On the Learning of Energy-Based Models Using Noise Contrastive Estimation

by

Boming Shi

Thesis submitted to the University of Ottawa
in partial fulfillment of the requirements for the degree of
Master of Computer Science, Concentration in Applied Artificial Intelligence
in
Electrical Engineering and Computer Science

Examining Committee

The following served on the Examining Committee for this thesis.

Internal Member: Wonsook Lee
Professor, School of Electrical Engineering & Computer Science
University of Ottawa

External Member: Oliver van Kaick
Associate Professor, School of Computer Science
Carleton Univeristy

Supervisor: Yongyi Mao
Professor, School of Electrical Engineering & Computer Science
University of Ottawa

Declaration of Authorship

I hereby certify that this thesis is entirely my own original work except where otherwise indicated. I am aware of the University of Ottawa regulations concerning plagiarism, including those regarding consequent disciplinary actions. Any use of the works of any other author, in any form, is properly acknowledged at their point of use.

Abstract

Energy-Based Models (EBMs) are a family of unsupervised machine learning models that associate each point in the input space with an energy value in which low energy indicates a high likelihood. Specifically, an EBM can be viewed as an unnormalized probabilistic model, which upon normalization, gives rise to the probability density function of data. The main difficulty in learning EBMs lies in the computation of the normalization constant, or the partition function, a task known to be intractable in general. Several approaches have been proposed to avoid or overcome this difficulty, including Maximum Likelihood Estimation (MLE) with Markov chain Monte Carlo (MCMC), Score Matching (SM), Noise Contrastive Estimation (NCE), and so on.

This thesis studies the learning of EBMs using NCE. Briefly, in NCE, the EBM learning problem is converted to learning a binary classifier, which aims to distinguish the real data from fake data drawn from a noise distribution. This process allows the learning of the energy function in the EBM to bypass a direct estimation of the partition function and a certain theoretical guarantee is available under some assumptions in some asymptotic limit.

Despite the nice theoretical properties of NCE, in this work, we show that learning EBMs using NCE entails significant practical limitations. Specifically, there appears a tension between the quality of the learned model and the computational efficiency, due to which we must sacrifice one to achieve the other. We establish these limitations via empirical studies as well as a theoretical analysis based on a simple "Gaussian data learning problem". Our analysis inspires a revised NCE scheme, Adaptive Noise Contrastive Estimation (ANCE), to overcome these limitations. Empirically, we show that ANCE achieves a better quality-efficiency trade-off than the standard NCE in some regimes.

Acknowledgements

I would like to express my deepest gratitude to my supervisor, Professor Yongyi Mao, for his patience, solid support and continuous guidance throughout my research. His guidance helped me in all the time of research and writing of this thesis. Besides his academic knowledge in computer science and artificial intelligence, he also influenced me to shape a better life value. I believe I could not have a better tutor and mentor for my master's study than Prof. Mao.

I am also grateful to study with other students supervised by Prof. Mao. The various topics of the seminars vastly enriched my comprehensive understanding of the artificial intelligence industry. The opportunities of several presentations also helped me improve my speaking skills.

Last but not least, I would like to thank my parents and girlfriend for their never-ending support and encouragement throughout my study.

Dedication

This is dedicated to my mother Xinfen Qian and father Changkui Shi who always stand behind me, and my girlfriend Ying Ren who always encourages me.

Table of Contents

List of Tables

List of Figures

List of Algorithms

Abbreviations

ANCE Adaptive Noise Contrastive Estimation 69

CD Contrastive Divergence 20

CRB Cramér-Rao Bound 12

CRLB Cramér-Rao Lower Bound 12

DSM Denoising Score Matching 23

EBM Energy-based Model 1, 15

FD Fisher Divergence 22

FI Fisher Information 11

GAN Generative Adversarial Network 1, 32

GMM Gaussian Mixture Model 65

JSD Jensen–Shannon Divergence 8

KLD Kullback–Leibler Divergence 2, 5

MALA Metropolis-Adjusted Langevin Algorithm 20

MCMC Markov Chain Monte Carlo 19

MCMCML Markov Chain Monte Carlo Maximum Likelihood 19

MI Mutual Information 8

MI-NEE Mutual Information Neural Entropic Estimation 10

MINE Mutual Information Neural Estimation 10, 61

MLE Maximum Likelihood Estimation 2, 18

MMD Maximum Mean Discrepancy 74

NCE Noise Contrastive Estimation 2, 25

NF Normalizing Flow 1

NGD Normalizing Gradient Descent 82

RBF Radial Basis Function 78

SCE Self-contrastive Estimation 40

SFD Sliced Fisher Divergence 23

SM Score Matching 2, 22

SSM Sliced Score Matching 23

VAE Variational Autoencoder 1

WGAN Wasserstein Generative Adversarial Network 35

Chapter 1

Introduction

Energy-Based Models (EBM) are a family of unsupervised machine learning models that associate each point in the input space with an energy scalar where low energy suggests a high likelihood. Among various EBMs, Noise-Contrastive Estimation (NCE) is considered an efficient approach, aiming to learn the energy of data through binary classification involving noise data. However, there appear to be inadequate observations regarding the difficulty in the efficiency of NCE. This research aims to investigate the obstacles to learning EBMs using NCE and provide a revised framework, Adaptive Noise Contrastive Estimation (ANCE), that overcomes these limitations. This chapter will introduce the study by first discussing the research background, followed by the research motivation, research objectives, summary of contributions and chapter organizations.

1.1 Background

Neural networks and deep learning have made significant achievements and have been applied successfully in numerous fields [19] [41]. Most of the successes in deep learning lie in the area of supervised learning, while there is still significant room for improvement in unsupervised learning. Unsupervised learning is known as a machine learning task where training data are modelled without any information of labels or classes. Common unsupervised learning model includes Variational Autoencoder (VAE) [31], Generative Adversarial Network (GAN) [16], Normalizing Flow (NF) [61], Energy-based Model (EBM) [42] and so on. VAE is an architecture that consists of an encoder mapping the input data to latent space and a decoder reconstructing data from latent space, which are trained to minimize the loss function, which consists of a reconstruction term and a regularisation

term. [31] [32]. GAN is an adversarial network which consists of a generator mapping latent space to input space and a discriminator mapping input space to a scalar, which is trained alternatively to contest each other [16]. Both VAE and GAN can achieve state-of-art results in multiple scenarios as a generative model [60] [73] [66] [2] [81], however, both methods are designed as implicit model, where the likelihood of data cannot be evaluated. NF is a model where both generating and likelihood evaluation can be achieved simultaneously, because NF transforms a simple probability distribution into a complex one that can express the input space through a series of invertible and differentiable mappings [35] [61]. However, the disadvantages of NF include the failure on out-of-distribution detection [34] [83], expensive computing of the determinant of the Jacobian [36] and so on.

Except for the approaches above, EBM is another unified framework in unsupervised machine learning tasks which associates each data with an energy scalar where low energy suggests a high likelihood [59]. Applied in many machine learning problems [58] [24], EBM is often regarded as an unnormalized probabilistic model, similar to a density estimation model which is core and fundamental in unsupervised learning. Essentially, given any set of real-world observations, it can be assumed that the observations are some random samples from an underlying probability density function. Therefore, estimating the likelihood or energy from the observations becomes a "master" problem to other sub-problems. Good energy function can have various applications, such as evaluating the generating quality, providing a reward function in reinforcement learning [68] [23], and providing training signals for other deep learning tasks.

Different from many traditional density estimation approaches, EBM is an unnormalized probabilistic model that is difficult to learn because of the existence of normalization constant or partition function, whose evaluation is known to be intractable in high dimensional tasks. Many pieces of research from different angles are dedicated to overcoming or avoiding the calculation of partition function. For instance, one of the classical solutions to estimate the partition function in probabilistic graphical models is the message passing-based technique, such as belief propagation [1]. Other modern works manage to solve the problem by sampling or constructing variational bound to estimate the target, such as Maximum Likelihood Estimation (MLE) based methods [14] and variational bound of Kullback–Leibler Divergence (KLD) [56]. In addition to these mature methods, recent researches including Score Matching (SM) [70] and Noise Contrastive Estimation (NCE) [21], to some degree, avoid the direct calculation of partition, which also successfully solve the problem. Most of these methods can be unified under the framework of EBM and have connections to each other. For example, under certain conditions, MLE and NCE can be asymptotically equivalent [63]; SM can also be proved to have a similar objective function with NCE [21].

Among all the methods, NCE has been considered a highly efficient approach [21] [13] compared to other unnormalized methods. NCE has been widely applied in many different scenarios, including natural language process, speech recognition and so on [51] [50] [7] [79] [25] [78] [69]. Despite the elegant statistical consistency of NCE, the training of NCE may still encounter some obstacles. For example, it is widely believed that the noise should be chosen close to the data distribution, and it is reported that NCE is sensitive to the choice of noise distribution in natural language processing tasks [40] [47]. However, these problems of training NCE have not been adequately investigated either empirically or analytically.

1.2 Research Motivation and Objectives

This research is motivated by the curiosity to develop a deeper understanding of the training behaviour of NCE.

The objectives of this thesis include:

- Provide a comprehensive theoretical survey for modern EBMs and compare other methods with NCE.
- Investigate and summarize the training problems in NCE.
- Provide a revised version of NCE to overcome the problems.
- Evaluate the performance of revised NCE.

1.3 Summary of Contributions

In this thesis, we first gave a comprehensive introduction to modern EBMs. Specifically, we focus on the learning of EBMs using NCE. We specify and analyze three perspectives where learning obstacles exist and the trade-off between statistical and computational efficiency in NCE. According to the analysis results, we give several suggestions on the training of NCE. Finally, we propose a revised version of NCE, ANCE, which achieves a better quality-efficiency than the standard NCE.

1.4 Thesis Organization

The rest of this thesis will be arranged as follows:

Chapter 2 will introduce the mathematical preliminaries necessary for this study.

In Chapter 3, we will discuss the detailed background related to EBMs. A comprehensive survey on modern EBM approaches, including Maximum Likelihood Estimation (MLE) with Markov chain Monte Carlo (MCMC), Score Matching (SM), Noise Contrastive Estimation (NCE) and the connections between them will be given.

In Chapter 4, we will focus on EBM using NCE and discuss the training obstacles in NCE. There exists a trade-off between the quality of the learned model and the computational efficiency. We establish these obstacles through empirical studies by simplifying the model as a "Gaussian data learning problem," with which we suggest the criteria for training a good NCE model.

In Chapter 5, according to the analysis in Chapter 4, a revised NCE scheme, ANCE, is provided to overcome the limitations.

In Chapter 6, we show that ANCE achieves a better quality-efficiency trade-off than the standard NCE in some regimes through empirical studies.

In Chapter 7, we conclude the thesis with a discussion and present several possible directions for future works.

Chapter 2

Mathematical Preliminaries

2.1 Kullback–Leibler Divergence

2.1.1 Definition of KL Divergence

KL Divergence (KLD) is a measure of how a probability distribution is different from another [38]. Suppose there are two probabilities, P and Q, which are defined on the same probability space $\mathcal{X}$. The *KLD between P and Q* or *KLD from Q to P* (suggesting its asymmetry) is defined as following:

in the case of discrete probability distributions P and Q, where $p(x)$, $q(x)$ are probability mass functions of P and Q [29],

$$D_{KL}(P\|Q) = \sum_{x\in\mathcal{X}} p(x)\log\left(\frac{p(x)}{q(x)}\right); \tag{2.1}$$

in the case of continuous probability distributions P and Q, where $p(x)$, $q(x)$ are probability density functions of P and Q,

$$D_{KL}(P\|Q) = \int_{-\infty}^{+\infty} p(x)\log\left(\frac{p(x)}{q(x)}\right)\mathrm{d}x. \tag{2.2}$$

KLD is non-negative. $D_{KL}(P\|Q) \geq 0$ always holds for any distributions P and Q. $D_{KL}(P\|Q) = 0$ if and only if $P = Q$. It needs to be specified that the KL divergence

is not a metric measure because it is asymmetric. $D_{KL}(P\|Q) \neq D_{KL}(Q\|P)$ as long as $P(x) \neq Q(x)$. In addition, KL divergence does not always follow triangular inequality. That is, for any distributions P, Q and R, $D_{KL}(P\|Q) \leq D_{KL}(P\|R) + D_{KL}(R\|Q)$ does not always hold. Therefore, to some extent, KLD reflects the "distance" between two distributions; it is not a metric measure.

2.1.2 Dual Representation of KL Divergence

For high dimensional data, whose distribution underlying the data is unknown, it is not easy to calculate the KLD directly. A common technique is to transform the KLD to a variational bound which can be optimized by a family of the parameterized functions [64]. Donsker-Varadhan Representation is one of the variational bounds.

Theorem 2.1 (Donsker-Varadhan Representation)**.** *[3] The KL divergence admits the following dual representation:*

$$D_{KL}(P\|Q) = \sup_{T:\mathcal{X}\to\mathbb{R}} \mathbb{E}_{x\sim P}[T(x)] - \log \mathbb{E}_{x\sim Q}[e^{T(x)}], \tag{2.3}$$

where the supremum is taken over all functions T such that the two expectations are finite.

Proof. To prove the Donsker-Varadhan Representation, one can first construct a Gibbs distribution S which is related to function T. The pdf of S is defined by $s(x) = \frac{e^{T(x)}q(x)}{\mathbb{E}_{x\sim Q}[e^{T(x)}]\mathrm{d}x}$. Then $s(x)$ can be substituted into function $\mathbb{E}_{x\sim P}[\log \frac{s(x)}{q(x)}]$:

$$\begin{aligned}
\mathbb{E}_{x\sim P}[\log \frac{s(x)}{q(x)}] &= \mathbb{E}_{x\sim P}[\log \frac{\frac{e^{T(x)}q(x)}{\mathbb{E}_{x\sim Q}[e^{T(x)}]}}{q(x)}] \\
&= \mathbb{E}_{x\sim P}[\log \frac{e^{T(x)}}{\mathbb{E}_{x\sim Q}[e^{T(x)}]}] \\
&= \mathbb{E}_{x\sim P}[T] - \log \mathbb{E}_{x\sim Q}[e^{T(x)}].
\end{aligned} \tag{2.4}$$

We set:

$$\Delta = D_{KL}(P\|Q) - \mathbb{E}_{x\sim P}[T] - \log \mathbb{E}_{x\sim Q}[e^{T(x)}]. \tag{2.5}$$

Then eq.(2.4) can be plugged into eq.(2.5):

$$
\begin{aligned}
\Delta &= D_{KL}(P\|Q) - \mathbb{E}_{x\sim P}[\log \frac{s(x)}{q(x)}] \\
&= \mathbb{E}_{x\sim P}[\log \frac{p(x)}{q(x)}] - \mathbb{E}_{x\sim P}[\log \frac{s(x)}{q(x)}] \\
&= \mathbb{E}_{x\sim P}[\log \frac{p(x)}{q(x)} - \log \frac{s(x)}{q(x)}] \\
&= \mathbb{E}_{x\sim P}[\log \frac{p(x)}{s(x)}] \\
&= D_{KL}(P\|S) \\
&\geq 0,
\end{aligned} \tag{2.6}
$$

where $D_{KL}(P\|S) \geq 0$ always exists and the equality holds when and only when $P(x) = S(x)$, according to the Gibbs' inequality. Therefore, we can have:

$$
\begin{aligned}
\Delta &= D_{KL}(P\|Q) - \mathbb{E}_{x\sim P}[T] - \log \mathbb{E}_{x\sim Q}[e^T(x)] \\
&\geq 0,
\end{aligned} \tag{2.7}
$$

which is equivalent to following inequality:

$$
D_{KL}(P\|Q) \geq \mathbb{E}_{x\sim P}[T] - \log \mathbb{E}_{x\sim Q}[e^T(x)], \tag{2.8}
$$

for any function T. The equality is satisfied when and only when $s(x) = \frac{e^T(x)q(x)}{\mathbb{E}_{x\sim Q}[e^T(x)]\mathrm{d}x} = p(x)$, from which it can be derived as $T(x) = \log \frac{p(x)}{q(x)} + c$, where c is the log of partition function. □

An obvious result one can obtain from Donsker-Varadhan representation is a *tight* lower bound for KL divergence,

$$
D_{KL}(P\|Q) \geq \sup_{T:\mathcal{X}\to\mathbb{R}} \mathbb{E}_{x\sim P}[T(x)] - \log \mathbb{E}_{x\sim Q}[e^{T(x)}], \tag{2.9}
$$

where the equality is satisfied when and only when T is optimal.

Therefore, it is possible to approximate KL divergence through a variational method, by maximizing $\mathbb{E}_{x\sim P}[T(x)] - \log \mathbb{E}_{x\sim Q}[e^{T(x)}]$ when updating function $T(x)$, where $T(x)$ can be expressed as a form of deep neural network.

2.2 Jensen–Shannon Divergence

Jensen–Shannon Divergence (JSD) is also a measure of similarity of two distributions [53]. It is defined based on KLD:

$$D_{JS}(P\|Q) = \frac{1}{2}D_{KL}(P\|M) + \frac{1}{2}D_{KL}(Q\|M), \tag{2.10}$$

where $M = \frac{1}{2}(P+Q)$. JSD strictly satisfies the three conditions of a distance:

1. **Non-negativity:** $D_{JS}(P\|Q) \geq 0$;
2. **Symmetry:** $D_{JS}(P\|Q) = D_{JS}(Q\|P)$;
3. **Triangular Inequality:** $D_{JS}(P\|Q) \leq D_{JS}(P\|S) + D_{JS}(Q\|S)$.

Therefore JSD is a strictly defined metric measure.

2.3 Mutual Information

2.3.1 Definition of Mutual Information

Mutual Information (MI) is a concept in probability theory, and information theory [75]. Mutual information is a measure of mutual dependence between two random variables. It is a quantity indicating "how much information" is obtained about one random variable by observing the other random variable [75]. Intuitively, the value of MI shows how much information one can know from one random variable about another [54].

