Uncertainty-Aware Deep Classifiers Using Generative Models

Murat Sensoy,[1,2] **Lance Kaplan,**[3] **Federico Cerutti,**[4,5] **Maryam Saleki**[2]
[1]Blue Prism AI Labs, London, UK
[2]Department of Computer Science, Ozyegin University, Istanbul, Turkey
[3]US Army Research Lab, Adelphi, MD 20783, USA
[4]Department of Information Engineering, University of Brescia, 25123 Brescia, Italy
[5]Cardiff University, Cardiff, CF10 3AT, UK
murat.sensoy@blueprism.com, lkaplan@ieee.org, federico.cerutti@unibs.it, maryam.saleki@ozu.edu.tr

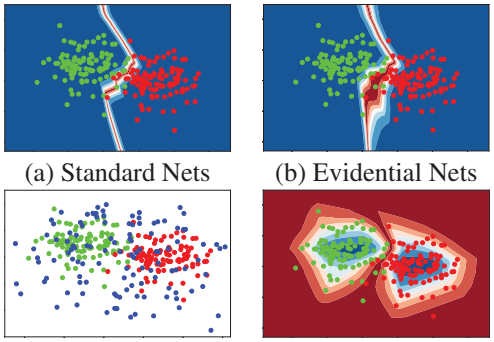
Abstract

Deep neural networks are often ignorant about what they do not know and overconfident when they make uninformed predictions. Some recent approaches quantify classification uncertainty directly by training the model to output high uncertainty for the data samples close to class boundaries or from the outside of the training distribution. These approaches use an auxiliary data set during training to represent out-of-distribution samples. However, selection or creation of such an auxiliary data set is non-trivial, especially for high dimensional data such as images. In this work we develop a novel neural network model that is able to express both aleatoric and epistemic uncertainty to distinguish decision boundary and out-of-distribution regions of the feature space. To this end, variational autoencoders and generative adversarial networks are incorporated to automatically generate out-of-distribution exemplars for training. Through extensive analysis, we demonstrate that the proposed approach provides better estimates of uncertainty for in- and out-of-distribution samples, and adversarial examples on well-known data sets against state-of-the-art approaches including recent Bayesian approaches for neural networks and anomaly detection methods.

Introduction

While deep learning models demonstrate remarkable generalization performance in light of the large number of parameters they exploit, they can be misleadingly overconfident when they do make mistakes. The false sense of trust these models create may have serious consequences, especially if they are used for high-risk tasks. A striking example is the misclassification of the white side of a trailer as bright sky: this caused a car operating with automated vehicle control systems to crash against a tractor-semitrailer truck near Williston, Florida, USA on 7th May 2016. The car driver died due to the sustained injury (NHTSA 2016).

There are two categories of uncertainty (Matthies 2007). *Epistemic uncertainty*, or *model uncertainty*, results from limited knowledge and could in principle be reduced: uncertain predictions for out-of-distribution samples fall into this category. Among other approaches, Bayesian deep learning methods try to estimate epistemic uncertainty by modeling the distributions for the parameters values, distributions that seldom admit closed-form representations, hence requiring expensive Monte Carlo sampling methods (Bishop 2006).



(a) Standard Nets (b) Evidential Nets (c) Generated Points (d) Proposed Model

Figure 1: Class boundaries for models on a simple 2D classification problem (green vs red dots): prediction confidence depicts on a color scale from maroon (low confidence) to blue (high confidence). The generated samples are shown as blue dots in (c) along with the original data points.

Aleatoric uncertainty, or *data uncertainty*, is the noise inherent in the observations (e.g., label noise), or class overlap: unlike epistemic uncertainty, aleatoric uncertainty cannot be reduced by observing more data samples. For instance, having identical samples with different labels, e.g., on the class boundary, is an example of aleatoric uncertainty. Approaches such as Evidential Neural Networks (EDL) and Lightweight Probabilistic Deep Networks, are proposed recently to estimate aleatoric uncertainties in deep neural networks by directly estimating parameters of the predictive posterior as their output (Sensoy, Kaplan, and Kandemir 2018; Gast and Roth 2018). These approaches do not require any sampling and assume minimal changes to the architecture of standard neural networks.

Aforementioned approaches may still make misleading and overconfident predictions for the samples out of the training distribution. Figures 1(a), (b), and (d) demonstrate predicted class boundaries for a simple 2D classification problem of green vs red dots. Standard deterministic neural networks (Fig. 1(a)) do not decrease their prediction confidence when classifying samples around the class boundary, while EDL

(Fig. 1(b)) does. However, both models have high prediction confidence when tested with out-of-distribution samples. To avoid such overconfident predictions, other approaches such as (Malinin and Gales 2018) propose to hand-pick an auxiliary data set as the out-of-distribution samples and explicitly train the neural networks to give highly uncertain output for them. This is often infeasible in high-dimensional real-life settings, given the very large space of possibilities.

In this paper, we propose a deterministic neural network that can effectively and efficiently estimate classification uncertainty for both in- and out-of-distribution samples. Using a generative model, it synthesizes out-of-distribution samples close to the training samples, e.g., the blue dots in Fig. 1 (c). Then, it trains a classifier using both training and the generated samples. Figure 1(d) depicts the result of our approach on our toy example: our model shows higher prediction confidence only for regions close to the training samples.

