Ph.D. Dissertation

Energy-Based Probabilistic Models for Epistemic Uncertainty Quantification

인식론적 불확실성 정량화을 위한 에너지 기반 확률 모형

August 2023

Department of Mechanical Engineering
Seoul National University

Sangwoong Yoon

Energy-Based Probabilistic Models for Epistemic Uncertainty Quantification

인식론적 불확실성 정량화을 위한 에너지 기반 확률 모형

지도교수 박 종 우

이 논문을 공학박사 학위논문으로 제출함

2023 년 4 월

서울대학교 대학원

기계공학부

윤 상 웅

윤상웅의 공학박사 학위논문을 인준함

2023 년 6 월

위 원 장	김 아 영	(인)
부위원장	박 종 우	(인)
위 원	양 인 순	(인)
위 원	이 준 석	(인)
위 원	김 세 훈	(인)

Abstract

Energy-Based Probabilistic Models for Epistemic Uncertainty Quantification

Sangwoong Yoon
Department of Mechanical Engineering
Seoul National University

Probability is a principled measure of uncertainty. Particularly, the probability of input data $p(x)$ in machine learning serves as a useful measure of epistemic uncertainty. The accurate estimation of the data probability can benefit multiple epistemic uncertainty quantification applications, such as detecting anomalous samples, avoiding over-confident predictions, and obtaining informative unlabeled samples. However, recent deep learning approaches for estimating the input data probability, often referred to as deep generative modeling, have shown limited uncertainty quantification capabilities, despite their impressive performance in synthesizing realistic samples. The failure of deep generative modeling has spurred skepticism on whether the modeling the data probability is a valid approach for capturing uncertainties.

In this thesis, we demonstrate that modeling the probability of data is essential for uncertainty quantification. We focus on autoencoders and Gaussian processes, popular algorithms for anomaly detection and decision-making under uncertainty, and show that incorporating generative modeling improves their performance. Both algorithms are non-generative and do not include input data probability in their formulation, leading to critical failure modes due to not

correctly reflecting the data probability distribution. We address these failure modes by introducing novel probabilistic formulations with generative modeling for autoencoders and Gaussian processes. The proposed methods are based on the energy-based model framework, which defines a probability distribution using an unnormalized scalar function called energy, where we introduce novel designs of energy functions built from autoencoders and Gaussian processes.

While introducing novel interpretations of autoencoders and Gaussian processes as generative models, we introduce additional contributions regarding energy-based modeling. First, we present a novel training algorithm for energy-based generative models that leverages the low-dimensional structure of data. The proposed algorithm can effectively suppress spuriously high likelihood in generative models and the resulting energy-based models show strong anomaly detection performance. Second, we also investigate the connection between adversarial attack and energy-based model formulation. We propose a generative adversarial attack algorithm for out-of-distribution detectors where the attack is formulated as sampling from an energy-based distribution.

Quantifying epistemic uncertainty is essential for robots and artificial intelligence agents to interact with the world safely and effectively. This paper demonstrated that the problem of quantifying epistemic uncertainty can be solved through probabilistic models. The energy-based probabilistic model techniques and algorithms discussed in this paper are expected to be widely applied to various applications in robotics and artificial intelligence systems.

Keywords: Generative Models, Energy-Based Models, Uncertainty Quantification, Anomaly Detection, Density Estimation, Epistemic Uncertainty

Student Number: 2020-38989

Contents

List of Tables

List of Figures

Chapter 1

Introduction

We can even say, strictly speaking, that almost all our knowledge is only probable.
—Pierre-Simon Laplace,
Théorie analytique des probabilités (1812)

Generative modeling, a task of learning the underlying probability distribution from observed data, is a fundamental problem in statistics and machine learning. The most popular and successful application of generative modeling has been the synthesis task, where the goal is to draw novel examples from the learned distribution. In recent years, substantial progress has been made in the synthesis application through the incorporation of large-scale deep neural network models and vast amounts of data. These deep generative models exhibit remarkable performance in generating realistic artificial sensory signals, such as images and texts.

Besides the synthesis task, another important application of generative models is uncertainty quantification. In essence, probability measures the level of surprise after observing a data point. Thus, an accurately estimated probability can offer valuable information about how eccentric or informative a given observation is. Leveraging such information is critical in modern real-world machine learning, where reliability and adpatibility of a machine learning system is of

high value.

Unfortunately, modern deep generative models struggle to perform uncertainty quantification effectively, despite their astonishing synthesis quality. They either are not capable of evaluating the probability of a datum, such as GAN or diffusion models, or assigns spuriously high likelihood to outliers, for instance VAE, normalizing flows, and autoregressive models. Some might question the effectiveness of generative approach for uncertainty quantification and even argued that the probability density is no longer an effective measure of uncertainty.

In this dissertation, we aim to demonstrate that generative modeling is still an essential component for effective uncertainty quantification. Our approach consists of three steps.

1. We find a well-established uncertainty quantification algorithm that does not incorporate generative modeling in its formulation.

2. We look for the systematic failure cases in uncertainty quantification task of the algorithm.

3. We introduce a novel generative formulation for the algorithm that can resolve the failure cases.

Using this strategy, we analyze autoencoders, a popular anomaly detection algorithm, and Gaussian processes, an algorithm that is widely used in applications where uncertainty-aware decisions are made. Both algorithms are not generative models and their formulations does not include any procedure of estimating the probability of data, at least explicitly.

Interestingly, we were able to find critical failure modes in both autoencoders and Gaussian processes. An autoencoder-based anomaly detection algorithm is

supposed to produce a large reconstruction error for outliers. However, we find that there is outlier reconstruction phenomenon, where autoencoders giving an unexpectedly small reconstruction error for an obvious outliers. In Gaussian processes, the predictive variance serves as the measure of uncertainty that is used for a number of downstream tasks, such as active learning and Bayesian optimization. We find that the predictive variance is often over-smoothed, giving uninformative uncertainty quantification useless in downstream applications. The oversmoothing is caused by the hyperparameters only optimized for the conditional likelihood of labels $p(y|x)$ and ignoring the likelihood of input data $p(x)$.

To remedy these failures, we propose improvements of these algorithms through a novel view of them as generative models. We employ the energy-based model framework where a probability distribution is defined from an unnormalized scalar function called energy. The energy-based model allows a great flexibility in the design of a probabilistic model through the choice of the energy function, and we leverage the flexibility to design novel energy functions for autoencoders and Gaussian processes. For autoencoders, we propose Normalized Autoencoders (NAE), an energy-based model with the reconstruction error as the energy function. Outlier reconstruction is significantly For Gaussian processes, we propose Generative Gaussian processes, also an energy-based model with the predictive variance as the energy. The resulting algorithms, normalized autoencoders and generative Gaussian processes, effectively resolves the problems of outlier reconstruction and over-smoothed variance.

This thesis makes additional contributions to anomaly detection by extending the probabilistic view on autoencoder-based anomaly detection. First, we propose a novel training algorithm for energy-based models that leverages the low-dimensional structure of data. The proposed algorithm, Manifold

projection-diffusion Recovery (MPDR), first perturbs a training datum along a pre-defined manifold and then trains the energy function through the process of recovering the original data from the perturbed data point. We show that the energy functions trained by MPDR is particularly effective at anomaly detection and MPDR is highly compatible with NAE. Second, we investigate the problem of measuring the robustness of an anomaly detection algorithm. We propose a novel type of adversarial attack that is formulated as sampling from an energy-based model defined on a manifold of plausible samples. Adversarial Generation on Manifold (AGOM) method is able to discover interesting and previously unexplored failure modes of state-of-the-art anomaly detection algorithms.

Chapter 2

Generative Modeling and Epistemic Uncertainty Quantification

Humans and animals can be surprised, but you can not surprise a computer, yet. In fact, the ability to be surprised reflects a notable level of intelligence. A surprisal is a result of inference judging whether a sensory signal is within expectation, and this inference requires an internal statistical model built from the past experience. However, modern machine learning systems have limited ability to construct an accurate model of the sensory inputs, unable to recognize what is surprising.

Building a statistical model of the world is the ultimate goal of **generative modeling**, which construct a probabilistic distribution model $P(X)$ from a finite set of empirical observations $\{X_i\}_{i=1}^N$. With an accurate generative model, we can measure how expected a signal $\mathbf{x}$ is by computing the probability density $p(\mathbf{x})$ or log-density $\log p(\mathbf{x})$, where a smaller value indicates a larger degree of surprisal.

Such a surprising signal is said to have high **epistemic uncertainty**, which means there is limited previous experience regarding that signal. This uncertainty is originated from the lack of experience, and therefore is different from **aleatoric uncertainty**, caused by the inherent randomness of the data generation process. There can be multiple ways to quantify epistemic uncertainty.

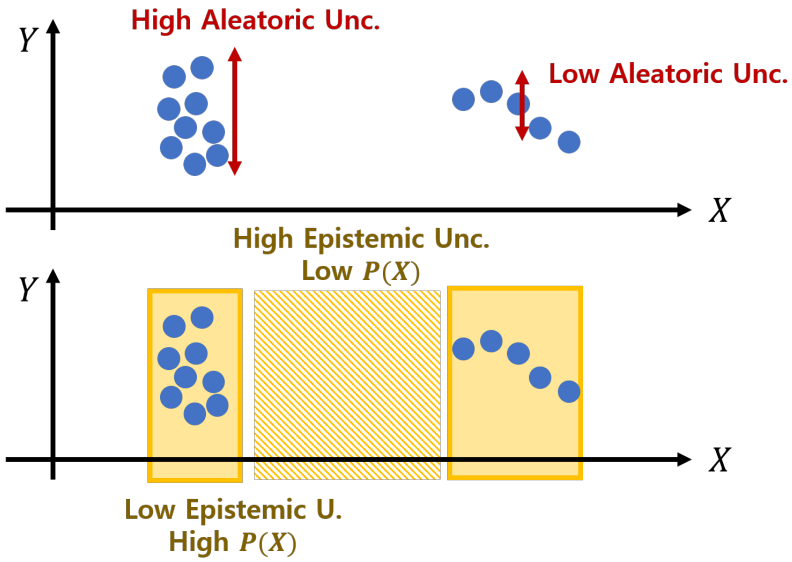

Figure 2.1 Illustrations of aleatoric uncertainty and epistemic uncertainty.

However, since the probability estimated by a generative model naturally measures how surprising a signal is, the probability is a natural measure of epistemic uncertainty.

In this chapter, we discuss the tight relationship between epistemic uncertainty and probability in detail.

2.1 Uncertainty Quantification in Machine Learning

The term *uncertainty quantification* covers a wide range of scientific methodologies for characterizing the uncertainties in observations or computations. Uncertainty quantification plays an important role in diverse fields of science and engineering, particularly when the risk of being wrong needs to be managed.

Prediction Problem Here, we focus on a specific setting which is relevant to typical machine learning scenarios. Consider a prediction problem, where

we want to predict outcome random variable Y from our observation on input random variable X using a predictor $Y = f_\theta(X)$.

$$X \xrightarrow{f_\theta} Y. \tag{2.1}$$

The predictor f_θ has a parameter θ that has to be determined from previous observations $\mathcal{D} = \{(X_i, Y_i)\}_{i=1}^N$ through the training phase. During the testing phase, we are asked to provide the prediction for a newly observed input.

Uncertainty of Y In machine learning, what we are typically interested in is the uncertainty of our prediction on Y given a new input X. The uncertainty of the prediction can be naturally quantified as the expected prediction error. The larger the uncertainty is, the larger the prediction error we expect.

The uncertainty of the prediction, i.e., the potential prediction error, can be originated from two sources, which give us two categories of uncertainties. The first source is the inherent randomness involved in the process $X \to Y$. For example, an outcome of a coin toss or a die roll is inherently random. This type of uncertainty is called **aleatoric uncertainty**. Even in a deterministic world, a process can be observed to be stochastic when there are unobserved variables affecting Y. The aleatoric uncertainty can be represented as $P(Y|X)$, where this probability indicates the true probability in the data generating process.

The second origin of uncertainty is the fact that we have only a finite number of data. This uncertainty is called **epistemic uncertainty**. With an infinite number of data, the predictor f_θ can converge to the optimal predictor which achieves the smallest possible error caused by aleatoric uncertainty. As the training dataset size becomes smaller than the infinity, we expect larger errors in prediction.

If we view the uncertainty as expected error, it becomes immediately clear that the accurate estimation of the uncertainty is very difficult. Uncertainty

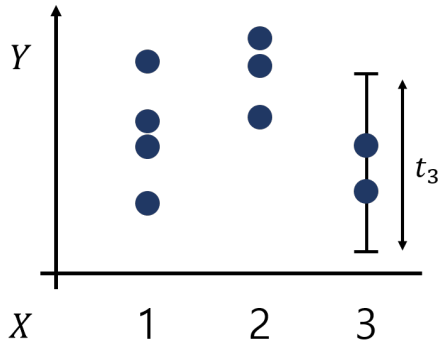

Figure 2.2 An illustration of the discrete input space regression example.

quantification is equivalent to estimation of test error, which is only possible under strong assumptions. Therefore, for practical epistemic uncertainty quantification, we need a proxy measure that approximately captures the uncertainty.

2.2 Probability is Relative Epistemic Uncertainty

In this section, we show that the input data probability $P(X)$ is a good proxy for quantifying epistemic uncertainty. In fact, what $P(X)$ reflects is *relative* epistemic uncertainty. Given two points X_1 and X_2, We can tell X_1 is higher in epistemic uncertainty than X_2 if $P(X_1)$ is lower than $P(X_2)$. It is our intuition that tells us the prediction will be relatively more accurate near our training data and it is more likely to be wrong on X that is far from our training data. In the following, we show an example where the relationship between relative epistemic uncertainty and the input data probability can be derived exactly.

Example: Discrete Input Space Consider a regression problem where the input variable has a discrete set of possible values, for example, $X \in \{1, 2, 3\}$.

Each X is associated with bounded $Y \in [0, 1]$. We are interested in building a predictor of Y and quantifying its epistemic uncertainty. This setting is illustrated in Fig. 2.2.

In this setting, empirical mean is a natural choice for the predictor, as it provides an unbiased estimate. Suppose we have n_k number of observations For $X = k$. The empirical mean is then given as $\mu_k = \frac{1}{n_k} \sum_{i=1}^{n_k} Y_k^{(i)}$, where $Y_k^{(i)}$ is Y value of i-th observation.

The epistemic uncertainty of estimation can be represented by a confidence interval, and Hoeffding's inequality is a popular method for computing a confidence interval of mean estimation. Hoeffding's inequality gives the interval around the estimated mean where the true mean is located with the high probability. If we write the width of a confidence interval for prediction at $X = k$ as t_k, Hoeffding's inequality is written as follows.

$$P\left(\left|\frac{1}{n_k}\sum_{i=1}^{n_k} Y_k^{(i)} - \mu_k\right| > \frac{t_k}{2}\right) \leq 2\exp(-n_k t_k^2/2). \tag{2.2}$$

where μ_k is the true mean. Let us denote the right-hand side of the inequality as δ. Then, we can express the relationship between the number of observation n_k and the confidence interval width t_k.

$$2\exp(-n_k t_k^2/2) = \delta, \tag{2.3}$$

$$n_k t_k^2 = 2\log(2/\delta), \tag{2.4}$$

$$t_k = \sqrt{\frac{2\log(2/\delta)}{n_k}} = \sqrt{\frac{2\log(2/\delta)}{N \cdot P(X = k)}}, \tag{2.5}$$

where $P(X = k) = n_k/N$ is the marginal probability estimate for $X = k$ and $N = \sum n_k$ is the total number of observations. We can see that the relationship between the confidence interval width and the input data probability: $t_k \propto N^{-1/2} P(X = k)^{-1/2}$. From this relationship, we can see that $P(X = k)$

determines *relative* epistemic uncertainty. Altering $P(X = k)$ while N fixed changes the relative sizes of confidence intervals. Meanwhile, increasing N while fixing $P(X = k)$ values for all k will narrow confidence intervals for all X's.

Chapter 3

Normalized Autoencoders: A Probabilistic View on Autoencoder-Based Anomaly Detection

3.1 Introduction

An autoencoder [3] is a neural network trained to reconstruct samples from a training data distribution. Since in principle the quality of reconstruction is expected to be poor for inputs that deviate significantly from the training data, autoencoders are widely used in outlier detection [4], in which an input with a large reconstruction error is classified as out-of-distribution (OOD). Autoencoders for outlier detection have been applied in domains ranging from video surveillance [5] to medical diagnosis [6].

However, autoencoders have been known to reconstruct outliers consistently across a wide range of experimental settings [7, 8, 9, 10]. Figure 3.1 shows further examples of some outliers reconstructed by an autoencoder trained with MNIST data; the autoencoder is able to reconstruct a wide range of OOD inputs, including constant black pixels, Omniglot characters, and fragments of MNIST digits. The early works on regularized autoencoders [11, 12, 13] focus for the most part on preventing the autoencoder from turning into an identity mapping that reconstructs every input. Nonetheless, outlier reconstruction can

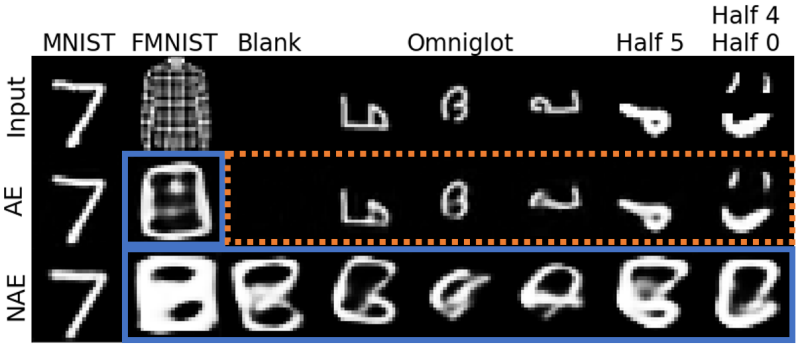

Figure 3.1 Examples of reconstructed outliers. The last two rows show the reconstructions from a conventional autoencoder (AE) and NAE. Both autoencoders are trained on MNIST, and other inputs are outliers. The architecture of the two autoencoders is identical. Successful detection of an outlier is highlighted with blue solid rectangles, while detection failures due to the reconstruction of outliers are denoted with an orange dotted rectangle. Note that AE is not the identity mapping, as it fails to reconstruct the shirt.

still occur even when the autoencoder is not the identity. Not surprisingly, outlier reconstruction is a leading cause of autoencoder's detection failure.

On the other hand, in a normalized probabilistic model, it is known that maximum likelihood learning suppresses the assignment of probability mass in OOD regions in order to keep the model normalized. Thus, the likelihood is widely used as a predictor for outlier detection [14]. An autoencoder does not have such a suppression mechanism that inhibits the reconstruction of an OOD input, because an autoencoder moreover is also not a generative model of the data. Therefore, the reconstruction error of an autoencoder usually lacks a meaningful probabilistic interpretation.

This paper formulates an autoencoder as a normalized probabilistic model to introduce a mechanism for preventing outlier reconstruction. In our formulation, which we call the **Normalized Autoencoder (NAE)**, the reconstruction error is re-interpreted as an energy function, i.e., the unnormalized negative log-density, and defines a probabilistic model from an autoencoder. During maximum likelihood learning of the model, normalization constraint is naturally enforced by increasing the reconstruction error of samples generated from the model. When the model distribution deviates from the data distribution, the generated samples can have low reconstruction error but are from OOD regions. Since the reconstruction of the samples from OOD regions are suppressed, NAE is significantly less prone to reconstruct outliers, as shown in Figure 3.1.

Samples used in training of NAE are generated via Markov Chain Monte Carlo (MCMC). As running MCMC until convergence for every training step is computationally infeasible, approximate sampling strategies have to be employed. However, we observe that training with popular sampling strategies such as Contrastive Divergence (CD; [15]) and Persistent CD (PCD; [16]) may

often produce poor density estimates. For improved training, we propose **on-manifold initialization (OMI)**, a method of selecting a good initial state for the MCMC chain in NAE. OMI draws an initial state in a high-density region by leveraging the manifold structure learned by an autoencoder. Trained with high-quality samples generated by OMI, NAE can accurately recover the data density and thus become an effective outlier detector.

Intriguingly, although technically a normalized probabilistic model, the variational autoencoder (VAE; [17]) also reconstructs outliers and assigns a spuriously high likelihood on OOD data [18] for reasons that are as-yet unclear.

Our main contributions can be summarized as follows:

- We propose NAE, a novel generative model constructed from an autoencoder;
- We propose OMI, a sampling strategy tailored for NAE;
- We empirically show that NAE is highly effective for outlier detection and can perform other generative tasks.

3.2 Background

3.2.1 Autoencoders

Autoencoders are neural networks trained to reconstruct an input datum $\mathbf{x} \in \mathcal{X} \subset \mathbb{R}^{D_{\mathbf{x}}}$. For an input $\mathbf{x}$, the quality of its reconstruction is measured in reconstruction error $l_\theta(\mathbf{x})$, where θ denotes parameters in an autoencoder. The loss function of an autoencoder L_{AE} for training is the expected reconstruction error of training data. Gradient descent training is performed via computing the gradient of L with respect to model parameters θ:

$$L_{\text{AE}} = \mathbb{E}_{\mathbf{x}\sim p(\mathbf{x})}[l_\theta(\mathbf{x})], \tag{3.1}$$

$$\nabla_\theta L_{\text{AE}} = \mathbb{E}_{\mathbf{x}\sim p(\mathbf{x})}[\nabla_\theta l_\theta(\mathbf{x})], \tag{3.2}$$

where ∇_θ is the gradient operator with respect to θ and $p(\mathbf{x})$ denotes the data density.

Architecture An autoencoder consists of two submodules, an encoder and a decoder. An encoder $f_e(\mathbf{x}) : \mathbb{R}^{D_\mathbf{x}} \to \mathbb{R}^{D_\mathbf{z}}$ maps an input $\mathbf{x}$ to a corresponding latent representation vector $\mathbf{z} \in \mathcal{Z} \subset \mathbb{R}^{D_\mathbf{z}}$, and a decoder $f_d(\mathbf{z}) : \mathbb{R}^{D_\mathbf{z}} \to \mathbb{R}^{D_\mathbf{x}}$ maps a latent vector $\mathbf{z}$ back to the input space. Then, the reconstruction error $l_\theta(\mathbf{x})$ is given as:

$$l_\theta(\mathbf{x}) = \mathrm{dist}(\mathbf{x}, f_d(f_e(\mathbf{x}))), \tag{3.3}$$

where $\mathrm{dist}(\cdot, \cdot)$ is a distance-like function measuring the deviation between an input $\mathbf{x}$ and a reconstruction $f_d(f_e(\mathbf{x}))$. A typical choice is the squared L^2 distance, i.e., $\mathrm{dist}(\mathbf{x}_1, \mathbf{x}_2) = ||\mathbf{x}_1 - \mathbf{x}_2||_2^2$. Other possible choices include L^1 distance, $\mathrm{dist}(\mathbf{x}_1, \mathbf{x}_2) = |\mathbf{x}_1 - \mathbf{x}_2|$, and the structural similarity (SSIM; [19, 20]).

Note that the reconstruction error (Eq. (3.3)) is *not* a likelihood of a datum, and therefore the minimization of the reconstruction error does not correspond to the maximization of the likelihood. Without modification, an autoencoder per se is not a probabilistic model.

Outlier Detection and Outlier Reconstruction A datum is an outlier or called OOD if it lies in the ρ-sublevel set of a data density $\{\mathbf{x}|p(\mathbf{x}) \leq \rho\}$ [21]. May ρ be set 0, an outlier is defined as an input from the outside of the support.

In the autoencoder-based outlier detection [4], an input is classified as OOD if its reconstruction error $l_\theta(\mathbf{x})$ is greater than a threshold τ: $l_\theta(\mathbf{x}) > \tau$. The outlier reconstruction indicates that there exists an input $\mathbf{x}^*$ with $p(\mathbf{x}^*) \leq \rho$, but $l_\theta(\mathbf{x}^*) < \tau$.

3.2.2 Energy-based Models

Unlike autoencoders, energy-based models (EBMs) are valid models for a normalized probability distribution. The EBM represents a probability distribution through the unnormalized negative log probability, also called the energy function $E_\theta(\mathbf{x})$. Here, θ denotes the model parameters.

For a continuous input $\mathbf{x} \in \mathcal{X} \subset \mathbb{R}^{D_\mathbf{x}}$, $E_\theta(\mathbf{x})$ defines the model density function $p_\theta(\mathbf{x})$ through Gibbs distribution:

$$p_\theta(\mathbf{x}) = \frac{1}{\Omega_\theta} \exp(-E_\theta(\mathbf{x})/T), \tag{3.4}$$

where $T \in \mathbb{R}^+$ is called the temperature and is often ignored by setting $T = 1$. Ω_θ is the normalization constant and is defined as:

$$\Omega_\theta = \int_{\mathcal{X}} \exp(-E_\theta(\mathbf{x})/T) \mathrm{d}\mathbf{x} < \infty. \tag{3.5}$$

The computation of Ω_θ is usually difficult for high-dimensional $\mathbf{x}$. However, maximum likelihood learning can still be performed without the explicit evaluation of Ω_θ. The gradient of negative log likelihood of data is given as follows [22]:

$$\begin{aligned}
&\mathbb{E}_{\mathbf{x}\sim p(\mathbf{x})}[-\nabla_\theta \log p_\theta(\mathbf{x})] \\
=&\mathbb{E}_{\mathbf{x}\sim p(\mathbf{x})}[\nabla_\theta E_\theta(\mathbf{x})]/T + \nabla_\theta \log \Omega_\theta && (3.6) \\
=&\mathbb{E}_{\mathbf{x}\sim p(\mathbf{x})}[\nabla_\theta E_\theta(\mathbf{x})]/T - \mathbb{E}_{\mathbf{x}'\sim p_\theta(\mathbf{x})}[\nabla_\theta E_\theta(\mathbf{x}')]/T && (3.7)
\end{aligned}$$

$\nabla_\theta \log \Omega_\theta$ in Eq. (3.6) is evaluated from the energy gradients of samples $\mathbf{x}'$ generated from the model in Eq. (3.7). The samples from $p_\theta(\mathbf{x})$ are often called "negative" samples.

In Eq. (3.7), the first term decreases the energy of the training data, or "positive" samples, while the second term increases the energy of the generated

samples, or "negative" samples. The training converges when $p_\theta(\mathbf{x})$ becomes identical to $p(\mathbf{x})$, as the two gradient terms cancel out. In practice, the two expectations in Eq. (3.7) are approximated with a mini-batch of samples during each iteration.

Langevin Monte Carlo (LMC) The negative samples are generated using MCMC. LMC ([23, 24]) is a simple yet effective MCMC method used in recent work on deep EBMs [25, 26, 27]. In LMC, a starting point $\mathbf{x}_0$ is drawn from a noise distribution $p_0(\mathbf{x})$, typically a Gaussian or uniform distribution. Starting from $\mathbf{x}_0$, a Markov chain evolves as follows:

$$\mathbf{x}_{t+1} = \mathbf{x}_t + \lambda_\mathbf{x} \nabla_\mathbf{x} \log p_\theta(\mathbf{x}_t) + \sigma_\mathbf{x} \epsilon_t, \tag{3.8}$$

where $\epsilon_t \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. $\lambda_\mathbf{x}$ and $\sigma_\mathbf{x}$ are the step size and the noise parameters, respectively. A theoretically motivated choice is $2\lambda_\mathbf{x} = \sigma_\mathbf{x}^2$, but the parameters are often tweaked separately for better performance [25, 26, 27]. As $\nabla_\mathbf{x} \log p_\theta(\mathbf{x}) = -\nabla_\mathbf{x} E(\mathbf{x})/T$, tweaking the step size can be seen as adjusting the temperature T.

To ensure the convergence of the chain, either Metropolis-Hastings rejection [28] or annealing of the noise parameter to zero [29] may be employed, but often omitted in practice.

We discuss specific strategies to evaluate the second term in Eq. (3.7) in Section 3.4. For a comprehensive review on various strategies for training an EBM, readers may refer to [30].

3.2.3 Outlier Reconstruction

The outlier reconstruction is a phenomenon that an autoencoder unexpectedly succeeds in reconstructing an input even though it is located outside of the training distribution. In this section, we provide illustrative examples that show

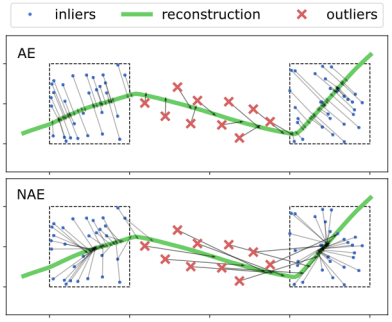

Figure 3.2 AE and NAE trained on a bi-modal distribution. Here, NAE is trained with its decoder fixed. The green lines denotes the decoder manifolds. The dotted lines link inputs and their reconstructions.

that outlier reconstruction is a consequence from the inductive biases of an autoencoder.

Multi-modal data When the training data distribution consists of multiple clusters, the outliers from the region between the clusters are likely to be reconstructed. Figure 3.2 depicts 2D synthetic data generated from a mixture of two disconnected uniform distributions and their reconstruction from autoencoders with one-dimensional latent space. The outliers (red crosses) from the middle of two clusters show reconstruction errors (the length of thin black lines) smaller than some inliers (blue dots). [8] noted this type of outlier reconstruction and mentioned that outliers "close to the mean" of data or "in the convex hull" of data are likely to be reconstructed.

This phenomenon arises from the inductive bias of an autoencoder that its encoder and decoder are smooth mappings. The extreme case of this inductive bias can be found in linear principal component analysis (PCA). PCA, a linear counterpart of an autoencoder [31], would reconstruct any outliers which reside on the principal axis. Note that this phenomenon is consistent with the objective

function of an autoencoder and PCA, as the objective does not penalize the reconstruction of outliers.

