Uncertainty, Distributional Robustness, and Anomaly Detection: Fundamentals, Current Research, and Open Problems

Stephan Rabanser
Department of Computer Science
University of Toronto
stephan@cs.toronto.edu

January 10, 2022

1 Introduction

In recent years, machine learning (ML) has seen a number of truly revolutionary breakthroughs in the fields of natural language processing, computer vision, and generative modeling, especially attributed to the surging popularity of deep neural nets. However, despite their power, many state-of-the-art machine learning models and algorithms are still extremely brittle, which prevents wide-scale deployment of ML techniques into high-stakes real-life decision-making-systems in sensitive areas like finance, transportation, and healthcare. Recently, researchers and practitioners have identified a set of concrete problems inhibiting real-world applicability of ML models. Specifically, many models fail to properly account for predictive uncertainty, are overconfident in low-evidence regions, break frequently under distribution shift, and are unable to reliably detect anomalous inputs.

In this position paper, we explore the state-of-the art in these ML research fields and highlight a limited set of frontiers and problems warranting further research.

2 Uncertainty

One key limitation of many off-the-shelf ML algorithms stems from insufficient or improper uncertainty quantification. In particular, in order for a model prediction to be considered trustworthy, it is crucial that the prediction is legitimately supported by the training data and made with high confidence. Unfortunately, many ML models lack this guarantee and produce high-confidence decision on unrecognizable, deliberately manipulated, or out-of-distribution inputs.

The need for proper uncertainty estimation is critical in many sub-fields of machine learning. Most notably, high-quality uncertainty scores are essential for decision making (especially in critical high-stakes domains such as clinical healthcare or automated driving), forecasting, learning from partially observable, missing, or limited data, anomaly detection, trading off exploration and exploitation in reinforcement learning (RL), selective classification, activate learning, as well as model understanding/explainability. Having access to such uncertainty scores enables ML algorithms to honestly acknowledge the limits of their knowledge, identify doubtful predictions, pinpoint differences between the training and deployment domains, and potentially completely abstain from prediction. Hence, to build trustworthy ML algorithms, it is crucial to consider uncertainty quantification a first-class citizen.

2.1 The Basics

2.1.1 Standard Supervised Learning Setup

Consider the following standard supervised learning setup. We are given a dataset $D_p = \{(x_i, y_i)\}_{i=1}^N$ with data points (x, y) sampled from distribution p, formally $(x, y) \sim p$. We further define the dataset space as $\mathcal{D} = \mathcal{X} \times \mathcal{Y}$ with $x \in \mathcal{X}$,

$y \in \mathcal{Y}$. Our goal is to learn a suitable prediction function $h_\theta : \mathcal{X} \to \mathcal{Y}$ producing labels $\hat{y} = h_\theta(x)$. In particular, we want to produce a prediction function $h_\theta(\cdot)$ performing well on unseen test data $D'_p = \{(x_j, y_j)\}_{j=1}^M$, $(x, y) \sim p$, $D'_p \cap D_p = \varnothing$ as measured by a loss function $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$. The overall learning objective is then to tune the prediction function h_θ such that the expected loss over the entire data distribution p is minimized. This concept is referred to as (true) risk minimization: $\min_\theta \mathcal{R}(h_\theta) \coloneqq \mathbb{E}_{p(x,y)}[\ell(h_\theta(x), y)] = \int_{\mathcal{Y}} \int_{\mathcal{X}} p(x, y)\ell(h_\theta(x), y)dxdy$.

Unfortunately, $p(x, y)$ is typically not known or intractable to compute and as a result $\mathcal{R}(h_\theta)$ cannot be computed. It is however possible to empirically approximate $\mathcal{R}(h_\theta)$ as $\hat{\mathcal{R}}(h_\theta)$ using samples from $p(x, y)$ (i.e. using D_p). This procedure is commonly referred to as empirical risk minimization (ERM): $\min_\theta \hat{\mathcal{R}}(h_\theta) \coloneqq \frac{1}{N} \sum_{i=1}^N \ell(h_\theta(x_i), y_i)$.

2.1.2 Probabilistic Learning

In order to introduce the concept of predictive uncertainty, it is now our goal to no longer model the prediction problem as deterministic but to take a probabilistic viewpoint on the learning problem instead. Specifically, we are interested in modifying the prediction function $h_\theta(\cdot)$ to fully account for uncertainty by (i) placing a distribution on the output variable y; and (ii) by also modeling model internals such as model parameters in a probabilistic fashion. As a result, both a prediction $\mathbb{E}[y|x]$ and the associated uncertainty $\text{Var}[y|x]$ are returned by the ML algorithm.

Computing uncertainties requires a rigorous probabilistic treatment of all variables involved. This can be achieved using the Bayesian inference framework by modeling the likelihood $p(y|x, \theta)$, the prior $p(\theta)$, the posterior $p(\theta|x, y)$, and the predictive distribution $p(y|x)$ using Bayes rule and the law of total probability:

$$\text{Posterior: } p(\theta|x, y) = \frac{p(y|x, \theta)p(\theta)}{\int p(y|x, \theta)p(\theta)d\theta} \qquad \text{Predictive distribution: } p(y|x) = \int \underbrace{p(y|x, \theta)}_{\text{aleatoric}} \underbrace{p(\theta|x, y)}_{\text{epistemic}} d\theta$$

Unfortunately, exact computation of the posterior and predictive distribution are computationally intractable in many applications. This is due to large high-dimensional datasets, millions of parameters, and complex non-convex loss landscapes. To address this limitation there is often a need to rely on approximate methods, some of which are introduced in Section 2.2.

