A Review of and Guidelines for Conveying Uncertainty in Medical Imaging Classification

Tin Oreskovic and Pranjal Bajaj

December 19th, 2018

I. Introduction

Advances in applying computer vision and, specifically, deep learning for classification have achieved near-human and human-level accuracy in certain sub-domains of medical diagnostics. Some of the most recent results, for instance in predicting the presence of breast cancer and diabetic retinopathy, have even achieved better-than-human performance as measured by what are in the medical field traditional measures of sensitivity (the fraction of correctly identified positive instances among all positive instances) and specificity (the fraction of correctly identified negative instances among all negative instances).

Despite this progress, neither the rate at which **(a)** using computer vision in medical diagnostics practice has been approved by the relevant authorities nor **(b)** the interest in adopting such methods by the medical community in developed countries, have matched the technical developments. Currently, within the USA, the only FDA-approved use of computer vision in medical diagnostics is within the mentioned detection of diabetic retinopathy, a disease of the eye commonly occurring among diabetes patients (this is a major success for computer vision in the medical domain).[1]

The driving forces underlying these two facts are complex, but some of the commonly expressed reasons for the slow increase in the interest to adopt deep-learning fueled progress in computer vision in medical practice is the relative lack of prominent methods to convey uncertainty in its output for each prediction/decision, which is a large obstacle to informing treatment decisions made by medical practitioners, even assuming satisfactory performance (Litjens et al., 2017). This article will summarize some of the known methods used in conveying uncertainty involved in deep learning generated classifications to medical practitioners, with a focus on ease of implementation, both in terms of **(i)** incorporating good practices into training the model so as to eventually allow better communication of its output; and **(ii)** given the model's output and

[1] FDA News Release, April 1, 2018

the accompanying uncertainty, how it may effectively be conveyed to medical practitioners. We will try to adjust the length of each subsection describing a method to correspond to our assessment of its usefulness and ease of implementation and thus importance for a non-expert practitioner. The last section of the review functions as succinct but almost self-sufficient summary and a set of guidelines to expressing uncertainty in neural networks for classification applications within medicine.

II. Training the Model in a Manner that Allows Conveying Uncertainty

As discussed in the previous section, obtaining uncertainties regarding predictions of deep neural networks is instrumental for automated disease detection to become a reality in the medical domain. This section discusses the various techniques that can be incorporated into training deep learning models to allow practitioners to gauge uncertainty estimates regarding the model's predictions. The discussion begins with a definition of uncertainty as it may arise in image classification problems. The "status quo" case of using solely point estimates is discussed; however, it is apparent that Bayesian approaches are more appropriate for modelling uncertainty as part of the training process in practical scenarios. Hence, a majority of this section discusses the theoretical case for Bayesian approaches and also draws comparisons with alternative approaches. The section ends with advice on how these methods can be used in practice.

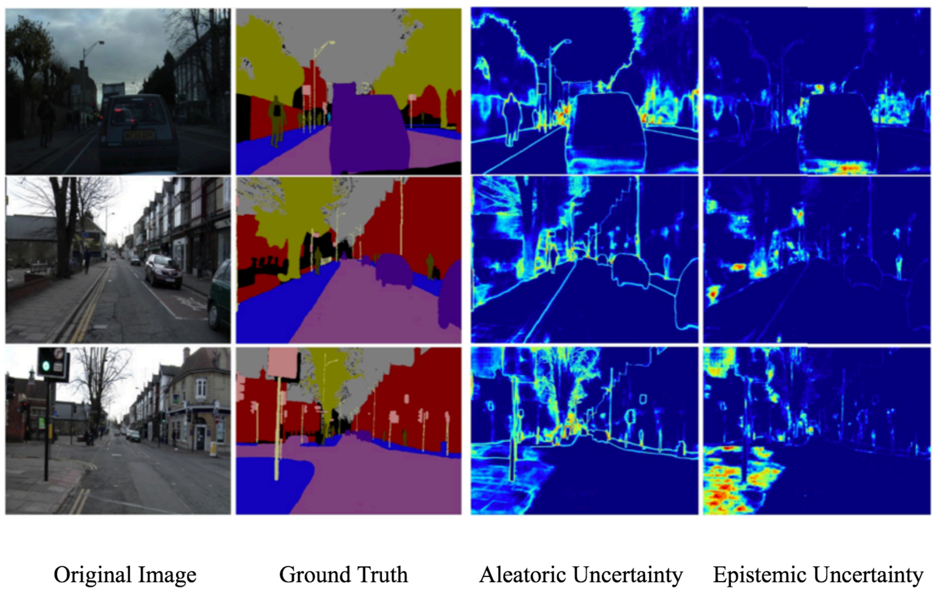
Two Notions of Uncertainty in Image Classification

Uncertainty in image classification can be divided into two broad categories:

1. Aleatoric uncertainty captures noise inherent in the observations (Kendall and Gal, 2017). However, it can be explained away with increasing precision by observing all explanatory variables. Since this can be easily handled by modifying the loss function in a standard deep neural network, this section will focus mainly on addressing the second type of uncertainty that arises in image classification. The term is derived from "alea" – latin for "die" – implying a relation to intrinsically "chancy" phenomena.

2. Epistemic uncertainty accounts for uncertainty in the model i.e. uncertainty which can be explained away given enough data. It is a lot more difficult to model; however, it can be addressed by applying Bayesian inference (Kendall and Gal, 2017). "Epistemic" refers to knowledge or the degree of (according to a naive definition) justified true belief; in this context, it is meant to suggest that "epistemic" uncertainty is a function of our ignorance about a phenomenon, rather than its intrinsically "aleatoric" nature.