Suppose there are two random variables, X and Y, whose joint distribution is $P_{XY}(x,y)$, then the formal definition of MI of two random variables X and Y, is defined as follows:

In the case of discrete probability distributions P,the probability mass function is $p(x)$,

$$I(X;Y) = \sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{Y}} p_{XY}(x,y)\log\left(\frac{p_{XY}(x,y)}{p_X(x)p_Y(y)}\right); \tag{2.11}$$

in the case of continuous probability distributions P, the probability density function is $p(x)$,

$$I(X;Y) = \int_{\mathcal{X}} \int_{\mathcal{Y}} p_{XY}(x,y) \log\left(\frac{p_{XY}(x,y)}{p_X(x)p_Y(y)}\right) \mathrm{d}x\mathrm{d}y. \tag{2.12}$$

2.3.2 Some Properties of Mutual Information

2.3.2.1 Dual Form of Mutual Information

Mutual information can also be rewritten by the following form:

$$I(X;Y) = D_{KL}(P_{XY} \| P_X P_Y). \tag{2.13}$$

According to eq.(2.13), mutual information can be expressed as the KL divergence between a joint distribution and the product of marginal distributions. Applying the dual representation of KL divergence, mutual information can also be derived to a variational lower bound:

$$I(X;Y) = \sup_{T_\theta : \mathcal{X} \to \mathbb{R}} \mathbb{E}_{x \sim P_{XY}}[T_\theta(x)] - \log \mathbb{E}_{x \sim P_X \otimes P_Y}[e^{T_\theta(x)}], \tag{2.14}$$

where T_θ is any function; P_{XY} is the joint distribution of X and Y; $P_X \otimes P_Y$ represents the product of two marginal distributions of random variables X and Y, by supposing X and Y are independent. The equality holds true when and only when $T_\theta = \log \frac{p_{XY}(x)}{p_X(x)p_Y(x)} + c$, where c is the value of partition function.

2.3.2.2 Entropic Form of Mutual Information

Following the definition of MI, mutual information can also be derived as:

$$I(X;Y) = H(X) + H(Y) - H(X,Y), \tag{2.15}$$

where $H(X)$, $H(Y)$, $H(X,Y)$ denote the *entropy* of X, Y, joint random variables (X,Y). The entropy is defined as:

in the case of discrete probability distributions P, the probability mass function is $p(x)$,

$$H(X) = \sum_{x \in \mathcal{X}} p_X(x)(\frac{1}{\log(p_X(x))}); \tag{2.16}$$

in the case of continuous probability distributions P, the probability density function is $p(x)$,

$$H(X) = \int_{\mathcal{X}} p_X(x)(\frac{1}{\log(p_X(x))})\mathrm{d}x. \tag{2.17}$$

2.3.3 Estimation of Mutual Information

The estimation of mutual information from a sample of (X, Y) is difficult, especially when the dimension of the data is large. However, when utilizing the properties of mutual information and techniques of neural network, the estimation of mutual information becomes more tractable in the era of deep learning.

2.3.3.1 Mutual Information Neural Estimation

Mutual Information Neural Estimation (MINE) [3] is a method recently developed. Using eq.(2.14), one can obtain a neural estimator, by defining T_θ as a neural network. MINE has been proved as an effective and scalable method in multiple scenarios. Here, it is necessary to be specified that MINE not only gives an estimation of MI, but also provides an optimal function T_θ, which is equals to $T_\theta = \log \frac{p_{XY}(x)}{p_X(x)p_Y(x)} + c$.

2.3.3.2 Mutual Information Neural Entropic Estimation

Mutual Information Neural Entropic Estimation (MI-NEE) [5] is another approach for mutual information estimation. By applying eq.(2.15) and entropy estimation, MI-NEE can approximate mutual information by estimating three entropy terms respectively. Although the estimation is neither a lower bound or a upper bound, it is shown that MI-NEE have a faster learning speed compared to MINE [5].

The neural entropic estimation technique applied in MI-NEE can be derived as a form of variational upper bound by performing the same trick of Donsker-Varadhan Representation. One can first transform the entropy of X, to a form of cross entropy with a reference distribution Q and KLD between P and Q:

$$H(X) = \mathbb{E}_{x\sim P}[\log \frac{1}{Q(x)}] - D_{KL}(P\|Q). \tag{2.18}$$

By applying the dual representation of KLD to eq. (2.18), a variational upper bound for $H(X)$ can be obtained:

$$H(X) \leq \mathbb{E}_{x\sim P}[\log \frac{1}{Q(x)}] - \mathbb{E}_{x\sim P}[T_\theta(x)] + \log \mathbb{E}_{x\sim Q}[e^{T_\theta(x)}]. \tag{2.19}$$

Minimizing eq.(2.19) through updating parameterized function T_θ, an approximation of entropy can be achieved. The equivalence holds when and only when $T_\theta(x) = \ln P(x) - \ln Q(x) + c$. All the three entropy terms in mutual information eq.(2.15) can be simulated with the technique of neural entropic estimation. It should be mentioned that neural entropic estimation applied in MI-NEE also provides a framework of density estimation, since the minimum of eq.(2.19) will be achieved when $T^*_\theta(x) = \ln P(x) - \ln Q(x) + c$.

2.4 Fisher Information

Fisher Information (FI) is a measure of the amount of information that an observable random variable X carries about an unknown parameter θ of a distribution which models X [48]. It is defined the variance of the score: [44]

$$\mathcal{I}_x(\theta) = \mathbb{E}_{x\sim P_\theta}\left[\left(\frac{\partial}{\partial\theta}\log p_\theta(x)\right)^2\Bigg|_\theta\right], \tag{2.20}$$

where $\mathcal{I}_x(\theta)$ denotes the Fisher Information of the observable random variable X with model parameter θ, X is parameterized by θ and the probability of X is P_θ.

According to lemma provided in [44], FI can also be written as expectation of second derivatives of log-likelihood:

$$\mathcal{I}_x(\theta) = -\mathbb{E}_{x\sim P_\theta}\left[\frac{\partial^2}{\partial\theta^2}\log p_\theta(x)\Bigg|_\theta\right], \tag{2.21}$$

because the expectation of score equals to zero $\mathbb{E}_{x\sim P_\theta}\left[\frac{\partial}{\partial\theta}\log p_\theta(x)\big|_\theta\right] = 0$.

The goal of FI is to quantify how "well" the observations of random variable X locate the true parameter θ. In other words, when the FI of an observation is high, it tells more information about θ. When the FI of an observation is low, it tells less information about θ. For intuition, we can take two one dimensional Gaussian distributions as examples: $X \sim \mathcal{N}_1(0, 0.1^2)$ and $X \sim \mathcal{N}_2(0, 1^2)$. Suppose the means of two distributions are unknown

and parameterized by θ, as shown in figure 2.1. Intuitively, one can imagine that a small bias of θ when estimating $\mathcal{N}_1$ will have a larger influence on the estimation result than estimating $\mathcal{N}_2$. In other words, we can get more information about true parameter θ in $\mathcal{N}_1$ than $\mathcal{N}_2$.

From the definition of FI eq.(2.20), we can also say that FI is to quantify the sensitivity of the log-pdf of variable X to the value of the parameter θ. High Fisher information suggests high sensitivity to the change of the parameters. For example, if a slight change in θ leads to a significant change in the probability value of X, then the Fisher information is high, which means the samples observed tell us much information about θ. On the contrary, when the derivatives of pdf with respect to θ is small, the Fisher information is low. The overall Fisher information will be the sum of all samples. The Figure 2.2 shows an example for a single value of X.

2.5 Cramér-Rao Bound

Cramér-Rao Bound (CRB) or Cramér-Rao Lower Bound (CRLB) is an inequality which indicates the relationships between the variance of an estimator and Fisher information of x at parameter θ [37]. In the simplest case, if $\hat{\theta}$ is an unbiased estimator of parameter θ, the CRLB states:

$$\mathbf{Var}\left(\hat{\theta}\right) = \mathbb{E}_{x\sim p_\theta}[(\hat{\theta}-\theta)^2|_\theta)] \geq \frac{1}{\mathcal{I}_x(\theta)}. \tag{2.22}$$

Mathematically, an unbiased estimator means:

$$\mathbb{E}_{x\sim p_\theta}[\hat{\theta}-\theta|_\theta] = 0. \tag{2.23}$$

CRLB is a lower bound, which implies how "best" an unbiased estimator can be expected to be. CRLB suggests that with the increase of Fisher Information $\mathcal{I}_x(\theta)$, the variance of the estimator decreases; therefore, the estimator is better. CRLB is commonly used in the asymptotic analysis of an estimator. We say an unbiased estimator to be an *efficient estimator*, when the estimator attains the CRLB.

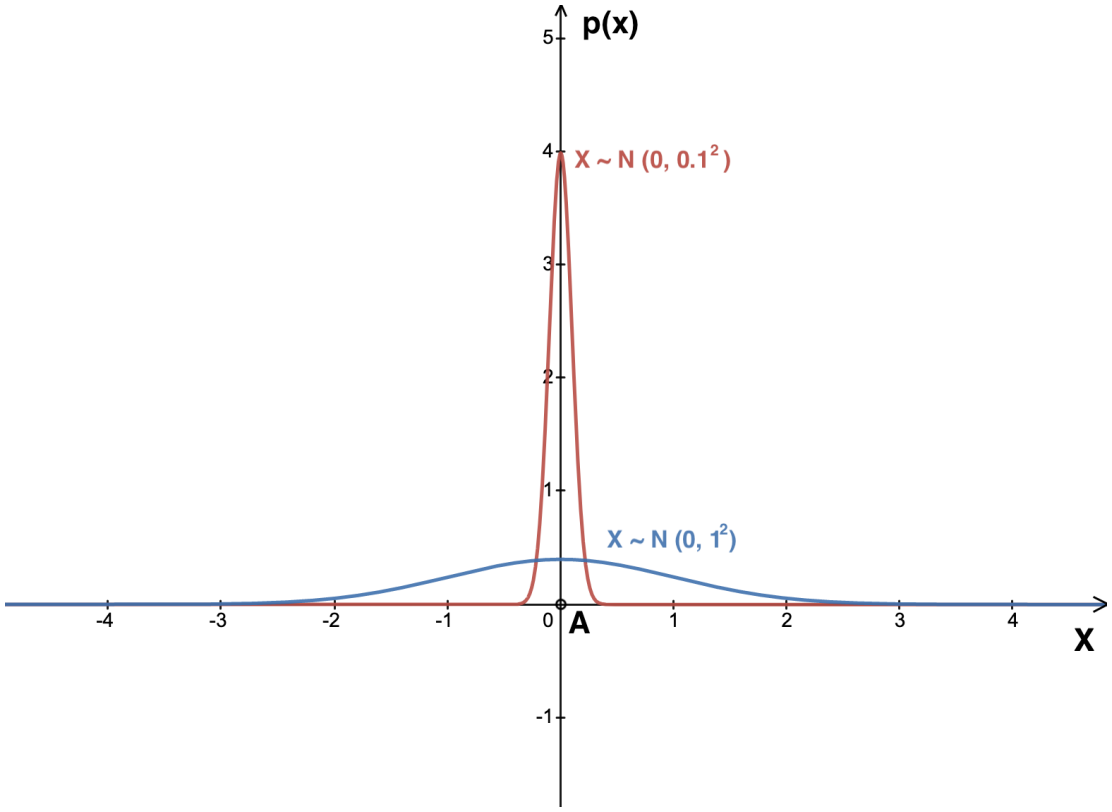

Figure 2.1: An intuitive understanding of Fisher Information on the example of two underlying Gaussian distributions. Suppose there are two one dimensional Gaussian distributions: $X \sim \mathcal{N}_1(0, 0.1^2)$ (red) and $X \sim \mathcal{N}_2(0, 1^2)$ (blue). The parameter θ is an estimator on the mean. The ground truth value of θ is zero. For $\mathcal{N}_1$, a small bias of theta will have larger influence in the estimation result, however, for $\mathcal{N}_2$, a small bias of theta will have smaller influence in the estimation result. In other words, θ is more sensitive in estimating $\mathcal{N}_1$. The fisher information of $\mathcal{N}_1$ is larger.

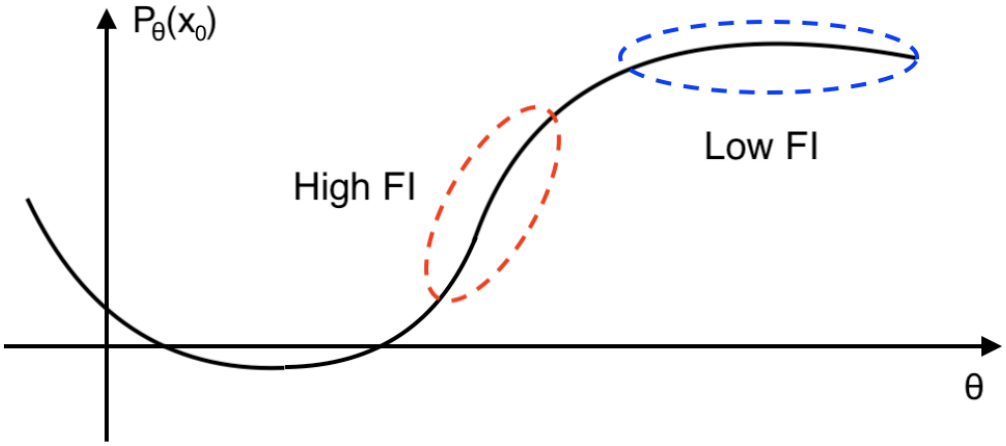

Figure 2.2: An example of function $P_\theta(x_0)$ with respect of θ, which can help understand the Fisher Information. In this example, it shows probability of one particular sample x_0. The part of the probability circled by the red dashed line changes rapidly as a function of θ. It suggests a large derivatives, which may lead to a high Fisher Information in this area. The area circled by the blue dashed line changes slowly as a function of θ, It suggests a small derivatives, which may lead to a low Fisher Information in this area. This example is just built on a single particular observation x_0, so it is just a demonstration for possible high or low FI. Real FI is calculated under all observations.

Chapter 3

Background

3.1 Energy-Based Model

3.1.1 Definition of Energy-Based Model

EBM is defined as a statistical model that associates scalar energy (a measure of compatibility) to each configuration of the variables [42]. In other words, EBM assigns a scalar *energy* $E(x)$ to the points x in data space $\mathcal{X}$, which indicates the "goodness" or "badness" of the x. EBM provides a common theoretical guide for various machine learning tasks and has a wide application in various scenarios, including supervised or unsupervised learning; decision-making tasks or probabilistic modelling; discriminative or generative tasks [42].

The standard form of EBM in the scenario of supervised learning and discriminative tasks is using a function of $E(X, Y)$ to indicate the compatibility between X and Y, while X is the random variable of observed data, the Y is the random variable of the predicted label. For example, X could be pixels of image and Y could be the label of the image. As shown in the following Figure 3.1. The lower value of energy indicates higher compatibility between label and data, while the higher value of energy suggests lower compatibility. More precisely, the energy is required to produce the most compatible or optimal label Y^* with the smallest value $E(X, Y)$ among all the possible Y, for all X Figure 3.1:

$$Y^* = \arg\min_{Y \in \mathcal{Y}} E(X, Y). \tag{3.1}$$

In addition to supervised learning, EBM is also commonly used in unsupervised learning. This thesis will mainly focus on the EBM of probabilistic modelling in unsupervised

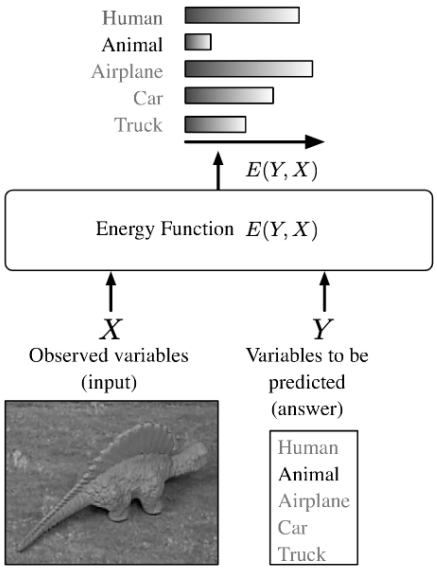

Figure 3.1: An example of energy based model, referenced from [42]. The lower value of energy indicates higher compatibility between label and data.

learning. Though the energy functions may have different forms, most probabilistic models can be regarded as particular types of energy-based models. The simplest and most common form of an energy model is through the *Gibbs distribution* or *Boltzmann distribution*:

$$p(x) = \frac{e^{-E(x)}}{\int_{x \in \mathcal{X}} e^{-E(x)}}, \tag{3.2}$$

where $p(x)$ is the normalized probability density function of x, the denominator is know as *partition function*. It should be noted that this transformation from energy to probability guarantees the integral of $p(s)$ is 1, as long as the integral term in eq.(3.2) converges. It is also important to know that the partition function is intractable when x is a high dimensional variable, and the integral term has no analytical solution. Hence, how to estimate the integral term or avoid the calculation of the partition function decides the feasibility of an energy-based model.

Through eq.(3.2),

$$E(x) = -\ln p(x) + c, \tag{3.3}$$

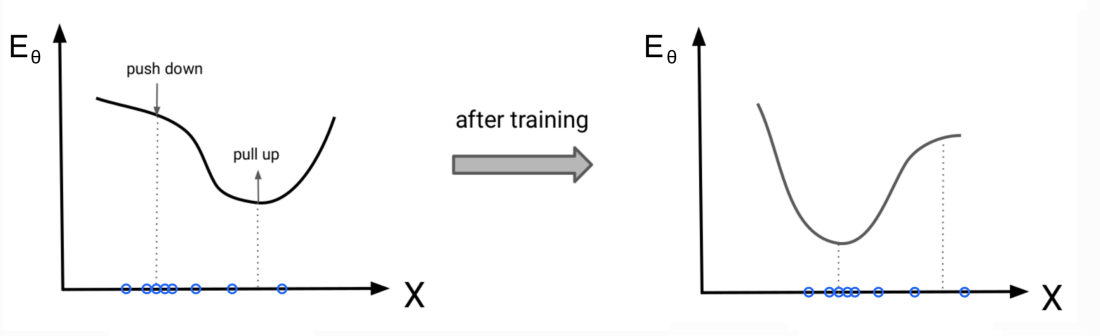

Figure 3.2: Pushing down and pulling up effect in energy model, referenced from [42].

where c is log of the partition function. The construction through eq.(3.2) and eq.(3.3) guarantee the energy equals the negative log-likelihood, which corresponds to the definition of energy of low-value mapping high compatibility and high-value mapping low compatibility. The essential purpose of training a probabilistic model EBM is to "push down" the energy value in high-density regions and "pull up" the energy value in low-density regions, as shown in figure 3.2.

Applying the framework of EBM to other machine learning tasks will enhance model performance. One obvious advantage is that it provides explicit probability density information, which can be used either to assist the training process or present as a result for further tasks. For example, in Generative Adversarial Network, the application of EBM has been proved that it has improved in better mode coverage [39]. Other advantages, including compositionality and flexibility, are also mentioned in other papers [12] [26].

3.1.2 Training of Energy-Based Model

There are many different ways to design EBMs. The traditional approaches of EBM include, *Restricted Boltzmann Machine* [84], *Product of Experts* [26] [27] and so on. Loss functions for EBM also has various forms, including *Perceptron Loss*, *Hinge Loss*, *LVQ2 Loss* and so on [43] [42]. With the introduction and application of deep neural network, more EBM has been explored to solve high dimensions problems recently, including Maximum Likelihood training with Markov chain Monte Carlo sampling; Score Matching (SM) [72]; Noise Contrastive Estimation (NCE) [21] and others methods. All these methods have some connections with others. We will introduce the common methods in EBM in the following sections.

