Deep Hybrid Models for Out-of-Distribution Detection

Senqi Cao, Zhongfei Zhang
State University of New York at Binghamton
scao16@binghamton.edu, zhongfei@cs.binghamton.edu

Abstract

We propose a principled and practical method for out-of-distribution (OoD) detection with deep hybrid models (DHMs), which model the joint density $p(\boldsymbol{x}, y)$ of features and labels with a single forward pass. By factorizing the joint density $p(\boldsymbol{x}, y)$ into three sources of uncertainty, we show that our approach has the ability to identify samples semantically different from the training data. To ensure computational scalability, we add a weight normalization step during training, which enables us to plug in state-of-the-art (SoTA) deep neural network (DNN) architectures for approximately modeling and inferring expressive probability distributions. Our method provides an efficient, general, and flexible framework for predictive uncertainty estimation with promising results and theoretical support. To our knowledge, this is the first work to reach 100% in OoD detection tasks on both vision and language datasets, especially on notably difficult dataset pairs such as CIFAR-10 vs. SVHN and CIFAR-100 vs. CIFAR-10. This work is a step towards enabling DNNs in real-world deployment for safety-critical applications.

1. Introduction

One significant obstacle to deploying DNN models in real-world applications is that deep learning systems often break down in novel situations. Specifically, DNNs tend to yield unreliable predictive uncertainty estimates and make high-confident yet incorrect predictions when exposed to inputs drawn from unfamiliar distributions. Therefore, accurate quantification of predictive uncertainty of DNNs is critical in high-stake applications such as medical diagnosis [16, 87], self-driving vehicles [20, 45], and financial decision-making [65], where silent mistakes can lead to catastrophic consequences.

Suppose that the training data come from the distribution $p(y, \mathbf{x})$, where y denotes labels and $\mathbf{x}$ denotes features. The majority of the literature uses DNNs to model the conditional distribution $p(y|\mathbf{x})$, which has achieved impressive performance when the test data are restricted to $p(y, \mathbf{x})$

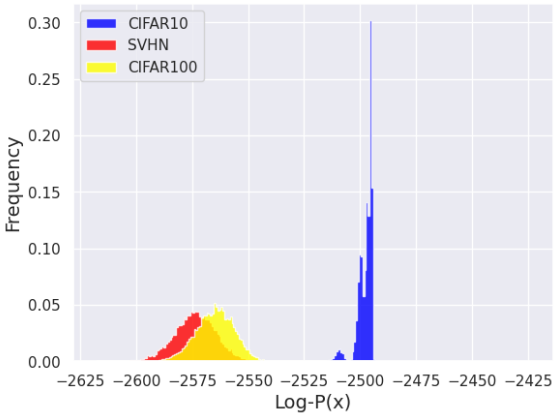

Figure 1. The histograms of $\log p(\mathbf{x})$ for both in-distributional (CIFAR-10) and OoD datasets (CIFAR-100 and SVHN).

[12, 24, 99, 100]. However, when faced with OoD test samples $\mathbf{x}^*$ drawn from a different distribution, $p(y|\mathbf{x}^*)$ often yields incorrect predictions with low uncertainties, resulting in silent failures. The main reason for this phenomenon is that $p(y|\mathbf{x}^*)$ does not fit a probability distribution over the whole $(\mathbf{x}, y)$ space. This motivates the need for a family of models endowed with a meaningful notion of probability over $(\mathbf{x}, y)$: *deep hybrid models* (DHMs), which aim to learn the joint distribution $p(y, \mathbf{x}) = p(y|\mathbf{x})p(\mathbf{x})$. Since $p(y, \mathbf{x})$ is a probability distribution which integrates to one over its support, $p(y, \mathbf{x}^*)$ is automatically decreased after maximizing the probability of the in-distributional (ID) data during training. As a result, even though $p(y|\mathbf{x}^*)$ might yield an incorrect label, $p(y|\mathbf{x}^*)p(\mathbf{x}^*)$ should be a small value suggesting a high predictive uncertainty and an alarming signal for potentially wrong predictions.

This paper aims to construct a family of DHMs that are computationally efficient and practically effective for detecting OoD samples. However, we face two main challenges. The first challenge is the model's expressibility and computational scalability. Constructing a DHM usually requires modeling $p(y|\mathbf{x})$ and $p(\mathbf{x})$ with two DNNs sharing a subset of their parameters [8, 44, 47, 50, 53, 69, 76, 77, 85]. Training a high-fidelity DHM on high dimensional data is difficult typically because $p(\mathbf{x})$ often underfits, and probabilistic inference for flexible $p(\mathbf{x})$ is usually computation-

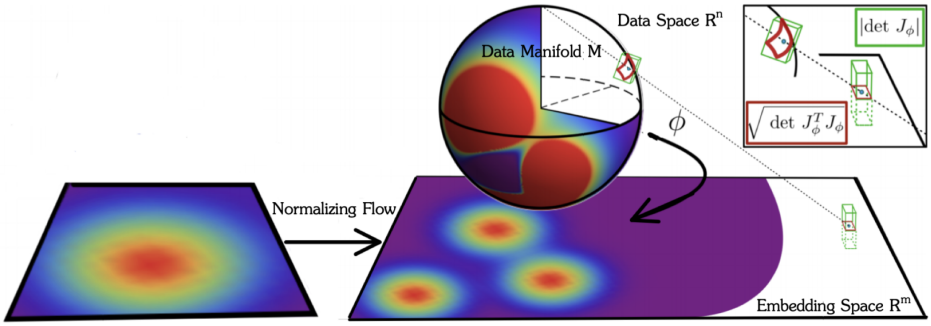

Figure 2. This figure (adapted and redrawn from [29]) illustrates the basic structure of our proposed DHMs. The data manifold $\mathcal{M}$ embedded in R^n is mapped, under an approximately volume-preserving mapping ϕ, to the low dimensional embedding space $R^m (m < n)$, where an NF learns to capture its probability density and a discriminator learns to distinguish semantically different objects.

ally expensive. Recently, normalizing flows (NFs) [15, 79] have been adopted to model $p(\mathbf{x})$ [42, 72] due to their great promise for modeling complex distributions. A standard Euclidean NF is a transform $\phi : \mathbf{R}^n \to \mathbf{R}^n$ that maps the data points $\mathbf{x}$ to their latent embeddings $\mathbf{h}$ equipped with a tractable base density $p(\mathbf{h})$, where the density $p(\mathbf{x})$ can be computed in closed-form: $p(\mathbf{x}) = p(\mathbf{h})|\det J_\phi|$. However, this result requires the NF ϕ to be invertible and dimension-preserving, imposing significant structural constraints on ϕ and limiting the model's expressibility [4, 13, 14, 21, 23, 32, 41, 46, 84]. To overcome the limitations on the dimensionality, a few attempts have been made to generalize NFs from Euclidean spaces to Riemannian manifolds where $|\det J_\phi|$ becomes a *pseudo determinant* $\sqrt{|\det J_\phi^T J_\phi|}$ [11, 29, 43, 68]. However, to make computing the pseudo determinant tractable (yet still very expensive), additional restrictions are usually placed on ϕ, significantly limiting the model's expressiveness and performance.

The second challenge is that NFs tend to learn low-level features of their inputs, such as simple graphical structures for image data, rather than high-level semantic information [48]. This is due to the fact that the features learned by NFs are primarily aimed at reconstructing the inputs rather than distinguishing semantically different objects, such as a cat and a dog. Although the semantic structures of ID and OoD data are generally different, they may share common low-level features. As a result, NFs, in general, fail to detect OoD samples.