Dissecting Predictive Uncertainty While analyzing the scenarios that can lead to uncertainty within a ML algorithm, we can identify two distinct types of uncertainty:

- *Aleatoric uncertainty*: This uncertainty type stems from noise present the data itself and is typically caused by a large degree of variability in the data collection process. This type of uncertainty is typically considered to be irreducible, which means that collecting more data does not help to reduce uncertainty[1]. Aleatoric uncertainty modeling is achieved through the likelihood $p(y|x, \theta)$.

- *Epistemic uncertainty*: This uncertainty type stems from an imperfect modeling choice (e.g. model parameters, model architecture, model class). This type of uncertainty is typically considered to be reducible, which means that collecting more data does indeed help to reduce uncertainty. Epistemic uncertainty modeling is achieved through the posterior $p(\theta|x, y)$. Current research in ML mostly focusses on epistemic uncertainty.

2.2 Popular Uncertainty Quantification Techniques

Variational Bayesian Approximations Bayes by Backprop, as introduced in **Blundell et al. (2015)**, provides an approximate solution to the posterior over the weights of a deep neural network through the application of variational inference (VI). To that end, the authors restrict the weight posterior to a Gaussian distribution and train the neural network to minimize the KL divergence between the Gaussian variational posterior $q(\theta|w)$ and the true posterior $q(\theta|x, y)$. This yields a cost function known as the variational free energy: $\mathcal{F}(\theta, x, y) := \text{KL}[q(\theta|w)||q(\theta)] - \mathbb{E}_{q(\theta|w)}[\log p(y|x, \theta)]$.

[1] Reducing noise stemming from the data collection process itself can help to reduce aleatoric uncertainty, though.

This loss encodes an interpretable tradeoff between satisfying the complexity of the data set D_p and the simplicity of the prior $p(\theta)$.

Unfortunately, exactly minimizing this cost is computationally intractable. The authors therefore suggest to use Monte Carlo sampling to approximate the cost function $\mathcal{F}(\theta, x, y) \approx \sum_i \log q(\theta_i|w) - \log p(\theta_i) - \log p(y|x, \theta_i)$ where θ_i corresponds to the i-th sample from the variational posterior: $\theta_i \sim q(\theta|w)$. Although other works had previously explored similar approximations, the this particular choice of approximation yields unbiased gradients during backpropagation. Moreover, the presented solution is not closed form and therefore allows for flexible choices of the posterior and prior beyond Gaussians. To that end, the authors motivate the usage of a scale-mixture prior of two zero-centered Gaussians with different scales. This particular choice of prior allows for heavier tails and encourages a-priori concentration of the weights around zero at the same time.

Training the Bayesian neural network then involves both multiple forward and a backward passes. In the forward pass, a noise term is sampled from a standard Gaussian $\epsilon \sim \mathcal{N}(0, I)$ and transformed using the learned mean μ and scale ρ parameters over the weights[2]: $\theta_i = \mu + \log(1 + \exp(\rho)) \cdot \epsilon$. After computing the empirical variational free energy, backpropagation with respect to the variational parameters $w = (\mu, \rho)$ yields the respective gradient update rules.

The experimental results demonstrate strong classification performance on MNIST and improved uncertainty quantification in low-data regions. It is noteworthy that employing Bayes by Backprop does not just yield predictive uncertainty but it also provides regularization through the introduction of a prior over the weights. Moreover, their evaluation shows that the signal-to-noise-ratio (which can be computed for each weight) allows for effective weight pruning: The experimental results suggest that removing the noisiest 98% of all weights (leaving only the 2% of the weights with the highest signal-to-noise-ratio) does not cause a significant degradation in predictive performance. However, since the core uncertainty estimation technique in Bayes by Backprop is based on variational inference, the well known limitations of VI apply. In particular, the variance of the predictive distribution is underestimated, the posterior is biased towards only approximating one single mode of the weight distribution, the uncertainties are sensitive to the particular choice of prior, and the approximation procedure can still be computationally slow to train.

Deep Ensembling In **Lakshminarayanan et al. (2017)**, the authors introduce deep ensembles as a scalable way for non-Bayesian uncertainty quantification. The core idea consists of three steps:

1. *Train the neural network through the maximization of a proper scoring rule (capturing aleatoric uncertainty).* A scoring rule (Gneiting and Raftery, 2007) is a numerical quantization of the calibration of a predictive distribution $p(y|x)$. If the underlying true distribution over data points is denoted $q(x, y)$, then the expected scoring rule is defined as $S(p, q) = \mathbb{E}[S(p, (x, y))]$ for a scoring function $S(p, (x, y))$. A proper scoring rule is a rule which ensures that $S(p, q) \leq S(q, q)$ with equality iff $p(y|x) = q(y|x)$. The authors establish that many popular neural network losses are proper scoring rules. In particular, the log likelihood in both regression and classification (softmax) is a proper scoring rule with score function $S(p, (x, y)) = \log p(y|x)$.

2. *Employ adversarial training (Goodfellow et al., 2014) to smooth the predictive distribution.* By incorporating adversarial examples generated using the fast-gradient-sign-method into the training stage, the likelihood for a given sample x is increased in its ϵ-neighborhood. While smoothing the loss in all directions would be desirable, this smoothing effect occurs in the direction of worst-case loss and therefore provides the largest gains if smoothing is limited to one single direction.

