scientific reports

Check for updates

OPEN

DropConnect is effective in modeling uncertainty of Bayesian deep networks

Aryan Mobiny[1]✉, Pengyu Yuan[1], Supratik K. Moulik[2], Naveen Garg[3], Carol C. Wu[3] & Hien Van Nguyen[1]

Deep neural networks (DNNs) have achieved state-of-the-art performance in many important domains, including medical diagnosis, security, and autonomous driving. In domains where safety is highly critical, an erroneous decision can result in serious consequences. While a perfect prediction accuracy is not always achievable, recent work on Bayesian deep networks shows that it is possible to know when DNNs are more likely to make mistakes. Knowing what DNNs do not know is desirable to increase the safety of deep learning technology in sensitive applications; Bayesian neural networks attempt to address this challenge. Traditional approaches are computationally intractable and do not scale well to large, complex neural network architectures. In this paper, we develop a theoretical framework to approximate Bayesian inference for DNNs by imposing a Bernoulli distribution on the model weights. This method called Monte Carlo DropConnect (MC-DropConnect) gives us a tool to represent the model uncertainty with little change in the overall model structure or computational cost. We extensively validate the proposed algorithm on multiple network architectures and datasets for classification and semantic segmentation tasks. We also propose new metrics to quantify uncertainty estimates. This enables an objective comparison between MC-DropConnect and prior approaches. Our empirical results demonstrate that the proposed framework yields significant improvement in both prediction accuracy and uncertainty estimation quality compared to the state of the art.

Deep neural networks (DNNs) have revolutionized various applied fields, including engineering and computer science (such as AI, language processing and computer vision)[1–4], as well as the classical sciences (such as biology, physics, and medicine)[5–8]. DNNs can learn abstract concepts and extract desirable information from some high dimensional input. This is done through stacks of convolutions followed by appropriate non-linear rectifiers. DNNs alleviate the need for time-consuming hand-engineered algorithms. Due to the high model complexity, DNNs require a huge amount of data to regularize training and prevent the networks from over-fitting the training examples. This reduces their applicability in settings where data are scarce. This is often the case in scenarios where data collection is expensive or time-consuming, e.g. annotation of computed tomography scans by radiologists.

More importantly, popular deep learning models are often trained with maximum likelihood (ML) or maximum a posteriori (MAP) procedures, thus producing a point estimate but not an uncertainty value. In a classifier model, for example, the probability vector obtained at the end of the pipeline (the softmax output) is often erroneously interpreted as model confidence. In reality, a model can be uncertain in its predictions even with a high softmax output. In other words, the softmax probability is the probability that an input is a given class relative to the other classes; it does not help explain the model's overall confidence[9].

In applications of automated decision making or recommendation systems, which might involve life-threatening situations, information about the *reliability* of automated decisions is crucial to improve the system's safety. In other words, it is necessary to know how confident the model is about its predictions[10,11]. Understanding if the model is under-confident or falsely over-confident can inform users to perform necessary actions to ensure safety[9]. Take an automated cancer detection system as an example which might encounter an out-of-distribution test sample. A traditional DNN-based system makes unreasonable suggestions, and as a result may unjustifiably

[1]Department of Electrical and Computer Engineering, University of Houston, Houston, TX 77004, USA. [2]Triradiate Industries, Sugar Land, TX 77479, USA. [3]Department of Diagnostic Radiology, The University of Texas MD Anderson Cancer Center, Houston, TX 77030, USA. ✉email: amobiny@uh.edu

bias the expert. Given information about the model's confidence, an expert could rely more on his own judgment when the automated system is essentially guessing at random.

Most of the studies on uncertainty estimation techniques are inspired by Bayesian statistics. Bayesian Neural Networks (BNNs)[12] are the probabilistic version of the traditional NNs with a prior distribution on the weights of the network. Such networks are intrinsically suitable for generating uncertainty estimates as they produce a distribution over the output for a given input sample[13]. These probabilistic systems are computationally expensive for large neural network models due to the huge number of parameters and the intractable inference of the model posterior. This limitation has prompted the scientific community to develop scalable, approximated BNNs.

Variational inference[14] is the most common approach used for approximating the model posterior using a simple variational distribution such as the Gaussian distribution[15]. The parameters of the distribution are then set in a way that minimizes the difference to the true distribution (usually by minimizing the Kullback-Leibler divergence). The use of the Gaussian distribution considerably increases the required number of parameters and makes it computationally expensive. In this paper, we propose a mathematically-grounded method called Monte Carlo DropConnect (MC-DropConnect) to approximate variational inference in BNNs. The main contributions of this paper are:

1. We propose imposing the Bernoulli distribution *directly* to the weights of the deep neural network to estimate the posterior distribution over its weight matrices. We derive the required equations to show that this generalization provides a computationally tractable approximation of a BNN, only using the existing tools and no additional model parameters.
2. We propose metrics to evaluate the uncertainty estimation performance of the Bayesian models in the classification and segmentation settings. Using these metrics, we show that our method is superior compared to the recently proposed technique called MC-Dropout.
3. We make an in-depth analysis of the uncertainty estimations in both classification and segmentation settings to investigate the robust generalization of MC-DropConnect. Our extensive evaluations show that the proposed uncertainty-informed decision is able to significantly improve the prediction accuracy compared to standard techniques.

Our experimental results (achieved using the proposed method and metrics) provide a new benchmark for other researchers to evaluate and compare their uncertainty estimation in pursuit of safer and more reliable deep networks. The rest of this paper is organized as follows: works related to approximating Bayesian inference and estimating uncertainty are presented in Related Work section. The Methodology section explains our proposed method along with the mathematical proofs to approximate variational inference in deep neural networks. We then present our findings and their interpretations in the Experimental Results and Discussion section. Finally, Conclusion section concludes the paper with future research directions.

Related work

In recent years, many studies have been conducted on approximate Bayesian inference for neural networks using deterministic approaches[13], Markov Chain Monte Carlo with Hamiltonian Dynamics[16], and variational inference[15]. In particular, Neal et al. introduced the Hamiltonian Monte Carlo for Bayesian neural network learning which gives a set of posterior samples[16]. This method does not require the direct calculation of the posterior but is computationally prohibitive.

Recently, Gal et al.[17] showed that Dropout, a well-known regularization technique[18], is mathematically equivalent to approximate variational inference in the deep Gaussian process[19]. This method, commonly known as MC-Dropout, uses a Bernoulli approximating variational distribution on the network units and introduces no additional parameters for the approximate posterior. The main disadvantage of this method is that it often requires many forward-pass sampling which makes it resource-intensive[20]. Moreover, a fully Bayesian network approximated using this method (i.e. dropout applied to all layers) results in excessive regularization[21] that learns slowly and does not achieve high prediction accuracy. While Bernoulli dropout is the most common approach used in the literature due to its ease of use and computation speed, several dropout variations with other distributions such as Gaussian dropout have been studied[18,22]. Concrete dropout[23] was later proposed to use a continuous relaxation of dropout's discrete masks to allow for automatic tuning of the dropout probability in large models. However, it introduces bias to the gradients of the model and reduces its prediction performance. Motivated by concrete dropout, Boluki et al. proposed a learnable Bernoulli dropout (LBD) mechanism for general deep neural networks. In LBDs, the dropout probabilities are defined as variational parameters and are jointly trained with the other parameters of the DNN[24]. Their experimental results show that LBD is able to achieve improved accuracy and uncertainty estimates in image classification and semantic segmentation. Multiplicative Normalizing Flows[25] is another technique which is introduced as a family of approximate posteriors for the parameters of a variational BNN, capable of producing uncertainty estimates; this technique does not scale well with very large convolutional networks.

Another proposed approach is Deep Ensembles[26] which have been shown to achieve high-quality uncertainty estimates. This method takes the frequentist approach to estimate the model uncertainty by training several models and calculating the variance of their output prediction. This technique is quite resource-intensive as it requires the storage of several separate models while performing forward passes through all of them to generate the inference. An alternative to such methods was proposed by Devries et al. which proposes to *learn* uncertainty from the given input[27].

Several approaches have been designed to compute the uncertainty estimates in the segmentation setting. The most commonly used approach is to induce a probability distribution by using dropout over extracted feature

values to obtain independent pixel-wise probabilities[21,28]. However, these approaches have been shown to be prone to result in inconsistent outputs which is not plausible[29]. In contrast, a body of work designed various approaches that can result in a diverse set of outcomes to account for the inherent ambiguities observed in real-world applications. Several approaches trained models with oracle set loss which only accounts for the closest prediction to the ground truth[30–32]. Kohl et al. proposed the probabilistic U-Net, in which a separate network named prior-net is trained along with the base segmentation network and maps the input to an embedding hypothesis space[29]. Thus this network is able to generate multiple plausible segmentations with sampling different points from the learned hypothesis embedding space.