In this paper, we introduce a simple and effective method to deal with both challenges. First, to relax the dimension-preserving requirement of ϕ while also circumventing the expensive step of computing the pseudo determinant, we replace the invertible mapping ϕ with a *bi-Lipschitz continuous* DNN $\phi : \mathbf{R}^n \to \mathbf{R}^m$ by applying *spectral normalization* [70] to its weights. A bi-Lipschitz continuous function is distance-preserving and bijective (restricted to its image) but has no restrictions on the dimensionality of its output layer. This property enables us to use SoTA DNN architectures to obtain geometry-preserving and low dimensional representations of the inputs $\mathbf{x}$, namely $\mathbf{h} = \phi(\mathbf{x})$. We show that a bi-Lipschitz continuous function ϕ also approximately preserves the measure, intuitively the local volumes of the data manifold, implying that $p(\mathbf{x}) \approx p(\mathbf{h})$. Second, to capture the density of $\mathbf{h}$, we add a Euclidean NF $f : \mathbf{h} \mapsto \mathbf{z}$ on top of it, where $\mathbf{z} \sim \mathcal{N}(0, I)$. In order for ϕ to learn high-level semantic features of its inputs $\mathbf{x}$, we also add a fully connected layer as a discriminator $c : \mathbf{h} \mapsto y$. The structure of our proposed DHMs is illustrated in Figure 2.

While the lack of restrictions on ϕ increases a DHM's expressivity, a DHM is computationally efficient because it only requires one feed-forward pass to obtain both $p(y|\mathbf{x})$ and $p(\mathbf{x})$ without the expensive step of calculating the pseudo determinant. By learning a joint embedding $\mathbf{h}$ for both the density estimator f and the discriminator c, a DHM improves the discriminative expressiveness of the features. Therefore, the density estimator f is built on high-level features and thus able to distinguish objects that are semantically different from the training data. To our knowledge, this is the first work to reach 100% in OoD detection tasks on both vision and language datasets. This work is a step towards enabling DNNs in real-world deployment for safety-critical applications since any result lower than 100% still provides no safety guarantee.

Contributions: 1: in Section 2, we introduce a novel uncertainty factorization: factorizing $p(y, \mathbf{x})$ into three sources of uncertainty. It provides theoretical support for DHMs to do OoD/anomaly detection. **2**: we propose one type of DHMs that is both computationally efficient and practically effective for OoD detection.

2. Uncertainty Factorization

Notation and Problem Setup We assume that the training data $D = \{\mathbf{x}_i, y_i\}_{i=1}^N$ are generated from a joint distribution $p(y, \mathbf{x})$. The features $\{\mathbf{x}_i\}_{i=1}^N$ are $i.i.d.$ samples of a random variable $\mathbf{x} \in R^n$ that takes values on a low-dimensional manifold $\mathcal{M}$ (equipped with a metric $\|.\|_\mathcal{M}$) diffeomorphic to R^m with typically $m < n$ (the well-known *manifold hypothesis*). Our goal is to train a DHM $p(y, \mathbf{x}; \theta) = p(y|\mathbf{x}; \theta)p(\mathbf{x}; \theta)$ parameterized by θ to generate reliable predictive uncertainty estimates that can be used to detect OoD samples $\mathbf{x}^*$.

2.1. Different Types of Uncertainty

In machine learning modeling, there are two types of uncertainty that are crucial to distinguish: *aleatoric uncertainty* and *epistemic uncertainty* [83]. Aleatoric uncertainty, or *data uncertainty*, arises due to the inherent randomness of the data generating mechanism, which is not reducible by observing more data points. On the other hand, epistemic uncertainty describes the uncertainty due to the lack of data and knowledge. It can be sufficiently reduced theoretically as the size of training data grows to infinity. A model's epistemic uncertainty mainly comes from two sources [60]: *parametric uncertainty*, or *model uncertainty*, which measures the uncertainty in estimating the model parameters under the current model specification, and *distributional uncertainty* that arises due to the discrepancy between the training and test distributions. A well-calibrated model should assign higher distributional uncertainty to OoD samples than to ID samples.

2.2. Uncertainty Factorization

What Is a Good Uncertainty Factorization? In the literature of uncertainty factorization, the main goal is to factorize the model into two or three types of uncertainty so that we can directly access the uncertainty information once we learn the model. First, we want the uncertainty factorization to be general and natural so that it can be used in a wide variety of models. Second, a good uncertainty factorization should be semantically accurate, which means that the results should be aligned with the definitions described in Section 2.1.

The existing literature [19, 59, 67] focuses on factorizing the posterior predictive distribution $p(y|\mathbf{x}, \theta, D)$. The main shortcoming of this approach is that the proposed factorizations are often complicated and not general, only suitable for specific models. Furthermore, many factorizations fail to correctly capture the semantics of different sources of uncertainty described in Section 2.1. A more detailed literature review is presented in Appendix A.

Our Proposed Uncertainty Factorization Instead of factorizing the posterior predictive distribution $p(y|\mathbf{x}, \theta, D)$, we propose to factorize the posterior joint distribution $p(y, \mathbf{x}, \theta|D)$ into three sources of uncertainty. We assume that $\mathbf{x}$ is independent of θ. This assumption is generally valid for the following reason: θ is trained from the data D; thus, θ is dependent on D. However, in our problem settings, $\mathbf{x}$ can be OoD, which means that the distribution of $\mathbf{x}$ is independent of (can be different from) the distribution of the data D. Therefore, $\mathbf{x}$ is independent of θ. With this assumption, the posterior joint distribution can be factorized as follows:

$$p(y, \mathbf{x}, \theta|D) = \underbrace{p(y|\mathbf{x}, \theta)}_{\text{data}} \underbrace{p(\mathbf{x}|D)}_{\text{distributional}} \underbrace{p(\theta|D)}_{\text{model}} \quad (1)$$

$p(y|\mathbf{x}, \theta)$ is the aleatoric uncertainty which measures the intrinsic randomness of y at a particular point $\mathbf{x}$; $p(\mathbf{x}|D)$ represents the distributional uncertainty, reflecting how likely it is to see a new point $\mathbf{x}$ in light of the training data D; $p(\theta|D)$ is the parametric uncertainty which can be reduced to a Dirac delta function (point estimate) given infinite data points.

This uncertainty factorization is quite neat and semantically accurate. More importantly, it is also quite general since it does not introduce auxiliary variables and therefore can be used in a wide variety of models. This uncertainty factorization can be used in both Bayesian and non-Bayesian models since the parametric uncertainty is separated (Equation (1)). To be specific, for Bayesian models, when making predictions on a new data point $\mathbf{x}^*$, we integrate out θ and obtain:

$$p(y, \mathbf{x}^*|D) = p(\mathbf{x}^*|D) \int p(y|\mathbf{x}^*, \theta) p(\theta|D) d\theta \quad (2)$$

For non-Bayesian models, if we use the maximum likelihood method, we have:

$$p(y, \mathbf{x}^*|D) = \underbrace{p(\mathbf{x}^*|D)}_{\text{data}} \underbrace{p(y|\mathbf{x}^*, \hat{\theta}_{MLE})}_{\text{distributional}} \quad (3)$$

It is crucial to distinguish different types of predictive uncertainty in real-world problems so that different actions can be taken depending on the source of uncertainty. Our uncertainty factorization enables us to model different sources of uncertainty separately. For example, from Equations (2) and (3), the parametric uncertainty can only be estimated in Bayesian models; however, the distributional uncertainty and the aleatoric uncertainty can be modeled independently regardless of whether the model is under the Bayesian setting. Thus, this uncertainty factorization is practical when one needs to differentiate between different sources of uncertainty with a single model.

2.3. Connecting Uncertainty Factorization to OoD Detection

Our proposed uncertainty factorization (Equation (1)) sheds light on what type of models are capable of detecting OoD samples. Without combining additional techniques, the widely used conditional models $p(y|\mathbf{x}, \theta)$ alone are not good at OoD detection because they only capture the data uncertainty. However, a well-trained DHM $p(y, \mathbf{x}, \theta)$ should be able to detect OoD samples since it contains the distributional uncertainty in nature. Furthermore, since a DHM fits a probability distribution over the whole $(\mathbf{x}, y)$ space, it also learns the interactions between $\mathbf{x}$ and y. Labels y provide the interested ID semantics of the features $\mathbf{x}$ and tell the model what is ID and what is OoD. Consequently, modeling $p(\mathbf{x}, \theta)$ alone with NFs generally fails to detect

Figure 3. An example to show that labels contain ID semantic information.

OoD samples [48] since the model does not learn what it means to be ID.

