Received: 25 September 2019 | Revised: 19 June 2020 | Accepted: 6 September 2020

DOI: 10.1111/coin.12411

Check for updates

SPECIAL ISSUE ARTICLE

Computational Intelligence WILEY

Estimating uncertainty in deep learning for reporting confidence to clinicians in medical image segmentation and diseases detection

Biraja Ghoshal[1] | **Allan Tucker**[1] | **Bal Sanghera**[2] | **Wai Lup Wong**[2]

[1]Department of Computer Science, Brunel University, London, UK

[2]Paul Strickland Scanner Centre, Mount Vernon Hospital, London, UK

Correspondence
Biraja Ghoshal, Department of Computer Science, Brunel University, London, UK.
Email: biraja.ghoshal@brunel.ac.uk

Abstract

Deep learning (DL), which involves powerful black box predictors, has achieved a remarkable performance in medical image analysis, such as segmentation and classification for diagnosis. However, in spite of these successes, these methods focus exclusively on improving the accuracy of point predictions without assessing the quality of their outputs. Knowing how much confidence there is in a prediction is essential for gaining clinicians' trust in the technology. In this article, we propose an uncertainty estimation framework, called MC-DropWeights, to approximate Bayesian inference in DL by imposing a Bernoulli distribution on the incoming or outgoing weights of the model, including neurones. We demonstrate that by decomposing predictive probabilities into two main types of uncertainty, aleatoric and epistemic, using the Bayesian Residual U-Net (BRUNet) in image segmentation. Approximation methods in Bayesian DL suffer from the "mode collapse" phenomenon in variational inference. To address this problem, we propose a model which Ensembles of Monte-Carlo DropWeights by varying the DropWeights rate. In segmentation, we introduce a predictive uncertainty estimator, which takes the mean of the standard deviations of the class probabilities associated with every

This is an open access article under the terms of the Creative Commons Attribution License, which permits use, distribution and reproduction in any medium, provided the original work is properly cited.
© 2020 The Authors. *Computational Intelligence* published by Wiley Periodicals LLC.

class. However, in classification, we need an alternative approach since the predictive probabilities from a forward pass through the model does not capture uncertainty. The entropy of the predictive distribution is a measure of uncertainty, but its exponential depends on sample size. The plug-in estimate in mutual information is subject to sampling bias. We propose Jackknife resampling, to correct for sample bias, which improves estimating uncertainty quality in image classification. We demonstrate that our deep ensemble MC-DropWeights method, using the bias-corrected estimator produces an equally good or better result in both quantified uncertainty estimation and quality of uncertainty estimates than approximate Bayesian neural networks in practice.

KEYWORDS

bias-corrected uncertainty estimation, classification, deep learning, dropweights, ensembles, medical image segmentation

1 | INTRODUCTION

Recently, deep learning (DL), which has achieved state-of-the-art performance across applied sciences (biology, physics, chemistry), engineering (autonomous driving), and advancing medical diagnostics such as lung disease classification and metastasis detection for breast cancer and magnetic resonance imaging (MRI), PET/CT imaging.[1] In applications of computer-based medical systems, an erroneous decision, especially in life-threatening situations, can have fatal consequences.

Uncertainty is the most common and unavoidable feature of DL tasks. DL models produce a point estimate, which is often incorrectly interpreted as a probability of model confidence. In reality, it is a normalized network output for a given class relative to the other classes. This cannot explain the model's overall confidence and leads to generalization issues, such as over-confidence in their predictions and unpredictable behavior on out-of-distribution (OOD) samples. For example, DL-based diagnosis of MRI images of brain tumours needs a way to express the uncertainty of an image in the same way as a doctor may express ambiguity and ask for experts help for further inspection and correction.

Therefore, it is not sufficient to depend on the classification or regression score alone from DL models. In order to address this problem, the deep neural networks need to provide uncertainty estimation as an additional insight to point prediction to improve the reliability in the decision-making process.

Estimating uncertainty in deep neural networks is a challenging and yet unsolved problem. Bayesian neural networks (BNNs) learn a distribution over each of the network's weight parameters[2] and are currently considered state-of-the-art for estimating predictive uncertainty. There are many methods proposed for quantifying uncertainty or confidence estimates

approximated by Markov chain Monte Carlo (MCMC), variational inference (VI) including deep ensembles.[3]

In medical image segmentation such as cell detection or localization to be meaningful, tolerance must typically be much tighter (eg, >50% overlap with the actual bounding box). Based on the input medical image, a network can be certain with high or less confident about its decision, indicated by the predictive posterior distribution. However, predictive uncertainty in DL results from two separate forms of uncertainty:[4,5]

1. Model uncertainty or epistemic uncertainty (EU) accounts for uncertainty in the model parameters due to the lack of training data. EU associated with the model reduces as the training data size increases.
2. Data uncertainty or aleatoric uncertainty (AU) accounts for noise inherent in the observations due to class overlap, label noise, homoscedastic and heteroscedastic noise, which cannot be reduced even if more data were to be collected unless it is possible to observe all explanatory variables with increased precision.

We would like to note that, in our conference paper,[6] we proposed to quantify uncertainty of segmentation in DL by decomposing predictive uncertainty into the correct interpretation of AU and EU and provided additional insights into the corresponding medical image segmentation with point prediction. However, we need an alternative approach in classification, since the predictive probabilities from a forward pass through the model does not capture uncertainty. In this article, we introduce deep ensembles of Monte-Carlo DropWeights to address the "mode collapse" phenomenon in VI as well as propose a method to estimate bias-corrected uncertainty leveraging Jackknife resampling method to improve estimating uncertainty in classification in DL. We also propose metrics to quantify the uncertainty estimates and quality of estimated uncertainty. We show that our method produces as good if not better results than the recently proposed approximate BNNs technique.

Our objective is not to achieve the state-of-the-art performance in DL, but rather to define a framework for estimating uncertainty in DL and evaluate the usefulness of predictive uncertainty for segmentation and classification to avoid overconfident, incorrect predictions during decision making in computer-based medical systems.

2 | RELATED RESEARCH

An artificial neural network is a parameterized function. DL systems are neural network models with architectural and algorithmic innovations (eg, many convolution layers, activation functions, better initialization and learning rates, Dropout, batch normalization). However, DL is generally very data-hungry, computationally intensive to train and deploy, poor at representing uncertainty, easily fooled by adversarial situations, uninterpretable (black-box), lacking in transparency, and lacks trust in the model outcome.

Nevertheless, DL provides a framework for a powerful class of flexible, rich nonlinear models for classification and prediction, for scalable learning using stochastic approximations. DL models are most commonly trained with maximum likelihood estimate (MLE) or maximum a posterior (MAP) procedures, which only yields point estimates of the parameters with fixed, deterministic values. However, this is unable to provide a notion of uncertainty in the parameters, inherent stochastic noise and model specification. In reality, a model can provide overconfident

predictions or incorrect classifications with a high confidence, especially when a model is trained on data outside the distribution or adversarial situations. It is essential to know how confident our model is for each prediction and what a model does not know in DL systems, especially when making vital decisions in medical applications. Uncertainty in regression and classification tasks is important to improve the reliability and safety of computer-based medical systems.

The Bayesian framework provides a natural and principled way of modeling uncertainty via probability density over outcomes, which is resistant to overfitting.[7,8] There are a variety of approximations[5,7–10] that have been developed, including Laplace approximation, MCMC methods, stochastic gradient MCMC variants such as Langevin dynamics, Hamiltonian methods, including multiplicative normalizing flows, stochastic batch normalization, maximum softmax probability, heteroscedastic classifier, and learned confidence estimates, including deep ensembles to adapt neural networks to BNN. The most common approach is to replace the weight parameters of deterministic network with a prior distribution (often a Gaussian) on its weights and, instead of optimizing the network weights directly, averaging it over all possible weights (referred to as marginalization). Although these models are simple to formulate and offer stability against over-fitting, the inference is computationally intractable due to the huge computational costs, size, and nonlinearity.

Therefore, many studies have been conducted to approximate the posterior using VI.[3] All models assume independence between the individual weights. Variational Dropout[11] and VI[11] methods do not scale because they are dependent on the choice of the family of fully factorized approximate distributions and rarely rich enough to contain the exact posterior. In 2011, Graves[12] proposed a model using sampling methods to estimate a factorized posterior or a biased estimator. Blundell et al introduced Bayes By Backprop (BBP), a stochastic gradient VI algorithm to estimate uncertainty from deep Q-networks.[2]

The optimization is performed efficiently by using the generalized reparametrization trick to obtain an unbiased estimate with respect to the variational parameters. Another approach to VI is probabilistic backpropagation (PBP), which can also estimate factorized posterior based on expectation propagation.[13]

An alternative to Bayesian inference[14] ensembles of deep networks (a.k.a. the frequentist approach) to estimate predictive uncertainty based on the sample difference due to different initialization and noise in the stochastic gradients. Although this technique requires minimal hyperparameter tuning, it has to maintain several independent models and execute forward passes through all of them to calculate the variance of their output prediction to make the inference. Unlike Bayesian methods, ensembles approach are effectively sensitivity analysis of model and can not quantify uncertainty.

In 2016, Gal[3] showed that neural networks with arbitrary depth and nonlinearities, trained with Dropout, a well-known regularization technique,[15] applied before every weight layer, approximate Bayesian inference in deep Gaussian processes (marginalized over its covariance function parameters). Uncertainty estimates is obtained by training a network with Dropout and then taking Monte Carlo (MC) samples of the prediction using Dropout on at test time. The amount of noise in the input data is considered to be constant. This method captures variance of network parameters and commonly known as MC-Dropout.

It has been noted that MC-Dropout provides measures of risk, but not uncertainty.[16] More recently, very deep convolutional architectures have been proposed (eg, residual networks, etc.), with more than a hundred layers that have no Dropout layer to avoid accuracy degradation. Lewandowski[17] showed that batch normalization was a way of incorporating weight uncertainty in deep kernel learning, which corresponds to VI on the neural network weights.

Training a deep network using a batch normalization formulation of propagating uncertainty in deep kernel learning is equivalent to approximate VI in Bayesian models, which can estimate meaningful model uncertainty without any change in the overall model structure or the training procedure.

In BNNs, predictive uncertainty can be decomposed into two types of uncertainties characterized as EU, which is also known as model uncertainty, and AU, which depends on the inherent noise in the observations.[5,18]

3 | METHODOLOGY

Recently, Gal[3] proved that neural network trained with Dropout is an approximate Bayesian model. During test time, Dropout is turned on to keep the Bernoulli distribution over the network weights, whereas each forward pass through the trained neural network with Dropout generates a Monte Carlo sample from the posterior distribution. The mean of these samples can be interpreted as the prediction and the model uncertainty can be estimated by computing the variance on multiple predictions.

In this section, following Gal,[3] we briefly show that a neural network with DropWeights applied on fully connected layers for regularizing neural network to prevent over-fitting is mathematically equivalent to an approximation to the probabilistic deep Gaussian process. We then approximate Bayesian modeling via VI with a specific variational distribution to obtain uncertainties in DL.

3.1 | DropWeights in neural network

Convolutional neural networks (CNNs) in DL have shown outstanding performances in biomedical image processing. However, CNNs are prone to over-fitting when trained with small datasets. A number of techniques have been developed for regularizing neural networks, such as adding an l2 penalty on the network, Bayesian methods,[8] weight elimination and early stopping of training.[19] In deep neural networks, co-adaptation means that some neurons are highly dependent on others which significantly impacts the model performance. Overfitting can be reduced by using Dropout[15] and DropConnects,[20] to prevent complex co-adaptations on the training data. Network pruning by dropping connections has been widely studied to compress a pre-trained, fully connected neural network models. It can also reduce the network complexity and over-fitting.[21,22] BNNs is used to mitigate overfitting and can be trained with small datasets.[7,8]

The number of neurons in a human brain stays constant throughout its life, but synapse connectivity changes dramatically over time.[23] Using this fact, we have developed a technique called "DropWeights" which randomly drops connections, that is, incoming or outgoing weights are set to zeros, including drop neurones. DropWeights can be considered as the combination of generalized version of Dropout and DropConnects, and this comprises of the method used for regularizing deep neural networks. DropWeights is a kind of ensemble and approximates the output by a moment matched Gaussian, and it produces even more possible models, since there are almost always more connections than units. Figure 1 illustrates the DropWeights strategy. This DropWeights method converts a dense, fully connected neural network to dynamically sparse representations on the weights during training and test time, when DropWeights are turned on References 24 and 25.

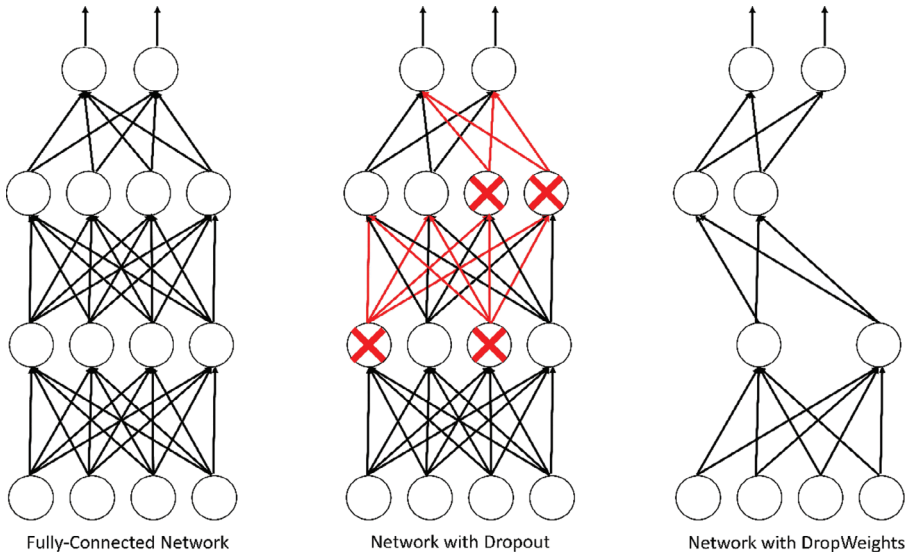

FIGURE 1 A graphical illustration of DropWeights strategy [Color figure can be viewed at wileyonlinelibrary.com]

We consider DropWeights applied to a single fully connected layer of deep neural network K_{i-1} dimensional input $X = \{x_1, x_2 \ \ldots \ x_N\}$, ith layer of neural network K_i units would output a K_i dimensional activation vectors $a_i = \sigma(W_i x)$ where W_i is the $K_{i-1} x K_i$ weight parameters including biases and $\sigma(.)$ is the nonlinear activation function.