Figure 1: Aleatoric and Epistemic Uncertainty (Kendall and Gal, 2017)



Uncertainty and Point Estimates

Given a deep learning model h, in the context of computer vision applied to medical diagnostics, typically a convolutional neural network, the model's standard output usually consists of values equal in number, k, to that of the mutually exclusive classes for the diagnosis (positive/negative; severe, non-severe, healthy). The mapping from the input, i.e. the images X, via the model's intermediate layers, has as its final element a "fully connected" layer whose output is passed on to a non-linear activation function - in the case of binary classification this is usually a sigmoid function, while for multiclass classification the generalization of the former is the softmax function. The final output of the activation function and thus of the model is a set of k values $\hat{p}_i$ corresponding to the k classes, each $\in [0, 1]$, and $\sum_i \hat{p}_i = 1$. This is a mapping of a non-normalized output to a probability distribution over the predicted output classes. The range and nature of the model's output lends itself to probabilistic interpretations, and the class $\hat{Y}$ whose corresponding output value is maximal is the predicted class for the example (image) at hand. For instance, in an object detection task, assuming there are $k = 3$ classes, say "car," "airplane," and "train." Softmax output values $\hat{p}_i$ of $[0.2, 0.2, 0.6]$ for an image in the training set corresponding to a probability score/estimate of the image belonging to each class.[2] Here, since the probability score of the *airplane* class is the highest, which means that this image

[2]Softmax cannot simply be understood as actual probabilities and as such are referred to as probability scores; more on this is in section III

will be labelled as an airplane. For a model that performs poorly, one can expect most softmax scores to be close to random, i.e. $0.3, 0.34 0.36$; however, for a model that performs well on a reasonable task that it is trained for, one can often expect softmax scores such as, for instance, $0.1, 0.05, 0.8$, i.e. indicating greater "confidence" in the predictions. It has to be noted that the converse does not hold: as will be more elaborately discussed in the following section, a high score for a particular class does not indicate that the model performs well. As more data are added for training and the model is increasingly well tuned, it performs better, with softmax scores usually increasingly more confident about a class prediction. This is what treating the aleatoric uncertainty in the model amounts to.

It might be tempting to treat the softmax scores as the only and the sufficient expression of uncertainty about predictions; however, it is important to note that these softmax scores correspond to a single set of network parameters only and hence can only explain, under certain assumption, one notion of uncertainty or confidence, but they cannot explain the uncertainty over the point estimates themselves – an important notion of uncertainty that many practitioners want to address, and for good reason (Leibig et al., 2017). Understanding the point estimates as indications of confidence and the associated challenges will be further discussed in section III, after obtaining and expressing estimates of uncertainty around the point estimates is elaborated on below.

Moving away from Point Estimates

Traditionally, the performance of deep neural networks (DNNs) is gauged using point estimates, which entails minimizing the cross-entropy between the distributions of the true class labels and the predicted softmax scores, i.e. by learning a maximum likelihood estimate. However, these models also have a probabilistic interpretation. This involves first assuming that the parameters of the model come from a particular distribution, i.e. a prior distribution over the parameters. As the model becomes exposed to more training data, these prior beliefs get updated to obtain a distribution over the parameters known as a posterior distribution. Mathematically, this is proportional to the product of the prior distribution and the likelihood observed from the data.

$$p(\theta|y) \propto p(\theta)p(y|\theta)$$

It is worth noting here that a probability distribution over the parameter has been obtained, rather than a point estimate. Further, the prior distribution over the parameter can be seen as regularizing the model by constraining the parameter space. The expectation of this posterior distribution, the maximum a posteriori (MAP) probability estimate is found to lie somewhere between the prior distribution and the training data likelihood (Gelman et al., 2014). Hence, this MAP estimate can be seen a compromise between the prior distribution and the data likelihood. As more data are obtained, the posterior mean i.e. MAP shifts further away from the prior and towards the data likelihood and the importance of the prior distribution weakens. At this stage, the mode of the posterior distribution is equal to the Maximum Likelihood Estimate (Gelman et al., 2014).

The posterior predictive distribution over an unknown label y conditional on a test data point X and the posterior distribution over parameters of the model is given by:

$$p(y^*|X, y, x^*) = \int p(y^*|\theta)p(\theta|X, y, x^*)d\theta$$

The predictive posterior distribution defines a distribution of softmax scores rather than a single softmax score. Uncertainty in the model's predictions corresponds to the width of predictive posterior, which can be defined by its variance. Determining the moments of this predictive posterior is intractable in practice and a common approach to do this is by using variational inference (Leibig et al., 2017). This can be achieved by sequentially drawing the parameters from the joint posterior distribution of the parameters, simulating the data point from this joint distribution and then taking its first and second moments to arrive at predictions and their corresponding uncertainties (Gelman et al., 2014). In the case of DNNs, there are many approaches to this; however, Bayesian Neural Networks have recently gained a lot of traction in this area (Osband, 2016), and have even been implmented in the medical domain.

Bayesian Neural Networks

In the case of DNNs, it was recently shown that a multi-layer network with dropout after every weight layer is mathematically equivalent to an approximation to the predictive posterior in a Bayesian sense (Gal, Y. Ghahramani, 2015; Leibig et al., 2017). This technique is known as *Monte Carlo (MC) Dropout*. Each *dropout* layer can be viewed as a *prior* from a Bernoulli density function for the weights in each weight layer. The dropout rate, then, can be seen as the parameter of the Bernoulli density (Mullachery, 2017 and Leibig et al., 2017). This is one way to define a Bayesian neural network. The result holds for any number of layers and arbitrary non-linearities (Leibig et al., 2017). The idea easily extends to convolutional layers as well (ibid). Hence, it is possible to treat this Neural Network as a posterior distribution and then draw Monte Carlo samples from it to approximate the predictive posterior distribution, as discussed above. The next step involves computing the predictive mean and the predictive standard deviation, i.e. the first two moments of the predictive posterior distribution (below).

$$\mu_{pred} \approx \hat{\mu}_{pred} = \frac{1}{T} \sum_{t=1}^{T} p(y^*|\mathbf{x}^*, \theta(\hat{\omega}))$$

$$\sigma_{pred} \approx \hat{\sigma}_{pred} = \frac{1}{T-1} \sqrt{\sum_{t=1}^{T} (p(y^*|\mathbf{x}^*, \theta(\hat{\omega}_t)) - \hat{\mu}_{pred})^2}$$

T denotes the number of testing samples and comes from the Bernoulli prior, which is equivalent to dropout during training.

In terms of computation, for a small T, MC samples can be performed in parallel and the computation is fast. Leibeg et al. (2017) find that serial computation involves 200ms per image for $T = 100$.

The next step involves using the posterior predictive sigma-square to rank images based on uncertainty estimates and, with these rankings, determining which set of images require further inspection (Leiber et al., 2017). The thresholds for uncertainty can be set by the doctors in accordance with their preferences and constraints.

Implementation Advice and Resources

Implementation of Bayesian Neural Networks with MC Dropout for image classification is quite straightforward: for every Conv2D layer, add a dropout layer with a fixed dropout rate. The next step involves sequentially drawing the parameters from the joint posterior distribution (approximated by the CNN with dropout priors to each layer) of the parameters, simulating the data points from this joint distribution and then taking its first and second moments to arrive at predictions and their corresponding uncertainties. The last two steps aren't easy to implement and there are several libraries in Python that support the implementation of Bayesian Neural Networks, including Edward, which supports TensorFlow objects as well. Similarly, PyMC supports Theano and will be supporting TensorFlow in its upcoming version.