Methodology

In this section, we address the limitations of BNNs, variational inference as the standard technique in Bayesian modeling, and DropConnect as a method for regularizing NNs. We then use these tools to approximate Bayesian networks using standard NNs equipped with Bernoulli distributions applied *directly* to their weights. Finally, we explain the methods used for measuring and evaluating model uncertainty.

Bayesian neural networks. From a probabilistic perspective, standard NN training via optimization is equivalent to maximum likelihood estimation (MLE) for the weights. Using MLE ignores any uncertainty that we may have in the proper weight values. BNNs are the extension over NNs to address this shortcoming by placing a prior distribution (often a Gaussian) over a NN's weight. This brings vital advantages like automatic model regularization and uncertainty estimates on predictions[13,15].

Given a BNN model with L layers parametrized by weights $\mathbf{w} = \{\mathbf{W}_i\}_{i=1}^{L}$ and a dataset $\mathscr{D} = (\mathbf{X}, \mathbf{y})$, Bayesian inference calculates the posterior distribution of the weights given the data, $p(\mathbf{w}|\mathscr{D})$. The predictive distribution of an unknown label $\mathbf{y}^*$ of a test input data $\mathbf{x}^*$ is given by:

$$p(\mathbf{y}^*|\mathbf{x}^*, \mathscr{D}) = \mathbb{E}_{p(\mathbf{w}|\mathscr{D})}[p(\mathbf{y}^*|\mathbf{x}^*, \mathbf{w})] = \int p(\mathbf{y}^*|\mathbf{x}^*, \mathbf{w}) p(\mathbf{w}|\mathscr{D})\, d\mathbf{w} \tag{1}$$

which shows that making a prediction about the unknown label is equivalent to using an ensemble of an infinite number of neural networks with various configuration of the weights. This is computationally intractable for neural networks of any size; the posterior distribution $p(\mathbf{w}|\mathscr{D})$ cannot generally be evaluated analytically. This limitation has prompted the scientific community to develop ways to approximate BNNs to make them easier to train[33,34].

One common approach is to use variational inference to approximate the posterior distribution of the weights. It introduces a variational distribution, $q_\theta(\mathbf{w})$, parametrized on θ that minimizes the Kullback-Leibler (KL) divergence between q and the true posterior distribution:

$$\mathrm{KL}(q_\theta(\mathbf{w})||p(\mathbf{w}|\mathscr{D})) \tag{2}$$

Minimising the KL divergence is equivalent to minimizing the negative evidence lower bound (ELBO):

$$\mathscr{L}(\theta) = -\int q_\theta(\mathbf{w}) \log p(\mathbf{y}|\mathbf{X}, \mathbf{w})\, d\mathbf{w} + \mathrm{KL}(q_\theta(\mathbf{w})||p(\mathbf{w})) \tag{3}$$

with respect to variational parameter θ. The first term (commonly referred to as the *expected log likelihood*) encourages $q_\theta(\mathbf{w})$ to place its mass on configurations of the latent variable that explain the observed data. The second term (referred to as *prior KL*) encourages $q_\theta(\mathbf{w})$ to be similar to the prior, preventing the model from overfitting. The prior KL term can be analytically evaluated to properly select the prior and variational distributions, while the expectation (i.e. integral term) cannot be computed exactly for a non-linear neural network. Our goal in the next section is to develop an explicit and accurate approximation for this expectation. Our approach extends on the results of Gal et al.[35] and uses Bernoulli approximating variational inference and Monte-Carlo sampling.

DropConnect. DropConnect[36], known as the generalized version of Dropout[18], is a method used for regularizing deep neural networks. Here, we briefly review Dropout and DropConnect applied to a single fully-connected layer of a standard NN. For a single K_{i-1} dimensional input $\mathbf{v}$, the ith layer of an NN with K_i units would output a K_i dimensional activation vector $\mathbf{a}_i = \sigma(\mathbf{W}_i\mathbf{v})$ where $\mathbf{W}_i$ is the $K_i \times K_{i-1}$ weight matrix and $\sigma(.)$ is the nonlinear activation function (biases included in the weight matrix with a corresponding fixed input of one for the ease of notation).

When Dropout is applied to the output of a layer, the output activations can be written as $\mathbf{a}_i^{\mathrm{DO}} = \sigma(\mathbf{z}_i \odot (\mathbf{W}_i\mathbf{v}))$ where $\odot$ signifies the Hadamard product and $\mathbf{z}_i$ is a K_i dimensional binary vector with its elements drawn independently from $z_i^{(k)} \sim \mathrm{Bernoulli}(p_i)$ for $k = 1, \ldots, K_i$ and p_i to be the probability of keeping the output activation. DropConnect is the generalization of Dropout where the Bernoulli dropping is applied directly to each weight, rather than each output unit, thus the output activation is re-written as $\mathbf{a}_i^{\mathrm{DC}} = \sigma((\mathbf{Z}_i \odot \mathbf{W}_i)\mathbf{v})$. Here, $\mathbf{Z}_i$ is the binary matrix of the same shape as $\mathbf{W}_i$, i.e. $K_i \times K_{i-1}$. Wan et al.[36] showed that adding DropConnect helps regularize large neural network models and outperforms Dropout on a range of data sets.

DropConnect for approximate Bayesian neural network. Assume the same Bayesian NN with L layers parametrized by weights $\mathbf{w} = \{\mathbf{W}_i\}_{i=1}^{L}$. We perform variational learning by approximating the variational distribution $q(\mathbf{W}_i|\boldsymbol{\Theta}_i)$ for every layer i as:

$$\mathbf{W}_i = \mathbf{\Theta}_i \odot \mathbf{Z}_i \tag{4}$$

where $\mathbf{\Theta}_i$ is the matrix of variational parameters to be optimised, and $\mathbf{Z}_i$ the binary matrix whose elements are distributed as:

$$z_i^{(l,k)} \sim \text{Bernoulli}(p_i) \quad \text{for} \quad i = 1, \ldots, L \tag{5}$$

Here, $z_i^{(l,k)}$ is the random binary value associated with the weight connecting the lth unit of the $(i-1)$th layer to the kth unit of the ith layer. p_i is the probability that the random variables $z_i^{(l,k)}$ take the value one (assuming the same probability for all the weights in a layer). Therefore, $z_i^{(l,k)} = 0$ corresponds to the weight being dropped out.

We start with rewriting the first term of Eq. (3) as a sum over all samples. Then we use Eq. (4) to re-parametrize the integrand so that it only depends on the Bernoulli distribution instead of $\mathbf{w}$ directly. We estimate the intractable integral with Monte Carlo sampling over $\mathbf{w}$ with a single sample as:

$$-\int q_\theta(\mathbf{w}) \log p(\mathbf{y}|\mathbf{X}, \mathbf{w})\, d\mathbf{w} = \sum_{n=1}^{N} \int -q_\theta(\mathbf{w}) \log p(\mathbf{y}_n|\mathbf{x}_n, \mathbf{w}) = \frac{1}{N} \sum_{n=1}^{N} -\log p(\mathbf{y}_n|\mathbf{x}_n, \hat{\mathbf{w}}_n) \tag{6}$$

Note that $\hat{\mathbf{w}}_n$ is not maximum a posteriori estimate, but random variable realisations from the Bernoulli distribution, $\hat{\mathbf{w}}_n \sim q_\theta(\mathbf{w})$, which is identical to applying DropConnect to the weights of the network. The final sum of the log probabilities is the loss of the NN, thus we set:

$$\mathrm{I}_{\mathrm{NN}}(\mathbf{y}_n, \hat{\mathbf{y}}(\mathbf{x}_n, \hat{\mathbf{w}}_n)) = -\log p(\mathbf{y}_n|\mathbf{x}_n, \hat{\mathbf{w}}_n) \tag{7}$$

where $\hat{\mathbf{y}}(\mathbf{x}_n, \hat{\mathbf{w}}_n)$ is the random output of the BNN. I_{NN} is defined according to the task with the sum of squared loss and softmax loss commonly selected for the regression and classification respectively.

The second term in Eq. (3) can be approximated following[35]. It has been shown that the KL term is equivalent to $\sum_{i=1}^{L} ||\Theta_i||_2^2$. Thus, the objective function can be re-written as:

$$\hat{\mathscr{L}}_{\mathrm{MC}} = \frac{1}{N} \sum_{n=1}^{N} \mathrm{I}_{\mathrm{NN}}(\mathbf{y}_n, \hat{\mathbf{y}}_n) + \lambda \sum_{i=1}^{L} ||\Theta_i||_2^2 \tag{8}$$

which is a scaled unbiased estimator of Eq. (3). More interestingly, it is identical to the objective function used in a standard neural network with L2 weight regularization and DropConnect applied to all the weights of the network. Therefore, training such a neural network with stochastic gradient descent has the same effect as minimizing the KL term in Eq. (2). This scheme, similar to a BNN, results in a set of parameters that best explains the observed data while preventing over-fitting.

