Predictive Uncertainty Estimation via Prior Networks

Andrey Malinin
Department of Engineering
University of Cambridge
am969@cam.ac.uk

Mark Gales
Department of Engineering
University of Cambridge
mjfg@eng.cam.ac.uk

Abstract

Estimating how uncertain an AI system is in its predictions is important to improve the safety of such systems. Uncertainty in predictive can result from uncertainty in model parameters, irreducible data uncertainty and uncertainty due to distributional mismatch between the test and training data distributions. Different actions might be taken depending on the source of the uncertainty so it is important to be able to distinguish between them. Recently, baseline tasks and metrics have been defined and several practical methods to estimate uncertainty developed. These methods, however, attempt to model uncertainty due to distributional mismatch either implicitly through model uncertainty or as data uncertainty. This work proposes a new framework for modeling predictive uncertainty called Prior Networks (PNs) which explicitly models distributional uncertainty. PNs do this by parameterizing a prior distribution over predictive distributions. This work focuses on uncertainty for classification and evaluates PNs on the tasks of identifying out-of-distribution (OOD) samples and detecting misclassification on the MNIST dataset, where they are found to outperform previous methods. Experiments on synthetic and MNIST data show that unlike previous non-Bayesian methods PNs are able to distinguish between data and distributional uncertainty.

1 Introduction

Neural Networks (NNs) have become the dominant approach to addressing computer vision (CV) [1, 2, 3], natural language processing (NLP) [4, 5, 6], speech recognition (ASR) [7, 8] and bio-informatics (BI) [9, 10] tasks. Despite impressive, and ever improving, supervised learning performance, NNs tend to make over-confident predictions [11] and until recently have been unable to provide measures of uncertainty in their predictions. Estimating uncertainty in a model's predictions is important, as it enables, for example, the safety of an AI system [12] to be increased by acting on the model's prediction in an informed manner. This is crucial to applications where the cost of an error is high, such as in autonomous vehicle control and medical, financial and legal fields.

Recently notable progress has been made on predictive uncertainty for Deep Learning through the definition of baselines, tasks and metrics [13] and the development of practical methods for estimating uncertainty. One class of approaches stems from Bayesian Neural Networks [14, 15, 16, 17]. Traditionally, these approaches have been computationally more demanding and conceptually more complicated than non-Bayesian NNs. Crucially, their performance depends on the form of approximation made due to computational constraints and the nature of the prior distribution over parameters. A recent development has been the technique of Monte-Carlo Dropout [18], which estimates predictive uncertainty using an ensemble of multiple stochastic forward passes and computing the mean and spread of the ensemble. This technique has been successfully applied to tasks in computer vision [19, 20]. A number of non-Bayesian ensemble approaches have also been proposed. One approach based on explicitly training an ensemble of DNNs, called Deep Ensembles [11], yields competitive uncertainty estimates to MC dropout. Another class of approaches, developed

for both regression [21] and classification [22], involves explicitly training a model in a multi-task fashion to minimize its Kullback-Leibler (KL) divergence to both a sharp in-domain predictive posterior and a flat out-of-domain predictive posterior, where the out-of-domain inputs are sampled either from a synthetic noise distribution or a different dataset during training. These methods are explicitly trained to detect out-of-distribution inputs and have the advantage of being more computationally efficient at test time.

The primary issue with these approaches is that they conflate different aspects of predictive uncertainty, which results from three separate factors - *model uncertainty*, *data uncertainty* and *distributional uncertainty*. Model uncertainty, or *epistemic uncertainty* [23], measures the uncertainty in estimating the model parameters given the training data - this measures how well the model is matched to the data. Model uncertainty is reducible[1] as the size of training data increases. Data uncertainty, or *aleatoric uncertainty* [23], is irreducible uncertainty which arises from the natural complexity of the data, such as class overlap, label noise, homoscedastic and heteroscedastic noise. Data uncertainty can be considered a 'known-unknown' - the model understands (knows) the data and can confidently state whether a given input is difficult to classify (an unknown). Distributional uncertainty arises due to mismatch between the training and test distributions (also called dataset shift [24]) - a situation which often arises for real world problems. Distributional uncertainty is an 'unknown-unknown' - the model is unfamiliar with the test data and thus cannot confidently make predictions. The approaches discussed above either conflate distributional uncertainty with data (aleatoric) uncertainty or implicitly model distributional uncertainty through model (epistemic) uncertainty, as in Bayesian approaches. The ability to separately model the 3 types of predictive uncertainty is important, as different actions can be taken by the model depending on the source of uncertainty. For example, in active learning tasks detection of distributional uncertainty would indicate the need to collect training data from this distribution. This work addresses the explicit prediction of each of the three types of predictive uncertainty by extending the work done in [21, 22] while taking inspiration from Bayesian approaches.

Summary of Contributions. This work describes the limitations of previous methods of obtaining uncertainty estimates and proposes a new framework for modeling predictive uncertainty, called Prior Networks (PNs), which allows distributional uncertainty to be treated as distinct from both data and model uncertainty. This work focuses on the application of PNs to classification tasks. Additionally, this work presents a discussion of a range of uncertainty metrics in the context of each source of uncertainty. Experiments on synthetic and real data show that unlike previous non-Bayesian methods PNs are able to distinguish between data and distributional uncertainty. Finally, PNs are evaluated on the tasks of identifying out-of-distribution (OOD) samples and detecting misclassification outlined in [13], where they outperform previous methods on the MNIST dataset.

2   Current Approaches to Uncertainty Estimation

This section describes current approaches to predictive uncertainty estimation. Consider a distribution $\mathrm{p}(\boldsymbol{x}, y)$ over input features $\boldsymbol{x}$ and labels y. For image classification $\boldsymbol{x}$ corresponds to images and y object labels. In a Bayesian framework the predictive uncertainty of a classification model $\mathrm{P}(\omega_c|\boldsymbol{x}^*, \mathcal{D})$[2] trained on a finite dataset $\mathcal{D} = \{\boldsymbol{x}_j, y_j\}_{j=1}^N \sim \mathrm{p}(\boldsymbol{x}, y)$ will result from data (aleatoric) and model (epistemic) uncertainty. A model's estimates of data uncertainty are described by the posterior distribution over class labels and model uncertainty is described by the posterior distribution over the parameters given the data (eq. 1).

$$\mathrm{P}(\omega_c|\boldsymbol{x}^*, \mathcal{D}) = \int \underbrace{\mathrm{P}(\omega_c|\boldsymbol{x}^*, \boldsymbol{\theta})}_{Data} \underbrace{\mathrm{p}(\boldsymbol{\theta}|\mathcal{D})}_{Model} d\boldsymbol{\theta} \tag{1}$$

Here, uncertainty in the model parameters induces a distribution over distributions $\mathrm{P}(\omega_c|\boldsymbol{x}, \boldsymbol{\theta})$. The expected distribution $\mathrm{P}(\omega_c|\boldsymbol{x}^*, \mathcal{D})$ is obtained by marginalizing out the parameters $\boldsymbol{\theta}$. The integral in eq. 1 is intractable for neural networks and is typically approximated via sampling (eq. 2), using approaches like Monte-Carlo dropout [18], Langevin Dynamics [25] or explicit ensembling [11].

---

[1]Up to identifiability limits. In the limit of infinite data $\mathrm{p}(\boldsymbol{\theta}|\mathcal{D})$ yields equivalent parameterizations.
[2]Using the standard shorthand for $\mathrm{P}(y = \omega_c|\boldsymbol{x}^*, \mathcal{D})$.

Thus,

$$\mathrm{P}(\omega_c|\boldsymbol{x}^*, \mathcal{D}) \approx \frac{1}{M} \sum_{i=1}^{M} \mathrm{P}(\omega_c|\boldsymbol{x}^*, \boldsymbol{\theta}^{(i)}), \ \boldsymbol{\theta}^{(i)} \sim \mathrm{p}(\boldsymbol{\theta}|\mathcal{D}) \tag{2}$$

Each $\mathrm{P}(\omega_c|\boldsymbol{x}^*, \boldsymbol{\theta}^{(i)})$ in an ensemble obtained via eq. 2 is a categorical distribution $\boldsymbol{\mu}$ over class labels y conditioned on the input $\boldsymbol{x}^*$, and can be visualized as a point on a simplex.

$$\boldsymbol{\mu} = \begin{bmatrix} \mu_1, \cdots, \mu_K \end{bmatrix}^T = \begin{bmatrix} \mathrm{P}(y = \omega_1), \cdots, \mathrm{P}(y = \omega_K) \end{bmatrix}^T \tag{3}$$