Compositionality When there is a compositional structure in data, we can still observe a reconstructed outlier even if it lies outside of the convex hull of training data. The data are compositional if each datum can be broken down into smaller reusable components; For example, MNIST can be considered highly compositional, since a digit image can be decomposed into smaller sub-patterns, such as straight lines and curves. An outlier can be successfully reconstructed when composed of a subset of components existing in the training data.

HalfMNIST and ChimeraMNIST datasets are constructed to demonstrate the effect of compositionality in outlier reconstruction. Although these images are not in the convex hull of MNIST digits, they share components found in MNIST. As shown in Figure 3.1, an autoencoder trained on MNIST have no problem reconstructing them and achieves poor AUC scores in classifying HalfMNIST and ChimeraMNIST from MNIST (See Table 3.3).

It seems that an autoencoder learns to reconstruct each part of an image separately but is not able to judge whether the combination of the parts is valid as a whole. This compositional way of processing facilitates generalization of a model [32], but the generalization of reconstruction in OOD inputs is not desirable for an autoencoder-based outlier detector.

Distributed representation We suspect the outlier reconstruction due to compositional processing may be attributed to the distributed representation [33] used in an autoencoder. To show the effect of the distributed representation, we train autoencoders on MNIST with the digit 9 excluded (MNISTnot9) and measure the reconstruction error of the digit 9 (MNIST9) under multiple values of latent dimensionality $D_{\mathbf{z}}$. Figure 3.3 shows the result. We observe the outlier

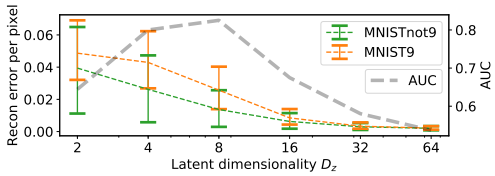

Figure 3.3 Detecting hold-out digit 9 from the rest of MNIST. Reconstruction errors and AUC scores are shown across multiple values of $D_{\mathbf{z}}$. The error bars denote 80-percentile around the means.

reconstruction of 9 possibly due to the compositional processing mentioned above. However, the outlier reconstruction occurs only when $D_{\mathbf{z}}$ is large. The latent representation is more distribution for large $D_{\mathbf{z}}$, as a larger number of hidden neurons are used to represent an input. This observation suggests that the distributed representation used in an autoencoder enables the compositional processing and thus facilitates outlier reconstruction.

3.3 Normalized Autoencoders

3.3.1 Definition

We propose **Normalized Autoencoder (NAE)**, a normalized probabilistic model defined from an autoencoder. The probability density of NAE $p_\theta(\mathbf{x})$ is defined as a Gibbs distribution (Eq. (3.4)) the energy of which is defined as the reconstruction error of an autoencoder:

$$E_\theta(\mathbf{x}) = l_\theta(\mathbf{x}). \tag{3.9}$$

Thus, the model density of NAE is given as

$$p_\theta(\mathbf{x}) = \frac{1}{\Omega_\theta} \exp(-l_\theta(\mathbf{x})/T), \tag{3.10}$$

where Ω_θ is defined as in Eq. (3.5). Due to the normalization constant, $p_\theta(\mathbf{x})$ is a properly normalized probability density.

As a probabilistic model, NAE is trained to maximize the likelihood of data. The loss function to be minimized is the negative log-likelihood of data:

$$\mathbb{E}_{\mathbf{x}\sim p(\mathbf{x})}[-\log p_\theta(\mathbf{x})] = \mathbb{E}_{\mathbf{x}\sim p(\mathbf{x})}[l_\theta(\mathbf{x})]/T + \log \Omega_\theta. \tag{3.11}$$

The gradient for the negative log-likelihood is evaluated as in conventional EBMs (Eq. (3.7)).

$$\begin{aligned} &\mathbb{E}_{\mathbf{x}\sim p(\mathbf{x})}[-\nabla_\theta \log p_\theta(\mathbf{x})] \\ &= \mathbb{E}_{\mathbf{x}\sim p(\mathbf{x})}[\nabla_\theta l_\theta(\mathbf{x})]/T - \mathbb{E}_{\mathbf{x}'\sim p_\theta(\mathbf{x})}[\nabla_\theta l(\mathbf{x}')]/T. \end{aligned} \tag{3.12}$$

Therefore, each gradient step decreases the reconstruction error of training data $\mathbf{x}$, while increasing the reconstruction error of negative samples $\mathbf{x}'$ generated from $p_\theta(\mathbf{x})$.

3.3.2 Remarks

Normalization as Regularization In NAE, enforcement of normalization can be viewed as a regularizer for the reconstruction loss (3.1). A typical formulation for a regularized autoencoder is given as $L = L_{\mathrm{AE}} + L_{\mathrm{reg}}$, where L_{reg} is a regularizer. By setting the loss function of NAE as $L = T\mathbb{E}_{\mathbf{x}\sim p(\mathbf{x})}[-\log p_\theta(\mathbf{x})]$, we have $L = L_{\mathrm{AE}} + T \log \Omega_\theta$. Therefore, the normalization constant contributes as a regularizer: $L_{\mathrm{reg}} = T \log \Omega_\theta$.

Suppression of Outlier Reconstruction During the training of NAE, the reconstruction of an outlier is inhibited by enforcing the normalization constraint. Given a successful sampling process, the negative samples should cover all high density regions of $p_\theta(\mathbf{x})$. A sample from a high density region of $p_\theta(\mathbf{x})$ has a low $l_\theta(\mathbf{x})$ by definition (Eq. (3.9)). Hence, if there exist a reconstructable

outlier, which has high $p_\theta(\mathbf{x})$ due to low $l_\theta(\mathbf{x})$, it will appear as a negative sample from MCMC. As the gradient update given in Eq. (3.12) increases the reconstruction error of negative samples, the reconstruction quality of a reconstructable outlier will be degraded. As a result, the reconstruction error of NAE becomes a more informative predictor that discriminates outliers from inliers than that of a conventional autoencoder.

Outlier Detection with Likelihood NAE bridges the two popular outlier detection criteria, namely, the reconstruction error [4] and the likelihood [14]. The reconstruction error criterion classifies an input with a large reconstruction error as OOD $l_\theta(\mathbf{x}) > \tau$, whereas the likelihood criterion predicts an input as an outlier if the log-likelihood is smaller than the threshold $\log p_\theta(\mathbf{x}) < \tau'$. These two criteria are equivalent in NAE for appropriately set τ and τ', as the reconstruction error and the log-likelihood has a linear relationship: $\log p_\theta(\mathbf{x}) = -l_\theta(\mathbf{x}) - \log \Omega_\theta$. Note that the two criteria rarely coincide in other models, for example, denoising autoencoders (DAE, [11]), VAE [17]), and DSEBMs [34], causing confusion on which of the decision rules should be employed for outlier detection.

Sample Generation Samples from $p_\theta(\mathbf{x})$ are generated through MCMC. Unlike VAE, the forward pass of a decoder should not be considered as sample generation.

3.4 On-Manifold Initialization

The main challenge in the training of NAE through Eq. (3.12) is that each iteration requires negative sample generation using MCMC, which is computationally expensive. In this section, we first discuss the failure modes of popular approximate sampling strategies for EBMs, namely Contrastive Divergence

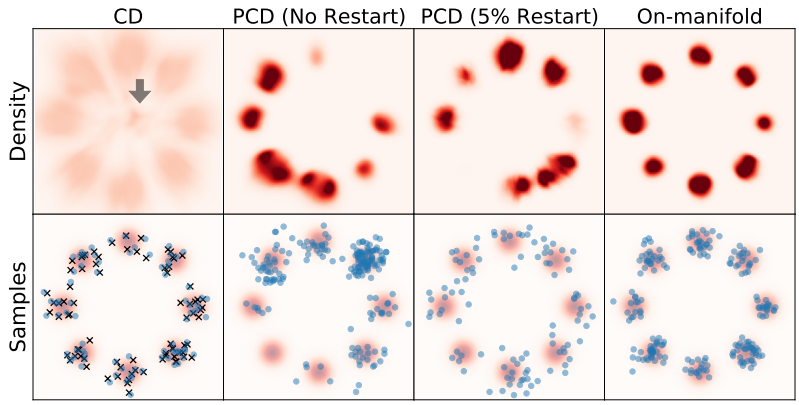

Figure 3.4 Density estimates and negative samples from NAEs trained by various approximate sampling methods. The generated samples (blue dots) are visualized along with the true density, a 2D mixture of 8 Gaussians. (CD) The learned density has a spurious mode, marked by an arrow. The black crosses denote training data. (PCD without restart) The highly correlated samples result in an oscillating density estimate. (PCD with restart) Despite the good quality of sampling, the density is poorly estimated. (On-manifold) Both density estimation and sample generation are performed well. More details are specified in Section 3.4.1 and Section 3.6.2.

(CD; [15]) and Persistent CD (PCD; [16]). We argue that the method on how the initial state of MCMC is chosen have incurred such failure modes. Then, we propose on-manifold initialization, an approximate sampling strategy effective in training the NAE. On-manifold initialization provides a better initial state for MCMC by leveraging the structure of an autoencoder.

There exist other training methods for EBMs which do not rely on MCMC, for example denoising score matching [35] or noise contrastive estimation [36], and they may also be applicable to NAE. We leave application of such methods on NAE as future work.

3.4.1 Failure Modes of CD and PCD

Failure Mode of CD CD, often called CD-k, draws a negative sample by first initializing a Markov chain of MCMC at a training data point, then proceeding k steps of MCMC transitions. The strength of CD is that the number of steps k can be radically smaller, e.g., $k = 1$, than the usual number of steps required in a convergent MCMC run, significantly reducing the amount of computation.

However, when k is small, CD-k is not able to suppress a spuriously high mode in the model density $p_\theta(\mathbf{x})$ located far from the data distribution $p(\mathbf{x})$, because negative samples are only generated in the vicinity of training data. Figure 3.4 shows an instance of a spurious mode in the model density. Negative samples (blue dots) are close to training data (black crosses) so that they do not reach for the density mode in the middle. As a result, the mode is not suppressed. Such a spurious mode will result in outlier detection failures and, in case of NAE, reconstructed outliers. The possibility of accidentally assigning high density in the unvisited area was acknowledged in the original article (Section 3 of [15]). Spurious modes are also observed in DAE, where a corrupted datum is located only in the neighborhood of a training data point [37]. Increasing k will decrease the chance of have spurious modes, but the computational advantage of CD will be lost when k is large.

Failure Mode of PCD An initial state of MCMC in PCD is given as the negative sample generated from MCMC in the previous training iteration. PCD was originally implemented using fully persistent MCMC [16]. However, without a restart, MCMC chains in a mini-batch may become highly correlated to each other. When $p_\theta(\mathbf{x})$ is multi-modal, the correlated chains yield degenerate negative samples which only cover a subset of density modes as in Figure 3.4. The degenerate samples make the density estimate oscillatory, slowing the

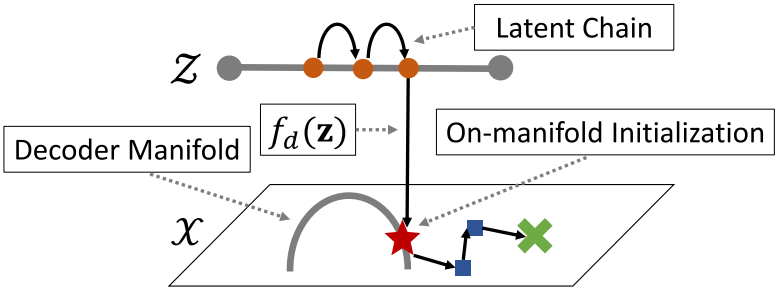

Figure 3.5 An illustration of the on-manifold initialization. The one-dimensional latent space $\mathcal{Z}$ and the two-dimensional input space $\mathcal{X}$ are shown. The red star is the on-manifold initialized state. The cross denotes a negative sample obtained at the end of the whole process.

convergence of the model.

The degeneracy between chains can be mitigated by randomly resetting the initial state to a sample from the noise distribution $p_0(\mathbf{x})$ with a small probability (typically 5%) [25, 26]. However, learning with PCD still fails to yield an accurate density estimate (Figure 3.4). This failure mode can be explained by the study of [27]: When a short MCMC chain initialized from $p_0(\mathbf{x})$ is used in training, an EBM simply learns a flow that maps $p_0(\mathbf{x})$ to $p(\mathbf{x})$, and the energy no longer models the data density. Using a restart drives an EBM to become such a flow, as restarted chains are short and start from $p_0(\mathbf{x})$.

In summary, CD initializes MCMC from the data distribution $p_\theta(\mathbf{x})$, and PCD initializes MCMC from a noise distribution $p_0(\mathbf{x})$. The convergence of MCMC is independent of its initialization in theory, but the initialization method can be crucial in practice, as shown in Figure 3.4. When $p_\theta(\mathbf{x})$, from which we want to sample, deviates significantly from $p_\theta(\mathbf{x})$ or $p_0(\mathbf{x})$, these initialization methods may lead to a poor density estimate and a suboptimal performance in outlier detection.

3.4.2 On-Manifold Initialization

We propose **on-manifold initialization (OMI)**, a novel MCMC initialization strategy which eventually leads to a significantly better density estimate. We aim to initialize a MCMC chain from a high-density region of $p_\theta(\mathbf{x})$ instead of $p_0(\mathbf{x})$ or $p(\mathbf{x})$. While finding a high-density region given an energy function is difficult in general, it is possible for NAE's distribution, since we can exploit the structure of an autoencoder. For a sufficiently well-trained autoencoder, a point with high $p_\theta(\mathbf{x})$, i.e., a small reconstruction error, will lie near the *decoder manifold*, which we define as:

$$\mathcal{M} = \{\mathbf{x} | \mathbf{x} = f_d(\mathbf{z}), \mathbf{z} \in \mathcal{Z}\}. \tag{3.13}$$

In on-manifold initialization, we initialize MCMC from a point in the decoder manifold $\mathbf{x}_0 \in \mathcal{M}$.

Not all points in $\mathcal{M}$ have high $p_\theta(\mathbf{x})$. To find points with high $p_\theta(\mathbf{x})$, we run a preliminary MCMC named as *latent chain* in the latent space $\mathcal{Z}$. The latent chain generates a sample from *on-manifold density* $q_\theta(\mathbf{z})$ defined from *on-manifold energy* $H_\theta(\mathbf{z})$.

$$q_\theta(\mathbf{z}) = \frac{1}{\Psi_\theta} \exp(-H_\theta(\mathbf{z})/T_\mathbf{z}), \tag{3.14}$$

$$H_\theta(\mathbf{z}) = E_\theta(f_d(\mathbf{z})), \tag{3.15}$$

where $\Psi_\theta = \int \exp(-H_\theta(\mathbf{z})/T_\mathbf{z}) d\mathbf{z}$ is the normalization constant and $T_\mathbf{z}$ is the temperature. A latent vector $\mathbf{x}$ with a small $H_\theta(\mathbf{z})$ will result in a small $E_\theta(\mathbf{x})$ when it is mapped to the input space by $\mathbf{x} = f_d(\mathbf{z})$. Thus, $H_\theta(\mathbf{z})$ guides the latent chain to find $\mathbf{z}$ which produce $\mathbf{x}_0 \in \mathcal{M}$ which has a small energy, i.e., a small reconstruction error.

Similarly to Eq. (3.8), we use LMC to run the latent chain. An initial state $\mathbf{z}_0$ is drawn from a noise distribution defined on the latent space. Then the state

propagates as:

$$\mathbf{z}_{t+1} = \mathbf{z}_t + \lambda_{\mathbf{z}} \nabla_{\mathbf{z}} \log q_\theta(\mathbf{z}_t) + \sigma_{\mathbf{z}} \epsilon_t, \tag{3.16}$$

where $\lambda_{\mathbf{z}}$ and $\sigma_{\mathbf{z}}$ are the step size and the noise parameters as in Eq. (3.8). A sample replay buffer [25] is applicable in the latent chain. Figure 3.5 illustrates negative sample generation process using the on-manifold initialization.

3.5 Related Work

Autoencoders There have been several attempts to formulate a probabilistic model from an autoencoder. VAE uses a latent variable model by introducing a prior distribution $p(\mathbf{z})$. However, the prior may deviate from the actual distribution of data in $\mathcal{Z}$, which may cause problems. GPND [38] models probability density by factorizing into on- and off-manifold components but still requires a prior distribution. $\mathcal{M}$-flow [39] only defines a probability density on the decoder manifold and does not assign a likelihood to off-manifold data. DAE models a density by learning the gradient of log-density [37].

MemAE [10] is a rare example that directly tackles the outlier reconstruction problem. MemAE employs a memory module that memorizes training data to prevent outlier reconstruction, but in this case, the reconstruction error for an inlier can be large because the model's generalization ability is also limited.

Design of Energy Functions Specifying the class of $E_\theta(\mathbf{x})$ not only has computational consequences but alters the inductive bias that an EBM encodes. Feed-forward convolutional networks are used in [25] and [26] and are shown to effectively model the distribution of images. The energy can also be modeled in an auto-regressive manner [40, 41]. Auto-regressive energy functions are very flexible and thus are capable of modeling high-frequency patterns in data.

Table 3.1 MNIST hold-out class detection AUC scores. The intervals denote the standard error of mean after 10 training runs.

Hold-out:	0	1	2	3	4	5	6	7	8	9	avg
NAE-OMI	**.989** ±.002	**.919** ±.013	**.992** ± .001	**.949** ±.004	**.949** ±.005	**.978** ± .003	**.938** ±.004	**.975** ±.024	**.929** ±.004	**.934** ±.005	**.955**
NAE-CD	.799	.098	.878	.769	.656	.806	.874	.537	.876	.500	.679
NAE-PCD	.745	.114	.879	.754	.690	.813	.872	.509	.902	.544	.682
AE	.819	.131	.843	.734	.661	.755	.844	.542	.902	.537	.677
DAE	.769	.124	.872	.935	.884	.793	.865	.533	.910	.625	.731
VAE(R)	.954	.391	.978	.910	.860	.939	.916	.774	.946	.721	.839
VAE(L)	.967	.326	.976	.906	.798	.927	.928	.751	.935	.614	.813
WAE	.817	.145	.975	**.950**	.751	.942	.853	.912	.907	.799	.805
GLOW	.803	.014	.624	.625	.364	.561	.583	.326	.721	.426	.505
PXCNN++	.757	.030	.663	.663	.483	.642	.596	.307	.810	.497	.545
IGEBM	.926	.401	.642	.644	.664	.752	.851	.572	.747	.522	.672
DAGMM	.386	.304	.407	.435	.444	.429	.446	.349	.609	.420	.423

The reconstruction error of an autoencoder is used as a discriminator in EBGAN [34]. Although the reconstruction error was called "energy" in EBGAN, the formulation is clearly different from NAE. EBGAN does not utilize Gibbs distribution formulation (Eq. (3.4)) to model a distribution, and samples are generated from a separate generator network. In DSEBM [34], the difference between an input and its reconstruction is interpreted as the gradient of log-density.

3.6 Experiments

3.6.1 Technicalities for NAE Training

Pre-training as a Conventional Autoencoder NAE can be pre-trained as a conventional autoencoder by minimizing the reconstruction error following Eq. (3.2), before the main training. By providing a good initialization for network weights and the decoder manifold, pre-training greatly reduces the number of NAE training iterations (Eq. (3.12)) required until convergence. Pre-training is not always necessary: In our experiments, we observe that NAE can be trained successfully without pre-training for synthetic data. However, pre-training was essential to obtain decent results for larger scale data, such as MNIST and CIFAR-10.

Latent Space Structure The configuration of the latent space is important in the stable learning using on-manifold initialization. We found the two configurations that work: the unbounded real space $\mathbb{R}^{D_{\mathbf{z}}}$ and the surface of a hypersphere $\mathbb{S}^{D_{\mathbf{z}}-1}$. When $\mathcal{Z} = \mathbb{R}^{D_{\mathbf{z}}}$, a linear layer is used as the output of an encoder. $q_0(\mathbf{z})$ is set as $\mathcal{N}(\mathbf{0}, \mathbf{I})$. The squared norm of the latent vectors are added to the loss function as a regularizer so that $\mathbf{z}$'s concentrate near the origin [42]. For $\mathcal{Z} = \mathbb{S}^{D_{\mathbf{z}}-1}$ [43, 44, 45], the output of an encoder is projected to the surface of a unit ball through the division by its norm: $\mathbf{z} \leftarrow \mathbf{z}/||\mathbf{z}||$. In Langevin dynamics, a sample is projected to $\mathbb{S}^{D_{\mathbf{z}}-1}$ at the end of each step. $q_0(\mathbf{z})$ is set to a uniform distribution on $\mathbb{S}^{D_{\mathbf{z}}-1}$.

Regularizing Negative Sample Energy As introduced in [25], we regularize the energy of negative samples to prevent its divergence. We add the average squared energy of negative samples in a mini-batch to the loss function: $L = L_{\text{NAE}} + \alpha \sum_{i=1}^{B} E(\mathbf{x}_i')^2/B$ for the batch size B and the hyperparameter α. We set $\alpha = 1$.

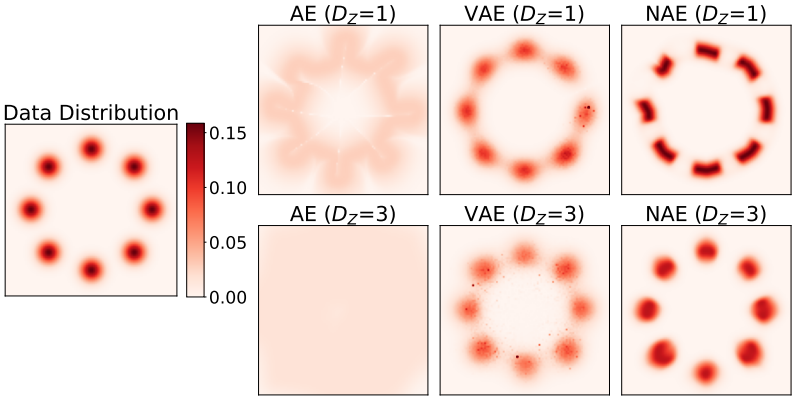

Figure 3.6 Estimating 8 Gaussians using various autoencoders. The density of an autoencoder (AE) is computed from Eq. (3.10). AE gives a significant amount of probability to low-data-density area. VAE also assigns some probability mass in between Gaussians. Meanwhile, the density estimate from NAE agrees well with the data distribution.

3.6.2 2D Density Estimation

We demonstrate the density estimation capability of NAE with a two-dimensional mixture of 8 Gaussians. First, we benchmark negative sample generation strategies for NAE, including CD, PCD with and without restart, and on-manifold initialization. The results are shown in Figure 3.4 and discussed in Section 3.4.1 in detail.

Second, we compare NAE trained with the on-manifold initialization to a conventional autoencoder and VAE (Figure 3.6). An autoencoder assigns high densities on regions between Gaussian modes, meaning that an autoencoder gives a small reconstruction error from a points from the region. For the overcomplete case ($D_{\mathbf{z}} = 3 > D_{\mathbf{x}}$), an autoencoder almost becomes the identity map, and its reconstruction error is not an informative predictor for an outlier.

VAE and NAE learn a non-identity function under the overcomplete setting, showing the effectiveness of their regularizers.

In the experiments, the identical network architecture is used, and the temperature is optimized by gradient descent. In on-manifold initialization, temperature values are shared by the main MCMC and the latent chain. When performing MCMC in $\mathcal{X}$, Metropolis-Hastings rejection is applied to ensure the detailed balance but is not applied in the latent chain. For visualization, the normalization constants for an autoencoder and NAE are computed by numerically integrating over the domain, $[-4, 4]^2$.

3.6.3 Outlier Detection

Experimental Setting We empirically demonstrate the effectiveness of NAE as an outlier detector. In outlier detection tasks, an outlier detector is trained only using inlier data and then asked to discriminate outliers from inliers during test phase. Given an input, a detector is assumed to produce a scalar decision function which indicates the outlierness of the input. We measure the detection performance in AUC, i.e., the area under the receiver operating characteristic curve. Following the protocol of [46] and [47], we use an OOD dataset different from the datasets used in test phase to tune model hyperpamraeters. Additional details on model implementation and datasets can be found in the supplementary material.

The identical networks architectures are used for all autoencoder-based methods. The reconstruction error is used as the decision function, except for VAE. For deep generative models, PixelCNN++ (PXCNN++, [48]), Glow [49] and a feed-forward EBM (IGEBM, [25]), we use the negative log-likelihood (i.e., the energy) as the decision function. For VAE, we show two results from using the reconstruction error (R) or the negative log-likelihood (L) as decision

functions.

MNIST Hold-Out Class Detection One class from MNIST is set as the outlier class and the rest as the inlier class. Then, the procedure is repeated for all ten classes in MNIST. ConstantGray dataset is used for model selection.

This problem is not as easy as it seems, as confirmed in the very low performance of various algorithms in Table 3.1. When a class is held out from MNIST, the remaining 9 classes may contain a set of visual features sufficient to reconstruct the hold-out class, i.e., the outlier reconstruction occurs. The outlier reconstruction is particularly severe for the digit 1, 4, 7 and 9, possibly because their shape can be reconstructed from the recombination of other digits. For example, overlapping 4 and 7 produces a shape similar to 9. Interestingly, most of the other baseline algorithms also show poor performance when 1, 4, 7 or 9 are held out as the outlier. NAE shows the highest AUC score for all classes and effectively suppresses the reconstruction of the outlier class (Figure 3.7).

We also compare CD and PCD along with OMI in training NAEs. Using CD and PCD show poor outlier detection performance, although given the identical set of MCMC parameters.

Out-of-Distribution Detection The samples from different datasets are used as the outlier class. We test two inlier datasets, CIFAR-10 or ImageNet 32×32 (ImageNet32). Zero-padded 32×32 MNIST images are used for model selection. Results are shown in Table 3.2.

It is known that constant images and SVHN images are particularly difficult outliers for generative models trained on a set of images with rich visual features [18, 50]. However, NAE detect such difficult outliers successfully. All models are able to discriminate noise outliers, indicating that their poor performance is not from the failure of training.

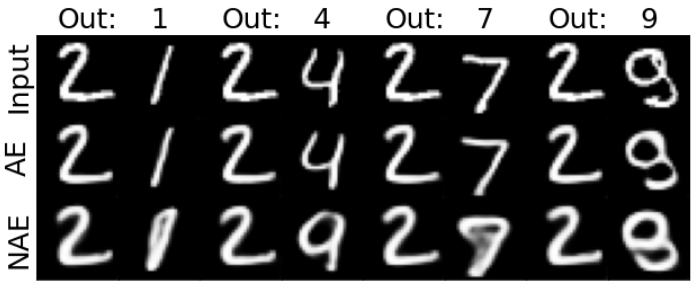

Figure 3.7 MNIST hold-out class detection examples from four different hold-out settings (1, 4, 7 and 9). The bottom two rows depict the reconstructions from four autoencoders (AE) and four NAEs trained on each setting. AEs reconstruct the outlier class well, while NAEs selectively reconstruct only inliers.

3.6.4 Sample Generation

Samples are generated from NAE using MCMC with OMI. Figure 3.9 shows the samples from NAEs trained on MNIST and on CelebA 64×64. The random initial states of the latent chain ($\mathbf{z}_0$) map to unrecognizable images. After the latent chain, OMI produces somewhat realistic images. MCMC on $\mathcal{X}$ refines the OMI images. Although quantitative image quality metric for samples generated from NAE is not on a par with that of generative models which specialize in sampling, but the generated samples are indeed visually sensible.

We generate 50,000 samples from NAE trained on CelebA 64×64 and compute FID score [51]. We also visualize some of images generated by NAE in Figure 3.8. While FID score of NAE was not as low as ones from models specialized in generation, such as NCSN [52], FID score of NAE resides in a ballpark of what is achievable by autoencoder-based methods. We believe that tuning network architecture and sampling procedure will result in enhanced samples.

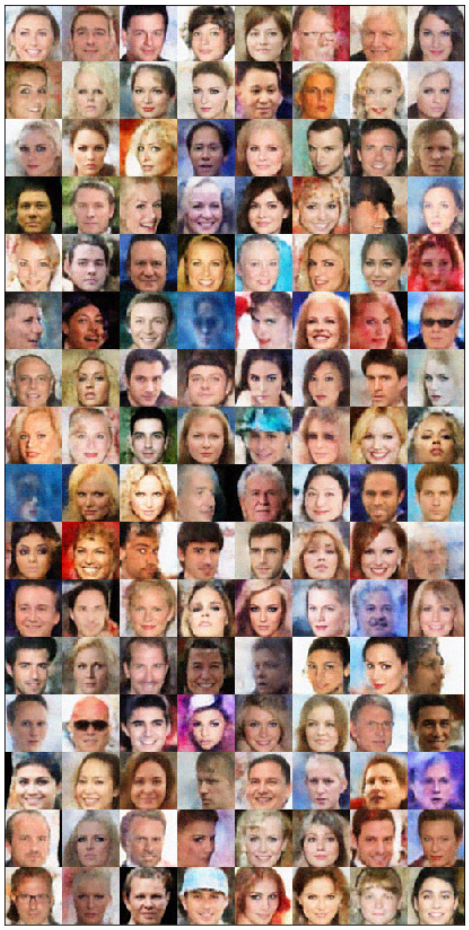

Figure 3.8 More samples from NAE trained on CelebA 64×64. While most of the samples are visually sensible, a few generation failures can be spotted. Improving the sample generation process will be able to eliminate such non-realistic images.

Table 3.2 OOD detection performance in AUC.