When DropWeights is applied to the outputs of a full-connected layer, different neurons allocate different drop probabilities to enable the model to dynamically adjust the drop probability of the weight, eventually leading more sparse features of network model extraction.[25]

The feed-forward operation of neural networks with DropWeights can be described as:

$$\rho_{ij}^{(l)} = p_{drop}(y_j^{(l-1)}) \tag{1}$$

$$\mathrm{M}_{ij}^{(1)} \sim \mathrm{Bernoulli}(\rho_{ij}^{(l)}) \tag{2}$$

$$\tilde{W}_{ij}^{(l)} = \tilde{W}_{ij}^{(l)} \odot (M_{ij}^{(l)} > \rho_{ij}^{(l)}) \tag{3}$$

$$z_i^l = \sum_{j=1}^{n} \tilde{W}_{ij}^{(l)} y_j^{(l-1)} + b_i^{(l)} \tag{4}$$

$$y^{(l)} = f(z^l). \tag{5}$$

For a DropWeights layer, the output activations can be written as:

$$\sum_M f((M \odot W)x) \approx f\left(\sum_M (M \odot W)x\right), \tag{6}$$

where M is a binary mask encoding the connection information drawn independently from a Bernoulli distribution with probability p, $W_{ij}^{(l)}$ is for the connection weight between the j neuron

in the $l-1$ layer and the i neuron in the l layer; $\rho_{ij}^{(l)}$ is for the drop probability of the weight associated with the weight $W_{ij}^{(l)}$ being set to 0; $p_{drop}(.)$ is for the calculation function of weight drop probability. Hadamard product $\odot$ denotes the element-wise product of matrices. Input of the activation function is a weighted sum of Bernoulli variables and can be approximated by a Gaussian distribution. During test time, we drew samples of $z^{(l+1)}$and fed the samples into the activation function f. This represents the mixture model interpretation of DropWeights, where the output is a total of 2^M different network architectures possible, each with weight $p(M)$. Each of these correspond to some of the connections being present and some being dropped. DropWeights is functionally equivalent to an ensemble rather than a single model. The output value of the sparsity formula is in the range [0, 1.0].

3.2 | Bayesian neural networks

Neural networks can be successfully applied to tasks such as regression or classification when viewed as probabilistic models. In this section, we briefly introduce the BNNs,[7,8] which provides a probabilistic interpretation of DL models and a principled method for modeling uncertainty. The idea behind Bayesian modeling is to extend the standard Neural Networks by placing a prior probability distribution (often a Gaussian) over the weight parameters when making predictions.[2] However, due to a large number of parameters for neural networks, such models are computationally intractable. Gal et al showed that neural networks with Dropout is equivalent to approximating VI in the deep Gaussian process, marginalized over its covariance function parameters.[3] This approach addresses the issues with overconfidence and providing quantification of predictive uncertainty. This is because the uncertainty in weight space captured by the posterior is incorporated into the predictive uncertainty, giving us a way DL model to say "I Don't Know."

Given dataset $D(X, Y)$, where $X = \{x_1, x_2 \ldots x_N\}$ and the corresponding labels $Y = \{y_1, y_2 \ldots y_N\}$ where $X \in R^d$ be a d-dimensional input vector and $Y \in \{1 \ldots C\}$ with $y_i \in \{1 \ldots C\}$, given C class label, a set of independent and identically distributed (i.i.d.) training samples size $N\{x_i, y_i\}$ for $i = 1$ to N, the task is to find a function $f : X \to Y$ using weights of neural net parameters w as close as possible to the original function that has generated the outputs Y.

For the regression problem of predicting a continuous output $\hat{y}$ given an input $\hat{x}$ and training dataset $D(X, Y)$, a neural network can be used to model a probability distribution over $\hat{y}$, for example, by placing a normal distribution over $\hat{y}$ and using the network to predict its mean and variance. Similarly for classification, a neural network can be used to predict a categorical distribution over the possible classes. Learning the network parameters w using the maximum likelihood estimation (MLE) criterion:

$$w_{MLE} = \text{argmax}_w p(D|w) = \text{argmax}_w \log p(D|w), \tag{7}$$

can lead to severe overfitting.

By Bayes' theorem, multiplying the likelihood with a prior distribution $p(w)$ is proportional to the posterior distribution $p(w|D) \propto p(D|w)p(w)$. Maximizing $p(D|w)p(w)$ gives the MAP estimate of w:

$$w_{MAP} = \text{argmax}_w \log p(w|D) = \text{argmax}_w \left[\underbrace{\log p(D|w)}_{\text{Likelihood}} + \underbrace{\log p(w)}_{\text{Prior}} \right]. \tag{8}$$

The optimization objectives in MAP are the same as for MLE plus a regularization term from the log prior. However, both MLE and MAP provide point estimates of parameters.

In BNNs, we treat the weights in a neural network as random variables instead of fixed parameters, and performing posterior inference on these weights. Assuming, we place a prior distribution $p(w)$ over the weights and bias of the network, the marginal likelihood $p(X, Y)$, and a likelihood function $p(y|x, w)$, this results posterior distribution:

$$p(w|D) = \frac{p(D|W)p(W)}{p(D)} = \frac{\prod_{i=1}^{N} p(y_i|x_i, w)p(w)}{p(X, Y)}, \tag{9}$$

given the data $p(w|X, Y)$. The predictive distribution of an unknown label $\hat{y}$ for a new input $\hat{x}$ by marginalizing the parameters:

$$p(\hat{y}|\hat{x}, X, Y) = \int p(\hat{y}|\hat{x}, w)p(w|X, Y)dw. \tag{10}$$

This is equivalent to averaging predictions from an ensemble of neural networks weighted by the posterior distribution $p(w|X, Y)$ and all the model parameters w. Unfortunately, an analytical solution for the posterior distribution $p(w|X, Y)$ in neural networks is intractable.

VI[11,26] converts the integration the task of computing a posterior into an optimization problem. Our objective is to use VI[11,26] to approximate the posterior distribution on the weights by a tractable variational distribution $q_\theta(w)$ indexed by a variational parameter θ.

3.3 | Variational inference

The underlying idea in VI is to approximate the (intractable) posterior distribution with (tractable) variational distribution on the weights $q(w|\theta)$ into an optimization (minimization or maximization) problem, parametrized on θ that minimizes the Kullback-Leibler (KL) divergence between variational posterior q and the true posterior: $KL(q(w|\theta)||p(w|Y, X))$. Minimizing the KL divergence is equivalent to maximizing the evidence lower bound (ELBO) which also contains the integral with respect to the distribution over latent variables. Now we take the approach of maximizing a lower bound to the model evidence $logp(X, Y)$ by applying Jensen's inequality to the KL divergence between the approximating distribution and the true posterior, to obtain the log-ELBO:

$$\text{ELBO} \equiv \log p(Y|X) \geq \log p(Y|X) - \text{KL}(q(w|\theta)||p(w|Y, X)). \tag{11}$$

The KL divergence between variational distribution $q(w|\theta)$ and the posterior $p(w|Y, X)$ is defined as:

$$\text{KL}(q(w|\theta)||p(w|Y, X)) = \int q(w|\theta) \log \frac{q(w|\theta)}{p(w|Y, X)} \geq 0. \tag{12}$$

The ELBO of the RHS of the inequality in Equation (10) can be rearranged:

$$\log p(Y|X)) \geq \text{ELBO}\varsigma = \int q(w|\theta) \log \frac{p(Y, w|X)}{q(w|\theta)}. \tag{13}$$

The loss function used to train the BNN corresponds to the negative ELBO.[27] It is simple to evaluate as opposed to the exact one. We use the variational distribution $q(\omega)$ instead of $p(w|X,Y)$. This variational distribution is chosen close to $p(.|X,Y)$, as it minimizes the Kullback Leibler divergence between the approximate posterior an the prior over w:

$$L_{VI} := \underbrace{q(w|\theta)\log p(Y|X,w)dw}_{\text{Log Likelihood}} - \underbrace{\int q(w|\theta)\log\frac{q(w|\theta)}{p(w)}dw}_{\text{KL divergence}} = E_{q(w|\theta)}[\log p(Y|X,w)] - D_{KL}(q(w|\theta)\|p(w)). \tag{14}$$

Hence, the ELBO can be decomposed into two terms: Log-likelihood and KL divergence.

1. Maximizes the likelihood of the training data, which measures how well we fitted the data close to the prior—preventing the model over-fitting. We can approximate first term by Monte Carlo integration with the modeling assumption that $p(w) = p(w|X)$ to get an unbiased estimate.
2. The second term is KL divergence of the prior w.r.t., the approximated posterior which prevents the variational posterior from becoming very different from the prior. The model prior w.r.t. for the approximated true distribution $p(\omega)$ by $q(w|\theta)$ cannot be computed exactly for a nonlinear neural network, that is: still L_{VI} is intractable.

Instead of the posterior distribution, we only need a likelihood to compute the ELBO which produces both a good fit (likelihood term), but is also regularized according to how different variational distribution is from true distribution (KL term) by maximizing ELBO with respect to θ.

3.4 | Deep neural network with DropWeights as Bayesian neural network approximation

We have to define a variational distribution on weight parameters and to develop the objective of maximization on the log ELBO. In neural networks, like Dropout,[10] we can consider approximating distribution is DropWeights. This means weights are drawn from the BNN with DropWeights, $W_i = M_i \odot Z_i = M_i \odot diag([z_{i,j}]_{j=1}^{K_i})$, where $w = \{W_i\}_{i=1}^{L}$ and M_i is the matrix of variational parameters, that is: weight matrix multiplied by a diagonal matrix formed by binary random vector Z_i, whose elements are distributed as: $M_{ij}^{(1)} \sim \text{Bernoulli}(\rho_{ij}^{(l)})$ for $i = 1, \ldots, L$ and $j = 1, \ldots, K_{i-1}$.

In VI, performing DropWeights can be interpreted as sampling weights from the variational distribution $q_\theta(w)$, where θ is the optimized variational parameter, interpreted as trained weights of the neural network. This is equivalent the mixture distribution:

$$q(w|\theta) = (1 - p_{drop})N(w;\theta,\sigma^2 I) + p_{drop}N(w;0,\sigma^2 I), \tag{15}$$

where ρ_{drop} is the probability of individual weight being set to 0, variational parameters are denoted by θ and let $\sigma \to 0$. Assuming a prior on w of the form $N(0,\sigma_w^2)$. The approximate posterior takes the form of a mixture of deltas.

We can now reparametrize[11] the integral of the first term in Equation (13) as a sum over all samples so that it only depends on the Bernoulli distribution instead of weights w directly. We estimate the expected value of the variational predictive distribution with Monte Carlo sampling over T sets of weights $\hat{w}_t$ from the variational (DropWeights) distribution as:

$$E_{q(w|\theta)}[\log p(Y|X,w)] = \sum_{i=1}^{N} \int q(w|\theta) \log p(Y|X,w) dw \approx \frac{1}{T} \sum_{t=1}^{T} \log p(y_t|x_t, \hat{w}_t). \tag{16}$$

Note that $\hat{w}_t$ is a random variable from the Bernoulli distribution, which is identical to applying DropWeights to the network.

Next, we have to approximate the second term of the ELBO in Equation (13), the KL divergence between the variational distribution and the prior over w, where $p(w)$ is a multivariate normal distribution. Thus, following Gal at al,[3] we can derive this approximation and the objective function can be expressed as:

$$\text{ELBO} \approx L_{drop}(m) = \log p(y|x,w) - \frac{\lambda}{2} \|m\|_2^2; \tag{17}$$

Note that, maximizing this ELBO is identical to the loss function used in a standard neural network with L2 weight regularization. Therefore, training a neural network with DropWeights has the same effect as minimizing the KL term in Equation (13).

In summary, the core idea of a BNN is neural networks with DropWeights VI and Gaussian prior weights is Bayesian. By re-parametrizing the approximate variational distribution $Q(w|v)$ on the Bernoulli distribution instead of weights w. Thus the loss is:

$$L_{DropWeights} = \frac{1}{D} \sum_{b=1}^{batch} (Loss_b) + \lambda \sum_{l=1}^{layer} (Weight_l)^2. \tag{18}$$

The DropWeights layers are kept active during inference, to keep Bernoulli distribution over weights. We inferred using Equation (10) that, after multiple forward passes to approximate the posterior distribution of class probabilities from the trained network with DropWeights, the predictive distribution of an unknown label $\hat{y}$ for a new input $\hat{x}$ by marginalizing the parameters is:

$$p(\hat{y}|\hat{x}, X, Y) \approx P(\hat{y}|\hat{x}, w) P(w|X, Y) dw \approx \frac{1}{T} \sum_{t=1}^{T} p(\hat{y}|\hat{x}, \hat{w}_t). \tag{19}$$

Intuitively, the mean of the predictive posterior corresponds to the point estimates, and the width of the predictive posterior reflects the reliability of the predictions. We call this approach MC-DropWeights which is a generalization over the previous work referred to as MC-Dropout.[3] A summary of these steps is provided in Algorithm 1.