For the same $\boldsymbol{x}^*$ this ensemble is a collection of points on a simplex (fig. 1a), which can be seen as samples of categorical distributions from an *implicit* conditional distribution over a simplex (fig. 1b) induced via the posterior over model parameters.
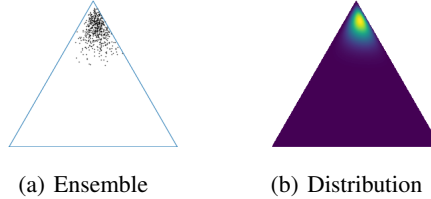

(a) Ensemble   (b) Distribution

Figure 1: Distributions on a Simplex

By selecting an appropriate approximate inference scheme and model prior $\mathrm{p}(\boldsymbol{\theta})$ Bayesian approaches aim to craft a model posterior $\mathrm{p}(\boldsymbol{\theta}|\mathcal{D})$ such that different instantiations of the parameters $\boldsymbol{\theta}^{(i)}$ yield increasingly varied decision boundaries in $\mathrm{P}(\omega_c|\boldsymbol{x}^*, \boldsymbol{\theta}^{(i)})$ the further away $\boldsymbol{x}^*$ is from the region of training data. Thus, these approaches aim to craft an implicit conditional distribution over a simplex (fig. 1b) with the attributes that it is sharp at the corners of a simplex for inputs similar to the training data and flat over the simplex for out-of-distribution inputs. Given an ensemble from such a distribution, the entropy of the expected distribution $\mathrm{P}(\omega_c|\boldsymbol{x}^*, \mathcal{D})$ will indicate uncertainty in predictions. It is not possible, however, to determine from the entropy whether this uncertainty is due to a high degree of data uncertainty, or whether the input is far from the region of training data.

Measures of spread of the ensemble, such as Mutual Information, assess uncertainty in predictions due to model uncertainty. This does allow sources of uncertainty to be determined. In practice, however, for deep, distributed black-box models with tens of millions of parameters, such as DNNs, it is difficult to select an appropriate model prior and approximate inference scheme to craft a model posterior which induces an implicit distribution with the desired properties. This makes it is hard to guarantee the desired properties of the induced distribution for current state-of-the-art Deep Learning approaches. Furthermore, creating an ensemble can be computationally expensive.

An alternative, non-Bayesian class of approaches derives measures of uncertainty via the predictive posteriors of regression [21] and classification [13, 22, 26] DNNs. Here, DNNs are explicitly trained [22, 21] to yield high entropy posterior distributions for out-of-distribution inputs. These approaches are easy to train and inference is computationally cheap. However, a high entropy posterior over classes could indicate uncertainty in the prediction due to *either* an in-distribution input in a region of class overlap or an out-of-distribution input far from the training data. Thus, it is not possible to robustly determine the source of uncertainty using these approaches. Further discussion of uncertainty measures can be found in section 4.

3 Prior Networks

Having described existing approaches, an alternative approach to modeling predictive uncertainty, called Prior Networks, is proposed in this section. As previously described, Bayesian approaches aim to construct an implicit conditional distribution over distributions on a simplex (fig 1b) with certain desirable attributes by appropriate selection of model prior and approximate inference method. In practice this is a difficult task and an open research problem.

This work proposes to instead *explicitly* parametrize a distribution over distributions on a simplex, $\mathrm{p}(\boldsymbol{\mu}|\boldsymbol{x}^*, \boldsymbol{\theta})$, using a DNN referred to as a *Prior Network* and train it to behave like the implicit

distribution in the Bayesian approach. Specifically, when it is confident in its prediction a Prior Network should yield a sharp distribution centered on one of the corners of the simplex. For an input in a region with high degrees of noise or class overlap (data uncertainty) a Prior Network should yield a sharp distribution focused on the center of the simplex, which corresponds to being confident in predicting a flat categorical distribution over class labels (known-unknown). Finally, for 'out-of-distribution' inputs the Prior Network should yield a flat distribution over the simplex, indicating large uncertainty in the mapping $x \mapsto y$ (unknown-unknown).

In the Bayesian framework *distributional uncertainty*, or uncertainty due to mismatch between the distributions of test and training data, is considered a part of model uncertainty. In this work it will be considered to be a source of uncertainty separate from data or model uncertainty. Prior Networks will be explicitly constructed to capture data and distributional uncertainty. In Prior Networks data uncertainty is described by the point-estimate categorical distribution μ and distributional uncertainty is described by the distribution over predictive categoricals $p(\mu|x^*, \theta)$. The parameters θ of the Prior Network must encapsulate knowledge both about the in-domain distribution and the decision boundary which separates the in-domain region from everything else. Construction of a Prior Network is discussed in sections 3.1 and 3.2. Before this it is necessary to discuss its theoretical properties.

Consider modifying eq. 1 by introducing the term $p(\mu|x^*, \theta)$ as follows:

$$P(\omega_c|x^*, \mathcal{D}) = \int \int \underbrace{p(\omega_c|\mu)}_{Data} \underbrace{p(\mu|x^*, \theta)}_{Distributional} \underbrace{p(\theta|\mathcal{D})}_{Model} d\mu d\theta \qquad (4)$$

In this expression data, distribution and model uncertainty are now each modeled by a separate term within an interpretable probabilistic framework. The relationship between uncertainties is made explicit - model uncertainty affects estimates of distributional uncertainty, which in turn affects the estimates of data uncertainty. This is expected, as a large degree of model uncertainty will yield a large variation in $p(\mu|x^*, \theta)$, and large uncertainty in μ will lead to a large uncertainty in estimates of data uncertainty. Thus, model uncertainty affects estimates of data and distributional uncertainties, and distributional uncertainty affects estimates of data uncertainty. Consider marginalizing out μ in eq. 4, thus re-obtaining eq. 1:

$$\int \left[\int p(\omega_c|\mu)p(\mu|x^*, \theta)d\mu \right] p(\theta|\mathcal{D})d\theta = \int P(\omega_c|x^*, \theta)p(\theta|\mathcal{D})d\theta \qquad (5)$$

Since the distribution over μ is lost in the marginalization it is unknown how sharp or flat it was around the point estimate. If the expected categorical $P(\omega_c|x^*, \theta)$ is "flat" it is now unknown whether this is due to high data or distributional uncertainty. In this situation, it will be necessary to again rely on measures which assess the spread of an MC ensemble, like mutual information (section 4), to establish the source of uncertainty. Thus, Prior Networks are consistent with previous approaches to modeling uncertainty, both Bayesian and non-Bayesian - they can be viewed as an 'extra tool in the uncertainty toolbox' which is explicitly crafted to capture the effects of distributional mismatch in a probabilistically interpretable way. Alternatively, consider marginalizing out θ in eq. 4 as follows:

$$\int p(\omega_c|\mu) \left[\int p(\mu|x^*, \theta)p(\theta|\mathcal{D})d\theta \right] d\mu = \int p(\omega_c|\mu)p(\mu|x^*, \mathcal{D})d\mu \qquad (6)$$

This yields expected estimates of data and distributional uncertainty given model uncertainty. Eq. 6 can be seen as a modification of eq. 1 where the model is redefined as $p(\omega_c|\mu)$ and the distribution over model parameters $p(\mu|x^*, \mathcal{D})$ is now conditional on both the training data $\mathcal{D}$ and the test input x^*. This explicitly yields the distribution over the simplex which the Bayesian approach implicitly induces. Unfortunately, like eq. 1, the marginalization in eq. 6 is generally intractable, though it can be approximated via Bayesian MC methods. For simplicity, this work will assume that a point-estimate (eq. 7) of the parameters will be sufficient given appropriate regularization and training data size.

$$p(\theta|\mathcal{D}) = \delta(\theta - \hat{\theta}) \qquad (7)$$

$$p(\mu|x^*; \mathcal{D}) \approx p(\mu|x^*; \hat{\theta}) \qquad (8)$$

3.1 Dirichlet Prior Networks

A Prior Network for classification parametrizes a distribution over a simplex, such as a Dirichlet (eq. 9), Mixture of Dirichlet distributions or the Logistic-Normal distribution. In this work the

Dirichlet distribution is chosen due to its tractable analytic properties. A Dirichlet distribution is a prior distribution over categorical distribution, which is parameterized by its concentration parameters $\boldsymbol{\alpha}$, where α_0, the sum of all α_c, is called the *precision* of the Dirichlet distribution. Higher values of α_0 lead to sharper distributions.

$$\texttt{Dir}(\boldsymbol{\mu}|\boldsymbol{\alpha}) = \frac{\Gamma(\alpha_0)}{\prod_c^K \Gamma(\alpha_c)} \prod_{c=1}^K \mu_c^{\alpha_c - 1}, \quad \alpha_c > 0, \ \alpha_0 = \sum_c^K \alpha_c \tag{9}$$

A Prior Network which parametrizes a Dirichlet will be referred to as a *Dirichlet Prior Network* (DPN). A DPN will generate the concentration parameters $\boldsymbol{\alpha}$ of the Dirichlet distribution.