A Brief Discussion of BNNs and Other Methods

In general, Bayesian neural networks (BNNs) are especially useful in analyzing uncertainty in cases when there is a dearth of training data, which is often the case in practice (Leiber et al., 2017). An alternate approach to MC Dropout is using Gaussian Processes for prediction; they are theoretically rich but scale badly with both the dimensionality of the feature space and the size of the dataset. Leiber et al. (2017), find that the standard Gaussian Processes model exhibits a runtime complexity of $\mathcal{O}(N^3)$ and memory complexity of $\mathcal{O}(N^2)$.

Despite all the advantages MC Dropout may present, current literature, namely Osband (2016), highlights that such approximations to uncertainty in fact models risk, given a fixed model, rather than uncertainty itself. Osband identifies risk as the inherent stochasticity in a model (corresponding to aleatoric uncertainty) and "uncertainty as the confusion over which model parameters apply," i.e. presumably corresponding to epistemic uncertainty. "For example, a coin may have a fixed $p = 0.5$ of heads and so the outcome of any single flip holds some risk; a learning agent may also be uncertain of p." The "dropout posterior" that is obtained by applying MC Dropout, writes Osband, can have an "arbitrarily small or large variance depending on the interaction between the dropout rate p and the model size K." Furthermore, he notes the distribution is not guaranteed to concentrate with more data. Others have remarked that while the findings on MC Dropout as an approximation are almost beyond doubt correct, there may be unrecognized assumptions to the results, which should be explicated and the methods should be used with this in mind.[3] The solution presented in Osband (2016) is to bootstrap samples and fit estimators on them rather than using a fixed dropout rate for each Convolutional layer.

Notwithstanding the unsettled exact nature of the parameter distributions obtained through MC Dropout and the still ongoing academic discussions, there are two strong additional reasons we have chosen to focus most of this section to Bayesian approaches and within these to MC Dropout, beyond its also notable theoretical strengths. The first of these is the fact that the approach to gain estimates of uncertainty around point estimates that MC Dropout enables are simpler to implement compared to the competing methods. The second reason, crucially, is that this approach has recently been successfully utilized in the medical domain of disease detection. Leibig, Allken, Ayhan, Berens Wahl (2017) employ the method to evaluate uncertainty measure in diagnosing diabetic retinopathy, finding that it "captures uncertainty better than straightforward alternatives." What is more, the authors also find significant spillover benefits to diagnostics performance across several datasets, surpassing "85% sensitivity and 80% specificity as recommended by the NHS when referring 020% of the most uncertain decisions for further inspection." The very encouraging improvements in performance are themselves a pragmatic indication that MC Dropout is much more appropriate at accounting for uncertainty than the standard methods (assuming no adjustments/calibration is performed). It is to the best of our knowledge a singular example of a direct application of findings from this emergent uncertainty literature to the medical domain.

Given the strong theoretical grounding and computational advantages of Bayesian approaches, a further step in this direction involves using Bayesian Estimation in backpropagation itself, in order to optimize uncertainty in the training process. A new, efficient, principled and backpropagation-compatible algorithm

[3] Note that this article's relatively pessimistic outlook is in part a function of not considering parts of the literature that is referenced in this review, large parts of which have been published only very recently

called *Bayes by Backprop* is discussed in Blundell et al. (2017). A further challenge for Bayesian deep neural networks comes from the fact that the implied uncertainty intervals are often miscalibrated – a serious issue which we will leave for the discussion section at the end of this review, along with its tentative solution, although the practical first success of MC Dropout in the medical domain may provide an even more compelling reason to take the challenge as an opportunity for further improvement rather than a problem that jeopardizes the potential of the method in practice.

III. Conveying Uncertainty in a Standard Model's Output

There are several exciting recent developments in addressing issues concerning the communication of uncertainty in a model's output once it is already trained and the standard version of the output is available. Writing on these methods, often described as falling under the *(pre and) postprocessing* umbrella, requires a further discussion of the standard output of a deep (here convolutional) neural network. As mentioned in the previous section, the final output of an activation function and thus of a model is a set of k values $\hat{p}_i$ corresponding to the k classes, each $\in [0,1]$, and $\sum_i \hat{p}_i = 1$. The range and nature of the output lends itself readily to probabilistic interpretations, appropriately, the maximal among the sigmoid/softmax output values, $\hat{p}_{i\,\max}$, under certain assumptions corresponds to the model's confidence that the class it predicts the sample belongs to, $\hat{Y}$, matches the true class Y of the example. Conversely, then, the uncertainty in the classification decision can somewhat naively be interpreted as equal to $1 - \hat{p}_{i\,\max}$. This of course isn't merely because the values of sigmoid or softmax output are between 0 and 1 and sum up to 1; the overall structure of the model, along with the nature of the final layer's nonlinear activation functions together yield certain properties that to some extent support this interpretation.

Assuming the above-described understanding of what the standard output of a deep learning model is, beyond the possibilities explicated in the previous section, we will here synthesize a number of methods that, given a model: enable **(a)** an assessment of and improvement in how well the standard output conveys the the intrinsic uncertainty in the model's predictions; and **(b)** methods that allow additional expressions of uncertainty by providing a quantification of how "far away" a test-set sample is from the training distribution. Before proceeding, it should be noted that the MC Dropout method of obtaining distributions and thus intervals for the point estimate of the otherwise standard output, which is described in the previous section, of course also results in an expression of uncertainty in the model's output, but the substantive difference is that the departure from the traditional methods is in a theoretical sense in training a different type of a model (a Bayesian neural network), while the (preprocessing and) postprocessing methods to be discussed here take as given a standard model.

A: Assessments of and Improvements in how the Standard Output Conveys Uncertainty

A relatively high specificity and sensitivity (proportion of truly positive instances the model predicts as such as well as a high fraction of correctly identified negative instances among all negative instances in the test set) are both necessary conditions to adopt computer vision in medical diagnostics, along with the requirement that those relying on the technology be aware of these metrics. Still, high performance on these metrics is *not sufficient* to gain trust from the medical practitioners, satisfy the needs to understand each element of the diagnostics process at the level of an individual prediction, and to enable flagging for additional attention when confidence in the prediction is low. "In real-world decision making systems, classification networks must not only be accurate, but also should indicate when they are likely to be incorrect" (Guo et al., 2016). This is especially so in medical applications - as mentioned in the introduction, anything enabling a medical practitioner to gain a reliable sense of how confident is each individual prediction would contribute significantly to satisfying these conditions of trust and understanding (as well as delivering other benefits).