3.5 | Uncertainty decomposition: Estimating uncertainty in DropWeights deep neural nets

There are two major sources of uncertainty in DL model:[5,18]

1. AU or data uncertainty accounts for inherent stochasticity in the data, due to class overlap, label noise, homoscedastic and heteroscedastic noise, which leads predictions with high uncertainty. AU cannot be reduced even if more data were to be collected, unless it is possible to observe all explanatory variables with greater precision. We can define AU as: information required—information available.
2. EU, also known as, model uncertainty, is a consequence of insufficient learning of model parameters, due to a finite set of training data, which leads to broad posteriors. It is impossible to determine a model's parameters exactly with limited observations. This uncertainty measurement captures "what the model does not know." EU associated with the model reduces as the training data size increases. We can compute EU as: information available—information expressed.

Kendall and Gal[18] derived a unified Bayesian DL framework for both classification and regression on pixel-based semantic segmentation, by decomposing uncertainty into AU—modeled by placing a distribution over the output of the model—and EU. It does this by placing a prior distribution over the model's parameters. The last layer in the network has extra nodes before activation, consisting of mean and variance of logits. Disentangling these two sources of uncertainty can be useful for risk sensitive learning, rejecting OOD samples, and balancing exploration and exploitation in a reinforcement learning settings. The predictive uncertainty (ie: variational predictive distribution) for classification with either softmax (multiclass) or sigmoid (binary) likelihood $p(\hat{y}|\hat{x}, w) = \text{activation function}(f^w(\hat{x}))$ in the model can be approximated by:[4]

$$\begin{aligned}
&\text{Var}_{q_{\hat{\theta}}(\hat{y}|\hat{x})}(\hat{y}) = E_{q_{\hat{\theta}}(\hat{y}|\hat{x})}\{\hat{y}^{\otimes 2}\} - (E_{q_{\hat{\theta}}(\hat{y}|\hat{x})}\hat{y})^{\otimes 2} \\
&= \underbrace{\int_{\Omega}[\text{diag}\{E_{p(\hat{y}|\hat{x},\omega)}(\hat{y})\} - E_{p(\hat{y}|\hat{x},\omega)}(\hat{y})^{\otimes 2}]q_{\hat{\theta}}(\omega)d\omega}_{\text{aleatoric}} + \underbrace{\int_{\Omega}\{E_{p(\hat{y}|\hat{x},\omega)}(\hat{y}) - E_{q_{\hat{\theta}}(\hat{y}|\hat{x})}(\hat{y})\}^{\otimes 2}q_{\hat{\theta}}(\omega)d\omega}_{\text{epistemic}} \\
&\text{Aleatoric uncertainty estimator: } \frac{1}{T}\sum_{t=1}^{T}\text{diag}(\hat{\sigma}_t^2) \text{ and} \\
&\text{Epistemic uncertainty estimator: } \frac{1}{T}\sum_{t=1}^{T}(\hat{\mu}_t - \overline{\mu})^{\otimes 2} \\
&\text{where } \overline{\mu} = \sum_{i=1}^{T}\frac{\hat{\mu}_t}{T}.
\end{aligned} \tag{20}$$

During training, the variance estimate is sampled and added to the probability logits, which are used to calculate the training loss in the network.

There are two important points to note here. First, for confidence in prediction, it is important to marginalize over the learned posterior, exact or otherwise, so that appropriate uncertainty about parameters is propagated into the prediction. This marginalization is mostly ignored in DL, where point estimates of the parameters are used for prediction. Second, the computation of the posterior predictive distribution is used for aggregating parameter uncertainty. This is particularly challenging for classification, where a model's confidence in its prediction is not readily available.

In Bayesian classification, the predictive probability of $\hat{y}$ for given $\hat{x}$ belongs to $y_i \in \{1 \ldots C\}$ class level and each feature is weighed differently to determine output of the neural network $p(\omega) = \omega\{f^{\omega}(\hat{x})\}$ where ω is random, propagated through the network function with all

parameters weights of all layers $\{1 \ldots L\}$ and ω activation function. The predictive uncertainty is $E_{q_{\hat{\theta}}(\hat{y}|\hat{x})}\{\hat{y}^{\otimes 2}\} - E_{q_{\hat{\theta}}(\hat{y}|\hat{x})}(\hat{y})^{\otimes 2}$. The expectation $E[\text{diag}\{p(\omega)\} - p(\omega)^{\otimes 2}]$ over $q_{\hat{\theta}}(\omega)$ denotes the AU, which captures inherent randomness of an output $\hat{y}$. The EU, $E[p(\omega) - E\{p(\omega)\}]^{\otimes 2}$, originates from the variability of ω input dataset.

During training, the variance estimate is sampled and added to the probability logits, which are used to calculate the training loss in the network.[4] In the above approach for estimating uncertainty, there are mainly two limitations:

1. Equation (1) captures the variance of class probabilities associated with the predicted class c from sample T sets. Which essentially quantifies the uncertainty of the variability in the specification of the probability distribution of the linear predictor function instead of the predictive probabilities in the outcome class of the model.[4]
2. The network produces two outputs, prediction probability logits and a variance estimate. The above method requires extra parameters at the last hidden layer and often causes unstable parameter updates in a training phase.

To address the above limitations, we introduce a predictive uncertainty estimator, which averages the standard deviations of the class probabilities associated with every class based on DropWeights regularization, reinterpreted as VI.

The estimate of the vector of Softmax probabilities can be denoted: $\hat{\mu}_c = \frac{1}{T}\sum_{t=1}^{T} p(\hat{y} = c|\hat{x}, \hat{\omega}_t); \quad c \in \{1, \ldots, C\}$

$$\text{Aleatoric uncertainity: } \frac{1}{T}\sum_{t=1}^{T}\text{diag}(\hat{y}_t) - \hat{y}_t^{\otimes 2} \tag{21}$$

$$\text{Epistemic uncertainity: } \frac{1}{C}\sum_{i=1}^{C}\sqrt{\frac{1}{T}\sum_{t=1}^{T}[p(\hat{y}_t = c|\hat{x}, \hat{\omega}_t) - \hat{\mu}_c]^2} \tag{22}$$

$$\text{where } \hat{y}_t = y(\hat{\omega}_t) = \text{Softmax}\{f^{\hat{\omega}_t}(\hat{x})\}. \tag{23}$$

In practice, the predictive probability is estimated as follows:

I Repeat the stochastic forward pass T times through the neural networks with Dropweights.
II For each stochastic forward pass, a different network is making predictions because DropWeights randomly switched off units.
III As a result, each stochastic forward pass returns different vectors of class predictions, which is equivalent to stochastic VI drawing new independent prediction (see eq. (6.3), p. 109 and Prop. 4, p. 149 in Reference 3).
IV Finally, average the predictions to get the final prediction as an uncertainty estimator associated with the sample in prediction exercise.

The above method reduces the required hyperparameters and improves computation. It considers the model uncertainty associated with every class prediction.

3.6 | Bayesian deep ensembles of DropWeights

It has long been observed that ensembles perform model combination to obtain a more powerful model. This improves predictive performance when the true model does not lie within the hypothesis class.[28]

The current state-of-the-art BNNs learn a distribution overweight for estimating predictive uncertainty. However, they suffer from the "mode in collapse" problem in deep CNNs when dealing with complex high-dimensional image data such as medical images (X-Rays, PET/CT, SPECT, MRI, Ultrasound, EEG, ECG etc). To estimate uncertainty in DL, the quality of Bayesian posterior distribution depends on prior specification and posterior approximation. This translates weight uncertainty into predictive uncertainty. Therefore, DL models can be easily fooled by adversarial examples such as small perturbations in the input images, which results in overconfident predictions in VI.

We proposed a novel technique "Bayesian deep ensembles of DropWeights" for estimating uncertainty in DL that yields high-quality predictive uncertainty estimates and outperforms existing methods (eg, MC-Dropout and our MC-DropWeights). We present the first approach (to the best of our knowledge) a stochastic ensemble of MC-DropWeights models characterized by a different set of drop weights probabilities, for estimating uncertainty in Bayesian DL.

Considering the ensemble as a mixture model, where each model is the connection information, that is, the binary mask drawn from a Bernoulli distribution and the model's DropWeights rate between 0 and 1. We can estimate the model uncertainty by training several models and calculating the variance of their output prediction by approximately marginalizing over model parameters using MC-DropWeights sampling as:

$$q(y|x) = E_{q(w)}[logp(y|x,w)] \approx \frac{1}{M}\frac{1}{N}\sum_{m=0.0}^{M}\sum_{i=1}^{N} logp(y|x,w^{(i)}); \tag{24}$$

This can be seen as drawing from an infinite ensemble of networks with N number of forward passes, M number of network models with DropWeights rate between 0 and 1 to estimate uncertainty.

3.7 | Estimating uncertainty in classification

In segmentation, we introduced a predictive uncertainty estimator, which averages the standard deviations of the class probabilities associated with every class. However, we need an alternative approach, since the predictive probabilities from a forward pass through the model does not capture uncertainty in classification.

Entropy is the basic principle of information theory proposed by Shannon.[29,30] It depends on sample size and typically exhibits substantial bias. The model output in classification is a conditional probability distribution $P(y|x)$ over a discrete set of outcomes Y. We can use the entropy of the predictive distribution as an uncertainty measure. Recently, Smith and Depeweg[5,31] used predictive entropy to decompose uncertainty into its epistemic and aleatoric components.

We have analyzed two approaches to estimate uncertainty within classification: tractable view of the mutual information (MI)[30,32] and bias-corrected MI.[33] The MI between the prediction y and the posterior over the model parameters w captures a different measure of uncertainty.

MI is well-known in information theory and quantifies the divergence between the joint probability density function (PDF) $p(x,y)$ and the product $p(x) \odot p(y)$ of the independent PDFs, which

captures the information overlap between quantities. The approximate predictive posterior's entropy is given by:

$$H(y_1, \ldots y_n|x_1, \ldots x_n) = H(q_{y_1, \ldots y_n|x_1, \ldots x_n}) = H_a(y_1, \ldots y_n|x_1, \ldots x_n) + H_e(y_1, \ldots y_n|x_1, \ldots x_n), \tag{25}$$

where H(.) is the differential entropy of a probability distribution and the expected value (E) $q_{y|x} = E_{q(w)}[p(y_1, \ldots y_n|x_1, \ldots x_n, w)]$. This value $H(y_1, \ldots y_n|x_1, \ldots x_n)$ represents the total uncertainty in the model prediction.

The average uncertainty in Bayesian deep neural network predictions can be computed as:

$$H_a(y_1, \ldots y_n|x_1, \ldots x_n) = E_{q(w|\theta)}[H(y_1, \ldots y_n|x_1, \ldots x_n, w)]. \tag{26}$$

Thus, the above equation is a measurement of the model's AU.

Information gain (I) about the model parameters, that is: information shared by multiple variables in DL can be expressed by the relationship:

$$\sum_{i=1}^{N} I(y_1, \ldots y_n; \omega|x_1, \ldots x_n, D) = H(y_1, \ldots y_n|x_1, \ldots x_n) - E_{q(w|\theta)}[H(y_1, \ldots y_n|x_1, \ldots x_n, w)]. \tag{27}$$

The above equation (25) maximizes the MI between predictions and model posterior. Intuitively, it quantifies the "amount of information" obtained about model parameters by observing the model predictions for a given sample. Thus, the MI between model label y and model parameters w is a measurement of the model's EU. The first term is the entropy of the model prediction, which is high when the model's prediction is uncertain. The second term is an expectation of the entropy of the model prediction over the approximate posterior around the model parameters. This is low when the model captures the expected uncertainty in the predictions for each weight configurations drawn from approximate posteriors.[34]

The above equation double counts MI between data points and overestimates the true MI.[35] Estimation of entropy from the finite set of data suffers from a severe downward bias when the data is under-sampled. Even small biases can result in significant inaccuracies when estimating epistemic entropy.[36]

3.7.1 | Estimating bias-corrected epistemic uncertainty

Consider class probabilities $p(y_{x_i} = c|x_i, \omega_t, D)$ with $\omega_t \sim q(\omega|D)$ with $W = (\omega_t)_{t=1}^{T}$, a set of i.i.d. samples, drawn from $q(\omega|, D)$. The below procedure computes the Monte Carlo estimate of the posterior predictive distribution, its entropy and MI:

$$\sum_{i=1}^{N} I_{MC}(y_i; \omega|x_i, D) = H(\hat{p}(y_i|x_i, D)) - \frac{1}{|W|} \sum_{\omega \in W} H(p(y_i \mid x_i, \omega, D)) \ , \tag{28}$$

where

$$\hat{p}(y_i|x_i, D) = \frac{1}{|W|} \sum_{\omega \in W} p(y_i|x_i, \omega, D) \ . \tag{29}$$

The first term in the MC estimate of the MI is called as the plug-in estimator of the entropy:

$$\hat{H} = H(\hat{p}) = -\sum_c \hat{p}_c \log \hat{p}_c, \tag{30}$$

where $\hat{p}_c = \frac{1}{T}\sum_t p_{tc}$ are the maximum likelihood estimates of each probability $\hat{p}_c$. It has long been known that the plug-in estimator underestimates the true entropy and plug-in estimate is biased.[37,38]

A classic method for bias correction is the Jackknife resampling method.[39] In order to alleviate the bias problem, we propose Jackknife estimator to estimate EU to improve entropy-based estimation model. Unlike MC-Dropout, it does not assume constant variance. If $D(X, Y)$ is the observed random sample, the ith Jackknife sample, x_i is the subset of the sample that is a "leaves-one-out" observation $x_i : x_{(i)} = (x_1, \ldots x_{i-1}, x_{i+1} \ldots x_n)$. For sample size N, the Jackknife standard error $\hat{\sigma}$ is defined as: $\sqrt{\frac{(N-1)}{N}\sum_{i=1}^{N}(\hat{\sigma}_i - \hat{\sigma}_{(\odot)})^2}$, where $\hat{\sigma}_{(\odot)}$ is the empirical average of the Jackknife replicates: $\frac{1}{N}\sum_{i=1}^{N}\hat{\sigma}_{(i)}$. Here, the Jackknife estimator is an unbiased estimator of the variance of the sample mean.