$$\texttt{p}(\boldsymbol{\mu}|\boldsymbol{x}^*; \hat{\boldsymbol{\theta}}) = \texttt{Dir}(\boldsymbol{\mu}|\boldsymbol{\alpha}) \tag{10}$$

$$\boldsymbol{\alpha} = \boldsymbol{f}(\boldsymbol{x}^*; \hat{\boldsymbol{\theta}}) \tag{11}$$

The posterior over class labels will be given by the mean of the Dirichlet:

$$\texttt{P}(\omega_c|\boldsymbol{x}^*; \hat{\boldsymbol{\theta}}) = \int \texttt{p}(\omega_c|\boldsymbol{\mu}) \texttt{p}(\boldsymbol{\mu}|\boldsymbol{x}^*; \hat{\boldsymbol{\theta}}) d\boldsymbol{\mu} = \frac{\alpha_c}{\alpha_0} \tag{12}$$

If an exponential output function is used for the DPN, where $\alpha_c = e^{z_c}$, then the expected posterior probability of a label ω_c is given by the output of the softmax (eq. 13).

$$\texttt{P}(\omega_c|\boldsymbol{x}^*; \hat{\boldsymbol{\theta}}) = \frac{e^{z_c(\boldsymbol{x}^*)}}{\sum_k^K e^{z_k(\boldsymbol{x}^*)}} \tag{13}$$

Thus, standard DNNs for classification with a softmax output function can be viewed as predicting the expected categorical distribution under a Dirichlet prior. The mean, however, is insensitive to arbitrary scaling of α_c. Thus the precision α_0, which controls the sharpness of the Dirichlet, is degenerate under standard cross-entropy training. It is necessary to change the cost function to explicitly train a DPN to yield a sharp or flat prior distribution around the expected categorical depending on the input data.

3.2 Dirichlet Prior Network Training

There are potentially many ways in which a Prior Network can be trained and it is not the focus of this work to investigate them all. This work considers one approach to training a DPN based on the work done in [21, 22] and here. The DPN is *explicitly* trained in a multi-task fashion to minimize the KL divergence (eq. 14) between the model and a sharp Dirichlet distribution focused on the appropriate class for in-distribution data, and between the model and a flat Dirichlet distribution for out-of-distribution data. A flat Dirichlet is chosen as the uncertain distribution in accordance with the principle of insufficient reason [27], as all possible categorical distributions are equiprobable.

$$\mathcal{L}(\boldsymbol{\theta}) = \texttt{E}_{\texttt{p}_{\text{in}}(\boldsymbol{x})}[KL[\texttt{Dir}(\boldsymbol{\mu}|\hat{\boldsymbol{\alpha}})||\texttt{p}(\boldsymbol{\mu}|\boldsymbol{x}; \boldsymbol{\theta})]] + \texttt{E}_{\texttt{p}_{\text{out}}(\boldsymbol{x})}[KL[\texttt{Dir}(\boldsymbol{\mu}|\tilde{\boldsymbol{\alpha}})||\texttt{p}(\boldsymbol{\mu}|\boldsymbol{x}; \boldsymbol{\theta})]] \tag{14}$$

In order to train using this loss function the in-distribution targets $\hat{\boldsymbol{\alpha}}$ and out-of-distribution targets $\tilde{\boldsymbol{\alpha}}$ must be defined. It is simple to specify a flat Dirichlet distribution by setting all $\tilde{\alpha}_c = 1$. However, directly setting the in-distribution target $\hat{\alpha}_c$ is not convenient. Instead the concentration parameters $\hat{\alpha}_c$ are re-parametrized into $\hat{\alpha}_0$, the target precision, and the means $\hat{\mu}_c = \frac{\hat{\alpha}_c}{\hat{\alpha}_0}$. $\hat{\alpha}_0$ is a hyper-parameter set during training and the means are simply the 1-hot targets used for classification. A further complication is that learning sparse '1-hot' continuous distributions, which are effectively delta functions, is challenging under the defined KL loss, as the error surface becomes poorly suited for optimization. There are two solutions - first, it is possible to smooth the target means (eq. 15), which redistributes a small amount of probability density to the other corners of the Dirichlet. Alternatively, teacher-student training [28] can be used to specify non-sparse target means $\hat{\boldsymbol{\mu}}$. The smoothing approach is used in this work. Additionally, cross-entropy can be used as an auxiliary loss for in-distribution data.

$$\hat{\mu}_c = \frac{1 + \delta(y = \omega_c) \cdot (\hat{\alpha}_0 - K)}{\hat{\alpha}_0} \tag{15}$$

The multi-task training objective (eq. 14) requires samples of $\tilde{\boldsymbol{x}}$ from the out-of-domain distribution $\texttt{p}_{\text{out}}(\boldsymbol{x})$. However, the true out-of-domain distribution is unknown and samples are unavailable. One solution is to synthetically generate points on the boundary of the in-domain region using a generative model [21, 22]. An alternative is to use a different, real dataset as a set of samples from the out-of-domain distribution [22].

4   Uncertainty Measures

The previous section introduced a new framework for modeling uncertainty. This section explores a range of measures for quantifying uncertainty given a trained DNN, DPN or Bayesian MC ensemble. The discussion is broken down into 4 classes of measure, depending on how eq. 4 is marginalized. Details of derivation can be found in Appendix C.

The first class derives measures of data uncertainty from the expected predictive categorical $P(\omega_c|\boldsymbol{x}^*; \mathcal{D})$, given a full marginalization of eq. 4 which can be approximated either with a point estimate of the parameters $\hat{\boldsymbol{\theta}}$ or a Bayesian MC ensemble. The first measure is the probability of the predicted class (mode), or *max probability* (eq. 16), which is a measure of confidence in the prediction, used in [13, 22, 26, 23, 11].

$$\mathcal{P} = \max_c \; P(\omega_c|\boldsymbol{x}^*; \mathcal{D}) \tag{16}$$

The second measure is the *entropy* (eq. 17) of the predictive distribution [23, 18, 11]. It behaves similar to max probability, but represents the uncertainty encapsulated in the entire distribution.

$$\mathcal{H}[P(y|\boldsymbol{x}^*; \mathcal{D})] = -\sum_{c=1}^{K} P(\omega_c|\boldsymbol{x}^*; \mathcal{D}) \ln(P(\omega_c|\boldsymbol{x}^*; \mathcal{D})) \tag{17}$$

The second class of measures considers marginalizing out $\boldsymbol{\mu}$ in eq. 4, yielding eq. 1. Model uncertainty is used to obtain a spread in the expected predictive categoricals. *Mutual Information* (MI) [23] between the categorical label y and the parameters of the model $\boldsymbol{\theta}$ assesses the spread of predictive categoricals of an MC ensemble [18]. MI assesses uncertainty in predictions due to model uncertainty, which implicitly also captures elements of distributional uncertainty. MI can be expressed as the difference of the entropy of expected distribution and the expected entropy of the distribution (eq. 18).

$$\mathcal{I}[y, \boldsymbol{\theta}|\boldsymbol{x}^*, \mathcal{D}] = \mathcal{H}[E_{p(\boldsymbol{\theta}|\mathcal{D})}[P(y|\boldsymbol{x}^*, \boldsymbol{\theta})]] - E_{p(\boldsymbol{\theta}|\mathcal{D})}[\mathcal{H}[P(y|\boldsymbol{x}^*, \boldsymbol{\theta})]] \tag{18}$$

The third class of measures considers marginalizing out $\boldsymbol{\theta}$ in eq. 4, yielding eq. 6. The first measure is the *differential entropy* (eq. 19) of the DPN. This measure is maximized when all categorical distributions are equi-probable, which occurs when the Dirichlet Distribution is flat - in other words when there is the greatest variety of samples from the Dirichlet prior. Differential entropy is well suited to measuring distributional uncertainty, as it can be low even if the expected categorical under the Dirichlet prior has high entropy, and also captures elements of data uncertainty.

$$\mathcal{H}[p(\boldsymbol{\mu}|\boldsymbol{x}^*; \mathcal{D})] = -\int_{\mathcal{S}^{K-1}} p(\boldsymbol{\mu}|\boldsymbol{x}^*; \mathcal{D}) \ln(p(\boldsymbol{\mu}|\boldsymbol{x}^*; \mathcal{D})) d\boldsymbol{\mu} \tag{19}$$

Another measure of uncertainty in this class is the mutual information between y and $\boldsymbol{\mu}$ (eq. 20), which behaves in exactly the same way as MI between y and $\boldsymbol{\theta}$, but the spread is now explicitly due to distributional uncertainty, rather than model uncertainty.

$$\mathcal{I}[y, \boldsymbol{\mu}|\boldsymbol{x}^*; \mathcal{D}] = \mathcal{H}[E_{p(\boldsymbol{\mu}|\boldsymbol{x}^*; \mathcal{D})}[P(y|\boldsymbol{\mu})]] - E_{p(\boldsymbol{\mu}|\boldsymbol{x}^*; \mathcal{D})}[\mathcal{H}[P(y|\boldsymbol{\mu})]] \tag{20}$$

The final class of measures uses the full eq. 4 and assesses the spread of $p(\boldsymbol{\mu}|\boldsymbol{x}^*; \boldsymbol{\theta})$ due to model uncertainty via the MI between $\boldsymbol{\mu}$ and $\boldsymbol{\theta}$, which can be computed via Bayesian ensemble approaches.