An obvious way to look, then, is at the maximal among the sigmoid/softmax output values, $\hat{p}_{i\,\max}$, since it can at face value be taken to correspond to the confidence in the model's prediction of the class that a sample belongs to, while conversely, the uncertainty in the classification decision can somewhat naively be interpreted as equal to $1 - p_{i\,\max}$. Beyond the fact that each of the output values p_i are between 0 and 1 and that they sum up to 1, under the assumption that cross-entropy[4] is used as the objective function/loss function to be minimized (binary cross-entropy for binary classification, categorical cross-entropy for multiclass classification), using the softmax function to transform the output of a network's final fully connected layer yields the posterior categorical distribution over the class labels. Taking this interpretation at face value, then, if the maximal $p_{i\,\max}$ outputted by the sigmoid/softmax function is 0.7, it can be taken not only as an indication that the class associated with the output is the best guess of where the example at hand belongs, but that 0.7 is an expression of the model's confidence in the prediction and $1 - 0.7 = 0.3$ of the epistemic uncertainty associated with the prediction. This, in turn, implies that of all the predictions whose softmax output was 0.7 made by the same model, about 70% should in fact be correct. If appropriate, it would be one significant sort of tangible interpretation of the model's output that could contribute to its trustworthiness in medical practice and further the understanding of the diagnostics process on the part of the model.

Unfortunately, according to Guo et al. in *On Calibration of Modern Neural Networks*, contemporary neural networks (2016-2018) are not well calibrated: in other words, taken as probability estimates, the

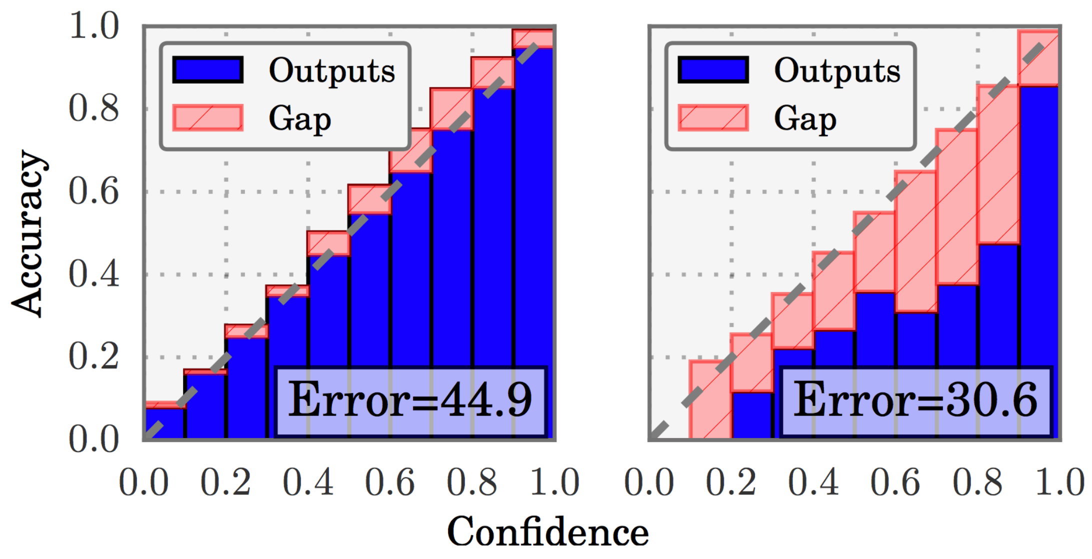
[4]
$$\mathcal{L} = \sum_{i=1}^{n} \log(\hat{\pi}(y_i|x_i))$$

output of neural networks is often not "representative of the true correctness likelihood." The result to be presented in figure 3 below holds for a variety of deep learning applications, including most relevantly for medical diagnostics applications, computer vision tasks. The idealized condition for a perfectly calibrated model is:

$$\mathcal{P}(\hat{Y} = Y | \hat{p}_{i\,\max} = p) = p, \quad \forall \in [0, 1]$$

That is, the probability that the predicted class $\hat{Y}_i$ for an example i matches the true class of the example, Y_i, given that the maximum output value of the sigmoid/softmax function of the model's final layer is $p_{i\,\max} = p$, is exactly equal in number to that maximum output value, and thus to p. Determining whether this holds, and how far off the calibration is, is impossible with a finite set of examples, so the authors review a number of existing useful empirical approximations capturing the essence of the above condition for calibrated models. The first of these are reliability diagrams[5], a visual representations of model calibration, plotting expected sample accuracy as a function of confidence. Below is an example:

Figure 2: Reliability Diagrams (Guo et al., 2017)



For a perfectly calibrated model, the identity function appears, with any divergence from the perfect diagonal an indication of miscalibration. Estimating expected accuracy from finite samples, the predictions

[5]these are also known as calibration plots or curves

$\hat{Y}$ of a model h are distributed into M bins of equal interval size. Setting B_m to consist of "all the indices of samples whose prediction confidence falls into the interval $I_m = (\frac{m-1}{M}, \frac{m}{M}]$", the average accuracy of B_m then is given by

$$acc(B_m) = \frac{1}{|B_m|} \sum_{i \in B_m} \mathbf{1}(\hat{Y}_i = Y_i).$$

The average accuracy of each set B_m turns out to be a consistent and unbiased estimator of

$$\mathcal{P}(\hat{Y} = Y | \hat{p}_{i\,\max} \in I_m).$$

The average confidence, shown on the other axis of a reliability diagram, is defined for each bin B_m as:

$$conf(B_m) = \frac{1}{|B_m|} \sum_{i \in B_m} \hat{p}_{i\,\max}.$$

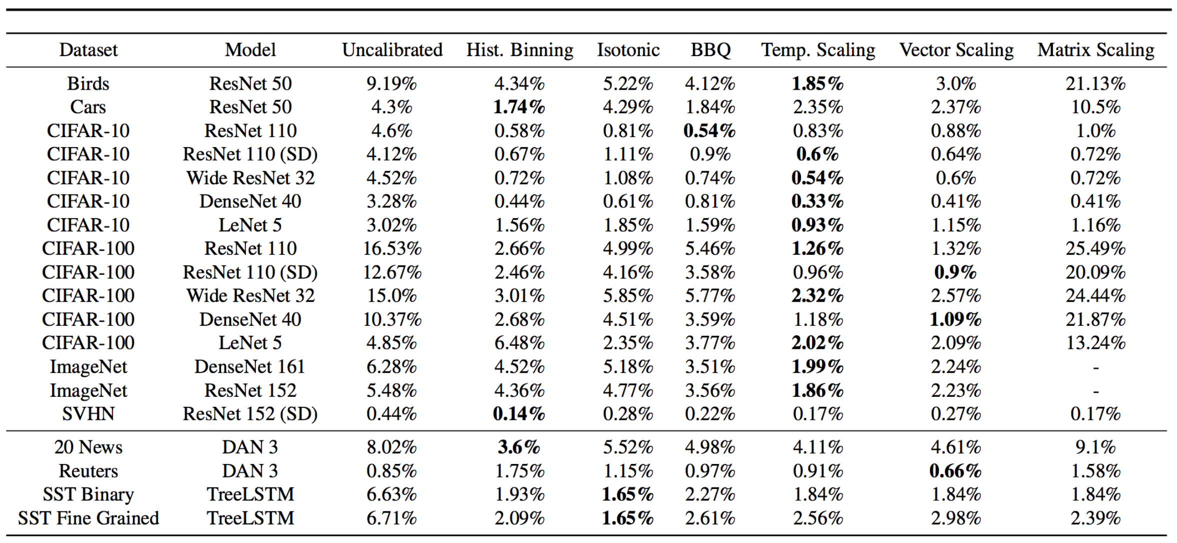
A scalar metric of calibration is the **expected calibration error**. In an intuitive sense, it conveys the gap between the bin accuracy and the confidence for a given bin B_m as defined above:

$$ECE = \sum_{m=1}^{M} \frac{|B_m|}{n} \mid acc(B_m) - conf(B_m) \mid .$$

The number of samples is represented by n in the above formula, which then gives the "weighted average of the bins' accuracy/confidence difference. [...] The difference between *acc* and *conf* for a given bin represents the calibration gap (red bars in reliability diagrams)." Finally, in high-risk applications, which likely fits the description of certain situations within the medical domain, one appropriate measure of a model's overall unreliability/epistemic uncertainty might be the **maximum calibration error**, which corresponds to the largest calibration gap across all bins. ECE is the primary metric that the authors use for their analyses.

The paper by Guo et al. presents several methods to calibrate binary and multiclass classification models, each meant to bring the function expressing the relation between accuracy and confidence for each bin closer to the perfectly diagonal identity function, and to reduce the ECE and MCE measures of (mis-)calibration. Among the results below are ECE measures for 6 image classification datasets, across several state-of-the-art models at the time of writing:

Figure 3: Results from Guo et al., 2017



Dataset	Model	Uncalibrated	Hist. Binning	Isotonic	BBQ	Temp. Scaling	Vector Scaling	Matrix Scaling
Birds	ResNet 50	9.19%	4.34%	5.22%	4.12%	**1.85%**	3.0%	21.13%
Cars	ResNet 50	4.3%	**1.74%**	4.29%	1.84%	2.35%	2.37%	10.5%
CIFAR-10	ResNet 110	4.6%	0.58%	0.81%	**0.54%**	0.83%	0.88%	1.0%
CIFAR-10	ResNet 110 (SD)	4.12%	0.67%	1.11%	0.9%	**0.6%**	0.64%	0.72%
CIFAR-10	Wide ResNet 32	4.52%	0.72%	1.08%	0.74%	**0.54%**	0.6%	0.72%
CIFAR-10	DenseNet 40	3.28%	0.44%	0.61%	0.81%	**0.33%**	0.41%	0.41%
CIFAR-10	LeNet 5	3.02%	1.56%	1.85%	1.59%	**0.93%**	1.15%	1.16%
CIFAR-100	ResNet 110	16.53%	2.66%	4.99%	5.46%	**1.26%**	1.32%	25.49%
CIFAR-100	ResNet 110 (SD)	12.67%	2.46%	4.16%	3.58%	0.96%	**0.9%**	20.09%
CIFAR-100	Wide ResNet 32	15.0%	3.01%	5.85%	5.77%	**2.32%**	2.57%	24.44%
CIFAR-100	DenseNet 40	10.37%	2.68%	4.51%	3.59%	1.18%	**1.09%**	21.87%
CIFAR-100	LeNet 5	4.85%	6.48%	2.35%	3.77%	**2.02%**	2.09%	13.24%
ImageNet	DenseNet 161	6.28%	4.52%	5.18%	3.51%	**1.99%**	2.24%	-
ImageNet	ResNet 152	5.48%	4.36%	4.77%	3.56%	**1.86%**	2.23%	-
SVHN	ResNet 152 (SD)	0.44%	**0.14%**	0.28%	0.22%	0.17%	0.27%	0.17%
20 News	DAN 3	8.02%	**3.6%**	5.52%	4.98%	4.11%	4.61%	9.1%
Reuters	DAN 3	0.85%	1.75%	1.15%	0.97%	0.91%	**0.66%**	1.58%
SST Binary	TreeLSTM	6.63%	1.93%	**1.65%**	2.27%	1.84%	1.84%	1.84%
SST Fine Grained	TreeLSTM	6.71%	2.09%	**1.65%**	2.61%	2.56%	2.98%	2.39%

Table 1. ECE (%) (with $M = 15$ bins) on standard vision and NLP datasets before calibration and with various calibration methods. The number following a model's name denotes the network depth.

The study arrived at the somewhat surprising but fortunate finding that across most image datasets, temperature scaling, one of the simplest surveyed calibration methods, was shown to be the most effective. For brevity's sake, we will here focus on explaining only this method for multiclass problems and isotonic regression as the one widely regarded as very effective for binary classification, while the reader can refer to the original article to find information on the remaining methods. In short, temperature scaling (an extension of Platt scaling to multiclass problems) involves optimizing a single parameter $T > 0$. Given the output of a model, call each of these values z_j (j ranging from 1 to k), before they are passed to the softmax function of the final layer the following transformation is applied, producing new confidence estimates, call them $\bar{p}_i$:

$$\hat{p}_i = \frac{e^{z_i/T}}{\sum_j e^{e^{z_j/T}}}.$$

The T parameter is learned from the validation-set examples and labels, with the negative log likelihood as the optimization metric, much like when the model itself is trained. During training, T is set to 1. In effect, the T parameter softens the softmax output, with the uncertainty in the final output increasing with T.

Since T does not change the rank of the maximal $\hat{p}_{i\,\max}$ when replacing it with $\bar{p}_{i\,\max}$, the class prediction $\hat{Y}$ remains the same and the accuracy of the model remains the same, but its calibration improves.