3. *Repeat this process M-many times to yield an ensemble of neural networks (capturing epistemic uncertainty).* The randomization between different members of the ensemble is restricted to the initialization of the parameters of the neural network, along with random shuffling of the data points[3]. In contrast to VI-based methods, this specific choice of randomization ensures that different regions of the complex multi-modal loss landscape are explored. The

[2] This is an application of the re-parameterization trick (Kingma et al., 2015) to separate the sources of randomness from the parameters to be learned.

[3] Hence, neither a classical bagging nor boosting strategy are applied.

predictive mixture distribution is then given by $p(y|x) = \frac{1}{M}\sum_{m=1}^{M} p(y|x, \theta_m)$. For ease of computing quantiles of the predictive distribution, the ensemble parameterizes a Gaussian distribution with the mean and variance of the ensemble mixture.

Experiments performed on synthetic regression problems and standard image datasets (MNIST, SVHN, ImageNet) show comparable or better accuracy and negative log likelihood while significantly improving calibration on out-of-distribution samples (MNIST vs NotMNIST). Although simple to implement, Deep Ensembles also come with some drawbacks. While the computation can be parallelized, they still require multiple complete training runs, which might be prohibitively expensive in some scenarios. Moreover, the inference stage also has to be performed for each model independently. Finally, as the ensemble does not directly yield a predictive distribution but merely multiple possible outcomes, it is still necessary to approximate the distribution (typically in a parametric form) to yield an expectation-variance decomposition of the predictive distribution.

2.3 Frontiers in Uncertainty Quantification

Quality of Posteriors & Priors Many Bayesian deep learning works do not use the true Bayes posterior but instead make use of a cold posterior, which involves re-scaling the posterior by a temperature parameter T with $0 \leq T \leq 1$. A temperature of 1 yields the true posterior while a temperature of 0 corresponds to the maximum a-posteriori estimate. This is usually needed to improve raw predictive performance as results obtained using stochastic gradient descent are hard to beat for current Bayesian methods. While cooling the posterior has its empirical advantages, **Wenzel et al. (2020)** argues that this finding points to a deeper problem in current Bayesian deep learning: What is the use of a more accurate approximation of the posterior if the posterior gives poor results? The authors first present empirical evidence for this finding on both a ResNet-20 on CIFAR-10 and CNN-LSTM model for IMDB text classification. In both cases, $T \ll 1$ yields optimal test set performance. The remainder of the paper surveys multiple potential sources of the cold posterior problem, such as:

1. *Improper assumptions for the inference procedure.* The inference procedure presented in this work is based on Stochastic Gradient Markov Chain Monte Carlo (SG-MCMC), a recent adaptation of traditional MCMC which utilizes data subsampling techniques to reduce the per-iteration cost of MCMC. The generation of approximate parameters from the posterior $p(\theta|x, y)$ is based on Langevin dynamics over the parameters θ. Simulating this system of stochastic differential equations can yield two sources of error: (i) mini-batch noise over the gradients of the posterior energy function; and (ii) discretization errors incurred during discrete-time approximations in the dynamics. The authors use techniques inspired by layer-wise preconditioning and cyclical time stepping from related work to mitigate these sources of error. A detailed analysis on this inference procedure concludes that (i) the SDE simulation is accurate (measured in agreement of configurational and kinetic temperatures); that (ii) SG-MCMC is unbiased (measured in agreement of HMC and SG-MCMC); and that (iii) multiple choices for the mini-batch size all show best performance at $T < 1$.

2. *Dirty likelihoods due to batch-normalization, dropout, or data augmentation.* These techniques could alter the likelihood in potentially unintended ways. However, an experimental evaluation performed by adding or removing these techniques still shows the presence of the cold posterior effect, even with clean likelihoods.

3. *Poor choices of prior.* Having eliminated all other options, the paper claims that the prior is responsible for the cold posterior effect. In particular, the typical standard multivariate Gaussian prior $p(\theta) = \mathcal{N}(0, I)$ is unintentionally informative and intensifies the cold posterior effect with increasing model depth and width. This is investigated in the paper by demonstrating that this choice of prior produces concentrated class distributions. Increasing the variance of the Gaussian was also not found to mitigate this effect.

Overall, the investigation suggests that the neither (1) nor (2) have been found to significantly contribute to the cold posterior effect but that typically a poor choice of prior can be blamed. While the analysis presented in this work is of

vital importance to understand the performance guarantees of Bayesian neural networks, the authors were not able to fully resolve nor fix the problem from cold posteriors. Hence, additional research is needed to better align the practical benefits Bayesian methods with their theoretical guarantees.

Ground Truth & Unified Framework For Evaluation As noted in Abdar et al. (2021) and Gawlikowski et al. (2021), the lack of ground truth uncertainties, the inability to test on single instances, and standardized benchmarking protocols are currently inhibiting the establishment of a unified framework for benchmarking competing approaches for uncertainty modeling. While detecting unreliable predictions on the WILDS dataset (Koh et al., 2021) can be insightful, additional metrics such as calibration and confidence intervals would be more desirable. The fact that alignment of ML predictive uncertainty and human uncertainty can be hard to establish due to the subjectivity of the latter further complicates this issue.