5   Experiments

The previous sections discussed modeling different aspects of predictive uncertainty and presented several measures of quantifying it. This section compares the proposed and previous methods in two sets of experiments. The first experiment illustrates the advantages of a DPN over other non-Bayesian methods [22, 26] on synthetic data and the second set of experiments evaluate DPNs on MNIST and compares them to DNNs and ensembles generated via Monte-Carlo Dropout (MCDP) on the tasks of misclassification detection and out-of-distribution data detection. The experimental setup is described in Appendix A and additional experiments are described in Appendix B.

5.1 Synthetic Experiments

A synthetic experiment was designed to illustrate the limitation of using uncertainty measures derived from $P(\omega_c|\boldsymbol{x}^*; \mathcal{D})$ [22, 26] to detect out-of-distribution samples. A simple dataset with 3 Gaussian distributed classes with equidistant means and tied isotropic variance σ is created. The classes are non-overlapping when $\sigma = 1$ (fig. 2a) and overlap when $\sigma = 4$ (fig. 2d). The entropy of the *true* posterior over class labels is plotted in blue in fig. 2a and 2d, which show that when the classes are distinct the entropy is high only on the decision boundaries, but when the classes overlap the entropy is high also within the data region. A small DPN with 1 hidden layer of 50 neurons is trained on this data. Fig. 2b and 2c show that when classes are distinct both the entropy of the DPN's predictive posterior and the differential entropy of the DPN are both low in the region of data and high elsewhere, allowing in-distribution and out-of-distribution regions to be distinguished. Fig. 2e and 2f, however, show that when there is a large degree of class overlap the entropy is high both in region of class overlap and far from training data, making difficult to distinguish out-of-distribution samples and in-distribution samples at a decision boundary. In contrast, the differential entropy is low over the whole region of training data and high outside, allowing the in-distribution region to be clearly distinguished from the out-of-distribution region.
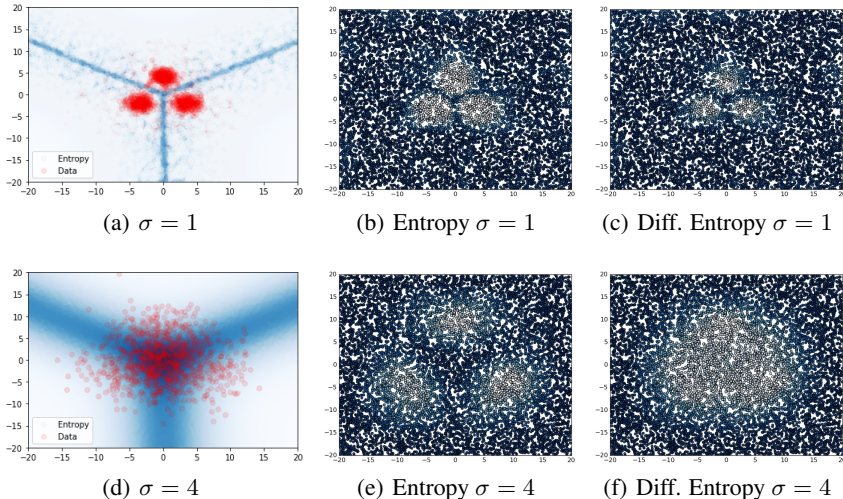

(a) $\sigma = 1$ (b) Entropy $\sigma = 1$ (c) Diff. Entropy $\sigma = 1$

(d) $\sigma = 4$ (e) Entropy $\sigma = 4$ (f) Diff. Entropy $\sigma = 4$

Figure 2: Synthetic Experiment

5.2 MNIST Experiments

An in-domain misclassification detection experiment and an out-of-distribution (OOD) input detection experiment were run on the MNIST dataset [29] to assess the DPN's ability to estimate uncertainty. The misclassification detection experiment involves detecting whether a given prediction is incorrect given an uncertainty metric. Misclassifications are chosen as the positive class. The Misclassification detection experiment was run on a joint MNIST valid+test set. The out-of-distribution detection experiment involves detecting whether an input is out-of-distribution given the uncertainty score. Out-of-distribution samples are chosen as the positive class. The OMNIGLOT dataset [30], scaled down to 28x28 pixels, was used as real 'OOD' data for evaluation. 15000 samples of OMNIGLOT data were randomly selected to form a balanced set of positive (OMNIGLOT) and negative (MNIST valid+test) samples. The two considered baseline approaches derive uncertainty measures from either the class posterior of a DNN [13] or an ensemble generated via MC dropout applied to the same DNN [23, 18]. All uncertainty measures described in section 4 are explored for both tasks in order to see which yield best performance. The performance is assessed by area under the ROC (AUROC) and Precision-Recall (AUPR) curves in both experiments as in [13].

Table 1 shows that the DPN consistently outperforms both a DNN, and a MC dropout ensemble (MCDP) in misclassification detection performance, although there is a negligible drop in accuracy of the DPN as compared to a DNN or MCDP. Max probability yields the best results, closely followed

by the entropy of the predictive distribution. This is expected, as max probability is directly related to the predicted class, while the other metrics measure the uncertainty of the entire distribution. The performance difference is more pronounced on AUPR, which is sensitive to mis-balanced classes.

Table 1: MNIST valid+test misclassification detection

Model	AUROC				AUPR				% Err.
	Max.P	Ent.	M.I.	D.Ent.	Max.P	Ent.	M.I.	D.Ent.	
DNN	98.0	98.6	-	-	26.6	25.0	-	-	0.4
MCDP	97.2	97.2	96.9	-	33.0	29.0	27.8	-	0.4
DPN	**99.0**	98.9	98.6	92.9	**43.6**	39.7	30.7	25.5	0.6

Table 2 shows that a DPN consistently outperforms the baselines in OOD sample detection and is able to perfectly classify all samples using max probability, entropy and differential entropy.

Table 2: MNIST valid+test vs OMNIGLOT out-of-distribution detection

Model	AUROC				AUPR			
	Max.P	Ent.	M.I.	D.Ent.	Max.P	Ent.	M.I.	D.Ent.
DNN	98.7	98.8	-	-	98.3	98.5	-	-
MCDP	99.2	99.2	99.3	-	99.0	99.1	99.3	-
DPN	**100.0**	**100.0**	99.5	**100.0**	**100.0**	**100.0**	97.5	**100.0**

MNIST is a simple task, so it is unsurprising that all considered models and measures perform well. It is interesting to see whether differential entropy of the Dirichlet prior will be able to distinguish in-domain and out-of-distribution data better than entropy when the classes are less distinct. To this end zero mean isotropic Gaussian noise with a standard deviation $\sigma = 3$ noise is added to the inputs of the DNN and DPN during both training and evaluation. Table 3 shows that in the presence of strong noise entropy and MI fail to successfully discriminate between in-domain and out-of-distribution samples, while performance using differential entropy barely falls.

Table 3: MNIST valid+test set vs OMNIGLOT. Out-of-distribution detection AUROC on noisy data.

	Ent.		M.I.		D.Ent.	
σ	0.0	3.0	0.0	3.0	0.0	3.0
DNN	98.8	58.4	-	-	-	-
MCDP	98.8	58.4	99.3	79.1	-	-
DPN	100.0	51.8	99.5	22.3	100.0	99.8

6   Conclusion

This work describes the limitations of previous work on predictive uncertainty estimations within the context of sources of uncertainty and proposes to treat out-of-distribution (OOD) inputs as a separate source of uncertainty, called *Distributional Uncertainty*. To this end, this work presents a novel framework, called Prior Networks (PN), which allows data, distributional and model uncertainty to be treated separately within a consistent probabilistically interpretable framework. A particular form of these PNs are applied to classification, Dirichlet Prior Networks (DPNs). DPNs are shown to yield more accurate estimates of distributional uncertainty than MC Dropout and standard DNNs on the task of OOD detection on the MNIST dataset. The DPNs also outperform other methods on the task of misclassification detection. A range of uncertainty measures is presented and analyzed in the context of the types of uncertainty which they assess. It was noted that the max probability of the predictive distribution yielded the best results on misclassification detection. Differential entropy of DPN was best for OOD detection, especially when classes are less distinct. This was illustrated on both a synthetic experiment and on a noise-corrupted MNIST task. Uncertainty measures can be analytically calculated at test time for DPNs, reducing computational cost relative to ensemble approaches. Having investigated PNs for image classification, it is interesting to apply them to other tasks computer vision, NLP, machine translation, speech recognition and reinforcement learning. Finally, it is necessary to explore Prior Networks for regression tasks.