After training the NN with DropConnect and proper regularization, we follow Eq. (1) to generate our inference. We replace the posterior $p(\mathbf{w}|\mathscr{D})$ with the approximate posterior distribution $q_\theta(\mathbf{w})$ and approximate the integral with Monte Carlo integration:

$$p(\mathbf{y}^*|\mathbf{x}^*, \mathscr{D}) \approx \int p(\mathbf{y}^*|\mathbf{x}^*, \mathbf{w}) q_\theta(\mathbf{w})\, d\mathbf{w} \approx \frac{1}{T} \sum_{t=1}^{T} p(\mathbf{y}^*|\mathbf{x}^*, \hat{\mathbf{w}}_t) = p_{\mathrm{MC}}(\mathbf{y}^*|\mathbf{x}^*) \tag{9}$$

with $\hat{\mathbf{w}}_t \sim q_\theta(\mathbf{w})$. This means that at test time, unlike common practice, the DropConnect layers is kept active to keep the Bernoulli distribution over the network weights. Then each forward pass through the trained network generates a Monte Carlo sample from the posterior distribution. Several of such forward passes are needed to approximate the posterior distribution of softmax class probabilities. According to Eq. (9), the mean of these samples can be interpreted as the network prediction. We call this approach MC DropConnect which is a generalization over the previous work referred to as MC Dropout[35] and will show its superiority in terms of achieving higher prediction accuracy and more precise uncertainty estimation in different ML tasks.

Measuring the model uncertainty. Generally, there are two types of uncertainty in Bayesian modeling[10]. Model uncertainty, also known as Epistemic uncertainty, measures what the model does not know due to the lack of training data. This uncertainty captures our ignorance about which model generated our collected data, thus can be explained away given enough data[9]. Aleatoric uncertainty, however, captures noise (such as motion or sensor noise) that is inherently present in the data and cannot be reduced by collecting more data[28].

After computing the result of stochastic forward passes through the model, we can estimate the model confidence to its output. In the classification setting, several metrics are introduced to measure uncertainty. One straightforward approach used by Kendall *et al.* is to take the *variance* of the MC samples from the posterior distribution as the output model uncertainty for each class[21]. Predictive entropy is also suggested by Gal et al. which captures both epistemic and aleatoric uncertainty; in our case, this is not the proper choice as we are interested in regions of the data space where the model is uncertain[9].

To specifically measure the model uncertainty for a new test sample $\mathbf{x}^*$, we can see it as the amount of information we would gain about the model parameters if we were to receive the true label $\mathbf{y}^*$. Theoretically, if the model is well-established in a region, knowing the output label conveys little information. In contrast, knowing the label would be informative in regions of data space where the model is uncertain[37]. Therefore, the mutual information (MI) between the true label and the model parameters are defined as as:

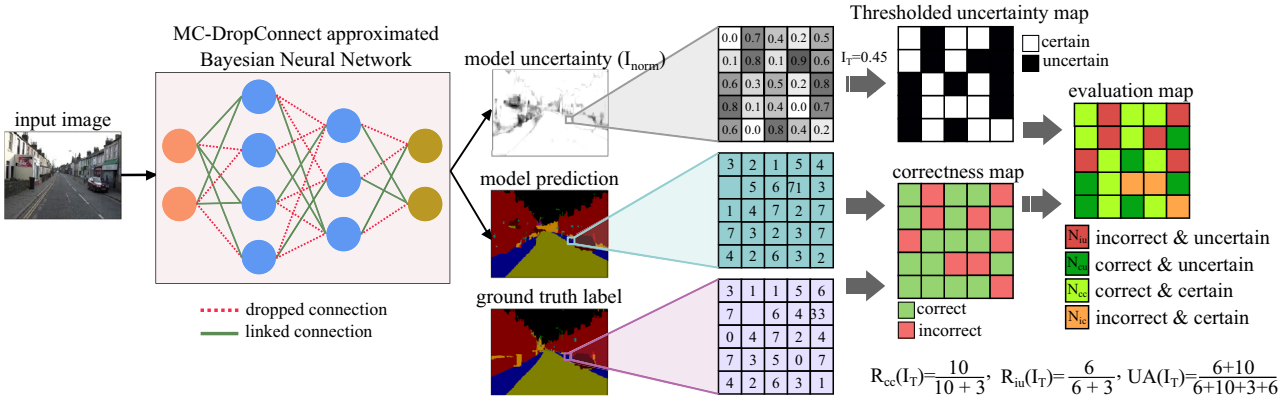

Figure 1. Overview of the proposed approximate Bayesian model (Left) and metrics to evaluate the uncertainty quality (Right) in a semantic segmentation example. Model uncertainty (I) is estimated as the amount of mutual information between the model parameters and the true label. I_T is the uncertainty threshold which divides the prediction into certain ($I_{\text{norm}} < I_T$) and uncertain ($I_{\text{norm}} > I_T$) groups. Since segmentation is identical to pixel-wise classification, similar computations hold true for the classification task.

$$I(\mathbf{y}^*, \mathbf{w}|\mathbf{x}^*, \mathscr{D}) = H(\mathbf{y}^*|\mathbf{x}^*, \mathscr{D}) - \mathbb{E}_{p(\mathbf{w}|\mathscr{D})} H[p(\mathbf{y}^*|\mathbf{x}^*, \mathbf{w})] \tag{10}$$

where given the training data set $\mathscr{D}$, $\mathbf{y}^*, I(\mathbf{y}^*, \mathbf{w}|\mathbf{x}^*, \mathscr{D})$ measures the amount of information we gain about the model parameters $\mathbf{w}$ by receiving a test input $\mathbf{x}^*$ and its corresponding true label, $\mathbf{y}^*$. This can be approximated using the Bayesian interpretation of DropConnect derived earlier. H is the entropy, commonly referred to as the *predictive entropy*, which captures the existing amount of information in the predictive distribution:

$$H(\mathbf{y}^*|\mathbf{x}^*, \mathscr{D}) = -\sum_c p(\mathbf{y}^* = c|\mathbf{x}^*, \mathscr{D}) \log p(\mathbf{y}^* = c|\mathbf{x}^*, \mathscr{D}) \tag{11}$$

where c ranges over all classes. This is not analytically tractable for deep NNs; we use Eq. (9) to approximate it as:

$$\hat{H}(\mathbf{y}^*|\mathbf{x}^*, \mathscr{D}) = -\sum_c p_{\text{MC}}(\mathbf{y}^* = c|\mathbf{x}^*) \log p_{\text{MC}}(\mathbf{y}^* = c|\mathbf{x}^*) \tag{12}$$

where $p_{\text{MC}}(\mathbf{y}^* = c|\mathbf{x}^*)$ is the average of the softmax probabilities of input $\mathbf{x}^*$ being in class c over T Monte Carlo samples. Finally, MI can be re-written as:

$$\hat{I}(\mathbf{y}^*, \mathbf{w}|\mathbf{x}^*, \mathscr{D}) = \hat{H}(\mathbf{y}^*|\mathbf{x}^*, \mathscr{D}) + \sum_c \frac{1}{T} \sum_{t=1}^{T} p(\mathbf{y}^* = c|\mathbf{x}^*, \hat{\mathbf{w}}_t) \log p(\mathbf{y}^* = c|\mathbf{x}^*, \hat{\mathbf{w}}_t) \tag{13}$$

which can be computed for each model configuration at tth Monte Carlo run, $\hat{\mathbf{w}}_t$, obtained by the DropConnect. Note that the range of the obtained uncertainty values is not fixed across different data sets, network architectures, number of MC samples, etc. Therefore, we use the normalized mutual information $I_{\text{norm}} \in [0, 1]$ computed as $I_{\text{norm}} = \frac{I - I_{\min}}{I_{\max} - I_{\min}}$ to report our results and facilitate the comparison across various sets and configurations. $I_{\min}$ and $I_{\max}$ are the minimum and maximum uncertainty values computed over the whole data set.

Uncertainty evaluation metrics. The proposed MC-DropConnect approach is a light-weight, scalable method to approximate Bayesian inference in deep neural networks. This enables us to perform inference and estimate the uncertainty in DNNs at once. Unlike model predictions, there is no ground truth for uncertainty values which makes evaluating the uncertainty estimates a challenging task. Therefore, there is no clear and direct approach to define a good uncertainty estimate.