3.2 Maximum Likelihood Estimation

3.2.1 Markov Chain Monte Carlo Maximum Likelihood

One of the most popular density estimation methods is MLE. Let $p_\theta(x)$ be a probabilistic model parameterized by θ; $p_d(x)$ be the data distribution. MLE fits $p_\theta(x)$ to $p_d(x)$ by maximizing the expectation of log-likelihood over the data distribution, as following:

$$\mathcal{L}_{MLE} = \mathbb{E}_{x\sim p_d}[\log p_\theta(x)]. \tag{3.4}$$

here p_θ is a normalized probability. For an unnormalised energy model $E(x)$, it can be normalized to a probability through:

$$p_\theta(x) = \frac{e^{-E_\theta(x)}}{Z_\theta}, \tag{3.5}$$

where $Z_\theta = \int e^{-E_\theta(x)}\mathrm{d}x$, then:

$$\nabla_\theta \log p_\theta(x) = -\nabla_\theta E_\theta(x) - \nabla_\theta \log Z_\theta, \tag{3.6}$$

the first term is straightforward to compute, however the second term, which cannot be calculated directly, requires approximation. The second term could be derived furthermore by:

$$\begin{aligned}
\nabla_\theta \log Z_\theta &= \nabla_\theta \log \int e^{-E_\theta(x)}\mathrm{d}x \\
&= \frac{1}{\int e^{-E_\theta(x)}\mathrm{d}x} \cdot \nabla_\theta \int e^{-E_\theta(x)}\mathrm{d}x \\
&= \frac{1}{\int e^{-E_\theta(x)}\mathrm{d}x} \cdot \int \nabla_\theta e^{-E_\theta(x)}\mathrm{d}x \\
&= \frac{1}{\int e^{-E_\theta(x)}\mathrm{d}x} \cdot \int e^{-E_\theta(x)} \cdot (-\nabla_\theta E_\theta(x))\mathrm{d}x \\
&= \int \frac{e^{-E_\theta(x)}}{\int e^{-E_\theta(x)}\mathrm{d}x} \cdot (-\nabla_\theta E_\theta(x))\mathrm{d}x \\
&= \mathbb{E}_{x\sim p_\theta(x)}(-\nabla_\theta E_\theta(x)).
\end{aligned} \tag{3.7}$$

Thus instead of computing the partition function, one can calculate the expectation

of $-\nabla_\theta E_\theta(x)$, over the estimated model distribution, as long as it is feasible to sample accurately from it. Markov Chain Monte Carlo (MCMC), provides a general framework for simulation of complex stochastic processes. It is a common and important family of approaches for sampling from a known distribution [14].

The framework of the algorithm of Markov Chain Monte Carlo Maximum Likelihood (MCMCML) by gradient descent can be summarized as 3.1:

Algorithm 3.1 Algorithm of MCMCML

Input: epoch: N; the number of observed data: k; the number of MCMC data: j; learning rate η; initialized energy parameter: θ_0;
Output: Parameter θ_N of energy
  Initialize θ_0;
  Set $n = 0$;
  while $n \leq N$ **do**
    Apply MCMC process to get simulated data for $x \sim p_\theta(x)$;
    Calculate $Gradient = \frac{1}{k} \sum_{x \sim p_d(x)} (\nabla_\theta E_\theta(x)) - \frac{1}{j} \sum_{x \sim p_\theta(x)} (\nabla_\theta E_\theta(x))$;
    $\theta_{n+1} = \theta_n - \eta \cdot Gradient$;
    $n = n + 1$
  end while

The MCMC process may vary according to different MCMC algorithms. Since sampling from an probability is an important topic, some efficient MCMC methods and their variants have been researched widely and deeply, such as Langevin MCMC, Hamiltonian MCMC. For example, Langevin MCMC takes an initial sample x^0 from a prior distribution, and then simulates Langevin dynamics for K steps with step size $\epsilon > 0$ [8]:

$$x^{k+1} = x^k + \frac{\epsilon^2}{2} \nabla_x \log p_\theta(x^k) + \epsilon \mathbf{n}, \tag{3.8}$$

where $\mathbf{n} \sim \mathcal{N}(0, I)$ is a random Gaussian noise.

Theoretically, under some regularity conditions, when $\epsilon \to 0$ and $k \to \infty$, a group of x^k is guaranteed to correspond to the target distribution $p_\theta(x)$ after repeating to get x^k with random initialized x^0 for many times [8]. The algorithm of Langevin MCMC is shown in 3.2

In practice, it is common to use a small, finite, ϵ and apply Metropolis-Hastings step, to enhance training efficiency [80]. Langevin MCMC combined with Metropolis-Adjusted

Algorithm 3.2 Algorithm of Langevin MCMC

Input: Number of burn-in steps: K; Number of samples: N;
Output: N samples sampling from the target distribution: $x^{K+1}, x^{K+2}, x^{K+3}, ..., x^{K+N}$
Set $i = 0$
Initialize x^0;
while $i \leq K + N$ **do**
Sample $x^{i+1} \sim \mathcal{N}(x^i + \frac{\epsilon^2}{2}\nabla_x \log p_\theta(x^i), \epsilon^2\mathbf{I})$ ▷ Equivalent to $x^{i+1} = x^i + \frac{\epsilon^2}{2}\nabla_x \log p_\theta(x^i) + \epsilon\mathbf{n}$
$i = i + 1$
end while

MCMC is also know as Metropolis-Adjusted Langevin Algorithm (MALA). The algorithm is as shown in 3.3.

Algorithm 3.3 Metropolis-Adjusted Langevin Algorithm

Input: Number of burn-in steps: K; Number of samples: N;
Output: N samples sampling from the target distribution: $x^{K+1}, x^{K+2}, x^{K+3}, ..., x^{K+N}$
Initialize x^0;
Set $i = 0$
while $i \leq K + N$ **do**
Sample $y^{i+1} \sim \mathcal{N}(x^i + \frac{\epsilon^2}{2}\nabla_x \log p_\theta(x^i), \epsilon^2\mathbf{I})$
Compute $\alpha_{i+1} = \min\left(1, \frac{p_\theta(y^{i+1}) \exp\left[-\|x^i - y^{i+1} - \frac{\epsilon^2}{2}\nabla \log p_\theta(y^{i+1})\|_2^2/2\epsilon^2\right]}{p_\theta(x^i) \exp\left[-\|y^{i+1} - x^i - \frac{\epsilon^2}{2}\nabla \log p_\theta(x^i)\|_2^2/2\epsilon^2\right]}\right)$ ▷ which is derived from $\alpha = \min\{\frac{p_\theta(y_{i+1})q(x_i|y_{i+1})}{p_\theta(x_i)q(y_{i+1}|x_i)}, 1\}$
Sample $u_{i+1} \sim U[0, 1]$
if $u_{i+1} \leq \alpha_{i+1}$ **then** Accept Sample $x^{i+1} = y^{i+1}$;
else Reject Sample $x^{i+1} = x^i$;
end if
$i = i + 1$
end while

Although the MCMC family provides a way to sample from any distribution, it is difficult to run MCMC till convergence because it is usually computationally expensive, and the performance may not be as ideal as expected, especially when the distribution is high dimensional and complex [45] [10]. Numerous papers concentrate on improving the efficiency and accuracy of MCMC sampling [65] [49] [67]. A significant method is Contrastive

Divergence (CD) [27], which can highly enhance the efficiency. However, overall, MCMC family approaches still have difficulties with the accuracy of sampling [4].

3.2.2 Adversarial Training of Maximum Likelihood Estimation

Avoiding the sampling procedure is one idea to overcome the hurdle of the partition function. One approach is to learn an auxiliary model through adversarial training, which allows fast sampling or sample generation [82]. One can introduce a variational distribution $q_\phi(x)$ parameterized by ϕ as an auxiliary distribution:

$$\begin{aligned}
\mathcal{L}_{MLE} &= \mathbb{E}_{x\sim p_d}[\log p_\theta(x)] \\
&= \mathbb{E}_{x\sim p_d}[-E_\theta(x)] - \log \int e^{-E_\theta(x)} \mathrm{d}x \\
&= \mathbb{E}_{x\sim p_d}[-E_\theta(x)] - \log \int q_\phi(x) \cdot \frac{e^{-E_\theta(x)}}{q_\phi(x)} \mathrm{d}x \\
&\leq \mathbb{E}_{x\sim p_d}[-E_\theta(x)] - \int q_\phi(x) \cdot \log \frac{e^{-E_\theta(x)}}{q_\phi(x)} \mathrm{d}x \\
&= \mathbb{E}_{x\sim p_d}[-E_\theta(x)] - \mathbb{E}_{x\sim q_\phi}[-E_\theta(x)] - H[q_\phi(x)],
\end{aligned} \tag{3.9}$$

where $H[q_\phi(x)]$ denotes the entropy of $q_\phi(x)$. The equality holds when and only when $q_\phi(x) = p_d(x)$ This inequality identifies an upper bound to the expected log-likelihood. Therefore, it suggests an adversarial learning in which one can first minimize the upper bound in eq.(3.9) by updating $q_\phi(x)$ to make it closer to the likelihood objective, and then maximize eq.(3.9) by updating $E_\theta(x)$ to implement MLE. The adversarial training can be summarized as following mini-max loss function:

$$\max_\theta \min_\phi \mathbb{E}_{x\sim p_d}[-E_\theta(x)] - \mathbb{E}_{x\sim q_\phi}[-E_\theta(x)] - H[q_\phi(x)]. \tag{3.10}$$

The ideal modelling of q_ϕ should satisfy two criteria: easy to sample and easy to compute the entropy [71]. It restricts the family of $q_\phi(x)$ within invertible probabilistic models, such as *inverse autoregressive flow* [33], *normalizing flow* [55], *non-linear independent components estimation* [11] and so on.

Another feasible plan of this model is that q_ϕ can be designed as an implicit probabilistic model, such as a neural network generator mapping latent space to data space $x = g_\phi(z)$.

However, it may sacrifice the efficiency of calculation of entropy term because the density of $q_\phi(x)$ does not have an explicit form. Therefore, it requires an approximation of entropy. In [30] and [82], authors propose several heuristics to estimate the entropy. In [39], authors propose that estimation of entropy is equivalent to estimate the mutual information by $H(g_\phi(z)) = I(g_\phi(z); z)$, which can be approximated through a variational lower bound provided by MINE. In [9], authors connect entropy to KLD between $q_\phi(x)$ and a reference distribution that can likewise be estimated by a lower bound similar to the concept of entropic estimation mentioned in section 2.3.3.2. However, all methods by constructing a generator involve a careful handle with the entropy term. Some of them may need an additional inner loop for entropy estimation, which may take more computation resources than GAN, while GAN is known to be difficult to train.

3.3 Score Matching

SM is a family of EBM which has been developed recently. SM completely bypasses the obstacles of the partition function estimation [46]. A *score* of a probability density is defined as the first-order gradient function of the log-pdf [46]:

$$score(x) = \nabla_x \log p(x), \tag{3.11}$$

for two continuously differentiable functions $f(x)$ and $g(x)$ whose first derivatives are equal everywhere, it is easy to know that $f(x) = g(x) + constant$. This provides a basic logic for score matching methods. SM usually utilizes a neural networks $\mathbf{S}(x)$ to estimate the score. It has also been proved to be an efficient EBM, by modeling $E(x)$ [70]:

$$\nabla_x \log p_\theta(x) = -\nabla_x E_\theta(x). \tag{3.12}$$

The objective of SM is to minimize a discrepancy between two distributions, which is named as the Fisher Divergence (FD) [71]:

$$\begin{aligned} D_F(p_d \| p_\theta) &= \mathbb{E}_{x\sim P_d}[\frac{1}{2}\|\nabla_x \log p_d(x) - \nabla_x \log p_\theta(x)\|^2] \\ &= \mathbb{E}_{x\sim P_d}[\frac{1}{2}\|\nabla_x \log p_d(x) + \nabla_x E_\theta(x)\|^2]. \end{aligned} \tag{3.13}$$

Because the probability of data is unknown, the FD can not be directly utilized as a loss function. A transformation of the FD has to be performed. It has been proved in [28]

that under certain regularity conditions, FD is equivalent to following equation [46]:

$$\begin{aligned} D_F(p_d\|p_\theta) &= \mathbb{E}_{x\sim P_d}[tr(\nabla_x^2 \log p_\theta(x)) + \frac{1}{2}\|\nabla_x \log p_\theta(x)\|^2] + constant \\ &= \mathbb{E}_{x\sim P_d}[-tr(\nabla_x^2 E_\theta(x)) + \frac{1}{2}\|\nabla_x E_\theta(x)\|^2] + constant \\ &= \mathbb{E}_{x\sim P_d}[-\sum_{i=1}^{d}\frac{\partial^2 E_\theta(x)}{(\partial x_i)^2} + \frac{1}{2}\sum_{i=1}^{d}(\frac{\partial E_\theta(x)}{\partial x_i})^2] + constant, \end{aligned} \tag{3.14}$$

where d is the dimensionality of x. The constant term is irrelevant to θ, and its derivatives with respect to θ are zero, so it can be ignored while training. Eq.(3.14) is well known as the objective function of SM. In eq.(3.14), the second term $\frac{1}{2}\sum_{i=1}^{d}(\frac{\partial E_\theta(x)}{\partial x_i})^2$ is fast to compute, because it involves the first derivative of energy function. The computational cost is $O(d)$. However, the first term, $-\sum_{i=1}^{d}\frac{\partial^2 E_\theta(x)}{(\partial x_i)^2}$, is much more computational expensive, whose cost is $O(d^2)$, because it requires the trace of second derivatives matrix of energy function, also known as Hessian matrix. In order to overcome the difficulty, several methods including Denoising Score Matching (DSM) [77], Sliced Score Matching (SSM) [70] have been developed. Especially, SSM elegantly reduces the computational cost to $O(2d)$ by projecting the scores onto random vectors [70] through the following equation, also known as Sliced Fisher Divergence (SFD) [70]:

$$\begin{aligned} D_{SF}(p_d\|p_\theta) &= \mathbb{E}_{x\sim P_d}\mathbb{E}_{v\sim P_v}[\frac{1}{2}\|v^T\nabla_x \log p_d(x) - v^T\nabla_x \log p_\theta(x)\|^2] \\ &= \mathbb{E}_{x\sim P_d}\mathbb{E}_{v\sim P_v}[\frac{1}{2}\|v^T\nabla_x \log p_d(x) + v^T\nabla_x E_\theta(x)\|^2], \end{aligned} \tag{3.15}$$

where P_v is a projection distribution such that $E_{p(v)}[vv^T]$ is positive definite [70]. Instead of comparing the score function directly, SFD projects the score onto a random vector. Similar to FD, SFD can also be rewritten in a form without explicit representation of data distribution [70]:

$$
\begin{aligned}
D_{SF}(p_d\|p_\theta) &= \mathbb{E}_{x\sim P_d}\mathbb{E}_{v\sim P_v}[v^T\nabla_x^2 \log p_\theta(x)v + \frac{1}{2}(v^T\nabla_x \log p_\theta(x))^2] + constant \\
&= \mathbb{E}_{x\sim P_d}\mathbb{E}_{v\sim P_v}[-v^T\nabla_x^2 E_\theta(x)v + \frac{1}{2}(v^T\nabla_x E_\theta(x))^2] + constant \\
&= \mathbb{E}_{x\sim P_d}\mathbb{E}_{x\sim P_v}[-\sum_{i=1}^{d}\sum_{j=1}^{d}\frac{\partial^2 E_\theta(x)}{\partial x_i \partial x_j}v_i v_j + \frac{1}{2}\sum_{i=1}^{d}(\frac{\partial E_\theta(x)}{\partial x_i}v_i)^2] + constant \\
&= \mathbb{E}_{x\sim P_d}\mathbb{E}_{x\sim P_v}[-\sum_{i=1}^{d}\sum_{j=1}^{d}\frac{\partial^2 E_\theta(x)}{\partial x_i \partial x_j}v_i v_j + \frac{1}{2}\sum_{i=1}^{d}(\frac{\partial E_\theta(x)}{\partial x_i}v_i)^2] + constant.
\end{aligned}
\tag{3.16}
$$

Compared to the eq. (3.15), eq. (3.16) also involves the second order derivatives of $E_\theta(x)$. However, it can be written as follows [70], and can be computed with $O(2d)$ [70]:

$$
\sum_{i=1}^{d}\sum_{j=1}^{d}\frac{\partial^2 E_\theta(x)}{\partial x_i \partial x_j}v_i v_j = \sum_{i=1}^{d}\frac{\partial E_\theta(x)}{\partial x_i}(\sum_{j=1}^{d}\frac{\partial E_\theta(x)}{\partial x_j}v_j)v_i. \tag{3.17}
$$

3.4 Variational Bound of KL Divergence

Recall the objective function, eq. (2.9), for entropic estimation introduced in Section 2. Actually, the variational bound of KL divergence can be regarded as EBM loss function. According to eq. (2.9), a loss function can be defined as following:

$$
\mathcal{L} = \mathbb{E}_{x\sim P_d}[-E_\theta(x) - \ln p_r(x)] - \ln \mathbb{E}_{x\sim P_r}[e^{-E_\theta(x)-\ln p_{ref}(x)}], \tag{3.18}
$$

where $E_\theta(x)$ is an unnormalised energy parameterized by θ; p_r is a reference distribution arbitrarily chose. The loss function will reach its maximum when and only when energy function reaches its optimality $E_\theta^*(x) = -\ln p_d + c$. The constant term c is the normalized term or the value of partition function. Therefore the constant term will also be learned in this situation theoretically. (3.18) can also be transformed as a mini-max form which is know as Fenchel Mini-Max Learning [74].

3.5 Noise Contrastive Estimation

NCE is another family of density estimation methods or energy-based models. The name comes from its underlying idea, which is to estimate the density by discriminating between data and contrastive noise with a fixed distribution [21].