We briefly illustrate the above idea with a toy example. In Figure 3, given a training dataset of six pictures and two test pictures, without information about the labels, we have no idea of what it means to be OoD. If the labels are "1", "2", and "3", the test data are ID. On the other hand, if the labels are "red" and "blue", the test pictures should be considered OoD. With different labels, the ID semantics can change. As a result, the notion of OoD also changes. Therefore, labels contain semantic information that is useful for OoD detection, especially for hard OoD samples.

3. L-Measure-Preserving Mappings

In this section, our goal is to construct an approximately measure-preserving (volume-preserving) mapping ϕ such that we have $p(\mathbf{x}) \approx p(\mathbf{h})$ where $\mathbf{h} = \phi(\mathbf{x})$. To begin with, we provide some mathematical background.

Definition 3.1 (Volume of a Matrix A) *Consider a matrix $A \in R^{m\times n}$ with nonzero singular values σ_1, σ_2,...,σ_r, the volume of A, $volA := \Pi_{i=1}^{r}\sigma_i$.*

$volA$ is a generalization of $|\det(A)|$, which intuitively measures the volume of the parallelepiped generated by the rows (columns) of A. Any r-dimensional unit cube embedded in R^n is mapped, through A, into a parallelepiped of volume $volA$. If A is of full row (column) rank, then $volA = \sqrt{|\det(G)|}$ where $G = A^TA$ is the Gram matrix of A [5].

Normalizing Flow on Riemannian Manifolds Let $\mathbf{x} \in R^n$ be a random variable that takes values on an m-manifold $\mathcal{M}$ with $m < n$ (the manifold hypothesis). Consider an NF $\phi : \mathbf{R}^n \to \mathbf{R}^m$ that is bijective and differentiable *almost* everywhere (allowing for piecewise differentiable functions). Denote the Gram matrix of ϕ as $G(\mathbf{x}) = J_\phi(\mathbf{x})^T J_\phi(\mathbf{x})$ where $J_\phi(\mathbf{x})$ is the Jacobian of ϕ; then we have $volJ_\phi = \sqrt{|\det G(\mathbf{x})|}$. For any measurable set $B \subset R^n$, one can calculate the probability of its image under ϕ [6,11,29,43,68]:

$$P(\mathbf{h} \in \phi(B)) = \int_B p(\mathbf{x})d\mathbf{x} = \int_{\phi(B)} p(\mathbf{h})dV_\phi(\mathbf{h}) \quad (4)$$

where

$$p(\mathbf{h}) = p(\mathbf{x})/volJ_\phi \quad (5)$$

$$dV_\phi(\mathbf{h}) = volJ_\phi d\mathbf{x} \quad (6)$$

Equation (5) is a general version of the "change-of-variables" formula, and it illustrates the relationship between $p(\mathbf{x})$ and $p(\mathbf{h})$. To evaluate the density $p(\mathbf{h})$, we need to compute the volume term $volJ_\phi$, which is not scalable to high-dimensional data. Equation (6) elucidates the geometric intuition of $volJ_\phi$: it represents the ratio between two differentials (infinitesimal volumes) before and after the mapping ϕ. In light of this observation, one way to bypass the computation of $volJ_\phi$ is to make ϕ a measure-preserving (volume-preserving) mapping where $volJ_\phi \equiv 1$, which is the goal of this section. However, in practice, this requirement might be too stringent. Therefore we consider a more general version of measure-preserving mappings (Definition 3.4) to keep ϕ's expressivity.

3.1. Bi-Lipschitz Mappings Approximately Preseve the Measure (Volume)

Definition 3.2 (Bi-Lipschitz mappings). *Consider two metric spaces $(X, \|.\|_X)$ and $(H, \|.\|_H)$. A function $\phi : X \to H$ is called bi-Lipschitz continuous if there exists a constant $L \geq 1$, s.t. $\frac{1}{L} * \|\boldsymbol{x}_1 - \boldsymbol{x}_2\|_X \leq \|\phi(\boldsymbol{x}_1) - \phi(\boldsymbol{x}_2)\|_H \leq L * \|\boldsymbol{x}_1 - \boldsymbol{x}_2\|_X$, $\forall \boldsymbol{x}_1, \boldsymbol{x}_2 \in X$.*

The smallest L that satisfies the inequality is called the *bi-Lipschitz constant* of ϕ, and we say that ϕ is *L-bi-Lipschitz continuous*. If ϕ only satisfies half of the inequality $\|\phi(\mathbf{x}_1) - \phi(\mathbf{x}_2)\|_H \leq L * \|\mathbf{x}_1 - \mathbf{x}_2\|_X$, we say that ϕ is *L-Lipschitz continuous*. Note that invertible functions are not necessary to be bi-Lipschitz continuous, whereas bi-Lipschitz continuous functions are always bijective.

Proposition 3.3 *If a function ϕ is L-bi-Lipschitz continuous, then ϕ is differentiable almost everywhere (Theorem 3.1.6 of [26]), and the singular values of its Jacobian lie in the interval (L^{-1}, L).*

Definition 3.4 (L-measure-preserving). *Consider two measure spaces $(X, \mathscr{A}, vol_X)$ and $(H, \mathscr{H}, vol_H)$. A function $\phi : X \to H$ is called L-measure-preserving (volume-preserving) if there exists a constant $L \geq 1$, s.t. $\frac{1}{L} * vol_X(B) \leq vol_H(\phi(B)) \leq L * vol_X(B), \forall B \in \mathscr{A}$.*

Theorem 3.5 (Bi-Lipschitz mappings are approximately measure-preserving). *Specifically, if a mapping $\phi : R^n \to R^m$ is $L^{1/m}$-bi-Lipschitz continuous, then it is L-measure-preserving.*

The proof is in Appendix A. Suppose that ϕ is L-measure-preserving and $B_\mathbf{x} \in \mathscr{A}$ is an infinitesimal open cover of a point $\mathbf{x} \in X$. The volume ratio $vol_H(\phi(B_\mathbf{x}))/vol_X(B_\mathbf{x}) = dV_\phi(\mathbf{h})/d\mathbf{x} = volJ_\phi$ lies in the interval (L^{-1}, L) where $L \geq 1$ and thus approximately equal to 1 when L is close to 1. Therefore, from Equation (5), we have $p(\mathbf{x}) = p(\mathbf{h})volJ_\phi \approx p(\mathbf{h})$, which gives us a chance to circumvent the expensive step of computing $volJ_\phi$. The bounds are tight when $L = 1$ and ϕ becomes a strictly measure-preserving mapping.

3.2. Constructing Measure-Preserving Mappings via Spectral Normalization

Modern DNNs, such as ResNets, BERT, and Transformers, are often constructed with residual blocks. Spectral normalization (SN) [70] provides a simple yet efficient method to ensure the L-measure-preserving property on such DNN architectures. We state the theorem formally below:

Proposition 3.6 (Lipschitz-bounded residual blocks lead to bi-Lipschitz continuous DNNs [3,58]). *Consider a residual DNN $\phi = \phi_d \circ ... \circ \phi_2 \circ \phi_1$ where $\phi_l(\boldsymbol{x}) = \boldsymbol{x} + g_l(\boldsymbol{x})$ for $l = 1, \dots, d$. If all $g_l(\boldsymbol{x})$ are β-Lipschitz continuous where $0 < \beta < 1$, then ϕ is L-bi-Lipschitz continuous where $L = max\{(1-\beta)^{-d}, (1+\beta)^d\}$.*

The proof is in Appendix A. We consider $g_l(\mathbf{x}) = \sigma(W_l\mathbf{x} + b_l)$, where σ is the activation function such as ReLU. We notice that the Lipschitz constant of the affine transformation $W_l\mathbf{x} + b_l$ is the spectral norm of its weight matrix W_l, denoted as $\|W_l\|_2$ (the largest singular value of W_l) [70,75], and the spectral norm of the activation function σ is less than 1. Therefore, to ensure g_l to be β-Lipschitz continuous where $0 < \beta < 1$, it is sufficient to restrict $\|W_l\|_2$ to be less than 1. To this end, we apply SN to the weight matrices $\{W_l\}_{l=1}^d$. Following [4], at each training step, we first estimate the spectral norm $\hat{\eta}_l \approx \|W_l\|_2$ with the *power iteration* method [31, 70] and then normalize the weights via:

$$\tilde{W}_l = \begin{cases} c * W_l/\hat{\eta}_l & \text{if } c < \hat{\eta}_l \\ W_l & \text{otherwise} \end{cases} \tag{7}$$

$c > 0$ is a scaling coefficient called the SN upper bound since it is the upper bound for the scaled spectral norm ($\|\tilde{W}_l\|_2 \leq c$), and therefore the upper bound for the Lipschitz constant of $g_l(\mathbf{x})$ ($\beta < c$).