We propose metrics that incorporate the ground-truth label, model prediction, and uncertainty value to evaluate the uncertainty estimation performance of such models. Figure 1 shows the required processing steps to prepare these quantities for our metrics in a segmentation example. Note that these metrics can be used for both classification and semantic segmentation tasks; semantic segmentation is identical to pixel-wise classification. The conversions applied to a pixel explains the classification task.

We first compute the map of *correct* and *incorrect* values (correctness map) by matching the ground truth labels and model predictions. Likewise, we can apply a threshold $I_T \in [0, 1]$ on the continuous uncertainty estimation values of I_{norm} to split the predictions into *certain* ($I_{\text{norm}} < I_T$) and *uncertain* ($I_{\text{norm}} > I_T$) groups. Therefore, when making inference in the Bayesian setting, we generally face four scenarios which are incorrect-uncertain (*iu*), correct-uncertain (*cu*), correct-certain (*cc*), and incorrect-certain (*ic*) predictions (see Fig. 1). The following metrics reflects the characteristics of a good uncertainty estimator:

1. Correct-certain ratio (R_{cc}): If a model is certain about its prediction, the prediction has the highest probability of being correct. This can be written as a conditional probability:

$$R_{cc}(I_T) = P_{I_T}(\text{correct}|\text{certain}) = \frac{P(\text{correct, certain})}{P(\text{certain})} = \frac{\text{N}_{cc}}{\text{N}_{cc} + \text{N}_{ic}} \tag{14}$$

where N represents the count for each combination and R represents the ratio.

2. Incorrect-uncertain ratio (R_{iu}): If a model is making an incorrect prediction, it is desirable for the uncertainty to be high.

$$R_{iu}(I_T) = P_{I_T}(\text{uncertain}|\text{incorrect}) = \frac{P(\text{uncertain, incorrect})}{P(\text{incorrect})} = \frac{\text{N}_{iu}}{\text{N}_{iu} + \text{N}_{ic}} \tag{15}$$

In this scenario, the model is capable of flagging a wrong prediction with a high epistemic uncertainty value to help the user take further precautions.

Note that the converse of the above two assumptions is not necessarily the case. This means that if a model is making a correct prediction on a sample, it does not necessarily need to be certain. A model might, for instance, be able to correctly detect an object, but with a relatively higher uncertainty because it has rarely seen that instance with such a pose or condition.

3. Uncertainty Accuracy (UA): Finally, the overall accuracy of the uncertainty estimation can be measured as the ratio of the desired cases explained above (N_{cc} and N_{iu}) over all possible cases:

$$\text{UA}(I_T) = \frac{\text{N}_{cc} + \text{N}_{iu}}{\text{N}_{cc} + \text{N}_{iu} + \text{N}_{cu} + \text{N}_{ic}} \tag{16}$$

Clearly, for all the metrics proposed above, higher values correspond to the model that performs better. The value of these metrics depend on the uncertainty threshold, thus we plot each metric w.r.t the uncertainty threshold (I_T) and compare them using the area under each curve (AUC) metric. This helps to summarize the value of each metric over various uncertainty thresholds in a single scalar.

Medical data collection methodology. Our paper performs all medical data collection following relevant guidelines and regulations. Specifically, all CT scans were anonymized to remove any patient-specific information. Our protocol waives the patient consent as the data were de-identified (Protocol PA12-1084). The data collection was approved by the Institutional Review Board 4 of the MD Anderson Cancer Center whose chair designee is Vera J. DeLaCruz (IRB 4 IRB00005015).

Experimental results and discussion

In this section, we assess the performance of uncertainty estimates obtained from DropConnect CNNs on the tasks of classification and semantic segmentation. We also compare the uncertainty obtained from our proposed method with a state-of-the-art method, MC-Dropout, on a range of data sets and show considerable improvement in prediction accuracy and uncertainty estimation quality. We quantitatively evaluate the uncertainty estimates using our proposed evaluation metrics. Note that in all experiments throughout the paper, MC-Dropconnect and MC-Dropout techniques were never used simultaneously in the same network. If a network is trained with Dropout or Dropconnect regularization, it would be tested with the same Dropout or Dropconnect, respectively. All the experiments are done using TensorFlow (version 1.13.1) framework[38].

Classification. We implement fully Bernoulli Bayesian CNNs using DropConnect to assess the theoretical insights explained above in the classification setting. We show that applying the mathematically principled DropConnect to all the weights of a CNN results in a test accuracy comparable with the state-of-the-art techniques in the literature while considerably improving the models' uncertainty estimation.

We adopt the LeNet structure (described in[39]) for the MNIST[40] and a fully-convolutional network (FCNet) for the CIFAR-10 dataset[41]. FCNet is composed of three blocks, each containing two convolutional layers (filter size of three and stride of one) followed by a max-pooling layer (with filter size and stride of two). The numbers of filters in the convolution layers of the three blocks are 32, 64, and 128, respectively. Each convolutional layer is also followed by a batch normalization layer and Relu non-linear activation function. We refer to the tests applied to the Bayesian CNN with DropConnect applied to all the weights of the network as "MC-DropConnect" and will compare it with "None" (no dropout or drop connect), as well as "MC-Dropout"[20] which has dropout used after all layers. To make the comparison fair, Dropout and DropConnect are applied with the same rate of $p = 0.5$. We evaluate the networks using two testing techniques. The first is the standard test applied to each structure keeping everything in place (no weight or unit drop). The second test incorporates the Bayesian methodology, generating the MC test equivalent to model averaging over $T = 100$ stochastic forward passes.

Our experimental results (Table 1, Fig. 2) show that MC-DropConnect yields marginally improved prediction accuracy when applying MC-sampling. More importantly, the uncertainty estimation metrics show a significant improvement when using MC-DropConnect. Example predictions are provided in Fig. 3. We also test the LeNet networks (trained on MNIST) on rotated and background MNIST data. These are the distorted versions of MNIST which can be assumed as the out-of-distribution examples[42] that the model has never seen before. This test is conducted to investigate the generalization of the predictive uncertainty to domain shift.

As shown in Fig. 3, MC-DropConnect BNN often yields a high uncertainty estimate when the prediction is wrong and makes accurate predictions when it is certain. We observed fewer failure cases using MC-DropConnect compared with MC-Dropout (also reflected in the R_{iu} and R_{cc} values in Fig. 2). Similar observations were made in Fig. 4 which illustrates the distribution of the model uncertainty over the correct and incorrect predictions separately. It implies that the MC-DropConnect approximation produces significantly higher model

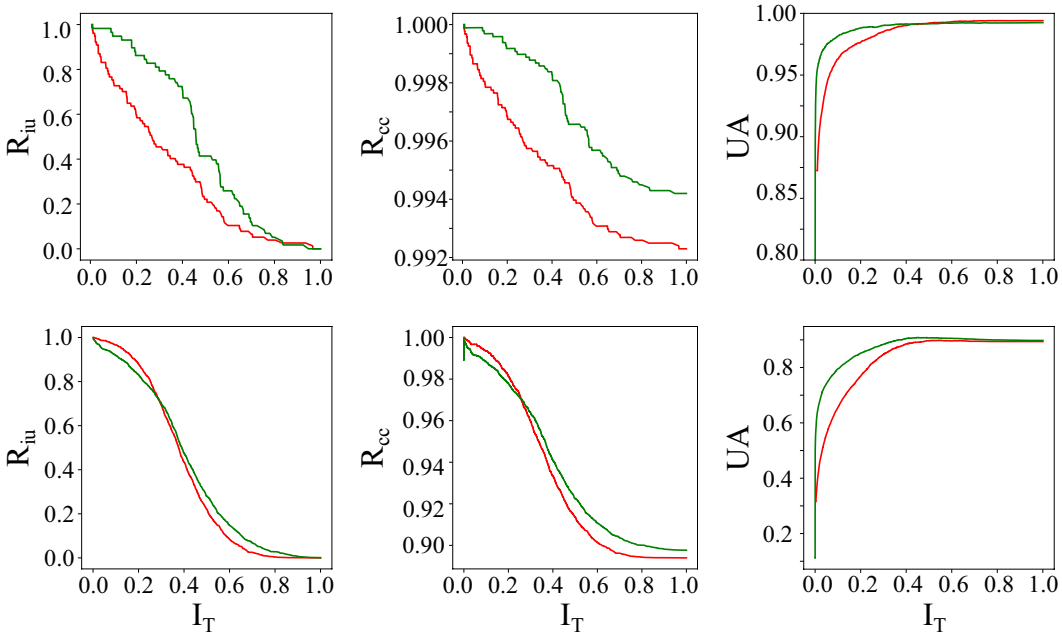

Figure 2. Illustrating the quantitative uncertainty estimation performance for the classification task using the proposed evaluation metrics. Note that when varying the uncertainty threshold, our proposed MC-DropConnect approximated BNN (shown in green) generally performs better than MC-Dropout (shown in red) for both MNIST (Top) and CIFAR-10 (Bottom) datasets.

uncertainty values (Kolmogorov-Smirnov test yields p value < 0.001) when the prediction is erroneous. Thus, this adds complementary information to the conventional network output which can be leveraged by the automated system to reject the prediction and send it for further inspection.