3 Distributional Shifts & Out-of-Distribution Samples

It is crucial for a trustworthy ML algorithm to be robust to perturbations of its inputs. This issue is typically discussed in the context of adversarial examples, data samples with small but deliberately crafted feature perturbations that lead the machine learning model to make a false prediction (Szegedy et al., 2013). However, ML models have been proven to fail in real-life scenarios without the presence of an adversary. Two instances of this problem are (i) distribution shifts; and (ii) out-of-distribution samples.

3.1 The Basics

3.1.1 Distribution Shifts

The key question distribution shift detection aims to answer is whether the data generating distribution of two distinct environments is in fact the same. Formally, we wish to determine whether the source distribution p over data points $\boldsymbol{x} \in \mathcal{X}$ and labels $y \in \mathcal{Y}$ matches the target distribution q over data points $\boldsymbol{x}' \in \mathcal{X}$ and labels $y' \in \mathcal{Y}$, formally $p(\boldsymbol{x}, y) \stackrel{?}{=} q(\boldsymbol{x}', y')$. Since test data is unlabeled in real-life settings, it is usually not possible to directly assess whether the joint distribution between data points and true labels differs between the two environments. Instead, we often only have estimates $\hat{y}'$ of the true label y' at our disposal, altering the problem formulation as follows: $p(\boldsymbol{x}, y) \stackrel{?}{=} q(\boldsymbol{x}', \hat{y}')$. Note that if data was collected in an unsupervised fashion (or when we simply intend to analyze the data in an unsupervised manner), the shift detection question simplifies to $p(\boldsymbol{x}) \stackrel{?}{=} q(\boldsymbol{x}')$.

Basic Shift Types We introduce the most basic shift types as introduced in Quiñonero-Candela et al. (2009).

- *Covariate shift*: Under the covariate shift model, only the distribution over covariates $\boldsymbol{x}$ (i.e. the input features) changes between the two domains, i.e. $p(\boldsymbol{x}) \neq q(\boldsymbol{x})$, while the conditional label distribution remains fixed, i.e. $p(y|\boldsymbol{x}) = q(y|\boldsymbol{x})$.
- *Label shift*: Under the label shift model, only the distribution over labels y changes between the two domains, i.e. $p(y) \neq q(y)$, while the conditional covariate distribution remains fixed, i.e. $p(\boldsymbol{x}|y) = q(\boldsymbol{x}|y)$.

More complex shifts include concept drift, sample selection bias, domain shift, source component shift, subpopulation shift, and open-set extension (Quiñonero-Candela et al., 2009).

3.1.2 Anomalous/Out-of-Distribution Samples

Reliable ML models should be able to detect when an input into a machine learning model is anomalous. As per the independent-and-identically-distributed (iid) assumption from statistical learning theory, arbitrary performance degradation is possible for data points that do not come from the original training distribution. Most importantly,

since performance monitoring is limited in practice due to the absence of labels, erroneous decisions due to anomalous points can remain unnoticed for extended periods of time. It is therefore critical to validate such inputs in order to ensure that predictions are only generated for faithful inputs. Specifically, assuming that a machine learning model was trained on distribution p, we wish to detect samples x coming from a different distribution q, i.e. $x \sim q$, and flag them as anomalous. Unfortunately, classical input validation techniques as frequently proposed in software engineering are based on hard-coded thresholding rules which are infeasible to translate to high-dimensional data settings.

3.2 Tackling Distribution Shifts

We briefly discuss two ideas for addressing two essential problems encountered when dealing with distribution shifts: (i) detecting distributional changes; and (ii) ensuring invariance to certain shifts.

3.2.1 Shift Detection Through Statistical Hypothesis Testing

As noted by Quiñonero-Candela et al. (2009), there is an inherent need for methods to first detect the mere presence of distribution shift and, as a consequence, characterize the shift in a second step. While methods explicitly developed to address covariate or label shift are usually favorable to use since their explicit assumption allows them to derive mathematically sound algorithms for addressing shift, they can lead to false predictions if these assumptions are not met. Also, pure covariate or label shifts rarely occur in real life settings, which is why techniques designed to correct for only one type of shift has only limited applicability in practice. At the same time, it would be desirable for a general purpose detection scheme to seamlessly fit into the existing machine learning pipeline. **Rabanser et al. (2018)** presents a general shift detection method fulfilling these desiderata. They propose a framework for shift detection that is based on statistical hypothesis testing of dimensionality-reduced data representations. While statistical hypothesis testing can be considered a mature science on univariate or low-dimensional data, best practices for detecting shift in high-dimensional, real-life shifts have been hard to establish. Although kernel-based testing techniques are popular approaches to detect such differences, their practical usefulness is limited by the dataset size and their power decays drastically in high-dimensional spaces.

The paper first revises popular dimensionality reduction techniques, such as principal components analysis, sparse random projections, both trained and untrained autoencoders, and label classifiers with soft and hard-tresholded labels[4]. Moreover, a domain classifier is introduced, whose purpose is to explicitly detect shift by finding a discriminatory boundary between samples from the source domain and from the target domain. Next, depending on which exact dimensionality reduction technique is used, a suitable statistical test is chosen. Multi-dimensional representations can be tested for shift either using the maximum mean discrepancy (MMD) or univariate testing using the Kolmogorov-Smirnov (KS) test with subsequent multiple hypothesis correction. For the hard-thresholded label classifier, a χ^2-test is chosen, while a binomial test is recommended for the accuracy of the domain classifier.