Tighter Bi-Lipschitz Bounds It is crucial to note that even if L_1 and L_2 are individually the best bi-Lipschitz constants of ϕ_1 and ϕ_2, respectively, L_1L_2 will not necessarily be the smallest bi-Lipschitz constant of $\phi_2 \circ \phi_1$. It is possible to obtain a much tighter bi-Lipschitz bound by considering the entire DNN as a whole rather than each layer in isolation. Under mild assumptions, we can provide some analysis on the bi-Lipschitz bound of a residual DNN.

Corollary 3.7 (A tighter bi-Lipschitz bound for residual DNNs). *Consider a DNN ϕ composed of d residual blocks, each of which is β-Lipschitz continuous where $0 < \beta < 1$. The expected value of the bi-Lipschitz constant of the DNN is $0.5^d(1 + \beta + \frac{1}{1+\beta})^d$.*

The proof (informal) is in Appendix A. Let us look at an example. If our model is a DNN composed of 10 residual blocks and each block is 0.1-Lipschitz continuous. From the manifold hypothesis, suppose that the data lie on a three-dimensional manifold. Then, after the model is trained on the dataset, we would expect it to become a 1.14-measure-preserving mapping.

It is crucial to note that an overly strong measure-preserving property (L is too close to 1) might harm OoD detection in practice. This is due to the fact that the bi-Lipschitz condition, by definition, leads the model to preserve a naive metric $\|.\|_X$ (such as the Euclidean distance) in the original data space X rather than a semantically meaningful distance in the data manifold $\mathcal{M}$. Due to highly non-convexity, OoD data in space X can be even closer to the center of the ID data than the ID data themselves. Consequently, an overly strong measure-preserving condition makes detecting OoD samples even harder. For example, a strictly measure-preserving mapping would become a naive rotation or identity function (for residual networks), in which case detecting OoD samples is impossible. Therefore, we hope that our model is measure-preserving only to the extent that it is safe to drop the volume term $volJ_\phi$, and we can still do OoD detection. For example, if the probabilities of the ID data are ten times larger than the probabilities of the OoD data, it is safe to drop the volume term even if the measure-preserving constant is as large as 2. In practice, the SN upper bound c is a critical hyperparameter to control the strictness of the measure-preserving property.

4. Method Summary for DHMs

Architecture Given a DNN ϕ, $logits(\mathbf{x}) = c \circ \phi(\mathbf{x}; \theta)$ where c is a fully connected layer, a DHM makes two changes to the model: applying SN on the weights of ϕ and adding an NF $f : \phi(\mathbf{x}; \theta) \mapsto \mathbf{z}$, where $\mathbf{z} \sim \mathcal{N}(0, I)$.

Objective In DHMs, the maximum likelihood objective is naturally separated into $\log p(y, \mathbf{x}; \theta) = \log p(y|\mathbf{x}; \theta) + \log p(\mathbf{x}; \theta)$. In practice, though, a weighted maximum likelihood objective $\log p(y, \mathbf{x}; \theta) = \log p(y|\mathbf{x}; \theta) + \lambda \log p(\mathbf{x}; \theta)$ is commonly used where λ is a scaling constant, as prediction accuracy is usually of more interest.

Training and Prediction We summarize the methods in Algorithm 1 and Algorithm 2.

Algorithm 1 DHM Training

Input: batches $B = \{\mathbf{x}_i, y_i\}_{i=1}^M$
Initialize parameters θ
for step = 1 **to** max_step **do**
    SGD update θ
    Apply SN to θ
end for

Algorithm 2 DHM Prediction

Input: test example $\mathbf{x}^*$
Compute $p(y|\mathbf{x}^*) = \max \text{softmax} \circ c \circ \phi(\mathbf{x}^*; \theta)$
Compute $p(\mathbf{x}^*) \approx p(\mathbf{z})|\det J_f(\phi(\mathbf{x}^*; \theta))|$
Compute Uncertainty $= p(y|\mathbf{x}^*)p(\mathbf{x}^*)$ or $p(\mathbf{x}^*)$
Compute Label $= \arg\max c \circ \phi(\mathbf{x}^*; \theta)$
Return Label, Uncertainty

Why Do DHMs Work? **1**: a DHM aims to learn a joint density $p(\mathbf{x}, y)$ which contains the distributional uncertainty in nature (Equation (1)); **2**: by learning a joint embedding for both $p(y|\mathbf{x})$ and $p(\mathbf{x})$, a DHM encodes high-level semantic information of the data (partly coming from labels y); **3**: by applying SN, we construct an approximately measure-preserving mapping ϕ ($p(\mathbf{x}) \approx p(\mathbf{h})$ where $\mathbf{h} = \phi(\mathbf{x})$). This technique enables us to use SoTA DNNs to learn low-dimensional representations of the inputs $\mathbf{x}$ without computing the expensive volume term (pseudo determinant), which solves the problem of expressibility and computational scalability.

5. Related Work

Hybrid Models The idea of hybrid modeling might originate from C.M. Bishop (1994) [8], who uses it for novelty detection. Hybrid models are then shown to be useful for supervised learning [44, 69], semi-supervised learning [47, 50, 53, 76, 77], and information regularization [85]. Recently, hybrid models have been applied to OoD detection [72] and open set recognition [42]. [72] proposes constructing hybrid models where $p(\mathbf{x})$ is modeled with an NF and $p(y|\mathbf{x})$ is modeled with a generalized linear model stacked on top of $p(\mathbf{x})$. Unfortunately, this method produces low OoD detection performance because the latent embeddings learned from the NF do not have sufficient discriminative expressiveness. One way to improve the discriminative expressiveness of the latent embeddings is to learn a joint embedding for both $p(\mathbf{x})$ and $p(y|\mathbf{x})$. To this end, [42] (OpenHybrid) proposes to attach an NF ($p(\mathbf{x})$) to the penultimate layer of a DNN ($p(y|\mathbf{x})$), which outperforms baselines in the open set recognition field. However, OpenHybrid still performs poorly in OoD detection tasks on difficult dataset pairs such as CIFAR-100 vs. CIFAR-10. One main reason is that the authors mistakenly use $p(\mathbf{h})$ to substitute $p(\mathbf{x})$ as the score for OoD detection, whereas $p(\mathbf{h})$ and $p(\mathbf{x})$ can be very different everywhere in practice. Furthermore, the gradients of the NF and the DNN are propagated separately instead of jointly at each training step. In this paper, we improve upon OpenHybrid and correct their mistake by applying SN to the weights of the DNN to enforce its approximately measure-preserving property ($p(\mathbf{h}) \approx p(\mathbf{x})$) and training all the weights concurrently, which achieves surprising results.

OoD Detection A principled approach to estimate predictive uncertainty is Bayesian DNNs [40, 62, 74] that learn a posterior distribution over their parameters with MCMC [2, 89], variational inference [9, 10, 25, 34, 61, 90, 92], or other posterior approximations such as MC-Dropout [28], SWAG [64], and Laplace approximation [80]. In practice, Bayesian approaches are outperformed by Deep Ensembles [51], which come at the expense of more computational cost and memory. Another approach is to make use of real [38, 67] or synthetic [54] auxiliary OoD examples to learn to distinguish OoD inputs. For example, OE [38] and DPN [67] train a DNN on OoD examples through regularization, while others use OoD examples to tune hyperparameters such as Temperature Scaling [36], ODIN [56], and Mahalanobis [55]. However, this type of methods is unable to generalize to other unseen OoD datasets in practice. Furthermore, recent studies on unsupervised methods, including density-based [17, 22, 27, 33, 63, 66, 71, 73, 78, 81, 82, 97] and self-supervised [7, 30, 39, 57, 86, 91] approaches, have shown that more elaborate data augmentation such as rotation, reflection, blurring [18], mixup [98], or adversarial training [51] significantly helps to learn discriminative features for OoD detection. In this paper, we do not assume access to OoD datasets in advance or rely on complicated data augmentation. More related to our work are SoTA single-model approaches: DUQ [88] and SNGP [58]. Both methods impose "distance-preserving" on the DNNs with either Jacobian penalty [35] or SN and suggest "distance-aware" output layers in the form of RBF kernels and Gaussian processes, respectively. However, as discussed in Section 3.2, an overly strict distance-preserving condition can harm OoD detection. Our method is a novel application of the approximately measure-preserving property, which outperforms both methods on various datasets.