In: CIFAR-10	Constant	Gray	FMNIST	SVHN	CelebA	Noise
NAE	**.923**	**.819**	**.818**	**.789**	1.0
AE	.006	.650	.175	.655	1.0
DAE	.001	.671	.175	.669	1.0
VAE(R)	.002	.700	.191	.662	1.0
VAE(L)	.002	.767	.185	.684	1.0
WAE	.000	.649	.168	.652	1.0
GLOW	.384	.222	.260	.419	1.0
PXCNN++	.000	.013	.074	.639	1.0
IGEBM	.192	.216	.371	.477	1.0
In: ImageNet32	**Constant**	**Gray**	**FMNIST**	**SVHN**	**CelebA**	**Noise**
NAE	**.966**	**.994**	**.985**	**.949**	1.0
AE	.005	.915	.102	.325	1.0
DAE	.069	.991	.102	.426	1.0
VAE(R)	.030	.936	.132	.501	1.0
VAE(L)	.028	.950	.132	.545	1.0
WAE	.069	.991	.081	.364	1.0
GLOW	.413	.856	.169	.479	1.0
PXCNN++	.000	.004	.027	.238	1.0

3.7 Discussion

3.7.1 Comparison to Other EBMs

NAE uses Gibbs distribution to define a density function as in other EBMs (Eq. 3.4). The main difference between NAE and other EBMs is the choice of an energy function. However, this difference results in significant theoretical and practical consequences. First, we naturally incorporate the manifold hypothesis, i.e. the assumption that high-dimensional data lie on a low-dimensional manifold, into a model. Second, the energy function of NAE can be pre-trained as a

Table 3.3 FID score of 50,000 images generated from a model trained on CelebA 64×64. A low FID score indicates that the generated images have similar statistics to the real images in Inception network's feature space.

Model	FID
NAE	94.00
From [42]	
AE	127.85
AE-L2	346.29
VAE	48.12
RAE-GP	116.30
RAE-L2	51.13
RAE-SN	44.74
From [52]	
NCSN (w/ denoising)	26.89
NCSNv2 (w/ denoising)	10.23

conventional autoencoder. Third, more effective sampling can be performed by using OMI, leading to a more accurate density estimate.

3.7.2 Likelihood-based Outlier Detection

[18] reported that deep generative models, such as VAEs, autoregressive models, and normalizing flows, fail at outlier detection by producing spuriously high likelihoods for OOD data, which are also observed in our experiments. This observation raised skepticism on the likelihood-based outlier detection, leading to the proposal of alternative metrics to the likelihood, e.g., [46]. However, we speculate that the failure of outlier detection should be attributed to the specifics of the models and not to the use of the likelihood as a metric. EBMs

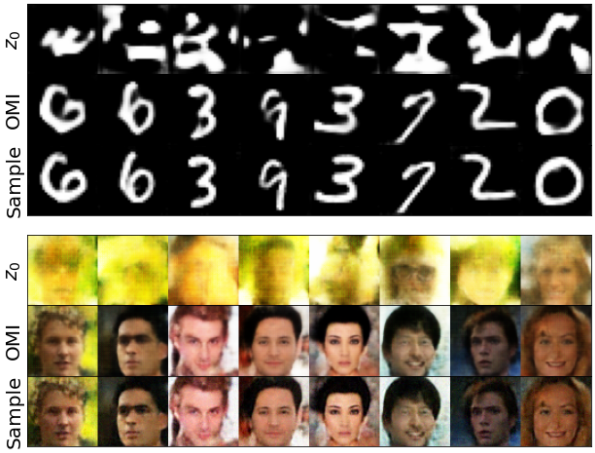

Figure 3.9 Sampling with NAEs trained on MNIST and CelebA 64×64. ($\mathbf{z}_0$) The random initialization of the latent chain. We visualize $f_d(\mathbf{z}_0)$. (OMI) Images after OMI. (Samples) Samples obtained after MCMC starting from OMI. OMI images and Samples corresponds to the red start and the green cross in Figure 3.5, respectively.

have been shown to be effective in outlier detection [25, 26], even though the model uses the likelihood as a decision function. The experimental results from NAE further confirms the effectiveness of the likelihood in outlier detection.

3.7.3 Analytic Solution for Linear Case

Linear NAEs reduce to Gaussian distributions. Consider $f_e(\mathbf{x}) = W\mathbf{x}$ and $f_d(\mathbf{z}) = W^\top \mathbf{z}$ with $W \in \mathbb{R}^{D_\mathbf{z} \times D_\mathbf{x}}$. Given the squared L^2 distance reconstruction error, the density of NAE is written as:

$$p_\theta(\mathbf{x}) = \exp(-\mathbf{x}^\top \Sigma^{-1} \mathbf{x}/2)/\Omega_\theta, \tag{3.17}$$

where $\Sigma^{-1} = 2(I - W^\top W)^2/T$. When the determinant of $I - W^\top W$ is non-zero, $p_\theta(\mathbf{x})$ becomes a well-defined Gaussian. Under certain conditions, the maximum

likelihood estimate of Σ becomes the empirical covariance of data, as in a usual Gaussian distribution.

It is interesting to note that a linear VAE also reduces into a Gaussian, as it is equivalent to probabilistic PCA[17]. On the other hand, a linear autoencoder is equivalent to PCA [31], which is not a generative model.

We provide a more detailed derivation for linear NAE. When a linear overcomplete autoencoder is used, NAE reduces into a Gaussian distribution. Consider a linear deterministic encoder $f_e(\mathbf{x}) = W\mathbf{x}$ and a decoder $f_d(\mathbf{z}) = W^\top \mathbf{z}$, where $W \in \mathbb{R}^{D_\mathbf{z} \times D_\mathbf{x}}$. For the squared L^2 distance reconstruction error,

$$l(\mathbf{x}) = ||\mathbf{x} - W^\top W\mathbf{x}||^2 \tag{3.18}$$

$$= \mathbf{x}^\top (I - W^\top W)^2 \mathbf{x}, \tag{3.19}$$

and therefore the density of NAE can be written as:

$$p_\theta(\mathbf{x}) = \frac{1}{\Omega_\theta} \exp(-\mathbf{x}^\top \Sigma^{-1} \mathbf{x}/2), \tag{3.20}$$

where $\Sigma^{-1} = 2(I - W^\top W)^2/T$.

Eq. (3.20) is a Gaussian distribution with zero mean and Σ covariance. For this Gaussian to be well-defined, the normalization constant should be finite $\Omega_\theta < \infty$. To the covariance positive definite, we need that the determinant of $(I - W^\top W)$ is non-zero, i.e., no eigenvalue of $W^\top W$ should be one. This means that the autoencoder should not be the identity along any of orthogonal bases.

As an interesting special case, consider $W = 0$. This zero-autoencoder is uninformative, since all inputs are mapped to the origin, but it still defines a valid probability distribution. In fact, $p_\theta(\mathbf{x})$ becomes a standard normal distribution.

Now, we consider the overcomplete setting, where $D_\mathbf{z} \geq D_\mathbf{x}$, and look for the maximum likelihood parameter estimate. When $D_\mathbf{z} \geq D_\mathbf{x}$, the matrix $(I - W^\top W)^2$ spans all positive semidefinite matrices. Given a zero-centered dataset

$\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^N$, Σ that maximizes the the likelihood of data is the empirical covariance $\Sigma_{\mathrm{ML}} = \sum_{i=1}^N \mathbf{x}_i \mathbf{x}_i^\top / N$. Therefore, NAE trained to maximum the likelihood of data is identical to a Gaussian distribution fitted via maximum likelihood.

Recall that a conventional linear autoencoder becomes the identity when $D_{\mathbf{z}} \geq D_{\mathbf{x}}$.

3.8 Conclusion

We have introduced a novel interpretation of the reconstruction error as an energy function. Our interpretation leads to a novel class of probabilistic autoencoders, which shows impressive OOD detection performance and bridges EBMs and autoencoders.

Chapter 4

Manifold Projection-Diffusion Recovery: Leveraging Manifold Structure in Energy-Based Model Training

4.1 Introduction

The unsupervised detection of anomalous data is a task appearing frequently in practical applications, such as industrial surface inspection [53], machine fault detection [54], and particle physics [55]. Modeling the probability distribution of normal data $p_{data}(\mathbf{x})$ is a principled approach for anomaly detection [14]. Anomalies, often called outliers, lie outside of the data distribution and thus can be characterized by low probability under the distribution. However, many deep generative models capable of evaluating the likelihood of data, including variational autoencoders (VAE), autoregressive models, and flow-based models are known to perform poorly on well-known anomaly detection benchmarks such as CIFAR-10 (in) vs SVHN (out), by assigning high likelihood on seemingly trivial outliers [47, 18].

On the other hand, deep energy-based models (EBMs) have demonstrated notable improvement in anomaly detection compared to other deep generative models [25]. While the specific reason for the superior performance of EBMs has not been formally analyzed, one probable factor is the explicit mechanism

employed in EBM's maximum likelihood training that reduces the likelihood of negative samples. These negative samples are generated from the model distribution $p_\theta(\mathbf{x})$ using Markov Chain Monte Carlo (MCMC). Since modern EBMs operate in high-dimensional data spaces, covering the entire space with a Markov chain of finite length is an extremely difficult task. Generating negative samples has been a significant obstacle in EBM training, leading to the development of several heuristics such as using truncated chains [27], persistent chains [16], sample replay buffers [25], and data augmentation in the middle of the chain [56].

Instead of requiring a Markov chain to cover the entire space, EBM can be trained with MCMC running within the vicinity of each training datum. Contrastive Divergence (CD; [15]) uses a short Markov chain initialized on training data to generates negative samples close to the training distribution. Diffusion Recovery Likelihood (DRL; [1]) lets MCMC focus on the neighborhood of training data by employing the recovery likelihood as the objective function for EBM training. Recovery likelihood $p_\theta(\mathbf{x}|\tilde{\mathbf{x}})$ is the conditional probability of training datum $\mathbf{x}$ given the observation of its copy $\tilde{\mathbf{x}}$ perturbed with a Gaussian noise. Training of DRL requires sampling from the recovery likelihood distribution $p_\theta(\mathbf{x}|\tilde{\mathbf{x}})$, which is easier than drawing samples from the model distribution $p_\theta(\mathbf{x})$, as $p_\theta(\mathbf{x}|\tilde{\mathbf{x}})$ is close to uni-modal and concentrated near $\mathbf{x}$. While CD and DRL significantly stabilize the negative sampling process, the resulting EBMs exhibit limited anomaly detection performance due to the insufficient coverage of negative samples in the input space.

In this paper, we present a novel training algorithm for EBM that does not require MCMC covering the entire space while achieving accurate density estimation needed for anomaly detection. The proposed algorithm, **Manifold Projection-Diffusion Recovery** (MPDR), extends the recovery likelihood

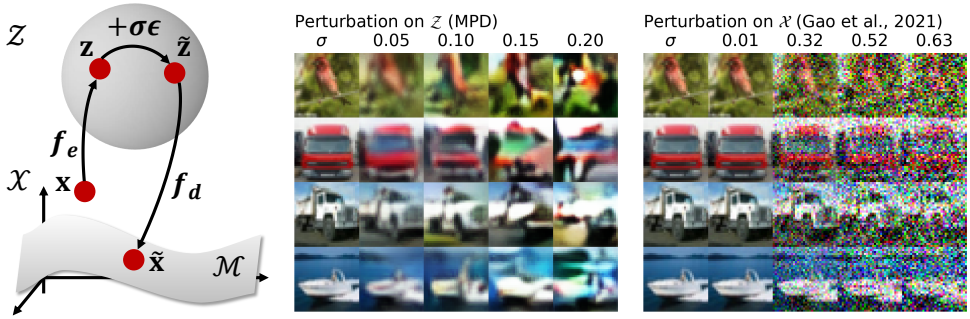

Figure 4.1 (Left) An illustration of Manifold Projection-Diffusion. A datum $\mathbf{x}$ is first projected into the latent space $\mathcal{Z}$ through the encoder $f_e(\mathbf{x})$ and then diffused with a local noise, such as Gaussian. The perturbed sample $\tilde{\mathbf{x}}$ is obtained by projecting it back to the input space $\mathcal{X}$ through the decoder $f_d(\mathbf{z})$. (Right) Comparison between the perturbations in MPDR and DRL [1] on CIFAR-10 examples.

framework by replacing Gaussian noise with **Manifold Projection-Diffusion** (MPD), a novel perturbation operation that reflects low-dimensional structure of data. In MPD, a training datum is projected onto a smooth low-dimensional manifold that approximately spans the training data and then diffused along the manifold, where we employ an autoencoder to obtain the manifold. Compared to Gaussian noise, MPD captures relevant modes of variation in data, such as change in in colors or shapes in an image, as shown in Fig. 4.1. A MPD-perturbed sample serves as an informative starting point for MCMC that generates a negative sample, teaching EBM to discriminate challenging outliers that has partial similarity to training data.

With MPD-perturbed datum $\tilde{\mathbf{x}}$, EBM is trained by maximizing the recovery likelihood $p(\mathbf{x}|\tilde{\mathbf{x}})$, where the maximization requires negative samples drawn from $p(\mathbf{x}|\tilde{\mathbf{x}})$. We derive a simple expression for evaluating $p(\mathbf{x}|\tilde{\mathbf{x}})$ in MPDR and

propose an efficient two-stage MCMC strategy for sampling from $p(\mathbf{x}|\tilde{\mathbf{x}})$. Our sampling strategy first uses the latent space of the autoencoder for faster exploration and then perform MCMC in the input space for exploring off-manifold region.

MPDR presents a novel way of employing encoder-decoder modules in EBM training. Compared to existing autoencoder-based EBM training methods such as VAEBM [57], Flow Contrastive Estimation [58], and Divergence Triangle [59, 60], MPDR not only show stronger performance in anomaly detection tasks but also provides some practical advantages. First, MPDR performs well with lightweight autoencoders that has significantly fewer parameters than NVAE [61] or Glow [49]. Second, the formulation of MPDR naturally supports the use of multiple autoencoders during training. Ensembling over multiple autoencoder manifolds improves the performance and also reduces the risk of not choosing the best autoencoder.

Our contributions can be summarized as follows:

- We propose Manifold Projection-Diffusion Recovery, a novel objective function for training EBMs which gives consistent estimation of density. MPDR is the recovery likelihood with a manifold-informed perturbation.
- We provide a suite of practical strategies for achieving successful anomaly detection with MPDR, including two-stage sampling, energy function design, and perturbation ensemble.
- We demonstrate the effectiveness of MPDR on unsupervised anomaly detection tasks with various data types, including images, high-dimensional representations, and acoustic signals collected from machines.

4.2 Preliminaries

Energy-Based Models (EBM) An energy-based generative model, or an unnormalized probabilistic model, represents a probability density function using a scalar energy function $E_\theta : \mathcal{X} \to \mathbb{R}$, where $\mathcal{X}$ denotes the domain of data. The energy function E_θ defines a probability density function p_θ through the following relationship:

$$p_\theta(\mathbf{x}) \propto \exp(-E_\theta(\mathbf{x})). \tag{4.1}$$

The parameters θ can be learned by maximum likelihood estimation given iid samples from the underlying data distribution $p_{data}(\mathbf{x})$. The gradient of the log-likelihood for a training sample $\mathbf{x}$ is well-known [15] and can be written as follows:

$$\nabla_\theta \log p_\theta(\mathbf{x}) = \nabla_\theta E_\theta(\mathbf{x}) - \mathbb{E}_{\mathbf{x}^- \sim p_\theta(\mathbf{x})}[\nabla_\theta E_\theta(\mathbf{x}^-)], \tag{4.2}$$

where $\mathbf{x}^-$ denotes a "negative" sample drawn from the model distribution $p_\theta(\mathbf{x})$. Typically, $\mathbf{x}^-$ is generated using Langevin Monte Carlo (LMC). In LMC, points are initialized arbitrarily and then iteratively updated in a stochastic manner to simulate independent sampling from $p_\theta(\mathbf{x})$. For each time step t, a point $\mathbf{x}^{(t)}$ is updated by $\mathbf{x}^{(t+1)} = \mathbf{x}^{(t)} - \lambda_1 \nabla_{\mathbf{x}} E_\theta(\mathbf{x}^{(t)}) + \lambda_2 \epsilon^{(t)}$, for $\epsilon^{(t)} \sim \mathcal{N}(0, \mathbf{I})$. The step size λ_1 and the noise scale λ_2 are often tuned separately in practice. Since LMC needs to be performed in every iteration of training, it is infeasible to run the sampling until convergence, and a compromise must be made. Popular heuristics include initializing MCMC on training data [15], using short-run LMC [27], and utilizing replay buffer [16, 25].

Recovery Likelihood [62, 1] Instead of directly maximizing the likelihood (Eq. (4.2)), θ can be learned through the process of denoising data from artificially injected Gaussian noises. Denoising corresponds to maximizing *recovery*

likelihood $p(\mathbf{x}|\tilde{\mathbf{x}})$, the probability of recovering data $\mathbf{x}$ from its noise-corrupted version $\tilde{\mathbf{x}} = \mathbf{x} + \sigma\epsilon$, where $\epsilon \sim \mathcal{N}(0, \mathbf{I})$.

$$p_\theta(\mathbf{x}|\tilde{\mathbf{x}}) \propto p_\theta(\mathbf{x})p(\tilde{\mathbf{x}}|\mathbf{x}) \propto \exp\left(-E_\theta(\mathbf{x}) - \frac{1}{2\sigma^2}||\mathbf{x} - \tilde{\mathbf{x}}||^2\right), \tag{4.3}$$

where Bayes' rule is applied. The model parameter θ is estimated by maximizing the log recovery likelihood, i.e., $\max_\theta \mathbb{E}_{\mathbf{x},\tilde{\mathbf{x}}}[\log p_\theta(\mathbf{x}|\tilde{\mathbf{x}})]$, for $\mathbf{x} \sim p_{data}(\mathbf{x}), \tilde{\mathbf{x}} \sim p(\tilde{\mathbf{x}}|\mathbf{x})$, where $p(\tilde{\mathbf{x}}|\mathbf{x}) = \mathcal{N}(\mathbf{x}, \sigma^2\mathbf{I})$. This estimation is shown to be consistent under the same conditions where maximum likelihood estimation is consistent (Appendix A.2 in [1]). DRL [1] uses a slightly modified perturbation $\tilde{\mathbf{x}} = \sqrt{1-\sigma^2}\mathbf{x} + \sigma\epsilon$ in training EBM, following [63]. This change introduces minor modification of Eq. (4.3).

The recovery likelihood $p_\theta(\mathbf{x}|\tilde{\mathbf{x}})$ (Eq. 4.3) can be viewed as another EBM with the energy $\tilde{E}_\theta(\mathbf{x}|\tilde{\mathbf{x}}) = E_\theta(\mathbf{x}) + ||\mathbf{x} - \tilde{\mathbf{x}}||^2/2\sigma^2$. Therefore, the gradient $\nabla_\theta \log p_\theta(\mathbf{x}|\tilde{\mathbf{x}})$ has the same form with the log-likelihood gradient of EBM (Eq. (4.2)), except that negative samples are drawn from $p_\theta(\mathbf{x}|\tilde{\mathbf{x}})$ instead of $p_\theta(\mathbf{x})$:

$$\nabla_\theta \log p_\theta(\mathbf{x}|\tilde{\mathbf{x}}) = \nabla_\theta E_\theta(\mathbf{x}) - \mathbb{E}_{\mathbf{x}^- \sim p_\theta(\mathbf{x}|\tilde{\mathbf{x}})}[\nabla_\theta E_\theta(\mathbf{x}^-)], \tag{4.4}$$

from $\nabla_\theta \tilde{E}_\theta(\mathbf{x}|\tilde{\mathbf{x}}) = \nabla_\theta E_\theta(\mathbf{x})$ as $p(\mathbf{x}|\tilde{\mathbf{x}})$ is independent of θ. Unlike $p_\theta(\mathbf{x})$ which may be highly multi-modal, $p_\theta(\mathbf{x}|\tilde{\mathbf{x}})$ is more likely to be concentrated near $\mathbf{x}$ and thus sampling using MCMC is more stable [1]. However, it is questionable whether Gaussian noise is the most informative way to perturb the data in a high-dimensional space.

4.3 Manifold Projection-Diffusion Recovery

We introduce the Manifold Projection-Diffusion Recovery (MPDR) algorithm, which trains EBM by recovering from perturbations that are more informative

than Gaussian noise. We first propose Manifold Projection-Diffusion (MPD), a novel perturbation operation leveraging the low-dimensional structure inherent in the data. Then, we derive the recovery likelihood for MPD. We also provide an efficient sampling strategy and practical implementation techniques for MPDR.

Autoencoders MPDR assumes that a pretrained autoencoder approximating the training data manifold is given. The autoencoder consists of an encoder $f_e : \mathcal{X} \to \mathcal{Z}$ and a decoder $f_d : \mathcal{Z} \to \mathcal{X}$, both are assumed to be deterministic and differentiable. The latent space is denoted as $\mathcal{Z}$. The dimensionalities of $\mathcal{X}$ and $\mathcal{Z}$ are denoted as $D_{\mathbf{x}}$ and $D_{\mathbf{z}}$, respectively. We assume f_e and f_d as typical deep neural networks jointly trained to reduce the training data's reconstruction error $l(\mathbf{x})$, where $l(\mathbf{x})$ is typically an l_2 error, $l(\mathbf{x}) = ||\mathbf{x} - f_d(f_e(\mathbf{x}))||^2$.

4.3.1 Manifold Projection-Diffusion

We propose a novel perturbation operation, **Manifold Projection-Diffusion** (MPD). Instead of adding Gaussian noise directly to a datum $\mathbf{x} \xrightarrow{+\sigma\epsilon} \tilde{\mathbf{x}}$ as in the conventional recovery likelihood, MPD first encodes $\mathbf{x}$ using the autoencoder and then applies a noise in the latent space:

$$\mathbf{x} \xrightarrow{f_e} \mathbf{z} \xrightarrow{+\sigma\epsilon} \tilde{\mathbf{z}} \xrightarrow{f_d} \tilde{\mathbf{x}}, \tag{4.5}$$

where $\mathbf{z} = f_e(\mathbf{x})$, $\tilde{\mathbf{z}} = \mathbf{z} + \sigma\epsilon$, and $\tilde{\mathbf{x}} = f_d(\tilde{\mathbf{z}})$. The noise magnitude σ is a predefined constant and $\epsilon \sim \mathcal{N}(0, \mathbf{I})$. The first step **projects** $\mathbf{x}$ into $\mathcal{Z}$, and the second step **diffuses** the encoded vector $\mathbf{z}$. When decoded through f_d, the resulting perturbation $\tilde{\mathbf{x}}$ always lies on the **decoder manifold** $\mathcal{M} = \{\mathbf{x} | \mathbf{x} = f_d(\mathbf{z}), \mathbf{z} \in \mathcal{Z}\}$, which is the collection of all possible outputs from the decoder $f_d(\mathbf{z})$. The process is visualized in the left panel of Fig. 4.1.

Since $\mathbf{z}$ serves as a coordinate of $\mathcal{M}$, a Gaussian noise in $\mathcal{Z}$ corresponds to perturbation of data along the manifold $\mathcal{M}$, reflecting more relevant mode of

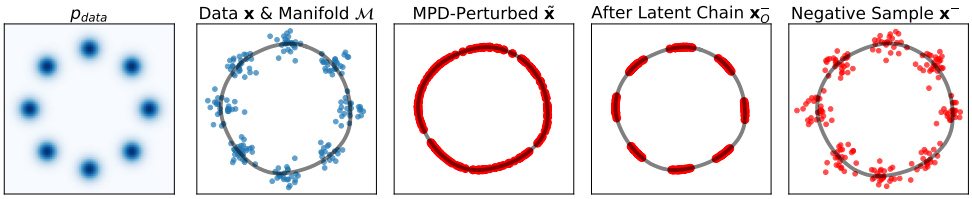

Figure 4.2 Negative sample generation process in MDPR.

variations in data than Gaussian perturbation in $\mathcal{X}$ (Fig. 4.1). Note that MPD reduces to the conventional Gaussian perturbation if we set $\mathcal{Z} = \mathcal{X}$ and set f_e and f_d as the identity mappings.

4.3.2 Manifold Projection-Diffusion Recovery Likelihood

We define the recovery likelihood for MPD as the probability of $\mathbf{x}$ given the perturbed latent vector $\tilde{\mathbf{z}}$.

$$p_\theta(\mathbf{x}|\tilde{\mathbf{z}}) \overset{(i)}{\propto} p(\tilde{\mathbf{z}}|\mathbf{x})p_\theta(\mathbf{x}) = \left(\int p(\tilde{\mathbf{z}}|\mathbf{z})p(\mathbf{z}|\mathbf{x})\mathrm{d}\mathbf{z}\right) p_\theta(\mathbf{x}) \tag{4.6}$$

$$\overset{(ii)}{=} p(\tilde{\mathbf{z}}|\mathbf{z})p_\theta(\mathbf{x}) \propto \exp(-E_\theta(\mathbf{x}) + \log p(\tilde{\mathbf{z}}|\mathbf{z})). \tag{4.7}$$

In (i), we apply Bayes' rule. In equality (ii), we use the fact that the encoder is deterministic and thus $p(\mathbf{z}|\mathbf{x})$ can be treated as $\delta_{\mathbf{z}}(\cdot)$, a Dirac measure on $\mathcal{Z}$ at $\mathbf{z} = f_e(\mathbf{x})$. Since the perturbation probability $p(\tilde{\mathbf{z}}|\mathbf{z})$ is an isotropic Gaussian distribution with standard deviation σ, the log recovery likelihood is evaluated as: $\log p_\theta(\mathbf{x}|\tilde{\mathbf{z}}) = -\tilde{E}_\theta(\mathbf{x}|\tilde{\mathbf{z}}) + \text{const} = -E_\theta(\mathbf{x}) - \frac{1}{2\sigma^2}||\tilde{\mathbf{z}} - f_e(\mathbf{x})||^2 + \text{const}$.

Now, $E_\theta(\mathbf{x})$ is trained by maximizing $\log p_\theta(\mathbf{x}|\tilde{\mathbf{z}})$. The gradient with respect to θ results in the same form with the log likelihood gradient (Eq. 4.2) and the log recovery likelihood gradient (Eq. 4.4) but with a different negative sample distribution $p_\theta(\mathbf{x}|\tilde{\mathbf{z}})$:

$$\nabla_\theta \log p_\theta(\mathbf{x}|\tilde{\mathbf{z}}) = \nabla_\theta E_\theta(\mathbf{x}) - \mathbb{E}_{\mathbf{x}^- \sim p_\theta(\mathbf{x}|\tilde{\mathbf{z}})}[\nabla_\theta E_\theta(\mathbf{x}^-)]. \tag{4.8}$$

Negative samples $\mathbf{x}^-$ are drawn from $p_\theta(\mathbf{x}|\tilde{\mathbf{z}})$ using LMC.

An alternative definition for the recovery likelihood is $p(\mathbf{x}|\tilde{\mathbf{x}})$, which becomes equivalent to $p(\mathbf{x}|\tilde{\mathbf{z}})$ when f_d is an injective function, i.e., no two instances of $\tilde{\mathbf{z}}$ map to the same $\tilde{\mathbf{x}}$. However, if f_d is not injective, additional information loss may occur and becomes difficult to compute. As a result, $p(\mathbf{x}|\tilde{\mathbf{z}})$ serves as a more general and also convenient choice for the recovery likelihood.

4.3.3 Consistency of MPDR

Maximizing $\log p(\mathbf{x}|\tilde{\mathbf{z}})$ results in a consistent estimation of θ, regardless of the choice of f_e, f_d and σ. The required assumptions are similar to those for maximum likelihood estimation, such as infinite data, identifiable and correctly specified model. The key additional assumption for achieving consistency is the independence of $p(\tilde{\mathbf{z}})$ from θ. This can be ensured by maintaining constant values for all parameters of f_e, f_d, and the magnitude of σ throughout the training procedure.

Let us denote our model for recovery likelihood as $p(\mathbf{x}|\tilde{\mathbf{z}};\theta)$. We assume that this model is identifiable (different model parameters correspond to different distribution) and correctly specified (there exists θ^* such that $p(\mathbf{x}|\tilde{\mathbf{z}};\theta^*) = p_{data}(\mathbf{x}|\tilde{\mathbf{z}})$). We also assume that $p(\mathbf{x}|\tilde{\mathbf{z}};\theta)$ is non-zero for all $\mathbf{x}$, $\tilde{\mathbf{z}}$, and θ. Our objective is to $\max_\theta \frac{1}{N}\sum_{i=1}^{N} \log p(\mathbf{x}_i|\tilde{\mathbf{z}}_i;\theta)$ where $(\mathbf{x}_i, \tilde{\mathbf{z}}_i) \sim p(\mathbf{x}, \tilde{\mathbf{z}}) = p_{data}(\mathbf{x})p(\tilde{\mathbf{z}}|\mathbf{x})$. In the limit of $N \to \infty$, the average converges to the expectation $\frac{1}{N}\sum_{i=1}^{N} \log p(\mathbf{x}_i|\tilde{\mathbf{z}}_i;\theta) \to \mathbb{E}_{(\mathbf{x},\tilde{\mathbf{x}})}[\log p(\mathbf{x}|\tilde{\mathbf{z}};\theta)]$. If we subtract $\mathbb{E}_{(\mathbf{x},\tilde{\mathbf{x}})}[\log p(\mathbf{x}|\tilde{\mathbf{z}};\theta^*)]$ which is constant with respect to θ, then the expression can be written with respect to KL divergence as follows:

$$\mathbb{E}_{(\mathbf{x},\tilde{\mathbf{x}})}[\log p(\mathbf{x}|\tilde{\mathbf{z}};\theta) - \log p(\mathbf{x}|\tilde{\mathbf{z}};\theta^*)] = \int p(\mathbf{x},\tilde{\mathbf{z}}) \log \frac{p(\mathbf{x}|\tilde{\mathbf{z}};\theta)}{p(\mathbf{x}|\tilde{\mathbf{z}};\theta^*)} d\mathbf{x}d\tilde{\mathbf{z}} \tag{4.9}$$

$$= -\int p(\tilde{\mathbf{z}})\ \mathrm{KL}(p(\mathbf{x}|\tilde{\mathbf{z}};\theta^*)||p(\mathbf{x}|\tilde{\mathbf{z}};\theta))d\tilde{\mathbf{z}} \tag{4.10}$$

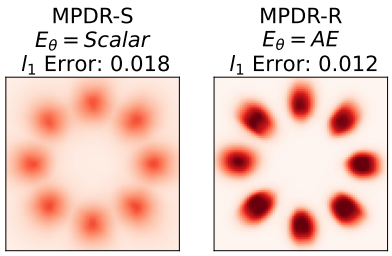

Figure 4.3 2D density estimation with a scalar energy function (left) and an autoencoder-based energy function (right).