Our contribution in the paper is threefold. (1) We consider the output of the neural network as the parameters of a Dirichlet distribution with uniform prior, instead of a categorical distribution over possible labels. (2) These parameters are calculated by learning an implicit density estimation for each category by treating each output of the network as an output of a binary classifier, which learns to discriminate samples of the category from the samples of other categories and out-of-distribution samples. (3) We also propose a novel generative adversarial network, which learns to distort the training samples to automatically generate the most informative out-of-distribution samples during training, and so overcoming the need to hand-pick an auxiliary data set as the out-of-distribution samples.

Through extensive experiments, we compare our model not only with state-of-the-art Bayesian networks and other models for uncertainty estimation, but also with recent anomaly detection models, which are specifically designed to determine out-of-distribution samples using deep neural networks. Our experiments on MNIST and CIFAR10 data sets and adversarial examples indicate that our approach outperforms the existing approaches significantly in these tasks.

Generative Evidential Neural Networks

Our work can be considered as an extension of approaches for classification, which take the output of a neural network for an input sample to estimate parameters of a Dirichlet distribution (Sensoy, Kaplan, and Kandemir 2018; Malinin and Gales 2018; Gast and Roth 2018) for its classification. That is, the resulting Dirichlet distribution represents the likelihood of each possible categorical distribution over the labels for the classification of the sample.

More formally, the Dirichlet distribution is a probability density function (pdf) for possible values of the probability mass function (pmf) $\mathbf{p}$. It is characterized by K parameters $\boldsymbol{\alpha} = [\alpha_1, \cdots, \alpha_K]$ and is given by

$$D(\mathbf{p}|\boldsymbol{\alpha}) = \begin{cases} \frac{1}{B(\boldsymbol{\alpha})} \prod_{i=1}^{K} p_i^{\alpha_i - 1} & \text{for } \mathbf{p} \in \mathcal{S}_K, \\ 0 & \text{otherwise,} \end{cases} \tag{1}$$

where $\mathcal{S}_K$ is the K-dimensional unit simplex and $B(\boldsymbol{\alpha})$ is the K-dimensional multinomial beta function (Kotz, Balakrishnan, and Johnson 2000).

In classical neural networks for classification, softmax function is used to predict class assignment probabilities. However, it provides only a point estimate for the class probabilities of a sample and does not provide the associated uncertainty for this prediction. On the other hand, Dirichlet distributions can be used to model a probability distribution for the class probabilities. For instance, a Dirichlet distribution whose all parameters are one, i.e., $D(\mathbf{p}|\langle 1, \ldots, 1\rangle)$ or shortly $D(\boldsymbol{p}|\mathbf{1})$, represents the uniform distribution over all possible assignment of class probabilities and means total uncertainty for the classification of a sample. As the parameter referring to specific class increases, the likelihood of probability assignments with higher values for this class also increases. For instance, $D(\mathbf{p}|\langle 2, \ldots, 1\rangle)$ indicates that probability distributions placing more mass on the first class are slightly more likely, while their likelihood increases further for $D(\mathbf{p}|\langle 10, \ldots, 1\rangle)$, which indicates that 8 more pieces of evidence is observed for the assignment of the sample to the first class (Josang 2016).

The parameters of a Dirichlet distribution are associated with pseudocounts representing the number of observations or evidence in each class. Hence, the predicted Dirichlet distribution for a sample may refer to the amount of evidence observed on the training set for the assignment of the sample to classes. If there is no evidence for the assignment, we consider a uniform prior, i.e., $D(\mathbf{p}|\langle 1, \ldots, 1\rangle)$ and any evidence e_i for class i should add up to the relevant parameter of this prior (i.e., $\alpha_i = 1 + e_i$) to generate the predicted Dirichlet distribution for the sample. The mean and the variance of a Dirichlet distribution for the class probability p_k are computed as

$$\hat{p}_k = \frac{\alpha_k}{S} \text{ and } Var(p_k) = \frac{\alpha_k(S - \alpha_k)}{S^2(S+1)}, \tag{2}$$

where $S = \sum_{i=1}^{K} \alpha_k$. The Dirichlet distribution after incorporation of a number of evidence, $\hat{\mathbf{p}}$ represents the minimum mean square error (mmse) estimate of the ground truth appearance probabilities given these observations.

In this paper, we consider Dirichlet distributions with uniform prior, which means that $S \geq K$. Then, the total evidence used to update the uniform Dirichlet to the predicted Dirichlet distribution becomes $S - K$. After predicting the parameters of the Dirichlet distribution for each sample, previous approaches have used its mean, i.e., $\hat{\mathbf{p}}$, as the class assignment probabilities for decision making, while using $K/S \in [0-1]$ as the associated uncertainty of this assignment (Sensoy, Kaplan, and Kandemir 2018). As the total evidence increases, the variance of the Dirichlet distribution decreases, and so does the uncertainty of prediction.

In this work, we also use the mean of Dirichlet distributions as the predictive categorical distribution for classification. However, to be consistent with literature in general and for benchmark comparisons, we adopt the entropy of class probabilities as a proxy for classification uncertainty (Gal and Ghahramani 2016b; Louizos and Welling 2017).

Learning to Quantify Classification uncertainty

Recent approaches also used Dirichlet distribution to quantify classification uncertainty in deep neural nets. However,

they failed to link the calculated Dirichlet parameters to the observations or the evidence derived from the distribution of the training set. That is why these models could still be able to derive large amount of evidence and become overconfident in their predictions for out-of-distribution samples.