3.5.1 Definition of Noise Contrastive Estimation

Given an unnormalized probabilistic model $p_m^0(x, \alpha)$ that is parameterized by α, one can obtain a normalized model $p_m(x, \theta)$ whose integral is 1 by adding an additional parameter c to p_m^0:

$$\ln p_m(x, \theta) = \ln p_m^0(x, \alpha) + c, \tag{3.19}$$

where c is the normalizing term or the value of log partition function. We can capsule parameter c in θ where $\theta = \{\alpha, c\}$. θ is the estimator parameters in NCE. Suppose $X = (x_1, x_2, ..., x_{T_d})$ are T_d independent observations from the dataset to be estimated. $x \in \mathbb{R}^n$, and underlying pdf of data distribution is denoted as $p_d(x)$. $Y = (y_1, y_2, ..., y_{T_n})$ are the T_n noise samples drew from an artificially defined noise distribution. $y \in \mathbb{R}^n$ and the pdf of noise distribution is denoted as $p_n(x)$. $T_n = \nu T_d$, where ν is the ratio of samples amount between noise and data. The objective function of NCE is defined as [20]:

$$J_T(\theta) = \frac{1}{T_d}\{\sum_{t=1}^{T_d} \ln[D(x_t; \theta)] + \sum_{t=1}^{T_n} \ln[1 - D(y_t; \theta)]\}, \tag{3.20}$$

where,

$$D(x, \theta) = \sigma[\ln p_m(x, \theta) - \ln p_n(x)], \tag{3.21}$$

σ is a modified sigmoid function: $\sigma(x) = \frac{1}{1+\nu e^{-x}}$, which restricts the outputs of $D(x, \theta)$ between 0 and 1. Because $T_n = \nu T_d$, eq. (3.20) can also be written as:

$$J_T(\theta) = \frac{1}{T_d} \sum_{t=1}^{T_d} \ln[D(x_t; \theta)] + \nu \frac{1}{T_n} \sum_{t=1}^{T_n} \ln[1 - D(y_t; \theta)]. \tag{3.22}$$

Since the normalizing model $p_m(x, \theta)$ can be expressed as an EBM, according to eq (3.5), eq. (3.22) can be derived as:

$$J_T(\theta) = \frac{1}{T_d}\sum_{t=1}^{T_d} \ln\{\sigma[-E_\theta(x_t) - \ln p_n(x_t)]\} + \nu\frac{1}{T_n}\sum_{t=1}^{T_n} \ln\{1 - \sigma[-E_\theta(y_t) - \ln p_n(y_t)]\}, \tag{3.23}$$

where $E_\theta(x_t)$ is a normalized energy model, which is $\int \frac{e^{-E_\theta(x)}}{Z_\theta} = 1$. $E_\theta(x_t)$ can be defined by an unnormalised energy model $E^0_\alpha(x_t)$ with an adding constant term c, $E_\theta(x_t) = E^0_\alpha(x_t) + c$. Since $\theta = \{\alpha, c\}$ includes normalizing term c, both α and c will be learnt during the training of NCE. Therefore it is not necessary to define the $E_\theta(x_t)$ as a normalized energy model. $E_\theta(x_t)$ will have a self-normalizing effect during the training of NCE.

Intuitively, when $[-E_\theta(x_t, \theta) - \ln p_n(x_t)] \to \infty$, $D(x, \theta) \to 1$; when $[-E_\theta(x_t, \theta) - \ln p_n(x_t)] \to -\infty$, $D(x, \theta) \to 0$. Therefore, intuitively, by maximizing $J_T(\theta)$, the estimator has the effects of pushing down the energy as low as possible where $x \sim p_d$ and pulling up the the energy as high as possible where $y \sim p_n$. Therefore NCE is a good energy based model at first sight. The algorithm of an energy based NCE is shown in 3.4.

Algorithm 3.4 Noise-Contrastive Estimation Algorithm

Input: Number of data samples: T_d; Number of noise samples: $\nu \cdot T_d$; Noise distribution: P_n; Epoch: *epoch*; Learning Rate: η

Output: Initialize energy E_θ;

Set $i = 0$

while $i \leq epoch$ **do**

Sample $y \sim P_n$

Compute $J_T(\theta) = -\frac{1}{T_d}\sum_{t=1}^{T_d} \ln\{\sigma[-E(x_t, \theta) - \ln p_n(x_t)]\} - \nu\frac{1}{\nu \cdot T_d}\sum_{t=1}^{\nu \cdot T_d} \ln\{1 - \sigma[-E(y_t, \theta) - \ln p_n(y_t)]\}$

Compute gradient: $\frac{\partial J_T(\theta)}{\partial \theta}$

Update θ: $\theta = \theta - \eta \cdot \frac{\partial J_T(\theta)}{\partial \theta}$

$i = i + 1$

end while

3.5.2 Connection to Supervised Learning

NCE is an unsupervised learning methods but Eq. (3.20) can be understood from a perspective of supervised learning, because it is similar to training a classifier based on logistic regression which discriminates the observed data X from the noise Y [22].

Suppose we mix samples from X and Y together as one set. $U = (u_1, ..., u_{Td+Tn})$ denotes the union of two datasets. Each data $u_t \in U$ has a label $C_t : C_t = 1$ if $u_t \in X$ or $C_t = 0$ if $u_t \in Y$. In this situation, NCE can be regarded as a logistic regression which models the posterior probabilities of classes given X. $p_m(u, theta)$ is normalized probabilistic model, which is also the conditional probability of u given the label $C_t : C_t = 1$. Similarly, noise distribution is the conditional probability of u given the label $C_t : C_t = 0$.

$$p(u|C = 1, \theta) = p_m(u, \theta); \tag{3.24}$$

$$p(u|C = 0) = p_n(u). \tag{3.25}$$

The prior probabilities are also known as $P(C = 1) = \frac{T_d}{T_d+T_n} = \frac{1}{1+\nu}$ and $P(C = 0) = \frac{T_n}{T_d+T_n} = \frac{\nu}{1+\nu}$. Therefore, The posterior probabilities of two classes are respectively:

$$P(C = 1|u, \theta) = \frac{p_m(u, \theta)}{p_m(u, \theta) + \nu p_n(u)}; \tag{3.26}$$

$$P(C = 0|u, \theta) = \frac{\nu p_n(u)}{p_m(u, \theta) + \nu p_n(u)}. \tag{3.27}$$

Since the labels C_t are Bernoulli distributed, a loss function is defined as following:

$$\begin{aligned}
\mathcal{L}(\theta) &= \sum_{t=1}^{T_d+T_n} C_t \ln P(C = 1, u_t|\theta) + (1 - C_t) \ln P(C = 1, u_t|\theta) \\
&= \sum_{t=1}^{T_d} \ln P(C = 1|u, \theta) + \sum_{t=1}^{T_n} \ln P(C = 0|u, \theta) \\
&= \sum_{t=1}^{T_d} \ln \frac{p_m(u, \theta)}{p_m(u, \theta) + \nu p_n(u)} + \sum_{t=1}^{T_n} \ln \frac{\nu p_n(u)}{p_m(u, \theta) + \nu p_n(u)} \\
&= \sum_{t=1}^{T_d} \ln \frac{1}{1 + \frac{\nu p_n(u)}{p_m(u,\theta)}} + \sum_{t=1}^{T_n} \ln[1 - \frac{1}{1 + \frac{\nu p_n(u)}{p_m(u,\theta)}}] \\
&= \sum_{t=1}^{T_d} \ln \frac{1}{1 + \nu e^{-[\ln p_m(u,\theta) - \ln p_n(u)]}} + \sum_{t=1}^{T_n} \ln[1 - \frac{1}{1 + \nu e^{-[\ln p_m(u,\theta) - \ln p_n(u)]}}],
\end{aligned} \tag{3.28}$$

which is equivalent to eq.(3.22), which builds a connection between unsupervised and supervised learning on NCE. Maximizing the loss function $\mathcal{L}(\theta)$ by updating p_m will yield an estimation of p_d.

3.5.3 Theorems and Corollaries

In this section, we will reference some fundamental statistical properties of NCE which can bring us deeper understanding. The performance of NCE is related to sample sizes T_d, the fixed ratio ν and the choice of the noise distribution. We will discuss how these factors impact the estimator through the theorems and corollaries of NCE.

3.5.3.1 Non-parametric estimation

According to the weak law of large numbers, it is easy to know that with the increasing of T_d, the objective function of $J_T(\theta)$ in eq.(3.23) converges to $J(\theta)$, where $J(\theta)$ is:

$$J(\theta) = \mathbb{E}_{x\sim p_d} \ln[D(x;\theta)] + \nu \mathbb{E}_{y\sim p_n} \ln[1 - D(y;\theta)]\}, \tag{3.29}$$

suppose that θ is parameters of unnormalised energy, $J(\theta)$ can also be written as:

$$J(E_\theta) = \mathbb{E}_{x\sim p_d} \ln[\sigma(-E_\theta(x) - \ln p_n(x))] + \nu \mathbb{E}_{y\sim p_n} \ln[1 - \sigma(-E_\theta(y) - \ln p_n(y))]\}. \tag{3.30}$$

The first theorem of NCE is important because it shows that the log-pdf or energy of data distribution can be obtained by maximization of J, under certain regulations of noise data. This theorem is called non-parametric estimation in [22], because the NCE is to learn a non-parametric classifier which can distinguish noise and data. The theorem states [20]:

Theorem 3.1. *$J(E_\theta)$ attains a maximum at $-E_\theta(x)^* = \ln p_d$. There are no other extrema if the noise density p_n is chosen such that it is nonzero whenever p_d is nonzero.*

This theorem tells us two key pieces of information about NCE. First of all, the estimator $E_\theta(x)$ here can be defined as an unnormalised model which is different from estimator for MLE, because $E_\theta(x)$ will be "self-normalized" through training of NCE [20]. $E_\theta(x)$ is defined through $E_\theta(x) = -\ln p^0_m(x,\alpha) + c$. The estimator will not only learn the unnormalized model parameters α but learn the normalizing constant c as well. In other words,

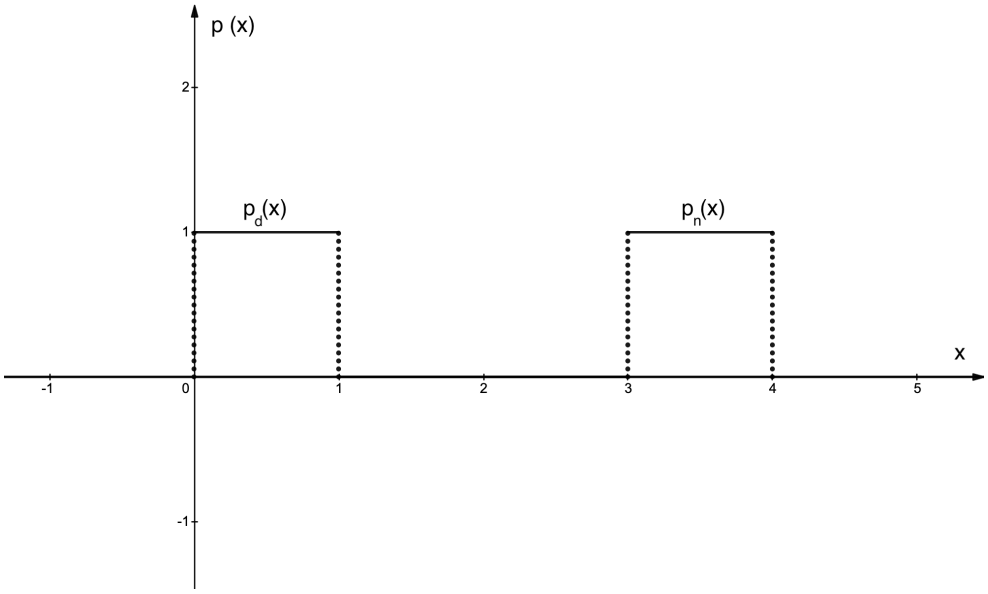

Figure 3.3: p_n is zero when p_d is zero, it is impossible to estimate p_d under this situation.

the NCE estimator is able to learn the value of the partition function. Theoretically, we should split the model parameters into two parts: model parameters α and normalizing term c. However, in practice, setting c as 0 or a fixed value will not affect the performance of NCE [52]. It is believed that the NCE estimator has many free parameters to meet the normalizing constraint during the training process [52]. Therefore, the estimator $E_\theta(x)$ can be defined as an unnormalised model and the optimal energy $-E_\theta(x)^* = \ln p_d$ will be reached when $J(E_\theta)$ reaches maximum.

Secondly, the non-zero condition for noise distribution p_n in this theorem suggests that p_d cannot be estimated when the pdf of noise distribution of data point is zero. For example, when the data distribution is a uniform distribution $x \sim \mathcal{U}(0, 1)$, while the noise data distribution is a uniform distribution $y \sim \mathcal{U}(3, 4)$, as showed in figure 3.3, it is impossible to train NCE.

The non-zero condition can be easily fulfilled by designing a proper noise data distribution. For instance, choosing a Gaussian distribution or a mixture Gaussian model as contrastive noise distribution can guarantee positivity everywhere. However, it is still a need to be cautious when choosing noise. The probability of noise distribution could be extremely small or almost zero at some points far away from the mean, even if Gaussian distribution is positive everywhere, which will also cause the failure of NCE in practice; although theoretically, the NCE estimator should work, for example as shown in figure 3.4.

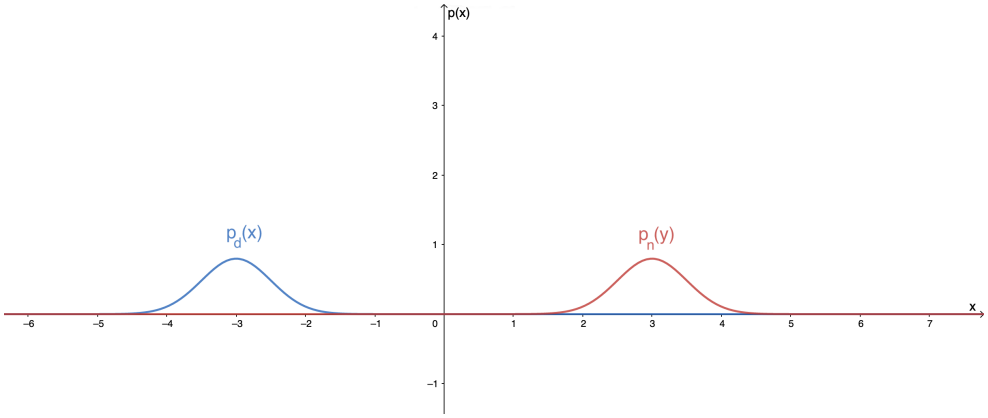

Figure 3.4: $p_d \sim \mathcal{N}(-3, 0.5)$, $p_n \sim \mathcal{N}(3, 0.5)$, Although the positivity condition is satisfied, it is almost impossible to estimate p_d with noise p_n in practice.

This situation will be further discussed in Chapter 4.

3.5.3.2 Consistency

The second theorem which is named as consistency theorem in [22] shows that $\hat{\theta}_T$, which is the value obtained by maximizing J_T, converges to the ground truth value θ^* [20].

Theorem 3.2. *If conditions (a) to (c) are fulfilled then $\hat{\theta}_T$ converges in probability to θ^*, $\hat{\theta}_T \xrightarrow{P} \theta^*$.*

(a) p_n is nonzero whenever p_d is nonzero

(b) $\sup_\theta |J_T(\theta) - J(\theta)| \xrightarrow{P} 0$

(c) The matrix $\mathcal{I}_\nu = \int g(u)g(u)^T P_\nu(u)p_d(u)\mathrm{d}u$ has full rank,

where $g(u) = \ln p_m(u, \theta)|_{\theta=\theta^*}$, $P_\nu(u) = \frac{\nu p_n(u)}{p_d(u)+\nu p_n(u)}$.

3.5.3.3 Asymptotic normality

The third theorem is called asymptotic normality theorem in [22], which describes the distribution of the estimation error $(\hat{\theta}_T - \theta^*)$. The theorem states [20]:

Theorem 3.3. *$\sqrt{T_d}(\hat{\theta}_T - \theta^*)$ is asymptotically normal with mean zero and covariance matrix Σ,*

$$\Sigma = {\mathcal{I}_\nu}^{-1} - (a + \frac{1}{\nu}){\mathcal{I}_\nu}^{-1}\mathbb{E}(P_\nu g)\mathbb{E}(P_\nu g)^T{\mathcal{I}_\nu}^{-1}, \tag{3.31}$$

where $\mathbb{E}(P_\nu g) = \int P_\nu(u)g(u)p_d(u)\mathrm{d}u$.

From the above theorem, several corollaries can be derived as following [22]:

Corollary 3.1. *For large sample sizes T_d, the mean squared error $\mathbb{E}(\|\hat{\theta}_T - \theta^*\|^2)$ equals to $tr(\Sigma)/T_d$.*

From Corollary 3.1, it is obvious that as the sample sizes T_d increase, the accuracy of the estimator will improve. Also from Theorem 3.3, it can be found that both noise distribution p_n and ratio $\nu = T_n/T_d$ have influence on Σ. Σ will affect the accuracy of the estimator. The following question is how p_n and ν will affect the estimator statistically and what are the best options for p_n and ν [20].

Corollary 3.2. *For $\nu \to \infty$, Σ is independent of the choice of p_n and equals*

$$\Sigma = \mathcal{I}^{-1} - \mathcal{I}^{-1}\mathbb{E}(g)\mathbb{E}(g)^T\mathcal{I}^{-1}, \tag{3.32}$$

where $\mathbb{E}(g) = \int g(u)p_d(u)\mathrm{d}u$ and $\mathcal{I} = \int g(u)g(u)^T p_d(u)\mathrm{d}u$.

Specifically, if the noise distribution is fixed as same as data distribution, the following corollary shows the relation between σ and ν [20].

Corollary 3.3. *if $p_n = p_d$, then $\Sigma = (1 + \frac{1}{\nu})(\mathcal{I}^{-1} - \mathcal{I}^{-1}\mathbb{E}(g)\mathbb{E}(g)^T\mathcal{I}^{-1})$.*

Corollary 3.2 tells that as $\nu \to \infty$, Σ is independent from the choice of p_n. According to this corollary, when the estimator is optimal, Σ equals the inverse of FI or the CRLB since the expectation of score $\mathbb{E}(g)$ equals zero. Therefore, NCE can be said to be Fisher-efficient for all choices of p_n when ν is sufficiently large. In other words, the choice of p_n is not important when the ratio of the size of noise and data samples is large enough. In the special case of Corollary 3.3, when the noise distribution equals the data distribution, Σ is equivalent to $1 + \frac{1}{\nu}$ times CRLB. With increasing of ν, Σ is closer to CRLB. However, in practice, restricted to the limited computational resources, it is impossible to load infinite large noise samples, so there exists a natural trade-off between statistical and computational efficiency in NCE.

Although the proof for the optimal choice of noise has not been given in the original research of NCE [22] [21], they gave several suggestions on how to choose a good noise, including:

- Choosing a noise that has an analytical expression of log-pdf;
- Choosing a noise that is easy to sample from;
- Choosing a noise which is close to data noise;
- Making ν as large as possible.

Intuitively, the conclusion in the original NCE papers seems correct. Because one can imagine if p_n is too different from p_d, the classifier of NCE might be too easy to distinguish the data from noise. However, the intuition has been challenged by related works on the choice of optimal noise [6]. In [6], it is reported that the optimal choice noise $p_n^{opt} \propto p_d(x)\|\mathcal{I}^{-1}g(x)\|$ under some conditions. Interestingly, this conclusion is quite similar to the optimal noise in Monte Carlo MLE with Importance Sampling [57]. In both situations, optimal noises are related to the data distribution, scaled by something akin to its natural gradient. However, how to reach the optimal noise and how this formula can assist the training of NCE remains unexplored and unsolved.