The comprehensive empirical evaluation of synthetic perturbations applied on top of MNIST and CIFAR-10 finds that univariate testing (and subsequent Bonferroni correction) of the softmax representations of a neural network classifier yields the strongest shift detector (measured in shift detection accuracy) on as few test samples as possible. The applied shifts consist of adversarial perturbations, removing a certain percentage of samples from a particular class, random image rotations and (x, y)-coordinate translation, as well as combinations of various label and covariate shifts. The domain classifier performs very poorly in the low-data regime, but offers additional qualitative advantages. In particular, the domain classifier can score individual points on their anomalousness as it returns a decision on how confidently these points are assigned either to the target or source domain. This enables visual inspection of the shift by creating a ranking of the samples that are most confidently assigned to either of the two domains. Assessing the accuracy on these top confident samples can further act as a proxy for shift malignancy. One additional curious finding is presented in evaluating the default MNIST train/test split, which appears to exhibit a dataset shift.

[4]Since the softmax output layer is typically of much lower dimension than the input domain this approach can indeed be considered a dimensionality reduction technique.

While the proposed method allows for shift detection, shift quantification and shift malignancy are only briefly discussed and addressed using preliminary approaches with strong assumptions. Future work needs to prioritize shift characterization and the effects of performance degradation of the underlying ML model.

3.2.2 Shift Invariance Through Distributionally Robust Optimization (DRO)

While detecting shifts is an important task, our end goal is to ensure that an ML model is robust towards the encountered shift. Recall the standard supervised learning setup described in Section 2.1.1. One central assumption in this setup is the stationarity of the underlying data distribution p: Our model should generalize well to samples from the same distribution encountered at training time. However, this assumption is hard to satisfy in practice. In a more realistic scenario, we assume that the test time distribution q is not exactly the same but still close to the training distribution p. We formalize this setup as follows: $D'_q = \{(x_j, y_j)\}_{j=1}^{M}$ with $(x, y) \sim q,\ 0 \leq d(p, q) \leq \delta$ and $D'_q \cap D_p = \varnothing$.

Note that $d(p, q)$ is a divergence measure between p and q and is bounded by δ. The exact choice of divergence measure $d(\cdot, \cdot)$ typically either comes from (i) the family of Integral Probability Metrics (e.g. Wasserstein Distance, Maximum Mean Discrepancy, Dudley Metric); or (ii) the family of ϕ-divergences (e.g. Kullback-Leibler Divergence, Pearson χ^2 Divergence, Jensen-Shannon Divergence).

In the distributionally robust optimization (DRO) framework (Bagnell, 2005), instead of optimizing for good test time performance on the original distribution p, we optimize for good test time performance under the worst possible shift. Hence, our risk minimization problem turns into: $\min_\theta \mathcal{R}(h_\theta, q) \coloneqq \max_{q \in \mathcal{Q}_p} \mathbb{E}_{q(x,y)}[\ell(h_\theta(x), y)]$. $\mathcal{Q}_p$ is called our uncertainty set of data distributions. We constrain distribution realizations $q \in \mathcal{Q}_p$ to be absolutely continuous and bounded in divergence wrt p: $\mathcal{Q}_p = \{q \ll p \mid d(p, q) \leq \delta\}$.

3.3 OOD Detection Methods

Deep k-Nearest Neighbors In **Papernot and McDaniel (2018)**, the authors present an augmented inference procedure for an already trained deep neural network. The core idea is to harness the latent activations for a given input at each layer and perform a nearest neighbor analysis in each of the induced latent spaces. The intuition behind this idea is that legitimate inputs from a specific class should be close to points from the same class in all latent representations. Any discrepancy in the neighborhood structure in the latent spaces could point to a noteworthy deviation that should be detected. The nearest neighbor search is conducted with respect to a validation set and the purity of the neighborhood acts as a nonconformity score. These scores are then aggregated across layers to derive a credibility score, which can be used as a basis of rejecting illegitimate samples. The empirical evaluation demonstrates improved calibration over the base network and low-confidence assignment to OOD samples. Moreover, this new inference procedure is capable of detecting adversarial examples, a particularly hard case of OOD detection. This is corroborated by the finding that adversarial examples are close to representations from the true class in earlier layers while switching to the adversarial class in later layers. The authors further propose an adaptive attack on DkNN and demonstrate that generating adversarial examples for DkNN requires semantic-breaking perturbations. One key advantage of inferring the neighborhood structure is that it allows for example-based interpretability: To explain the decision of a neural network to a human, the k-nearest neighbors in each layer could be returned to the user as visual evidence of the model's decision. This is studied in the paper by examining a ResNet model's bias and to identify mislabelled examples in the MNIST and SVHN training sets.

Follow-up work presented in Frosst et al. (2019) discards the assumption of a trained but fixed base neural network. Instead, the authors propose to tune the latent representations using the soft nearest neighbor loss. They establish the counter-intuitive finding that representations should not be disentangled at every latent layer but should instead be entangled at lower layers and disentangled at higher layers. Entangling lower layers leads to better generalization ability (as the network is supposed to extract common patterns across classes), while disentangling higher layers leads to better linear separation. As a byproduct of enforcing either entanglement or disentanglement through the soft

nearest neighbor loss, the authors further demonstrate that the yielded representations are better suited to detect ambiguous/OOD data points. Outlier inputs that do not share any commonalities with the learned class clusters will be projected off the data manifold and form their own distinct clusters, making OOD data points easier to identify as a result.