6. Experiments

6.1. Vision and Language Understanding

Baseline Methods and Experimental Setup For fair comparisons, all the baseline approaches are supervised methods and do not require auxiliary OoD datasets either to train the models or to tune hyperparameters. We compare a **DHM** against a **deterministic** baseline (vanilla DNN) and an ensemble baseline: **Deep Ensembles** (with 10 independent DNNs). We also consider two SoTA single-model approaches: **DUQ** and **SNGP** (see Section 5). We include two ablated versions of a **DHM**: **DNN+SN** which performs SN on a vanilla DNN and **DNN+NF** which adds an NF on top of a vanilla DNN. For all the methods, we use a Wide ResNet 28-10 [96] for image classification and XL-Net [94] for language understanding. For CIFAR-10 [49] and CIFAR-100 [49], we apply the standard data augmentation (horizontal flips and random cropping) for both ID and OoD datasets. Further training and evaluation configurations are described in Appendix B.

Vision Domain: CIFAR-10 and CIFAR-100 We first evaluate a DHM's predictive accuracy and calibration error on ID test data. As shown in Table 1-2, the DHM slightly outperforms the vanilla DNN, DUQ, and SNGP in terms of predictive accuracy and is competitive with them in terms of calibration error. Deep Ensembles (10 DNNs) still have

	In-distribution			OoD: SVHN		OoD: CIFAR-100		Latency (↓) (ms/example)	
	Accuracy (↑)	ECE (↓)	NLL (↓)	AUROC (↑)	AUPR (↑)	AUROC (↑)	AUPR (↑)	Train	Test
Deterministic	96.0 ± 0.01	0.024 ± 0.002	0.158 ± 0.02	0.923 ± 0.01	0.907± 0.01	0.868 ± 0.01	0.820 ± 0.01	**1.39**	**0.62**
Deep Ensemb	**96.7 ± 0.01**	**0.010 ± 0.001**	**0.113 ± 0.01**	0.980 ± 0.01	0.978 ± 0.01	0.921 ± 0.01	0.919 ± 0.01	14.01	6.33
DUQ	95.5 ± 0.02	0.034 ± 0.002	0.234 ± 0.02	0.972 ± 0.01	0.969 ± 0.01	0.908± 0.01	0.888 ± 0.01	3.07	1.14
SNGP	96.0 ± 0.01	0.020 ± 0.002	0.150 ± 0.01	0.978 ± 0.01	0.975 ± 0.01	0.916 ± 0.01	0.911 ± 0.01	2.21	0.82
DNN+SN	96.1 ± 0.01	0.024 ± 0.003	0.155 ± 0.02	0.956 ± 0.01	0.937 ± 0.01	0.872 ± 0.01	0.840 ± 0.01	1.58	0.65
DNN+NF	96.2 ± 0.01	0.023 ± 0.003	0.152 ± 0.02	0.998 ± 0.01	0.995 ± 0.01	0.951 ± 0.03	0.938 ± 0.04	2.25	0.83
DHM (Ours)	96.3 ± 0.01	0.022 ± 0.003	0.149 ± 0.02	**1.000 ± 0.00**	**1.000 ± 0.00**	**1.000 ± 0.00**	**1.000 ± 0.00**	2.43	0.84

Table 1. Results for Wide ResNet-28-10 on CIFAR-10, averaged over 10 independent seeds.

	In-distribution			OoD: SVHN		OoD: CIFAR-10		Latency (↓) (ms/example)	
	Accuracy (↑)	ECE (↓)	NLL (↓)	AUROC (↑)	AUPR (↑)	AUROC (↑)	AUPR (↑)	Train	Test
Deterministic	80.5 ± 0.02	0.055 ± 0.004	0.783 ± 0.02	0.862 ± 0.01	0. 893 ± 0.01	0.806 ± 0.01	0.812 ± 0.01	**1.40**	**0.63**
Deep Ensemb	**81.9 ± 0.01**	**0.022 ± 0.001**	**0.658 ± 0.01**	0.919 ± 0.01	0.933 ± 0.01	0.854 ± 0.01	0.887 ± 0.01	14.22	6.42
DUQ	79.9 ± 0.02	0.084 ± 0.004	0.880 ± 0.02	0.897 ± 0.01	0.908 ± 0.01	0.839 ± 0.02	0.872 ± 0.02	1.75	0.67
SNGP	80.5 ± 0.03	0.040 ± 0.003	0.759 ± 0.02	0.928 ± 0.01	0.935 ± 0.01	0.863 ± 0.01	0.875 ± 0.01	1.86	0.70
DNN+SN	80.8 ± 0.02	0.055 ± 0.003	0.780 ± 0.02	0.886 ± 0.01	0.901 ± 0.01	0.811 ± 0.02	0.819 ± 0.02	1.60	0.65
DNN+NF	81.1 ± 0.02	0.051 ± 0.004	0.760 ± 0.02	0.989 ± 0.02	0.985 ± 0.02	0.945 ± 0.04	0.925 ± 0.04	2.31	0.84
DHM (Ours)	81.3 ± 0.02	0.049 ± 0.004	0.757 ± 0.02	**1.000 ± 0.00**	**1.000 ± 0.00**	**1.000 ± 0.00**	**1.000 ± 0.00**	2.54	0.88

Table 2. Results for Wide ResNet-28-10 on CIFAR-100, averaged over 10 independent seeds.

more powerful classification capacities than single-model approaches. We then evaluate a DHM's OoD detection performance for notably difficult dataset pairs, such as CIFAR-10/-100 vs. SVHN, CIFAR-10 vs. CIFAR-100, and CIFAR-100 vs. CIFAR-10. Note that the classes of CIFAR-10 and CIFAR-100 are mutually exclusive. Results in Table 1-2 show that the DHM clearly outperforms the existing SoTA, including Deep Ensembles, and achieves 100% AUROC and AUPR on all the OoD detection tasks.

Text Domain: CLINC150 We report a DHM's performance on a real-world dataset: CLINC150 [52]. CLINC150 is designed for evaluating the performance of an intent classification system in the presence of "out-of-scope" queries. We train XLNet models only on in-scope data and evaluate their predictive accuracy and calibration error on the in-scope test data and their OoD detection performance on the out-of-scope data. As shown in Table 3, compared to the vanilla DNN, DUQ, and SNGP, the DHM achieves a slightly superior predictive accuracy and a competitive calibration error. For the OoD detection tasks, consistent with the vision experiments, the DHM outperforms all the baselines and achieves 100% AUROC and AUPR.

Ablation Study DNN+NF without SN does not achieve the best OoD detection performance in both vision and language understanding tasks. Empirically, it produces unstable results, sometimes achieving 100%, whereas on other occasions (different seeds or parameter initializations) falling below 90%. This fact illustrates the necessity of the measure-preserving property obtained by SN for stable and high-quality predictive uncertainty quantification.

Additional Experiments To further validate a DHM, we perform two more sets of experiments. First, we evaluate a DHM's OoD detection performance on additional widely-used but relatively easier OoD datasets such as TinyImageNet [1], LSUN [95], and iSUN [93]. Consistent with the previous experiments, the DHM achieves 100% AUROC and AUPR on all the OoD datasets. Second, to increase the task's difficulty, we conduct all the vision experiments with a smaller network, ResNet18 [37]. While the performance of all the baseline methods decreases by a large margin, a DHM still retains 100% AUROC and AUPR. Detailed results are reported in Appendix C.