The maximum of Eq. 4.10 is 0, as the minimum of KL divergence is 0. The maximum is achieved if and only if $\theta = \theta^*$. Note that $p(\tilde{\mathbf{z}})$ is assumed to be constant with respect to θ.

Since we did not rely on the specifics of how a perturbation $p(\tilde{\mathbf{z}}|\mathbf{x})$ is actually performed, this consistency result holds for any choices of the encoder f_e and the noise magnitude σ, as long as the recovery likelihood $p(\mathbf{x}|\tilde{\mathbf{z}};\theta)$ remains non-zero for all $\mathbf{x}$, $\tilde{\mathbf{z}}$, and θ.

Algorithm 1 Manifold Projection-Diffusion Recovery

1: **while** converged **do**
2: Sample a mini-batch of positive samples $\mathbf{x}$
3: Compute $\tilde{\mathbf{z}} = f_e(\mathbf{x}) + \sigma\epsilon$
4: Sample $\mathbf{z}^-$ from energy $\tilde{H}_\theta(\mathbf{z}|\tilde{\mathbf{z}})$ with LMC on $\mathcal{Z}$ starting from $\tilde{\mathbf{z}}$
5: Sample $\mathbf{x}^-$ from energy $\tilde{E}_\theta(\mathbf{x}|\tilde{\mathbf{z}})$ with LMC
on $\mathcal{X}$ starting from $\mathbf{x}_0^- = f_d(\mathbf{z}^-)$
6: Update θ with the gradient: $-\frac{\partial}{\partial\theta}E_\theta(\mathbf{x}) + \frac{\partial}{\partial\theta}E_\theta(\mathbf{x}^-)$
7: **end while**

4.3.4 Two-Stage Sampling

A default method for drawing $\mathbf{x}^-$ from $p_\theta(\mathbf{x}|\tilde{\mathbf{z}})$ is to execute LMC on $\mathcal{X}$, starting from $\tilde{\mathbf{x}}$, as done in DRL [1]. While this **visible chain** should suffice in principle with infinite chain length, sampling can be improved by leveraging the latent space $\mathcal{Z}$, as demonstrated in [64, 57, 60]. For MPDR, we propose a **latent chain**, a short LMC operating on $\mathcal{Z}$ that generates a better starting point $\mathbf{x}_0^-$ for the visible chain. We first define the auxiliary latent energy $\tilde{H}_\theta(\mathbf{z}) = \tilde{E}_\theta(f_d(\mathbf{z})|\tilde{\mathbf{z}})$, the pullback of the recovery energy through the decoder $f_d(\mathbf{z})$. Next, we run a latent LMC that samples from a probability density proportional to $\exp(-\tilde{H}_\theta(\mathbf{z}))$. The latent chain's outcome, $\mathbf{z}_0^-$, is fed to the decoder to produce the visible chain's starting point $\mathbf{x}_0^- = f_d(\mathbf{z})$, which is likely to have a smaller $\tilde{E}$ value than $\tilde{\mathbf{x}}$. Introducing a small steps of a latent chain improves anomaly detection performance in our experiments. A similar latent-space LMC method appears in [64] but requires a sample replay buffer not used in MPDR. Fig.4.2 illustrates the sampling procedure, and Algorithm1 summarizes the training algorithm.

4.3.5 Perturbation Ensemble

Although the consistency of estimation is independent of the specifics of the perturbation, the design of perturbation, such as perturbation magnitude σ and the architecture of an autoencoder, plays an important role in achieving practical performance. To mitigate the risk of adhering to a single choice of perturbation operation, we propose to employ *multiple* perturbation operations simultaneously during training. Perturbation ensemble retains the consistency of MPDR while improves training stability and anomaly detection performance significantly in our experiments.

Noise Magnitude Ensemble We randomly draw the perturbation magnitude σ from a uniform distribution over a pre-defined interval for each sample in a mini-batch. In our implementation, we draw σ from the interval $[0.05, 0.3]$ throughout all the experiments.

Manifold Ensemble We can also utilize multiple autoencoder manifolds $\mathcal{M}$ in MPD. Given K autoencoders, a mini-batch is divided into K equally sized groups. For each group, negative samples are separately generated using the corresponding autoencoder. The resulting negative samples are then aggregated again to a mini-batch for model update. Only a minimal increase in training time is observed as K increases, since each autoencoder processes a smaller mini-batch. Memory overhead does exist, as multiple autoencoders must be loaded onto the GPU. However, this overhead is manageable, since MPDR achieves good performance with relatively smaller autoencoders.

Among multiple possible options for constructing a manifold ensemble, an effective strategy is to utilize autoencoders with varying latent space dimensionalities $D_{\mathbf{z}}$. For high-dimensional data, such as images, $\mathcal{M}$ with different $D_{\mathbf{z}}$ tend to capture different mode of variation in data. A perturbation on $\mathcal{M}$ with small $D_{\mathbf{z}}$ corresponds to low-frequency variation in $\mathcal{X}$, whereas for $\mathcal{M}$ with large $D_{\mathbf{z}}$, it corresponds to higher-frequency variation. Using multiples $D_{\mathbf{z}}$'s in MPDR gives us more diverse $\mathbf{x}^-$ and eventually better anomaly detection performance.

4.3.6 Energy Function Design

MPDR is a versatile training algorithm for general EBMs, compatible with various types of energy functions. The design of an energy function plays a crucial role in anomaly detection performance, as the inductive bias of an energy governs its behavior in out-of-distribution regions. We primarily explore

Table 4.1 MNIST hold-out digit detection. Performance is measured in AUPR. Standard deviation of AUPR is computed over the last 10 epochs. The largest mean value is marked in bold, while the second-largest is underlined.

Hold-Out Digit	1	4	5	7	9
AE	0.062 ± 0.000	0.204 ± 0.003	0.259 ± 0.006	0.125 ± 0.003	0.113 ± 0.001
IGEBM	0.101 ± 0.020	0.106 ± 0.019	0.205 ± 0.108	0.100 ± 0.042	0.079 ± 0.015
MEG [65]	0.281 ± 0.035	0.401 ± 0.061	0.402 ± 0.062	0.290 ± 0.040	0.324 ± 0.034
BiGAN-σ [66]	0.287 ± 0.023	0.443 ± 0.029	0.514 ± 0.029	0.347 ± 0.017	0.307 ± 0.028
Latent EBM[60]	0.336 ± 0.008	0.630 ± 0.017	0.619 ± 0.013	0.463 ± 0.009	0.413 ± 0.010
VAE+EBM [59]	0.297 ± 0.033	<u>0.723</u> ± 0.042	0.676 ± 0.041	0.490 ± 0.041	0.383 ± 0.025
NAE [64]	<u>0.802</u> ± 0.079	0.648 ± 0.045	0.716 ± 0.032	0.789 ± 0.041	0.441 ± 0.067
MPDR-S (ours)	0.764 ± 0.045	**0.823** ± 0.018	<u>0.741</u> ± 0.041	**0.857** ± 0.022	<u>0.478</u> ± 0.048
MPDR-R (ours)	**0.844** ± 0.030	0.711 ± 0.029	**0.757** ± 0.024	<u>0.850</u> ± 0.014	**0.569** ± 0.036

two designs for energy functions: MPDR-Scalar (**MPDR-S**), a feed-forward neural network that takes input $\mathbf{x}$ and produces a scalar output, and MPDR-Reconstruction (**MPDR-R**), the reconstruction error from an autoencoder, $E_\theta(\mathbf{x}) = ||\mathbf{x} - g_d(g_e(\mathbf{x}))||^2$, for an encoder g_e and a decoder g_d. The autoencoder (g_e, g_d) is separate from the autoencoder used in MPD. First proposed in [64], an reconstruction-based energy function has an inductive bias towards assigning high energy values to off-manifold regions (Fig. 4.3). Training such an energy function using conventional techniques like CD [15] or persistent chains [16, 25] is reported to be challenging [64]. However, MPDR effectively trains both scalar and reconstruction energies. Additionally, in Sec. 4.4.4, we demonstrate that MPDR is also compatible with an energy function based on a masked autoencoder.

Table 4.2 MNIST OOD detection performance measured in AUPR. We test models from hold-out digit 9 experiment (Table 1). The overall performance is high, as detecting these outliers is easier than identifying the hold-out digit.

	KMNIST	EMNIST	Omniglot	FashionMNIST	Constant
AE	0.999	0.977	0.947	1.000	0.954
IGEBM	0.990	0.923	0.845	0.996	1.000
NAE	1.000	0.993	0.997	1.000	1.000
MPDR-FF	0.999	0.995	0.994	0.999	1.000
MPDR-AE	0.999	0.989	0.997	0.999	0.990

4.4 Experiment

4.4.1 Implementation of MPDR

An autoencoder (f_e, f_d) is trained by minimizing the reconstruction error of the training data and remains fixed during the training of $E_\theta(\mathbf{x})$. When using an ensemble of manifolds, each autoencoder is trained independently. For anomaly detection, the energy value $E_\theta(\mathbf{x})$ serves as an anomaly score which assigns a high value for anomalous $\mathbf{x}$. All optimizations are performed using Adam with a learning rate of 0.0001. Each run is executed on a single Tesla V100 GPU.

Spherical Latent Space In all our implementations of autoencoders, we utilize a hyperspherical latent space $\mathcal{Z} = \mathbb{S}^{D_\mathbf{z}-1}$ [43, 44, 45]. The encoder output is projected onto $\mathbb{S}^{D_\mathbf{z}-1}$ via division by its norm before being fed into the decoder. Employing $\mathbb{S}^{D_\mathbf{z}-1}$ standardizes the length scale of $\mathcal{Z}$, allowing us to use the same value of σ across various data sets and autoencoder architectures. Meanwhile, the impact of $\mathbb{S}^{D_\mathbf{z}-1}$ on the reconstruction error is minimal.

Regularization For a scalar energy function, MPDR-S, we add $L_{reg} = E_\theta(\mathbf{x})^2 + E_\theta(\mathbf{x}^-)^2$ to the loss function, as proposed by [25]. For a reconstruction

energy function, MPDR-R, we add $L_{reg} = E_\theta(\mathbf{x}^-)^2$, following [64].

Scaling Perturbation Probability Applying regularization to an energy restricts its scale, causing a mismatch in scales between the two terms in the recovery likelihood (Eq. (4.3.2)). To remedy this mismatch, we heuristically introduce a scale factor $\gamma < 1$ to $\log p(\mathbf{z}|\tilde{\mathbf{z}})$, resulting in the modified recovery energy $\tilde{E}_\theta^{(\gamma)}(\mathbf{x}|\tilde{\mathbf{z}}) = E_\theta(\mathbf{x}) + \frac{\gamma}{2\sigma^2}||\tilde{\mathbf{z}} - f_e(\mathbf{x})||^2$. We use $\gamma = 0.0001$ for all experiments.

4.4.2 2D Density Estimation

We show MPDR's ability to estimate multi-modal density using a mixture of eight circularly arranged 2D Gaussians (Fig. 4.2). We construct an autoencoder with $\mathcal{S}^1$ latent space, which approximately captures the circular arrangement. The encoder and the decoder are MLPs with two layers of 128 hidden neurons. To show the impact of the design of energy functions, we implement both scalar energy and reconstruction energy. Three-hidden-layer MLPs are used for the scalar energy function, and the encoder and the decoder in the reconstruction energy function. Note that the network architecture of the reconstruction energy is not the same as the autoencoder used for MPD. The density estimation results are presented in Fig. 4.3. We quantify density estimation performance using l_1 error. After numerically normalizing the energy function and true density on the visualized bounded domain, we compute the l_1 error at 10,000 grid points. While both energies capture the overall landscape of the density, the reconstruction energy achieves a smaller error by suppressing probability density at off-manifold regions.

4.4.3 Image Out-of-Distribution Detection

MNIST Hold-Out Digit Detection Following the protocol of [65, 66], we evaluated the performance of MPDR on MNIST hold-out digit detection benchmark, where one of the ten digits in the MNIST dataset is considered anomalous, and the remaining digits are treated as in-distribution. This is a challenging task due to the diversity of the in-distribution data and a high degree of similarity between target anomalies and inliers. In particular, selecting digits 1, 4, 5, 7, and 9 as anomalous is known to be especially difficult. The results are shown in Table 4.1.

In MPDR, we use a single autoencoder (f_e, f_d) with $D_{\mathbf{z}} = 32$. The energy function of MPDR-S is initialized from scratch, and the energy function of MPDR-R is initialized from the (f_e, f_d) used in MPD. Even without a manifold ensemble, MPDR shows significant improvement over existing algorithms, including ones leveraging an autoencoder in EBM training [59, 64].

MNIST OOD Detection To ensure that MPDR is not overfitted to the hold-out digit, we test MPDR in detecting five non-MNIST outlier datasets (Table 4.2). The results demonstrated that MPDR excels in detecting a wide range of outliers, surpassing the performance of naive algorithms such as autoencoders (AE) and scalar EBMs (IGEBM). Although MPDR achieves high overall detection performance, MPDR-R exhibits slightly weaker performance on EMNIST and Constant datasets. This can be attributed to the limited flexibility of the autoencoder-based energy function employed in MPDR-R.

CIFAR-10 OOD Detection We evaluate MPDR on the CIFAR-10 inliers, a standard benchmark for EBM-based OOD detection. The manifold ensemble includes three convolutional autoencoders, with $D_{\mathbf{z}} = 32, 64, 128$. MPDR-S uses

Table 4.3 OOD detection with CIFAR-10 as in-distribution. AUROC values are presented. The largest value in the column is marked as boldface, and the second and the third largest values are underlined.

	SVHN	Textures	Constant	CIFAR100	CelebA
PixelCNN++ [48]	0.32	0.33	0.71	0.63	-
GLOW [49]	0.24	0.27	-	0.55	0.57
IGEBM [25]	0.63	0.48	0.39	-	
NVAE [61]	0.42	-	-	0.56	0.68
VAEBM [57]	0.83	-	-	0.62	0.77
JEM [26]	0.67	0.60	-	0.67	0.75
Improved CD [56]	0.7843	0.7275	0.8000	0.5298	0.5399
NAE [64]	0.9352	<u>0.7472</u>	0.9793	0.6316	**0.8735**
DRL [1]	0.8816	0.4465	0.4377	<u>0.6398</u>	
CLEL [67]	<u>0.9848</u>	**0.9437**	-	**0.7161**	0.7717
MPDR-S (ours)	**0.9860**	0.6583	**0.9996**	0.5576	0.7313
MPDR-R (ours)	<u>0.9807</u>	<u>0.7978</u>	**0.9996**	<u>0.6354</u>	<u>0.8282</u>

Table 4.4 OOD detection on pretrained ViT-B_16 representation with CIFAR-100 as in-distribution. Performance is measured in AUROC.

	CIFAR10	SVHN	CelebA
Supervised			
MD [68]	0.8634	0.9638	0.8833
RMD [69]	0.9159	0.9685	0.4971
Unsupervised			
AE	0.8580	0.9645	0.8103
NAE	0.8041	0.9082	0.8181
IGEBM	0.8217	0.9584	0.9004
MPDR-S	0.8338	0.9911	**0.9183**
MPDR-R	**0.8626**	**0.9932**	0.8662

a ResNet energy function used in IGEBM [25]. MPDR-R adopts the ResNet-based autoencoder architecture used in NAE [64].

Table 4.3 compares MPDR to state-of-the-art EBMs. MPDR-R shows competitive performance across five OOD datasets, while MPDR-S also achieves high AUROC on SVHN and Constant. As both MPDR-R and NAE use the same autoencoder architecture for the energy, the discrepancy in performance can be attributed to the MPDR training algorithm. MPDR-R outperforms NAE on four out of five OOD datasets. Comparison between MPDR-S and DRL demonstrates the effectiveness of non-Gaussian manifold-aware perturbation used in MPDR. CLEL shows strong overall performance, indicating that learning semantic information is important in this benchmark. Incorporating contrastive learning into MPDR framework is an interesting future direction.

CIFAR-100 OOD Detection on Pretrained Representation In Table 4.4, we test MPDR on OOD detection with CIFAR-100 inliers. To model a distribution of diverse images like CIFAR-100, we follow [68] and apply generative modeling in the representation space from a large-scale pretrained model. As we assume an unsupervised setting, we use pretrained representations without fine-tuning. Input images are transformed into 768D vectors by ViT-B_16 model [70]. ViT outputs are normalized with its norm and projected onto a hypersphere. We observe that adding a small Gaussian noise of 0.01 to training data improves stability of all algorithms. We use MLP for all energy functions and autoencoders. In MPDR, the manifold ensemble comprises three autoencoders with $D_{\mathbf{z}} = 128, 256, 1024$. We also implement supervised baselines (MD [68] and RMD [69]). The spherical projection is not applied for MD and RMD to respect their original implementation.

MPDR demonstrates strong anomaly detection performance in the representation space, with MPDR-S and MPDR-R outperforming IGEBM and AE/NAE, respectively. This success can be attributed to the low-dimensional structure often found in the representation space of in-distribution data, as observed in [71]. MPDR's average performance is nearly on par with supervised methods, MD and RMD, which utilize class information. Note that EBM inputs are no longer images, making previous EBM training techniques based on image transformation [56, 67] inapplicable.

4.4.4 Acoustic Anomaly Detection

We apply MPDR to anomaly detection with acoustic signals another non-image data. We use DCASE 2020 Challenge Task 2 dataset [72], which contains recordings from six different machines, with three to four instances per machine type. The task is to detect anomalous sounds from deliberately damaged

machines, which are unavailable during training. Following the standard preprocessing [72], each 10s audio clip in the dataset is splitted into 64ms frames and transformed into 128-dimensional Mel spectrogram feature vectors. Each model prediction is based on five consecutive frames which is a 640-dimensional vector. Many challenge submissions exploit dataset-specific heuristics and ensembles for high performance, e.g., [73, 74]. Rather than competing, we focus on demonstrating MPDR's effectiveness in improving common approaches, such as autoencoder-based anomaly detection and Interpolation Deep Neural Network (IDNN) [75].

IDNN is an instance of a masked autoencoder which predicts the middle (the third) frame given the remaining frames. Similarly to autoencoders, IDNN predicts an input as anomaly when the prediction error is large. We first train AE and IDNN for 100 epochs and then apply MPDR by treating the reconstruction (or prediction) error as the energy. Manifold ensemble consists of autoencoders with $D_{\mathbf{z}} = 32, 64, 128$.

Table 4.7 shows that MPDR improves anomaly detection performance for both AE and IDNN. The known failure mode of AE and IDNN is producing unexpected low prediction error for anomalous inputs. By treating the prediction error as the energy and applying generative training through MPDR suppresses low prediction error for anomalies, resulting in improved anomaly detection performance.

4.4.5 Ablation Study

Table 4.3 and 4.4 also report the results from single-manifold MPDR-R with varying latent dimensionality $D_{\mathbf{z}}$ to show MPDR's sensitivity to a choice of an autoencoder manifold. Manifold ensemble effectively hedges the risk of relying on a single autoencoder which may not be optimal for detecting all types of

outliers. Furthermore, manifold ensemble often achieves better AUROC score than each component autoencoder. First, we examine the sensitivity of MPDR to noise magnitude σ and the effectiveness of the noise magnitude ensemble. Second, we investigate the effect to scaling parameter γ and show that the training is unstable when γ is too large. Third, we also explore multiple choices of autoencoder design.

Sensitivity to $D_{\mathbf{z}}$ As an ablation study for manifold ensemble, we investigate the sensitivity of MPDR to the choice of the latent space dimensionality of the autoencoder. We evaluate the OOD detection performance of MPDR-S on the MNIST hold-out digit 9 setting with varying values of $D_{\mathbf{z}}$. Table 4.6 presents the results. Consequently, MPDR runs stably for a large range of $D_{\mathbf{z}}$, producing decent OOD performance. One hypothesis is that, for MNIST, it is relatively easy for these autoencoders to capture the manifold structure of MNIST sufficiently well.

Meanwhile, we do observe that the choice of $D_{\mathbf{z}}$ affects OOD performance in an interesting way. Increasing $D_{\mathbf{z}}$ enhances AUPR for certain OOD datasets but deteriorates AUPR for others. For example, AUPR of Omniglot is increased with larger $D_{\mathbf{z}}$, but AUPR of EMNIST, FashionMNIST, and Constant dataset decreases. No single autoencoder is optimal for detecting all outlier datasets. This observation motivates the use of manifold ensemble, employed in non-MNIST MPDR experiments.

Table 4.5 Sensitivity to σ. MPDR-S is run with an autoencoder with varying values of noise magnitude σ. AUPR against various outlier datasets are presented. For MNIST 9, we present the standard deviation computed over the last 10 epochs.

σ	MNIST 9	KMNIST	EMNIST	Omniglot	FashionMNIST	Constant
0.01	0.098 ± 0.009	0.667	0.827	0.940	0.944	0.895
0.1	0.330 ± 0.063	0.983	0.995	0.971	0.994	0.998
0.2	0.522 ± 0.053	0.999	0.993	0.997	0.999	1.000
0.3	0.558 ± 0.039	1.000	0.990	0.997	1.000	1.000
Ens.	0.478 ± 0.048	0.999	0.995	0.994	0.999	1.000

Table 4.6 Sensitivity to $D_{\mathbf{z}}$. MPDR-S is run with an autoencoder with varying values of $D_{\mathbf{z}}$. AUPR against various outlier datasets are presented. For MNIST 9, we present the standard deviation computed over the last 10 epochs. Noise magnitude ensemble is applied.

$D_{\mathbf{z}}$	MNIST 9	KMNIST	EMNIST	Omniglot	FashionMNIST	Constant
16	0.611 ± 0.041	0.979	0.996	0.958	0.999	1.000
32	0.525 ± 0.039	0.999	0.994	0.994	0.999	1.000
64	0.512 ± 0.048	0.999	0.993	0.998	0.998	0.999
128	0.505 ± 0.051	0.999	0.991	0.999	0.988	0.946
256	0.590 ± 0.041	0.996	0.983	0.996	0.958	0.866

Table 4.7 Acoustic anomaly detection on DCASE2020 Track 2 Dataset. AUROC and pAUROC (in parenthesis) are displayed per cent.

	Toy Car	Toy Conveyor	Fan	Pump	Slider	Valve
AE [72]	75.40 (62.03)	77.38 (63.02)	66.44 (53.40)	71.42 (61.77)	89.65 (74.69)	72.52 (**52.02**)
MPDR-R	**81.54** (**68.21**)	**78.61** (**63.99**)	**71.72** (**55.95**)	**78.27** (**68.14**)	**90.91** (**76.58**)	**75.23** (51.04)
IDNN [75]	76.15 (72.36)	78.87 (62.50)	72.74 (54.30)	73.15 (61.25)	90.83 (74.16)	90.27 (69.46)
MPDR-IDNN	**78.53** (**73.34**)	**79.54** (**65.35**)	**73.27** (**54.57**)	**76.58** (**66.49**)	**91.56** (**75.19**)	**91.10** (**70.87**)

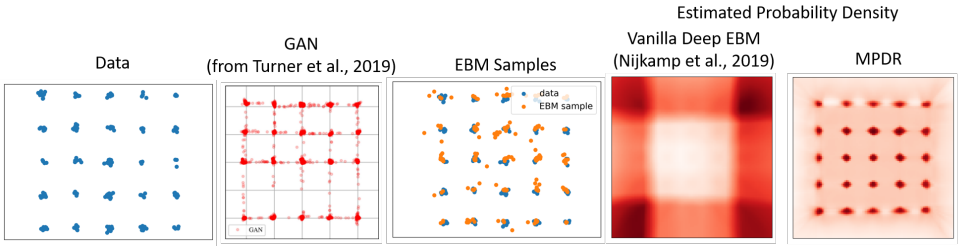

Figure 4.4 Mode collapse experiment. (First) Samples drawn from 25 Gaussians. (Second) Samples from GAN. The figure is adopted from [2]. (Third) Samples generated from MPDR. Samples generated from a vanilla EBM trained with short-run MCMC are distributed similarly. (Fourth) Density estimated by EBM trained with short-run MCMC. (Last) Density estimated using MPDR.

4.4.6 Mode Collapse

We investigate whether MPDR can cover multiple modes of a data distribution. Some generative models, such as generative adversarial networks (GAN) are prone to exhibit the mode collapse phenomenon, where a model only covers a subset of modes in a data distribution. This phenomenon is demonstrated in the second figure of Fig. 4.4.

We train MPDR on 25 Gaussians dataset, a standard synthetic dataset for demonstrating the mode collapsing behavior. GAN is known to fail at covering all 25 modes of data [2]. We find that MPDR as well as a vanilla EBM trained with short-run MCMC are capable of generating samples covering all modes of data. However, the vanilla EBM does not produce an accurate density estimate. Meanwhile, MPDR produces a density estimate that captures all 25 modes.

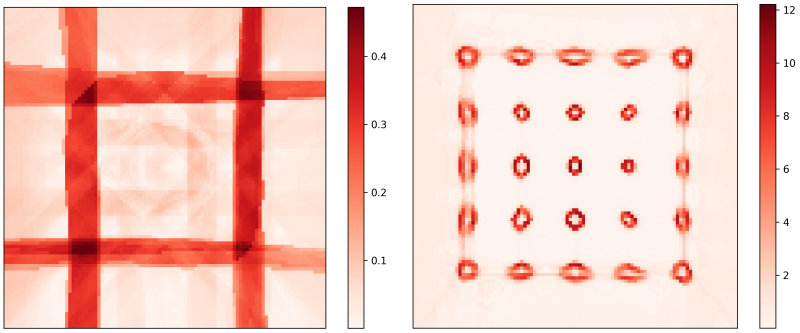

Figure 4.5 Visualization of $||\nabla_{\mathbf{x}} \log p(\mathbf{x})||$ from a vanilla EBM (left) and MPDR (right).

4.4.7 Comparison to Score-Based OOD Detection

We check whether the norm of score $||\nabla_{\mathbf{x}} \log p_\theta(\mathbf{x})||$ can be a better OOD score. The score norm is shown to improve OOD detection performance in some cases [26]. The score norm can be easily computed as the norm of energy $||\nabla_{\mathbf{x}} E_\theta(\mathbf{x})||$. Fig 4.5 visualizes $||\nabla_{\mathbf{x}} E_\theta(\mathbf{x})||$ of a vanilla EBM and MPDR, trained in Fig 4.4. As can be seen in the figure, the region with high score norm does not overlap with the high density region. Therefore, $||\nabla_{\mathbf{x}} E_\theta(\mathbf{x})||$ is not a good OOD score in 2D setting.

We also investigate whether the score norm is effective in higher-dimensional problems. We compute $||\nabla_{\mathbf{x}} E_\theta(\mathbf{x})||$ of MPDR trained on CIFAR-10. The result is shown in Table 4.8. Replacing the OOD score from $E_\theta(\mathbf{x})$ to $||\nabla_{\mathbf{x}} E_\theta(\mathbf{x})||$ degrades OOD detection performance in all tested OOD datasets.

4.5 Conclusion

Contributions In this paper, we proposed MPDR, a novel objective function for EBM, which leverages the low-dimensional structure of data. MPDR

MPDR-R	SVHN	Texture	constant	CIFAR-100	CelebA
$E_\theta(\mathbf{x})$	**0.9807**	**0.7978**	**0.9996**	**0.6354**	**0.8282**
$\|\|\nabla_{\mathbf{x}} E_\theta(\mathbf{x})\|\|$	0.7895	0.6450	0.2638	0.5758	0.7543

Table 4.8 CIFAR-10 OOD detection experiment with the score norm as OOD score.

achieves strong out-of-detection performance while circumventing the need for entire-space MCMC by employing recovery likelihood formulation, perturbation ensemble, and autoencoder-based energy functions. Our work serves as a contribution towards building generative models capable of identifying anomalies in high-dimensional data.

Limitations First, the theory behind the MPDR algorithm does not offer specific guidance on designing the autoencoder for optimal anomaly detection performance, resulting in some degree of uncertainty. To mitigate the risk of selecting a suboptimal autoencoder, we introduce the perturbation ensemble technique and provide guidance based on our experimental findings. Second, not all data exhibit a distinct low-dimensional structure. More challenging data types, such as high-resolution images or texts, may lack a single pronounced manifold structure or, at the very least, may not be effectively modeled by an autoencoder. In such cases, generative approaches, including MPDR, might be more successful in the representation space learned by a separate model, potentially trained with a larger dataset.

Table 4.9 Hyperparameters for LMC. Latent chain hyperparameters are denoted by $\mathcal{Z}$ and $\mathcal{X}$ indicates visible chain hyperparameters. "scale (γ)" refers to the multiplicative scale factor on the perturbation probability.