For binary classification problems, a nonparametric method known as isotonic regression is widely used and found to be effective (Niculescu-Mizil and Caruana, 2005). It involves dividing again all uncalibrated output predictions $\hat{p}_i$ into bins of equal interval size, and finding a function $f(\hat{p}_i)$ that is within each bin subject to minimizing the squared error loss, i.e. $\min \sum_i^n (f(\hat{p}_i))$, as well as to the additional constraint that the learned function be monotonically non-decreasing. Convenient Python implementations exist both for temperature scaling and for isotonic regression.

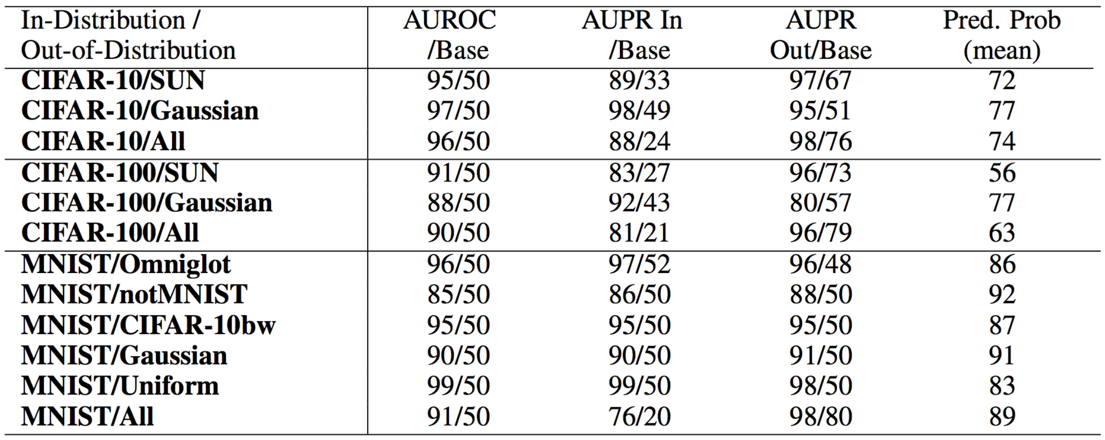
The authors also explore potential causes of miscalibration (which they find to vary between with ECE typically between 4% to 10%), though they mostly present associations, rather than probing into causal relations (or more appropriately, the theoretical reasons that could underlie the unfortunate miscalibration). Greater miscalibration is associated with wider and deeper networks, and those with more weight decay and batch normalization. Though this is not as directly pertinent to the subject of conveying uncertainty, it can provide information of the type of model that practitioners should be careful about before applying in medical tasks, calibrating them before relying on their confidence expressions – this happens to describe most contemporary neural networks.

B: Detecting Out of Distribution Examples (Images Far-Away from the Training Data)

For applications in the medical domain, having models that are both high-performing in terms of sensitivity and specificity as well as well-calibrated allows a medical practitioner to rely on the model's predictions knowing at once that the model is overall highly accurate and that the output probability associated with the predicted class is an expression of the model's confidence. This would bring the output of a deep learning model applied to imagining in the medical domain closer to satisfying the uncertainty-related needs of medical practitioners. The first sentence in this paragraph, however, is only correct under the significant assumption that the training and testing data distributions do not differ. When they are different, naturally, classifiers perform much worse, but, what is more pertinent to the topic of this review, the classifiers often "fail silently by providing high-confidence predictions" while being highly unreliable (Goodfellow et al., 2014).

Conveniently, in *A Baseline for Detecting Misclassified and Out-of-Distribution Examples in Neural Networks* (2016), Hendrycks and Gimpel show that examples distant from the distribution of the training examples typically have lower maximal softmax output scores $\hat{p}_{i\,\max}$, compared to within-distribution examples.

Figure 4: Out-of-Distribution Example Softmax Scores (Hendrycks and Kevin Gimpel, 2016)



In-Distribution / Out-of-Distribution	AUROC /Base	AUPR In /Base	AUPR Out/Base	Pred. Prob (mean)
CIFAR-10/SUN	95/50	89/33	97/67	72
CIFAR-10/Gaussian	97/50	98/49	95/51	77
CIFAR-10/All	96/50	88/24	98/76	74
CIFAR-100/SUN	91/50	83/27	96/73	56
CIFAR-100/Gaussian	88/50	92/43	80/57	77
CIFAR-100/All	90/50	81/21	96/79	63
MNIST/Omniglot	96/50	97/52	96/48	86
MNIST/notMNIST	85/50	86/50	88/50	92
MNIST/CIFAR-10bw	95/50	95/50	95/50	87
MNIST/Gaussian	90/50	90/50	91/50	91
MNIST/Uniform	99/50	99/50	98/50	83
MNIST/All	91/50	76/20	98/80	89

Table 2: Distinguishing in- and out-of-distribution test set data for image classification. CIFAR-10/All is the same as CIFAR-10/(SUN, Gaussian). All values are percentages.

The authors take the test-set examples, along with the maximal softmax output scores $\hat{p}_{i\,\max}$ for each, as well as the predicted class $\hat{Y}$ and the true class Y of the example. The table above shows that out-of-distribution examples differ significantly from within-distribution examples, which makes their detection easier.

Extending on this baseline method of detecting out-of-distribution examples, Liang et al. (2018), also seek to address the issue that "neural networks tend to make high confidence predictions even for completely uncognizable [...] or irrelevant inputs." The authors' suggestion involves not only postprocessing through the already discussed temperature scaling, but also changes to the model's input, i.e. preprocessing by adding small perturbations to the images - an extension of the previously discussed literature operating within the same paradigm of adjustments to convey uncertainty in output/confidence expressions without changes to a given model. An image preprocessed by the perturbations is given by:

$$\tilde{x} = x - \epsilon \text{sign}(-\nabla_x \log S_{\hat{y}}(\mathbf{x}; T)),$$

ϵ is here to be understood as the perturbation magnitude. The authors note that the perturbations can be computed straightforwardly by back-propagating the gradient of the cross-entropy objective function with respect to the input. They use the following decision rule, with 1 indicating an out-of-distribution sample:

$$g(\mathbf{x}; \delta, T, \epsilon) = \begin{cases} \hat{p}_{i\,\max} \leq \delta \to 1, \\ \hat{p}_{i\,\max} > \delta \to 0. \end{cases}$$