References

[1] Ross Girshick, "Fast R-CNN," in *Proc. 2015 IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 1440–1448.

[2] Karen Simonyan and Andrew Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," in *Proc. International Conference on Learning Representations (ICLR)*, 2015.

[3] Ruben Villegas, Jimei Yang, Yuliang Zou, Sungryull Sohn, Xunyu Lin, and Honglak Lee, "Learning to Generate Long-term Future via Hierarchical Prediction," in *Proc. International Conference on Machine Learning (ICML)*, 2017.

[4] Tomas Mikolov et al., "Linguistic Regularities in Continuous Space Word Representations," in *Proc. NAACL-HLT*, 2013.

[5] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean, "Efficient Estimation of Word Representations in Vector Space," 2013, arXiv:1301.3781.

[6] Tomas Mikolov, Martin Karafiát, Lukás Burget, Jan Cernocký, and Sanjeev Khudanpur, "Recurrent Neural Network Based Language Model," in *Proc. INTERSPEECH*, 2010.

[7] Geoffrey Hinton, Li Deng, Dong Yu, George Dahl, Abdel rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara Sainath, and Brian Kingsbury, "Deep neural networks for acoustic modeling in speech recognition," *Signal Processing Magazine*, 2012.

[8] Awni Y. Hannun, Carl Case, Jared Casper, Bryan Catanzaro, Greg Diamos, Erich Elsen, Ryan Prenger, Sanjeev Satheesh, Shubho Sengupta, Adam Coates, and Andrew Y. Ng, "Deep speech: Scaling up end-to-end speech recognition," 2014, arXiv:1412.5567.

[9] Rich Caruana, Yin Lou, Johannes Gehrke, Paul Koch, Marc Sturm, and Noemie Elhadad, "Intelligible models for healthcare: Predicting pneumonia risk and hospital 30-day readmission," in *Proc. 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, New York, NY, USA, 2015, KDD '15, pp. 1721–1730, ACM.

[10] Babak Alipanahi, Andrew Delong, Matthew T. Weirauch, and Brendan J. Frey, "Predicting the sequence specificities of DNA- and RNA-binding proteins by deep learning," *Nature Biotechnology*, vol. 33, no. 8, pp. 831–838, July 2015.

[11] B. Lakshminarayanan, A. Pritzel, and C. Blundell, "Simple and Scalable Predictive Uncertainty Estimation using Deep Ensembles," in *Proc. Conference on Neural Information Processing Systems (NIPS)*, 2017.

[12] Dario Amodei, Chris Olah, Jacob Steinhardt, Paul F. Christiano, John Schulman, and Dan Mané, "Concrete problems in AI safety," `http://arxiv.org/abs/1606.06565`, 2016, arXiv: 1606.06565.

[13] Dan Hendrycks and Kevin Gimpel, "A Baseline for Detecting Misclassified and Out-of-Distribution Examples in Neural Networks," `http://arxiv.org/abs/1610.02136`, 2016, arXiv:1610.02136.

[14] David JC MacKay, "A practical bayesian framework for backpropagation networks," *Neural computation*, vol. 4, no. 3, pp. 448–472, 1992.

[15] David JC MacKay, *Bayesian methods for adaptive models*, Ph.D. thesis, California Institute of Technology, 1992.

[16] Geoffrey E. Hinton and Drew van Camp, "Keeping the neural networks simple by minimizing the description length of the weights," in *Proc. Sixth Annual Conference on Computational Learning Theory*, New York, NY, USA, 1993, COLT '93, pp. 5–13, ACM.

[17] Radford M. Neal, *Bayesian learning for neural networks*, Springer Science & Business Media, 1996.

[18] Yarin Gal and Zoubin Ghahramani, "Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning," in *Proc. 33rd International Conference on Machine Learning (ICML-16)*, 2016.

[19] A. Kendall, Y. Gal, and R. Cipolla, "Multi-Task Learning Using Uncertainty to Weight Losses for Scene Geometry and Semantics," in *Proc. Conference on Neural Information Processing Systems (NIPS)*, 2017.

[20] A. Kendall and Y. Gal, "What Uncertainties Do We Need in Bayesian Deep Learning for Computer Vision," in *Proc. Conference on Neural Information Processing Systems (NIPS)*, 2017.

[21] A. Malinin, A. Ragni, M.J.F. Gales, and K.M. Knill, "Incorporating Uncertainty into Deep Learning for Spoken Language Assessment," in *Proc. 55th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2017.

[22] Kimin Lee, Honglak Lee, Kibok Lee, and Jinwoo Shin, "Training confidence-calibrated classifiers for detecting out-of-distribution samples," *International Conference on Learning Representations*, 2018.

[23] Yarin Gal, *Uncertainty in Deep Learning*, Ph.D. thesis, University of Cambridge, 2016.

[24] Joaquin Quiñonero-Candela, *Dataset Shift in Machine Learning*, The MIT Press, 2009.

[25] Max Welling and Yee Whye Teh, "Bayesian Learning via Stochastic Gradient Langevin Dynamics," in *Proc. International Conference on Machine Learning (ICML)*, 2011.

[26] Shiyu Liang, Yixuan Li, and R. Srikant, "Enhancing the reliability of out-of-distribution image detection in neural networks," in *Proc. International Conference on Learning Representations*, 2018.

[27] Kevin P. Murphy, *Machine Learning*, The MIT Press, 2012.

[28] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean, "Distilling the knowledge in a neural network," 2015, arXiv:1503.02531.

[29] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the* IEEE, vol. 86, pp. 2278–2324, 1998.

[30] Brenden M. Lake, Ruslan Salakhutdinov, and Joshua B. Tenenbaum, "Human-level concept learning through probabilistic program induction," *Science*, vol. 350, no. 6266, pp. 1332–1338, 2015.

[31] Ian J. Goodfellow, Yaroslav Bulatov, Julian Ibarz, Sacha Arnoud, and Vinay D. Shet, "Multi-digit number recognition from street view imagery using deep convolutional neural networks," 2013, arXiv:1312.6082.

[32] Alex Krizhevsky, "Learning multiple layers of features from tiny images," 2009.

[33] M Buscema, "Metanet: The theory of independent judges," *Substance Use & Misuse*, vol. 33, no. 2, pp. 439–461, 1998.

[34] Fisher Yu, Yinda Zhang, Shuran Song, Ari Seff, and Jianxiong Xiao, "LSUN: construction of a large-scale image dataset using deep learning with humans in the loop," 2015, arXiv:1506.03365.

[35] Stanford CS231N, "Tiny ImageNet," `https://tiny-imagenet.herokuapp.com/`, 2017.

[36] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Deep residual learning for image recognition," in *Proc. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.

[37] Martín Abadi et al., "TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems," 2015, Software available from tensorflow.org.

[38] Timothy Dozat, "Incorporating Nesterov Momentum into Adam," in *Proc. International Conference on Learning Representations (ICLR)*, 2016.

[39] Sergey Ioffe and Christian Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift," *Journal of Machine Learning Research*, 2015.

Appendix A   Experimental Setup and Datasets

For both core and additional experiments models were trained on the MNIST [29], SVHN [31] and CIFAR [32] datasets. Dataset sizes can be found in table 4. In addition to the datasets described

Table 4: Training and Evaluation Datasets

Dataset	Train	Valid	Test	Classes
MNIST	55000	5000	10000	
SVHN	73257	-	26032	10
CIFAR-10	50000	-	10000	
CIFAR-100	50000	-	10000	100

above, the OMNIGLOT [30], SEMEION [33], LSUN [34] and TinyImagenet [35] datasets were used in out-of-distribution input detection experiments. For these datasets only their test sets were used, described in table 4. TinyImagenet was resized down to 32x32 from 64x64 and OMNIGLOT was resized down to 28x28. For all datasets the input features were scaled to be between -1.0 and 1.0

Table 5: Additional Evaluation Datasets

Dataset	Size
OMNIGLOT	32460
SEMEION	1593
LSUN	10000
tinyImagenet	10000

rather than 0 and 255. No additional preprocessing was done. For models trained on CIFAR-10, augmentation was used by flipping the images left-right randomly.

All experiments were run on convolutional VGG [2] and ResNet [36] models for image classification. Models were implemented in Tensorflow [37], trained using the NADAM optimizer [38] with an exponentially decaying learning rate. Details of the architectures used for each dataset can be found in table 6. For convolution architectures after every pooling layer the number of filters was doubled. ResNet architecture a residual block has 2 convolution layers and pooling occurred after 2 residual blocks. Notably, batch normalization [39] was not used for the ResNet DNN and DPN because it is not clear how batch normalization interacts with noise data. Instead, leaky-ReLU units activations were used. All convolutions were 3x3, except for the first layer in the ResNet architecture, which has 5x5 convolutions.