We use ideas from implicit density models (Mohamed and Lakshminarayanan 2016) and noise-contrastive estimation (NCE) (Gutmann and Hyvärinen 2012) to derive Dirichlet parameters for samples. Let us consider a classification problem with K classes and assume P_{in}, P_k, and P_{out} represent respectively the data distributions of the training set, class k, and out-of-distribution samples, i.e., the samples that do not belong to any of K classes. A convenient way to describe density of samples from a class k is to describe it relative to the density of some other reference data. By using the same reference data for all classes in the training set, we desire to get comparable quantities for their density estimations. In NCE, noisy training data (Hafner et al. 2018) is usually used as reference, here, we generalize this as the out-of-distribution samples, which may also include the noisy data.

Using the dummy labels y, we can reformulate the ratio of the densities $P_k(\boldsymbol{x})$ and $P_{out}(\boldsymbol{x})$ for a sample $\boldsymbol{x}$ as follows:

$$\frac{P_k(\boldsymbol{x})}{P_{out}(\boldsymbol{x})} = \frac{p(\boldsymbol{x}|y=k)}{p(\boldsymbol{x}|y=out)} = \frac{p(y=k|\boldsymbol{x})}{p(y=out|\boldsymbol{x})}\left(\frac{1-\pi_k}{\pi_k}\right) \tag{3}$$

where π_k is the marginal probability $p(y=k)$ and $(1-\pi_k)/\pi_k$ can be approximated as the ratio of sample size, i.e., n_k/n_{out}, which is taken as one in this work for simplicity without loss of generality.

As shown in Eq. 3, one can approximate the log density ratio $\log\big(P_k(\boldsymbol{x})/P_{out}(\boldsymbol{x})\big)$ as the logit output of a binary classifier (Mohamed and Lakshminarayanan 2016), which is trained to discriminate between the samples from P_k and P_{out}. Let a neural network classifier $\boldsymbol{f}(\boldsymbol{x}|\theta)$ parameterized by weights θ have K outputs for a given sample $\boldsymbol{x}$, where each output $f_k(\boldsymbol{x}|\theta)$ corresponds to a logit for one of K classes and approximates $\log\big(P_k(\boldsymbol{x})/P_{out}(\boldsymbol{x})\big)$. To train such a network, we use the Bernoulli (logarithmic) loss as given by

$$\mathcal{L}_1(\theta) = -\sum_{k=1}^{K}\Big[\mathop{\mathbb{E}}_{P_k(\boldsymbol{x})}[log(\sigma(f_k(\boldsymbol{x}|\theta)))] + \mathop{\mathbb{E}}_{P_{out}(\boldsymbol{x})}\big[log(1-\sigma(f_k(\boldsymbol{x}|\theta)))\big]\Big]. \tag{4}$$

The expectations in Eq. 4 are computed by Monte Carlo integration using the equal number of samples from P_k and P_{out}. In this work, we use samples from P_{out}, which are generated by perturbing training set using a novel generative adversarial network. The generated samples are separable from the training samples in high dimensional input space, so they are out of distribution, while still having many similarities to the training samples in a lower dimensional representation space, which is learned to reconstruct training samples.

As a result, $exp(f_k(\boldsymbol{x}|\theta))$ approximates the relative density $P_k(\boldsymbol{x})/P_{out}(\boldsymbol{x})$ of class k, as it is trained using the samples from class k and the samples close to, but easily differentiable from, the samples belonging to all K classes in the training set. For each sample $\boldsymbol{x}$ in the training set, we take $\boldsymbol{e} = exp(\boldsymbol{f}(\boldsymbol{x}|\theta))$ as the pseudocounts (i.e., evidence) vector, where each element $e_k = exp(f_k(\boldsymbol{x}|\theta))$ is the pseudocount for $\boldsymbol{x}$ being assigned to class k. Then, the parameters of the Dirichlet distribution given the uniform Dirichlet prior is calculated as $\boldsymbol{\alpha} = \boldsymbol{e} + \mathbf{1}$. If the sample x is more similar to the samples from P_{out}, then almost zero evidence is generated by the neural network and the predicted Dirichlet distribution becomes very close to the uniform Dirichlet distribution. This should be the case for samples from P_{out} and for the outliers in the training set. On the other hand, if the sample is labelled as k in the training set and it is not an outlier, we expect $e_k > e_j \geq 0$ for any $j \neq k$.

Uncertainty for Misclassified Samples

The computed $\boldsymbol{\alpha}$ parameters defines a Dirichlet distribution $D(\boldsymbol{p}|\boldsymbol{\alpha})$, from which one can sample categorical distributions over possible classes of $\boldsymbol{x}$. However, only one of these classes is correct and assignment of $\boldsymbol{x}$ to any other class is considered as a misclassification. If k is the true class of $\boldsymbol{x}$, then the marginal distribution for p_k, i.e., the probability of correctly classifying $\boldsymbol{x}$, is a two-parameter Dirichlet distribution (also known as Beta distribution) with parameters $\langle \alpha_k, \sum_{j\neq k}\alpha_j \rangle$. Let $\boldsymbol{p}_{-k}$ refer to the vector of probabilities p_j such that $j \neq k$. Probabilities of misclassifying $\boldsymbol{x}$ to each class other than the true one is distributed based on the conditional Dirichlet distribution $\boldsymbol{p}'_{-k}|p_k \sim D(\boldsymbol{p}'_{-k}|\boldsymbol{\alpha}_{-k})$, where $\boldsymbol{p}'_{-k}$ is a categorical distribution over misclassified classes and created by normalizing $\boldsymbol{p}_{-k}$ with $(1-p_k)$, which is also equivalent to $\sum_{i\neq k} p_i$. Since we desire a classifier to be totally uncertain in its misclassifications (except near decision boundaries), we minimize Kullback–Leibler ($\mathbb{KL}$) divergence between $D(\boldsymbol{p}'_{-k}|\boldsymbol{\alpha}_{-i})$ and the uniform Dirichlet distribution $D(\boldsymbol{p}_{-k}|\mathbf{1})$ using the following regularizer:

$$\mathcal{L}_2(\theta|\boldsymbol{x}) = \beta\mathbb{KL}[D(\boldsymbol{p}_{-k}|\boldsymbol{\alpha}_{-k}) \;||\; D(\boldsymbol{p}_{-k}|\mathbf{1})], \tag{5}$$

where β is the weight of the $\mathbb{KL}$ term. It can be set to $(1-\hat{p}_k)$, which is the expectation for the probability of misclassification (i.e., $1-p_k$) and its usage as a weight of the $\mathbb{KL}$ term enables the learned loss attenuation; that is, it places a higher weight on epistemic uncertainty enforcement as the aleatoric uncertainty for misclassification decreases. Then, the generative evidential neural network learns the parameters θ by minimizing the overall loss defined as

$$\mathcal{L}(\theta) = \mathcal{L}_1(\theta) + \mathop{\mathbb{E}}_{P_{in}(\boldsymbol{x})}[\mathcal{L}_2(\theta|\boldsymbol{x})]. \tag{6}$$

Generating out-of-distribution samples

Previous approaches generate out-of-distribution samples by perturbing training samples. However, they usually require manual determination of how much perturbation should be made to these samples. Too little perturbation may not set the resulting samples apart from the actual samples, while too much perturbation may cast them far away from the training data and deteriorate their usefulness.

Variational Autoencoders (VAE) are probabilistic generative models that create low dimensional latent representations for high dimensional data by maximizing

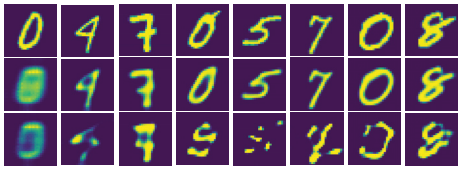


Figure 2: Original training samples (top), samples reconstructed by the VAE (middle), and the samples generated by the proposed method (bottom) over a number of epochs.

$$\max_{q_\theta, p_\phi} \sum_{i=1}^{N} \mathbb{E}_{q_\theta(\boldsymbol{z}|\boldsymbol{x}_i)} \big[\log p_\phi(\boldsymbol{x}_i \mid \boldsymbol{z}) - \mathbb{KL}(q_\theta(\boldsymbol{z} \mid \boldsymbol{x}_i) \,||\, p(\boldsymbol{z})) \big], \tag{7}$$

where $q_\theta(\boldsymbol{z} \mid \boldsymbol{x}_i)$ is the latent space distribution for each sample $\boldsymbol{x_i}$ and $p_\phi(\boldsymbol{x}_i \mid \boldsymbol{z})$ is the decoder likelihood distribution that is maximized for each sample $\boldsymbol{x_i}$. The $\mathbb{KL}$ term enforces $q_\theta(\boldsymbol{z} \mid \boldsymbol{x}_i)$ to be close to a prior distribution $p(\boldsymbol{z})$ and have a denser latent space.

Proximity of the encoded samples in the latent space of a VAE is commonly used as an indication of their semantic similarity and exploited for few-shot classification and anomaly detection tasks. In this work, we also use the latent space of a VAE as a proxy for semantic similarity between samples in input space. Hence, we exploit it to generate out-of-distribution samples, which are similar to, but at the same time clearly separable from, the training examples in the input space.

For each $\boldsymbol{x}_i$ in training set, we sample a latent point $\boldsymbol{z}$ from $q_\theta(\boldsymbol{z} \mid \boldsymbol{x}_i)$ and perturb it by $\boldsymbol{\epsilon} \sim q_\gamma(\boldsymbol{\epsilon}|\boldsymbol{z})$, which is implemented as a multivariate Gaussian distribution $\mathcal{N}(\mathbf{0}, G(\boldsymbol{z}))$, where $G(\cdot)$ is a fully connected neural network with non-negative output that is trained via

$$\max_{G} \; \mathbb{E}_{\substack{q_\theta(\boldsymbol{z}|\boldsymbol{x}_i),\\ q_\gamma(\boldsymbol{\epsilon}|\boldsymbol{z}),\\ p_\phi(\bar{\boldsymbol{x}}_i|\boldsymbol{z}+\boldsymbol{\epsilon})}} \big[\underbrace{\log D'(\boldsymbol{z}+\boldsymbol{\epsilon})}_{(a)} + \underbrace{\log(1 - D(\bar{\boldsymbol{x}}_i))}_{(b)} \big], \tag{8}$$

where $\bar{\boldsymbol{x}}_i \sim p_\phi(\bar{\boldsymbol{x}}_i \mid \boldsymbol{z}+\boldsymbol{\epsilon})$ is the decoded out-of-distribution sample from the perturbed sample $\boldsymbol{z}+\boldsymbol{\epsilon}$. The discriminators D and D' are binary classifiers with *sigmoid* output that try to distinguish real samples from the generated ones. That is, given an input, a discriminator gives as an output the probability that the sample is from the training set distribution. In Eq. 8, (a) forces the generated points to be similar to the real latent points through making them indistinguishable by D' in the latent space of the VAE and (b) encourages the generated samples to be distinguishable by D in the input space. The discriminators are optimized via

$$\max_{D'} \quad \log D'(\boldsymbol{z}) + \underbrace{\log(1 - D'(\boldsymbol{z}+\boldsymbol{\epsilon}))}_{(c)}, \tag{9}$$

$$\max_{D} \quad \log D(\boldsymbol{x}_i) + \log(1 - D(\bar{\boldsymbol{x}}_i)). \tag{10}$$

Note that (c) of Eq. 9 is also included in the objective of the VAE (Eq. 8) to adapt the latent space during the training of the generator. We trained the VAE, generator, and discriminators by iterating between maximizing Eq. 7 through Eq. 10 until convergence, as in the regular training of generator and discriminator in GANs. We demonstrate this approach in Fig 1 (c) and Fig. 2, where a number of real and generated MNIST images are shown.