The Jackknife correction of a plug-in estimator $H(\cdot)$ is computed as:[39]

1. Given a sample $(p_i)_{i=1}^N$ with p_i discrete distribution on multiclass classification $1 \ldots C$
2. for each $i = 1 \ldots N$
   compute the leave-one-out estimator: $\hat{p}_c^{-i} = \frac{1}{N-1}\sum_{j \neq i} p_{jk}$
   compute the Jackknife estimator of entropy: $\hat{H}_{-i} = H(\hat{p}^{-i})$
3. then compute the bias-corrected entropy estimator $\hat{H}_{jk} = N\hat{H} + \frac{(N-1)}{N}\sum_{t=1}^{N}\hat{H}_{(-i)}$, where $\hat{H}_{(-i)}$ is the observed entropy based on a sub-sample in which the ith individual is removed.

We leveraged the following relation:

$$\mu_{-i} = \frac{1}{N-1}\sum_{j \neq i} x_j = \frac{N}{N-1}\mu - \frac{1}{N-1}x_i = \mu + \frac{\mu - x_i}{N-1},$$

while resolving the ith data point out of the sample mean $\mu = \frac{1}{N}\sum_i x_i$ and recompute the mean μ_{-i}. This makes it possible to quickly compute leave-one-out estimators of discrete probability distribution.

The above method was simple to implement and computationally cheaper than the other re-sampling methods such as Bootstrap. It derives an estimate of the finite sample bias from the leave-one-out estimators of the entropy and reduces bias considerably down to $O(n^{-2})$.[39]

The bias-corrected EU (BCEU) estimation model explains regions of ambiguous data space that are hard to classify as data distribution due noise in the inputs or the fact that the model was trained with different domain data. Consequently, these inputs should be assigned a higher AU. As a result, we can expect a high model uncertainty in these regions. A summary of these steps is provided in Algorithm 2.

4 | ESTIMATING UNCERTAINTY IN MEDICAL IMAGE SEGMENTATION

In this section, we demonstrate that the uncertainty estimates obtained from DropWeights using the Bayesian residual U-Net (BRUNet) provide an additional insight for clinicians on the tasks of semantic segmentation with help from deep learners.

4.1 | Bayesian residual U-Net

In semantic segmentation, to get a better result, it is crucial to use low-level details while retaining high-level semantic information.[23,24] We used deep BRUNet architecture that takes advantage of strengths from both deep residual learning[40] and U-Net architecture.[41] Both the U-Net and residual network have a simple structure and faster training speed, but U-Net's accuracy of the experimental results, due to its lack of depth, is insufficient and the residual network effectively addresses the problem of degeneration of deep CNN. Therefore, we have combined the strengths of two networks effectively in our artificial neural network to implement with Monte Carlo Dropout layers, as shown in the Appendix below, to estimate model uncertainty.

We have designed BRUNet using convolution layer, an activation layer, pooling layer, and fully connected layer with a combination of max-pooling and batch normalization. Dropout is applied to the network as an approximation to the Gaussian process (GP) and to cast as approximate Bayesian inference, as shown in Figure 2.

DL models require to be initialized with the right weights to avoid vanishing/exploding gradients problem. "He" initialization[25] draws samples from a truncated normal distribution centered on 0 with stddev = sqrt(2 / fan-in) where fan-in is the number of input units in the weights, to asymptotically preserve variance of activations in the forward pass and variance of gradients in the backward pass. The exponential linear unit (ELU) is a recently introduced activation function in DL. It computes the function $f(x) = x$ if x $\geq$ 0 (identity function) and $f(x) = \alpha \cdot (e^x - 1), \alpha$ is a positive constant number, if x <0. ELU tends to converge mean activations closer to zero also causes faster learning, convergence and produce more accurate results. The last layer in the fully connected network holds the scores for each class from sigmoid the unction of the patch.

The entire network is still in the form of a U-shaped structure, which involves down-sampling first, followed by upsampling, the down-sampled features are merged with the corresponding up-sampled features. Finally, the result is obtained through the fully connected layer. The intuition behind this is that extracting low-level features to the correspondingly high levels creates a path for information propagation between low and high levels in a much easier way. This not only facilitates backward propagation during training but also compensates low level finer details to high-level semantic features in dense prediction task. The contraction and expansion layers are convolutions and deconvolution layers. Hence the image is recreated with segmented masks, like the image input size.

The performance of our model is evaluated on the mean average precision at different intersection over union (IoU) thresholds and dice similarity coefficient (DSC).[42]

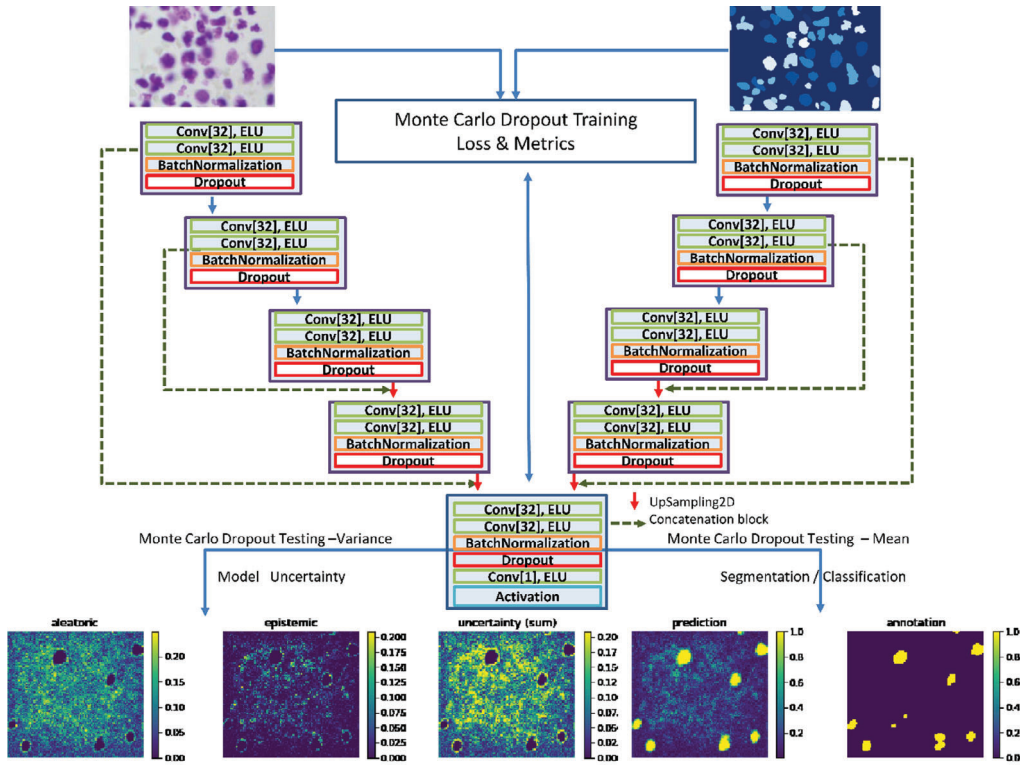

FIGURE 2 Bayesian residual U-Net (BRUNet) architecture [Color figure can be viewed at wileyonlinelibrary.com]

1. IoU, also known as the Jaccard index, is an evaluation metric for pixel-level image segmentation. The IoU is the percent overlap between the area of ground truth and the predicted area. The higher IOU to a certain threshold, the more accurate is the prediction.
2. The DSC is the most widely used measure of the reproducibility as a validation of manual annotation where clinicians repeatedly annotated the same image and the pair-wise spatial overlap accuracy of automated probabilistic segmentation of images. It ranges between 0 and 1.
3. Model accuracy is used to judge the performance of the model and is similar to a loss function. The loss function is set to 1—dice coefficient loss between the predicted and true labels as follows:

$$\mathcal{L} = 1 - \frac{2\sum y_{true}\hat{y}_{pred}}{\sum y_{true} + \sum \hat{y}_{pred}}. \tag{31}$$

We evaluated the validation accuracy after every epoch and saved the model with the best prediction accuracy (lowest loss) on the validation set.[43,44]

4.2 | Dataset

We have used the dataset provided in the Kaggle Data Science Bowl Challenge 2018[45] to demonstrate the merit of our proposed method. It consists of microscopy images of a large number of

segmented nuclei images.[46] The images were acquired under a variety of conditions and vary in the cell type, magnification, and imaging modality (brightfield vs fluorescence). In CNN architecture, it is necessary to convert all images to the same size. All images were cropped to a square-center region and resized to 128 × 128 pixels, so that they were standardized and uniform. This ensured the aspect ratio avoided distortion, to speed up the process. There are 670 train samples and around 4000 test samples.

4.3 | BRUNet parameters details

All models are trained and evaluated using Keras with Tensorflow backend. We used the following hyper-parameters. All Nuclei images were resized to the same dimension 128 × 128 × 3. The BRUNet was trained by stochastic gradient descent (SGD), with weights initialized using the "He" activation. The SGD optimizer with the Dropout rate of 0.10, 0.20, 0.25, 0.50, 0.75, and 0.95 and early stopping rule with 25 epoch patience was applied, with a batch size of 16. The total parameters of the network were 4,452,097. The binary-cross entropy function was used as a loss function to calculate the validation loss of various models for comparison. The VI with Dropout variational distribution was used. The number of realized sets T used in Monte Carlo integration was 10.

4.4 | Experimental results

In the previous sections, we have discussed modeling different aspects of predictive uncertainty and presented measures of quantifying it. This section evaluates our method when applied to the problem of discovering nuclei in divergent microscopy data images. This application is receiving much attention from the DL community[45,47–49].

4.4.1 | Network performance

We have observed high accuracy in the case of isolated Cells when compared with overlapping cells, as shown in Figures 3 and 4. The highest areas of AU occurred on class boundaries and EU increases for complex pixels.

Using a simple CNN, the nuclei were spotted with model accuracy of 91% for stage 1 train results with mean IoU score of 62%. Using the BRUNet model described above, we obtain a mean accuracy of 96.55% with mean IoU of 83%.

4.4.2 | Distribution of uncertainty estimates

The distribution of AU appears to be multimodal, with peaks close to 0.13, as shown in Figure 5. The incorrect classifications greatly contribute to the multimodality due to irreducible homoscedastic and heteroscedastic noise in data.

The distribution of EU appears to be normal. The incorrect predictions are centered around a higher uncertainty, whereas far more of the correctly predicted classes are concentrated around a low uncertainty value, as shown in Figure 6.

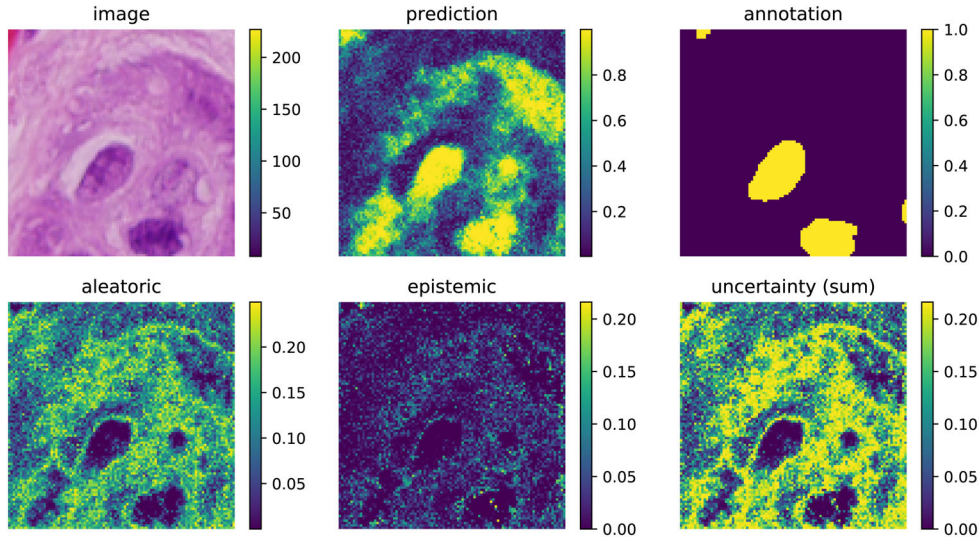

FIGURE 3 The segmentation was performed by the model on images such as those shown above with overlapping cells with uncertainty [Color figure can be viewed at wileyonlinelibrary.com]

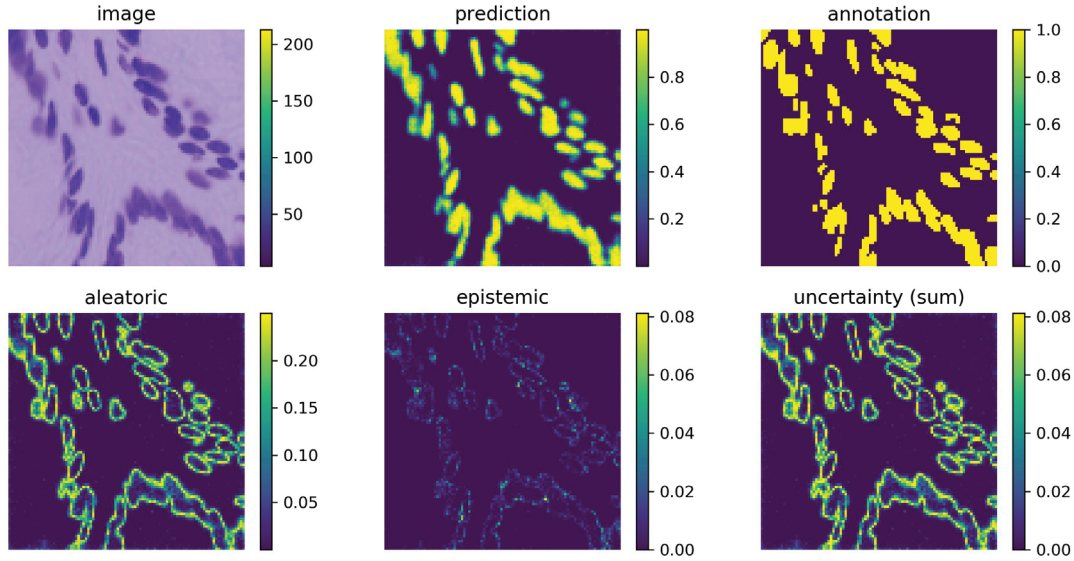

FIGURE 4 The segmentation was performed by the model on the image with isolated cells with uncertainty [Color figure can be viewed at wileyonlinelibrary.com]

4.4.3 | Correlation between aleatoric uncertainty and epistemic uncertainty with predictive probabilities

To study the correlation between model uncertainty and data uncertainty, we measured the estimated conditional expectations of the EU and AU given ranges of the predictive probabilities, respectively.