Table 6: Architecture Sizes

Dataset	Arch.	Activation	Conv Depth	# Base Filters	FC Layers	FC units
MNIST	VGG	ReLU	4	64	1	100
SVHN	VGG	ReLU	7	64	1	512
CIFAR-10	ResNet	Leaky ReLU	17	128	-	-

The training configuration for all models is described in table 7. Interestingly, it was necessary to use less dropout for the DPN, due to the regularization effect of the noise data.

Table 7: Training Configuration

Dataset	Model	Dropout	LR	$\hat{\alpha}_0$	CE weight	OOD data
MNIST	DNN	0.5	1e-3	-	-	-
	DPN	0.95	1e-3	1e3	0.0	MNIST FA
SVHN	DNN	0.5	1e-4	-	-	-
	DPN	0.9	5e-4	1e3	2.0	CIFAR-10
CIFAR-10	DNN	0.8	1e-4	-	-	-
	DPN	0.9	1e-4	1e2	1.0	CIFAR-100

For the DPN trained on MNIST data the out-of-distribution data was synthesized using a Factor Analysis model with a 50-dimensional latent space. In standard factor analysis the latent vectors have

an isotropic standard normal distribution. To push the FA model to produce data at the boundary of the in-domain region the variance on the latent distribution was increased.

Appendix B   Additional Experiments

Further experiments have been run in addition to the core experiments described in section 5. In appendix B.1 the MNIST DNN and DPN described in section 5.2 is evaluated against other out-of-distribution datasets. In appendix B.2 and B.3 a DPN is trained on the SVHN [31] and CIFAR-10 [32] datasets, respectively, and evaluated on the tasks of misclassification detection and out-of-distribution input detection. In these two sections only entropy and differential entropy are used as uncertainty measures, as they consistently yield best performance. Other metrics, such the maximum probability and mutual information are consistently outperformed. In section B.2 for the misclassification detection experiment only max probability is used for the same reason - it is the best measure of uncertainty for this task.

B.1   Additional MNIST experiments

In Table 8 out-of-distribution input detection is run against the SEMEION dataset of greyscale handwritten digits. This dataset's primary difference from MNIST is that it has no padding between the edge of the image and the digit. SEMEION digits were upscaled to 28x28 size for these experiments. Note, as SEMEION is a very small dataset it was not possible to get a balanced set of MNIST and SEMEION images, so AUPR is a better performance metric than AUROC on this particular experiment. The DPN again achieves the best performance whilst a standard DNN and MC Dropout have more difficulty discerning between MNIST and SEMEION images.

Table 8: MNIST vs SEMEION out-of-domain detection

Model	AUROC		AUPR	
	Ent.	D.Ent.	Ent.	D.Ent.
DNN	92.9	-	76.7	-
MCDP	95.5	-	84.2	-
DPN	99.6	**99.7**	97.5	**98.6**

In Table 9 SVHN images are used as the out-of-distribution dataset for MNIST. SVHN images are turned to greyscale and resized down to 28x28 pixels in size. Equal amounts of MNIST and SVHN images were used. All the models achieve a high performance, with the DPN yielding the best performance. This is unsurprising, as MNIST and SVHN are quite different.

Table 9: MNIST vs SVHN out-of-domain detection

Model	AUROC		AUPR	
	Ent.	D.Ent.	Ent.	D.Ent.
DNN	98.9	-	98.7	-
MCDP	98.4	-	98.3	-
DPN	**100.0**	**100.0**	**100.0**	**100.0**

Finally, resized 28x28 greyscale CIFAR-10 images are used as the out-of-distribution dataset for MNIST. Since CIFAR and MNIST are very different, the DNN and MC dropout approaches achieve near-perfect performance and the DPN achieve perfect performance. Equal amounts of MNIST and CIFAR images were used.

Table 10: MNIST vs CIFAR out-of-domain detection

Model	AUROC		AUPR	
	Ent.	D.Ent.	Ent.	D.Ent.
DNN	99.5	-	99.4	-
MCDP	99.3	-	99.2	-
DPN	**100.0**	**100.0**	**100.0**	**100.0**

B.2 SVHN Experiments

This section describes misclassification and out-of-distribution input detections experiments on the SVHN dataset. A DPN trained on SVHN used the CIFAR-10 dataset as the noise dataset, rather than using a generative model like Factor Analysis, VAE or GAN. Investigation of appropriate methods to synthesize out-of-distribution data for complex datasets is beyond the scope of this work.

Table 11 describes the misclassification detection experiment on SVHN. Note, all models achieve comparable classification error (5.8-6.1%). The DPN outperforms the baselines by a small margin in AUROC and AUPR misclassification detection measures, though all approaches achieve very similar performance. Performance using the max probability measure is reported.

Table 11: SVHN test misclassification detection

Model	AUROC	AUPR	% Err.
DNN	92.1	47.7	**5.8**
MCDP	92.4	48.0	5.9
DPN	**92.8**	**50.7**	6.1

Tables 12-14 describe out-of-distribution detection of SVHN vs CIFAR-100, LSUN and TinyImageNet datasets, respectively. In all experiments the DPN is seen to consistently achieve perfect performance and the baselines near perfect performance. Given the SVHN is very different from CIFAR-100, LSUN and TinyImageNet, this an easy task.

Table 12: SVHN vs CIFAR-100 out-of-domain detection

Model	AUROC		AUPR	
	Ent.	D.Ent.	Ent.	D.Ent.
DNN	97.4	-	97.2	-
MCDP	97.2	-	96.9	-
DPN	99.9	**100.0**	99.9	**100.0**

Table 13: SVHN vs LSUN out-of-domain detection

Model	AUROC		AUPR	
	Ent.	D.Ent.	Ent.	D.Ent.
DNN	97.4	-	97.0	-
MCDP	97.2	-	96.7	-
DPN	99.9	**100.0**	99.9	**100.0**

Table 14: SVHN vs TinyImageNet out-of-domain detection

Model	AUROC		AUPR	
	Ent.	D.Ent.	Ent.	D.Ent.
DNN	97.9	-	98.8	-
MCDP	97.7	-	98.7	-
DPN	99.9	**100.0**	**100.0**	**100.0**

B.3 CIFAR-10 Experiments

This section presents the results of misclassification and out-of-distribution input detections experiments on the CIFAR-10 dataset. A DPN trained on CIFAR-10 used the CIFAR-100 dataset as the noise dataset. CIFAR-100 describes different objects than CIFAR-10, so there is no class overlap. Otherwise, CIFAR-100 is very similar to CIFAR-10. This is the most challenging set of experiments, as visually CIFAR-10 is much more similar to CIFAR-100, LSUN and TinyImageNet, so out-of-distribution input detection is likely to more difficult than for simpler tasks like MNIST and SVHN. Performance using all measures of uncertainty are described here for completeness.

Table 15 gives the results of the misclassification detection experiment on CIFAR-10. All models achieve comparable classification error (15.7-16.3%). Interestingly, the noise data acts as a regularizer and improves classification performance using the DPN as compared to the baselines. Curiously, the DPN outperforms the baselines by a larger margin than previously than in the MNIST and SVHN experiments.

Table 15: CIFAR-10 test misclassification detection

Model	AUROC				AUPR				% Err.
	Max.P	Ent.	M.I.	D.Ent.	Max.P	Ent.	M.I.	D.Ent.	
DNN	86.9	87.0	-	-	51.4	51.3	-	-	16.3
MCDP	86.8	86.6	84.2	-	51.3	50.9	44.7	-	16.9
DPN	**88.3**	88.1	85.6	86.3	**56.9**	55.5	45.3	50.2	**15.7**

Tables 16-18 show out-of-distribution detection of CIFAR-10 vs SVHN, LSUN and TinyImageNet datasets. Across all measures, in each experiment the performance of the baseline approaches is noticeable lower than before, especially mutual information. DPNs achieve the best performance, outperforming the baselines by a larger margin than previously. The margin is largest on TinyImageNet (table 18), which seems to be most similar to CIFAR-10. This set of experiments clearly demonstrates that Prior Networks perform well on much more difficult datasets than MNIST and are able to outperform previously proposed Bayesian and non-Bayesian approaches.