In general, considering the above properties of NCE, we know that the optimal noise distribution should be similar or highly related to the data distribution. And when the noise distribution and data distribution are the same, it is possible to control the error in a reasonable range.

3.6 Generative Adversarial Network

GAN is a machine learning framework mainly used as a generative model which consists of two networks, a generative network and a discriminative network which are updated alternatively [16]. The core idea of GAN is to train the generator to "cheat" the discriminator and train the discriminator to distinguish the real data and generate data alternatively. GAN framework has now been researched and applied to many scenarios widely. GAN has both obvious advantages and disadvantages. The disadvantages include the problem of mode collapse, lack of representation of p_g, the synchronization problem of generator and discriminator while training and so on. The advantages include both computational and statistical advantages.

3.6.1 Definition of Generative Adversarial Network

GAN consists of two neural networks. The generative network G maps a latent space vector z to data space x through an implicit distribution p_g, while the discriminative network D maps a sample to a single scalar. Discriminative and generative network are trained through a mini-max game with value function $V(G, D)$:

$$\min_G \max_D V(G, D) = \min_G \max_D \mathbb{E}_{x \sim p_{data}}[\ln D(x)] + \mathbb{E}_{z \sim p_z}[\ln(1 - D(G(z)))]. \tag{3.33}$$

In practice, directly optimizing equation (3.33) to update the generator may not provide sufficient gradient signals. Especially when the generator is far from perfect, the discriminator will distinguish the generated data from the real data too easily. Therefore, a trick named *log trick* is commonly used which is to change the $\ln(1 - D(G(z)))$ term into the $-\ln D(G(z))$ to provide stronger gradient signal in the early stage of training [16]. Another training trick often mentioned is that discriminators should be trained in more steps than training generators. The algorithm of GAN is shown in 3.5.

Algorithm 3.5 Generative Adversarial Network Algorithm

Input: Prior distribution: P_z;
Initialize generator θ_G
Initialize discriminator θ_D
Set $j = 0$
while $j \leq epoch$ **do**
while k steps **do**
Sample $z_1, z_2, ..., z_n$ from$z \sim P_z$
Sample $x_1, x_2, ..., x_n$ from data
Update discriminator θ_D by ascending its stochastic gradient:$\nabla_{\theta_D} \frac{1}{n} \sum_{i=1}^{n} [\ln D(x_i)] +$
$\frac{1}{n} \sum_{i=1}^{n} [\ln(1 - D(G(z_i)))]$
end while
Sample $z_1, z_2, ..., z_n$ from$z \sim P_z$
Update generator θ_G by descending its stochastic gradient:
$\nabla_{\theta_G} \frac{1}{n} \sum_{i=1}^{n} [\ln(1 - D(G(z_i)))]$
$j = j + 1$
end while

3.6.2 Optimality Analysis

The optimality of GAN consists of two parts, the optimality of generator and discriminator respectively. For any given fixed generator G, the optimal discriminator D can be reached as $\frac{p_d}{p_d+p_g}$ [16].

Theorem 3.4. *For fixed generator G, the optimal discriminator D is*

$$D_G^*(x) = \frac{p_d}{p_d + p_g}, \tag{3.34}$$

where, p_d is the underlying probability density of the distribution that produces the dataset, p_g is generated data distribution.

Proof. For the function $\max_y a\ln(y) + b\ln(1-y)$, for $y \in [0,1]$, the function achieves its maximum when and only when $y = \frac{a}{a+b}$. Therefore, for the function:

$$\begin{aligned} V(G,D) &= \mathbb{E}_{x\sim p_d}[\ln D(x)] + \mathbb{E}_{z\sim p_z}[\ln(1-D(G(z)))] \\ &= \mathbb{E}_{x\sim p_d}[\ln D(x)] + \mathbb{E}_{x\sim p_g}[\ln(1-D(x))], \end{aligned} \tag{3.35}$$

the optimal $D(x)$ equals to $\frac{p_d}{p_d+p_g}$. □

Theorem 3.4 tells us that for any generator G, the optimal D can always be reached by $D_G^*(x) = \frac{p_d}{p_d+p_g}$. Then the optimal $D(x)$ can be substituted into the loss function to investigate the optimality of generator.

$$\begin{aligned} V(G,D^*) &= \mathbb{E}_{x\sim p_d}[\ln D^*(x)] + \mathbb{E}_{z\sim p_z}[\ln(1-D^*(G(z)))] \\ &= \mathbb{E}_{x\sim p_d}[\ln D^*(x)] + \mathbb{E}_{x\sim p_g}[\ln(1-D^*(x))] \\ &= \mathbb{E}_{x\sim p_d}[\ln \frac{p_d}{p_d+p_g}] + \mathbb{E}_{x\sim p_g}[\ln\left(1-\frac{p_d}{p_d+p_g}\right)] \\ &= \mathbb{E}_{x\sim p_d}[\ln \frac{p_d}{p_d+p_g}] + \mathbb{E}_{x\sim p_g}[\ln \frac{p_g}{p_d+p_g}], \end{aligned} \tag{3.36}$$

eq.(3.36) can be transformed into a form of JS divergence by:

$$
\begin{aligned}
V(G, D^*) &= \mathbb{E}_{x\sim p_d}[\ln D^*(x)] + \mathbb{E}_{z\sim p_z}[\ln(1 - D^*(G(z)))] \\
&= \mathbb{E}_{x\sim p_d}[\ln \frac{p_d}{p_d + p_g}] + \mathbb{E}_{x\sim p_g}[\ln \frac{p_g}{p_d + p_g}] \\
&= \mathbb{E}_{x\sim p_d}[\ln \frac{p_d}{(p_d + p_g)/2}] - \ln 2 + \mathbb{E}_{x\sim p_g}[\ln \frac{p_g}{(p_d + p_g)/2}] - \ln 2 \\
&= JSD(p_g, p_d) - \ln 4.
\end{aligned}
\tag{3.37}
$$

This transformation suggests that minimizing the $V(G, D^*)$ by updating the generator is equivalent to minimizing JS divergence between p_g and p_d. Because JSD between two distributions is always non-negative, it equals zero when and only when the two distributions are equal. Therefore the minimum of $V(G, D^*)$ holds when and only when $p_g = p_d$ [16].

Theorem 3.5. *The global minimum of $V(G, D^*)$ is reached if and only if $p_g = p_d$. At that point, $V(G, D^*)$ achieves the value - log 4.*

Supposing that the generator and discriminator have enough model capacity, the discriminator is updated first to reach its optimum given G, and the generator is updated according to the current discriminator. Given the new generator, the discriminator will be updated again. Repeating the iteration, p_g will converge to p_d theoretically. It should be mentioned that GAN is an implicit probabilistic model where the distribution of generated samples can be sampled but can not be represented. The generator maps the latent space to sample space directly. p_g in the formulas above is updated by the generator parameter θ instead of directly updating pdf p_g.

3.6.3 Variants of Generative Adversarial Network

There are several essential variants of GAN, including Energy-based GAN, cycle GAN, WGAN, etc. GAN-based algorithms have formed an enormous family in different applications. We will mainly introduce Wasserstein Generative Adversarial Network (WGAN) in this section. Instead of minimizing the JS divergence between p_g and p_d, WGAN changes the objective function with the Wasserstein distance. p-Wasserstein Distance is defined from the theory of transportation as following:

$$
WD_p(P_d, P_g) = (\inf_{\gamma\in\Pi(P_d,P_g)} \mathbb{E}_{(x,y)\sim\gamma(x,y)}[\|x - y\|^p])^{\frac{1}{p}},
\tag{3.38}
$$

where $\Pi(P_d, P_g)$ denotes the set of all possible joint distributions $\gamma(x, y)$whose marginal distributions are P_d, P_g respectively. Intuitively, Wasserstein distance indicates how much "mass" need to be transported from P_d to P_g minimally. The minimum cost of all possible transporting plan is the Wasserstein distance. Compared to other measures between two distributions, Wasserstein distance has obvious advantages. In some situations, it can be a better metric for the distributions. However, it is not easy to calculate the Wasserstein distance directly. It can be calculated through a duality, which is caller *Kantorovich-Rubinstein duality*.

$$WD_1(P_d, P_g) = \sup_{\|f\|_L \leq 1} \mathbb{E}_{x \in P_d}[f(x)] - \mathbb{E}_{x \in P_g}[f(x)], \tag{3.39}$$

where the supremum is over all the 1-Lipschitz functions $f : \mathcal{X} \to \mathbb{R}$.

Therefore as long as we have a parameterized family of 1-Lipschitz functions f, we can have an estimation of Wasserstein distance. The 1-Lipschitz functions condition can be approximated through weight clipping or gradient penalty. With the distance we can construct another GAN framework as following:

Algorithm 3.6 Wasserstein Generative Adversarial Network Algorithm

Input: Prior distribution: P_z;
Initialize generator θ_G
Initialize discriminator θ_D
Set $j = 0$
while $j \leq$ *epoch* **do**
while k steps **do**
Sample $z_1, z_2, ..., z_n$ from$z \sim P_z$
Sample $x_1, x_2, ..., x_n$ from data
Update discriminator θ_D by ascending its stochastic gradient(after weight clipping):$\nabla_{\theta_D} \frac{1}{n} \sum_{i=1}^{n} f(x_i) - \frac{1}{n} \sum_{i=1}^{n} [f(G(z_i))]$
end while
Sample $z_1, z_2, ..., z_n$ from$z \sim P_z$
Update generator θ_G by descending its stochastic gradient: $-\nabla_{\theta_G} \frac{1}{n} \sum_{i=1}^{n} [f(G(z_i))]$
$j = j + 1$
end while

WGAN has been proved having a more stable training process then vanilla GAN and reducing mode collapse. Also WGAN provides a meaningful loss metric that correlates with the generator's convergence and sample quality.

3.7 The Connection between Different Energy-Based Model

In this chapter, we will summarize the connections between different energy-based models and compare the advantages or disadvantages of all methods.

3.7.1 Connection between GAN and NCE

GAN is a generative model which mainly focuses on the quality of generating samples. The estimated distribution of data is not the first concern. However, NCE can be regarded as an EBM whose main purpose is to estimate the data distribution. However, GAN and NCE have a close connection. Both methods are based on a reduction to binary classification. NCE is an energy-based model which is trained to be able to distinguish data samples from noise samples. The discriminator of GAN is designed to distinguish the data samples from generated data.

Both methods are designed with similar objective functions for training. For a GAN model, when the generator is fixed, the objective function of updating the discriminator is similar to an NCE model. The objective function for updating the discriminator when given an arbitrary generator G, can be written as follows:

$$V(G, D) = \mathbb{E}_{x\sim p_d}[\ln D(x)] + \mathbb{E}_{z\sim p_z}[\ln(1 - D(G(z)))] \tag{3.40}$$

It can also be written as the following form when the discriminator reaches its optimal:

$$\begin{aligned} V(G, D^*) &= \mathbb{E}_{x\sim p_d}[\ln D^*(x)] + \mathbb{E}_{z\sim p_z}[\ln(1 - D^*(G(z)))] \\ &= \mathbb{E}_{x\sim p_d}[\ln \frac{p_d}{p_d + p_g}] + \mathbb{E}_{x\sim p_g}[\ln \frac{p_g}{p_d + p_g}], \end{aligned} \tag{3.41}$$

The objective function of NCE is can be written as a similar form:

$$J(p_m) = \mathbb{E}_{x\sim p_d} \ln\left(\frac{p_m(x)}{p_m(x)+p_n(x)}\right) + \mathbb{E}_{x\sim p_n} \ln\left(\frac{p_n(x)}{p_m(x)+p_n(x)}\right) \tag{3.42}$$

Comparing both objective functions, we can clearly see that the idea of GAN partly inherits from NCE. The term $\ln\left(\frac{p_m(x)}{p_m(x)+p_n(x)}\right)$ in NCE is similar to the discriminator network in GAN. Since GAN does not have an explicit representation of data distribution, the discriminator is defined as a whole neural network, which is parameterized by D_θ. However, in NCE, the discriminator compares the model and noise distribution and a sigmoid layer. As showed in figure 3.5, both discriminators of GAN and NCE map the samples from data space to a scalar but the term of $\ln\left(\frac{p_m(x)}{p_m(x)+p_n(x)}\right)$ in NCE can be regarded as a discriminator with a special form, which is parameterized by energy model E_θ, where $p_m(x) = \frac{e^{-E_\theta(x)}}{Z_\theta}$. In other words, compared to GAN, NCE trains an internal data model that belongs to part of the discriminator network with a fixed generator network.

3.7.2 Connections between MLE and NCE

Both MLE and NCE are important methods of density estimation or EBM in unsupervised learning. Although there are no directly connection between MLE and NCE and objectives of MLE and NCE are different, both of them are asymptotically consistent. It has been proved that under certain conditions that the gradient of MLE and NCE are equivalent to each other. First, if $\nu \to \infty$, the gradient of NCE is equivalent to MLE gradient:

According to eq.(3.6) and (3.7), the gradient of MLE can be expressed as following:

$$\begin{aligned}
\frac{\partial}{\partial\theta}\mathbb{E}_{x\sim p_d}[\ln p_\theta(x)] &= \frac{\partial}{\partial\theta}\mathbb{E}_{x\sim p_d}[\ln p_\theta(x)] - \frac{\partial}{\partial\theta}\mathbb{E}_{x\sim p_\theta}[\ln p_\theta(x)] \\
&= \sum_x \frac{\partial}{\partial\theta}[\ln p_\theta(x)]p_d(x) - \sum_x \frac{\partial}{\partial\theta}[\ln p_\theta(x)]p_\theta(x) \\
&= \sum_x [p_d(x) - p_\theta(x)]\frac{\partial \ln p_\theta(x)}{\partial\theta}.
\end{aligned} \tag{3.43}$$

The gradient of NCE can be expressed as following:

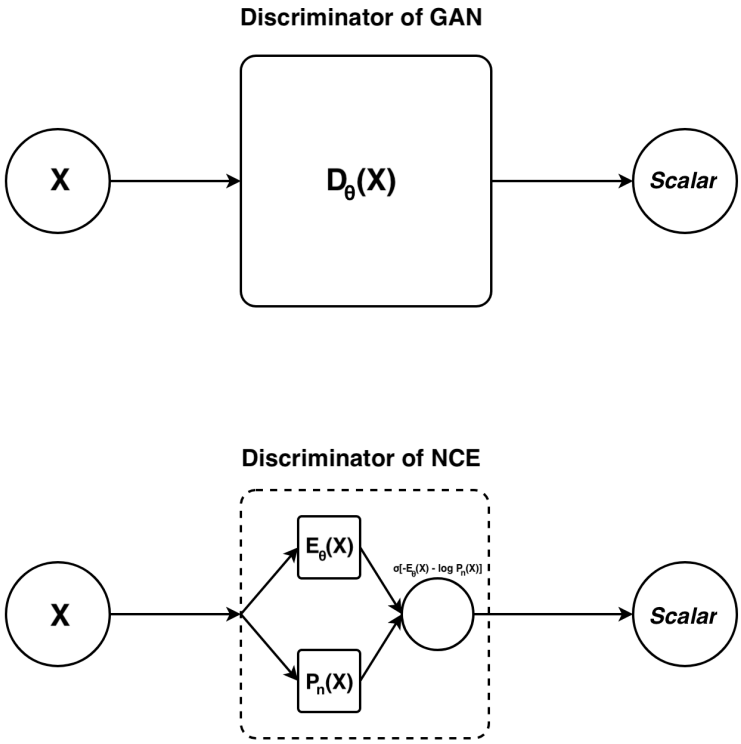

Figure 3.5: The figure shows the structures of discriminators of GAN and NCE. NCE trains an internal data model which belongs to the discriminator network with a fixed generator network.

$$
\begin{aligned}
\frac{\partial J(p_\theta)}{\partial \theta} &= \frac{\partial}{\partial \theta}\mathbb{E}_{x\sim p_d}\ln\left(\frac{p_\theta(x)}{p_\theta(x)+\nu p_n(x)}\right) + \frac{\partial}{\partial \theta}\nu\mathbb{E}_{x\sim p_n}\ln\left(\frac{\nu p_n(x)}{p_\theta(x)+\nu p_n(x)}\right) \\
&= \mathbb{E}_{x\sim p_d}[\frac{\nu p_n(x)}{p_\theta(x)+\nu p_n(x)}\frac{\partial \ln p_\theta(x)}{\partial \theta}] - \nu\mathbb{E}_{x\sim p_n}[\frac{p_\theta(x)}{p_\theta(x)+\nu p_n(x)}\frac{\partial \ln p_\theta(x)}{\partial \theta}] \\
&= \sum_x \frac{\nu p_n(x)}{p_\theta(x)+\nu p_n(x)}\frac{\partial \ln p_\theta(x)}{\partial \theta}\cdot p_d(x) - \nu\sum_x \frac{p_\theta(x)}{p_\theta(x)+\nu p_n(x)}\frac{\partial \ln p_\theta(x)}{\partial \theta}\cdot p_n(x) \\
&= \sum_x \frac{\nu p_n(x)}{p_\theta(x)+\nu p_n(x)}[p_d(x)-p_\theta(x)]\frac{\partial \ln p_\theta(x)}{\partial \theta}.
\end{aligned}
\tag{3.44}
$$

When $\nu \to \infty$, $\frac{\nu p_n(x)}{p_\theta(x)+\nu p_n(x)} \to 1$, then the gradient of NCE is equivalent to gradient of MCMCML. This conclusion gives a similar result as Corollary 3.2, which states that when $\nu \to \infty$, the estimator has same asymptotic normality as NCE.

In addition to this situation, if noise distribution of NCE is non-stationary, that is, the noise distribution in each epoch is chosen as a copy of the model result from the last epoch, the gradient of NCE is equivalent to the MLE [15]. This kind of NCE is called Self-contrastive Estimation (SCE) in [15]. The gradient of SCE is, following eq.(3.44):

$$
\begin{aligned}
\frac{\partial J(p_\theta)}{\partial \theta} &= \sum_x \frac{\nu p_n(x)}{p_\theta(x)+\nu p_n(x)}[p_d(x)-p_\theta(x)]\frac{\partial \ln p_\theta(x)}{\partial \theta} \\
&= \sum_x \frac{\nu p_n^\theta(x)}{p_\theta(x)+\nu p_n^\theta(x)}[p_d(x)-p_\theta(x)]\frac{\partial \ln p_\theta(x)}{\partial \theta} \\
&= \sum_x \frac{\nu}{1+\nu}[p_d(x)-p_\theta(x)]\frac{\partial \ln p_\theta(x)}{\partial \theta},
\end{aligned}
\tag{3.45}
$$

where $p_n = p_n^\theta$, and p_n^θ is copied from p_θ but independent from θ. the gradient of SCE is also equivalent to MLE. The $\frac{\nu}{1+\nu}$ can be folded into the learning rate.