Hyperparameter Analysis A DHM consists of three components: DNN+SN+NF. For training the DNN and the NF, we use almost the same hyperparameters and training setup as the original papers, namely [96] for Wide ResNet 28-10, [94] for XLNet, and [42] for the NF, with a few exceptions on learning rates and batch size. Detailed configurations are in Appendix B. SN contains two hyperparameters: the number of power iterations and the SN upper bound c. Following [58], we set power iteration to 1, and c to 6 for Wide ResNet 28-10 and 0.95 for XLNet. [58] proposes analysis and methods for selecting c, namely performing a grid search for $c \in \{0.95, 1, 2, ...\}$ to find the smallest possible value of c that still maintains competitive predictive performance. It is generally sufficient to apply only one step of power iteration to obtain a reasonable estimate of the spectral norms for dense layers. However, it highly overestimates the true spectral norms for the convolutional kernels. Therefore, a looser SN upper bound c is

	In-Scope Content			Out-of-Scope Content		Latency (↓) (ms/example)	
	Accuracy (↑)	ECE (↓)	NLL (↓)	AUROC (↑)	AUPR (↑)	Train	Test
Deterministic	97.1 ± 0.10	0.025 ± 0.002	0.160 ± 0.10	0.975 ± 0.02	0. 974 ± 0.02	**3.66**	**2.08**
Deep Ensemb	**98.2** ± **0.03**	**0.011** ± **0.001**	**0.124** ± **0.02**	0.990 ± 0.01	0.989 ± 0.01	36.70	20.82
DUQ	96.7 ± 0.05	0.030 ± 0.004	0.160 ± 0.08	0.974 ± 0.02	0.975 ± 0.02	5.49	3.45
SNGP	96.9 ± 0.05	0.015 ± 0.003	0.157 ± 0.04	0.989 ± 0.01	0.987 ± 0.01	6.07	3.97
DNN+SN	97.1 ± 0.11	0.025 ± 0.003	0.159 ± 0.09	0.976 ± 0.01	0.976 ± 0.01	3.71	2.13
DNN+NF	97.2 ± 0.10	0.022 ± 0.005	0.156 ± 0.05	0.991 ± 0.01	0.990 ± 0.02	3.98	3.08
DHM (Ours)	97.5 ± 0.10	0.020 ± 0.005	0.152 ± 0.05	**1.000** ± **0.00**	**1.000** ± **0.00**	4.01	3.19

Table 3. Results for XLNet on CLINC150, averaged over 10 independent seeds.

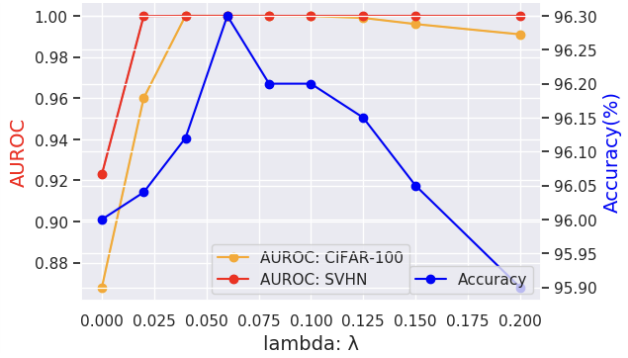

Figure 4. This figure shows, for CIFAR-10 as ID data, how the predictive accuracy and the AUROC change when λ varies.

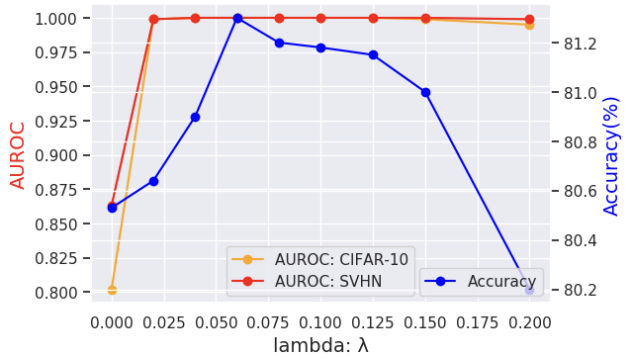

Figure 5. This figure shows, for CIFAR-100 as ID data, how the predictive accuracy and the AUROC change when λ varies.

generally needed for SoTA performance for convolutional layers. NF introduces one more hyperparameter: λ. A large value of λ may harm the model's predictive performance, while a small value of λ may lead to the loss of the measure-preserving property. Since we cannot assume access to the OoD data in advance, we recommend performing a binary search for $\lambda \in (0, 1)$ to choose the largest possible value of λ, not affecting the predictive performance. This strategy works well in practice because we observe that the predictive accuracy is generally more sensitive to λ than the OoD detection performance (Figure 4-5). In this paper, we set λ to be 0.06 for all the vision experiments and 6e-6 for the text experiments.

Computation Analysis Suppose that ϕ has d hidden layers of size $\{D_l\}_{l=1}^{d}$. The main computational burden comes from performing SN using the power iteration method, which has time complexity $O(\sum_{l=1}^{d} D_l)$; the NF also introduces some additional overhead depending on its size. During training, a DHM is approximately 1.7-1.8 times slower than a single DNN. However, the gap can become smaller after removing the SN step in the testing mode (approximately 1.3-1.4 times slower).

Limitations and Social Impacts Experimental results have showcased DHMs' ability in detecting OoD samples on a wide range of standard benchmark datasets in different modalities. Nonetheless, before they can be deployed in industrial-scale and safety-critical applications such as self-driving cars, DHMs still need to be tailored to specific situations. DHMs should be examined on more complex and harsh data such as highly corrupted and unclear samples to ensure safety guarantees. Another limitation of our method is that DHMs only work with residual-based DNNs. However, we notice that modern deep learning models (ResNets, BERT, Transformers) are commonly composed of residual blocks. Therefore, our method is still widely useful. Furthermore, any powerful theory is a mixed blessing. If misused or misinterpreted, a good theory may become a threat to humanity, resulting in negative social impacts such as discrimination. Hence, we suggest that users of DHMs not be over-dependent on them and always think about the results critically. We hope that our method can be used to bring positive impacts to society and improvements to applications on human well-being such as medical imaging, pathologies, and policy decision-making.

7. Conclusion

We propose DHMs, a practical and efficient methodology for modeling and inferring expressive probability distributions. They do well on OoD detection tasks on various datasets, including the most difficult ones, while remaining competitive predictive accuracy. We introduce a novel uncertainty factorization, and it is quite general and semantically accurate. It also sheds light on what types of models are capable of detecting OoD samples.

References

[1] https://tiny-imagenet.herokuapp.com/. 7

[2] C. Andrieu and Jordan. An introduction to mcmc for machine learning. In *Machine Learning, 50, 5–43*, 2003. 6

[3] Peter L. Bartlett, Steven N. Evans, and Philip M. Long. Representing smooth functions as compositions of near-identity functions with implications for deep network optimization. In *Journal of Machine Learning Research*, 2017. 5

[4] Jens Behrmann, Will Grathwohl, Ricky TQ Chen, David Duvenaud, and Jorn-Henrik . Jacobsen. Invertible residual networks. In *ICML*, 2019. 2, 5

[5] Adi Ben-Israel. A volume associated with m x n matrices. In *Linear Algebra and its Applications Volume 167*, 1992. 4

[6] Adi Ben-Israel. The change-of-variables formula using matrix volume. In *SIAM Journal on Matrix Analysis and Applications 21(1)*, 1999. 4

[7] L. Bergman and Y. Hoshen. Classification-based anomaly detection for general data. In *International Conference on Learning Representations*, 2020. 6

[8] C. M. Bishop. Novelty detection and neural network validation. In *IEE Proceedings-Vision, Image and Signal processing, 141(4):217–222)*, 1994. 1, 6

[9] D. M. Blei and A. Kucukelbir. Variational inference: A review for statisticians. In *Journal of the American Statistical Association*, 2017. 6

[10] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra. Weight uncertainty in neural networks. In *ICML*, 2015. 6

[11] Johann Brehmer and Kyle Cranmer. Flows for simultaneous manifold learning and density estimation. In *NeurIPS*, 2020. 2, 4

[12] Andrew Brock, Soham De, amuel L. Smith, and Karen Simonyan. High-performance large-scale image recognition without normalization. In *ICLR 2021*, 2021. 1