Table 16: CIFAR-10 vs SVHN out-of-domain detection

Model	AUROC				AUPR			
	Max.P	Ent.	M.I.	D.Ent.	Max.P	Ent.	M.I.	D.Ent.
DNN	87.5	88.8	-	-	81.7	83.1	-	-
MCDP	84.4	85.8	67.0	-	77.8	78.3	54.7	-
DPN	94.6	95.0	71.0	**95.9**	93.7	94.3	57.3	**95.7**

Table 17: CIFAR-10 vs LSUN out-of-domain detection

Model	AUROC				AUPR			
	Max.P	Ent.	M.I.	D.Ent.	Max.P	Ent.	M.I.	D.Ent.
DNN	77.3	78.7	-	-	72.8	75.6	-	-
MCDP	77.8	79.8	80.1	-	73.8	77.1	77.2	-
DPN	91.6	91.8	72.0	**93.0**	90.3	90.4	58.3	**91.9**

Table 18: CIFAR-10 vs TinyImageNet out-of-domain detection

Model	AUROC				AUPR			
	Max.P	Ent.	M.I.	D.Ent.	Max.P	Ent.	M.I.	D.Ent.
DNN	73.5	74.5	-	-	69.5	71.9	-	-
MCDP	74.2	75.7	75.6	-	70.7	73.4	73.0	-
DPN	89.4	89.5	73.0	**90.2**	93.4	93.5	75.0	**94.2**

Appendix C   Derivations for Uncertainty Measures

This appendix provides the derivations and shows how calculate the uncertainty measures discussed in section 4 for a DNN/DPN and a Bayesian Monte-Carlo Ensemble.

C.1   Entropy of Predictive Distribution for Bayesian MC Ensemble

Entropy of the predictive posterior can be calculated for a Bayesian MC Ensemble using the following derivation, which is taken from Yarin Gal's PhD thesis [23].

$$
\begin{aligned}
\mathcal{H}[\mathrm{P}(y|\boldsymbol{x}^*,\mathcal{D})] &= -\sum_{c=1}^{K} \mathrm{P}(\omega_c|\boldsymbol{x}^*,\mathcal{D}) \ln \mathrm{P}(\omega_c|\boldsymbol{x}^*,\mathcal{D}) \\
&= -\sum_{c=1}^{K} \Big(\int \mathrm{p}(\omega_c|\boldsymbol{x}^*,\boldsymbol{\theta}) \mathrm{p}(\boldsymbol{\theta}|\mathcal{D}) d\boldsymbol{\theta} \Big) \ln \Big(\int \mathrm{P}(\omega_c|\boldsymbol{x}^*,\boldsymbol{\theta}) \mathrm{p}(\boldsymbol{\theta}|\mathcal{D}) d\boldsymbol{\theta} \Big) \\
&\approx -\sum_{c=1}^{K} \Big(\int \mathrm{P}(\omega_c|\boldsymbol{x}^*,\boldsymbol{\theta}) \mathrm{q}^*(\boldsymbol{\theta}) d\boldsymbol{\theta} \Big) \ln \Big(\int \mathrm{P}(\omega_c|\boldsymbol{x}^*,\boldsymbol{\theta}) \mathrm{q}^*(\boldsymbol{\theta}) d\boldsymbol{\theta} \Big) \\
&\approx -\sum_{c=1}^{K} \Big(\frac{1}{N} \sum_{i}^{N} \mathrm{P}(\omega_c|\boldsymbol{x}^*,\boldsymbol{\theta}^{(i)}) \Big) \ln \Big(\frac{1}{N} \sum_{i}^{N} \mathrm{P}(\omega_c|\boldsymbol{x}^*,\boldsymbol{\theta}^{(i)}) \Big)
\end{aligned}
$$

C.2   Differential Entropy of Dirichlet Prior Network

The derivation of differential entropy simply quotes the standard result for Dirichlet distributions. Notably the α_c are a function of $\boldsymbol{x}^*$ and ψ is the *digamma function* and $Gamma$ is the *Gamma function*.

$$
\begin{aligned}
\mathcal{H}[\mathrm{p}(\boldsymbol{\mu}|\boldsymbol{x}^*;\hat{\boldsymbol{\theta}})] &= -\int_{\mathcal{S}^{K-1}} \mathrm{p}(\boldsymbol{\mu}|\boldsymbol{x};\hat{\boldsymbol{\theta}}) \ln(\mathrm{p}(\boldsymbol{\mu}|\boldsymbol{x};\hat{\boldsymbol{\theta}})) d\boldsymbol{\mu} \\
&= \sum_{c}^{K} \ln \Gamma(\alpha_c) - \ln \Gamma(\alpha_0) - \sum_{c}^{K} (\alpha_c - 1) \cdot (\psi(\alpha_c) - \psi(\alpha_0))
\end{aligned}
$$

C.3   Mutual Information for Bayesian MC Ensemble

The Mutual information between class label and parameters can be calculated for a Bayesian MC Ensemble using the following derivation, which is also taken from Yarin Gal's PhD thesis [23].

$$
\begin{aligned}
\mathcal{I}[y,\boldsymbol{\theta}|\boldsymbol{x}^*,\mathcal{D}] &= \mathcal{H}[\mathrm{P}(y|\boldsymbol{x}^*,\mathcal{D})] - \mathrm{E}_{\mathrm{p}(\boldsymbol{\theta}|\mathcal{D})}[\mathcal{H}[\mathrm{P}(y|\boldsymbol{x}^*,\boldsymbol{\theta})]] \\
&\approx \mathcal{H}[\mathrm{P}(y|\boldsymbol{x}^*,\mathcal{D})] - \mathrm{E}_{\mathrm{q}_\theta^*(\boldsymbol{\theta})}[\mathcal{H}[\mathrm{P}(y|\boldsymbol{x}^*,\boldsymbol{\theta})]] \\
&\approx \mathcal{H}[\mathrm{P}(y|\boldsymbol{x}^*,\mathcal{D})] - \frac{1}{N} \sum_{i}^{N} \mathcal{H}[\mathrm{P}(y|\boldsymbol{x}^*,\boldsymbol{\theta}^{(i)})]
\end{aligned}
$$

C.4   Mutual Information for Dirichlet Prior Network

The mutual information between the labels y and the categorical $\boldsymbol{\mu}$ for a DPN can be calculated as follows, using the fact that MI is the difference of the entropy of the expected distribution and the expected entropy of the distribution.

$$
\begin{aligned}
\mathcal{I}[y,\boldsymbol{\mu}|\boldsymbol{x}^*,\hat{\boldsymbol{\theta}}] &= \mathcal{H}[\mathrm{P}(y|\boldsymbol{x}^*,\hat{\boldsymbol{\theta}})] - \mathrm{E}_{\mathrm{p}(\boldsymbol{\mu}|\boldsymbol{x}^*,\hat{\boldsymbol{\theta}})}[\mathcal{H}[\mathrm{P}(y|\boldsymbol{\mu})]] \\
&= \mathcal{H}[\mathrm{P}(y|\boldsymbol{x}^*,\hat{\boldsymbol{\theta}})] + \sum_{c}^{K} \mathrm{E}_{\mathrm{p}(\boldsymbol{\mu}|\boldsymbol{x}^*,\hat{\boldsymbol{\theta}})}[\mu_c \ln \mu_c] \\
&= -\sum_{c}^{K} \frac{\alpha_c}{\alpha_0} \Big(\ln \frac{\alpha_c}{\alpha_0} - \psi(\alpha_c + 1) + \psi(\alpha_0 + 1) \Big)
\end{aligned}
$$

The second term in this derivation is a non-standard result. The expected entropy of the distribution can be calculated in the following way:

$$
\begin{aligned}
\mathrm{E}_{\mathrm{p}(\boldsymbol{\mu}|\boldsymbol{x}^*,\hat{\boldsymbol{\theta}})}[\mu_c \ln(\mu_c)] &= \frac{\Gamma(\alpha_0)}{\prod_c^K \Gamma(\alpha_c)} \int_{\mathcal{S}_K} \mu_c \ln(\mu_c) \prod_{c=1}^K \mu_c^{\alpha_c-1} d\boldsymbol{\mu} \\
&= \frac{\alpha_c}{\alpha_0} \frac{\Gamma(\alpha_0+1)}{\Gamma(\alpha_c+1)\prod_{c',\neq c}^K \Gamma(\alpha_{c'})} \int_{\mathcal{S}_K} \mu_c^{\alpha_c} \ln(\mu_c) \prod_{c'=1,\neq c}^K \mu_{c'}^{\alpha_{c'}-1} d\boldsymbol{\mu} \\
&= \frac{\alpha_c}{\alpha_0}(\psi(\alpha_c+1) - \psi(\alpha_0+1))
\end{aligned}
$$

Here the expectation is calculated by noting that the standard result of the expectation of $\ln \mu_c$ wrt a Dirichlet distribution can be used if the extra factor μ_c is accounted for by adding 1 to the associated concentration parameter α_c and multiplying by $\frac{\alpha_c}{\alpha_0}$ in order to have the correct normalizing constant.