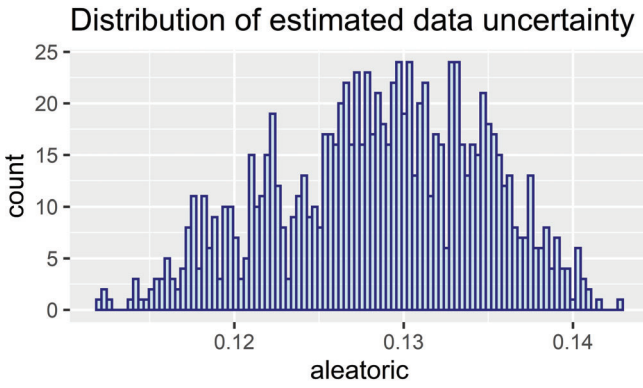

FIGURE 5 Distribution of estimated aleatoric uncertainty [Color figure can be viewed at wileyonlinelibrary.com]

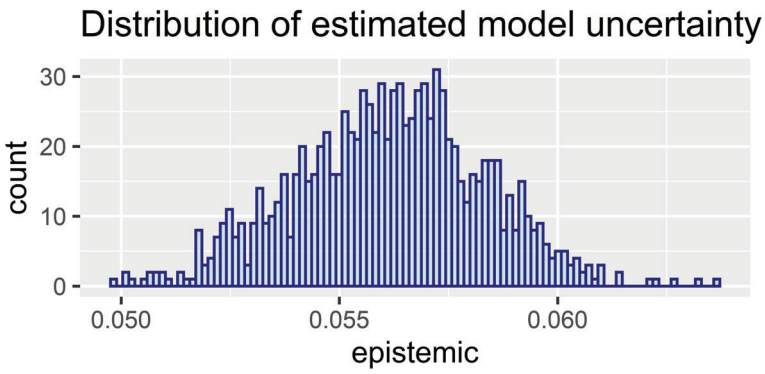

FIGURE 6 Distribution of estimated epistemic uncertainty [Color figure can be viewed at wileyonlinelibrary.com]

As expected, uncertainty decreases as the predictive probabilities increase, as shown in Figure 7. A blue point corresponds to a prediction with a low value of uncertainty. A red point corresponds to observation with a high value of uncertainty. It confirms that for the higher uncertainty predominately due to incorrect classifications, most of the points are concentrated around the area of maximum entropy.

This correlation between AU and EU with predictive probabilities indicates that the approximated uncertainty estimates indeed contain valuable information in incorrect cases.

4.4.4 | The effect of varying stochastic feed forwards

In practice, MC DropWeights is equivalent to performing T stochastic forward passes through the BRUNet and averaging the results. We have observed from Figure 8 that the AU decreases with the increase in the number of stochastic forward pass (T), whereas the rate of change of range for the EU not much significant with increases in the number of stochastic feed forwards.

4.4.5 | The contribution of uncertainty in predictive probabilities

The uncertainty adds complementary information to the conventional network output—for the correctly classified cases the model uncertainty is low, however, for the incorrectly classified

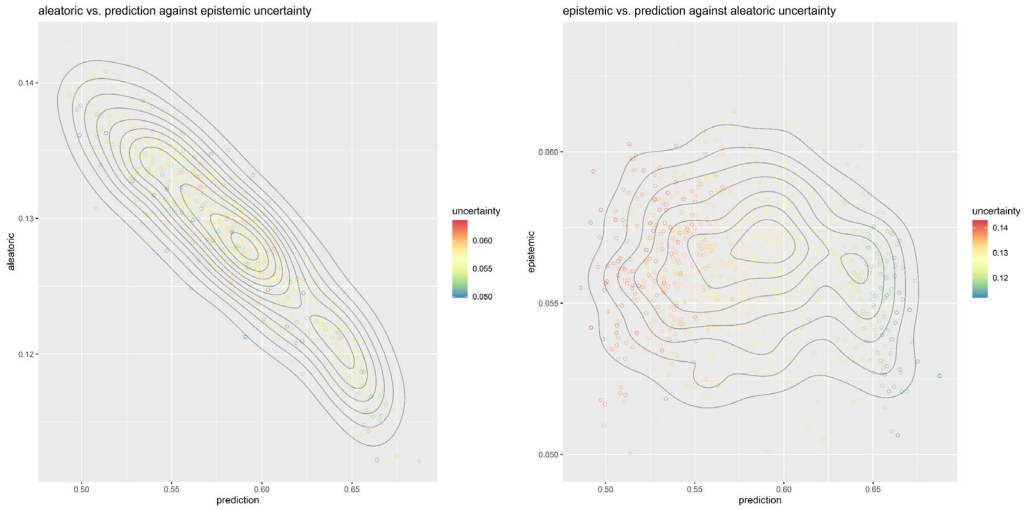

FIGURE 7 The joint distribution of between aleatoric uncertainty epistemic uncertainty vs prediction (2D kernel density estimate) [Color figure can be viewed at wileyonlinelibrary.com]

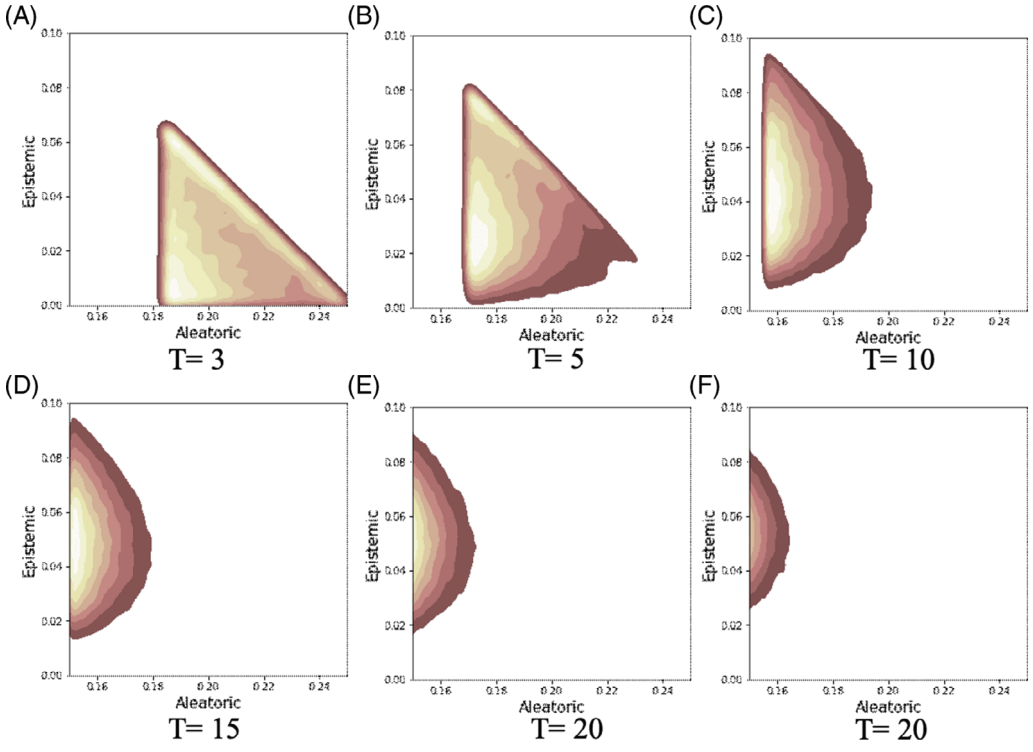

FIGURE 8 Bivariate density plot for aleatoric and epistemic uncertainties against fixed dropout with varied stochastic feed forward of the model. A, $T = 3$. B, $T = 5$. C, $T = 10$. D, $T = 15$. E, $T = 20$. F, $T = 20$ [Color figure can be viewed at wileyonlinelibrary.com]

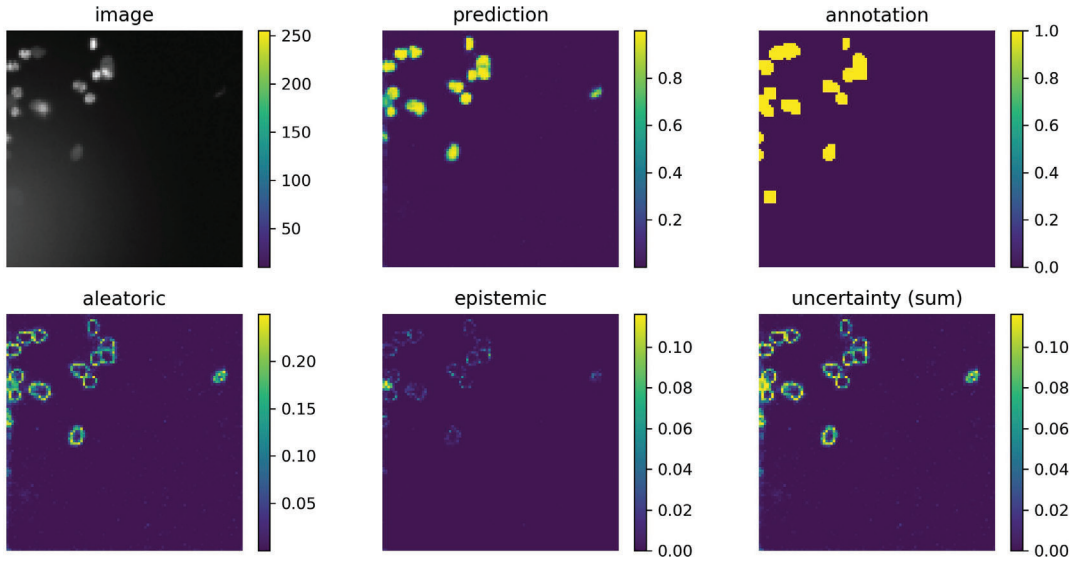

FIGURE 9 Segmentation predictions and uncertainty maps [Color figure can be viewed at wileyonlinelibrary.com]

images, the standard deviation of the predictive probabilities most likely class seems to be higher. When the prediction disagrees with the ground truth, the uncertainty identified the region missed by prediction as highlighted in Figure 9.

The model is generally highly certain or provides higher confidence in its prediction in the cases, where it predicted the correct class. Uncertainty information means the model can be easily interpreted, compared to the case with no uncertainty information in Microscopy cell images of nuclei segmentation.

5 | ESTIMATING UNCERTAINTY IN MULTICLASS DISEASE DETECTION

In this section, we demonstrate that the uncertainty estimates obtained from Bayesian deep ensembles of DropWeights using the BCEU provides additional insight for clinicians on the tasks of disease detection with help from deep learners.