	Layer	Filters/Neurons	Patch Size	Stride	Activation
Classifier	Conv_1	20	5×5	1	relu
	Max Pool	-	2×2	2	-
	Conv_2	50	5×5	1	relu
	Max Pool	-	2×2	2	-
	FC_1	500	-	-	relu
	FC_2	$K = 10$	-	-	-
D	$\text{Conv}_1 - \text{FC}_1$	repeat	repeat	repeat	repeat
	FC_3	1	-	-	sigmoid
G	FC_4	32	-	-	relu
	FC_5	32	-	-	relu
	FC_6	32	-	-	relu
	FC_7	$code\_sz = 50$	-	-	softplus
D'	$\text{FC}_4 - \text{FC}_6$	repeat	repeat	repeat	repeat
	FC_8	1	-	-	sigmoid

Table 1: Network architectures.

Model	MNIST	CIFAR 5
L2	99.4	76
Dropout	99.5	84
Deep Ensemble	99.3	79
FFGU	99.1	78
FFLU	99.1	77
MNFG	99.3	84
BBH	99.1	80
EDL	99.25	82
GEN	99.3	83

Table 2: Test accuracies (%) for MNIST and CIFAR5.

Evaluation

To be able to compare our approach with the recent work, we adopted the same strategy used for evaluation in (Louizos and Welling 2017; Sensoy, Kaplan, and Kandemir 2018; Pawlowski et al. 2017). That is, we use LeNet-5 (LeCun et al. 1998) with ReLu non-linearities and max pooling as the neural network architecture and evaluated our approach with MNIST and CIFAR10 datasets, to be able to make a fair comparison with the most related recent work. We implemented our approach [1] using Python and Tensorflow.

In this section, we compared our approach with the following approaches: (a) **L2** corresponds to the standard neural nets with softmax probabilities and L2 regularization, (b) **Dropout** refers to the Bayesian model used in (Gal and Ghahramani 2016a), (c) **Deep Ensemble** refers to the model proposed in (Lakshminarayanan, Pritzel, and Blundell 2017), (d) **FFG** refers to the Bayesian model used in (Kingma, Salimans, and Welling 2015) with the additive parametrization (Molchanov, Ashukha, and Vetrov 2017), (e) **MNFG**[2] refers to the variational approximation based model in (Louizos and Welling 2017), (f) **EDL** refers to the model in (Sensoy, Kaplan, and Kandemir 2018), (g) **BBH**[3] refers to the Bayesian model based on implicit weight uncertainty (Pawlowski et al. 2017), and (h) **GEN** refers to the proposed approach.

Predictive Uncertainty Estimation

We used the network architectures in Table 1 to train our model for the MNIST dataset. For CIFAR10, we used the same architectures; however, the classifier uses 192 filters for Conv_1 and Conv_2, also has 1000 neurons in FC_1 as described

[1] https://muratsensoy.github.io/gen.html

[2] https://github.com/AMLab-Amsterdam/MNF_VBNN

[3] https://github.com/pawni/BayesByHypernet

in (Louizos and Welling 2017). We used L2 regularization with coefficient 0.005 in the fully-connected layers. Other approaches are also trained using the same classifier architecture with the priors and posteriors described in (Louizos and Welling 2017) and (Pawlowski et al. 2017). The classification accuracy of each model on the MNIST test set can be seen at Table 2. While we do not explicitly aim for high classification accuracy, our results indicate that our approach is doing better than most of the other approaches.

We train models for MNIST using the images from 10 digit categories from the training set as usual. However, we then tested these models on notMNIST dataset,[4] which contains 10 letters A-J instead of digits. For CIFAR10, we trained models using the training data from the first five categories (referred to as CIFAR5) and tested these models using the images from the last five categories. For both MNIST and CIFAR10, the predicted label for any test sample is guaranteed to be wrong, since test samples are coming from a different distribution than the one for the training set. Hence, an ideal classifier should report totally uncertain predictions instead of associating a higher likelihood for a specific label for an out-of-distribution test sample.

To be consistent with recent works, we use the entropy of the predicted categorical distribution over labels as a proxy to quantify prediction uncertainty. That is, a prediction gets more uncertain as its entropy approaches the maximum entropy, i.e., the entropy of the uniform categorical distribution. As in (Louizos and Welling 2017), we used the empirical CDF of entropy distribution for predictions to quantify how uncertain they are. That is, as the predictions get more uncertain, the area under their entropy CDF curve gets smaller.

Figure 3 shows our results for MNIST and CIFAR10 datasets. Standard neural networks (referred to as L2) is very confident in its predictions as indicated by its entropy CDF curves in the figure. On the other hand, Bayesian neural network models appear to be more uncertain about their predictions with respect to the standard neural networks. The performances of these models in terms of predictive uncertainty vary for MNIST while they perform almost the same for CIFAR10. EDL and GEN perform much better than Bayesian approaches in both MNIST and CIFAR10. Especially, GEN associate very high uncertainty with its predictions for out-of-distribution samples.