The T, ϵ, and δ are set so that the true positive rate, the rate of within-distribution images correctly classified as within-distribution, is 95%. The considered for T and for ϵ, respectively, are 1 to 1000 and 0 to 0.004 in the experimentation section of the paper, with the goal of minimizing the false positive rate at the true positive rate of 95%. The details of how exactly these are optimized are non-trivial and can be found in the original paper as well as publicly available in its implementation on the researchers' github repository. The observed results on a number of commonly used image datasets are very promising: when the true positive rate is 95%, the false positive rate can be reduced down to 4.2%. The small perturbations along with the temperature scaling are meant to further separate the softmax output scores between within-distribution and out-of-distribution examples, and this result indeed holds across the considered datasets and model architectures. These two adjustments of the model output, if they preserve the impressive results when generalized, certainly contribute significantly towards meeting the needs of the medical applications with regards to expressions of confidence and uncertainty. We speculate, however, for reasons explained in the following section, that the relative importance of detecting out-of-distribution examples is within the medical domain not as critical as calibrating the model – in short, because the model should regardless always be used with great caution and avoided in contexts where images may come from sources different from the one that generated the training examples.

Still richer expressions of uncertainty, featuring distributions of the probability scores, are explained in the section focused on Bayesian neural networks, and how they relate to the (preprocessing and) postprocessing methods is briefly discussed in the following section. Finally, there are further recent developments in out-of-distribution example detection, but these are more involved in adjusting the model itself and as such, are likewise mentioned in the following section and left to the reader to further explore.

IV. Summary and Guidelines

Advances in applying deep learning for classification have achieved human-level and better-than-human accuracy in some domains of medical diagnostics. Despite this progress, the progress and interest in adopting such methods by the medical community in developed countries, have not yet matched the technical developments. Some of the often-expressed reasons for the slow increase in adoption of machine learning methods in medical diagnostics is the relative lack of prominent methods to convey uncertainty, which is a large obstacle to informing treatment decisions made by medical practitioners, even assuming satisfactory performance (Litjens et al., 2017). This review discusses several methods meant to convey uncertainty in class predictions generated by neural networks, with a particular focus on effective methods that are still somewhat easier to implement.

When it comes to conveying uncertainty in a standard model (a deep convolutional neural network) that is already trained, the first and major difficulty is posed by the fact that contemporary neural networks (in 2018) are poorly calibrated. There is a variety of relatively simple methods available to transform the output of a model, i.e. the softmax/sigmoid probability scores $\hat{p}_i$ (and the maximum among these $\hat{p}_{i\,\max}$) in a manner that adjusts the scores to make them more representative of the confidence that the model *should* have in its class prediction $\hat{Y}$. In other words, assuming the example to be classified comes from a distribution sufficiently similar to the one on which the model is trained, well-calibrated models produce probability scores that somewhat closely match the actual probability that the example is correctly classified. As shown in the above section (Guo et al., 2016), one of the simplest and seemingly the most effective methods of this sort for multiclass problems is temperature scaling, which "softens" the output probability scores to make them less confident and closer to the true probability of having classified the example correctly. A method very popular for binary classification problems, i.e. to tranform sigmoid scores, is isotonic regression (the hyperlinks lead to implementation resources). For applications in medical diagnostics, having models that are not only high-performing in terms of sensitivity and specificity (not a topic of this review) but also well-calibrated allows a medical practitioner to rely on the model's predictions knowing at once that the model is overall highly accurate and that the output probability associated with the predicted class is an appropriate expression of the model's confidence. This would bring the output of a deep learning model applied to imaging in the medical domain closer to satisfying the uncertainty-related needs of medical practitioners. Just as importantly, in addition to the available calibration methods, there are several metrics that allow a practitioner to check whether the model in question is well-calibrated: among these, reliability diagrams seem to be particularly user friendly, while the Expected Calibration Error is a useful scalar summary of calibration. For applications within the medical diagnostic domain, calibration seems to be a computationally cheap, relatively easy-to-implement improvement that should always be applied, along with the measures verifying its performance within the training data (before pushed into practice), such as the reliability diagrams and ECE summaries, as well as the maximum calibration error, which is important in high-risk contexts).

The calibration methods make the confidence in predictions implied by the model's output probability scores appropriate only if the example (image) to be classified is sufficiently close to the distribution of the images that the model is trained on. We review two methods to check this assumption. One is trivially simple, due to the helpful finding that softmax probability scores for out-of-distribution examples are typically significantly lower than those for within-distribution examples. To allow greater ability to discriminate between the two, others (Liang et al., 2018) have with impressive results applied temperature scaling (discussed above) to the model's output and small perturbations to the model's input images, further separating the scores. A novel method (Lee et al., 2018) involves adding two additional terms to the model's cross-entropy loss function to be optimized. The first forces predictions about out-of-distribution samples to be less confident and the second serves to generate the most suitable samples for the first one – "[the] method jointly trains both classification and generative neural networks." To share a speculative remark about out-of-distribution detection: the relative importance of calibrating the model's output seems to be greater, since models trained on images coming from one distribution should for medical purposes probably never be used on images which are suspected to be coming from unknown sources i.e. distributions. Of course, one may not in fact know whether there is a reasonable chance to encounter an out-of-distribution image, thus posing a threat to the diagnostics process. Some of the underlying reasons one could be in effect surprised by an unexpected image are that it could be generated from a different device (not the one training images were collected with), or generated for a human population that could be significantly different from the original one, as well as unknown unknowns of a different type. In such contexts, utilizing the out-of-distribution detection methods is especially useful, but cannot replace a generally cautious approach. On the other hand, calibration seems to deliver unambiguous benefits, while, critically, a set of tools makes it possible to check whether the calibration is successfully performed (within the training and test set distribution images).

The previously discussed methods in effect, if successful, allow one to take the (adjusted) standard output of a neural network as a reliable expression of the model's confidence about its own predictions, and, equivalently, the uncertainty of the predictions. If well calibrated, the softmax or sigmoid probability scores convey confidence/uncertainty in an absolute sense, by themselves, while to detect out-of-distribution examples in practice, for which no ground-truth labels are available, the probability scores are used (after also applying perturbations to the input images) so as to compare them with the scores for images within the training data.