MC-DropConnect is also observed to yield informative correct predictions with high uncertainty estimation values. Examples are highlighted with red boundaries in Fig. 3. These cases often correspond to visually complicated samples where the network is not confident. Such FPs are useful and can be considered red flags when a model is more likely to make inaccurate predictions.

Enhanced performance with uncertainty-informed referrals. An uncertainty estimation with such characteristics (i.e. high uncertainty as an indication of erroneous prediction, as well as informative FPs) provides valuable information in situations where the control is handed to automated systems in real-life settings, with the possibility of becoming life-threatening to humans. These include applications such as self-driving cars, autonomous control of drones, automated decision making and recommendation systems in the medical domain, etc. An automated cancer detection system, for example, trained on a limited number of data (which is often the case due to the expensive or time-consuming data collection process) could encounter test samples lying out of its observed data distribution. Therefore, it is prone to making unreasonable decisions or recommendations which could result in a biased decision being made by the expert. However, uncertainty estimation can be utilized in such scenarios to detect such undesirable behavior of the automated systems and enhance the overall performance by flagging appropriate subsets for further analysis.

We set up an experiment to test the usefulness of the proposed uncertainty estimation in mimicking the clinical work-flow, and referring samples with high uncertainty for further testing. First, the model predictions are sorted according to their corresponding epistemic uncertainty (measured by the mutual information metric). We then computed the prediction accuracy as a function of confidence. This is done by taking various levels of tolerated uncertainty and the fraction of retained data (see Fig. 5). We observed a monotonic increase in prediction accuracy with MC-DropConnect outperforming MC-Dropout for decreasing levels of tolerated uncertainty and a decreasing fraction of retained data. It is also compared with removing the same fraction of samples randomly, that is with no use of uncertainty information, which indicates the informativeness of the uncertainty about prediction performance as well. Note that in practice, the uncertainty cutoff threshold should be selected by taking the threshold that results in the best prediction performance on the validation dataset, and should not be changed when using the test set.

Convergence of the MC-DropConnect. Even though the proposed MC-DropConnect method results in better prediction accuracy and uncertainty estimation, it still comes with a price of prolonged test time. This is because we need to evaluate the network stochastically multiple times and average the results. Therefore, while the training time of the models and their probabilistic variant is identical, the test time is scaled by the number of averaged forward passes. This becomes more important in practice and for applications which the test-time efficiency is critical. To evaluate the MC-DropConnect approximation method, we assessed the prediction accu-

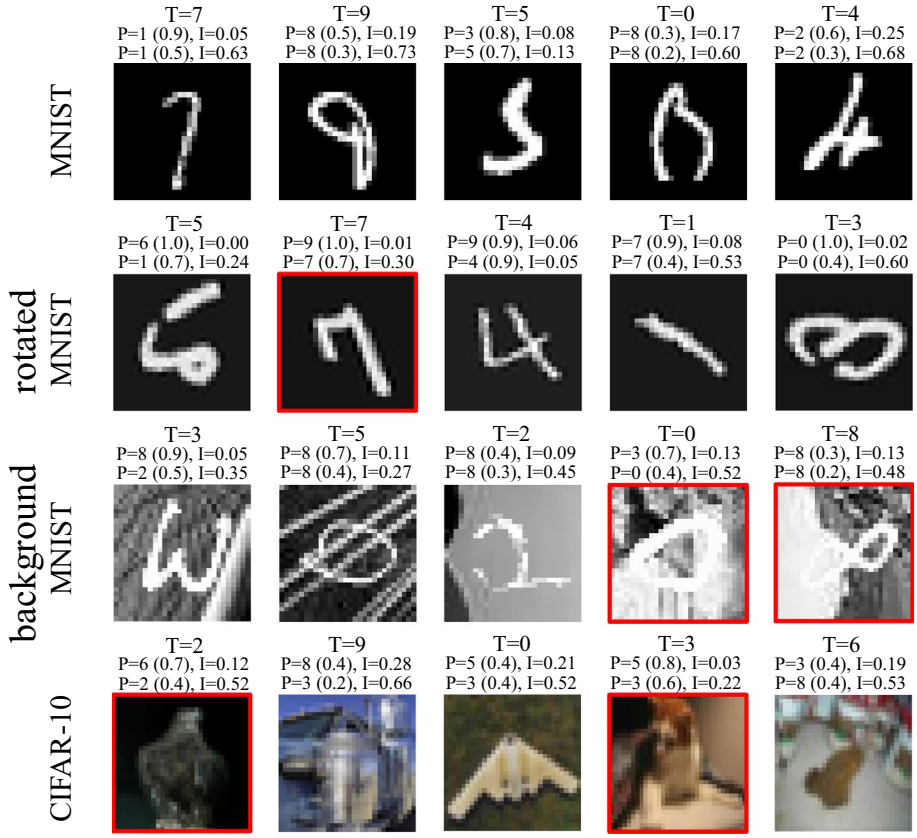

Figure 3. Sample model prediction and uncertainty estimation results on MNIST, rotated MNIST, background MNIST, and CIFAR-10 datasets. T: ground-truth label, P: model prediction (with the average MC prediction probability of the predicted class provided in the parentheses), and I: model uncertainty estimation. For each sample, the second and third line of the provided information are corresponding to MC-Dropout and MC-DropConnect respectively. The red boundary around images highlights correct-uncertain predictions of MC-DropConnect method.

	Prediction error (%)		Uncertainty metrics AUC (%)		
	Standard	MC-sampling	R_{iu}	R_{cc}	UA
MNIST (LeNet-5)					
None	0.99	–	–	–	–
MC-Dropout	0.75	0.77	31.24	98.77	97.48
MC-DropConnect	**0.70**	**0.57**	**41.67**	**99.57**	**98.87**
CIFAR-10 (FCNet)					
None	12.00	–	–	–	–
MC-Dropout	**10.92**	10.57	38.24	92.12	82.89
MC-DropConnect	11.34	**10.15**	**40.29**	**94.31**	**87.27**

Table 1. Test prediction error (%) and uncertainty estimation performance of the LeNet and FCNet networks and their Bayesian estimates on the MNIST and CIFAR-10 datasets. The models with the best performances are shown in bold.

racy of the FCNet on CIFAR-10 dataset and over a different number of Monte Carlo simulations (T). We then reported the average results over 10 runs in Fig. 6. As can be seen, MC-DropConnect results in a significantly lower prediction error than the baseline network (the black dotted line) after only 2 samples while this number is 6 for MC-Dropout. Moreover, MC-Dropconnect achieves an error less than one standard deviation away from its best performance (at T = 90) after only 18 samples, while this number is 54 for MC-Dropout (with its best performance at T = 94).

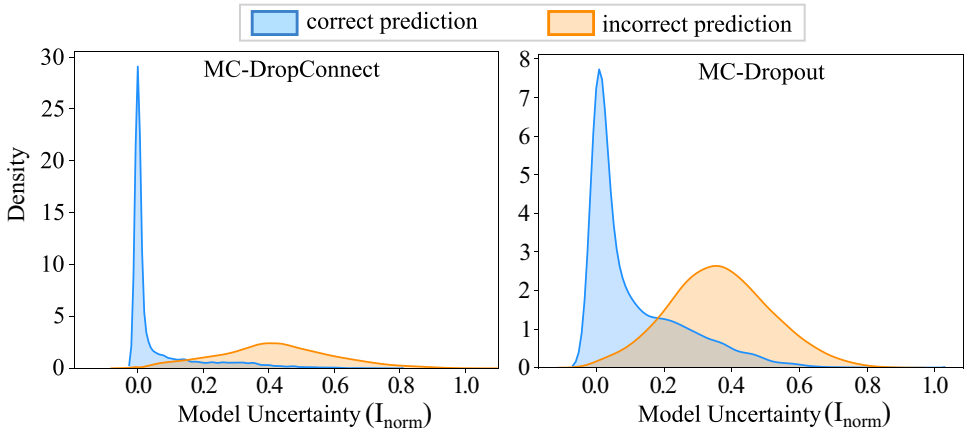

Figure 4. Illustrating the distribution of model uncertainty values for the CIFAR-10 test samples. Distributions are plotted separately for correct and incorrect predictions and for both MC-DropConnect (Left) and MC-Dropout (Right).