$$\sum_c^K \mathrm{E}[\mu_c \ln(\mu_c)] = \frac{1}{Z(\boldsymbol{\alpha})} \int_{\mathcal{S}_K} \mu_c \ln(\mu_c) \prod_{c=1}^K \mu_c^{\alpha_c-1} d\boldsymbol{\mu}$$

$$= \frac{1}{Z(\boldsymbol{\alpha})} \int_0^\infty \cdots \int_0^\infty \mu_c \ln(\mu_c) \prod_{c=1}^K \mu_c^{\alpha_c-1} \delta(1 - \sum_{k=1}^K \mu_k) d\mu_1 \cdots d\mu_K$$

$$= \frac{1}{Z(\boldsymbol{\alpha})} \int_0^\infty \cdots \int_0^\infty \mu_c \ln(\mu_c) \prod_{c=1}^K \mu_c^{\alpha_c-1} \frac{1}{2\pi} \int_{-\infty}^\infty e^{-i\theta(1-\sum_{k=1}^K \mu_k))} d\mu_1 \cdots d\mu_K d\theta$$

$$= \frac{1}{Z(\boldsymbol{\alpha})} \frac{1}{2\pi} \int_{-\infty}^\infty e^{-i\theta} \Big\{ \int_0^\infty \mu_1^{\alpha_1-1} e^{i\theta\mu_1} d\mu_1 \cdots \int_0^\infty \mu_K^{\alpha_K-1} e^{i\theta\mu_K} d\mu_K \int_0^\infty \mu_c^{\alpha_c} \ln(\mu_c) e^{i\theta\mu_c} d\mu_c \Big\} d\theta$$

$$= \frac{1}{Z(\boldsymbol{\alpha})} \frac{1}{2\pi i} \int_{-i\infty}^{i\infty} e^{\kappa t} \Big\{ \int_0^\infty \mu_1^{\alpha_1-1} e^{-\kappa\mu_1} d\mu_1 \cdots \int_0^\infty \mu_K^{\alpha_K-1} e^{-\kappa\mu_K} d\mu_K \int_0^\infty \mu_c^{\alpha_C} \ln(\mu_c) e^{-\kappa\mu_c} d\mu_c \Big\} d\kappa$$

$$= \frac{1}{Z(\boldsymbol{\alpha})} \frac{1}{2\pi i} \int_{-i\infty}^{i\infty} e^{\kappa t} \Big\{ \mathcal{L}\{\mu_1^{\alpha_1-1}\} \cdots \mathcal{L}\{\mu_K^{\alpha_K-1}\} \int_0^\infty \mu_c^{\alpha} \ln(\mu_c) e^{-\kappa\mu_c} d\mu_c \Big\} d\kappa$$

$$= \frac{1}{Z(\boldsymbol{\alpha})} \frac{1}{2\pi i} \int_{-i\infty}^{i\infty} e^{\kappa t} \Big\{ \mathcal{L}\{\mu_1^{\alpha_1-1}\} \cdots \mathcal{L}\{\mu_K^{\alpha_K-1}\} \int_0^\infty \frac{u^{\alpha_c}}{\kappa^{\alpha_c+1}} \ln(\frac{u}{\kappa}) e^{-u} du \Big\} d\kappa$$

$$= \frac{\Gamma(\alpha_0)}{\Gamma(\alpha_c)} \frac{1}{2\pi i} \int_{-i\infty}^{i\infty} e^{\kappa t} \Big\{ \frac{1}{\kappa^{\alpha_0-\alpha_c}} \int_0^\infty \frac{u^{\alpha_c}}{\kappa^{\alpha_c+1}} \ln(\frac{u}{\kappa}) e^{-u} du \Big\} d\kappa$$

$$= \frac{\Gamma(\alpha_0)}{\Gamma(\alpha_c)} \frac{1}{2\pi i} \int_{-i\infty}^{i\infty} e^{\kappa t} \Big\{ \frac{1}{\kappa^{\alpha_0+1}} \int_0^\infty u^{\alpha_c} \ln(\frac{u}{\kappa}) e^{-u} du \Big\} d\kappa$$

$$= \frac{\Gamma(\alpha_0)}{\Gamma(\alpha_c)} \frac{1}{2\pi i} \int_{-i\infty}^{i\infty} e^{\kappa t} \Big\{ \frac{1}{\kappa^{\alpha_0+1}} \Big[\int_0^\infty u^{\alpha_c} \ln(u) e^{-u} du - \int_0^\infty u^{\alpha_c} \ln(\kappa) e^{-u} du \Big] \Big\} d\kappa$$

$$= \frac{\Gamma(\alpha_0)}{\Gamma(\alpha_c)} \frac{1}{2\pi i} \int_{-i\infty}^{i\infty} e^{\kappa t} \Big\{ \frac{1}{\kappa^{\alpha_0+1}} \Big[\int_0^\infty u^{\alpha_c} \ln(u) e^{-u} du - \alpha_c \Gamma(\alpha_c) \ln(\kappa) \Big] \Big\} d\kappa$$

$$= \frac{\Gamma(\alpha_0)}{\Gamma(\alpha_c)} \frac{1}{2\pi i} \int_{-i\infty}^{i\infty} e^{\kappa t} \Big\{ \frac{1}{\kappa^{\alpha_0+1}} \Big[\alpha_c \Gamma(\alpha_c) \psi(\alpha_c) + \Gamma(\alpha_c) - \alpha_c \Gamma(\alpha_c) \ln(\kappa) \Big] \Big\} d\kappa$$

$$= \alpha_c \Gamma(\alpha_0) \frac{1}{2\pi i} \int_{-i\infty}^{i\infty} e^{\kappa t} \Big\{ \frac{1}{\kappa^{\alpha_0+1}} \Big[\psi(\alpha_c) + \frac{1}{\alpha_c} - \ln(\kappa) \Big] \Big\} d\kappa$$

$$= \alpha_c \Gamma(\alpha_0) \mathcal{L}^{-1} \Big\{ \frac{1}{\kappa^{\alpha_0+1}} \Big[\psi(\alpha_c) + \frac{1}{\alpha_c} - \ln(\kappa) \Big] \Big\}$$

$$= \alpha_c \Gamma(\alpha_0) \Big\{ \mathcal{L}^{-1} \Big\{ \frac{1}{\kappa^{\alpha_0+1}} \Big[\psi(\alpha_c) + \frac{1}{\alpha_c} \Big] \Big\} + \mathcal{L}^{-1} \Big\{ \frac{1}{\kappa^{\alpha_0+1}} \Big[-\ln(\kappa) \Big] \Big\} \Big\}$$

$$= \frac{\alpha_c}{\alpha_0} (\psi(\alpha_c) + \frac{1}{\alpha_c}) + \alpha_c \Gamma(\alpha_0) \mathcal{L}^{-1} \Big\{ \frac{1}{\kappa^{\alpha_0+1}} \Big[-\ln(\kappa) \Big] \Big\}$$

$$= \frac{\alpha_c}{\alpha_0} (\psi(\alpha_c) + \frac{1}{\alpha_c}) + \alpha_c \Gamma(\alpha_0) \mathcal{L}^{-1} \Big\{ \frac{1}{\kappa^{\alpha_0}} \Big(\frac{\gamma - \gamma - \ln(\kappa)}{\kappa} \Big) \Big\}$$

$$= \frac{\alpha_c}{\alpha_0} (\psi(\alpha_c) + \frac{1}{\alpha_c}) + \alpha_c \Gamma(\alpha_0) \mathcal{L}^{-1} \Big\{ \frac{\gamma}{\kappa^{\alpha_0+1}} - \frac{1}{\kappa^{\alpha_0}} \Big(\frac{\gamma + \ln(\kappa)}{\kappa} \Big) \Big\}$$

$$= \frac{\alpha_c}{\alpha_0} (\psi(\alpha_c) + \frac{1}{\alpha_c} + \gamma) + \alpha_c \Gamma(\alpha_0) \mathcal{L}^{-1} \Big\{ -\frac{1}{\kappa^{\alpha_0}} \Big(\frac{\gamma + \ln(\kappa)}{\kappa} \Big) \Big\}$$

$$= \frac{\alpha_c}{\alpha_0} (\psi(\alpha_c) + \frac{1}{\alpha_c} + \gamma) + \alpha_c \int_0^t \ln(t - \tau) \tau^{\alpha_0-1} d\tau$$

$$= \frac{\alpha_c}{\alpha_0} (\psi(\alpha_c + 1) - \psi(\alpha_0 + 1))$$

$$\Gamma(\alpha + 1) = \alpha\Gamma(\alpha)$$

$$\frac{d}{d\alpha}\left\{\int_0^\infty \mu^\alpha e^{-\mu}d\mu\right\} = \frac{d}{d\alpha}\left\{\alpha\int_0^\infty \mu^{\alpha-1}e^{-\mu}d\mu\right\}$$

$$\int_0^\infty \mu^\alpha \ln(\mu)e^{-\mu}d\mu = \alpha\int_0^\infty \mu^{\alpha-1}\ln(\mu)e^{-\mu}d\mu + \int_0^\infty \mu^{\alpha-1}e^{-\mu}d\mu$$

$$= \alpha\Gamma(\alpha)\psi(\alpha) + \Gamma(\alpha)$$