After conducting these benchmark analysis by following the very same procedure proposed in (Louizos and Welling 2017), we also tested these models with in-distribution samples and analyzed the certainty they assign to correct and incorrect predictions. To be more comprehensive and diverse, we also included *Bayes by Backprop* (**BBB**) proposed in (Blundell et al. 2015), which is frequently considered as a baseline for Bayesian neural network research. Figure 4 shows entropy CDF curves for successful and failed predictions in MNIST test set for different models. The figure indicates that standard networks and Bayesian neural networks are overconfident (i.e., have low entropy) for their failed predictions; that is, they have large area under the entropy CDF curve for their failed predictions (i.e., misclassifications). However, both EDL and GEN have significantly higher predictive uncertainty for their failed predictions. Furthermore, GEN gives a better disparity between the successful and failed predictions in terms of uncertainty. We also conducted the same analysis for the CIFAR10 dataset and obtained similar results.

Robustness to Adversarial Examples

Robustness to adversarial examples is an important challenge for machine learning models. While it is very hard to provide correct predictions for carefully crafted adversarial examples, a model should associate very high uncertainty with its prediction when tested on them. Hence, in this section, we test different models using a well-known white-box attack strategy, the Fast Sign Method (FSM), proposed by (Goodfellow, Shlens, and Szegedy 2014) and analyze how uncertain these models are when they fail to correctly classify the generated adversarial examples.

White-box attacks have access to model parameters and exploit gradients of the loss with respect to an input to perturb the input to create an adversarial example. The amount of perturbation is defined by the $\epsilon \in [0, 1]$ parameter. Figure 5 shows our results in terms of both accuracy and uncertainty for the MNIST test set for different ϵ values. The figure indicates that GEN demonstrates the ideal behavior; it associates the highest uncertainty (maximum entropy) with its predictions as it starts to fail making the right predictions for high values of ϵ. We observe the same behavior for CIFAR10 dataset as shown in Figure 6.

Comparisons with Anomaly Detection Methods

Our work is also related to anomaly detection approaches, which are specifically designed to detect out-of-distribution samples. Hence, in this section, we compare our approach with the existing and the most recent anomaly detection methods on MNIST and CIFAR10 datasets. We compared our approach with the following models: (a) **Calibrated** refers to the calibration-based model for out-of-distribution detection in (Lee et al. 2018), (b) **GEOTRANS** is the anomaly detection model in (Golan and El-Yaniv 2018), (c) **SVM** refers to the one-class SVM applied to the latent space of convolutional autoencoder as described in (Golan and El-Yaniv 2018), (d) **ADGAN** is the anomaly detection method based on generative adversarial networks in (Deecke et al. 2018), (e) **DAGMM** is the deep autoencoding Gaussian mixture model in (Zong et al. 2018), (f) **DSEBM** is the deep structured energy-based model in (Zhai et al. 2016). We use publicly available implementations of the Calibrated[5] and GEOTRANS[6] by their authors, which also contains implementations of other models above. Unlike our approach, these models predicts a *score* for out-of-distribution classification. To evaluate these approaches, the area under the ROC curve (AUC) is used as a measure of how well the produced scores can distinguish between in- and out-of-distribution samples.

Similarly, as before, we train models using samples only from the first five categories of the MNIST and CIFAR10

[4] https://www.kaggle.com/lubaroli/notmnist

[5] https://github.com/alinlab/Confident_classifier

[6] https://github.com/izikgo/AnomalyDetectionTransformations

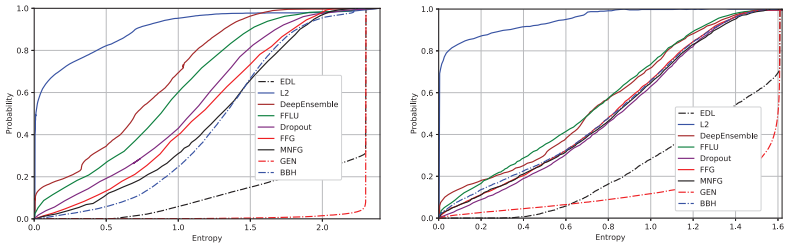


Figure 3: Empirical CDF for the entropy of the predictive distributions on the notMNIST dataset (left) and samples from the last five categories of CIFAR10 dataset (right).

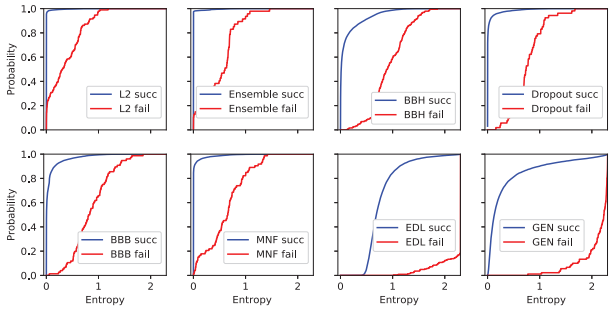


Figure 4: Entropy CDF curves of different models for their successful and failed predictions on the MNIST test set.

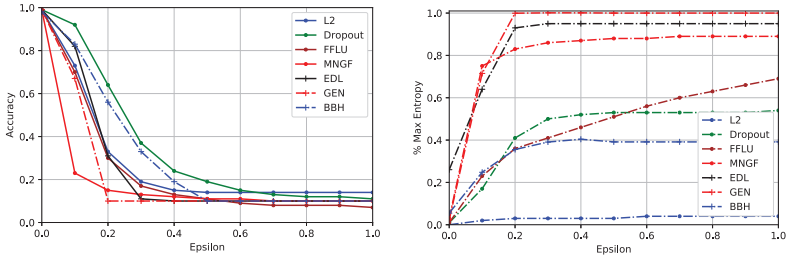


Figure 5: Accuracy and entropy as a function of the adversarial perturbation ϵ on the MNIST dataset.