5.1 | Dataset: Multimodality magnetic resonance imaging brain tumor images

MRI is one of the commonly used medical imaging tools, which provides informative data for diseases such as brain tumour diagnosis. The interpretation of medical images, including diagnosing the tumours from the MRI images which are an integral part of medical diagnosis, requires an experienced radiologist, a human whose skills are scarce and who is susceptible mistakes. We can use of artificial intelligence (AI), the development of DL techniques and simple features from the images, such as intensity, contours, and shapes as means of computer-based assisting

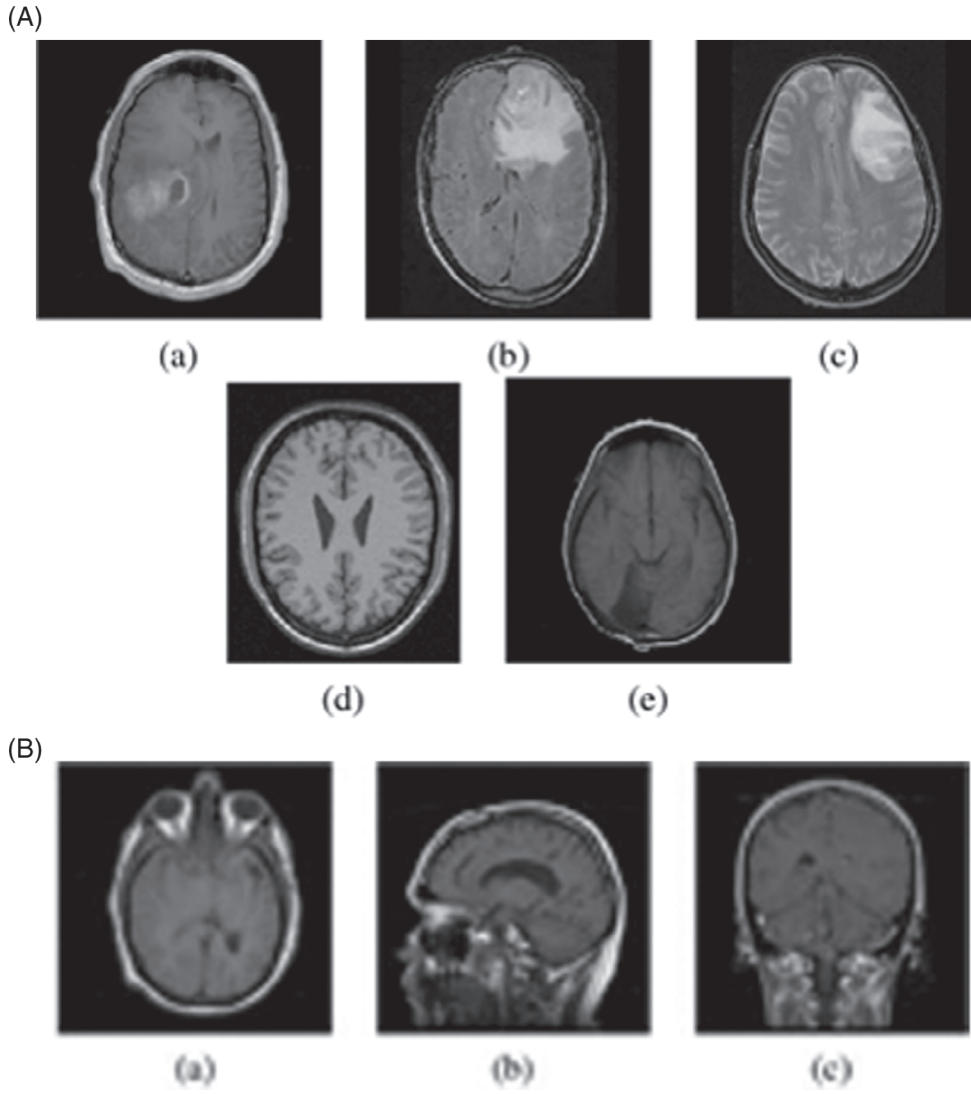

FIGURE 10 A, Types of brain tumors used. (a) Astrocytoma, (b) Glioblastoma multiforme, (c) Oligodendroglioma, (d) Healthy tissue, and (e) Unknown Tumor. B, Image planes of a brain MRI. (a) Axial plane, (b) Sagittal plane, and (c) Coronal plane

(classification and prediction) in medical diagnostic imaging. Here, DL-based solutions for detecting disease have been proposed with quantifying uncertainty in a decision, for example, image-based (aleatoric) and model (epistemic) uncertainties.

In order to validate the effectiveness of our framework, we performed experiments on brain MRI scan images of three brain tumour types (Astrocytoma, Glioblastoma, Oligodendroglioma) with additional two categories (Healthy brain MRI and Unidentified tumour), as shown in Figure 10.

The MRI images, which include Astrocytoma, Glioblastoma, Oligodendroglioma, and unidentified tumours, were obtained from the Repository of Molecular Brain Neoplasia Data (REMBRANDT) from Cancer Imaging Archive.[50] This dataset has 65 427 MRI images in DICOM

TABLE 1 The brain MRI dataset

Data source	Tumor type	No. of Images
REMBRANDT	Astrocytoma	21 307
REMBRANDT	Glioblastoma	17 983
REMBRANDT	Oligodendroglioma	12 460
REMBRANDT	Unidentified	13 677
MIRIAD	Healthy brain	30 688
BRAINS	Healthy brain	556
Total	—	96 115

format (the standard format of MRI images) categorized according to the 100 patient IDs. The images were converted into as standard image formats like JPEG and categorized according to the tumour types with the help of clinical metadata. The MRI images of healthy brain images were obtained from the Brain Images of Normal Subjects (BRAINS) Image Bank repository of University of Edinburgh[51] and from Minimal Interval Resonance Imaging in Alzheimer's Disease (MIRIAD), a dataset used in a research related to Alzheimer's disease (AD).[52] The MIRIAD dataset contains MRI images of healthy brain ad AD group. A single pickle file was created with these images along with their labels for quick access and computation.[53] The complete dataset with the number of images in each category are listed in Table 1.

This dataset contains 3064 MRI images of 233 patients, containing 708 meningiomas, 1426 gliomas, and 930 pituitary tumors diagnosed with one of the aforementioned three brain tumor types. The most important property of this dataset is that it includes both the brain images and the segmented tumors.

The classes in our dataset are not balanced. Class imbalance is one of the most common problems in real-world classification task. We have splitted the dataset to training (80%) and testing (20%) before sampling so data points will not be shared among training and test dataset. We took same number of samples from all classes to create a balanced dataset to train our models. We compared the performances between the two types of datasets, balanced, which contains 20% per class of brain tumor types and imbalanced, which have data distribution based on the abundance of classes of brain tumors in the image dataset: 22%, 19%, 13%, 32%, 14%.

5.2 | Experiments

All models were trained and evaluated using Keras with Tensorflow backend. Each image had three colour channels. The images are resized to 64 × 64 pixels for faster feature extraction.

Our objective was to define a framework for measuring uncertainty in DL models and evaluate its usefulness. It was not, however, to achieve the state-of-the-art performance in DL, so for the DNN architecture, we used a generic building block containing the following model structure: Conv-Relu-BatchNorm-MaxPool-Conv-Relu-BatchNorm-MaxPool-Dense-Relu-[Dropout or DropWeights]- Dense-Relu-[Dropout or DropWeights]-Dense-Softmax, with 32 convolution kernels, 3 × 3 kernel size, 2 × 2 pooling, dense layer with 64 units, 32 units, and drop rate probabilities ranging from 0.1 to 1.0, increasing by 0.05 to obtain models for uncertainty. One of the

most practical and more robust optimizers is Adam. Essentially Adam optimization algorithm is an extension to SGD and computes individual adaptive learning rates for different parameters. It combines the advantages of two SGD extensions—root mean square propagation (RMSProp) and adaptive gradient algorithm (AdaGrad). We trained the network to minimize the cross-entropy loss using ADAM optimizer, which had an initial learning rate of 0.0001. The batch size was set to 32 and training was performed for a maximum of 100 epochs. We evaluated the validation accuracy after every epoch and saved the model with the best prediction accuracy on the validation set.

5.3 | Results and discussion

In this section, we propose uncertainty estimation performance metrics in classification that incorporates the ground truth label, model prediction, and uncertainty threshold. We analyzed how the model uncertainty can be useful for ranking the model predictions, by referring uncertain MRI images of brain tumours. This will improve the overall model performance and improve clinical diagnosis.

We also compared the uncertainty-based classification performance obtained through our proposed method, using the state-of-the-art method, MC-Dropout, on balanced and imbalanced datasets, which showed considerable improvement in prediction accuracy and quality of uncertainty estimation.

5.3.1 | Uncertainty estimation performance metrics in classification

There is no ground truth for uncertainty threshold or tolerance for evaluation of estimated uncertainty in DL. We leveraged the estimated uncertainty to enhance classification performance metrics.[54] We first computed the accuracy map using the ground truth labels, model predictions, and confidence map, by normalizing uncertainty threshold values to develop the evaluation matrix.

Like in real-world referral situations, any medical diagnostic DL algorithm should be able to flag the least confident images that require more investigation by medical experts. Although the model does not necessarily require to be wholly certain for correctly predicting cases, for incorrect predictions, it is expected that the estimated uncertainty will be high.

The evaluation matrix itself is not an estimated uncertainty performance measure. However, based on that, we can measure uncertainty accuracy (UA) using diagnostic screening tests (sensitivity, specificity, positive (PPV) and negative (NPV) predictive values) as shown in Figure 11.

5.3.2 | Detecting infected patients with confidence

Importantly, EU as quantified by bias-corrected MI adds complementary information to the DL output. We observed with high probabilities, an image that is diseased are confined to lower EUs indicating in Figure 12A. In contrast, for healthy images, the uncertainty variation as seen on the scatter plot has a wider spread.

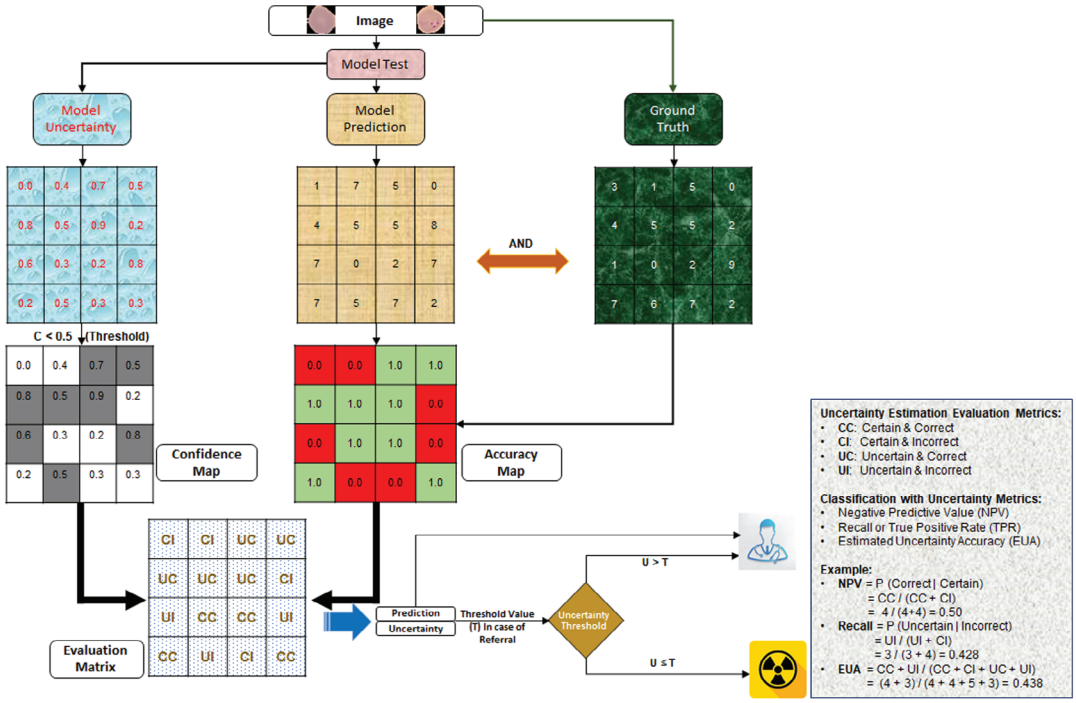

FIGURE 11 Overview to evaluate the uncertainty quality metrics in classification task in disease detection [Color figure can be viewed at wileyonlinelibrary.com]

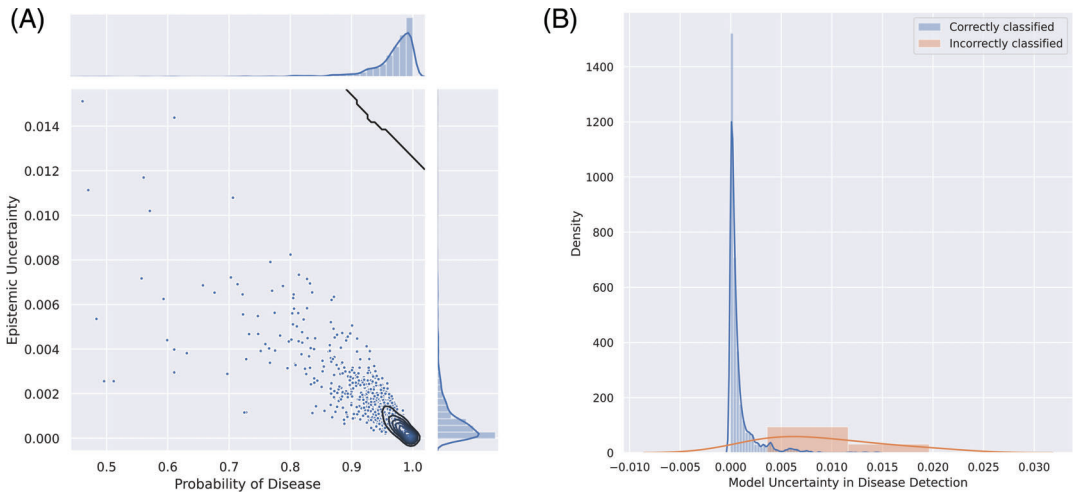

FIGURE 12 A, The scatter plot between predictions and uncertainty. It shows data with inherent noises might cause prediction errors. B, Illustrating the distributions of model uncertainty values are plotted separately for correct and incorrect predictions [Color figure can be viewed at wileyonlinelibrary.com]