Deep Mahalanobis Similar to the DkNN approach, **Lee et al. (2018)** also investigates hidden representations for OOD detection. The authors argue for a probabilistic approach towards analyzing the latent spaces induced by each layer. Specifically, they suggest to parameterize each class in each hidden hidden layer via a multivariate Gaussian distribution. This yields a generative classifier consisting of C-many class conditional Gaussian distributions with tied covariance matrices. Assume that the latent representation of an input x at layer l is given by $f_l(x)$, then the density of data points for a fixed class c is modeled by $p(f_l(x)|y=c) = \mathcal{N}(f_l(x)|\mu_{c,l}, \Sigma_l)$. Estimation of both $\hat{\mu}_{c,l} \approx \mu_{c,l}$ and $\hat{\Sigma}_l \approx \Sigma_l$ can be done from the training set by computing the empirical mean and empirical covariance of the data's representation at layer l. Making use of the class conditional Gaussian distributions, the Mahalanobis score at layer l is then given by $M_l(x) = \max_c -(f_l(x) - \hat{\mu}_{c,l})^\top \hat{\Sigma}_l^{-1}(f_l(x) - \hat{\mu}_{c,l})$ and the Mahalanobis classification result consequently by $\hat{y}_l(x) = \arg\min_c \ (f_l(x) - \hat{\mu}_{c,l})^\top \hat{\Sigma}_l^{-1}(f_l(x) - \hat{\mu}_{c,l})$. Using this classifier alone on the penultimate layer was already found to outperform the typical softmax classifier and Euclidian-based distance metrics. Borrowing an idea from ODIN (Liang et al., 2017), the authors also suggest pre-processing the inputs using adversarial-like perturbations as follows: $\hat{x} = x + \epsilon \cdot \text{sign}(\nabla_x M_L(x))$ where L is the pre-softmax layer. This pre-processing step is designed to make differentiating in- and out-of-distribution easier. Intuitively, in-distribution inputs should suffer from large variability in the softmax scores (since the neural network was explicitly trained to maximize likelihood in this region) while OOD inputs should show less variability. The overall anomaly score for a given input can then be derived by computing the maximum per-class Mahalanobis distance in each layer and aggregating said score across layers: $M(x) = \sum_l \alpha_l M_l(x)$.

Empirical results performed on CIFAR, SVHN, ImageNet, and LSUN using DenseNet and ResNet architectures demonstrate the effectiveness of the Mahalanobis score by consistently outperforming competing OOD and adversarial example detection approaches such as ODIN (Liang et al., 2017) and LID (Ma et al., 2018). Detection performance is assessed via detection accuracy, area under the receiver operating characteristic (AUROC), and the true negative rate at a fixed true positive rate of 95%. These results point at the fact that employing the Mahalanobis distance over the Euclidian distance appears significantly more effective. This is not surprising, though, since the covariance information present in Mahalanobis harnesses correlation information of the data manifold and can hence be considered an instance of metric learning. One limitation of this work however is the fact that the mixture coefficients α need to be tuned on a validation set containing OOD data, the latter of which is not explicitly acknowledged in the paper.

3.4 Frontiers in Distribution Shifts and OOD Sample Detection

The Role of Uncertainty in Distribution Shifts One would hope that models that are able to account for different notions of uncertainty should also be able to detect that they are doing worse on distributionally shifted data. Recently, **Ovadia et al. (2019)** presented a study of how uncertainty quantification methods (such as the ones introduced in Section 2.2) behave under distribution shift. In general, they found that a lot of uncertainty estimation techniques do in fact not realize that they are doing worse in the presence of shifts and produce overconfident decisions as a result. Before diving into the results, the authors first introduce the methods (maximum softmax probability, post-hoc calibration using temperature scaling, Monte Carlo dropout, deep ensembles, Bayes by Backprop, as well as last-layer stochastic variational inference and last layer dropout) and metrics (predictive accuracy, negative log likelihood, Brier score, and expected calibration error) of their empirical study.

Overall, the empirical evaluation yields the finding that post-hoc calibration techniques perform badly under distribution shift while ensembling methods, such as Deep Ensembles, still perform comparatively well. For variational Bayesian approximations, the verdict is mixed. This is first demonstrated on MNIST with shifted data obtained via the application of perturbations and corruptions to the covariates. While increasing the perturbation strength, accuracy

decreases as expected; but so do the uncertainty metrics. Most notably, the authors find that post-hoc calibration on an iid validation set does not lead to well-calibrated predictions on OOD data. Stunningly, some experiments show that explicit calibration on an iid validation set might actually hurt calibration under distribution shift. The same take-away message can be derived from more large-scale experiments on CIFAR-10 and ImageNet. Specifically, diminishing accuracy due to distribution shift also degrades the Brier score and expected calibration error. The results also show that the relative ranking of methods is mostly consistent throughout the experiments.

In general, this finding further corroborates the problem of uncertainty measures degrading under model misspecification: While the presented uncertainty estimation methods work well under the iid assumption, even slight departures from this assumption can render uncertainty scores meaningless. Hence, uncertainty quantification under dataset shift is still worth worrying about. In particular, we are wondering whether (i) we can identify an uncertainty quantification technique that accounts for the presence of shifts; and, if such a method exists, whether (ii) such a model could implicitly function as a distribution shift detector or already be invariant to shift by design.

