen to build the covariance matrix $K(\mathbf{X}, \mathbf{X}')$ but in our analyses we opt for the widely used radial basis function (RBF), given by

$$k(x, x') = \sigma_f^2 \exp\left(-\frac{(x - x')^2}{2\ell}\right), \tag{A.9}$$

in which the free parameters σ_f^2 and ℓ are the signal variance and the length-scale respectively. The latter denotes the scale over which the function $\boldsymbol{f}$ significantly changes. These hyperparameters are chosen in such a way as to optimize the log marginal likelihood given in eq. (A.4)

$$\begin{aligned} \log p(\mathbf{y}) = & -\frac{1}{2}\mathbf{y}^T [K(\mathbf{X}, \mathbf{X}) + \sigma^2 \mathbf{I}]^{-1} \mathbf{y} \\ & -\frac{1}{2}\log|K(\mathbf{X}, \mathbf{X}) + \sigma^2 \mathbf{I}| - \frac{n}{2}\log(2\pi), \end{aligned} \tag{A.10}$$

where n is the number of training data points. Ref. [44] provides a practical implementation of a GP regression. In a classification task, the objective is to predict a category of a given input. Therefore, using GP for classification amounts to *squashing* the latent functions $\boldsymbol{f}$ over which the GP prior is defined with a logistic function such as

$$p(y_i = 1|f_i) = \frac{1}{1 + \exp(-f_i)}, \tag{A.11}$$

where we assume we only have two classes, 0 and 1. In a noise-free GP, the predictive distribution which is conditioned on the training data and the input at some test point is given by

$$p(y_* = 1|X_*, \mathbf{X}, \mathbf{y}) = \int \mathrm{d}f_*\, p(y_* = 1|f_*) p(f_*|X_*, \mathbf{X}, \mathbf{y}), \tag{A.12}$$

JCAP10(2022)016

where the latent function evaluated at the test point f_* is averaged out. The second term in eq. (A.12) is the distribution over the latent functions at a test point and is computed by marginalizing over the latent functions $\boldsymbol{f}$ at the training data

$$p(f_*|X_*, \mathbf{X}, \mathbf{y}) = \int \mathrm{d}\boldsymbol{f}\; p(f_*|X_*, \mathbf{X}, \mathbf{y}, \boldsymbol{f}) p(\boldsymbol{f}|\mathbf{X}, \mathbf{y}), \tag{A.13}$$

where the second term $p(\boldsymbol{f}|\mathbf{X}, \mathbf{y})$ is the posterior distribution. Unfortunately, unlike GP regression where the likelihood is Gaussian, the posterior distribution in eq. (A.13) is intractable due to the logistic likelihood. The methods, fully described in [44], consist of approximating the posterior distribution by a Gaussian distribution, namely Laplace Approximation [45] and Expectation Propagation [46]. The predictive distribution in a GP involves a matrix inversion of the covariance matrix $K(\mathbf{X}, \mathbf{X}')$. This operation has a cubic complexity $\mathcal{O}(n^3)$. Therefore, given the nonscalability, dealing with large dataset using GP is quite challenging. To circumvent the issue with the computational complexity, [47] proposed a sparse GP method by using "*inducing points*" $\mathbf{Z}$ which can be points in the input space corresponding to real valued functions $\mathbf{u}$. The central idea is to select $\mathbf{Z}$ (the number of these inducing variables is $m < n$) , which are fewer than the original data $\mathbf{X}$, such that they capture the characteristics of the functions. Computation of the predictive distribution, by using $\mathbf{Z}$, scales as $\mathcal{O}(m^3)$. Ref. [47], in their approach, approximated the marginal likelihood by

$$p(\mathbf{y}) \sim \mathcal{N}(\mathbf{0}, Q_{nn} + \sigma^2 \mathbf{I}), \tag{A.14}$$

where $Q_{nn} = \mathrm{diag}[K_{nn} - K_{nm}K_{mm}^{-1}K_{mn}] + K_{nm}K_{mm}^{-1}K_{mn}$ is low-rank approximation of the original kernel function K_{nn}. The overfitting that the setup is prone to, due to the fact that the inducing variables $\mathbf{Z}$ are now part of the hyperparameters to be optimized, led [48] to adopt a different approach which approximates the posterior with a variational distribution which has $\mathbf{Z}$ as part of the variational parameters. Ref. [48] then prescribed a lower bound on the marginal likelihood

$$\log p(\mathbf{y}) \geq \mathcal{N}(\mathbf{0}, Q_{nn} + \sigma^2 \mathbf{I}) - \frac{1}{2\sigma^2}\mathrm{tr}(K_{nn} - Q_{nn}), \tag{A.15}$$

where the second term involving the trace is known as a regularizer preventing from overfitting. In [49], they improved on the prescription in [48] by proposing a more scalable bound such that stochastic variational inference is applicable to infer the hyperparameters.