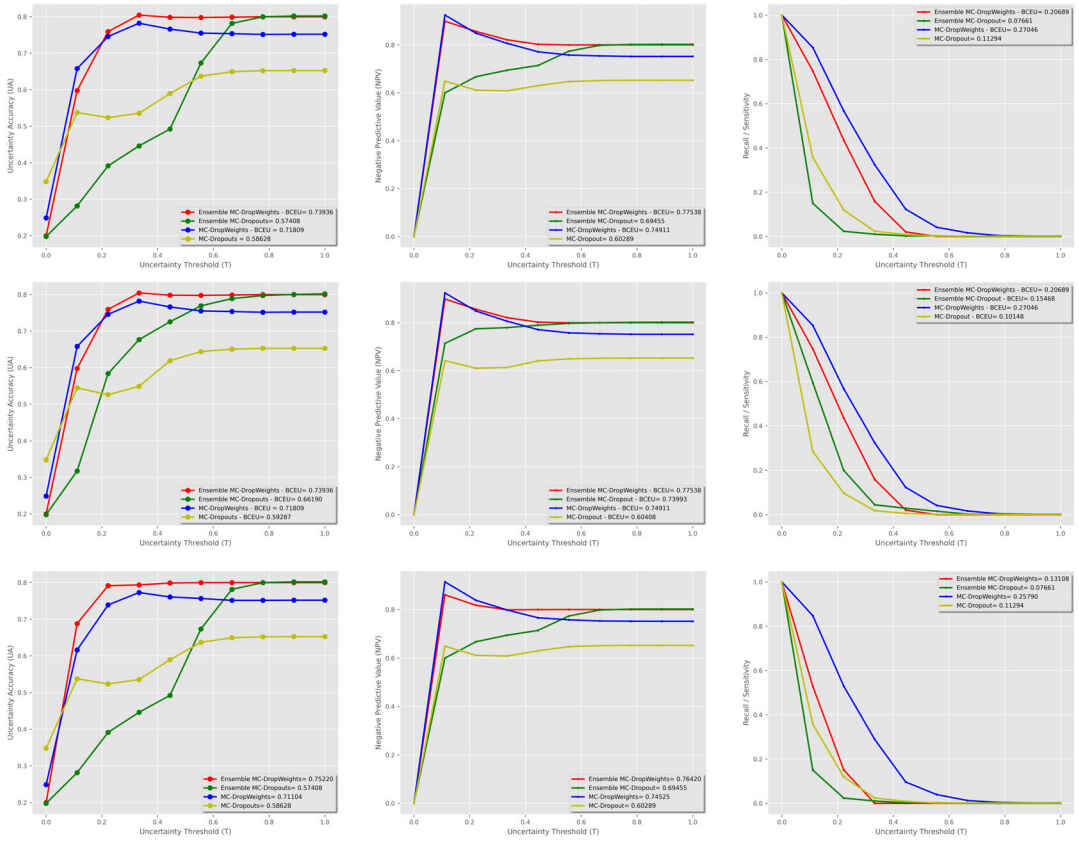

FIGURE 13 Comparison of using the Bayesian deep learning models with different uncertainty thresholds when applied to an imbalanced dataset result of multiclass classification of an MRI image dataset [Color figure can be viewed at wileyonlinelibrary.com]

In Figure 12B, we see the distribution of bias-corrected uncertainty values, grouped by correct and incorrect predictions for test images. Given a prediction is correct, there is a strong likelihood that the prediction uncertainty is also low. As a result, our model can confidently identify incorrectly classified images.

5.3.3 | Uncertainty-based classification performance comparison

As an application to the proposed uncertainty measures, we have evaluated the uncertainty estimation performance of Bayesian deep ensembles of MC-DropWeights with the Ensembles MC-Dropout, MC-Dropout, and MC-DropWeights using MI and bias-corrected uncertainty estimator (BCEU). Our experimental results (Figures 13 and 14) show, that BCEU using the ensemble MC-DropWeights model yields an improved prediction accuracy with the appropriate level of estimated uncertainty.

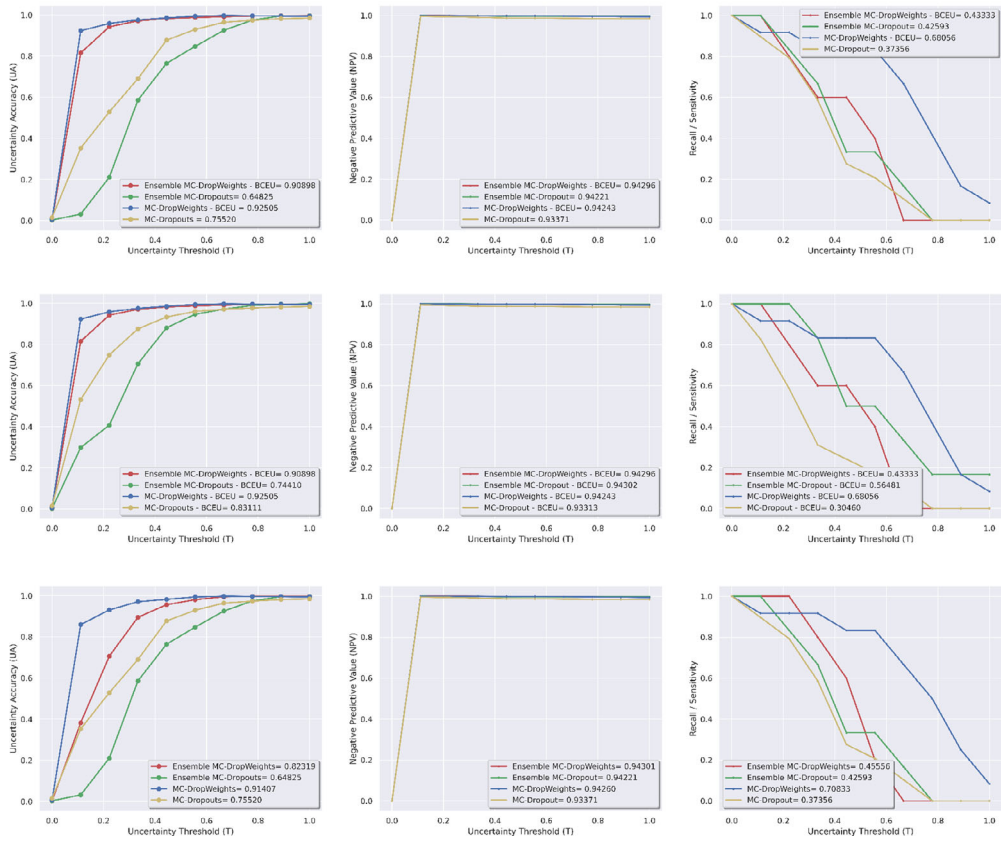

FIGURE 14 Comparison of using the Bayesian deep learning models with different uncertainty thresholds when applied to a balanced dataset result of multiclass classification of an MRI image dataset [Color figure can be viewed at wileyonlinelibrary.com]

6 | UNCERTAINTY QUALITY MATRICES

We have evaluated the quality of uncertainty estimates using five statistical matrices: Predictive log-likelihood (PLL), continuous ranked probability score (CRPS), negative log predictive density (NLPD), brier score (BS), and root mean squared error (RMSE).

The PLL and CRPS can be defined for test image (x_i, y_i), where F is cumulative distribution function (CDF) of the prediction and $\hat{\omega}_j$ is the parameter from posterior distribution of T stochastic feed-forward as below:

1. NLPD: NLPD takes the negative logarithm of the posterior class probabilities for classification and of the predictive density for regression. This loss penalizes both over and under-confident predictions but in general favours conservative models, that is models that tend to be under-confident rather than over-confident.

$$\text{NLPD (L)} = -\frac{1}{N}\sum_{i=1}^{N} logp(y_i = c_i|x_i), \tag{32}$$

NLPD infinitely penalizes wrong predictions made with zero uncertainty.

TABLE 2 Quality metrics

Metrics	Ensemble MCDW	Ensemble MCDO	MCDW	MCDO
Negative log predictive density (NLPD)	43.62	3.59	123.80	1313.34
Root mean square error (RMSE)	0.55	0.61	0.58	0.60
Predictive log likelihood (PLL)	0.25	0.27	0.27	0.28
Continuous ranked probability score (CRPS)	0.27	0.26	0.26	0.27
Brier score (BS)	0.66	0.71	0.65	0.70

2. RMSE: The RMSE is the standard deviation of the prediction errors. The higher the value, the greater the uncertainty.
3. PLL: PLL is a widely accepted metric as a marker for the quality of uncertainty, used as the main uncertainty quality metric in References 9 and 13. It captures how well a model fits the data. The key property is that PLL makes no assumptions about the form of the predictive distribution. There is no upper bound of PLL so larger values indicate better model fit. While PLL is an elegant measure, outliers have a negative effect on the score.

$$\mathrm{PLL}(f_\omega(x),(y_i,x_i)) = \log p(y_i|f_\omega(x_i)) = \log \int f_\omega(x_i,y_i)p(\omega|D)\omega \tag{33}$$

$$\approx \log \int f_\omega(x_i,y_i)q_\theta(\omega)\omega \approx \log \frac{1}{T}\sum_{j=1}^{T} p(y_i|f_{\hat{\omega}_j}(x_i)). \tag{34}$$

4. CRPS: CRPS is generally used to estimate respective accuracy of two probabilistic models. It generalizes mean absolute error for probabilistic estimation. It is a less sensitive measure that takes the full predicted PDF into account.[9]

$$\mathrm{CRPS}(f_\omega(x_i),(y_i,x_i)) = \int_{-\infty}^{\infty} (F(y) - 1(y \geq y_i))^2 y. \tag{35}$$

   In order for CRPS to be analytically tractable, we need to assume a Gaussian unimodal predictive distribution. A prediction with a low variance that is slightly offset from the true observation receives a higher score from CRPS than PLL. A perfect prediction with no variance yields a CRPS of 0; for all other cases, the value is greater than 0. CRPS also has no upper bound.
5. BS: The BS is a score function that measures the accuracy of probabilistic predictions. It calculates the mean squared difference between a binary label and its associated predicted probability. Therefore, in multiclass classification, the lower the BS, the better the predictions are calibrated.[14]

Our experimental results (Table 2, Figures 13 and 14) show that ensemble MC-DropWeights results improved prediction accuracy under estimated uncertainty. More importantly, the uncertainty quality metrics show a significant improvement when using ensemble MC-DropWeights.

7 | DISCUSSION

Our primary source of knowledge about the world is from our ability to learn from our observations. So how can a computer learn from data? Most of the work in that area targets well-defined, classic machine learning problems. Despite exceptional performance, the adoption of DL models in critical applications where security and safety is a concern, remains limited in part because

of its inability to interpret data being a black-box. Uncertainty quantification is an important challenge and one that the DL community needs to think about a lot more. DL based medical diagnosis has to express the uncertainty of an image in the same way as a doctor may express ambiguity and ask for expert advice. The existing approach to measure uncertainty in deep neural networks is often highly sensitive to hyperparameter choices. Furthermore, using the existing approach, it is hard to scale it up to higher dimensional digital image datasets and network architectures, limiting their general applicability in DL.

Here, we evaluate Basyesian neural networks with DropWeights, to measure uncertainty in cell nuclei segmentation. We also investigated the bias-corrected uncertainty distribution among the correctly and incorrectly classified brain tumour MRI images. It shows that estimated uncertainty provides an additional insight to a point estimate, which can help us better understand why our models predict certain inputs as being more aleatoric or epistemic. This can effectively improve the overall performance of the human-machine combination.

Though neural networks with DropWeights approximate Bayesian inference, to estimate uncertainty by Monte Carlo (MC) integration over the variational distribution, it is implemented by simulating the network stochastic forward passes multiple times with DropWeights turned on. Thus, the inference is theoretically scaled by the number of forward passes. Due to advancement of computer hardware (GPU and TPU), we can measure the uncertainty in almost real-time.

8 | CONCLUSION

In this article, we present an uncertainty estimation framework in DL for medical images , by decomposing the uncertainty into two categories. We present the first approach (to the best of our knowledge) of Monte-Carlo DropWeights and BRUNet, to model BNNs as a reliable, VI method, which accurately estimates the models' AU and EU. We also approximate the predictive uncertainty associated with every class in a multiclass setting, by calculating the mean of the standard deviations of the class probabilities. We have demonstrated that medical image segmentation and classification with uncertainty information provides additional insights into the corresponding analysis alongside point estimation, which can increase its ability to interpret the data, as well as improving confidence and so makes models based on DL more applicable in a medical setting. We address the "mode collapse" phenomenon in VI, by leveraging deep ensembles of Monte-Carlo DropWeights method. This intuitively captures the two sources of uncertainties and provides a baseline for the evaluation metrics for predictive uncertainty quantification. In this article, we introduce a bias-corrected uncertainty estimation function—a tractable approximation measurement of EU using Jackknife Estimator—which shows an improved performance in DL. The estimated uncertainty of the models is analyzed using probabilistic variants of metrics, such as negative predictive value (NPV), recall, and estimated uncertainty accuracy (EUA). We also evaluate the quality of estimated uncertainty measurement using RMSE, PLL, BS, and CRPS metrics.

Since the single DropWeights model collapses around a subspace of the posterior distribution, we have assumed that each model member of the ensemble will capture the behavior around a different local mode. This will require a more detailed theoretical analysis for future research.

The properties in the dataset, such as label correlations and label cardinality, can strongly affect the uncertainty quantification in predictive probability performance of a Bayesian DL algorithm in multilabel settings. There is no systematic study on how and why the performance varies over different data properties; any such study would be useful in deciding multilabel algorithms and active learning in medical imaging. Future research in this area should hopefully

include the extension of ideas for dataset shift, to represent better uncertainty estimates and the effect on the quality of uncertainty across different data modalities and network architecture.

ACKNOWLEDGMENTS

We would like to thank Dr Stephen Swift and all members of the Intelligent Data Analysis (IDA) Group, Brunel University, London for their valuable feedback on our manuscript.

AUTHOR CONTRIBUTIONS

B.G., A.T., B.S., and W. W. designed the concept of the study. B.S. and W. W. provided clinical and data expertise. B.G. performed the experiments. B.G. and A.T. wrote the manuscript. All authors reviewed the manuscript.

DATA AVAILABILITY STATEMENT

The data that support the findings of this study are available from the corresponding author upon reasonable request. All models were implemented using the TensorFlow and Keras library. All Codes are available from the corresponding author upon reasonable request.