[13] Nicola Cao, Wilker Aziz, and Ivan . Titov. Block neural autoregressive flow. In *Uncertainty in Artificial Intelligence, pp. 1263–1273. PMLR*, 2020. 2

[14] Ricky TQ Chen, Jens Behrmann, David K Duvenaud, and Jorn-Henrik . Jacobsen. Residual flows for invertible generative modeling. In *NeurIPS*, 2019. 2

[15] Ricky T. Q. Chen, Jens Behrmann, David Duvenaud, and Henrik Jacobsen. Residual flows for invertible generative modeling. In *Advances in Neural Information Processing Systems*, 2019. 2

[16] Travers Ching, Daniel S. Himmelstein, Brett K. Beaulieu-Jones, Alexandr A. Kalinin, and Brian T. Do. Opportunities and obstacles for deep learning in biology and medicine. In *http://doi.org/10.1098/rsif.2017.0387*, 2018. 1

[17] H. Choi, E. Jang, and A. Alemi. Waic, but why? generative ensembles for robust anomaly detection. In *arXiv:1810.01392*, 2018. 6

[18] Sungik Choi and Sae-Young Chung. Novelty detection via blurring. In *ICLR*, 2020. 6

[19] Stefan Depeweg, Jose Miguel Hernandez-Lobato, Finale Doshi-Velez, and Steffen Udluft. Decomposition of uncertainty in bayesian deep learning for efficient and risk-sensitive learning. In *ICML*, 2018. 3

[20] Feng Di, Rosenbaum Lars, and Dietmayer Klaus. Towards safe autonomous driving: Capture uncertainty in the deep neural network for lidar 3d vehicle detection. In *https://arxiv.org/pdf/1804.05132.pdf*, 2018. 1

[21] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp. In *ICLR*, 2017. 2

[22] Y. Du and I Mordatch. Implicit generation and modeling with energy based models. In *Advances in Neural Information Processing Systems*, 2019. 6

[23] Conor Durkan, Artur Bekasov, Iain Murray, and George. Papamakarios. Neural spline flows. In *NeurIPS*, 2019. 2

[24] Sergey Edunov, Myle Ott, Michael Auli, and David Grangier. Understanding back-translation at scale. In *Conference of the Association for Computational Linguistics (ACL)*, 2018. 1

[25] S. Farquhar, M. Osborne, and Y. Gal. Radial bayesian neural networks: Beyond discrete support in large-scale bayesian deep learning. In *International Conference on Artificial Intelligence and Statistics*, 2020. 6

[26] Herbert Federer. Geometric measure theory. In *Classics in Mathematics. Springer-Verlag Berlin Heidelberg*, 1969. 4

[27] Ethan Fetaya, Jörn-Henrik Jacobsen, Will Grathwohl, and Richard Zemel. Understanding the limitations of conditional generative models. In *ICLR*, 2020. 6

[28] Y. Gal and Z. Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *ICML*, 2016. 6

[29] Mevlana C. Gemici, Danilo J. Rezende, and Shakir Mohamed. Normalizing flows on riemannian manifolds. In *https://arxiv.org/abs/1611.02304*, 2016. 2, 4

[30] I. Golan and R. El-Yaniv. Deep anomaly detection using geometric transformations. In *Advances in Neural Information Processing Systems*, 2018. 6

[31] H. Gouk, E. Frank, B. Pfahringer, and M Cree. Regularisation of neural networks by enforcing lipschitz continuity. In *https://arxiv.org/pdf/1804.04368.pdf*, 2018. 5

[32] Will Grathwohl, Ricky TQ Chen, Jesse Betterncourt, Ilya Sutskever, and David . Duvenaud. Ffjord: Free-form continuous dynamics for scalable reversible generative models. In *ICLR*, 2019. 2

[33] W. Grathwohl, K.-C. Wang, D.and Norouzi M. Jacobsen, J.-H.and Duvenaud, and K. Swersky. Your classifier is secretly an energy based model and you should treat it like one. In *International Conference on Learning Representations*, 2020. 6

[34] A Graves. Practical variational inference for neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2011. 6

[35] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. In *NeurIPS*, 2017. 6

[36] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks. In *ICML*, 2017. 6

[37] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 7

[38] Dan Hendrycks, Mantas Mazeika, and Thomas Dietterich. Deep anomaly detection with outlier exposure. In *ICLR*, 2019. 6

[39] D. Hendrycks, M. Mazeika, S. Kadavath, , and D. Song. Using self-supervised learning can improve model robustness and uncertainty. In *Advances in Neural Information Processing Systems*, 2019. 6

[40] G. E. Hinton and D. van Camp. Keeping the neural networks simple by minimizing the description length of the weights. In *Sixth Annual Conference on Computational Learning Theory*, 1993. 6

[41] Jonathan Ho, Xi Chen, Aravind Srinivas, Yan Duan, and Pieter Abbeel. Flow++: Improving flowbased generative models with variational dequantization and architecture design. In *ICML*, 2019. 2

[42] Zhang Hongjie, Ang Li, Jie Guo, and Yanwen Guo. Hybrid models for open set recognition. In *ECCV*, 2020. 2, 6, 7

[43] Christian Horvat and Jean-Pascal Pfister. Density estimation on low-dimensional manifolds: an inflation-deflation approach. In *https://arxiv.org/pdf/2105.12152.pdf*, 2020. 2, 4

[44] Tommi S. Jaakkola and David Haussler. Exploiting generative models in discriminative classifiers. In *Advances in Neural Information Processing Systems (NeurIPS)*, 1999. 1, 6

[45] Levinson et al Jesse. Towards fully autonomous driving: Systems and algorithms. In *Intelligent Vehicles Symposium (IV), IEEE. IEEE. 2011, pp. 163–168*, 2011. 1

[46] Durk Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. In *NeurIPS*, 2018. 2

[47] D. P. Kingma, S. Mohamed, D. J. Rezende, and M. Welling. Semi-supervised learning with deep generative models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2014. 1, 6

[48] Polina Kirichenko, Pavel Izmailov, and Andrew Gordon Wilson. Why normalizing flows fail to detect out-of-distribution data. In *Neural Information Processing Systems (NeurIPS)*, 2020. 2, 4

[49] Alex Krizhevsky. Learning multiple layers of features from tiny images. 2009. 6

[50] Volodymyr Kuleshov and Stefano Ermon. Deep hybrid models: bridging discriminative and generative approaches. In *Uncertainty in Artificial Intelligence*, 2017. 1, 6

[51] B. Lakshminarayanan, A. Pritzel, and C. Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Conference on Neural Information Processing Systems (NIPS)*, 2017. 6

[52] S. Larson, A. Mahendran, J. J. Peper, C. Clarke, A. Lee, P. Hill, J. K. Kummerfeld, K. Leach, M. A. Laurenzano, L. Tang, and J. . Mars. An evaluation dataset for intent classification and out-of-scope prediction. In *arXiv: 1909.02027.*, 2019. 7

[53] J. A. Lasserre, C. M. Bishop, and T. P. Minka. Principled hybrids of generative and discriminative models. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition,*, 2006. 1, 6

[54] Kimin Lee, Honglak Lee, Kibok Lee, and Jinwoo . Shin. Training confidence-calibrated classifiers for detecting out-of-distribution samples. In *ICLR*, 2018. 6

[55] Kimin Lee, Kibok Lee, Honglak Lee, and Jinwoo Shin. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. In *NeurIPS*, 2018. 6

[56] Shiyu Liang, Yixuan Li, and R. Srikant. Enhancing the reliability of out-of-distribution image detection in neural networks. In *ICLR*, 2018. 6

[57] H. Liu and P. . Abbeel. Hybrid discriminative-generative training via contrastive learning. In *arXiv:2007.09070*, 2020. 6

[58] Jeremiah Liu, Zi Lin, Shreyas Padhy, Dustin Tran, Tania Bedrax Weiss, and Balaji Lakshminarayanan. Simple and principled uncertainty estimation with deterministic deep learning via distance awareness. In *Neural Information Processing Systems (NeurIPS)*, 2020. 5, 6, 7

[59] Jeremiah Liu, John Paisley, Marianthi-Anna Kioumourtzoglou, and Brent Coull. Accurate uncertainty estimation and decomposition in ensemble learning. In *NeurIPS*, 2019. 3