In conclusion, NCE is equivalent to MLE under certain conditions.

3.7.3 Connections between SM and NCE

As reported in [70], SSM can be regarded as a a special NCE where the noise distribution p_n is chosen as model distribution perturbed by a small vector v $p_\theta(x-v)$. Thus NCE objective function will be:

$$J(p_\theta) = \mathbb{E}_{x\sim p_d} \ln\left(\frac{p_\theta(x)}{p_\theta(x) + \nu p_\theta(x-v)}\right) + \nu \mathbb{E}_{x\sim p_n} \ln\left(\frac{\nu p_\theta(x-v)}{p_\theta(x) + \nu p_\theta(x-v)}\right), \tag{3.46}$$

the objective function can be proved to have the following equivalent form by Taylor expansion [71]:

$$\arg\min_\theta \frac{1}{4}\mathbb{E}_{x\sim p_d}[-\sum_{i=1}^{d}\sum_{j=1}^{d}\frac{\partial^2 E_\theta(x)}{\partial x_i \partial x_j}v_i v_j + \frac{1}{2}\sum_{i=1}^{d}(\frac{\partial E_\theta(x)}{\partial x_i}v_i)^2] + \ln 4 + o(\|v\|_2^2), \tag{3.47}$$

which is similar to the objective function of SSM.

3.8 Summary

In this chapter, we reviewed the definition of EBM and gave a comprehensive survey on some important modern methods in designing EBMs, which more or less have internal connections with each others. All these methods have difficulties in training. For example, for MLE-based methods, it is troublesome to estimate the partition function. Although MCMC provides a solution, accurately sampling from a complex distribution is still challenging. The adversarial training avoids the problem of sampling by constructing a generator, however, the algorithm seems too redundant for an EBM. Because it involves triple nested loops and the innermost loop actually is an EBM itself (MINE or Entropic Estimation). SM is another important family of EBM, but it is usually consumes more time than EBM and NCE. NCE is reported as the fastest methods for an unnormalised energy model [22], and it has the effect of self-normalizing. In practice, we found several obstacles in training a NCE, which will be discussed in Chapter 4.

Chapter 4

The Obstacles of Training in Noise-Contrastive Estimation

Although NCE is an effective estimator for EBM, there are some obstacles when training the estimator. There are three hyper-parameters in NCE which will influence the training in NCE, including the number of data samples N, the choice of noise distribution p_n and the portion of noise samples compared to data samples ν. All these parameters have an impact on the performance of the estimator. As stated in the original paper of NCE, it is believed that if noise distribution is closer to the data distribution, the result of the NCE estimator will be better. Although, in other research, it is shown the best noise choice is related to both data distribution and the natural gradient of data distribution when the estimator is optimal, the best noise is still highly related to data distribution. However, during the practical experiments of NCE, it is found that choosing the noise the same as data distribution does not necessarily yield an ideal output. Firstly, as shown in figure 4.1, we set up an eight mixture Gaussian estimation task for NCE. Given different fixed noise, we track the change of the unnormalized energy model, which is modelled by a neural network in different epochs. Some phenomena are not as expected as the theoretical results. When the noise distribution is exactly the same as the underlying data distribution, the estimator is difficult or very slow to converge. As the Gaussian distributions' variance increases, the estimator converges faster. Secondly, as shown in figure 4.2, if the noise of NCE is chosen as an 8 Gaussian mixture model with a larger radius, the estimator will converge faster; however, the result of the estimator is not satisfying. This experiment suggests that there may exist a trade-off between computational and statistical efficiency.

An elementary conclusion can be drawn from the two experiments. The choice of noise distribution could strongly impact the statistical and computational performance of NCE

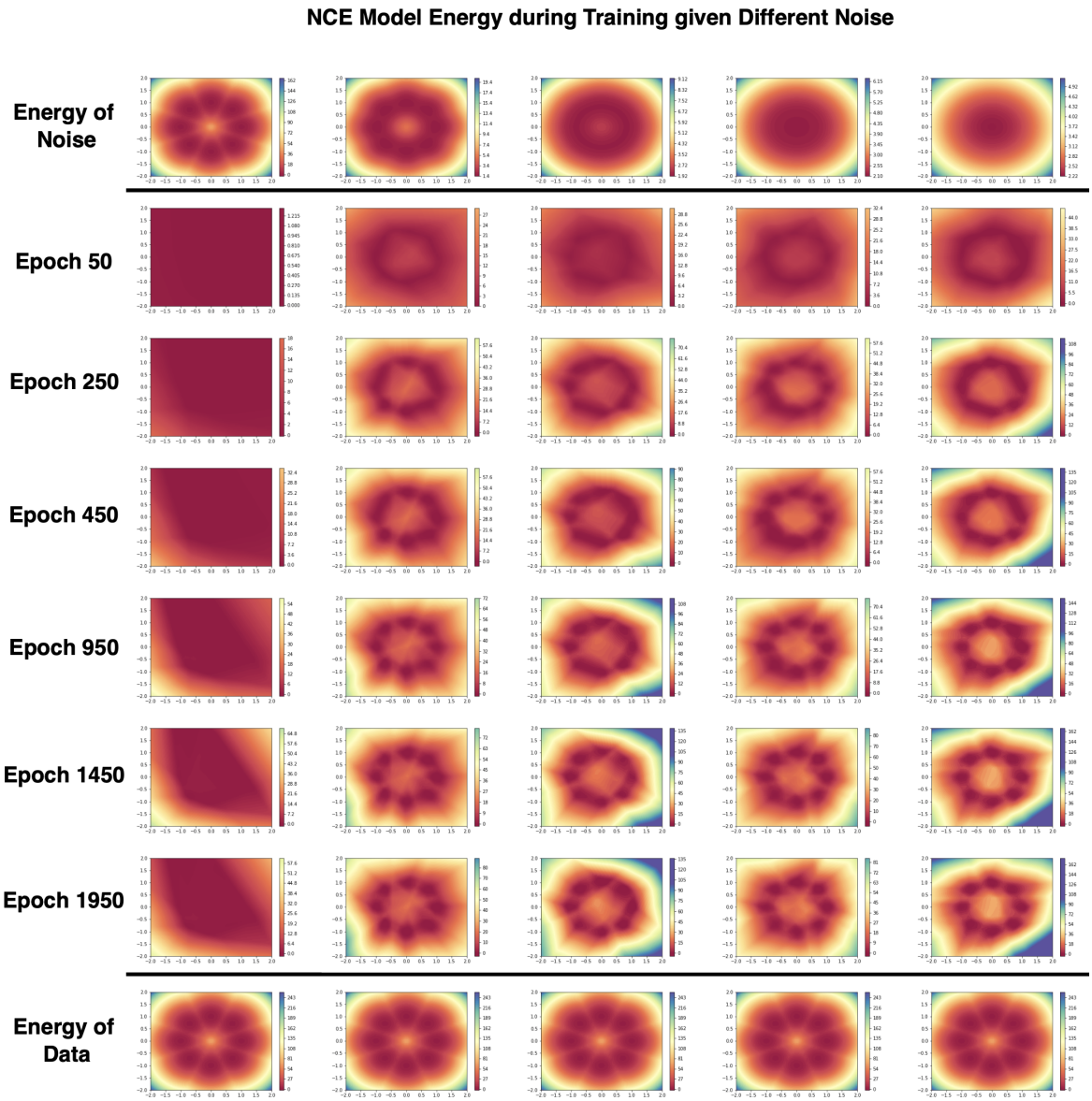

Figure 4.1: An experiment of NCE with 8 Mixture Gaussian toy data.

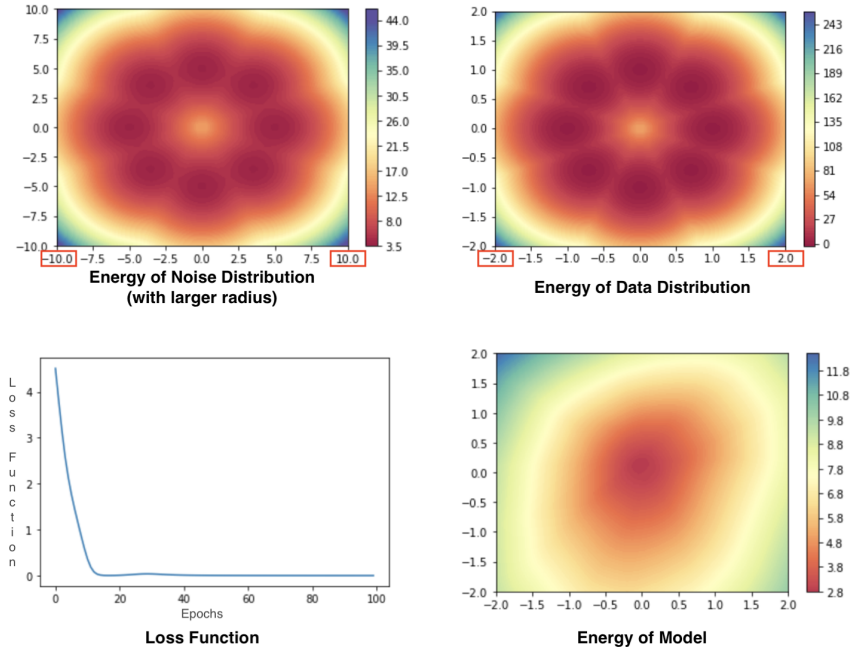

Figure 4.2: Another experiment of NCE with 8 Mixture Gaussian toy data. The noise distribution has a larger radius than the data distribution. The estimator converges fast, but the result is not satisfying.

when ν is limited. There could exist a trade-off between them. We will use an example of one-dimensional Gaussian estimation to decompose and investigate the obstacles in the training of NCE. Specifically, we summarize three difficulties which affect NCE training.

4.1 Trade-off between Computation and Accuracy

The first problem of NCE training is a trade-off between computational resources and accuracy. According to the theorems in Chapter 3. The NCE estimator will attain CRLB when the number of data samples is large enough and the portion of noise data is large enough. Given an arbitrary noise distribution, in order to achieve a better statistical result, more noise data need to be sampled, which will result in more computational costs.

Here we consider a one-dimensional Gaussian estimation task. Suppose that data are drawn randomly from a one dimensional Gaussian distribution, $p_d \sim \mathcal{N}(0, 0.5^2)$. Noise distribution is also a one-dimensional Gaussian distribution $p_n \sim \mathcal{N}(0, 0.5^2)$. Energy model is defined as $E(x)$, whose distribution is $P_m \sim \mathcal{N}(0, \sigma^2)$, $P_m(x) = \frac{e^{-E(x)}}{\int_{x \in \mathcal{X}} e^{-E(x)}}$. By changing the number of data samples and the portion of noise data samples, different results of NCE estimators under different situations can be achieved. After repeating the same experiments 50 times, the mean square error between θ and data standard variance is calculated.

As shown in figure 4.3, the figure indicates the log 10 mean square error after repeating experiments 50 times, with respect to the number of data samples. The red line shows the situation when ν is 1, which means the proportion of data sample and noise samples is 1:1. The green line shows the situation when ν is 5, which means the proportion of data sample and noise samples is 1:5. The green line shows the situation when ν is 50, which means the proportion of data sample and noise samples is 1:50.

With the increase in the number of data samples and the increase of the portion of noise data, the estimator will have a smaller mean squared error. The performance of the estimator is the same as expected. The NCE estimator is more accurate when ν and N increase, but at the same time, more samples in the training process will increase the training time of NCE. Although the increase of the computational cost is not obvious in the low dimensional task, it will largely increase when the dimension is high. For instance, when the data dimension is 784, loading and computing 32 data samples and 32 noise samples will take significantly less time than 32 data samples and 320 noise samples in a batch.

Therefore, although compared to other unnormalized models, such as score matching and contrastive divergence, NCE is believed to have advantages in computation cost, it

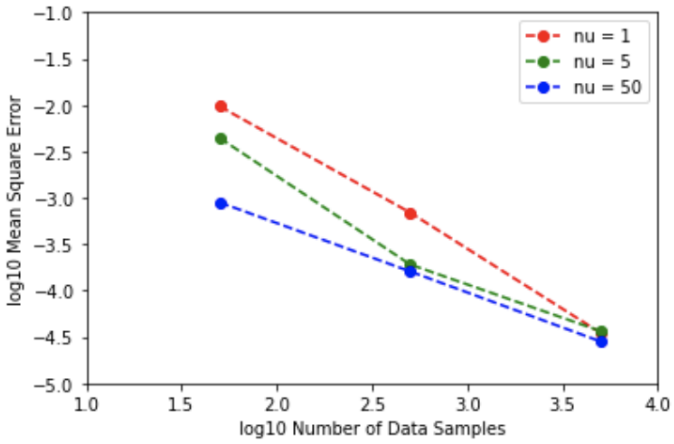

Figure 4.3: Estimation Accuracy. The figure indicates the log 10 mean square error with respect to the number of data samples. The red line shows the situation when ν is 1, which means the proportion of data sample and noise samples is 1:1. The green line shows the situation when ν is 5, which means the proportion of data sample and noise samples is 1:5. The green line shows the situation when ν is 50, which means the proportion of data sample and noise samples is 1:50. The X-axis shows the numbers of training data.

could also be computationally costly if the portion of noise data increase. Then when the computational resources are limited, the choice of noise should be considered seriously.

4.2 Density Chasm

4.2.1 Definition of Density Chasm

The second obstacle of training an NCE is called *density chasm*. The density chasm is firstly described in the classification tasks [62]. The density chasm is usually meant to describe the phenomenon of estimator failure due to the huge gap between data distribution and noise distribution. Usually, NCE can achieve a decent result when the KL divergence of the data distribution and noise distribution is small [47]. However, the estimator could be extremely inaccurate or even fail when the KL divergence of data distribution and noise distribution is sufficiently large. Theoretically, we already know when the data samples N are sufficiently large, according to Theorem 3.1, no matter what choice of noise distribution is, the estimator will always converge to the energy of data. However, practically, it is

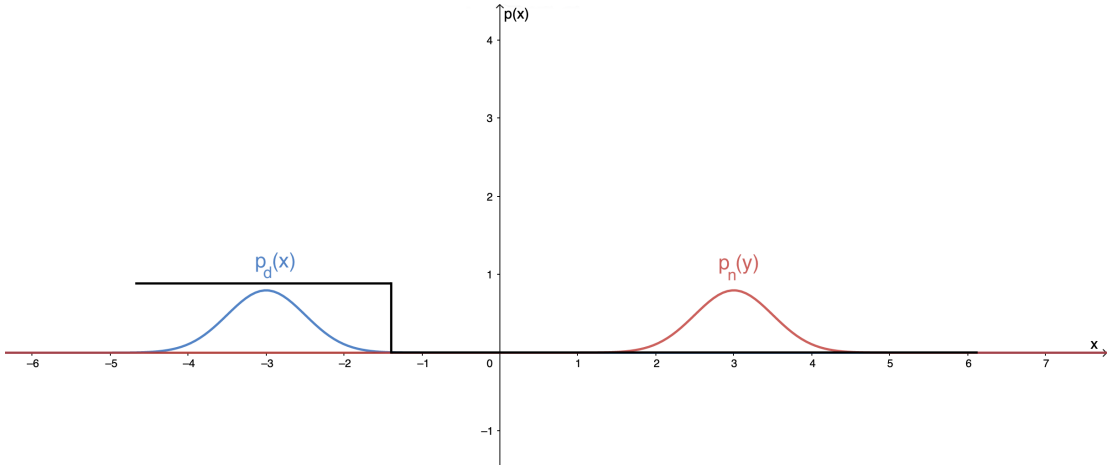

Figure 4.4: The black line could easily distinguish the difference between data and noise, but it is far from the distribution of data.

impossible to have infinitely large samples. In addition, it is difficult to directly find a noise distribution which has a sufficiently small KL divergence with the data distribution. In this situation, if a "bad" noise distribution is chosen, the estimator may never find the correct answer.

For instance, when both data distribution and noise distribution are Gaussian distributions. The noise distribution corresponds to a Gaussian distribution with a mean of 3 and a standard variance of 0.5, while the data distribution is defined as with a mean of -3 and a standard variance of 0.5. As shown in figure 4.4, intuitively, it is easy to find energy which can easily distinguish the sample data between noise data, which will cause the loss function to fail to go down anymore.

Figure 4.5 and figure 4.6 give us a hint about when and how the density chasm between data and noise will affect the NCE estimator. The experiment is conducted by randomly choosing ten samples for both data and noise. The energy is constructed by a simple three layers neural network, and NCE will be updated for ten epochs. The figure shows how the energy changes in these ten epochs. From the figure, it can be found that when the sample and noise data are not overlapped, the energy model is difficult to learn the distribution of data, but for the area where sample and noise data are overlapped, it has an expected "pulling up" and "pushing down" effect.

The above experiment provides an intuitive understanding of density chasm; however, its formal definition has never been mentioned. In the paper which mentioned density chasm first [62], it suggests that $KL(p_d \| p_n)$ could be the measure of density chasm. Es-

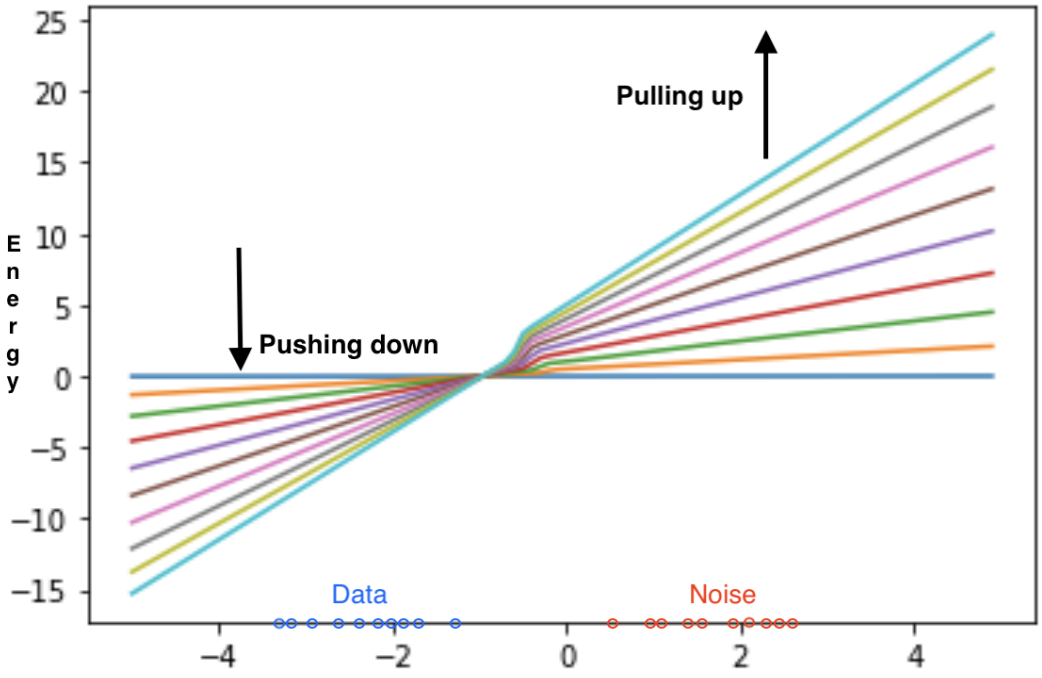

Figure 4.5: The blue and red dots are the samples of data and noise. Each group consists of ten points. The lines indicate the change during the training of energy. The colours are blue, orange, green, red, purple, brown, pink, grey, yellow and cyan. The energy will be pushed down where blue dots are located while pushing up where red dots are located. However, the energy will not be learnt correctly because blue dots and reds dots are not overlapped. This is the situation of density chasm affecting the performance of the energy estimator.