Experiment	$\mathcal{Z}$ Steps	$\mathcal{Z}$ Step Size	$\mathcal{Z}$ Noise	$\mathcal{Z}$ scale (γ)	$\mathcal{X}$ Steps	$\mathcal{X}$ Step Size	$\mathcal{X}$ Noise	$\mathcal{X}$ scale (γ)
MNIST								
MPDR-S	2	0.05	0.02	0.0001	5	10	0.005	0
MPDR-R	5	0.1	0.02	0.0001	5	10	0.005	0
CIFAR-10								
MPDR-S	10	0.1	0.01	0.0001	20	10	0.005	0
MPDR-R	10	0.1	0.01	0.0001	20	10	0.005	0
CIFAR-100 + ViT								
MPDR-S	0	-	-	-	30	1	0.005	0.0001
MPDR-R	0	-	-	-	30	1	0.005	0.0001
DCASE 2020 (Toy Car, Toy Conveyor, Pump)								
MPDR-R	0	-	-	-	5	10	0.005	0.0001
MPDR-IDNN	0	-	-	-	5	10	0.005	0.0001
DCASE 2020 (Fan, Slider, Valve)								
MPDR-R	0	-	-	-	5	10	0.005	0.0001
MPDR-IDNN	20	0.1	0.01	0.0001	20	10	0.005	0.0001

4.6 Experimental Details and Additional Results

In this section, we provide detailed information on how each experiment is conducted and also provide some additional supporting experimental results. The hyperparameters related to LMC is summarized in Table 4.9. ConvNet architectures are provided in Table 4.10. The contents are organized by the training dataset.

4.6.1 MNIST

Datasets

All input images are 28×28 grayscale, and each pixel value is represented as a floating number between $[0, 1]$. Models are trained on the training split of MNIST[1], excluding the digit designated to be held-out. The training split contains 60,000 images, and the hold-out procedure reduces the training set to approximately 90%. We evaluate the models on the test split of MNIST, which contains a total of 10,000 images. For non-MNIST datasets used in evaluation, we only use their test split. All non-MNIST datasets used in the experiment also follow the 28×28 grayscale format, similar to MNIST.

- KMNIST (KMNIST-MNIST) [76] [2] contains Japanese handwritten letters, pre-processed into the same format as MNIST. The license of KMNIST is CC BY-SA 4.0.
- EMNIST (EMNIST-Letters) [77] contains grayscale handwritten English alphabet images. Its test split contains 20,800 samples. No license information is available.
- For the Omniglot [78] dataset, the evaluation set is used. The set consists of

[1] `http://yann.lecun.com/exdb/mnist/`
[2] `https://github.com/rois-codh/kmnist`

13,180 images. No license information is available.

- We use the test split of FashionMNIST [79], which contains 10,000 images. The dataset is made public under the MIT license.
- Constant dataset is a synthetic dataset that contains 28×28 images, where all pixels have the same value. The value is randomly drawn from a uniform distribution over $[0, 1]$. We use 4,000 constant images.

Autoencoder Implementation and Training

The encoder and the decoder used in MPDR, f_e and f_d, have the architecture of MNIST Encoder and MNIST Decoder, provided in Table 4.10, respectively. We use the spherical latent space $\mathbb{S}^{D_{\mathbf{z}}-1}$ where $D_{\mathbf{z}} = 32$ for the main experiment. The autoencoder is trained to minimize the l_2 reconstruction error of the training data for 30 epochs with Adam of learning rate 1×10^{-4}. The batch size is 128 and no data augmentation is applied. The l_2 norm of the encoder is regularized with the coefficient of 1×10^{-4}. The same autoencoder is used for both MPDR-S and MPDR-R.

MPDR Implementation and Training

In MPDR-S, the energy function E_θ has the architecture of MNIST Encoder (Table 4.10) with $D_{\mathbf{z}} = 1$. The network is randomly initialized from PyTorch default setting and the spectral normalization is applied. The energy function is trained with MPDR algorithm for 50 epochs. The learning rate is 1×10^{-4}. The batch size is 128. The perturbation probability scaling factor γ for the visible LMC chain is set to zero. For an image-like data such as MNIST, the gradient of perturbation probability $\nabla_{\mathbf{x}}(||\tilde{\mathbf{z}} - f_e(\mathbf{x})||^2)$ introduces non-smooth high-frequency patterns resembling adversarial noise, harming the stability of the training. Therefore, we only use non-zero γ in latent chains in MNIST and

CIFAR-10 experiments.

For MPDR-R, the energy function is initialized from (f_e, f_d) and then trained through MPDR algorithm. The learning rate is set to 1×10^{-5}. All the other details are identical to the MPDR-S case.

Table 4.10 Convolutional neural network architectures used in experiments. The parenthesis following the network name indicates the activation function used in the network.

MNIST Encoder (ReLU)	MNIST Decoder (ReLU)	CIFAR-10 Encoder 1
Conv_3(1, 32)	ConvT_4($D_{\mathbf{z}}$, 128)	Conv_4(3, 32, stride=2)
Conv_3(32, 64)	Upsample(2x)	Conv_4(32, 64, stride=2)
MaxPool(2x)	ConvT_3(128, 64)	Conv_4(64, 128, stride=2)
Conv_3(64, 64)	ConvT_3(64, 64)	Conv_2(128, 256, stride=2)
Conv_3(64, 128)	Upsample(2x)	FC(256, $D_{\mathbf{z}}$)
MaxPool(2x)	ConvT_3(64, 32)	
Conv_4(128, 1024)	ConvT_3(32, 1)	
FC(1024, $D_{\mathbf{z}}$)	Sigmoid()	

CIFAR-10 Decoder 1
ConvT_8($D_{\mathbf{z}}$, 256)
ConvT_4(256, 128, stride=2, pad=1)
ConvT_4(128, 64, stride=2, pad=1)
ConvT_1(64, 3)
Sigmoid()

CIFAR-10 Encoder 2	CIFAR-10 Decoder 2
Conv_3(3, 128, pad=1)	ConvT_4($D_{\mathbf{z}}$, 128)
ResBlock(128, 128, down=True)	ResBlock(128, 128, up=True)
ResBlock(128, 128)	ResBlock(128, 128, up=True)
ResBlock(128, 256, down=True)	ResBlock(128, 128, up=True)
ResBlock(256, 256)	Conv_3(128, 3, pad=1)
ResBlock(256, 256, down=True)	
ResBlock(256, 256)	
GlobalAvgPool()	
FC(256, $D_{\mathbf{z}}$)	

4.6.2 Sensitivity to σ

As an ablation study for noise magnitude ensemble, we perform single-noise-magnitude experiment for MPDR and examine MPDR's sensitivity to the noise magnitude σ. We evaluate the OOD detection performance of MPDR-S on the

MNIST hold-out digit 9 setting with varying values of σ. Results are shown in Table 4.5.

The choice of σ has a significant impact on MPDR's OOD detection performance. In Table 4.5, $\sigma = 0.01$ gives poor OOD detection performance, particularly with respect to the hold-out digit. The performance generally improves as σ grows larger, but a large σ is not optimal for detecting EMNIST. Selecting a single optimal σ will be very difficult, and therefore, we employ noise magnitude ensemble which can hedge the risk of choosing a suboptimal value for σ.

Note on Reproduction

For an autoencoder-based outlier detector, denoted as "AE" in Table 4.1, we use the same autoencoder used in MPDR with $D_{\mathbf{z}}$.

NAE is reproduced based on its public code base[3].

We tried to reproduce DRL on MNIST but failed to train it stably. The original paper and the official code base also does not provide a guidance on training DRL on MNIST.

4.6.3 CIFAR-10 OOD Detection

Datasets

All data used in CIFAR-10 experiment are in the 32×32 RGB format. Models are only trained on CIFAR-10 training set, and evaluated on the testing set of each dataset.

- CIFAR-10 [80] contains 60,000 training images and 10,000 testing images. Models are trained on the training set. We don't use its class information, as we consider only unsupervised setting. No license information available.

[3] `https://github.com/swyoon/normalized-autoencoders`

- SVHN [81] is a set of digit images. Its test set containing 26,032 is used in the experiment. The dataset is non-commercial use only.
- Texture [82] dataset, also called Describable Textures Dataset (DTD), contains 1,880 test images. The images are resized into 32×32. No license information available.
- CelebA [83][4] is a dataset of cropped and aligned human face images. The test set contains 19,962 images. The dataset is for non-commercial research purposes only. We center-crop each image into 140×140 and then resize it into 32×32.
- Constant dataset is a synthetic dataset that contains 4,000 32×32 RGB monochrome image. All 32×32 pixels have the same RGB value which is drawn uniform-randomly from $[0, 1]^3$.
- CIFAR-100 [80] contains 60,000 training images and 10,000 testing images. No license information available.

Autoencoder Implementation and Training

The autoencoders for CIFAR-10 experiment have an architecture of "CIFAR-10 Encoder 1" and "CIFAR-10 Decoder 1" in Table 4.10 with $D_{\mathbf{z}} = 32, 64, 128$. Each autoencoder is trained for 40 epoch with learning rate 1×10^{-4} and batch size 128. During training the autoencoders, we apply the following data augmentation operations: random horizontal flipping with the probability of 0.5, random resize crop with the scale parameter $[0.08, 1]$ with the probability 0.2, color jittering with probability of 0.2, random grayscale operation with the probability 0.2. The color jittering parameters are the same with the one used in SimCLR [84] (brightness 0.8, contrast 0.8, saturation 0.8, hue 0.4, with respect to `torchvision.transorms.ColorJitter` implementation). The l_2 norm of an

[4] `https://mmlab.ie.cuhk.edu.hk/projects/CelebA.html`

encoder is regularized with the coefficient of 0.00001. The same autoencoder is used for both MPDR-S and MPDR-R.

MPDR Implementation and Training

The energy function in MPDR-S is "CIFAR-10 Encoder 2" with $D_{\mathbf{z}} = 1$. The energy function in MPDR-R is "CIFAR-10 Encoder 2" and "CIFAR-10 Decoder 2" with $D_{\mathbf{z}} = 1$. In MPDR-R, the energy function is pre-trained by minimizing the reconstruction error of the training data for 40 epochs. Only random horizontal flip is applied and no other data augmentation is used. Similarly to the MNIST case, we set $\gamma = 0$ for the visible LMC chain.

Note on Reproduction

For Improved CD, we train a CIFAR-10 model from scratch using the training script provided by the authors without any modification [5]. We use the model with the best Inception Score to compute AUROC scores.

For DRL, we use the official checkpoint for T6 CIFAR-10 model [6] and compute its energy to perform OOD detection.

Also for NVAE, we use the official CIFAR-10 checkpoint provided in the official repository [7]. We use negative log-likelihood as the outlier score. Due to computational budget constraint, we could only set the number of importance weighted sample to be one `--num_iw_samples=1`.

As in MNIST, we use the official CIFAR-10 checkpoint of NAE provided by the authors.

[5] `https://github.com/yilundu/improved_contrastive_divergence`
[6] `https://github.com/ruiqigao/recovery_likelihood`
[7] `https://github.com/NVlabs/NVAE`

4.6.4 CIFAR-100 OOD Detection on Pretrained Representation

Datasets

CIFAR-100, CIFAR-10, SVHN, and CelebA datasets are used and are described in the previous section. Each image is resized to 224×224 and fed to ViT-B_16 to produce a 768-dimensional vector. MD and RMD operate with this vector. For other methods, the 768D vector is projected onto a hypersphere.

Autoencoder Implementation and Training

Each encoder and decoder is an MLP with two hidden layers where each layer contains 1024 hidden neurons. The leaky ReLU activation function is used in all hidden layers. We use $D_{\mathbf{z}} = 128, 256, 1024$. The autoencoders are trained to minimize the reconstruction error of the training data. During training, the Gaussian noise with the standard deviation of 0.01 is added to each training sample. The l_2 norm of the encoder's weights are regularized with the coefficient of 1×10^{-6}.

MPDR Implementation and Training

The energy functions are also MLPs. The energy function of MPDR-S has the same architecture as the encoder of the autoencoder with $D_{\mathbf{z}} = 1$. The energy function of MPDR-R is an autoencoder with the latent dimensionality of 1024. The energy functions are trained for 30 epochs with the learning rate of 1×10^{-4}.

Sensitivity to γ

We examine how γ affects MPDR's performance. As seen in Table 4.11, MPDR shows the best performance on the small γ regime, roughly from 0.0001 to 0.001. Setting too large γ is detrimental for the performance and often even incurs training instabilities. It is interesting to note that $\gamma = 0$ also gives a

decent result. One possible explanation is that $\gamma = 0$ reduces the negative sample distribution $p_\theta(\mathbf{x}|\tilde{\mathbf{z}})$ to the model distribution $p_\theta(\mathbf{x})$, which is still a valid negative sample distribution for training an EBM.

Table 4.11 Sensitivity of γ, demonstrated in CIFAR-100 experiment. AUROC values are displayed.

γ	CIFAR10	SVHN	CelebA
0	0.8580	0.9931	0.8456
0.0001	0.8626	0.9932	0.8662
0.001	0.8639	0.9918	0.8625
0.01	0.8496	0.9894	0.8576
0.1	0.8186	0.9424	0.8511

4.6.5 Acoustic Anomaly Detection

Dataset

The dataset consists of audio recordings with a duration of approximately 10 seconds, obtained through a single channel and downsampled to 16kHz. Each recording includes both the operational sounds of the target machine and background noise from the surrounding environment. The addition of noise was intended to replicate the conditions of real-world inspections, which often occur in noisy factory environments. The dataset covers six types of machinery, including four sampled from the MIMII Dataset (i.e., valve, pump, fan, and slide rail) and two from the ToyADMOS dataset (i.e., toy-car and toy-conveyor).

Preprocessing

We follow the standard preprocessing scheme used in the challenge baseline and many of challenge participants. The approach involves the use of Short Time Fourier Transform (STFT) to transform each audio clip into a spectrogram, which is then converted to Mel-scale. We set the number of Mel bands as 128,

the STFT window length as 1024, and the hop length (i.e., the number of audio samples between adjacent STFT columns) as 512. This configuration results in a spectrogram with 128 columns that represented the number of Mel bands. To construct the final spectrogram, the mel spectra of five consecutive frames are collected and combined to form a single row, resulting in a spectrogram with 640 columns. Each row of the spectrogram is sampled and used as input to the models under investigation, with a batch size of 512, meaning that 512 rows were randomly selected from the spectrogram at each iteration. We standardize all data along the feature dimension to zero mean and unit variance.

Performance Measure

We refer to the scoring method introduced in [72] and use the area under the receiver operating characteristic curve (AUROC) and partial-AUROC (pAUROC) as a quantitative measure of performance. pAUROC measures the AUC over the area corresponding to the false positive rate from 0 to a reasonably small value p, which we set in all our experiments as 0.1. Each measure is defined as follows:

$$\mathsf{AUROC} = \frac{1}{N_- N_+} \sum_{i=1}^{N_-} \sum_{i=j}^{N_+} \mathcal{H}(\mathcal{A}_\theta(x_j^+) - \mathcal{A}_\theta(x_i^-)) \tag{4.11}$$

$$p\mathsf{AUROC} = \frac{1}{\lfloor pN_- \rfloor N_+} \sum_{i=1}^{\lfloor pN_- \rfloor} \sum_{i=j}^{N_+} \mathcal{H}(\mathcal{A}_\theta(x_j^+) - \mathcal{A}_\theta(x_i^-)) \tag{4.12}$$

where $\{x_i^-\}_{i=1}^{N_-}$ and $\{x_j^+\}_{j=1}^{N_+}$ are normal and anomalous samples, respectively, $\lfloor \cdot \rfloor$ indicates the flooring function, and $\mathcal{H}(\cdot)$ represents a hard-threshold function whose output is 1 for a positive argument and 0 otherwise.

Implementation of Models

We utilize a standard autoencoder model provided by DCASE 2020 Challenge organizers and compare it to our approach. The model comprises a symmetrical arrangement of fully-connected layers in the encoder and decoder; 5 fully connected hidden layers in the input and 5 in the output, with 128-dimensional hidden layers and 32-dimensional latent space. In addition, we incorporate the IDNN model, which predicts the excluded frame using all frames except the central one instead of reconstructing the entire sequence. The IDNN model outperforms the autoencoder on non-stationary sound signals, i.e., sounds with short durations. The model consists of a encoder and decoder, which is similar to the components of an autoencoder but has an asymmetric layout; 3 fully-connected hidden layers that contract in dimension (64, 32, 16) are used in the encoder and 3 layers that expand in dimension (16, 32, 64) compose the decoder. The architectural design of each model follows the specifications outlined in their respective papers.

The MPDR models used for the experiment, MPDR-R and MPDR-IDNN, consists of an ensemble of autoencoders and an energy function, where the terms "R" and "IDNN" specify whether an autoencoder of IDNN is used to compute the energy, respectively. Each autoencoder consists of an encoder, spherical embedding layer, decoder; the encoder and decoder consist of 3 fully-connected layers each, with 1024-dimensional hidden space and a latent space with a dimension chosen among 32, 64, or 128. The autoencoder energy function used in the experiment is built using layers 5 fully-connected encoder layers and 5 decoder layers, whose hidden layers have 128 nodes and latent layer is composed of 32 nodes. The layers of IDNN are indentical to that of the autoencoder version, except for the input and output layers whose dimensions differ and

add up to the total number of sampled frames. LMC hyperparameters used are listed in Table 4.9.

4.7 Empirical Guidelines for Implementing MPDR

Here, we present empirical tips and observations we found useful in achieving competitive OOD detection performance with MPDR. The list also includes heuristics that *did not* work when we tried.

- The training progress can be monitored by measuring an OOD detection metric (i.e., AUROC) computed between test (or validation) inliers and synthetic samples uniformly sampled over the autoencoder manifold $\mathcal{M}$. During a successful and stable run, this score tends to increase smoothly.
- Metropolis-style adjustment for LMC [28] did not improve OOD performance.
- The choice of activation function in the energy function affects the OOD detection performance significantly. We found that ReLU and LeakyReLU provide good results in general.
- In image datasets, stopping the training of the autoencoder manifold before convergence improves OOD detection performance.
- A longer Markov chains, both visible and latent, do not always lead to better OOD detection performance.
- A larger autoencoder does not always lead to better OOD detection performance.
- A larger energy function does not always lead to better OOD detection performance.

Chapter 5

Generating Adversarial Outliers with Energy-Based Models

5.1 Introduction

Outlier detection, also called out-of-distribution (OOD) detection, is concerned with determining whether an input lies outside the training data distribution [85, 86]. An OOD detector is an essential component of a trustworthy machine learning system, since it can prevent erroneously overconfident predictions, enable the detection of distribution shift, and facilitate active or continual learning [87]. Due to its significance, there has been increasing amount of attention has been devoted to building a reliable OOD detector.

The performance of OOD detection has been improved rapidly in recent years. There are a number of OOD detectors achieving AUROC higher than 0.99 on the popular CIFAR-10 (in) vs SVHN (out) benchmark which was once known to be difficult. It is tempting to conclude that the OOD detectors are good enough for the given setting and move our attention to more challenging benchmarks, for example, large-scale problems with a greater diversity among inliers [71], or *near-OOD* problems where outliers look more similar to inliers [68].

However, these excellent OOD detection results are often fragile, and care must be taken before we declare a detector to be reliable solely based on a high

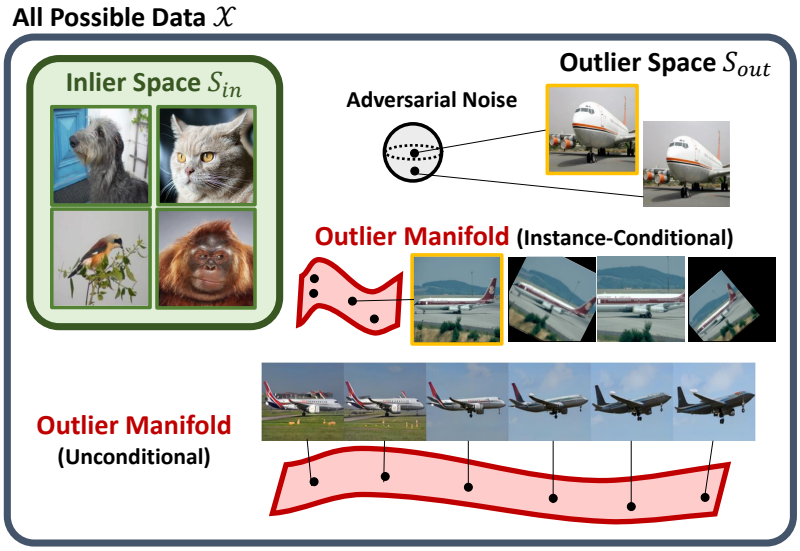

Figure 5.1 Illustration of outlier manifolds. Real images are highlighted with frames, and synthetic images are shown without frame. An instance-conditional outlier manifold is constructed from a real test outlier and spans the possible transformations of the sample. An unconditional outlier manifold is learned from multiple outliers. AGOM searches an outlier manifold for the "worst-case" outlier that fools a given OOD detector most strongly.

AUROC score. It is known that an OOD detector may misclassify an outlier as in-distribution when the outlier is corrupted (for example, blurred) [88] or perturbed with an adversarially generated noise [89]. Since the content of an outlier image will remain the same as OOD after these perturbations, these cases are obvious failure modes in OOD detection and are addressed in recent work [90, 91, 92, 93]. Meanwhile, these perturbation modes only represent a small fraction of possible perturbations that might occur to an outlier image.

In this paper, we question whether previously studied perturbations can validate the reliability of OOD detectors. In principle, the prediction from an OOD detector should be robust against any perturbation that preserves the outlierness of a datum. Although it may be infeasible to construct all such perturba-

tions, we believe that it is still important to investigate as many outlier-ness-preserving perturbations as possible to eventually obtain more reliable OOD detectors. The underlying premise behind our construction of the outlier manifold is that there exist outlier-ness-preserving visual perturbations described by a small number of continuous factors, motivated by the idea that variations in perceptual stimuli often form a continuous manifold [94]. Specifically, in our experiments, we construct outlier manifolds by using well-known data transformation operations, such as affine and color transforms, as well as a generative model, such as generative adversarial networks (GAN), that can learn the manifold of data. However, AGOM is not limited to a particular choice of outlier manifold and hence highly extensible.

The process of verifying an OOD detector's robustness with respect to an outlier manifold is formulated as an optimization problem, similar to adversarial attacks. On a given outlier manifold, AGOM searches for the outlier that is most confidently misclassified as in-distribution by the detector being tested. We address the optimization problem by employing an ensemble of gradient-free and gradient-based Markov Chain Monte Carlo (MCMC) methods. We find MCMC being highly effective in the optimization in AGOM, probably because the optimization problem in a low-dimensional manifold has multiple local optima and the ergodicity of MCMC helps to find a better optimum.

We apply AGOM on OOD detectors having AUROC scores near 1.0 on test OOD datasets and find that AGOM can successfully generate outlier images misclassified as in-distribution for all detectors. The OOD detectors are shown to be a different degree of vulnerability against AGOM, while the detector based on the vision transformer (ViT) [68] being the most robust one. Deeper analysis on the worst-case outliers found by AGOM reveals that many of OOD detectors have "blind spots", a certain visual feature in an outlier image that makes an

OOD detector mistake the outlier as an inlier.

AGOM addresses a different type of vulnerabilities in an OOD detector from adversarial noises, such as l_∞ attack. While an adversarial noise is indistinguishable and high-dimensional, perturbations generated from AGOM are low-dimensional and visually salient. Interestingly, improving robustness on one type of attack does not necessarily translate into the robustness to the other. Our experimental results indeed show that the OOD detectors trained with techniques for improving the robustness against adversarial noises exhibit limited robustness against AGOM (Table 1 and 2). On the other hand, ViT, possessing promising robustness against AGOM, is still susceptible to adversarial noises, according to [95]. We believe AGOM and the noise-like attacks are complementary to each other, and an ideal OOD detector should be robust to both types of attacks.

Our contributions can be summarized as follows:

- We propose Adversarial Generation of Outliers on Manifolds (AGOM), a novel generative attack to evaluate the robustness of OOD detection with respect to perceptually plausible variations of a given outlier dataset.
- We provide extensive benchmark on the existing OOD detectors using AGOM and show that the existing OOD detectors are vulnerability to AGOM.
- The results from AGOM reveal the previously unexplored failure mode of OOD detectors and provide insights on methods that make OOD detection more robust.

5.2 Related Work

OOD detection. The task of recognizing samples not from the training distribution has been referred to as outlier detection [85], novelty detection [96], one-class classification [21], and OOD detection [86]. We defer a comprehensive

review to a recent good survey [97]. While a large number of OOD detection methods have been proposed, there are several common high-level ideas. Finding out which approach yields the best robustness is one of our key interest. OOD detection is often performed based on the information in the output layer, e.g., the logits and the probabilities [86, 47, 98, 71]. Auxiliary OOD datasets can be used to calibrate OOD detectors [89, 91, 99, 88]. When a strong representation is available, Mahalanobis distance in the feature space can be a good OOD detector [100, 69, 68]. The disagreement among separately trained classifiers can also be used for OOD detection [101]. The effectiveness of self-supervised learning is investigated in [102, 103, 93]. Using synthetic outlier data during training can be beneficial for OOD detection. Synthetic outliers may be generated in the data space using GAN [104] or in a feature space in [105].

Robustness of OOD Detectors. The robustness of an OOD detector is measured by the amount of degradation in performance under perturbation on test outliers. Small-normed adversarial perturbations are the most extensively studied mode of perturbation. Most OOD detectors mistake an outlier as in-distribution when the outlier is perturbed with an adversarial noise, unless they are specifically trained to be robust to the attack. The perturbation is synthesized through an optimization method which may utilize the gradient of an OOD detector. The resulting perturbation is often constrained to have a very small norm, typically measured in l_∞ norm, so that the perturbed outlier looks visually the same to human eyes. Various techniques have been proposed to improve the robustness against adversarial perturbations, for example, adversarial training [89, 88], interval bound propagation [91], an informative outlier mining method combined with adversarial training [88], and a hybrid model with provable binary classifier [92]. Another class of perturbations used to benchmark the robustness in OOD detection is image corruption operations, such as JPEG

compression and Gaussian blur, proposed in [106]. When applied to outliers, the corruptions can significantly degrade OOD detection [88, 93]. Still, these corruptions are controlled by discrete parameters, unlike AGOM where visual perturbations are parametrized by continuous parameters.

Generative Models in Generating Perturbations. Recall that adversarially perturbed images are actually synthetic images. It is a natural extension to exploit more diverse synthetic data to evaluate the model's robustness. In [107, 108, 109, 110, 111], GAN is employed to generate adversarial examples to generate adversarial samples to fool a classifier. The paper show that a model robust to noise-like attack is not necessarily robust to other types of attack. The ideas of adversarially manipulating an image in a semantic code space [112] and finding adversarial image transformation [113, 114] are investigated in the context of adversarial attack towards supervised classifiers. However, these ideas has rarely been applied to OOD detection.

5.3 Robustness in OOD Detection

5.3.1 Out-of-Distribution Detection

OOD detection, or also called outlier detection, is a binary classification problem of discriminating in- and out-of-distribution samples. First of all, we represent a datum as a D-dimensional real-valued vector $\mathbf{x} \in \mathcal{X} \subset \mathbb{R}^D$, where $\mathcal{X}$ is the set of all possible values of a valid datum. We mainly consider image data and assume pixels have values between 0 and 1. Therefore, $\mathcal{X} = [0, 1]^D$.

A vector $\mathbf{x}$ is defined as an **in-distribution** sample or an inlier, when the vector is in the inlier space $\mathbf{x} \in \mathcal{S}_{in}$. Similarly, $\mathbf{x}$ is an **out-of-distribution (OOD)** sample or an outlier, when the vector belongs to the outlier space $\mathbf{x} \in \mathcal{S}_{out}$. The union of the inlier set and the outlier set is the whole data space $\mathcal{X} = \mathcal{S}_{in} \cup \mathcal{S}_{out}$. We assume no overlap between inliers and outliers $\mathcal{S}_{in} \cap \mathcal{S}_{out} = \varnothing$.

See Fig. 5.1 for the illustration. In OOD detection, we do not know the boundary between $\mathcal{S}_{in}$ and $\mathcal{S}_{out}$ and are only given a finite set of training in-distribution samples.

An **OOD detector** $f : \mathbb{R}^D \to \mathbb{R}$ is a function which outputs a larger value for an input more likely to be an outlier. A test sample $\mathbf{x}^*$ is classified as OOD if $f(\mathbf{x}^*) > \eta_f$ for some threshold η_f. The function value $f(\mathbf{x})$ will be referred to as an **outlier score** from the detector. We shall assume $f(\mathbf{x})$ is bounded. An OOD detector is trained from the in-distribution training set and does not have an access to test outliers. However, some OOD detectors, such as [47], use a dataset of auxiliary outliers during training. The detector $f(\mathbf{x})$ may or may not be differentiable. Some OOD detectors are not readily differentiable due to operations used in their forward pass. The performance of OOD detector is measured in metrics such as the area under the receiver operating characteristic curve (AUROC or AUC) or the false positive rate at the threshold of 95% of true positive rate (FPR95). Such metrics are computed from a pair of a test inlier dataset and a test OOD dataset $\mathcal{D}_{out} \subset \mathcal{S}_{out}$.

5.3.2 Robustness of OOD Detectors

The robustness of a machine learning model can be quantified by the worst-case performance under perturbations applied to input data. While the perturbation can be applied to both inliers and outliers, in this paper, we will only focus on perturbations on outliers for two reasons. First, the confident misclassification of an outlier is often more dangerous. For example, classifying a broken battery as normal in a manufacturing line may lead to an accidents such as explosion. Second, it is unclear whether an inlier sample can remain in-distribution after a perturbation, because we do not know the exact boundary of $\mathcal{S}_{in}$.