There are, however, limitations to using point estimate probability scores as the sole summary of uncertainty. Richer expressions of uncertainty, featuring distributions of the probability scores, are explained in section II, focused on Bayesian neural networks. An implementation termed MC Dropout, which delivers an approximation to the predictive posterior distribution (Gal, 2017) of each parameter via dropout layers is relatively easier to implement and computationally inexpensive. Having not only a point estimate but also a predictive posterior distribution, if successfully obtained, allows a practitioner to assess a range of likely values of true confidence/uncertainty. Utilizing the predictive sigma-square to rank images based on uncertainty estimates, the doctors can determine which set of images require further inspection (Leiber et al., 2015). This brings the neural-network-assisted diagnostic process even closer to mimicking that of the medical practitioners. There are several Python libraries supporting the implementation of Bayesian Neural Networks, including Edward, which also supports TensorFlow objects. Similarly, PyMC supports Theano and will be supporting TensorFlow in its upcoming version.

There are at least two challenges to using MC Dropout. The first **(i)** of these is theoretical, due to a curious result (Osband, 2016) that the predictive distribution does not concentrate with more data, as it should, with commentators speculating that certain assumptions in the original result by Gal (2015) are left unexplicated. The other significant challenge **(ii)** comes from the fact that although Bayesian methods provide the most natural framework for reasoning in light of uncertainty, in practice, often due to approximate inference, the uncertainty estimates are inaccurate, i.e. miscalibrated (Kuleshov et al., 2018). Since Bayesian expressions of uncertainty have more content than the standard ones based only on probability scores $\hat{p}_{i\,\max}$, i.e. they consist of distributions over the point estimates and thus allow the estimation of uncertainty intervals around the point estimates, there is likewise an additional aspect to the corresponding notion of calibration. In this context, it is expressed simply by the percentage of uncertainty intervals that contain the true values: 90% of 90% uncertainty intervals, for instance, should contain the true probability that the predicted class of an example $\hat{Y}_i$ matches the true class of the example, Y_i. Despite these challenges, MC Dropout in particular and Bayesian procedures for obtaining posterior predictive distributions for the output of neural networks are not only very promising but have already been successfully applied to the case of diagnosing diabetic retinopathy, surpassing “85% sensitivity and 80% specificity as recommended by the NHS when referring 020% of the most uncertain decisions for further inspection.” The pragmatic success of this study, including the benefits to performance, along with the relative ease of implementation, are strong reasons to utilize Bayesian neural networks via MC Dropout to obtain estimates of uncertainty for class predictions. Though the first of the above-mentioned challenges remains unadressed, it is for now of a theoretical nature, which means that while one should keep in mind that there may still be discoveries explicating in which conditions MC Dropout successfully approximates predictive posterior distributions, it seems in practice beneficial to use, both for richer expressions of uncertainty and for the benefits it seems to deliver. One should, however, always try to deal with the latter challenge of the potentially miscalibrated uncertainty intervals. A good first step is to produce a reliability diagram/calibration plot in accordance with the above-given definition of calibration for Bayesian estimates (the percentage of uncertainty intervals that contain the true values). If the diagnostics indicate a miscalibration issue, recent results by Kuleshov et al. (2018) suggest that (for classifiers), utilizing the classical recalibration methods resolves the issue, given sufficient data.[6] In practice, this would involve combining an MC Dropout Bayesian neural network architecture design, followed by a calibration procedure (for instance, temperature scaling), and finally, drawing the parameters from the joint posterior distribution of the parameters to simulate the data points to arrive at predictions and their corresponding uncertainties, which should be calibrated.

In summary, there are very strong reasons to, at the minimum, apply calibration methods to the standard output of a neural network in order to allow the interpretation of standard probability scores $\hat{p}_{i\,\max}$ as expressions of the model's confidence/uncertainty about the class predictions. Temperature scaling seems to be effective for multiclass problems, while isotonic regression is a very popular method to calibrate binary classifiers. There are also strong reasons to consider enriching this expression of uncertainty by obtaining posterior predictive distributions and thus uncertainty intervals around the probability scores, by utilizing a procedure such as MC Dropout; it should again be verified that the resulting intervals and point estimates are well calibrated, and, if needed, to correct them using a standard method. Finally, there are also additional steps (described in part B of section III) that may be taken to allow detecting Out-of-Distribution Examples, that is, images that may be understood as drawn from a distribution quite unlike the one from which the images used for training come from. Specifically, applying small perturbations to the input images makes the probability scores $\hat{p}_{i\,\max}$ for the out-of-distribution examples significantly lower than those for within-distribution examples. For careful applications within medical diagnostics but also more broadly, these three procedures should result in large improvements in conveying the uncertainty in the class predictions of neural networks.[7]

[6]Kuleshov et al. in fact focus on extending the classifier-calibration methods to regression and recurrent neural networks.

[7]Within each section, we have carefully mentioned other promising methods to express uncertainty, but have focused on the ones discussed in the summary as simultaneously very promising and not excessively difficult to implement from a practical or computational standpoint.

References and Bibliography

A survey on deep learning in medical image analysis

G. Litjens, T. Kooi, B. E. Bejnordi, A. A. A. Setio, F. Ciompi, M. Ghafoorian, J. A. van der Laak, B. van Ginneken, and C. I. Sanchez (2017)

Weight Uncertainty in Neural Networks

Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra (2017)

Risk versus Uncertainty in Deep Learning: Bayes, Bootstrap and the Dangers of Dropout

Ian Osband (2016)

A study of Bayesian Neural Networks

Vikram Mullachery, Aniruddh Khera, and Amir Husain (2017)

What Uncertainties Do We Need in Bayesian Deep Learning for Computer Vision?

Alex Kendall and Yarin Gal (2017)

Leveraging uncertainty information from deep neural networks for disease detection

Christian Leibig, Vaneeda Allken, Murat Sekin Ayhan, Philipp Berens, and Siegfried Wahl (2017)

Bayesian Convolutional Neural Networks with Bernoulli Approximate Variational Inference

Yarin Gal and Zoubin Ghahramani (2015)

Predicting Good Probabilities With Supervised Learning

Alexandru Niculescu-Mizil and Rich Caruana (2015)

On Calibration of Modern Neural Networks

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger (2015)

Explaining and Harnessing Adversarial Examples

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy (2015)

A Baseline for Detecting Misclassified and Out-of-Distribution Examples in Neural Networks

Dan Hendrycks and Kevin Gimpel (2016, modified 2018)

Enhancing the Reliability of Out-of-Distribution Image Detection in Neural Networks

Shiyu Liang, Yixuan Li, and R. Srikant (2018)

Training Confidence-Calibrated Classifiers for Detecting Out-of-Distribution Samples

Kimin Lee, Honglak Lee, Kibok Lee, and Jinwoo Shin (2018)

Accurate Uncertainties for Deep Learning Using Calibrated Regression

Volodymyr Kuleshov, Nathan Fenner, and Stefano Ermon (2018)