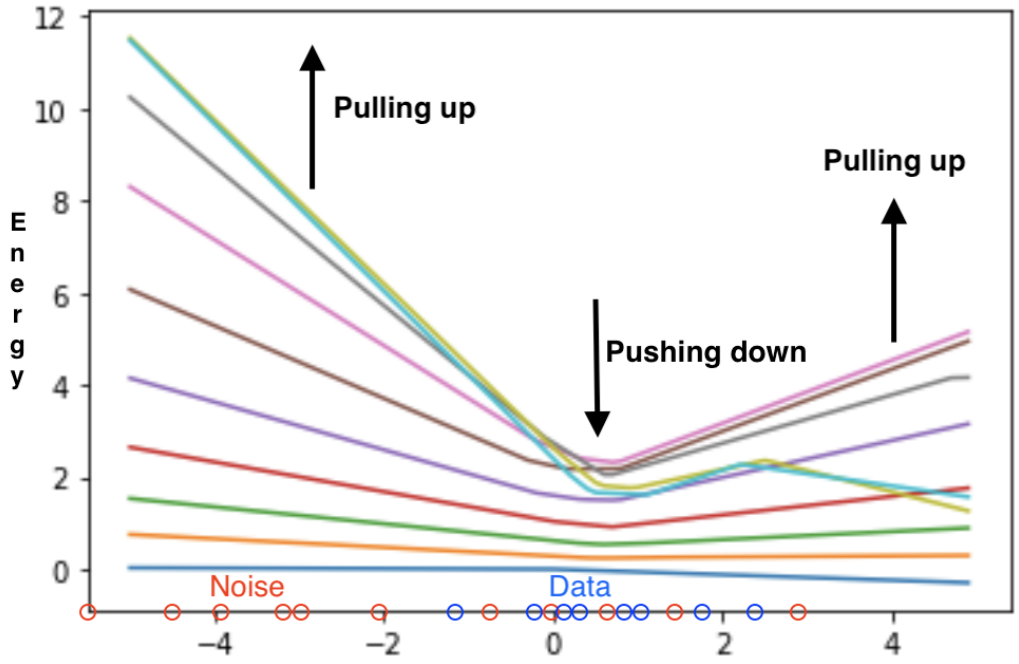

Figure 4.6: The blue and red dots are the samples of data and noise. Each group consists of ten points. The lines indicate the change during the training of energy. The colours are blue, orange, green, red, purple, brown, pink, grey, yellow and cyan. The energy will be pushed down where blue dots are located while pushing up where red dots are located. However, the energy will be learnt correctly because blue dots and reds dots are overlapped. This is the ideal situation for noise.

pecially, the NCE estimator will be failed, when the $KL(p_d\|p_n)$ is larger than 20 nats according to the empirical results [47]. However, it needs to specify that although one can use $KL(p_d\|p_n)$ as a uniformed metric to define the density chasm, the performance of the estimator could behave differently according to different sets of noise data. We argue that the KL divergence is not an accurate measure of the goodness of the noise choice by giving the following empirical experiments.

Taking two one-dimensional Gaussian distributions as example, the KL divergence can be expressed as (4.1). The KL divergence will have different performances when different parameters dominate the value of KL divergence. For example, the KL divergence is large, when σ_n is sufficiently large, or μ_n is sufficiently large. We argue that the result of NCE will behave much differently given different σ_n and μ_n under the same value of KL divergence.

$$KL(p_d\|p_n) = \ln \frac{\sigma_n}{\sigma_d} + \frac{\sigma_d^2 + (\mu_d - \mu_n)^2}{2\sigma_n^2} - \frac{1}{2}. \tag{4.1}$$

Suppose the data distribution is known. The experiment is to use an energy model $E_\theta(x)$, which is from the same parametric family as the data. The model is defined as normalized distribution $P_m \sim \mathcal{N}(\mu_\theta, \sigma_\theta^2)$, $P_m(x) = \frac{e^{-E_\theta(x)}}{\int_{x\in\mathcal{X}} e^{-E_\theta(x)}}$, which contains two parameters μ_θ and σ_θ. Data are drawn randomly from a one-dimensional Gaussian distribution, $p_d \sim \mathcal{N}(0.0, 1.0^2)$. According to eq.(4.1), the relation between $\log_{10}\|\mu_\theta\|$ and $\log_{10}\sigma_\theta$ of noise, when the KLD between data and noise is fixed is plotted as figure 4.7.

Given the fixed KLD, we compare the different performances of the estimator with different μ and σ. The results are listed in table 4.1. For example, when the noise distribution is $p_{n1} \sim \mathcal{N}(0, 14655478^2)$, $KL(p_d\|p_{n1}) \approx 16$, when the noise distribution is $p_{n2} \sim \mathcal{N}(5.657, 1.0^2)$, $KL(p_d\|p_{n2}) \approx 16$. However, the performance of the two estimators is different, as shown in Figure 4.8. The estimator with p_{n1} has a better performance than the estimator with p_{n2} when the number of data, and initialization of the energy model are exactly the same. Similar results also happen in other pairs of noises: when the KLD is the same, the noise distribution with a large variance and a same mean with data will yield a better estimation than the noise that has the same variance and a different mean with data.

Comparing two extreme situations in Figure 4.8, both noises have the same KL divergence, but the difference in mean and variance of the noise will change the behaviour of the estimator. Choosing a noise which has little bias on mean and large variance will lead to a better estimation result. On the contrary, a slight bias in choosing the mean of noise will lead to a larger bias in the estimation when the variance is small. In other words,

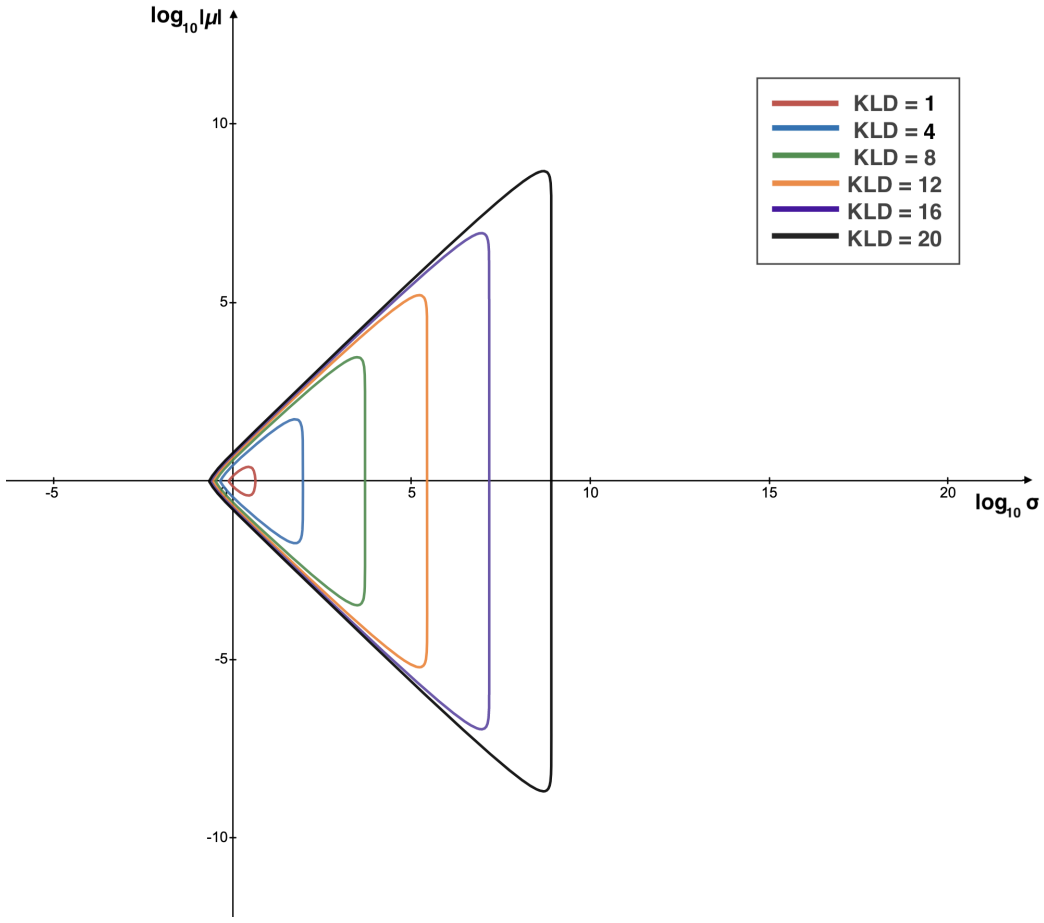

Figure 4.7: The figure shows the relation between $\log_{10} \|\mu_\theta\|$ and $\log_{10} \sigma_\theta$ of noise, when the KLD between data $p_d \sim \mathcal{N}(0.0, 1.0^2)$ and noise is fixed.

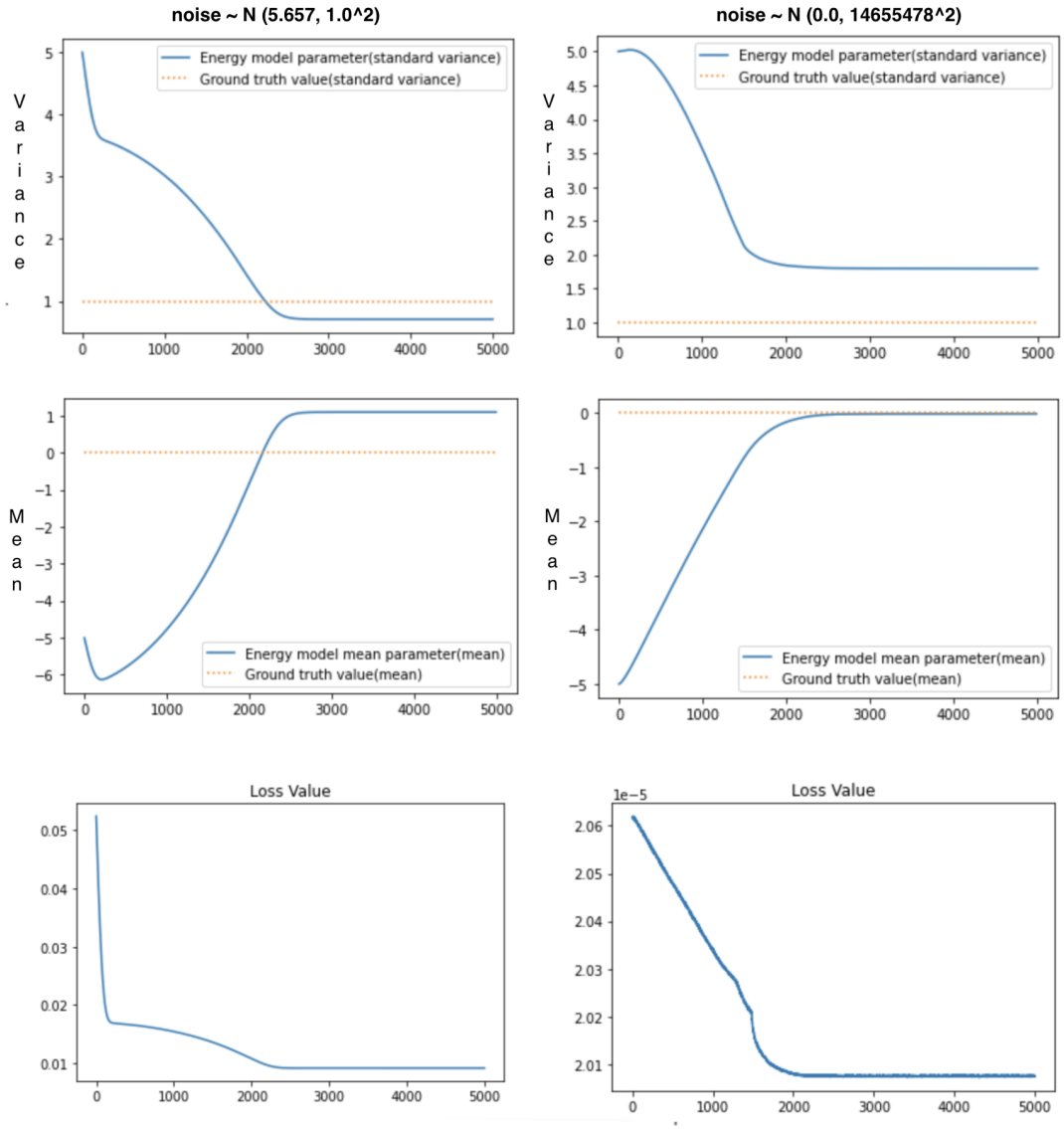

Figure 4.8: The figure shows the change of energy parameters σ_θ, μ_θ during the NCE training with the noise distribution of $p_{n1} \sim \mathcal{N}(0, 14655478^2)$ and $p_{n2} \sim \mathcal{N}(5.657, 1.0^2)$ respectively.

Data Distribution	Noise Distribution	KLD	MSE(σ_θ)	MSE(μ_θ)
$\mathcal{N}(0.0, 1.0^2)$	$\mathcal{N}(1.414, 1.0^2)$	1	0.01076	0.01015
$\mathcal{N}(0.0, 1.0^2)$	$\mathcal{N}(0.0, (10^{0.64})^2)$	1	0.002263	0.000030061
$\mathcal{N}(0.0, 1.0^2)$	$\mathcal{N}(2.828, 1.0^2)$	4	0.01910	0.05297
$\mathcal{N}(0.0, 1.0^2)$	$\mathcal{N}(0.0, (10^{1.954})^2)$	4	0.003082	0.0008922
$\mathcal{N}(0.0, 1.0^2)$	$\mathcal{N}(4.0, 1.0^2)$	8	0.1619	0.48700
$\mathcal{N}(0.0, 1.0^2)$	$\mathcal{N}(0.0, (10^{3.692})^2)$	8	0.03383	0.002947
$\mathcal{N}(0.0, 1.0^2)$	$\mathcal{N}(4.899, 1.0^2)$	12	0.1855	0.9475
$\mathcal{N}(0.0, 1.0^2)$	$\mathcal{N}(0.0, (10^{5.429})^2)$	12	0.105294	0.002840
$\mathcal{N}(0.0, 1.0^2)$	$\mathcal{N}(5.657, 1.0^2)$	16	0.3467	2.1905
$\mathcal{N}(0.0, 1.0^2)$	$\mathcal{N}(0.0, (10^{7.166})^2)$	16	0.72772	0.004389

Table 4.1: Accuracy of NCE estimators with different noises.

given the same KLD between data and noise, choosing a noise that has a larger variance is better than the noise that has a smaller variance. This experiment first tells us that the KLD between data and noise is not an absolute standard to measure how good or bad noise is. Secondly, it provides us with a hint about how to choose a noise which is good for NCE, when we have limited information about the data distribution.

4.2.2 Analysis of Density Chasm

4.2.2.1 Gradient of Loss Function

We continue to take one-dimension Gaussian distribution estimation as an example. Suppose we have a data distribution which has a known distribution as $P_d \sim \mathcal{N}(0, \sigma^{*2})$, and the noise distribution is chosen as a same parametric family distribution with the same mean and different variance, $P_n \sim \mathcal{N}(\mu_n, \sigma_n^2)$. And the energy model $E_\theta(x)$, its distribution $P_m \sim \mathcal{N}(\mu_\theta, \sigma_\theta^2)$, $P_m(x) = \frac{e^{-E_\theta(x)}}{\int_{x\in\mathcal{X}} e^{-E_\theta(x)}}$. According to the NCE loss function:

$$\begin{aligned} L(x) &= -\mathbb{E}_{x\sim P_d} \ln(S(\ln P_m(x) - \ln P_n(x))) - \mathbb{E}_{x\sim P_n} \ln(1 - S(\ln P_m(x) - \ln P_n(x))) \\ &= -\mathbb{E}_{x\sim P_d} \ln\left(S(\ln \frac{P_m(x)}{P_n(x)})\right) - \mathbb{E}_{x\sim P_n} \ln\left(1 - S(\ln \frac{P_m(x)}{P_n(x)})\right), \end{aligned} \tag{4.2}$$

$$S(\cdot) = \frac{1}{1+e^{-x}}, \tag{4.3}$$

$$S(\ln \frac{P_m(x)}{P_n(x)}) = \frac{P_m(x)}{P_m(x)+P_n(x)}. \tag{4.4}$$

According to the connection of NCE and supervised learning:

$$\begin{aligned} L(x) &= -\mathbb{E}_{x\sim P_d} \ln\left(\frac{P_m(x)}{P_m(x)+P_n(x)}\right) - \mathbb{E}_{x\sim P_n} \ln\left(1 - \frac{P_m(x)}{P_m(x)+P_n(x)}\right)) \\ &= -\mathbb{E}_{x\sim P_d} \ln\left(\frac{P_m(x)}{P_m(x)+P_n(x)}\right) - \mathbb{E}_{x\sim P_n} \ln\left(\frac{P_n(x)}{P_m(x)+P_n(x)}\right)) \end{aligned} \tag{4.5}$$