Deep Generative Models Are Not Reliable OOD Detectors Deep generative models (DGM) are widely believed to being able to reliably detect anomalous/out-of-distribution samples. This common belief stems from the fact that DGMs model the density of the input features $p(x)$ in order to produce faithful samples from the training distribution. Detecting an anomalous point would then involve rejecting inputs that have sufficiently low density under $p(x)$. **Nalisnick et al. (2018)** challenges the suitability of DGMs for OOD detection by demonstrating that flow-based models, variational autoencoders, and autoregressive models are not able to reliably differentiate between samples from popular image datasets. In fact, when using a DGM trained on CIFAR-10, SVHN inputs, i.e. out-of-distribution samples, are often assigned higher likelihood under the generative model than in-distribution samples. Similar findings are reported on CelebA and ImageNet.

The paper mainly investigates this counter-intuitive finding from the viewpoint of flow-based generative models. First, the authors provide some background on how neural generative models can be trained and introduce the basics of flow-based generative models. The specific model of interest in this paper is the Glow model introduced in Kingma and Dhariwal (2018). Investigating the average bits per dimension on a Glow model trained on FashionMNIST/CIFAR-10, the authors find that the MNIST/SVHN dataset is encoded using fewer bits than the FashionMNIST/CIFAR-10 dataset. Comparing the log likelihoods in both settings paints a similar picture: the OOD dataset is assigned higher likelihood that in-distribution data. Even more strikingly, the same effect arises in the PixelCNN and VAE models.

To further study this phenomenon, the authors dig deeper into the properties of flow-based models. In particular, invertible models allow for the computation of exact marginal likelihoods and impose specific constraints on the Jacobian that simplify analysis. First, the change-of-variables objective is decomposed into $\log p(z)$ and $\log |\frac{\partial f}{\partial x}|$ for non-volume preserving Glow. While the latent distribution $p(z)$ term behaves as expected, yielding lower likelihood for OOD data, the volume evaluations $\log |\frac{\partial f}{\partial x}|$ are larger for out-of-distribution data than for in-distribution data. Overall, the volume term dominates in magnitude and therefore causes overall larger OOD likelihood. The authors also find that constant inputs achieve the highest likelihood under the flow and that neither changing the flow to be volume preserving nor robustifying the likelihood assignment using ensembling helps to mitigate the observed effect.

Continuing this line of work, Kirichenko et al. (2020) further investigates the limitations of normalizing flows. The authors demonstrate that flows latch onto local pixel correlations and generic image-to-latent-space transformations which are not specific to the semantics of the target image dataset. However, determining whether an input is out-of- or in-distribution, is typically considered a problem of semantics and not so much a problem of local pixel correlations. Since off-the-shelf normalizing flows possess this pixel correlation bias, they fail at being reliable OOD detectors. To address this limitation, the authors show that modifying the architecture of flow coupling layers imposes a bias on the flow towards learning the semantic structure of the target data, thereby improving OOD detection as a result. This line of work motivates the importance of further understanding the inductive biases of state-of-the-art generative models.

More Natural/Subtle OOD Samples The Deep Mahalanobis OOD detection framework described in Section 3.3 achieves excellent performance on the tested benchmark OOD sets. Since this particular testing setup is conducted between sets which usually exhibit very stark semantical differences (e.g. SVHN vs CIFAR-10), there is growing concern in the ML community that current benchmarks overestimate the progress made in anomaly detection. In fact, OOD detection is still far from solved for more subtle perturbations. In **Madras and Zemel (2021)**, the authors argue for a more explicit definition of out-of-context (OOC) prediction problems. Specifically, they argue that the criterion and auxiliary information by which OOC samples are selected need to be clarified in more detail. The authors present a framework by which naturally-occurring OOC problems can be identified within a pre-existing dataset.

The task of interest is the binary decision of determining object presence within a scene. This is facilitated through the usage of the COCO dataset (Lin et al., 2014), a richly annotated dataset that contains multiple object labels, their corresponding bounding boxes, and image captions. The fact that multiple different OOC criterions can be applied to the same dataset enables the creation of both easier and harder OOC benchmark datasets. Two exemplary criterions are given: (i) the presence/absence of frequently co-occurring objects in a scene; and (ii) an unusual gist of a scene. The co-occurrence score is determined by estimating the area gap between the object class and the context (computable thanks to bounding boxes) and averaging this area over all instances of the object class. On the other hand, the gist of a scene is encoded in its caption and feeding these captions to a BERT language model yields a multidimensional embedding for each caption. Determining the unusualness of a scene in general can then be determined by computing the cosine similarity of the current input and the mean embedding of a specific class. Relying on these criterions enables the creation of two types of OOC examples: (i) hard positives, where the object class in question is present despite an unusual context; and (ii) hard negatives, where the class is not present, despite a usual context.

These strategies for deriving hard OOC samples enable the authors to distill a new dataset dubbed NOOCE for OOC prediction, which is evaluated in an empirical study. Specifically, performance of empirical risk minimization (focusing on generalization to the same environment), group distributionally robust optimization (focusing on generalization to the worst sub-group), and invariant risk minimization (focusing on generalization across domains) are reported on this dataset as well as on the waterbirds dataset. The evaluation shows that theses newly derived examples are indeed hard to detect with unusual gist scenes being particularly difficult to detect as such. Moreover, models that are better on hard examples tend to be worse on easy examples (and vice versa). Another finding is that access to auxiliary context information is more important than the particular choice of invariance algorithm. While the current empirical evaluation is already comprehensive, in its current state, this work still has room for improvement. For instance, the proposed approaches currently seem highly specialized to the COCO dataset (and its labeling strategy) and it is somewhat unclear how these approaches could be effectively applied to other datasets.