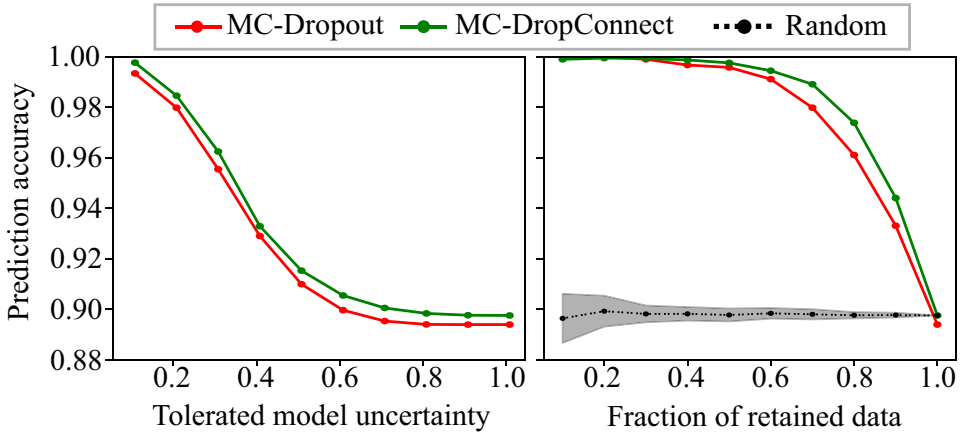

Figure 5. Enhanced prediction accuracy achieved via rejecting the highly uncertain samples. The prediction accuracy is computed over the test samples of the CIFAR-10 dataset and depicted as a function of the tolerated amount of model uncertainty (Left), and retained data size. The black curve in the right panel illustrates the effect of randomly rejecting the same number of samples. It is plotted as mean (±std) over 20 samplings. This shows that uncertainty is an effective measure of prediction accuracy.

Semantic segmentation. Here, we perform similar experiments to assess the performance of MC-DropConnect approximation of the BNNs and compare it with the benchmark MC-Dropout. The segmentation prediction performance is quantified using the pixel accuracy, mean accuracy and mean IOU metrics defined in[43]. Details of the data sets and network architectures used in each of the experiments are explained below briefly. Note that in all the experiments, dropout and dropconnect layers are placed in the same part of the network and with the same rate of $p = 0.5$.

CamVid with SegNet. CamVid[44] is a road scene understanding data set which contains 367, 100, and 233 training, validation, and test images respectively, with 12 classes. Images are size 360×480 and include both bright and dark scenes. We chose SegNet as the network architecture to be used for the semantic segmentation task to make the results of our approach to those of[21].

CityScapes with ENet. CityScapes[45] is one of the most popular data sets for the urban scene understanding with 5000, 500, and 1525 images for training, validation, and test. Images are of size 2048×1024 collected in 50 different cities and contains 20 different classes. Due to the large size of the images and more number of classes, we chose ENet[46] which is a more powerful network that requires fewer flops and parameters. The spatial dropout layers used in this framework are replaced with the regular dropout and dropconnect layers for our purpose.

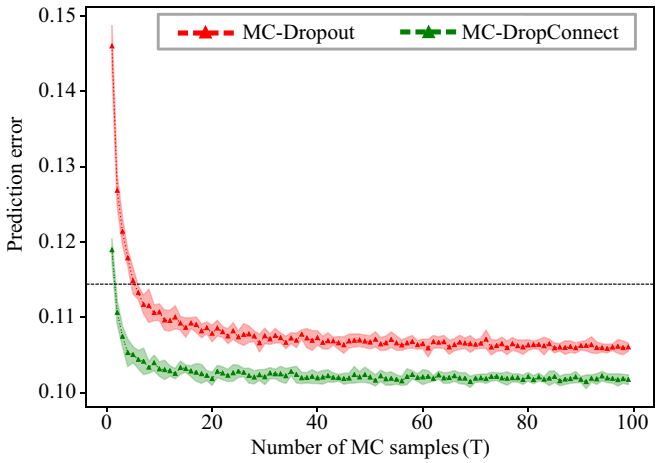

Figure 6. Test error of the FCNet on CIFAR-10 for different numbers of forward-passes in MC-Dropout and MC-DropConnect, averaged with 10 repetitions. The shaded area around each curve shows one standard deviation. The black dotted line shows the test error for the same neural network with no sampling.

3D CT-Organ with VNet. Since uncertainty estimates can play a crucial role in the medical diagnostics field, we also tested our model uncertainty estimation approach in the semantic segmentation of the body organs in abdominal 3D CT scans. The CT-Organ dataset includes 226 unique CT scans captured by General Electric and Siemens scanners at a single hospital. The study was approved by the Institutional Review Board (IRB) at the University of Texas MD Anderson Cancer Center. Informed consent requirement was waived by IRB as only deidentified data was used. The scans are down-sampled to 512×512 pixels and contain between 186 to 730 slices (mean=420, std=95). We used the volumetric CT scans from 180 patients for training and the rest are used for testing the models. We used V-Net[47] which is one of the most commonly used architectures for the segmentation of the volumetric medical images. The data include six classes including background, liver, spleen, kidney, bone, and vessel.

Qualitative observations. Figure 7 shows example segmentation and model uncertainty results from the various Bayesian frameworks on different datasets. This figure also compares the qualitative performance of MC-DropConnect with that of MC-Dropout. The correctness and confidence map highlights the misclassified and uncertain pixels respectively. Our observations show that MC-Dropconnect produces high-quality uncertainty estimation maps outperforming MC-Dropout, i.e. displays higher model uncertainty when models make wrong predictions.

We generally observe that higher uncertainty values are associated with three main scenarios. First, at the boundaries of the object classes (capturing the ambiguity in labels transition). Second, we observe a strong relationship between the frequency at which a class label appears and the model uncertainty. Models generally have significantly higher uncertainty for the rare class labels (the ones that are less frequent in the data; such as pole and sign symbol classes in CamVid). Conversely, models are more confident about class labels that are more prevalent in the data sets. Third, models are less confident in their prediction for objects that are visually difficult or ambiguous to the model. For example, (bicyclist, pedestrian) classes in CamVid and (car, truck) classes in CityScapes are visually similar which makes it difficult for the model to make a correct prediction, thus outputting higher uncertainty values.

Quantitative observations. We report the semantic segmentation results in Table 2 and Fig. 8. We find that MC-DropConnect generally improves the accuracy of the predicted segmentation masks for all three model-data set pairs.

Similar to what is done in the classification task, we computed the segmentation accuracies for varying levels of model confidence. The results are provided in Table 3. For all three data set-model pairs, we observed very high levels of accuracy for the 90th percentile confidence. This indicates that the proposed method results in the model uncertainty estimate which is an effective measure of confidence in the prediction.

Conclusion

We have presented MC-DropConnect as a mathematically grounded and computationally tractable approximate inference in Bayesian neural networks. This framework outputs a measure of model uncertainty with no additional computational cost; i.e. by extracting the information from the existing models that have been thrown away so far. We also developed new metrics to evaluate the uncertainty estimation of the models in all ML tasks, such as regression, classification, semantic segmentation, etc. We created the probabilistic variants of some of the most famous frameworks (in both classification and semantic segmentation tasks) using MC-DropConnect. Then we exploited the proposed metrics to evaluate and compare the uncertainty estimation performance of various models. Empirically, we observed that the MC-DropConnect improves the prediction accuracy, and