[60] Jeremiah Zhe Liu, John Paisley, Marianthi-Anna Kioumourtzoglou, and Brent A. Coull. Accurate uncertainty estimation and decomposition in ensemble learning. In *Neural Information Processing Systems (NeurIPS)*, 2019. 3

[61] C. Louizos and M Welling. Multiplicative normalizing flows for variational bayesian neural networks. In *ICML*, 2017. 6

[62] D. J. MacKay. A practical bayesian framework for backpropagation networks. In *Neural computation, 4(3):448–472,*, 1992. 6

[63] Radek Mackowiak, Lynton Ardizzone, Ullrich Kothe, and Carsten Rother. Generative classifiers as a basis for trustworthy image classification. In *https://arxiv.org/pdf/2007.15036.pdf*, 2020. 6

[64] W. J. Maddox and A. G. Wilson. A simple baseline for bayesian uncertainty in deep learning. In *Conference on Neural Information Processing Systems*, 2019. 6

[65] Iwao Maeda, Hiroyasu Matsushima, Hiroki Sakaji, Kiyoshi Izumi, David deGraw, Atsuo Kato, and Michiharu Kitano. Effectiveness of uncertainty consideration in neural-network-based financial forecasting. In *International Congress on Advanced Applied Informatics (IIAI-AAI)*, 2019. 1

[66] Ahsan Mahmood, Junier Oliva, and Martin Andreas Styner. Multiscale score matching for out-of-distribution detection. In *ICLR*, 2021. 6

[67] Andrey Malinin and Mark Gales. Predictive uncertainty estimation via prior networks. In *Neural Information Processing Systems*, 2018. 3, 6

[68] Emile Mathieu and Maximilian Nickel. Riemannian continuous normalizing flows. In *NeurIPS*, 2020. 2, 4

[69] A. McCallum, C. Pal, G. Druck, and M. Wang, X. Multiconditional learning: Generative / discriminative training for clustering and classification. In *AAAI*, 2006. 1, 6

[70] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. In *ICLR*, 2018. 2, 5

[71] E. Nalisnick, A. Matsukawa, Y. W. Teh, D. Gorur, and B Lakshminarayanan. Do deep generative models know what they don't know? In *International Conference on Learning Representations,*, 2019. 6

[72] Eric Nalisnick, Akihiro Matsukawa, Yee Whye Teh, Dilan Gorur, and Balaji Lakshminarayanan. Hybrid models with deep and invertible features. In *Proceedings of the 36th International Conference on Machine Learning*, 2019. 2, 6

[73] E. Nalisnick, A. Matsukawa, Y. W. Teh, and B. Lakshminarayanan. Detecting out-of-distribution inputs to deep generative models using a test for typicality. In *arXiv:1906.02994*, 2018. 6

[74] R. M. Neal. Bayesian learning for neural networks. In *Springer Volume 118*, 1996. 6

[75] B Neyshabur, S Bhojanapalli, D McAllester, and N Srebro. Exploring generalization in deep learning. In *arXiv:1706.08947*, 2017. 5

[76] Druck G. Pal, C., X. Zhu, and A. McCallum. Semi-supervised classification with hybrid generative/discriminative methods. In *Proceedings of the 13th ACM SIGKDD*, 2007. 1, 6

[77] Rajat Raina, Yirong Shen, Andrew Y. Ng, and Andrew McCallum. Classification with hybrid generative discriminative models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2004. 1, 6

[78] J. Ren, P. J. Liu, E. Fertig, J. Snoek, R. Poplin, M. Depristo, J. Dillon, and B Lakshminarayanan. Likelihood ratios for out-of-distribution detection. In *Advances in Neural Information Processing Systems*, 2019. 6

[79] Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *Proceedings of the 32nd International Conference on Machine Learning*, 2015. 2

[80] Hippolyt Ritter, Aleksandar Botev, and David Barber. A scalable laplace approximation for neural networks. In *ICLR*, 2018. 6

[81] R. Schirrmeister, Y. Zhou, T. Ball, and D. Zhang. Understanding anomaly detection with deep invertible networks through hierarchies of distributions and features. In *NeurIPS*, 2020. 6

[82] J. Serrà, D. Álvarez, V. Gómez, O. Slizovskaia, J. F. Núñez, and J Luque. Input complexity and out-of-distribution detection with likelihood-based generative models. In *International Conference on Learning Representations*, 2020. 6

[83] L. Smith and Y. Gal. Understanding measures of uncertainty for adversarial example detection. In *arXiv:1803.08533v1*, 2018. 3

[84] Yang Song, Chenlin Meng, and Stefano Ermon. Mintnet: Building invertible neural networks with masked convolutions. In *NeurIPS*, 2019. 2

[85] M. Szummer and T. S. Jaakkola. Information regularization with partially labeled data. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2003. 1, 6

[86] Jihoon Tack, Sangwoo Mo, Jongheon Jeong, and Jinwoo Shin. Csi: Novelty detection via contrastive learning on distributionally shifted instances. In *Advances in Neural Information Processing Systems*, 2020. 6

[87] Ryutaro Tanno, Daniel E. Worrall, Enrico Kaden, Aurobrata Ghosh, Francesco Grussu, Alberto Bizzi, Stamatios N. Sotiropoulos, Antonio Criminisi, and Daniel C. Alexander. Uncertainty quantification in deep learning for safer neuroimage enhancement. In *arXiv:1907.13418*, 2019. 1

[88] Joost Van Amersfoort, Lewis Smith, Yee Whye Teh, and Yarin Gal. Uncertainty estimation using a single deep deterministic neural network. In *Proceedings of the 37th International Conference on Machine Learning*, 2020. 6

[89] M. Welling and Y. W. Teh. Bayesian learning via stochastic gradient langevin dynamics. In *ICML*, 2011. 6

[90] Y. Wen, P. Vicol, J. Ba, D. Tran, and R. Grosse. Flipout: Efficient pseudo-independent weight perturbations on mini-batches. In *ICLR*, 2018. 6

[91] R. Winkens, J.and Bunel, A. G. Roy, R. Stanforth, V. Natarajan, J. R. Ledsam, P. MacWilliams, P. Kohli, A. Karthikesalingam, and S. Kohl. Contrastive training for improved out-of-distribution detection. In *arXiv:2007.05566*, 2020. 6

[92] A. Wu, S. Nowozin, E. Meeds, R. E. Turner, J. M. Hernandez-Lobato, and A. L. Gaunt. Deterministic variational inference for robust bayesian neural networks. In *ICLR*, 2019. 6

[93] Pingmei Xu, Krista A Ehinger, Yinda Zhang, Adam Finkelstein, Sanjeev R Kulkarni, and Jianxiong Xiao. Turkergaze: Crowdsourcing saliency with webcam based eye tracking. In *arXiv:1504.06755, 2015.*, 2019. 7

[94] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. Xlnet: Generalized autoregressive pretraining for language understanding. In *arxiv.org/abs/1906.08237*, 2019. 6, 7

[95] Fisher Yu, Ari Seff, Yinda Zhang, Shuran Song, Thomas Funkhouser, and Jianxiong . Xiao. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. In *arXiv:1506.03365, 2015.*, 2019. 7

[96] S. Zagoruyko and N. . Komodakis. Wide residual networks. In *arXiv: 1605.07146.*, 2017. 6, 7

[97] Y. Zhai, S.and Cheng, W. Lu, and Z Zhang. Deep structured energy based models for anomaly detection. In *International Conference on Machine Learning*, 2016. 6

[98] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. Mixup: Beyond empirical risk minimization. In *ICLR*, 2018. 6

[99] Yu Zhang, James Qin, Daniel S Park, Wei Han, ChungCheng Chiu, Ruoming Pang, Quoc V Le, and Yonghui Wu. Pushing the limits of semisupervised learning for automatic speech recognition. In *arXiv preprint arXiv:2010.10504.*, 2020. 1

[100] Zhen Zhang, Jiajun Bu, Martin Ester, Jianfeng Zhang, Chengwei Yao, Zhi Yu, and Can Wang. Hierarchical graph pooling with structure learning. *arXiv preprint arXiv:1911.05954*, 2019. 1