A threat model $\mathcal{T}$ is a set of possible perturbed data. When $\mathcal{T}$ is generated by

perturbing a single instance, then we call this threat model *instance-conditional*. A threat model can also be constructed from multiple instances, then we call the threat model *unconditional*. One of the most widely studied instance-conditional threat model is l_∞-ball around each datum: $\mathcal{T}_\infty(\mathbf{x}_{out}) = \{\mathbf{x} | \ \|\mathbf{x} - \mathbf{x}_{out}\|_\infty \leq \epsilon\}$, where $\|\cdot\|_\infty$ denotes l_∞ norm and $\epsilon > 0$ is a radius of the ball. In the evaluation of an OOD detector's robustness, we require and assume $\mathcal{T}$ to contain outliers only. We shall revisit this point in Sec. 5.4.1.

The robustness of an OOD detector f under a threat model $\mathcal{T}$ is evaluated by finding the worst-case outlier in $\mathcal{T}$ that has the lowest outlier score.

$$\mathbf{x}_{adv} = \arg\min_{\mathbf{x}} f(\mathbf{x}) \quad \text{such that} \quad \mathbf{x} \in \mathcal{T}. \tag{5.1}$$

We may call $\mathbf{x}_{adv}$ an adversarial sample. The robustness is measured by how low $f(\mathbf{x}_{adv})$ is compared to the outlier scores of the in-distribution test samples. If $\mathcal{T}$ is instance-conditional, it is common to collect multiple $\mathbf{x}_{adv}$ and compute a binary classification metric such as AUROC. A larger drop in AUROC score after a perturbation indicates more significant vulnerability.

Despite the flexibility in the choice of $\mathcal{T}$, very limited choices of $\mathcal{T}$ have been investigated in OOD detection so far. Our key contribution is investigation of the unexplored choices of $\mathcal{T}$ which can reflect more diverse and realistic visual variations that might occur to outliers.

5.4 Adversarial Generation of Outliers on Manifolds

Here, we introduce **Adversarial Generation of Outliers on Manifold**, a generative attack algorithm designed for OOD detectors. AGOM follows the formulation introduced in Eq. 5.1 and therefore is characterized by a threat model $\mathcal{T}$ and an optimization method to find an adversarial outlier $\mathbf{x}_{adv}$ within $\mathcal{T}$. The novelty of AGOM lies in the design of $\mathcal{T}$ particularly relevant to OOD

detection and the choice of an effective approach for optimization.

Given an OOD detector $f(\cdot)$ and a test OOD dataset $\mathcal{D}_{out}$, AGOM synthesizes an outlier $\mathbf{x}_{adv}$ that is misclassified as in-distribution with strong confidence by f, i.e., having low outlier scores. The adversarial outliers $\mathbf{x}_{adv}$ generated by AGOM reveal the unexplored weaknesses in highly competitive OOD detectors, demonstrating the significance of AGOM. Examples of $\mathbf{x}_{adv}$ will be provided in Sec. 5.5. As software, AGOM is an off-the-shelf tool to assess the robustness of an OOD detector and is applicable to a wide range of OOD detectors with significantly different designs.

5.4.1 Outlier Manifolds

We aim to build a threat model $\mathcal{T}$ that only contains OOD data while covers a significant degree of visual variations. To that end, we propose to enforce a threat model to have a low dimensionality. The low-dimensionality constraint is effective in suppressing a threat model from accidentally overlaps with the inlier space $\mathcal{S}_{in}$, while preserving the ability to represent plausible variations in visual features [94].

The **outlier manifold**, a threat model we present, is a set that is constructed from a low-dimensional latent factor $\mathbf{z} \in \mathbb{R}^{D'}$ $(D' < D)$ and a continuous and differentiable generator $g(\cdot)$:

$$\mathcal{T}_g = \{\mathbf{x} = g(\mathbf{z}) |\ \mathbf{z} \in \mathcal{Z}\}. \tag{5.2}$$

where the careful choice of $g(\mathbf{z})$ based on domain knowledge ensures $\mathcal{T}_g$ does not contain in-distribution data. We identify two classes of visual operations that ensure the generated sample is OOD and build $g(\mathbf{z})$ to represent such operations (Fig. 5.2): identity-preserving transforms and novel attribute combinations.

Identity-Preserving Transforms (IPT). IPT is an operation that alters the lower-level visual characteristics of an image but leaves the identity and the

high-level information unchanged. IPTs are widely used as data augmentation technique in classification [115], generative modeling [116], and self-supervised learning [84]. In IPT, the generator is conditioned on each outlier datum and therefore the resulting outlier manifold is *instance-conditional*:

$$\mathbf{x} = g(\mathbf{z}; \mathbf{x}_{out}), \quad \mathbf{x}_{out} \in \mathcal{D}_{out}. \tag{5.3}$$

We implement two IPT outlier manifolds, **Affine** and **Color**. Affine is a five-dimensional outlier manifold which spans images transformed by rotation, translation (x, y), scaling, and shear operation. Color outlier manifold contains images transformed in brightness, contrast, saturation and hue from $\mathbf{x}_{out}$. Even though visual variations that Affine and Color introduce might look trivial, our experiment shows that many of existing OOD detectors are not robust to these elementary transforms.

Novel Attribute Combination (NAC). Instead of perturbing a single outlier, we may leverage multiple outliers to build an outlier manifold. NAC uses OOD dataset $\mathcal{D}_{out}$ to construct a manifold of possible outliers by recombining visual features present in $\mathcal{D}_{out}$. Imagine a manifold of cat images generated by all possible combinations of attributes such as fur color (Fig. 5.2). Synthesizing a new example with a high fidelity is generally a difficult task, but recent developments in generative modeling provide strong tools, such as GANs [117]. We construct **GAN** outlier manifold, using StyleGAN2 generator [118] as $g(\mathbf{x})$. GAN outlier manifold is *unconditional*. We train $g(\mathbf{x})$ with Projected GAN technique [119] (See 5.5.1). Note that we define the outlier manifold in the "Z-space" of a stylegan generator, which resides before the mapping network, instead of "W-space" which are more frequently used in GAN inversion. Using Z-space significantly reduces the risk of accidentally generating an inlier.

GAN is one of multiple possible choices for constructing a NAC outlier

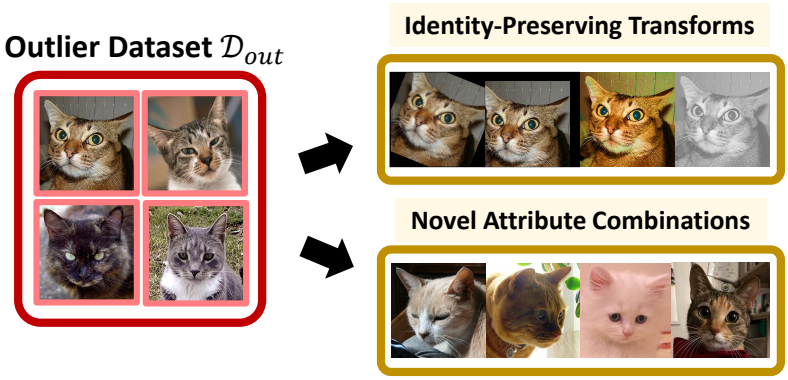

Figure 5.2 Two classes of outlier manifolds considered in AGOM.

manifold and is chosen from practical considerations over other options. An autoencoder [17] can be used in principle, but its generation quality is limited, and it may generate samples far from its training data [64]. Diffusion models [120] and flow-based models [49] are strong generative models, but they do not exploit the low-dimensional structure and are too slow in their generation, making optimization in Eq. 5.4 difficult. We also set the truncation parameter ψ less than 1 in StyleGAN2 to keep the generation faithful to the given dataset.

5.4.2 Adversarial Generation via MCMC Ensemble

With an outlier manifold $\mathcal{T}_g$, the robustness evaluation (Eq. 5.1) can now be conducted in the low-dimensional space of the latent codes $\mathcal{Z} \subset \mathbb{R}^{D'}$:

$$\mathbf{z}_{adv} = \arg\min_{\mathbf{z} \in \mathcal{Z}} f(g(\mathbf{z})); \quad \mathbf{x}_{adv} = g(\mathbf{z}_{adv}). \tag{5.4}$$

We find that a naive application of a gradient-based optimizer to Eq. 5.4 gives a limited success, due to the severe local optima problem.

We approach this optimization with an ensemble of three optimizers: random search [122], gradient-free MCMC (Metropolis-Hastings; MH; [123]), and gradient-based MCMC (Langevin Monte Carlo; LMC; [28]). Having a suite of

Metric	AUC								MinRank			
OOD	SVHN				CelebA				SVHN		CelebA	
Threat	Clean	Affine	Color	GAN	Clean	Affine	Color	GAN	Clean	GAN	Clean	GAN
Weak Detectors												
GLOW [49]	.069	.008	.000	.000	.542	.056	.001	.036	8	1	177	24
PIXELCNN [48]	.076	.002	.000	.000	.639	.062	.003	.109	0	0	391	369
AE [3]	.080	.011	.000	.000	.533	.055	.001	.036	0	0	18	75
Strong Detectors												
NAE [64]	.935	.755	.703	.104	.874	.706	.339	.092	11	11	616	48
GOOD [91]	.943	.565	.659	.474	.939	.685	.662	.551	2141	1249	3558	2741
ACET [89]	.966	.753	.868	.465	.986	.798	.904	.724	3414	1598	4845	4549
CEDA [89]	.979	.636	.913	.555	.981	.798	.906	.635	3348	0	3200	2936
SSD [103]	.989	.631	.936	.396	.780	.578	.676	.301	4132	223	2331	1669
MD [100]	.993	.747	.495	.159	.796	.294	.056	.082	69	3	5	3
SNGP [121]	.994	.761	.885	.395	.882	.404	.601	.056	2427	178	8	0
PROOD [92]	.995	.898	.964	.755	.996	.819	.977	**.942**	4683	791	6896	4535
ATOM [88]	.996	.908	.977	.661	.998	.911	.975	.848	6203	3002	7517	3634
OE [47]	.997	.751	.964	.742	.992	.766	.919	.565	5636	3049	5217	2248
ROWL [99]	.997	.675	.970	.469	.991	.778	.931	.513	5213	799	2454	1467
CSI [102]	.998	.940	**.992**	**.943**	.890	.778	.863	.519	**8208**	**7670**	3727	3225
ViT [68]	**1.000**	**.978**	.988	.816	**1.000**	**.963**	**.988**	.928	7218	5607	**8870**	**6596**

Table 5.1 CIFAR-10 experiment. Clean indicates the test split of a test OOD dataset. AUC scores are evaluated using 10,000 inliers and 1,000 outliers. MinRank is computed from a run which consists of 1,000 independent MCMC chains. The boldface are the largest numbers and the underlined are the smallest numbers among strong detectors.

optimizers makes the optimization robust across a wide variety of OOD detectors which are built based on distinct inductive biases and architectures.

Gradient-free and gradient-based MCMC is a key component in our optimizer suite. We employ MCMC because of its capability to traversing multiple local optima thanks to the ergodicity of a Markov chain. To apply MCMC, we define an auxiliary distribution constructed from f and $\mathcal{T}$, from which MCMC will generate samples:

$$p_f(\mathbf{z}) = \frac{1}{Z} \exp(-f(g(\mathbf{z}))/T), \tag{5.5}$$

where $T > 0$ is the temperature. Z is the normalization constant $Z = \int_{\mathcal{Z}} \exp(-f(g(\mathbf{z}))/T) d\mathbf{z}$. This formulation can be commonly found in simulated annealing techniques [124, 125]. For optimization, we take the sample $\mathbf{z}$ with the lowest $f(g(\mathbf{z}))$ value among the trajectory of MCMC.

We first run MH and then run LMC from the best point found from MH. In a black-box setting where the gradient information is unavailable, we only use the random search and MH. In fact, we find that MH is often very competitive and robust, especially in outlier manifolds in AGOM where the dimensionality is not too high. We defer the comparison of these three optimizers to Appendix.

The optimization problem in Eq. (5.4) can be approached in a number of other ways. Black-box global optimization methods, such as Bayesian optimization [126], are applicable, and there are other choices for MCMC samplers as well, for example, Hamiltonian Monte Carlo and Gibbs sampling. However, we aim to provide a simple and robust method that works.

5.5 Experiments

In our experiments, we evaluate the robustness of the state-of-the-art OOD detectors using AGOM. We deliberately chose OOD detection benchmarks where existing OOD detectors already achieved AUROC near 1.0 in order to test how robust the score is. AGOM is able to generate synthetic outliers that are misclassified by the detectors with high confidence.

5.5.1 Experimental Settings

Datasets. We use CIFAR-10 (32×32 RGB; [80]) and Restricted ImageNet (RImgNet; 224×224; [127]) as in-distribution datasets. CIFAR-10 is the most extensively studied dataset where a number of OOD detectors demonstrate AUROC near 1.0. RImgNet is a subset of ImageNet-1k that only contains animals (including insects). The original ImageNet labels are aggregated into 9 animal categories. RImgNet gives a relatively clear definition for in-distribution data, and therefore it is easier for a human to qualitatively inspect each cases in the benchmark.

For CIFAR-10 experiment, we use SVHN [81] and CelebA [83] as test OOD datasets. CIFAR-10 (in) vs SVHN (out) dataset pair becomes popular after the observation that deep generative models, such as PixelCNN++ or Glow, classifies SVHN examples are more likely to be CIFAR-10 than the actual CIFAR-10 samples [47, 18], i.e., producing AUC scores close to 0. Now, there are a number of OOD detectors that achieves very high OOD detection score in this benchmark (Table 5.1).

For RImgNet experiment, we use FGVC Aircraft [128], Oxford Flowers [129], and EuroSAT [130]. For Flowers dataset, we find that there can be some class overlap between RImgNet, since some flower images contain an insect or a bird. We manually inspect all test images and remove 219 images and will release this list. EuroSAT is a collection of satellite images from Sentinel-2 satellite. EuroSAT has a significantly different visual characteristic to FGVC and Flowers.

OOD Detectors. We reproduce or re-implement 16 OOD detectors for CIFAR-10 experiment and 4 OOD detectors for RImgNet. While implementing detectors, we try to diversify the idea which OOD detectors are based on. We also intentionally include weak OOD detectors as well. All OOD detectors are implemented in PyTorch. We use the pre-trained model from the publication whenever possible. Detailed information on OOD detectors and their implementation can be found in Appendix.

Evaluation Metrics. We compute AUC score when we measure how well an OOD detector separates inliers and outliers. When a perturbations is applied to a group of outliers, their AUC typically drops. The magnitude of the drop quantifies the vulnerability of an OOD detector to the perturbations.

Since we are interested in the worst-case performance of OOD detection, we also compute a new metric, **MinRank**. By rank, we refer the rank of an

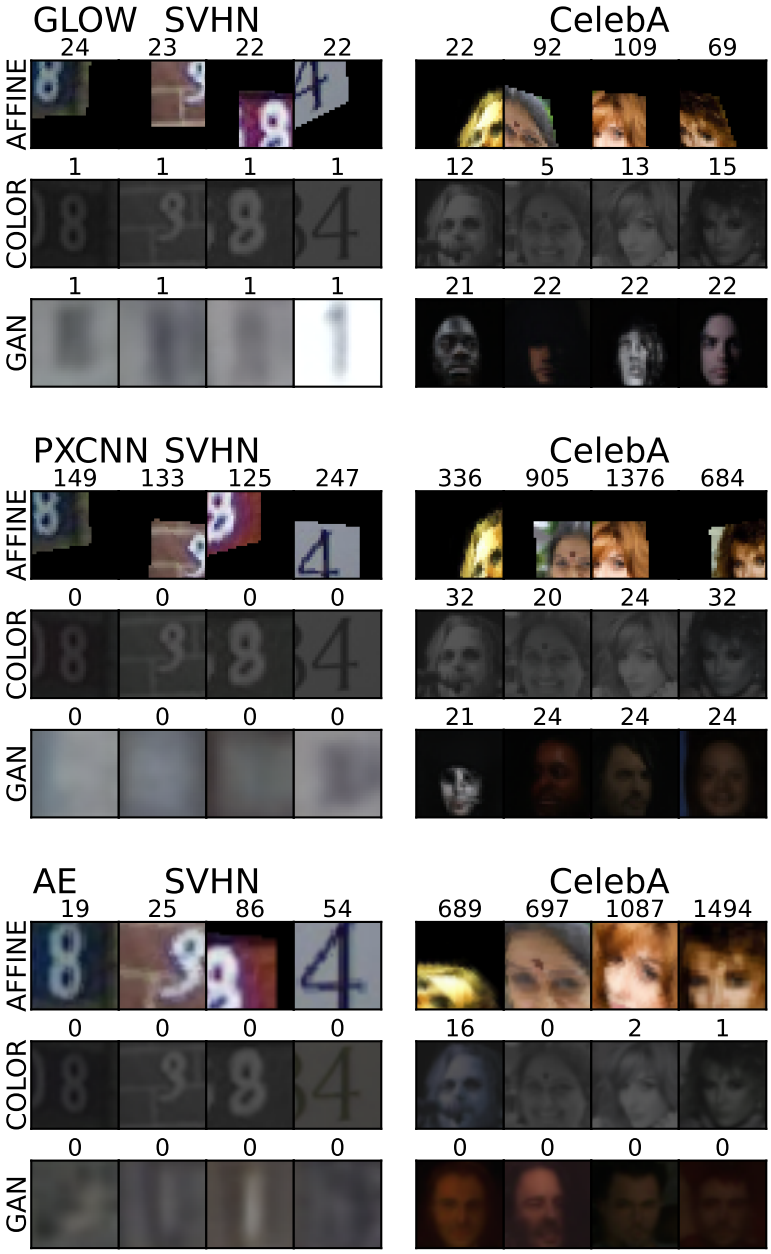

Figure 5.3 The images that Glow believes to be CIFAR-10, synthesize by AGOM. Glow trained on CIFAR-10 has an blind spot of misclassifying low-complexity images as CIFAR-10. Therefore, AGOM maximizes the size of the black area (Affine) or turns images into grayscale (Color). The numbers indicate the rank of outlier score among test inliers.

outlier score $f(\mathbf{x})$ among the outlier scores of the test in-distribution samples. The lowest rank is 0, which indicates that the detector believes the outlier is more likely to be in-distribution than any of test inliers. The number of test samples in CIFAR-10 is 10,000 and 10,150 for RImgNet, providing the upper bound of the rank. MinRank is the minimum rank among a group of outliers and it indicates how strong the detector is fooled in the worst case situation.

Metric	AUC									MinRank											
OOD	FGVC			Flowers			EuroSat			FGVC				Flowers				EuroSat			
Threat	Clean	Affine	Color	Clean	Affine	Color	Clean	Affine	Color	Clean	Affine	Color	GAN	Clean	Affine	Color	GAN	Clean	Affine	Color	GAN
MSP	.927	.697	.770	.918	.707	.747	.987	.859	.684	3390	1418	2001	1306	3367	1871	2825	1833	**8183**	7018	1213	4461
OE	.998	.948	.978	.968	.857	.889	.976	.936	.908	8650	5490	8813	8084	1256	825	1207	1384	7545	7907	6993	7763
PROOD	.998	.960	.974	.967	.860	.874	.975	.924	.877	9333	6490	7395	6707	4531	2742	1861	718	7836	8035	4897	7534
ViT	**1.000**	**.998**	**0.999**	**.998**	**.992**	**.986**	**.999**	**.993**	**.995**	**10075**	**9571**	**10059**	**9549**	**9715**	**9715**	**8328**	**7100**	8149	**8149**	**8149**	**8005**

Table 5.2 RImgNet Experiment. AGOM is applied to 4 OOD detectors. Other conditions are the same as Table 5.1.

Outlier Manifolds. We use Affine (5D), Color (4D), and GAN (16D and 64D) outlier manifolds. The first two are instance-conditional, and GAN is an unconditional outlier manifolds. Operations in Affine and Color are implemented to be differentiable by Kornia [131]. For GAN outlier manifolds, we train StyleGAN2 generator with Projected GAN discriminator on the test split of OOD datasets. Projected GAN enables data-efficient GAN training by utilizing representations from a pre-trained network. We are aware that using a pre-trained network may introduce a bias in FID score [132]. Nonetheless, we proceed with Projected GAN because it gives perceptually better samples and measuring FID score is not our goal. We train two GANs with two different latent dimensionalities, 16D and 64D. We ensemble the results by taking the most strong adversarial sample.

MCMC. To deal with the scale difference in scores $f(\mathbf{x})$, the scores are standardized to have the zero mean and the variance of one, when evaluated on test in-distribution data. For all experiments, the temperature is set $T = 1$.

MH algorithm with Gaussian proposal distribution, where the standard deviation of the proposal distribution is fixed to 0.1. A proposal is stochastically accepted based on Metropolis' criterion. For LMC, we set the step size as 0.05 and the standard deviation of the noise as 0.1. We find that the gradient clipping at 0.1 gives more stable results. Each MH chain runs for 2,000 steps, where each step corresponds to a single detector evaluation. Following a MH chain, LMC chain runs for 200 steps. We do not apply LMC on a black-box OOD detector which is MD in our experiment. For instance-conditional outlier manifolds, we use the first 1,000 examples of a test OOD dataset. Among the visited states in a trajectory, the sample with the smallest $f(\mathbf{x})$ is selected.

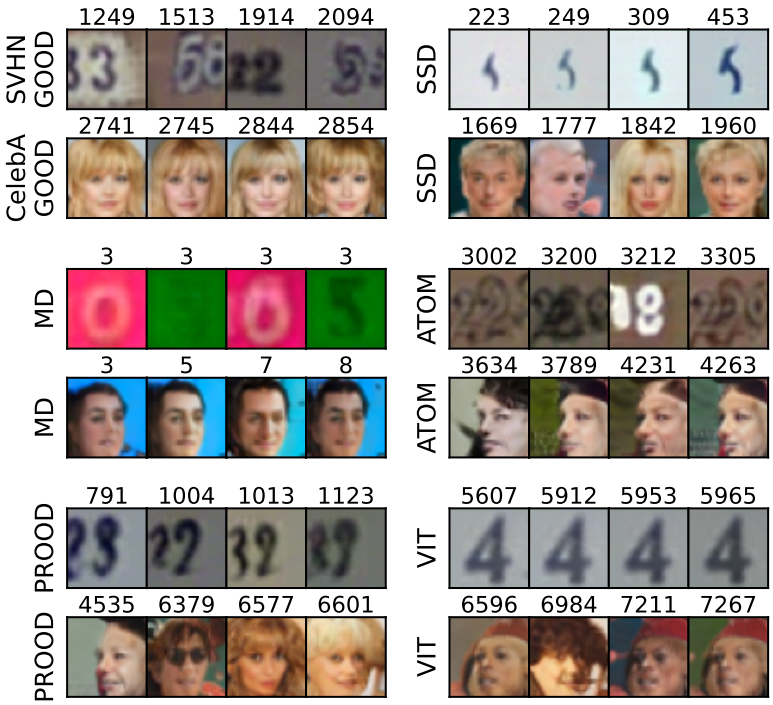

Figure 5.4 Adversarial samples from GAN outlier manifold. A subset of OOD detectors are shown due to the space contraint.

5.5.2 CIFAR-10 Experiment

AGOM with Affine, Color, and GAN outlier manifolds is applied to 16 OOD detectors. Table 5.1 provides the results. Overall, all OOD detectors exhibit drops in OOD detection performance upon the perturbation from AGOM. Also, AGOM is able to synthesizes pathological outliers which has a significantly low rank score. Overall, CSI, and ViT show better robustness than others although there exist several adversarial outliers found by AGOM that can fool CSI and ViT. We suspect that self-supervised learning and transformer architecture trained with a large body of data provide provide representation robust to low-level variation irrelevant to semantics.

Weak Detectors. We confirm the effectiveness of AGOM by applying them on the weak detectors, Glow, PixelCNN, and AE, where the weaknesses have already been analyzed. It is known that Glow, PXCNN, and AE erroneously classify low-complexity images, such as monotone images or highly blurry images, as in-distribution [50, 64]. The known failure mode can be clearly observed from Figure 5.3.

Effectiveness of Low-Dimensional Variation. Considering their low dimensionality, Affine and Color variation models are surprisingly effective at finding failure modes for some models. GOOD, ACET, and CEDA, the detectors trained to be robust against l_∞ threat model, show significant degree of vulnerability under Affine. This indicates that optimizing for the robustness against one threat model does not necessarily improve the robustness against other threat models. Meanwhile, CSI show strong robustness against Affine and Color, probably because transformations similar to Affine and Color are used during its training.

Characteristic Failure Modes. Adversarial samples found by AGOM

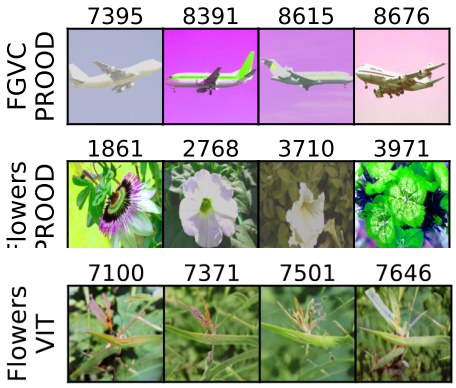

Figure 5.5 Adversarial samples from AGOM in RImgNet experiment. More examples can be found in Appendix. (Top two rows) Affine (Bottom) GAN. Even adversarial color distortion can fool an OOD detector that is designed to have improved robustness against l_∞ attack.

reveals blind spots in OOD detection. For example, there is a *color bias* in MD, as demonstrated by samples in Figure 5.4. MD classifies an image as in-distribution when there is vivid magenta, green or blue colors. GOOD and SSD classify blond women as CIFAR-10. Even though ViT shows a decent degree of robustness across outlier manifolds, it isclassifies digit "4" as in-distribution with confidence. In fact, we find that this example is indeed classified as airplane and its representation in ViT is very close to an in-distribution airplane image (details in Appendix). This example illustrates the importance of learning robust and accurate representation in reliable OOD detection.

5.5.3 RImgNet Experiment

We apply AGOM to 4 OOD detectors trained on RImgNet. The results are shown in Table 5.2. Maximum Softmax Probability (MSP; [86]) shows a significant degree of vulnerability to AGOM, despite its fair OOD detection AUC. OE and ProoD are particularly susceptible to Affine and Color outlier manifold, showing low MinRanks. Examples are shown in Fig. 5.5. It is interesting

OOD	FGVC		Flowers		EuroSAT	
Threat	Clean	l_∞	Clean	l_∞	Clean	l_∞
MSP	.926	.007	.918	.130	.986	.007
OE	.998	.015	.968	.007	.976	.000
Prood	.998	**.680**	.967	**.564**	.975	.429
ViT	**.999**	.067	**.998**	.312	**.999**	.037

Table 5.3 Robustness to l_∞ attack, measured in AUC.

to note that an OOD detector can be fooled by color jittering. Unlike l_∞ attack, AGOM can not turn any given image into an adversarial sample, as can be seen in relatively high AUC in Table 5.2. However, AGOM can search for unexpected failure cases which has low rank score.

ViT shows an excellent degree of overall robustness. However, AGOM reveals some of its failure modes. For example, MinRank of ViT for Flowers-Color is 8328, which is lower than the most commonly used 95% true-positive rate threshold. ViT is also fooled by images synthesized from GAN outlier manifold (Fig. 5.5).

For comparison, we also evaluate l_∞ robustness of OOD detectors following the protocol of [92]. The results are shown in Table 5.3. ProoD, a model with the improved l_∞ robustness has a limited AGOM-robustness, while ViT, having a good AGOM-robustness, is susceptible to l_∞ attack.

5.6 Discussion

Generative Attacks in Supervised Setting. The idea of using generative models [107, 108, 109, 110, 111] or image transformations [113, 114, 133] in adversarial attack are mainly investigated only in supervised learning setting, where a generative method modifies or generates adversarial in-distribution samples. However, it is problematic that a generation method can sometimes fail

and produce low-quality images that may not be considered as in-distribution. For example, suppose a human face is highly distorted by an adversarial transform so that the face is barely recognizable. It is questionable if a face recognition algorithm is supposed to classify such an example correctly, as the transformed sample can be considered OOD. Generation failure is a less critical problem when generative methods are applied to generate OOD samples, as in AGOM, because the distorted OOD images are still OOD that should be detected by OOD detectors.

Limitations First, we can only prove that an OOD detector has not achieved the robustness. Even though an OOD detector is shown to be robust to AGOM, the detector may be susceptible to other threat models or other OOD data. However, this limitation is fundamental to all empirical evaluation methods not just to AGOM. Second, AGOM relies heavily on human domain knowledge when constructing an outlier manifold. Application of AGOM to other OOD detection domains such as sound [134] may require an expert supervision. Third, building a GAN outlier manifold may require a non-trivial number of outlier data points which are not always available in practice. However, by leveraging recent techniques enabling GAN training with a small number of data, AGOM can operate even when only a few thousand data are given, as in our RImgNet experiment, where there are only 3334 samples in the FGVC test set.

Ethics Statement. Our main ethical concern is that a subset of OOD detectors used in our experiment, OE, CEDA, ACET, GOOD, ProoD are trained using 80 Million Tiny Images dataset [135], which is retracted by authors over ethical concerns. While we were aware of the issue of the dataset, the use of models trained on the dataset was inevitable because of the reproducibility. To minimize the effect of the retracted dataset, the dataset was never used directly. We only used the publicly available model checkpoints, and did not download

or access to a copy of dataset.

5.7 Conclusion

In this paper, we have addressed the limitations of the current robustness evaluation protocol and proposed a novel framework, Adversarial Generation of Outliers on Manifolds, which utilize a generative approach to investigate the failure modes of OOD detectors beyond noise-like perturbations or preset corruptions.