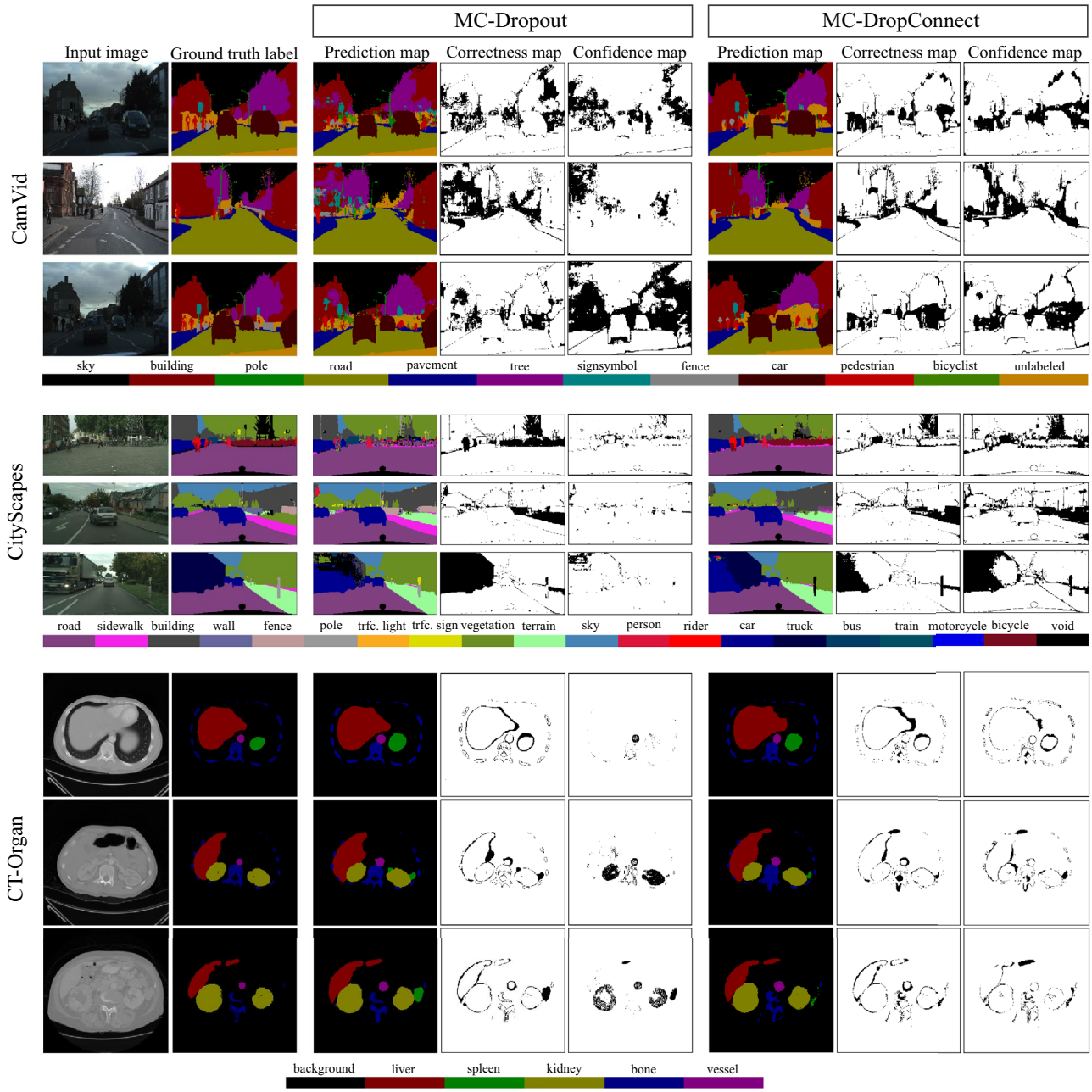

Figure 7. Qualitative results for semantic segmentation and uncertainty estimates on CamVid, CityScapes, and CT-Organ datasets. Each row depicts a single sample and includes the input image with ground truth, prediction, correctness, and confidence (using the mutual information metric) maps for both MC-Dropout and MC-DropConnect. Correctness map is the binary map that shows the correct and incorrect predictions. Confidence map is the thresholded map of uncertainty values computed over all classes. In all cases, the threshold is set manually to the one that achieves the highest UA. Correct and certain regions are respectively shown in white color in the correctness and confidence maps.

yields a precise estimation of the model confidence to its prediction. Analysis of the output uncertainty estimate via the proposed metrics shows that the model uncertainty estimates serve as an additive piece of information which can assist users in the decision-making process. We additionally recommend inserting the Dropconnect layers into non-regularized pre-trained networks and fine-tuning them in order to properly perform inference and uncertainty estimation at test time.

Future research includes the study of how imposing the Dropconnect (and with different drop probabilities) affects the trained convolutional kernels. While our method employs a fixed rate randomized weight dropping

Data (Model)	Uncertainty Estimation Method	Prediction Performance (%)			Uncertainty metrics AUC (%)		
		Pixel accuracy	Mean accuracy	Mean IOU	R_{iu}	R_{cc}	UA
CamVid (SegNet)	None	79.46	65.03	46.31	–	–	–
	MC-Dropout	80.99	65.46	47.31	17.23	82.48	80.18
	MC-DropConnect	**82.92**	**67.47**	**49.53**	**21.63**	**86.54**	**82.78**
CityScapes (ENet)	None	87.50	55.30	44.08	–	–	–
	MC-Dropout	87.38	56.35	44.11	6.12	88.67	84.89
	MC-DropConnect	**88.87**	**63.83**	**50.25**	**9.61**	**90.33**	**85.57**
CT-Organ (VNet)	None	95.19	96.44	65.49	–	–	–
	MC-Dropout	94.11	**97.73**	67.07	**10.81**	86.41	91.51
	MC-DropConnect	**97.90**	97.71	**72.77**	6.69	**87.03**	**92.59**

Table 2. Quantitative prediction and uncertainty estimation performance of the various frameworks on the CamVid, CityScapes, and CT-Organ datasets. Our quantitative analyses support the superior performance of the MC-DropConnect in terms of both segmentation accuracy and uncertainty estimation quality. The models with the best performances are shown in bold.

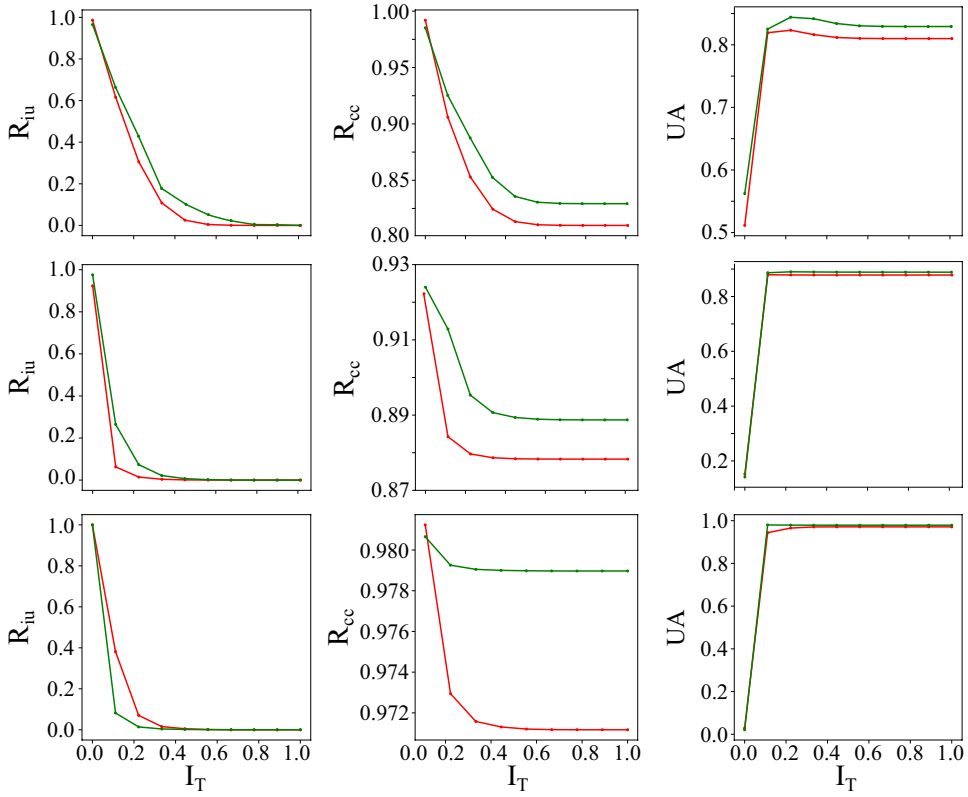

Figure 8. Illustrating the quantitative uncertainty estimation performance for the semantic segmentation task using the proposed evaluation metrics. Note that when varying the uncertainty threshold, our proposed MC-DropConnect approximated BNN (shown in green) generally performs better than MC-Dropout (shown in red) for CamVid (Top) and CityScapes (Middle), and CT-Organ (Bottom) datasets.

mechanism, it would be interesting to investigate a learnable weight dropping rate (similarly to Boluki et al.[24]) as a more flexible alternative. While we have effectively validated this method in classification and segmentation tasks, future works should investigate the feasibility of MC-Dropconnect in regression tasks. Leveraging the uncertainty in the training process to enrich the model's knowledge of the data domain is another interesting research direction that should be investigated.

Code availability

All scripts related to this work can be accessed without restriction at https://github.com/hula-ai/mc_dropconnect.

Confidence percentile	Pixel-wise classification accuracy		
	CamVid	CityScapes	CT-Organ
0	82.92	88.87	97.90
10	87.45	90.59	99.81
50	97.83	92.13	99.97
90	93.68	99.32	99.99

Table 3. Pixel-wise accuracy of the Bayesian frameworks as a function of confidence for the 0th percentile (all pixels) through to the 90th percentile (10% most certain pixels). This shows that the estimated model uncertainty is an effective measure of prediction accuracy.