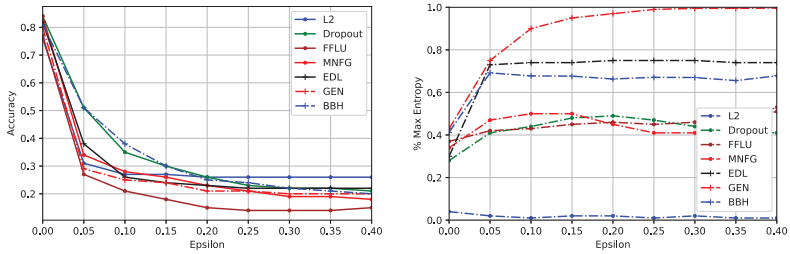


Figure 6: Accuracy and entropy as a function of the adversarial perturbation ϵ on the CIFAR dataset.

datasets. Then, we evaluate these models on the test samples half of which comes from the first five and other half comes from the last five categories. We use the entropy of the predictive probabilities from our model as a *score* to differentiate between in- and out-of-distribution samples. Let us note that, as shown before, our approach provides highly uncertain predictions not only for out-of-distribution samples, but also for the misclassified in-distribution samples. Hence, the in-distribution samples laying on the class boundary of first five categories may also be classified as out-of-distribution samples based on their entropy-based score.

Figure 7 shows our results with anomaly detection methods on MNIST and CIFAR10 datasets. Both Calibrated and GEOTRANS perform better than SVM, ADGAN, DAGMM, and DSEBM in our experiments. For MNIST, GEN and Calibrated achieve the best AUC values, respectively 0.965

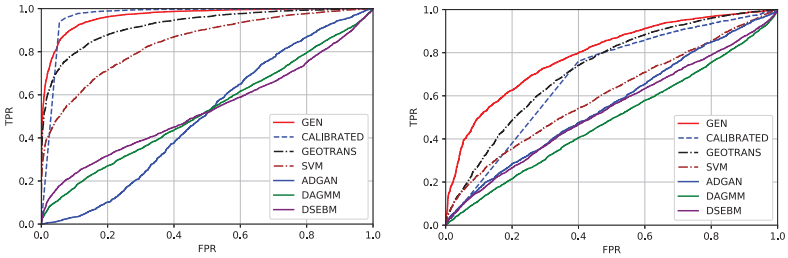


Figure 7: Anomaly detection results for MNIST (left) and CIFAR10 (right).

and 0.966. For CIFAR10, GEN achieves the best AUC (0.775), and performs significantly better than state-of-the-art anomaly detectors in recognizing out-of-distribution samples. While our approach has higher entropy for its predictions for both out-of-distribution samples and misclassified in-distribution samples,[7] its entropy-based score is still performing at least as good as the state-of-the-art anomaly detection methods.

Related Work

Quantification of predictive uncertainty has always been very important for machine learning models. Gaussian Processes (GPs) (Rasmussen and Williams 2006) has been very good both in making accurate predictions and estimate their predictive uncertainties. However, these kernel-based non-parametric models cannot easily deal with high-dimensional data such as images due to the curse of dimensionality.

In recent years, Bayesian deep learning has emerged as a field combining deep neural networks with Bayesian probability theory, which provides a principled way of modeling uncertainty of machine learning models by employing prior distribution on their parameters and inferring the posterior distribution for these parameters using approximations such as Variational Bayes (Blundell et al. 2015; Gal and Ghahramani 2016b). Then, the posterior predictive distribution is approximated with sampling methods, which brings a significant computational overhead and leads to noise in predictive uncertainty estimates.

In these models, predictive uncertainty is modelled by taking samples from the posterior distributions of model parameters and using the sampled parameters to create a distribution of predictions for each input of the network. However, as we show in our experiments, modeling uncertainty of network parameters may not necessarily lead to good estimates of the predictive uncertainty of neural networks (Hafner et al. 2018). This is the case especially for the misclassified in-distribution samples, where Bayesian models associate similar levels of uncertainties with their successful and failed predictions.

Recently, a number of approaches (Sensoy, Kaplan, and Kandemir 2018; Malinin and Gales 2018) have been proposed to use outputs of neural networks to estimate the parameters of the Dirichlet prior of the categorical distribution for classification, instead of predicting a categorical distribution through the *softmax* function. Then, the resulting Dirichlet distribution is used to calculate the predictive uncertainty for classification. While similar in principle, our work distinguishes from this line of work in two folds: (1) it relates the parameters (i.e., the pseudo counts) of the resulting Dirichlet distribution to the density of the training data through noise constructive estimation, (2) it automatically synthesizes out-of-distribution samples sufficiently close to the training data, instead of hand-picking an auxiliary dataset.

Previous approaches used manually-tuned noise (Hafner et al. 2018) or GAN in the input space (Lee et al. 2018) to create out-of-distribution samples. The approaches based on GAN may suffer from the so-called *mode collapse* problem. To avoid it in this work, we created samples by automatically perturbing each training example in the latent space separately. Also, to avoid generating samples too similar to or different from training examples, we used a generator with joint objectives defined over outputs of two discriminators.