AGOM implies an interesting connection between generative modeling and the model evaluation process. Conventionally, models are only evaluated using real samples which are scarce. Since our ability to generate realistic synthetic data is improving rapidly [136, 137], we now have an access to the essentially infinite number of realistic synthetic data. As in AGOM, the synthetic data and its generator can help revealing the hidden weaknesses of machine learning models that were not previously not accessible due to the lack of real test data.

Chapter 6

Generative Gaussian Process: Gaussian Process as an Energy-Based Model

6.1 Introduction

Gaussian Process Regression (GP) is a non-parametric Bayesian regression algorithm that supports the analytic evaluation of predictive variance. The variance of the predictive distribution $p(y|\mathbf{x})$ is often considered a proxy for how uncertain the prediction is, making GP an attractive choice for applications that require uncertainty quantification, such as bandit optimization, active learning [138], black-box optimization [139], and robust control.

A careful observation will tell us that the predictive variance of GP approximately reflects the scarcity of data around the point of prediction. The predictive variance will be likely to have a large value when the point is far from the training data and will have a small value as the point comes closer to the training data. This behavior justifies the use of predictive variance as the measure of uncertainty, as it is intuitive to be more sure about our prediction near the training data. Meanwhile, how densely populated data are can also be measured in a different yet well-established technique of density estimation. However, to the best of our knowledge, the connection between GP's predictive variance and density estimation has not been investigated yet.

In this chapter, we reveal that computing the predictive variance of GP is indeed equivalent to a certain form of density estimation. From the investigation of GP's predictive variance expression, we show that the predictive variance can be seen as the likelihood of a Gaussian distribution defined on the feature space of GP. The predictive variance will be large if the likelihood of this Gaussian distribution is small. However, performing density estimation on the feature space, not the original input space, may cause problems in uncertainty quantification under the current practice of GP. The input data density can be arbitrarily distorted in the feature space. A low-input-density point may have a large density when mapped into the feature space. In such case, GP produces a narrow predictive variance and becomes overconfident about its prediction on an obviously unexplored input point. This problem can be exacerbated under the current training algorithm of GP, where model is only optimized for predictive marginal likelihood $p(y|\mathbf{x})$ and ignores the input data density $p(\mathbf{x})$.

To address this need, we propose Generative GP (GenGP), which treats GP as an energy-based generative model trained to model the joint distribution of data. GenGP offers significantly better epistemic uncertainty estimates while maintaining the predictive performance of GP, while effectively remedies the overconfidence problem.

Our contributions are summarized as follows:

1. We analyze Gaussian Process Regression and reveal that computing the predictive variance can be seen as a form of density estimation.

2. We show that the predictive variance can be a misleading indicator of epistemic uncertainty under the current discriminative training scheme of GP.

3. We propose Generative Gaussian Process Regression (GenGP), which of-

fers improved epistemic uncertainty estimation and competitive performance in applications requiring accurate uncertainty quantification.

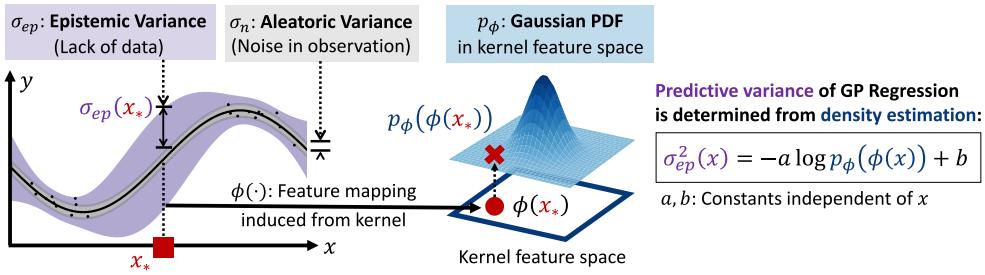

Figure 6.1 The connection between the predictive variance of GPR and density estimation. The predictive variance of GPR is decomposed into epistemic and aleatoric uncertainties. The epistemic variance σ^2_{ep} has a linear relationship with the log-density of a Gaussian distribution p_ϕ which estimates the density of regression inputs in the feature space.

6.2 Gaussian Processes Are Density Estimators

Here, we analyze the predictive variance of GP and show its connection to the estimation of input data density $p(\mathbf{x})$. The connection is revealed by viewing the predictive variance from the feature space of GP. Computing the predictive variance of GP is equivalent to computing the negative log-likelihood, i.e., the energy, of a Gaussian distribution in GP's feature space.

Gaussian Process Regression We first review GP. As a stochastic process, GP is completely defined by specifying a mean function $m(\mathbf{x})$ and a positive semi-definite covariance kernel $k(\mathbf{x}, \mathbf{x}')$. Given the training dataset $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^N$ where $\mathbf{x}_i \in \mathbb{R}^D$ and $y_i \in \mathbb{R}$ and a test point $\mathbf{x}$, the predictive distribution $p(y|\mathbf{x}, \mathcal{D})$ of GP is a Gaussian $\mathcal{N}(\mu(\mathbf{x}), \sigma^2(\mathbf{x}))$ where its mean $\mu(\mathbf{x})$

and variance $\sigma^2(\mathbf{x})$ are written as follows:

$$\mu(\mathbf{x}) = m(\mathbf{x}) + \mathbf{k}(\mathbf{x})^\top (K + \sigma_n^2 I)^{-1}(\mathbf{y} - \mathbf{m}), \tag{6.1}$$

$$\sigma^2(\mathbf{x}) = \sigma_n^2 + \sigma_{ep}^2(\mathbf{x}) \quad \text{and} \quad \sigma_{ep}^2(\mathbf{x}) = k(\mathbf{x}, \mathbf{x}) - \mathbf{k}(\mathbf{x})^\top (K + \sigma_n^2 I)^{-1}\mathbf{k}(\mathbf{x}), \tag{6.2}$$

where $\mathbf{k}(\mathbf{x}) = [k(\mathbf{x}, \mathbf{x}_1), \ldots, k(\mathbf{x}, \mathbf{x}_N)]^\top \in \mathbb{R}^N$ is a kernel vector and $K \in \mathbb{R}^{N \times N}$ is a pairwise kernel matrix with $(K)_{i,j} = k(\mathbf{x}_i, \mathbf{x}_j)$ for $i, j = 1, \ldots, N$. Also, $\mathbf{y} = [y_1, \ldots, y_N]^\top \in \mathbb{R}^N$ is the concatenated targets and $\mathbf{m} = [m(x_1), \ldots, m(x_N)]^\top \in \mathbb{R}^N$ is the concatenated means. The hyperparameter σ_n^2 represents the observation noise which is often referred to as aleatoric uncertainty.

Since σ_n^2 is independent of $\mathbf{x}$, i.e., homoscedastic, the epistemic uncertainty $\sigma_{ep}^2(\mathbf{x})$ is actually responsible for determining the predictive uncertainty. We can observe that σ_{ep}^2 behaves as if it encodes the information of probability of $p(\mathbf{x})$. First, $\sigma_{ep}^2(\mathbf{x})$ is (negatively) correlated with how densely populated $\mathbf{x}_i$'s are. Second, $\sigma_{ep}^2(\mathbf{x})$ only encodes the information about x_i's and is independent of y_i's. We will see that computing $\sigma_{ep}^2(\mathbf{x})$ is indeed related to density estimation.

Feature Space View We now view $\sigma_{ep}^2(\mathbf{x})$ (Eq. 6.2) from the feature space associated with GP. It is well known that a positive semi-definite kernel can be expressed in an inner product of a feature mapping $\phi : \mathbb{R}^D \to \mathbb{R}^{D'}$, and we can write $k(\mathbf{x}, \mathbf{x}') = \phi(\mathbf{x})^\top \phi(\mathbf{x}')$. Therefore,

$$k(\mathbf{x}, \mathbf{x}) = \phi(\mathbf{x})^\top \phi(\mathbf{x}), \quad \mathbf{k}(\mathbf{x}) = \Phi^\top \phi(\mathbf{x}), \quad \mathbf{K} = \Phi^\top \Phi \tag{6.3}$$

where $\Phi = [\phi(\mathbf{x}_1), \cdots, \phi(\mathbf{x}_N)] \in \mathbb{R}^{D' \times N}$. Plugging these expressions into $\sigma_{ep}^2(\mathbf{x})$ (Eq. (6.2)) and applying Woodbury matrix identity, we can express $\sigma_{ep}^2(\mathbf{x})$ with respect to feature vectors.

$$\sigma_{ep}^2(\mathbf{x}) = \phi(\mathbf{x})^\top \left(\frac{1}{\sigma_n^2} \Phi \Phi^\top + I \right)^{-1} \phi(\mathbf{x}) = \frac{\sigma_n^2}{N} \phi(\mathbf{x})^\top \Sigma_\phi^{-1} \phi(\mathbf{x}) \tag{6.4}$$

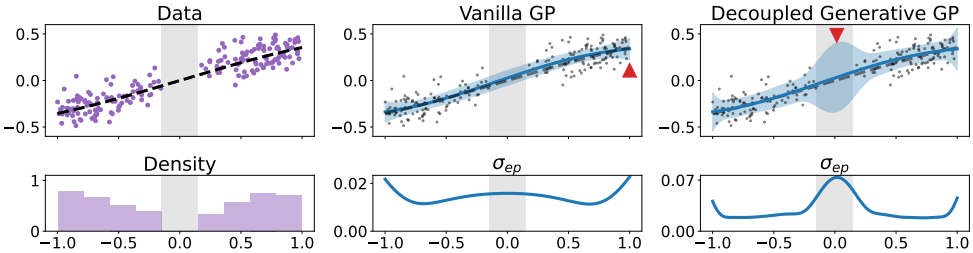

Figure 6.2 1D regression example where epistemic uncertainty quantification of vanilla GPR fails.

where we set $\Sigma_\phi = \frac{1}{N}\Phi\Phi^\top + \frac{\sigma_n^2}{N}I$. Note that Σ_ϕ can be seen as an empirical covariance of $\phi(\mathbf{x}_i)$'s with the regularization of σ_n^2/N. While the first equality in Eq. (6.4) is well-known [140], we provide detailed derivation in the appendix to be self-contained.

A Gaussian in Feature Space Now we present a novel interpretation of Eq. (6.4). Consider a Gaussian parametric density model $\mathcal{N}(\phi(\mathbf{x}); \mathbf{0}, \Sigma)$ defined on ϕ-space. Its mean is fixed as zero and only Σ is estimated from $\phi(\mathbf{x}_i)$'s. If we apply the diagonal regularization $(\sigma_n^2/N)I$, then we obtain Σ_ϕ as an estimate for Σ.

Then, the log-likelihood is given as $\log \mathcal{N}(\phi(\mathbf{x}); \mathbf{0}, \Sigma_\phi) = -\frac{1}{2}\phi(\mathbf{x})^\top \Sigma_\phi^{-1}\phi(\mathbf{x}) - \frac{1}{2}\log 2\pi - \frac{1}{2}\log|\Sigma_\phi|$. Rearranging with Eq. (6.4), we obtain the following expression:

$$\boxed{\sigma_{ep}^2(\mathbf{x}) = -\frac{2\sigma_n^2}{N}\left[\log \mathcal{N}(\phi(\mathbf{x}); \mathbf{0}, \Sigma_\phi) + C\right]} \tag{6.5}$$

Therefore, computing $\sigma_{ep}^2(\mathbf{x})$ in GP is equivalent to evaluating the energy, i.e., the negative log-likelihood of a zero-mean Gaussian distribution in the feature space. This relationship is graphically illustrated in Fig. 6.1.

6.3 Density Estimators in Gaussian Processes May Mislead

As we show in Proposition 1, GP uses the density estimator in the feature space $p_\phi(\phi(\mathbf{x}))$ to measure the epistemic uncertainty at $\mathbf{x}$. However, the epistemic uncertainty provided by $p_\phi(\phi(\mathbf{x}))$ can be significantly misleading, particularly under the current practices of GP.

Condition for good epistemic uncertainty indicator As an epistemic uncertainty indicator, $p_\phi(\phi(\mathbf{x}))$ should be able to identify the points with low $p_{data}(\mathbf{x})$ where few training data are around and thus epistemic uncertainty is high. To identify such points, we need to infer the relative ordering of $p_{data}(\mathbf{x})$ among $\mathbf{x}$'s using $p_\phi(\phi(\mathbf{x}))$. If the inference on the relative ordering is possible, we shall say the two densities are aligned. Two density functions $p_\phi(\phi)$ and $p_{data}(\mathbf{x})$ are **aligned** when for any $\mathbf{x}_1$ and $\mathbf{x}_2$ in the support of $p_{data}(\mathbf{x})$, with $\phi_1 = \phi(\mathbf{x}_1)$ and $\phi_2 = \phi(\mathbf{x}_2)$, the following is satisfied:

$$p_\phi(\phi_1) \leq p_\phi(\phi_2) \quad \text{iff} \quad p_{data}(\mathbf{x}_1) \leq p_{data}(\mathbf{x}_2). \tag{6.6}$$

However, depending on the choice of ϕ, $p_\phi(\phi(\mathbf{x}))$ and $p_{data}(\mathbf{x})$ may not be aligned. If the feature mapping ϕ is not injective, i.e., maps two $\mathbf{x}$'s into the same ϕ, the information loss occurs and the ordering of $p_{data}(\mathbf{x})$ can not be recovered from $p_\phi(\phi(\mathbf{x}))$. This phenomenon is often called the feature collapse. Even for an invertible ϕ which preserves information, $p_{data}(\mathbf{x})$ can be distorted arbitrarily in the feature space.

A misaligned $p_\phi(\phi(\mathbf{x}))$ will mislead epistemic uncertainty quantification. Suppose there are two points $\mathbf{x}_{out}$ and $\mathbf{x}_{in}$ where $p_{data}(\mathbf{x}_{out}) < p_{data}(\mathbf{x}_{in})$. A successful epistemic uncertainty indicator should able to tell the epistemic uncertainty of $\mathbf{x}_{out}$ is higher. However, the misalignment makes $p_\phi(\phi(\mathbf{x}_1)) < p_\phi(\phi(\mathbf{x}_2))$, and hence $\sigma^2_{ep}(\mathbf{x}_1) > \sigma^2_{ep}(\mathbf{x}_2)$. For example, if such misaligned GP is

applied in an active learning scenario where unlabeled $\mathbf{x}$ with high epistemic uncertainty is queried for labeling will result in querying data with no additional information.

Misalignment in GP The risk of misalignment is significant in most GP practices where kernel hyperparameters are optimized. Since a kernel implicitly determines the feature space, the hyperparameter optimization actually optimizes the feature space.

The most common objective function for the hyperparameter optimization is the marginal likelihood of regression $p(\mathcal{D}) = p(\mathbf{y}|\mathbf{x}_1, \ldots, \mathbf{x}_N)$. Nonetheless, a feature space optimal for regression is not necessarily optimal for density estimation and density alignment.

6.4 Generative Gaussian Process Regression

Two aspects of GP contribute to the failure of epistemic uncertainty quantification. First, the training, i.e., hyperparameter optimization, ignores density estimation. Second, the same hyperparameters are used for both regression and density estimation. Here, we address these points by proposing Generative Gaussian Process Regression (GenGP), where we introduce the generative training and decoupling density estimation from regression. GenGP achieves the same predictive performance with conventional GP by design, while providing significantly better epistemic uncertainty estimate.

GP as A Generative Model GP is conventionally viewed as a discriminative model which only models $p(y|\mathbf{x})$. Since the predictive variance of GP contains density information, we propose to treat GP as a generative model which models $p(\mathbf{x}, y)$. The feature space density estimator $p_\phi(\phi(\mathbf{x}))$ is not a valid probability density in the input space. Hence, we re-normalize $p_\phi(\phi(\mathbf{x}))$

in the input space to form a probabilistic model $p(\mathbf{x})$ in the input space.

$$p_{GP}(\mathbf{x}) = \frac{1}{Z}\exp(-E(\mathbf{x})/T), \quad E(\mathbf{x}) = \sigma^2_{ep}(\mathbf{x}), \quad Z = \int \exp(-E(\mathbf{x}))d\mathbf{x} \quad (6.7)$$

We propose to train GP in a generative manner by modeling the joint distribution $p(\mathbf{x}, y)$. From the factorization $\log p(\mathbf{x}, y) = \log p(y|\mathbf{x}) + \log p(\mathbf{x})$, maximizing the joint log-likelihood leads to the introduction of an additional term $\log p(\mathbf{x})$ representing the likelihood of inputs.

6.5 Bayesian Regressors Are Approximately Density Estimators

We can generalize the connection between the predictive variance and density estimation to a general Bayesian regression algorithms. Consider a Bayesian regression algorithm $f(\mathbf{x}, \mathbf{w})$ which models the conditional mean of y given $\mathbf{x}$ using a likelihood $p(y|\mathbf{x}, \mathbf{w}) = \mathcal{N}(f(\mathbf{x}, \mathbf{w}), \sigma_n^2)$. We show that the variance of the conditional mean is approximately a form of a density estimator.

Consider the first-order Taylor expansion of $f(\mathbf{x}, \mathbf{w})$ around the posterior mean of the parameter $\overline{\mathbf{w}} = \mathbb{E}_{p(\mathbf{w}|\mathcal{D})}[\mathbf{w}]$:

$$f(\mathbf{x}, \mathbf{w}) = f(\mathbf{x}, \overline{\mathbf{w}}) + (\mathbf{w} - \overline{\mathbf{w}})^\top \nabla_{\mathbf{w}} f(\mathbf{x}, \overline{\mathbf{w}}) + o(|\mathbf{w} - \overline{\mathbf{w}}|) \quad (6.8)$$

where $(\nabla_{\mathbf{w}} f(\mathbf{x}, \overline{\mathbf{w}}))_i = \frac{\partial}{\partial w_i} f(\mathbf{x}, \overline{\mathbf{w}})|_{\mathbf{w}=\overline{\mathbf{w}}}$. Then, the variance of the prediction $f(\mathbf{x}, \mathbf{w})$ with respect to the posterior $p(\mathbf{w}|\mathcal{D})$ is given as:

$$\begin{aligned}
\mathrm{Var}_{p(\mathbf{w}|\mathcal{D})}[f(\mathbf{x}, \mathbf{w})] &= \mathrm{Var}_{p(\mathbf{w}|\mathcal{D})}[(\mathbf{w} - \overline{\mathbf{w}})^\top \nabla_{\mathbf{w}} f(\mathbf{w}, \overline{\mathbf{w}})] && (6.9)\\
&= \nabla_{\mathbf{w}} f(\mathbf{x}, \overline{\mathbf{w}})^\top \mathbb{E}_{p(\mathbf{w}|\mathcal{D})}\left[(\mathbf{w} - \overline{\mathbf{w}})(\mathbf{w} - \overline{\mathbf{w}})^\top\right] \nabla_{\mathbf{w}} f(\mathbf{x}, \overline{\mathbf{w}}) && (6.10)\\
&= \nabla_{\mathbf{w}} f(\mathbf{x}, \overline{\mathbf{w}})^\top \mathrm{Cov}_{p(\mathbf{w}|\mathcal{D})}[\mathbf{w}] \nabla_{\mathbf{w}} f(\mathbf{x}, \overline{\mathbf{w}}) && (6.11)
\end{aligned}$$

Now, we assume that the number of data is large enough so that we can leverage the asymptotic normality (sometimes called Bernstein-von Mises theorem)

of the posterior $p(\mathbf{w}|\mathcal{D})$. The posterior can be approximated as a Gaussian distribution where the posterior mean $\overline{\mathbf{w}}$ is the mode of the distribution.

$$p(\mathbf{w}|\mathcal{D}) \approx \mathcal{N}(\overline{\mathbf{w}}, A^{-1}), \quad A = -\nabla\nabla_{\mathbf{w}} \log p(\overline{\mathbf{w}}|\mathcal{D}) \tag{6.12}$$

The covariance of $\mathbf{w}$ is A^{-1}.

$$A = \nabla\nabla_{\mathbf{w}} \log p(\overline{\mathbf{w}}|\mathcal{D}) \tag{6.13}$$

$$= \nabla\nabla_{\mathbf{w}} \log p(\overline{\mathbf{w}}) + \sum_{i=1}^{N} \nabla\nabla_{\mathbf{w}} \log p(y_i|\mathbf{x}_i, \overline{\mathbf{w}}) \tag{6.14}$$

$$\approx \nabla\nabla_{\mathbf{w}} \log p(\overline{\mathbf{w}}) + N \cdot \mathbb{E}_{\mathbf{x},y} \left[\nabla\nabla_{\mathbf{w}} \log p(y|\mathbf{x}, \overline{\mathbf{w}})\right] \tag{6.15}$$

where we apply the law of large numbers in the last line. The

$$\nabla_{\mathbf{w}} \log p(y|\mathbf{x}, \overline{\mathbf{w}}) = -\frac{1}{\sigma_n^2}(y - f(\mathbf{x}, \mathbf{w}))\nabla_{\mathbf{w}} f(\mathbf{x}, \mathbf{w}) \tag{6.16}$$

$$\nabla\nabla_{\mathbf{w}} \log p(y|\mathbf{x}, \overline{\mathbf{w}}) = \frac{1}{\sigma_n^2}\nabla_{\mathbf{w}} f(\mathbf{x}, \mathbf{w})\nabla_{\mathbf{w}} f(\mathbf{x}, \mathbf{w})^\top - \frac{1}{\sigma_n^2}(y - f(\mathbf{x}, \mathbf{w}))\nabla\nabla_{\mathbf{w}} f(\mathbf{x}, \mathbf{w}) \tag{6.17}$$

$$\mathbb{E}_{\mathbf{x},y}[\nabla\nabla_{\mathbf{w}} \log p(y|\mathbf{x}, \overline{\mathbf{w}})] = \frac{1}{\sigma_n^2}\mathbb{E}_{\mathbf{x}}\left[\nabla_{\mathbf{w}} f(\mathbf{x}, \mathbf{w})\nabla_{\mathbf{w}} f(\mathbf{x}, \mathbf{w})^\top\right] \tag{6.18}$$

$$- \frac{1}{\sigma_n^2}\mathbb{E}_{\mathbf{x}}\left[(\mathbb{E}_y[y|\mathbf{x}] - f(\mathbf{x}, \overline{\mathbf{w}}))\nabla\nabla_{\mathbf{w}} f(\mathbf{x}, \mathbf{w})\right] \tag{6.19}$$

$$\approx \frac{1}{\sigma_n^2}\mathbb{E}_{\mathbf{x}}\left[\nabla_{\mathbf{w}} f(\mathbf{x}, \mathbf{w})\nabla_{\mathbf{w}} f(\mathbf{x}, \mathbf{w})^\top\right] \tag{6.20}$$

where we assume that the prediction error of the regression algorithm is very small $\mathbb{E}_y[y|\mathbf{x}] - f(\mathbf{x}, \mathbf{w}) \approx 0$.

$$A = \nabla\nabla_{\mathbf{w}} \log p(\overline{\mathbf{w}}) + \frac{N}{\sigma_n^2}\mathbb{E}_{\mathbf{x}}\left[\nabla_{\mathbf{w}} f(\mathbf{x}, \mathbf{w})\nabla_{\mathbf{w}} f(\mathbf{x}, \mathbf{w})^\top\right] \tag{6.21}$$

If we write $\psi(\mathbf{x}) = \nabla_{\mathbf{w}} f(\mathbf{x}, \overline{\mathbf{w}})$, then

$$\mathrm{Var}_{p(\mathbf{w}|\mathcal{D})}[f(\mathbf{x}, \mathbf{w})] = \frac{\sigma_n^2}{N}\psi(\mathbf{x})^\top \Sigma_\psi^{-1} \psi(\mathbf{x}) \tag{6.22}$$

Bayesian prediction algorithms implicitly estimate the joint distribution of data $p(\mathbf{x}, y)$ when there is a large number of data.

6.6 Experiment

6.6.1 Oversmoothed Predictive Variance

We demonstrate that GP may produce predictive variance that does not faithfully reflect epistemic uncertainty. When the function being predicted has a relatively long length scale, i.e., the function varies slowly, the predictive variance of GP can be low even though the point being predicted has a high epistemic uncertainty. Fig. 6.2 shows an 1D example. A vanilla GP produces high predictive variance near the boundary of the domain, where a large number of training data already exist. This unexpectedly high predictive variance, if utilized in applications in active learning or Bayesian Optimization, will produce misleading exploration and suboptimal performance. Meanwhile, Decoupled Generative GP gives correctly identified epistemic uncertainty estimate.

A 2D example is provided in Fig. 6.3. Similarly to 1D case, a vanilla GP optimized for predictive marginal likelihood gives predictive variance that is too small at the center of the domain where no data is present. In other words, GP is highly over-confident even though it have not observed anything in this region. This over-confidence is remedied in Decoupled GenGP by optimizing the model parameters with the generative objective function. Note that the kernel length scale determined through optimization is 0.21, which is significantly smaller than the length scale (3.0) of the vanilla GP.

6.7 Active Learning

Active learning is one of the popular applications of GP. We show that the generative training for GP can benefit active learning performance. We test vanilla GP and Decoupled GenGP on 2D binary classification problem. The results are shown in Fig. 6.4. Each experiment is initialized with 5 labelled samples and

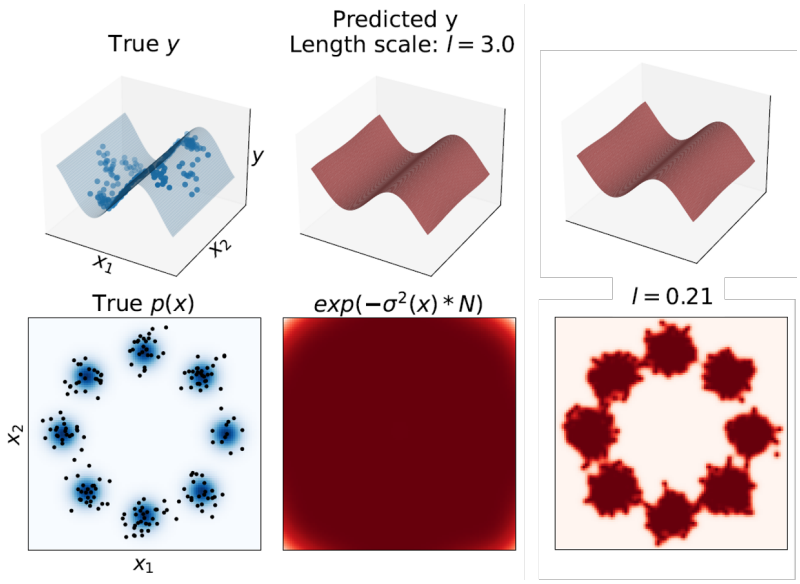

Figure 6.3 Generative GP in 2D. (First column) The true data distribution and the true function to be predicted. Data points are visualized as dots. (Second column) Prediction from a vanilla GP trained to maximize the marginal likelihood. (Third column) Prediction from a decoupled GenGP.

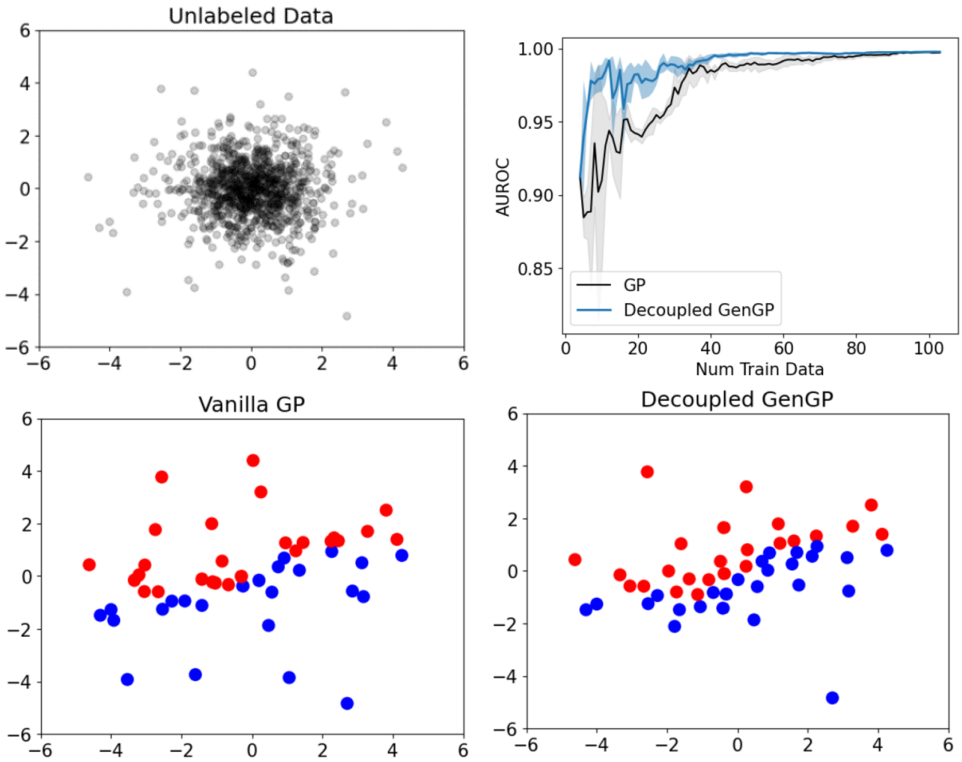

Figure 6.4 Active learning experiment. (Upper left) The distribution of unlabelled data. (Upper right) Active learning result averaged over 100 runs. (Lower left) Samples collected by vanilla GP. (Lower right) Samples collected by Decoupled GenGP.

995 unlabelled samples. In each active learning iteration, the most informative sample is selected and appended to training dataset. The hyper parameters of GP are updated every iteration. AUROC score of Decoupled GenGP grows faster than GP. The collected samples from GenGP, when visualized, located more evenly near the decision boundary, indicating that the informativeness of each sample is maximized.