Received: 2 June 2020; Accepted: 17 February 2021
Published online: 09 March 2021

References

1. Krizhevsky, A., Sutskever, I. & Hinton, G. E. Imagenet classification with deep convolutional neural networks. *Adv. Neural Inf. Process. Syst.* **25**, 1097–1105 (2012).
2. He, K., Gkioxari, G., Dollár, P. & Girshick, R. Mask r-cnn. In *Proceedings of the IEEE International Conference on Computer Vision* 2961–2969 (2017).
3. Mnih, V. *et al.* Playing atari with deep reinforcement learning. *arXiv preprint* arXiv:1312.5602 (2013).
4. Mobiny, A., Yuan, P., Cicalese, P. A. & Van Nguyen, H. Decaps: Detail-oriented capsule networks. In *International Conference on Medical Image Computing and Computer-Assisted Intervention* 148–158 (Springer, 2020).
5. Anjos, O. *et al.* Neural networks applied to discriminate botanical origin of honeys. *Food Chem.* **175**, 128–136 (2015).
6. Mobiny, A. & Van Nguyen, H. Fast capsnet for lung cancer screening. In *International Conference on Medical Image Computing and Computer-Assisted Intervention* 741–749 (Springer, 2018).
7. Cicalese, P. A. *et al.* Kidney level lupus nephritis classification using uncertainty guided Bayesian convolutional neural networks. *IEEE J. Biomed. Health Inform.* **25**, 315–324 (2020).
8. Mobiny, A. *et al.* Memory-augmented capsule network for adaptable lung nodule classification. *IEEE Trans. Med. Imaging* (2021).
9. Gal, Y. *Uncertainty in Deep Learning* (University of Cambridge, 2016).
10. Der Kiureghian, A. & Ditlevsen, O. Aleatory or epistemic? Does it matter?. *Struct. Saf.* **31**, 105–112 (2009).
11. Mobiny, A., Singh, A. & Van Nguyen, H. Risk-aware machine learning classifier for skin lesion diagnosis. *J. Clin. Med.* **8**, 1241 (2019).
12. Neal, R. M. *Bayesian Learning for Neural Networks* Vol. 118 (Springer, 2012).
13. MacKay, D. J. A practical Bayesian framework for backpropagation networks. *Neural Comput.* **4**, 448–472 (1992).
14. Blei, D. M., Kucukelbir, A. & McAuliffe, J. D. Variational inference: A review for statisticians. *J. Am. Stat. Assoc.* **112**, 859–877 (2017).
15. Graves, A. Practical variational inference for neural networks. *Advances in neural information processing systems* **2348–2356**, (2011).
16. Neal, R. M. Bayesian learning via stochastic dynamics. In *Advances in Neural Information Processing Systems* 475–482 (1993).
17. Gal, Y. & Ghahramani, Z. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In *International Conference on Machine Learning* 1050–1059 (2016).
18. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. & Salakhutdinov, R. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* **15**, 1929–1958 (2014).
19. Damianou, A. & Lawrence, N. Deep gaussian processes. In *Artificial Intelligence and Statistics* 207–215 (2013).
20. Gal, Y. & Ghahramani, Z. Bayesian convolutional neural networks with Bernoulli approximate variational inference. *arXiv preprint* arXiv:1506.02158 (2015).
21. Kendall, A., Badrinarayanan, V. & Cipolla, R. Bayesian segnet: Model uncertainty in deep convolutional encoder-decoder architectures for scene understanding. *arXiv preprint* arXiv:1511.02680 (2015).
22. Kingma, D. P., Salimans, T. & Welling, M. Variational dropout and the local reparameterization trick. In *Advances in Neural Information Processing Systems* 2575–2583 (2015).
23. Gal, Y., Hron, J. & Kendall, A. Concrete dropout. In *Advances in Neural Information Processing Systems* 3581–3590 (2017).
24. Boluki, S., Ardywibowo, R., Dadaneh, S. Z., Zhou, M. & Qian, X. Learnable bernoulli dropout for Bayesian deep learning. *arXiv preprint* arXiv:2002.05155 (2020).
25. Louizos, C. & Welling, M. Multiplicative normalizing flows for variational Bayesian neural networks. In *Proceedings of the 34th International Conference on Machine Learning* Vol. 70, 2218–2227 (JMLR. org, 2017).
26. Lakshminarayanan, B., Pritzel, A. & Blundell, C. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in Neural Information Processing Systems* 6402–6413 (2017).
27. DeVries, T. & Taylor, G. W. Learning confidence for out-of-distribution detection in neural networks. *arXiv preprint* arXiv:1802.04865 (2018).
28. Kendall, A. & Gal, Y. What uncertainties do we need in bayesian deep learning for computer vision? In *Advances in Neural Information Processing Systems* 5574–5584 (2017).
29. Kohl, S. *et al.* A probabilistic u-net for segmentation of ambiguous images. In *Advances in Neural Information Processing Systems* 6965–6975 (2018).
30. Guzman-Rivera, A., Batra, D. & Kohli, P. Multiple choice learning: Learning to produce multiple structured outputs. In *Advances in Neural Information Processing Systems* 1799–1807 (2012).
31. Lee, S., Purushwalkam, S., Cogswell, M., Crandall, D. & Batra, D. Why m heads are better than one: Training a diverse ensemble of deep networks. *arXiv preprint* arXiv:1511.06314 (2015).
32. Rupprecht, C. *et al.* Learning in an uncertain world: Representing ambiguity through multiple hypotheses. In *Proceedings of the IEEE International Conference on Computer Vision* 3591–3600 (2017).
33. MacKay, D. J. Probable networks and plausible predictions'a review of practical Bayesian methods for supervised neural networks. *Netw. Comput. Neural Syst.* **6**, 469–505 (1995).
34. Blundell, C., Cornebise, J., Kavukcuoglu, K. & Wierstra, D. Weight uncertainty in neural networks. *arXiv preprint* arXiv:1505.05424 (2015).

35. Gal, Y. & Ghahramani, Z. Dropout as a Bayesian approximation: Insights and applications. In *Deep Learning Workshop, ICML* Vol. 1, 2 (2015).
36. Wan, L., Zeiler, M., Zhang, S., Le Cun, Y. & Fergus, R. Regularization of neural networks using dropconnect. In *International Conference on Machine Learning* 1058–1066 (2013).
37. Smith, L. & Gal, Y. Understanding measures of uncertainty for adversarial example detection. *arXiv preprint* arXiv:1803.08533 (2018).
38. Abadi, M. *et al.* Tensorflow: A system for large-scale machine learning. In *12th{USENIX}Symposium on Operating Systems Design and Implementation({OSDI}16)* 265–283 (2016).
39. LeCun, Y. *et al.* Gradient-based learning applied to document recognition. *Proc. IEEE* **86**, 2278–2324 (1998).
40. LeCun, Y. & Cortes, C. MNIST handwritten digit database. http://yann.lecun.com/exdb/mnist/. Accessed 2020-11-30.
41. Krizhevsky, A., Hinton, G. *et al.* Learning multiple layers of features from tiny images (2009).
42. Hendrycks, D. & Gimpel, K. A baseline for detecting misclassified and out-of-distribution examples in neural networks. *arXiv preprint* arXiv:1610.02136 (2016).
43. Long, J., Shelhamer, E. & Darrell, T. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* 3431–3440 (2015).
44. Brostow, G. J., Fauqueur, J. & Cipolla, R. Semantic object classes in video: A high-definition ground truth database. *Pattern Recogn. Lett.* **30**, 88–97 (2009).
45. Cordts, M. *et al.* The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* 3213–3223 (2016).
46. Paszke, A., Chaurasia, A., Kim, S. & Culurciello, E. Enet: A deep neural network architecture for real-time semantic segmentation. *arXiv preprint* arXiv:1606.02147 (2016).
47. Milletari, F., Navab, N. & Ahmadi, S.-A. V-net: Fully convolutional neural networks for volumetric medical image segmentation. In *2016 Fourth International Conference on 3D Vision (3DV)* 565–571 (IEEE, 2016).

Author contributions

A.M. conducted the experiments was responsible for the development and execution of the methodology. P.Y. prepared the data and assisted in writing the introduction and related works sections. H.V.N., S.K.M., C.C.W., and N.G. conceptualized and supervised the study, and was responsible for the acquisition of financial support for the project. A.M. wrote the first draft; all authors reviewed, edited and provided final approval of the manuscript.

Competing interests

The authors declare no competing interests.

Additional information

Correspondence and requests for materials should be addressed to A.M.

Reprints and permissions information is available at www.nature.com/reprints.

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.