References

[1] R.A. Hulse and J.H. Taylor, *Discovery of a pulsar in a binary system*, *Astrophys. J. Lett.* **195** (1975) L51 [INSPIRE].

[2] J.H. Taylor and J.M. Weisberg, *A new test of general relativity: gravitational radiation and the binary pulsar PS R* 1913 + 16, *Astrophys. J.* **253** (1982) 908 [INSPIRE].

[3] LIGO Scientific and Virgo collaborations, *Observation of gravitational waves from a binary black hole merger*, *Phys. Rev. Lett.* **116** (2016) 061102 [arXiv:1602.03837] [INSPIRE].

[4] I.H. Stairs, S.E. Thorsett, J.H. Taylor and A. Wolszczan, *Studies of the relativistic binary pulsar PS R B*1534 + 12*: I. Timing analysis*, *Astrophys. J.* **581** (2002) 501 [astro-ph/0208357] [INSPIRE].

[5] M. Krämer et al., *Tests of general relativity from timing the double pulsar*, *Science* **314** (2006) 97 [astro-ph/0609417] [INSPIRE].

JCAP10(2022)016

[6] X. Zhang et al., *Constraints of general screened modified gravities from comprehensive analysis of binary pulsars*, *Astrophys. J.* **874** (2019) 121 [arXiv:1902.08374] [INSPIRE].

[7] J.A. Pons, D. Viganò and N. Rea, *A highly resistive layer within the crust of X-ray pulsars limits their spin periods*, *Nature Phys.* **9** (2013) 431.

[8] M. Krämer, *Pulsars as probes of gravity and fundamental physics*, *Int. J. Mod. Phys. D* **25** (2016) 1630029 [arXiv:1606.03843] [INSPIRE].

[9] X. Siemens et al., *Physics beyond the Standard Model with pulsar timing arrays*, arXiv:1907.04960 [INSPIRE].

[10] NANOGRAV collaboration, *The NANOGrav* 11*-year data set: pulsar-timing constraints on the stochastic gravitational-wave background*, *Astrophys. J.* **859** (2018) 47 [arXiv:1801.02617] [INSPIRE].

[11] NANOGRAV collaboration, *Multimessenger gravitational-wave searches with pulsar timing arrays: application to* 3*C* 66*B using the NANOGrav* 11*-year data set*, *Astrophys. J.* **900** (2020) 102 [arXiv:2005.07123] [INSPIRE].

[12] R. Smits, M. Krämer, B. Stappers, D.R. Lorimer, J. Cordes and A. Faulkner, *Pulsar searches and timing with the Square Kilometre Array*, *Astron. Astrophys.* **493** (2009) 1161 [arXiv:0811.0211] [INSPIRE].

[13] M. Bailes et al., *The MeerKAT telescope as a pulsar facility: system verification and early science results from MeerTime*, *Publ. Astron. Soc. Austral.* **37** (2020) e028.

[14] R.P. Eatough et al., *Selection of radio pulsar candidates using artificial neural networks*, *Mon. Not. Roy. Astron. Soc.* **407** (2010) 2443 [arXiv:1005.5068] [INSPIRE].

[15] S.D. Bates et al., *The high time resolution universe survey VI: an artificial neural network and timing of* 75 *pulsars*, *Mon. Not. Roy. Astron. Soc.* **427** (2012) 1052 [arXiv:1209.0793] [INSPIRE].

[16] V. Morello, E.D. Barr, M. Bailes, C.M. Flynn, E.F. Keane and W. van Straten, *SPINN: a straightforward machine learning solution to the pulsar candidate selection problem*, *Mon. Not. Roy. Astron. Soc.* **443** (2014) 1651 [arXiv:1406.3627] [INSPIRE].

[17] S. Bethapudi and S. Desai, *Separation of pulsar signals from noise using supervised machine learning algorithms*, *Astron. Comput.* **23** (2018) 15.

[18] W.W. Zhu et al., *Searching for pulsars using image pattern recognition*, *Astrophys. J.* **781** (2014) 117 [arXiv:1309.0776] [INSPIRE].

[19] A. Nguyen, J. Yosinski and J. Clune, *Deep neural networks are easily fooled: high confidence predictions for unrecognizable images*, in 2015 *IEEE conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, (2015), p. 427.

[20] M. Hein, M. Andriushchenko and J. Bitterwolf, *Why ReLU networks yield high-confidence predictions far away from the training data and how to mitigate the problem*, in 2019 *IEEE/CVF conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, (2019), p. 41.

[21] Y. Gal and Z. Ghahramani, *Dropout as a bayesian approximation: representing model uncertainty in deep learning*, in *International conference on machine learning*, PMLR, (2016), p. 1050.

[22] M.J. Keith et al., *The high time resolution universe pulsar survey I: system configuration and initial discoveries*, *Mon. Not. Roy. Astron. Soc.* **409** (2010) 619 [arXiv:1006.5744] [INSPIRE].