Chapter 7

Conclusion

7.1 Summary and Key Takeaways

This dissertation demonstrates the significance of generative modeling for effective uncertainty quantification by analyzing and enhancing widely-used algorithms such as autoencoders and Gaussian processes. The study identifies critical failure modes in these algorithms and proposes improvements by presenting a novel perspective on them as generative models within the energy-based model framework. The resulting algorithms, normalized autoencoders and generative Gaussian processes, effectively address the issues of outlier reconstruction and over-smoothed variance. Furthermore, the thesis contributes to anomaly detection by introducing a novel training algorithm for energy-based models, Manifold Projection-Diffusion Recovery (MPDR), and investigating the robustness of anomaly detection algorithms through the Adversarial Generation on Manifold (AGOM) method.

Three general insights are derived from this work. First, good generative modeling enhances uncertainty quantification. While not all generative models lead to improved uncertainty quantification, as evidenced by previous failures of deep generative models, properly executed generative modeling can serve as a guiding principle for building and optimizing superior uncertainty quantification algorithms.

Second, well-designed energy functions improve generative modeling. The flexibility in designing energy functions is a strength of energy-based models that has not yet been fully utilized. By exploring various energy function designs in different settings, we demonstrate the benefits of well-crafted energy functions.

Lastly, Bayesian predictive uncertainty may be connected to generative modeling. Our work reveals a previously undiscovered connection between generative modeling and Bayesian predictive uncertainty quantification, which is often taken for granted without deeper justification.

7.2 Future Directions

Our work, situated at the intersection of energy-based generative modeling and uncertainty quantification, offers multiple avenues for exciting future research. Here, we highlight a few possibilities.

First, developing improved training algorithms for energy-based models is a promising direction. Despite the existence of several proposed training algorithms, energy-based model training remains sensitive and unstable. Many training algorithms involve non-convergent Langevin Monte Carlo, which is difficult to tune. A possible theoretical approach to enhancing energy-based model training is to leverage connections to optimal control or reinforcement learning.

Second, the concept of using generative modeling for uncertainty quantification can be expanded and applied to trustworthy machine learning. This idea has already proven effective in out-of-distribution detection in our experiments with normalized autoencoders and MPDR.

Lastly, advancements in uncertainty quantification methods can be employed in robotics applications. Accurate uncertainty assessment is essential

in various robotics tasks, where either collecting large quantities of real-world experience is challenging or the risk associated with poor actions is considerable. Potential applications include active perception and uncertainty-aware planning and control.

Bibliography

[1] Ruiqi Gao, Yang Song, Ben Poole, Ying Nian Wu, and Diederik P Kingma. Learning energy-based models by diffusion recovery likelihood. In *International Conference on Learning Representations*, 2021.

[2] Ryan Turner, Jane Hung, Eric Frank, Yunus Saatchi, and Jason Yosinski. Metropolis-Hastings generative adversarial networks. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 6345–6353. PMLR, 09–15 Jun 2019.

[3] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. *Learning Internal Representations by Error Propagation*, page 318–362. MIT Press, Cambridge, MA, USA, 1986.

[4] Nathalie Japkowicz, Catherine Myers, Mark Gluck, et al. A novelty detection approach to classification. In *Proceedings of the International Joint Conference on Artificial Intelligence*, volume 1, pages 518–523, 1995.

[5] Yiru Zhao, Bing Deng, Chen Shen, Yao Liu, Hongtao Lu, and Xian-Sheng Hua. Spatio-temporal autoencoder for video anomaly detection. In *Proceedings of the 25th ACM international conference on Multimedia*, pages 1933–1941, 2017.

[6] Yuchen Lu and Peng Xu. Anomaly detection for skin disease images using variational autoencoder. *arXiv preprint arXiv:1807.01349*, 2018.

[7] Olga Lyudchik. Outlier detection using autoencoders. Technical report, 2016.

[8] Alexander Tong, Roozbah Yousefzadeh, Guy Wolf, and Smita Krishnaswamy. Fixing bias in reconstruction-based anomaly detection with lipschitz discriminators. *arXiv preprint arXiv:1905.10710*, 2019.

[9] Bo Zong, Qi Song, Martin Renqiang Min, Wei Cheng, Cristian Lumezanu, Daeki Cho, and Haifeng Chen. Deep autoencoding gaussian mixture model for unsupervised anomaly detection. In *International Conference on Learning Representations*, 2018.

[10] Dong Gong, Lingqiao Liu, Vuong Le, Budhaditya Saha, Moussa Reda Mansour, Svetha Venkatesh, and Anton van den Hengel. Memorizing normality to detect anomaly: Memory-augmented deep autoencoder for unsupervised anomaly detection. In *IEEE International Conference on Computer Vision (ICCV)*, 2019.

[11] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*, pages 1096–1103, 2008.

[12] Salah Rifai, Pascal Vincent, Xavier Muller, Xavier Glorot, and Yoshua Bengio. Contractive auto-encoders: explicit invariance during feature extraction. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, pages 833–840, 2011.

[13] Andrew Ng et al. Sparse autoencoder. *CS294A Lecture notes*, 2011.

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

[15] Geoffrey E Hinton. Training products of experts by minimizing contrastive divergence. *Neural computation*, 14(8):1771–1800, 2002.

[16] Tijmen Tieleman. Training restricted boltzmann machines using approximations to the likelihood gradient. In *Proceedings of the 25th international conference on Machine learning*, pages 1064–1071, 2008.

[17] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[18] Eric Nalisnick, Akihiro Matsukawa, Yee Whye Teh, Dilan Gorur, and Balaji Lakshminarayanan. Do deep generative models know what they don't know? In *International Conference on Learning Representations*, 2019.

[19] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

[20] Paul Bergmann, Sindy Löwe, Michael Fauser, David Sattlegger, and Carsten Steger. Improving unsupervised defect segmentation by applying structural similarity to autoencoders. *arXiv preprint arXiv:1807.02011*, 2018.

[21] Ingo Steinwart, Don Hush, and Clint Scovel. A classification framework for anomaly detection. *Journal of Machine Learning Research*, 6(Feb):211–232, 2005.

[22] Laurent Younes. On the convergence of markovian stochastic algorithms with rapidly decreasing ergodicity rates. *Stochastics: An International Journal of Probability and Stochastic Processes*, 65(3-4):177–228, 1999.

[23] Giorgio Parisi. Correlation functions and computer simulations. *Nuclear Physics B*, 180(3):378–384, 1981.

[24] Ulf Grenander and Michael I Miller. Representations of knowledge in complex systems. *Journal of the Royal Statistical Society: Series B (Methodological)*, 56(4):549–581, 1994.

[25] Yilun Du and Igor Mordatch. Implicit generation and modeling with energy based models. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d Alche-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 3608–3618. Curran Associates, Inc., 2019.

[26] Will Grathwohl, Kuan-Chieh Wang, Joern-Henrik Jacobsen, David Duvenaud, Mohammad Norouzi, and Kevin Swersky. Your classifier is secretly an energy based model and you should treat it like one. In *International Conference on Learning Representations*, 2020.

[27] Erik Nijkamp, Mitch Hill, Song-Chun Zhu, and Ying Nian Wu. Learning non-convergent non-persistent short-run mcmc toward energy-based model. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32, pages 5232–5242. Curran Associates, Inc., 2019.

[28] Gareth O Roberts, Richard L Tweedie, et al. Exponential convergence of langevin distributions and their discrete approximations. *Bernoulli*, 2(4):341–363, 1996.

[29] Max Welling and Yee W Teh. Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 681–688, 2011.

[30] Yang Song and Diederik P Kingma. How to train your energy-based models. *arXiv preprint arXiv:2101.03288*, 2021.

[31] Hervé Bourlard and Yves Kamp. Auto-association by multilayer perceptrons and singular value decomposition. *Biological cybernetics*, 59(4-5):291–294, 1988.

[32] Daniel Keysers, Nathanael Schärli, Nathan Scales, Hylke Buisman, Daniel Furrer, Sergii Kashubin, Nikola Momchev, Danila Sinopalnikov, Lukasz Stafiniak, Tibor Tihon, et al. Measuring compositional generalization: A comprehensive method on realistic data. *arXiv preprint arXiv:1912.09713*, 2019.

[33] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[34] Junbo Zhao, Michael Mathieu, and Yann LeCun. Energy-based generative adversarial network. *arXiv preprint arXiv:1609.03126*, 2016.

[35] Pascal Vincent. A connection between score matching and denoising autoencoders. *Neural computation*, 23(7):1661–1674, 2011.

[36] Michael Gutmann and Aapo Hyvärinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 297–304, 2010.

[37] Guillaume Alain and Yoshua Bengio. What regularized auto-encoders learn from the data-generating distribution. *The Journal of Machine Learning Research*, 15(1):3563–3593, 2014.

[38] Stanislav Pidhorskyi, Ranya Almohsen, and Gianfranco Doretto. Generative probabilistic novelty detection with adversarial autoencoders. In *Advances in neural information processing systems*, pages 6822–6833, 2018.

[39] Johann Brehmer and Kyle Cranmer. Flows for simultaneous manifold learning and density estimation. *arXiv preprint arXiv:2003.13913*, 2020.

[40] Charlie Nash and Conor Durkan. Autoregressive energy machines. In *International Conference on Machine Learning*, pages 1735–1744. PMLR, 2019.

[41] Chenlin Meng, Lantao Yu, Yang Song, Jiaming Song, and Stefano Ermon. Autoregressive score matching. *arXiv preprint arXiv:2010.12810*, 2020.

[42] Partha Ghosh, Mehdi S. M. Sajjadi, Antonio Vergari, Michael Black, and Bernhard Scholkopf. From variational to deterministic autoencoders. In *International Conference on Learning Representations*, 2020.

[43] Tim R Davidson, Luca Falorsi, Nicola De Cao, Thomas Kipf, and Jakub M Tomczak. Hyperspherical variational auto-encoders. In *34th Conference on Uncertainty in Artificial Intelligence 2018, UAI 2018*, pages 856–865. Association For Uncertainty in Artificial Intelligence (AUAI), 2018.

[44] Jiacheng Xu and Greg Durrett. Spherical latent spaces for stable variational autoencoders. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4503–4513, 2018.

[45] Deli Zhao, Jiapeng Zhu, and Bo Zhang. Latent variables on spheres for autoencoders in high dimensions. *arXiv*, pages arXiv–1912, 2019.

[46] Jie Ren, Peter J Liu, Emily Fertig, Jasper Snoek, Ryan Poplin, Mark Depristo, Joshua Dillon, and Balaji Lakshminarayanan. Likelihood ratios for out-of-distribution detection. In *Advances in Neural Information Processing Systems*, pages 14680–14691, 2019.

[47] Dan Hendrycks, Mantas Mazeika, and Thomas Dietterich. Deep anomaly detection with outlier exposure. In *International Conference on Learning Representations*, 2019.

[48] Tim Salimans, Andrej Karpathy, Xi Chen, and Diederik P Kingma. Pixelcnn++: Improving the pixelcnn with discretized logistic mixture likelihood and other modifications. *arXiv preprint arXiv:1701.05517*, 2017.

[49] Durk P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. In *Advances in Neural Information Processing Systems*, pages 10215–10224, 2018.

[50] Joan Serrà, David Álvarez, Vicenç Gómez, Olga Slizovskaia, José F. Núñez, and Jordi Luque. Input complexity and out-of-distribution detection with likelihood-based generative models. In *International Conference on Learning Representations*, 2020.

[51] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *arXiv preprint arXiv:1706.08500*, 2017.

[52] Yang Song and Stefano Ermon. Improved techniques for training score-based generative models. *arXiv preprint arXiv:2006.09011*, 2020.

[53] Paul Bergmann, Michael Fauser, David Sattlegger, and Carsten Steger. Mvtec ad – a comprehensive real-world dataset for unsupervised anomaly detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[54] Yuma Koizumi, Shoichiro Saito, Hisashi Uematsu, Noboru Harada, and Keisuke Imoto. Toyadmos: A dataset of miniature-machine operating sounds for anomalous sound detection. In *2019 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pages 313–317, 2019.

[55] Barry M Dillon, Luigi Favaro, Tilman Plehn, Peter Sorrenson, and Michael Krämer. A normalized autoencoder for lhc triggers. *arXiv preprint arXiv:2206.14225*, 2022.

[56] Yilun Du, Shuang Li, B. Joshua Tenenbaum, and Igor Mordatch. Improved contrastive divergence training of energy based models. In *Proceedings of the 38th International Conference on Machine Learning (ICML-21)*, 2021.

[57] Zhisheng Xiao, Karsten Kreis, Jan Kautz, and Arash Vahdat. {VAEBM}: A symbiosis between variational autoencoders and energy-based models. In *International Conference on Learning Representations*, 2021.

[58] Ruiqi Gao, Erik Nijkamp, Diederik P Kingma, Zhen Xu, Andrew M Dai, and Ying Nian Wu. Flow contrastive estimation of energy-based models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7518–7528, 2020.

[59] Tian Han, Erik Nijkamp, Linqi Zhou, Bo Pang, Song-Chun Zhu, and Ying Nian Wu. Joint training of variational auto-encoder and latent energy-based model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[60] Bo Pang, Tian Han, Erik Nijkamp, Song-Chun Zhu, and Ying Nian Wu. Learning latent space energy-based prior model. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 21994–22008. Curran Associates, Inc., 2020.

[61] Arash Vahdat and Jan Kautz. NVAE: A deep hierarchical variational autoencoder. In *Neural Information Processing Systems (NeurIPS)*, 2020.

[62] Yoshua Bengio, Li Yao, Guillaume Alain, and Pascal Vincent. Generalized denoising auto-encoders as generative models. In C.J. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc., 2013.

[63] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 6840–6851. Curran Associates, Inc., 2020.

[64] Sangwoong Yoon, Yung-Kyun Noh, and Frank Park. Autoencoding under normalization constraints. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 12087–12097. PMLR, 18–24 Jul 2021.

[65] Rithesh Kumar, Sherjil Ozair, Anirudh Goyal, Aaron Courville, and Yoshua Bengio. Maximum entropy generators for energy-based models. *arXiv preprint arXiv:1901.08508*, 2019.

[66] Houssam Zenati, Chuan Sheng Foo, Bruno Lecouat, Gaurav Manek, and Vijay Ramaseshan Chandrasekhar. Efficient gan-based anomaly detection. *arXiv preprint arXiv:1802.06222*, 2018.

[67] Hankook Lee, Jongheon Jeong, Sejun Park, and Jinwoo Shin. Guiding energy-based models via contrastive latent variables. In *International Conference on Learning Representations*, 2023.

[68] Stanislav Fort, Jie Ren, and Balaji Lakshminarayanan. Exploring the limits of out-of-distribution detection. *Advances in Neural Information Processing Systems*, 34:7068–7081, 2021.

[69] Jie Ren, Stanislav Fort, Jeremiah Liu, Abhijit Guha Roy, Shreyas Padhy, and Balaji Lakshminarayanan. A simple fix to mahalanobis distance for improving near-ood detection. *arXiv preprint arXiv:2106.09022*, 2021.

[70] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021.

[71] Haoqi Wang, Zhizhong Li, Litong Feng, and Wayne Zhang. Vim: Out-of-distribution with virtual-logit matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4921–4930, June 2022.

[72] Yuma Koizumi, Yohei Kawaguchi, Keisuke Imoto, Toshiki Nakamura, Yuki Nikaido, Ryo Tanabe, Harsh Purohit, Kaori Suefusa, Takashi Endo, Masahiro Yasuda, and Noboru Harada. Description and discussion on dcase2020 challenge task2: Unsupervised anomalous sound detection for machine condition monitoring. *arXiv preprint arXiv:2006.05822*, 2020.

[73] Ritwik Giri, Srikanth V. Tenneti, Karim Helwani, Fangzhou Cheng, Umut Isik, and Arvindh Krishnaswamy. Unsupervised anomalous sound detection using self-supervised classification and group masked autoencoder for density estimation. Technical report, DCASE2020 Challenge, July 2020.

[74] Pawel Daniluk, Marcin Gozdziewski, Slawomir Kapka, and Michal Kosmider. Ensemble of auto-encoder based systems for anomaly detection. Technical report, DCASE2020 Challenge, July 2020.

[75] Kaori Suefusa, Tomoya Nishida, Harsh Purohit, Ryo Tanabe, Takashi Endo, and Yohei Kawaguchi. Anomalous sound detection based on interpolation deep neural network. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 271–275, 2020.

[76] Tarin Clanuwat, Mikel Bober-Irizar, Asanobu Kitamoto, Alex Lamb, Kazuaki Yamamoto, and David Ha. Deep learning for classical japanese literature. *arXiv preprint arXiv:1812.01718*, 2018.

[77] Gregory Cohen, Saeed Afshar, Jonathan Tapson, and Andre Van Schaik. Emnist: Extending mnist to handwritten letters. In *2017 international joint conference on neural networks (IJCNN)*, pages 2921–2926. IEEE, 2017.

[78] Brenden M Lake, Ruslan Salakhutdinov, and Joshua B Tenenbaum. The omniglot challenge: a 3-year progress report. *Current Opinion in Behavioral Sciences*, 29:97–104, 2019.

[79] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

[80] A Krizhevsky. Learning multiple layers of features from tiny images. *Master's thesis, University of Tront*, 2009.

[81] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.

[82] M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, , and A. Vedaldi. Describing textures in the wild. In *Proceedings of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[83] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

[84] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

[85] Douglas M Hawkins. *Identification of outliers*, volume 11. Springer, 1980.

[86] Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. *arXiv preprint arXiv:1610.02136*, 2016.

[87] Dustin Tran, Jeremiah Liu, Michael W Dusenberry, Du Phan, Mark Collier, Jie Ren, Kehang Han, Zi Wang, Zelda Mariet, Huiyi Hu, et al. Plex: Towards reliability using pretrained large model extensions. *arXiv preprint arXiv:2207.07411*, 2022.

[88] Jiefeng Chen, Yixuan Li, Xi Wu, Yingyu Liang, and Somesh Jha. Atom: Robustifying out-of-distribution detection using outlier mining. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 430–445. Springer, 2021.

[89] Matthias Hein, Maksym Andriushchenko, and Julian Bitterwolf. Why relu networks yield high-confidence predictions far away from the training data and how to mitigate the problem. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[90] Alexander Meinke and Matthias Hein. Towards neural networks that provably know when they don't know. In *International Conference on Learning Representations*, 2020.

[91] Julian Bitterwolf, Alexander Meinke, and Matthias Hein. Certifiably adversarially robust detection of out-of-distribution data. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 16085–16095. Curran Associates, Inc., 2020.

[92] Alexander Meinke, Julian Bitterwolf, and Matthias Hein. Provably robust detection of out-of-distribution data (almost) for free. *arXiv preprint arXiv:2106.04260*, 2021.

[93] Umar Khalid, Ashkan Esmaeili, Nazmul Karim, and Nazanin Rahnavard. Rodd: A self-supervised approach for robust out-of-distribution detection. *arXiv preprint arXiv:2204.02553*, 2022.

[94] H Sebastian Seung and Daniel D Lee. The manifold ways of perception. *science*, 290(5500):2268–2269, 2000.

[95] Stanislav Fort. Adversarial vulnerability of powerful near out-of-distribution detection. *arXiv preprint arXiv:2201.07012*, 2022.

[96] Markos Markou and Sameer Singh. Novelty detection: a review—part 1: statistical approaches. *Signal processing*, 83(12):2481–2497, 2003.

[97] Jingkang Yang, Kaiyang Zhou, Yixuan Li, and Ziwei Liu. Generalized out-of-distribution detection: A survey. *arXiv preprint arXiv:2110.11334*, 2021.

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

[99] Vikash Sehwag, Arjun Nitin Bhagoji, Liwei Song, Chawin Sitawarin, Daniel Cullina, Mung Chiang, and Prateek Mittal. Analyzing the robustness of open-world machine learning. In *Proceedings of the 12th ACM Workshop on Artificial Intelligence and Security*, AISec'19, page 105–116, New York, NY, USA, 2019. Association for Computing Machinery.

[100] KIMIN Lee, Kibok Lee, Honglak Lee, and Jinwoo Shin. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. In *32nd Conference on Neural Information Processing Systems (NIPS)*. Neural Information Processing Systems Foundation, 2018.

[101] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in neural information processing systems*, 30, 2017.

[102] Jihoon Tack, Sangwoo Mo, Jongheon Jeong, and Jinwoo Shin. Csi: Novelty detection via contrastive learning on distributionally shifted instances. *arXiv preprint arXiv:2007.08176*, 2020.

[103] Vikash Sehwag, Mung Chiang, and Prateek Mittal. {SSD}: A unified framework for self-supervised outlier detection. In *International Conference on Learning Representations*, 2021.

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

[105] Xuefeng Du, Zhaoning Wang, Mu Cai, and Sharon Li. Towards unknown-aware learning with virtual outlier synthesis. In *International Conference on Learning Representations*, 2022.

[106] Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *arXiv preprint arXiv:1903.12261*, 2019.

[107] Yang Song, Rui Shu, Nate Kushman, and Stefano Ermon. Constructing unrestricted adversarial examples with generative models. *Advances in Neural Information Processing Systems*, 31:8312–8323, 2018.

[108] Omid Poursaeed, Tianxing Jiang, Yordanos Goshu, Harry Yang, Serge Belongie, and Ser-Nam Lim. Fine-grained synthesis of unrestricted adversarial examples. *arXiv preprint arXiv:1911.09058*, 2019.

[109] Jamie Hayes and George Danezis. Learning universal adversarial perturbations with generative models. In *2018 IEEE Security and Privacy Workshops (SPW)*, pages 43–49. IEEE, 2018.

[110] Yanghao Zhang, Wenjie Ruan, Fu Wang, and Xiaowei Huang. Generalizing universal adversarial attacks beyond additive perturbations. In *2020 IEEE International Conference on Data Mining (ICDM)*, pages 1412–1417. IEEE, 2020.

[111] Yuzhen Ding, Nupur Thakur, and Baoxin Li. Advfoolgen: Creating persistent troubles for deep classifiers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*, pages 142–151, October 2021.

[112] Ameya Joshi, Amitangshu Mukherjee, Soumik Sarkar, and Chinmay Hegde. Semantic adversarial attacks: Parametric transformations that fool deep classifiers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4773–4783, 2019.

[113] Jiyu Chen, David Wang, and Hao Chen. Explore the transformation space for adversarial images. In *Proceedings of the Tenth ACM Conference on Data and Application Security and Privacy*, pages 109–120, 2020.

[114] Bo Yang, Kaiyong Xu, Hengjun Wang, and Hengwei Zhang. Random transformation of image brightness for adversarial attack. *Journal of Intelligent & Fuzzy Systems*, (Preprint):1–12, 2021.

[115] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C.J. Burges, L. Bottou, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012.

[116] Tero Karras, Miika Aittala, Janne Hellsten, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Training generative adversarial networks with limited data. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 12104–12114. Curran Associates, Inc., 2020.

[117] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014.

[118] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[119] Axel Sauer, Kashyap Chitta, Jens Müller, and Andreas Geiger. Projected gans converge faster. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 17480–17492. Curran Associates, Inc., 2021.

[120] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 8780–8794. Curran Associates, Inc., 2021.

[121] 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. *Advances in Neural Information Processing Systems*, 33:7498–7512, 2020.

[122] Leonard Andreevich Rastrigin. About convergence of random search method in extremal control of multi-parameter systems. *Avtomat. i Telemekh*, 24(11):1467–1473, 1963.

[123] Nicholas Metropolis, Arianna W Rosenbluth, Marshall N Rosenbluth, Augusta H Teller, and Edward Teller. Equation of state calculations by fast computing machines. *The journal of chemical physics*, 21(6):1087–1092, 1953.

[124] Martin Pincus. Letter to the editor—a monte carlo method for the approximate solution of certain types of constrained optimization problems. *Operations research*, 18(6):1225–1228, 1970.

[125] Scott Kirkpatrick, C Daniel Gelatt, and Mario P Vecchi. Optimization by simulated annealing. *science*, 220(4598):671–680, 1983.

[126] Jasper Snoek, Hugo Larochelle, and Ryan P Adams. Practical bayesian optimization of machine learning algorithms. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012.

[127] Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. Robustness may be at odds with accuracy. *arXiv preprint arXiv:1805.12152*, 2018.

[128] S. Maji, J. Kannala, E. Rahtu, M. Blaschko, and A. Vedaldi. Fine-grained visual classification of aircraft. Technical report, 2013.

[129] Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *2008 Sixth Indian Conference on Computer Vision, Graphics & Image Processing*, pages 722–729. IEEE, 2008.

[130] Patrick Helber, Benjamin Bischke, Andreas Dengel, and Damian Borth. Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 2019.

[131] Edgar Riba, Dmytro Mishkin, Daniel Ponsa, Ethan Rublee, and Gary Bradski. Kornia: an open source differentiable computer vision library for pytorch. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 3674–3683, 2020.

[132] Tuomas Kynkäänniemi, Tero Karras, Miika Aittala, Timo Aila, and Jaakko Lehtinen. The role of imagenet classes in fr\'echet inception distance. *arXiv preprint arXiv:2203.06026*, 2022.

[133] Shasha Li, Abhishek Aich, Shitong Zhu, Salman Asif, Chengyu Song, Amit Roy-Chowdhury, and Srikanth Krishnamurthy. Adversarial attacks on black box video classifiers: Leveraging the power of geometric transformations. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 2085–2096. Curran Associates, Inc., 2021.

[134] Yuma Koizumi, Shoichiro Saito, Hisashi Uematsu, Noboru Harada, and Keisuke Imoto. Toyadmos: A dataset of miniature-machine operating sounds for anomalous sound detection. In *2019 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pages 313–317. IEEE, 2019.

[135] Antonio Torralba, Rob Fergus, and William T Freeman. 80 million tiny images: A large data set for nonparametric object and scene recognition. *IEEE transactions on pattern analysis and machine intelligence*, 30(11):1958–1970, 2008.

[136] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc., 2020.

[137] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 2022.

[138] Sambu Seo, Marko Wallat, Thore Graepel, and Klaus Obermayer. Gaussian process regression: Active data selection and test point rejection. In *Mustererkennung 2000*, pages 27–34. Springer, 2000.

[139] Bobak Shahriari, Kevin Swersky, Ziyu Wang, Ryan P. Adams, and Nando de Freitas. Taking the human out of the loop: A review of bayesian optimization. *Proceedings of the IEEE*, 104(1):148–175, 2016.

[140] Christopher K Williams and Carl Edward Rasmussen. *Gaussian processes for machine learning*. MIT press Cambridge, MA, 2006.

국문초록

확률은 불확실성을 측정하는 수단이다. 특히 기계 학습에서의 입력 데이터의 확률은 인식론적 불확실성을 측정하는 유용한 척도이다. 입력 데이터의 확률을 정확하게 추정할 수 있다면, 불확실성을 정량화해야하는 다양한 종류의 응용문제에 유용하게 활용될 수 있다. 그러나 딥러닝을 이용하여 입력 데이터 확률을 추정하는 딥 생성 모형은, 그럴듯한 데이터를 생성하는 데에 있어서는 인상적인 성능을 보였지만 불확실성 정량화 능력이 제한적이라는 것이 알려져 있다. 딥 생성 모형의 실패는 데이터 확률을 학습하는 것이 불확실성을 포착하는 유효한 접근 방법인지에 대한 회의론을 불러일으켰다.

이 논문에서는 데이터의 확률 분포를 추정하는 것이 불확실성 정량화에 중요한 접근 방법임을 논한다. 우리는 이상 검출과 불확실성 하에서의 의사결정 문제에서 널리 사용되는 알고리즘인 오토인코더와 가우시안 프로세스에 초점을 맞추어, 생성 모형의 접근 방법을 도입하면 이들의 성능을 향상시킬 수 있다는 것을 보인다. 오토인코더와 가우시안 프로세스 모두 데이터의 확률 분포를 명시적으로 학습하지 않는다. 따라서 데이터 확률 분포를 정확히 반영하지 못함으로 인해 심각한 실패를 일으킬 수 있다. 우리는 이러한 실패 사례를 보고하고, 이를 해결하기 위해 오토인코더와 가우시안 프로세스를 위한 새로운 생성 모형 기반 형식화를 도입한다. 제안된 방법들에서는 에너지 함수를 이용하여 확률 분포를 표현하는 에너지 기반 모형 방법론에 기반하여, 오토인코더와 가우시안 프로세스를 에너지 기반 확률 모형으로 새롭게 해석한다.

추가적으로, 본 논문에서는 에너지 기반 모형 기법에 관한 추가적인 아이디어들을 제시한다. 첫째, 데이터의 저차원 구조를 활용하는 에너지 기반 생성 모형의 새로운 학습 알고리즘을 제시한다. 제안된 알고리즘은 생성 모형이 학습 데이터가

아닌 곳에 높은 우도를 부여하는 현상을 효과적으로 억제할 수 있으며, 결과적으로 이러한 에너지 기반 모형들은 강력한 이상 탐지 성능을 보여준다. 둘째, 적대적 공격과 에너지 기반 모형 기법 사이의 연결을 탐색한다. 이러한 연결관계에 기반하여 이상치 탐지기를 위한 새로운 적대적인 생성 공격 알고리즘을 제안한다. 제안된 알고리즘은 에너지 기반 분포에서 샘플링하는 과정을 통해 적대적 공격을 수행한다.

인식론적 불확실성 정량화는 로봇 및 인공지능 에이전트가 세상과 안전하고 효과적으로 상호작용하는 데에 필수적인 요소이다. 이 논문에서는 확률 모형을 통해 인식론적 불확실성 정량화 문제를 해결할 수 있다는 것을 보였으며, 이 논문에서 논의된 에너지 기반 확률모형 기법과 알고리즘들은 로보틱스 및 인공지능 시스템의 다양한 응용에 널리 적용될 것으로 기대된다.

주요어: 생성모형, 에너지기반 모형, 불확실성 정량화, 이상탐지, 확률밀도 추정, 인식론적 불확실성

학번: 2020-38989