4 Conclusion

While we have seen recent advances in making ML algorithms more trustworthy, current approaches for uncertainty quantification, addressing distributional shift, as well as detecting – and potentially rejecting – out-of-distribution inputs are still insufficient, especially in high-stakes decision-making pipelines. Calibrated uncertainty scores are often too expensive to obtain and current approximations suffer from inherent biases that still need to be better understood. Also properly evaluating the quality of predictive uncertainty remains and open issue. Distributional shift is ubiquitous in real-life deployed systems and can often be detected. Unfortunately, characterization and correcting ML models in the presence of shifts both remain challenging, which limits the practical applicability of the presented approaches. Finally, state-of-the-art out-of-distribution sample detection seems to suggest that current techniques are able to reliably handle anomalous inputs. However, many OOD approaches still operate in a supervised fashion, requiring explicit access to OOD data during validation. Moreover, there is growing concern that these methods are overestimating progress due to the simplicity of the de-facto standardized benchmarking setup.

References

M. Abdar, F. Pourpanah, S. Hussain, D. Rezazadegan, L. Liu, M. Ghavamzadeh, P. Fieguth, X. Cao, A. Khosravi, U. R. Acharya, et al. A review of uncertainty quantification in deep learning: Techniques, applications and challenges. *Information Fusion*, 2021.

J. A. Bagnell. Robust supervised learning. In *AAAI*, pages 714–719, 2005.

C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra. Weight uncertainty in neural network. In *International Conference on Machine Learning*, pages 1613–1622. PMLR, 2015.

N. Frosst, N. Papernot, and G. Hinton. Analyzing and improving representations with the soft nearest neighbor loss. In *International Conference on Machine Learning*, pages 2012–2020. PMLR, 2019.

J. Gawlikowski, C. R. N. Tassi, M. Ali, J. Lee, M. Humt, J. Feng, A. Kruspe, R. Triebel, P. Jung, R. Roscher, et al. A survey of uncertainty in deep neural networks. *arXiv preprint arXiv:2107.03342*, 2021.

T. Gneiting and A. E. Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American statistical Association*, 102(477):359–378, 2007.

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

D. P. Kingma and P. Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. *arXiv preprint arXiv:1807.03039*, 2018.

D. P. Kingma, T. Salimans, and M. Welling. Variational dropout and the local reparameterization trick. *Advances in neural information processing systems*, 28:2575–2583, 2015.

P. Kirichenko, P. Izmailov, and A. G. Wilson. Why normalizing flows fail to detect out-of-distribution data. *arXiv preprint arXiv:2006.08545*, 2020.

P. W. Koh, S. Sagawa, H. Marklund, S. M. Xie, M. Zhang, A. Balsubramani, W. Hu, M. Yasunaga, R. L. Phillips, I. Gao, et al. Wilds: A benchmark of in-the-wild distribution shifts. In *International Conference on Machine Learning*, pages 5637–5664. PMLR, 2021.

B. Lakshminarayanan, A. Pritzel, and C. Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in neural information processing systems*, 30, 2017.

K. Lee, K. Lee, H. Lee, and J. Shin. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. *Advances in neural information processing systems*, 31, 2018.

S. Liang, Y. Li, and R. Srikant. Enhancing the reliability of out-of-distribution image detection in neural networks. *arXiv preprint arXiv:1706.02690*, 2017.

T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

X. Ma, B. Li, Y. Wang, S. M. Erfani, S. Wijewickrema, G. Schoenebeck, D. Song, M. E. Houle, and J. Bailey. Characterizing adversarial subspaces using local intrinsic dimensionality. *arXiv preprint arXiv:1801.02613*, 2018.

D. Madras and R. Zemel. Identifying and benchmarking natural out-of-context prediction problems. In *Thirty-Fifth Conference on Neural Information Processing Systems*, 2021.

E. Nalisnick, A. Matsukawa, Y. W. Teh, D. Gorur, and B. Lakshminarayanan. Do deep generative models know what they don't know? *arXiv preprint arXiv:1810.09136*, 2018.

Y. Ovadia, E. Fertig, J. Ren, Z. Nado, D. Sculley, S. Nowozin, J. V. Dillon, B. Lakshminarayanan, and J. Snoek. Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift. *arXiv preprint*

arXiv:1906.02530, 2019.

N. Papernot and P. McDaniel. Deep k-nearest neighbors: Towards confident, interpretable and robust deep learning. *arXiv preprint arXiv:1803.04765*, 2018.

J. Quiñonero-Candela, M. Sugiyama, N. D. Lawrence, and A. Schwaighofer. *Dataset shift in machine learning*. Mit Press, 2009.

S. Rabanser, S. Günnemann, and Z. C. Lipton. Failing loudly: An empirical study of methods for detecting dataset shift. *arXiv preprint arXiv:1810.11953*, 2018.

C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

F. Wenzel, K. Roth, B. S. Veeling, J. Świątkowski, L. Tran, S. Mandt, J. Snoek, T. Salimans, R. Jenatton, and S. Nowozin. How good is the bayes posterior in deep neural networks really? *arXiv preprint arXiv:2002.02405*, 2020.