[23] N. Metropolis, A.W. Rosenbluth, M.N. Rosenbluth, A.H. Teller and E. Teller, *Equation of state calculations by fast computing machines*, *J. Chem. Phys.* **21** (1953) 1087 [INSPIRE].

JCAP10(2022)016

[24] W.K. Hastings, *Monte Carlo sampling methods using Markov chains and their applications*, *Biometrika* **57** (1970) 97.

[25] S. Chib and E. Greenberg, *Understanding the Metropolis-Hastings algorithm*, *Amer. Statist.* **49** (1995) 327.

[26] M.J. Beal, *Variational algorithms for approximate Bayesian inference*, Ph.D. thesis, University College, London, U.K. (2003).

[27] D. Wingate and T. Weber, *Automated variational inference in probabilistic programming*, arXiv:1301.1299.

[28] A. Damianou and N. Lawrence, *Deep gaussian processes*, in *Artificial intelligence and statistics*, PMLR, (2013), p. 207.

[29] E. Bingham et al., *Pyro: deep universal probabilistic programming*, *J. Machine Learn. Res.* **20** (2019) 973.

[30] A.G. Wilson, Z. Hu, R.R. Salakhutdinov and E.P. Xing, *Stochastic variational deep kernel learning*, in *Advances in neural information processing systems*, NeurIPS proceedings, (2016), p. 2586.

[31] P. Guo et al., *Pulsar candidate classification using generative adversary networks*, *Mon. Not. Roy. Astron. Soc.* **490** (2019) 5424.

[32] Y.-C. Wang, M.-T. Li, Z.-C. Pan and J.-H. Zheng, *Pulsar candidate classification with deep convolutional neural networks*, *Res. Astron. Astrophys.* **19** (2019) 133.

[33] S. Andrianomena, M. Rafieferantsoa and R. Davé, *Classifying galaxies according to their H I content*, *Mon. Not. Roy. Astron. Soc.* **492** (2020) 5743.

[34] Y. Gal, *Uncertainty in deep learning*, Ph.D. thesis, University of Cambridge, Cambridge, U.K. (2016).

[35] A. Kendall and Y. Gal, *What uncertainties do we need in bayesian deep learning for computer vision?*, *Adv. Neural Inf. Proc. Syst.* **30** (2017) 5580.

[36] D. Mohan, A.M.M. Scaife, F. Porter, M. Walmsley and M. Bowles, *Quantifying uncertainty in deep learning approaches to radio galaxy classification*, *Mon. Not. Roy. Astron. Soc.* **511** (2022) 3722 [arXiv:2201.01203] [INSPIRE].

[37] M.-H. Laves, S. Ihler, K.-P. Kortmann and T. Ortmaier, *Well-calibrated model uncertainty with temperature scaling for dropout variational inference*, arXiv:1909.13550.

[38] A. Krizhevsky, V. Nair and G. Hinton, *Cifar-10 (Canadian institute for advanced research)*, http://www.cs.toronto.edu/~kriz/cifar.html.

[39] Y. Li et al., *Deep Bayesian Gaussian processes for uncertainty estimation in electronic health records*, *Sci. Rept.* **11** (2021) 20685.

[40] Y. LeCun, *The MNIST database of handwritten digits*, http://yann.lecun.com/exdb/mnist/, (1998).

[41] Y. Gal, R. Islam and Z. Ghahramani, *Deep bayesian active learning with image data*, in *International conference on machine learning*, PMLR, (2017), p. 1183.

[42] P. Atighehchian, F. Branchaud-Charron and A. Lacoste, *Bayesian active learning for production, a systematic study and a reusable library*, arXiv:2006.09916.

[43] N. Houlsby, F. Huszár, Z. Ghahramani and M. Lengyel, *Bayesian active learning for classification and preference learning*, arXiv:1112.5745.

[44] C. Rasmussen and C. Williams, *Gaussian processes for machine learning*, MIT Press, Cambridge, MA, U.S.A. (2006).

JCAP10(2022)016

[45] C.K. Williams and D. Barber, *Bayesian classification with gaussian processes*, *IEEE Trans. Pattern Anal. Machine Intell.* **20** (1998) 1342.

[46] T.P. Minka, *The ep energy function and minimization schemes*, https://tminka.github.io/papers/ep/minka-ep-energy.pdf, (2001).

[47] E. Snelson and Z. Ghahramani, *Sparse Gaussian processes using pseudo-inputs*, *Adv. Neur. Inf. Proc. Syst.* **18** (2005) 1257.

[48] M.K. Titsias, *Variational model selection for sparse Gaussian process regression*, report, University of Manchester, Manchester, U.K. (2009).

[49] J. Hensman, N. Fusi and N.D. Lawrence, *Gaussian processes for big data*, arXiv:1309.6835.

JCAP10(2022)016