ORCID

Biraja Ghoshal https://orcid.org/0000-0001-5456-2197

REFERENCES

1. Litjens G, Kooi T, Bejnordi BE, et al. A survey on deep learning in medical image analysis. *Med Image Anal*. 2017;42:60-88.
2. Blundell C, Cornebise J, Kavukcuoglu K, Wierstra D. Weight uncertainty in neural networks. Paper presented at: Proceedings of the 32nd International Conference on Machine Learning, Lille, France, JMLR: W&CP; 2015:37:1613-1622.
3. Gal Y. Uncertainty in Deep Learning [PhD thesis]. Department of Engineering, University of Cambridge, UK; 2016.
4. Kwon Y, Won JH, Kim BJ, Paik MC. Uncertainty quantification using Bayesian neural networks in classification: Application to biomedical image segmentation. *Computational Statistics & Data Analysis*. 2020;142:106816. http://dx.doi.org/10.1016/j.csda.2019.106816.
5. Depeweg S, Hernández-Lobato JM, Doshi-Velez F, Udluft S. Decomposition of uncertainty in Bayesian deep learning for efficient and risk-sensitive learning. Proceedings of the 35 th International Conference on Machine Learning, Stockholm, Sweden, PMLR; 2018;80:1184-1193.
6. Ghoshal B, Tucker A, Sanghera B, Wong W. Estimating uncertainty in deep learning for reporting confidence to clinicians when segmenting nuclei image data. Paper presented at: Proceedings of the 2019 IEEE 32nd International Symposium on Computer-Based Medical Systems (CBMS), Cordoba, Spain; June 2019:318-324. https://doi.org/10.1109/CBMS.2019.00072.
7. Neal RM. Bayesian learning via stochastic dynamics. *Advances in Neural Information Processing Systems*, San Mateo, California; 1993:475-482.
8. MacKay DJ. A practical Bayesian framework for backpropagation networks. *Neural Comput*. 1992;4(3):448-472.
9. Teye M, Azizpour H, Smith K. Bayesian uncertainty estimation for batch normalized deep networks. Proceedings of the 35 th International Conference on Machine Learning, Stockholm, Sweden, PMLR 80, 2018. arXiv preprint arXiv:1802.06455.
10. Gal Y, Ghahramani Z. Dropout as a Bayesian approximation: representing model uncertainty in deep learning. Paper presented at: Proceedings of the 33rd International Conference on Machine Learning, ICML 2016, New York, NY; 2016;3:1651-1660.
11. Kingma DP, Salimans T, Welling M. Variational Dropout and the Local Reparameterization Trick. *Stat*. 2015;1050:8.
12. Graves A. Practical variational inference for neural networks. *Advances in Neural Information Processing Systems; 2011*, Granada, Spain; 2011:2348-2356.

13. Hernández-Lobato D, Hernandez-Lobato J, Shah A, Adams R. Predictive entropy search for multi-objective bayesian optimization. Paper presented at: Proceedings of the 33 rd International Conference on Machine Learning, New York, NY; 2016:1492-1501.
14. Lakshminarayanan B, Pritzel A, Blundell C. Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in Neural Information Processing Advances in Neural Information Processing Systems*; Long Beach, CA. NIPS'17: Proceedings of the 31st International Conference on Neural Information Processing Systems; December 2017 ; 2017:6405-6416.
15. Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R. Dropout: a simple way to prevent neural networks from overfitting. *J Mach Learn Res*. 2014;15(1):1929-1958.
16. Osband I, Blundell C, Pritzel A, Van Roy B. Deep exploration via bootstrapped DQN. NIPS'16: Proceedings of the 30th International Conference on Neural Information Processing Systems; December 2016; 2016:4033-4041.
17. Lewandowski A. Batch normalized deep kernel learning for weight uncertainty. Paper presented at: Proceedings of the Conference on Neural Information Processing Systems, Long Beach, CA; 2017.
18. Kendall A, Gal Y. What uncertainties do we need in Bayesian deep learning for computer vision? *Advances in Neural Information Processing SyPublication*. NIPS'17: Proceedings of the 31st International Conference on Neural Information Processing Systems; Long Beach, CA; December 2017; 2017:5580-5590.
19. Caruana R, Lawrence S, Giles CL. Overfitting in neural nets: backpropagation, conjugate gradient, and early stopping. *Advances in Neural Information Processing Systems; Vancouver, British Columbia, Canada*. NIPS'00: Proceedings of the 13th International Conference on Neural Information Processing Systems; January 2000; 2001:381-387.
20. Wan L, Zeiler M, Zhang S, Le Cun Y, Fergus R. Regularization of neural networks using dropconnect. Paper presented at: Proceedings of the International Conference on Machine Learning, Atlanta, Georgia, USA; 2013:1058-1066.
21. LeCun Y, Bengio Y, Hinton G. Deep learning. *Nature*. 2015;521(7553):436.
22. Hassibi B, Stork DG. Second order derivatives for network pruning: optimal brain surgeon. *Advances in Neural Information Processing Systems 5, [NIPS Conference]*; Denver, Colorado, USA; November 1992; 1993:164-171.
23. Stiles J, Jernigan TL. The basics of brain development. *Neuropsychol Rev*. 2010;20(4):327-348.
24. Olshausen BA, Field DJ. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*. 1996;381(6583):607.
25. Goodfellow I, Bengio Y, Courville A. *Deep Learning*. Adaptive Computation and Machine Learning series One Rogers Street, Cambridge, MA: MIT Press; 2016. ISBN: 9780262035613.
26. Welling M, Teh YW. Bayesian learning via stochastic gradient Langevin dynamics. Paper presented at: Proceedings of the 28th International Conference on Machine Learning (ICML-11), Bellevue, WA; 2011:681-688.
27. Hinton G, van Cramp D. Keeping neural networks simple by minimising the description length of weights. Paper presented at: Proceedings of 6th Annual Conference on Computational Learning Theory; Santa Cruz, California, USA; July 1993; 1993:5-13.
28. Dietterich TG. Ensemble Methods in Machine Learning. In: Multiple Classifier Systems. MCS 2000. Lecture Notes in Computer Science, vol 1857. Springer, Berlin, Heidelberg; 2000:1-15. https://doi.org/10.1007/3-540-45014-9_1.
29. Yeung RW. A new outlook on Shannon's information measures. *IEEE Trans Inf Theory*. 1991;37(3):466-474.
30. Shannon CE. A mathematical theory of communication. *Bell Syst Techn J*. 1948;27(3):379-423.
31. Smith L, Gal Y. Understanding measures of uncertainty for adversarial example detection. Conference on Uncertainty in Artificial Intelligence (UAI); Monterey, California, USA; August 2018; 2018. arXiv preprint arXiv:1803.08533.
32. Houlsby N, Huszár F, Ghahramani Z, Lengyel M. Bayesian Active Learning for Classification and Preference Learning. *Stat*. 2011;1050:24.
33. Quenouille MH. Notes on bias in estimation. *Biometrika*. 1956;43(3/4):353-360.
34. Houlsby N. Efficient Bayesian Active Learning and Matrix Modelling [PhD thesis]. University of Cambridge; 2014.
35. Kirsch A, van Amersfoort J, Gal Y. BatchBALD: efficient and diverse batch acquisition for deep bayesian active learning; 33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, Canada; 2019.

36. Macke J, Murray I, Latham P. Estimation bias in maximum entropy models. *Entropy*. 2013;15(8):3109-3129.
37. Basharin GP. On a statistical estimate for the entropy of a sequence of independent random variables. *Theory Probab Appl*. 1959;4(3):333-336.
38. Harris B. *The Statistical Estimation of Entropy in the Non-parametric Case. Technical Report*. Wisconsin, Milwaukee: Wisconsin University-Madison Mathematics Research Center; 1975.
39. DasGupta A. *Asymptotic Theory of Statistics and Probability*. Berlin, Germany: Springer Science & Business Media; 2008.
40. Ronneberger O, Fischer P, Brox T. U-net: convolutional networks for biomedical image segmentation. Paper presented at: Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention; 2015:234-241; Springer, New York, NY.
41. Gu J, Wang Z, Kuen J, et al. Recent advances in convolutional neural networks. *Pattern Recogn*. 2018;77:354-377.
42. DeVries T, Taylor GW. Leveraging uncertainty estimates for predicting segmentation quality; 2018. arXiv preprint arXiv:1807.00502.
43. Badrinarayanan V, Kendall A, Cipolla R. Segnet: a deep convolutional encoder-decoder architecture for image segmentation. *IEEE Trans Pattern Anal Mach Intell*. 2017;39(12):2481-2495.
44. Chollet F. Keras documentation; 2015. https://keras.io**.**
45. Hamilton BA. Kaggle. 2018 data science bowl: find the nuclei in divergent images to advance medical discovery; 2018.
46. Valen DAV, Kudo T, Lane KM, et al. Deep learning automates the quantitative analysis of individual cells in live-cell imaging experiments. *PLoS Comput Biol*. nov 2016;12(11):e1005177.
47. Meijering E. Cell segmentation: 50 years down the road [life sciences]. *IEEE Signal Process Mag*. 2012;29(5):140-145.
48. Veta M, van Diest PJ, Kornegoor R, Huisman A, Viergever MA, Pluim JPW. Automatic Nuclei Segmentation in H&E Stained Breast Cancer Histopathology Images. *PLoS ONE*. 2013;8(7):e70221. http://dx.doi.org/10.1371/journal.pone.0070221.
49. Irshad H, Veillard A, Roux L, Racoceanu D. Methods for nuclei detection, segmentation, and classification in digital histopathology: a review - current status and future potential. *IEEE Rev Biomed Eng*. 2013;7:97-114.
50. Clark K, Vendt B, Smith K, et al. The cancer imaging archive (TCIA): maintaining and operating a public information repository. *J Digit Imag*. 2013;26(6):1045-1057.
51. Dickie DA, Job DE, Rodriguez D, et al. Brain imaging of normal subjects (BRAINS) age-specific MRI atlases from young adults to the very elderly (v1. 0), [dataset]. University of Edinburgh, Edinburgh Imaging, CCBS, BRAINS Imagebank; 2016. http://dx.doi.org/10.7488/ds/1369.
52. Malone IB, Cash D, Ridgway GR, et al. MIRIAD-public release of a multiple time point Alzheimer's MR imaging dataset. *NeuroImage*. 2013;70:33-36.
53. Balasooriya NM, Nawarathna RD. A sophisticated convolutional neural network model for brain tumor classification. Paper presented at: Proceedings of the 2017 IEEE International Conference on Industrial and Information Systems (ICIIS); 2017:1-5.
54. Mobiny A, Nguyen HV, Moulik S, Garg N, Wu CC. DropConnect is effective in modeling uncertainty of Bayesian deep networks; 2019. arXiv preprint arXiv:1906.04569.
55. He K, Zhang X, Ren S, Sun J. Delving deep into rectifiers: surpassing human-level performance on imagenet classification. Paper presented at: Proceedings of the IEEE International Conference on Computer Vision; 2015:1026-1034.

How to cite this article: Ghoshal B, Tucker A, Sanghera B, Lup Wong W. Estimating uncertainty in deep learning for reporting confidence to clinicians in medical image segmentation and diseases detection. *Computational Intelligence*. 2020;1–34. https://doi.org/10.1111/coin.12411

APPENDIX A. ALGORITHM

Algorithm 1. Training a Bayesian Neural Network with DropWeights

Input: Dataset: $D = \{(x_i, y_i)_{i=1}^{N}\}$, given C number of classes; Learning rate β; Number of epoch e; Dropweights rate p

Initialization: Model weights θ in neural network by He48

Result: Model with updated weights θ

for $i \leftarrow 1$ ***to*** e **do**

Forward Pass:

g is Convolutional Neural Network (CNN), with W_g being the CNN filters (and biases)

Extract Features from multilayered CNN: $v \leftarrow g(x; W_g)$

Random sample M mask: $M_{ij} \leftarrow Bernoulli(p)$

W is a fully connected weight matrix, a is a nonlinear activation function and M is the binary mask matrix

Compute activations: $r = a((M * W)v)$

Softmax function s takes input r and uses parameters W_k to map to a C dimensional output

Compute output: $o = r(s; W_s)$

Backpropagation:

Differentiate loss L_θ wrt to θ

Update softmax layer: $W_s = W_s - \beta * L_{W_s}$

Update Weights in DropWeights Layer: $W = W - \beta(M * L_w)$

Update Weights in hidden Layer: $W_g = W_g - \beta L_{W_g}$

end

Algorithm 2. Estimating Bias-Corrected Uncertainty with MC-DropWeights

Input: Dataset: $D = \{(\hat{x})\}$; Model f with optimized parameters θ;

Initialization: Dropweights rate r; Number of Inferences (/stochastic forward pass) T

Result: Mean prediction $\hat{y}$; Uncertainty σ

Reference3 [Gal, Y. 2016 (eq. (6.3) p.109, Prop. 4 p.149)]

$p = \{\}$

for $t \leftarrow 1$ ***to*** T **do**

Neural network with Dropweights rate r performs stochastic variational inference

Independently drawn a set of weights vector $(\hat{w}_t)_{t=1}^{T}$ from $q_{\hat{\theta}}(w)$

$\hat{p}_t = p \cup f^{\theta^t}(\hat{x}_t, r)$

end

Compute predictive probability: $\hat{p} = \frac{1}{T}\sum_{t=1}^{T}\hat{p}_t$

Compute prediction: $\hat{y} = argmax(\hat{p})$

for $t \leftarrow 1$ ***to*** T **do**

Compute the leave-one-out estimator: $\hat{p}_c^{-t} = \frac{1}{T-1}\sum_{j \neq t} p_{jc}$

Compute the Jackknife estimator of entropy: $\hat{H}_{-t} = -\sum_c \hat{p}_c^{-t} log(\hat{p}_c^{-t})$

end

Compute the bias-corrected uncertainty: $\sigma = \hat{H} + (T-1)(\hat{H} - \frac{1}{T}\sum_t \hat{H}^{-t})$