Conclusions

In this work, we proposed to combine ideas from implicit density models, noise constructive density estimation, and evidential deep learning in a novel way to quantify classification uncertainty in neural networks. We also proposed to generate out-of-distribution samples by combining the strengths of VAEs and GANs. The generated examples are used for learning an implicit density model of the training data, which is then utilized to generate pseudocounts (i.e., evidence) for Dirichlet parameters. Through extensive experiments with well-studied datasets and comprehensive comparisons with recent approaches, we show that our approach significantly enhances the state of the art in two uncertainty estimation benchmarks: i) detection of out-of-distribution samples, and ii) robustness to adversarial examples.

Acknowledgments

This research was sponsored by the U.S. Army Research Laboratory (ARL) and the U.K. Ministry of Defence under Agreement Number W911NF-16-3-0001. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Army Research Laboratory, the U.S. Government, the U.K. Ministry of Defence or the U.K. Government. The U.S. and U.K. Governments are authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon. Also, Dr. Sensoy thanks to ARL for its support

[7] Table 2 indicates 17% of CIFAR5 test samples are misclassified.

under grant W911NF-16-2-0173, and Newton-Katip Çelebi Fund and TUBITAK for their support under grant 116E918.

References

Bishop, C. M. 2006. *Pattern recognition and machine learning*. springer.

Blundell, C.; Cornebise, J.; Kavukcuoglu, K.; and Wiestra, D. 2015. Weight uncertainty in neural networks. In *ICML*.

Deecke, L.; Vandermeulen, R.; Ruff, L.; Mandt, S.; and Kloft, M. 2018. Anomaly detection with generative adversarial networks.

Gal, Y., and Ghahramani, Z. 2016a. Bayesian convolutional neural networks with bernoulli approximate variational inference. *arXiv:1506.021*.

Gal, Y., and Ghahramani, Z. 2016b. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In *ICML*.

Gast, J., and Roth, S. 2018. Lightweight probabilistic deep networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 3369–3378.

Golan, I., and El-Yaniv, R. 2018. Deep anomaly detection using geometric transformations. In *Advances in Neural Information Processing Systems*.

Goodfellow, I.; Shlens, J.; and Szegedy, C. 2014. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*.

Gutmann, M. U., and Hyvärinen, A. 2012. Noise-contrastive estimation of unnormalized statistical models, with applications to natural image statistics. *Journal of Machine Learning Research* 13(Feb):307–361.

Hafner, D.; Tran, D.; Irpan, A.; Lillicrap, T.; and Davidson, J. 2018. Reliable uncertainty estimates in deep neural networks using noise contrastive priors. *arXiv:1807.09289*.

Josang, A. 2016. *Subjective Logic: A Formalism for Reasoning Under Uncertainty*. Springer.

Kingma, D.; Salimans, T.; and Welling, M. 2015. Variational dropout and the local reparameterization trick. In *NIPS*.

Kotz, S.; Balakrishnan, N.; and Johnson, N. 2000. *Continuous Multivariate Distributions*, volume 1. New York: Wiley.

Lakshminarayanan, B.; Pritzel, A.; and Blundell, C. 2017. Simple and scalable predictive uncertainty estimation using deep ensembles. In *NIPS*.

LeCun, Y.; Bottou, L.; Bengio, Y.; Haffner, P.; et al. 1998. Gradient-based learning applied to document recognition. *In the IEEE* 86(11):2278–2324.

Lee, K.; Lee, H.; Lee, K.; and Shin, J. 2018. Training confidence-calibrated classifiers for detecting out-of-distribution samples. In *International Conference on Learning Representations*.

Louizos, C., and Welling, M. 2017. Multiplicative normalizing flows for variational bayesian neural networks. In *ICML*.

Malinin, A., and Gales, M. 2018. Predictive uncertainty estimation via prior networks. In Bengio, S.; Wallach, H.; Larochelle, H.; Grauman, K.; Cesa-Bianchi, N.; and Garnett, R., eds., *Advances in Neural Information Processing Systems 31*. 7047–7058.

Matthies, H. G. 2007. Quantifying Uncertainty: Modern Computational Representation of Probability and Applications. In *Extreme Man-Made and Natural Hazards in Dynamics of Structures*. 105–135.

Mohamed, S., and Lakshminarayanan, B. 2016. Learning in implicit generative models. *arXiv preprint arXiv:1610.03483*.

Molchanov, D.; Ashukha, A.; and Vetrov, D. 2017. Variational dropout sparsifies deep neural networks. In *ICML*.

NHTSA. 2016. Tesla crash preliminary evaluation report (PE 16-007), U.S. Department of Transportation.

Pawlowski, N.; Brock, A.; Lee, M. C.; Rajchl, M.; and Glocker, B. 2017. Implicit weight uncertainty in neural networks. *arXiv:1711.01297*.

Rasmussen, C., and Williams, C. 2006. *Gaussian Processes for Machine Learning*. MIT Press.

Sensoy, M.; Kaplan, L.; and Kandemir, M. 2018. Evidential deep learning to quantify classification uncertainty. In Bengio, S.; Wallach, H.; Larochelle, H.; Grauman, K.; Cesa-Bianchi, N.; and Garnett, R., eds., *Advances in Neural Information Processing Systems 31*. 3179–3189.

Zhai, S.; Cheng, Y.; Lu, W.; and Zhang, Z. 2016. Deep structured energy based models for anomaly detection. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16, 1100–1109.

Zong, B.; Song, Q.; Min, M. R.; Cheng, W.; Lumezanu, C.; Cho, D.; and Chen, H. 2018. Deep autoencoding gaussian mixture model for unsupervised anomaly detection. In *International Conference on Learning Representations*.