Substitute the P_m and P_n with Gaussian distribution. Then the following can be obtained:

$$\begin{aligned} \frac{P_m(x)}{P_m(x)+P_n(x)} &= \frac{\frac{1}{\sigma_\theta\sqrt{2\pi}} e^{-\frac{(x-\mu_\theta)^2}{2\sigma_\theta^2}}}{\frac{1}{\sigma_\theta\sqrt{2\pi}} e^{-\frac{(x-\mu_\theta)^2}{2\sigma_\theta^2}} + \frac{1}{\sigma_n\sqrt{2\pi}} e^{-\frac{(x-\mu_n)^2}{2\sigma_n^2}}} \\ &= \frac{1}{1+\frac{\sigma_\theta}{\sigma_n} e^{\frac{(x-\mu_\theta)^2}{2\sigma_\theta^2} - \frac{(x-\mu_n)^2}{2\sigma_n^2}}}, \end{aligned} \tag{4.6}$$

$$\begin{aligned} \frac{P_n(x)}{P_m(x)+P_n(x)} &= \frac{\frac{1}{\sigma_n\sqrt{2\pi}} e^{-\frac{(x-\mu_n)^2}{2\sigma_n^2}}}{\frac{1}{\sigma_\theta\sqrt{2\pi}} e^{-\frac{(x-\mu_\theta)^2}{2\sigma_\theta^2}} + \frac{1}{\sigma_n\sqrt{2\pi}} e^{-\frac{(x-\mu_n)^2}{2\sigma_n^2}}} \\ &= \frac{1}{1+\frac{\sigma_n}{\sigma_\theta} e^{\frac{(x-\mu_n)^2}{2\sigma_n^2} - \frac{(x-\mu_\theta)^2}{2\sigma_\theta^2}}}, \end{aligned} \tag{4.7}$$

$$L(x, \sigma_\theta, \mu_\theta) = \mathbb{E}_{x\sim P_d} \ln\left(1 + \frac{\sigma_\theta}{\sigma_n} e^{\frac{(x-\mu_\theta)^2}{2\sigma_\theta^2} - \frac{(x-\mu_n)^2}{2\sigma_n^2}}\right) + \mathbb{E}_{x\sim P_n} \ln\left(1 + \frac{\sigma_n}{\sigma_\theta} e^{\frac{(x-\mu_n)^2}{2\sigma_n^2} - \frac{(x-\mu_\theta)^2}{2\sigma_\theta^2}}\right). \tag{4.8}$$

In order to optimize through loss function (4.8), the gradient of the $L(x, \sigma_\theta, \mu_\theta)$, with respect to σ_θ and μ_θ is calculated as following:

$$
\begin{aligned}
\frac{\partial L(x,\sigma_\theta,\mu_\theta)}{\partial \sigma_\theta} &= \mathbb{E}_{x\sim P_d}\frac{\partial \ln\left(1+\frac{\sigma_\theta}{\sigma_n}e^{\frac{(x-\mu_\theta)^2}{2\sigma_\theta^2}-\frac{(x-\mu_n)^2}{2\sigma_n^2}}\right)}{\partial \sigma_\theta} + \mathbb{E}_{x\sim P_n}\frac{\partial \ln\left(1+\frac{\sigma_n}{\sigma_\theta}e^{\frac{(x-\mu_n)^2}{2\sigma_n^2}-\frac{(x-\mu_\theta)^2}{2\sigma_\theta^2}}\right)}{\partial \sigma_\theta} \\
&= \mathbb{E}_{x\sim P_d}\frac{\left(\sigma_\theta - x + \mu_\theta\right)\left(\sigma_\theta + x - \mu_\theta\right)\mathrm{e}^{\frac{(x-\mu_\theta)^2}{2\sigma_\theta^2}}}{\sigma_\theta^2\cdot\left(\sigma_\theta \mathrm{e}^{\frac{(x-\mu_\theta)^2}{2\sigma_\theta^2}} + \sigma_n \mathrm{e}^{\frac{(x-\mu_n)^2}{2\sigma_n^2}}\right)} \\
&- \mathbb{E}_{x\sim P_n}\frac{\sigma_n \mathrm{e}^{\frac{(x-\mu_n)^2}{2\sigma_n^2}}\cdot\left(\sigma_\theta - x + \mu_\theta\right)\left(\sigma_\theta + x - \mu_\theta\right)}{\sigma_\theta^3\cdot\left(\sigma_\theta \mathrm{e}^{\frac{(x-\mu_\theta)^2}{2\sigma_\theta^2}} + \sigma_n \mathrm{e}^{\frac{(x-\mu_n)^2}{2\sigma_n^2}}\right)} \\
&= \mathbb{E}_{x\sim P_d}\frac{\left(\sigma_\theta - x + \mu_\theta\right)\left(\sigma_\theta + x - \mu_\theta\right)\mathrm{e}^{\frac{(x-\mu_\theta)^2}{2\sigma_\theta^2}}}{\sigma_\theta^2\cdot\left(\sigma_\theta \mathrm{e}^{\frac{(x-\mu_\theta)^2}{2\sigma_\theta^2}} + \sigma_n \mathrm{e}^{\frac{(x-\mu_n)^2}{2\sigma_n^2}}\right)} \\
&- \frac{\sigma_n}{\sigma_\theta}\mathbb{E}_{x\sim P_n}\frac{\left(\sigma_\theta - x + \mu_\theta\right)\left(\sigma_\theta + x - \mu_\theta\right)\mathrm{e}^{\frac{(x-\mu_n)^2}{2\sigma_n^2}}}{\sigma_\theta^2\cdot\left(\sigma_\theta \mathrm{e}^{\frac{(x-\mu_\theta)^2}{2\sigma_\theta^2}} + \sigma_n \mathrm{e}^{\frac{(x-\mu_n)^2}{2\sigma_n^2}}\right)} \\
&= \int\frac{\left(\sigma_\theta - x + \mu_\theta\right)\left(\sigma_\theta + x - \mu_\theta\right)\mathrm{e}^{\frac{(x-\mu_\theta)^2}{2\sigma_\theta^2}}}{\sigma_\theta^2\cdot\left(\sigma_\theta \mathrm{e}^{\frac{(x-\mu_\theta)^2}{2\sigma_\theta^2}} + \sigma_n \mathrm{e}^{\frac{(x-\mu_n)^2}{2\sigma_n^2}}\right)}\cdot P_d(x)dx \\
&- \frac{\sigma_n}{\sigma_\theta}\int\frac{\left(\sigma_\theta - x + \mu_\theta\right)\left(\sigma_\theta + x - \mu_\theta\right)\mathrm{e}^{\frac{(x-\mu_n)^2}{2\sigma_n^2}}}{\sigma_\theta^2\cdot\left(\sigma_\theta \mathrm{e}^{\frac{(x-\mu_\theta)^2}{2\sigma_\theta^2}} + \sigma_n \mathrm{e}^{\frac{(x-\mu_n)^2}{2\sigma_n^2}}\right)}\cdot P_n(x)dx.
\end{aligned}
\tag{4.9}
$$

$$\frac{\partial L(x,\sigma_\theta,\mu_\theta)}{\partial \sigma_\theta} = \sum_{x\sim P_d} \frac{(\sigma_\theta - x + \mu_\theta)\,(\sigma_\theta + x - \mu_\theta)\,\mathrm{e}^{\frac{(x-\mu_\theta)^2}{2\sigma_\theta^2}}}{\sigma_\theta^2 \cdot \left(\sigma_\theta \mathrm{e}^{\frac{(x-\mu_\theta)^2}{2\sigma_\theta^2}} + \sigma_n \mathrm{e}^{\frac{(x-\mu_n)^2}{2\sigma_n^2}}\right)}$$
$$- \frac{\sigma_n}{\sigma_\theta} \sum_{x\sim P_n} \frac{(\sigma_\theta - x + \mu_\theta)\,(\sigma_\theta + x - \mu_\theta)\,\mathrm{e}^{\frac{(x-\mu_n)^2}{2\sigma_n^2}}}{\sigma_\theta^2 \cdot \left(\sigma_\theta \mathrm{e}^{\frac{(x-\mu_\theta)^2}{2\sigma_\theta^2}} + \sigma_n \mathrm{e}^{\frac{(x-\mu_n)^2}{2\sigma_n^2}}\right)}. \tag{4.10}$$

From the result of eq.(4.10), let $\mu_n = 0$, $\mu_d = 0$, $\mu_\theta = 0$, the following equation be derived:

$$\frac{\partial L(x,\sigma_\theta,0)}{\partial \sigma_\theta} = \sum_{x\sim P_d} \frac{(\sigma_\theta^2 - x^2)e^{\frac{x^2}{2\sigma_\theta^2}}}{\sigma_\theta^2(\sigma_\theta e^{\frac{x^2}{2\sigma_\theta^2}} + \sigma_n e^{\frac{x^2}{2\sigma_n^2}})} - \sum_{x\sim P_n} \frac{\sigma_n}{\sigma} \cdot \frac{(\sigma_\theta^2 - x^2)e^{\frac{x^2}{2\sigma_n^2}}}{\sigma_\theta^2(\sigma_\theta e^{\frac{x^2}{2\sigma_\theta^2}} + \sigma_n e^{\frac{x^2}{2\sigma_n^2}})}. \tag{4.11}$$

Figure 4.9 plots the graph of function 4.11 given different choice of noise. Because it is difficult to calculate the integral, we sample ($N = 50000$) data to simulate the integral. As shown in figure 4.9, the figure plots the values of gradients with respect to the value of model parameters, given different choices of the noise distribution. Suppose the data distribution corresponds to a one-dimensional Gaussian distribution $\mathcal{N}(0, 1)$, the model distribution corresponds to a one-dimensional Gaussian distribution $\mathcal{N}(0, \sigma_m^2)$, and the noise are chosen as $\mathcal{N}(0, 1^2)$, $\mathcal{N}(0, 2^2)$, $\mathcal{N}(0, 5^2)$, $\mathcal{N}(0, 10^2)$, $\mathcal{N}(0, 1000^2)$, respectively. From the graph, it can be observed that with the increasing standard variance of noise distribution, the gradient approaches zero. When the noise variance is large enough, the gradient is almost zero everywhere. When the noise is close to the data distribution, the signal of the gradient is stronger.

We consider another situation of mean estimation. Suppose we have a data distribution of one-dimensional Gaussian $\mathcal{N}(0, 1)$, the model distribution corresponds to a one-dimensional Gaussian distribution $\mathcal{N}(\mu_m, 1^2)$, and the noise distribution are $\mathcal{N}(0, 1^2)$, $\mathcal{N}(2, 1^2)$, $\mathcal{N}(4, 1^2)$, $\mathcal{N}(6, 1^2)$, $\mathcal{N}(8, 1^2)$, $\mathcal{N}(10, 1^2)$ respectively. We plot the graph of gradient value with respect to the mean of the model under different choices of noise mean, as shown in figure 4.10. From the graph, we can find that when the mean of noise is close to the data distribution, the gradient signal is very strong; however, when the mean of noise is different from the data mean, the gradient of the model will approach zero almost

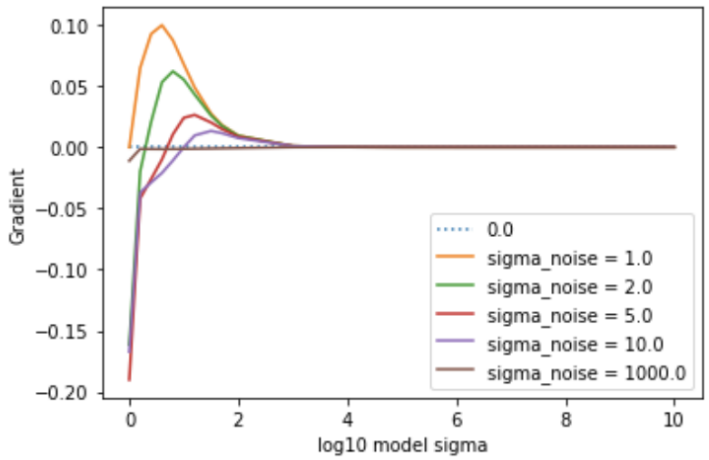

Figure 4.9: The figure shows how the value of gradient changes with respect to the sigma of model parameters. From the graph, it can be observed that with the increase in the standard deviation of noise distribution, the gradient approaches zero. When the noise variance is large enough, the gradient is almost zero everywhere. When the noise is close to the data distribution, the signal of the gradient is stronger.

everywhere (pink line).

Comparing the two experiments, it can be summarized that the behaviour of the NCE estimator is sensitive to the noise distribution. From a perspective of the absolute value of mean and variance, NCE is more sensitive to the change in mean. The choice of mean and variance are in different orders of magnitude. For example, when the data distribution is $\mathcal{N}(0, 1)$, the noise distribution could be chosen as $\mathcal{N}(0, 1000^2)$ which still works for NCE. However, if the noise distribution is chosen as $\mathcal{N}(1000, 1^2)$, the NCE will completely fail to converge. The redundancy for value choice of variance is much larger than the choice of the mean. Therefore this experiment provides a guide for choosing noise for NCE. Given the same KLD, a noise which has a larger variance should be considered. This tip will be helpful in the practical training of NCE. By applying this tip, if the noise of NCE is defined as a Gaussian or mixture Gaussian distribution, one should first find the mean or means of the data and apply a large variance.

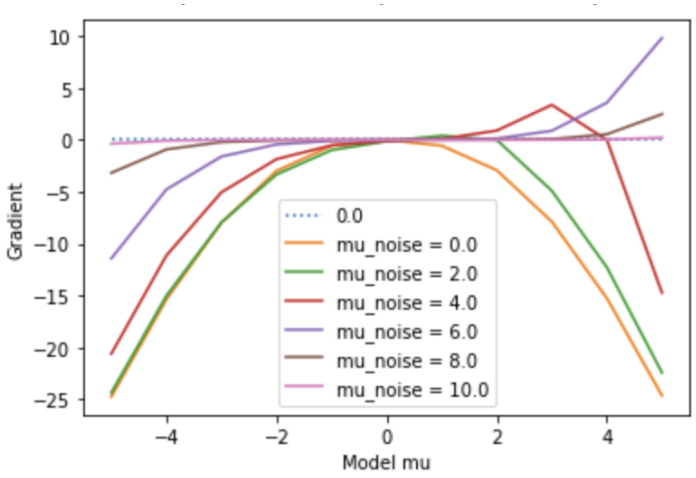

Figure 4.10: The figure shows how the value of gradient changes with respect to the mean of model parameters under different choices of noises. The graph shows when the mean of noise is close to the data distribution, the gradient signal is very strong; however, when the mean of noise is slightly different from the data mean, the gradient of the model will approach zero (pink line).

4.2.3 Solutions to the Density Chasm

4.2.3.1 Choice of Noise Data

The problem of density chasm happens when the data distribution and noise distribution are too far away from each other in the measure of KL divergence. Therefore, a straightforward method is to have a better choice of the noise distribution. Given a fixed amount and portion of noise data, there should exist an optimal noise that can result in the most accurate estimation. In the original paper, the author suggests the optimal noise should be as close as the data distribution [20]. Another article argues that the best noise data should be related to the data distribution times its natural gradient [6]; however, the conclusion is hardly useful for practical training. Although it is still in doubt what the best noise data should be, the choice of noise should be highly related to data distribution which needs to be estimated. This becomes a chicken or egg problem. If one could know the density of data distribution, then it is unnecessary to estimate with NCE. So it is impossible to choose an "optimal" noise for NCE, but it is possible to choose a "not bad" noise that will not fail NCE.

We give several criteria for a good initialization of noise distribution.

- First, the noise distribution should be a distribution that can be easily sampled from.
- Second, the noise distribution should be a distribution that can be easily analytical expression for the log-pdf.
- Third, the noise distribution should be a distribution that has a similar mean of data distribution and has a large variance of data distribution.

Therefore, we suggest one can use a Gaussian mixture distribution as noise whose means are the centers of data found by the k-means algorithm. With the training of NCE, one can adjust the noise as an adaptive noise that moves towards data distribution gradually to enhance the accuracy of the NCE estimator.

4.2.3.2 eNCE and Normalized Gradient Descent

The consequence of the density chasm caused by the gap between KL divergence can also be solved by applying the normalized gradient descent (NGD). The density chasm will cause the flatness of the loss function. A combination of changing the loss will also solve the problem. It is reported that if we change the objective function to an eNCE, which is defined through the following objective function [47]:

$$J(p_m) = \mathbb{E}_{x\sim p_d}\sqrt{\frac{p_n(x)}{p_m(x)}} + \mathbb{E}_{x\sim p_n}\sqrt{\frac{p_m(x)}{p_n(x)}}, \tag{4.12}$$

It is also reported that combining NGD with the eNCE can solve the density chasm problem [47].

4.3 Stuck Stage of Training

The third obstacle to training the NCE is when the NCE estimator is defined as an unnormalized model, the loss function seems to be "stuck" in the process of training when we observe the loss function. In the above discussion, the model is restricted as a normalized model. We extend our experiments into the unnormalized model. During the training process, it can be found that the loss function has a stage of fake convergence, which will be misunderstood as the finish of training. We investigate the reason for the stuck stage of the NCE training with empirical experiments.

The experiment settings are set similarly to above, as a one-dimensional Gaussian estimation task. Suppose the data distribution is known. Data are drawn randomly from a one-dimensional Gaussian distribution, $p_d \sim \mathcal{N}(0, 1.0^2)$. Noise distribution also are defined as one-dimensional Gaussian distribution $p_{n1} \sim \mathcal{N}(0, 1.0^2)$, $p_{n2} \sim \mathcal{N}(0, 2.0^2)$, $p_{n3} \sim \mathcal{N}(0, 5.0^2)$, $p_{n4} \sim \mathcal{N}(0, 10.0^2)$ respectively. Energy model is defined as $E(x) + c$, its distribution $P_m \sim k * \mathcal{N}(0, \sigma^2)$, $P_m(x) = \frac{e^{-E(x)+c}}{\int_{x \in \mathcal{X}} e^{-E(x)+c}}$. The change of normalizing constant, mean and variance of the energy model are tracked. The figure 4.11 and figure 4.12 show the result of NCE parameters under different noise settings.

From the graphs, it can be observed that the learning of normalizing constant have an influence on the behaviour of the loss function. The loss seems to be stuck during the learning estimator. A similar phenomenon also happens in MINE. The main reason behind it could be the self-normalizing effect of the estimator. At the beginning of training, the gradient signal comes from the changing of the normalizing constant. Then the normalizing constant converges to the ground truth slowly. It can also be found that the stuck stage of loss function will be longer when the noise distribution is close to the data distribution. With the increase of the noise variance, the phenomenon will be largely reduced. In addition, the noise with a larger variance will lead to a faster convergence for the loss. These two phenomena also encourage us to choose a noise that has a larger variance for NCE. Recall the experiments of 8 Gaussian estimation experiments mentioned at the beginning of this chapter; the stuck stage can explain the slow convergence of NCE when the noise is chosen as the same as data distribution.

4.4 Summary

In this chapter, we decompose and investigate several possible training obstacles in NCE training. The first obstacle tells us that although theoretically, NCE guarantees its feasibility with all noises as long as it is positive everywhere, practically with a bad noise, NCE may require a large portion of noise data which costs more computational resources. The second obstacle is the density chasm, which will fail the training of NCE. The third obstacle tells us that when the noise is close to the data distribution, it may cause the stuck in training. The first one tells us that the portion of noise sample matters. The second and third one tells us choice of noise distribution will affect both accuracy and efficiency. All the obstacles actually are related to one question, which is what is a relatively good noise which makes NCE work both statistically and computationally efficient. Through our experiments, we specify that a noise which has a close mean to the data and a wider variance is a good choice. It can not only generate enough gradient signals but also can accelerate

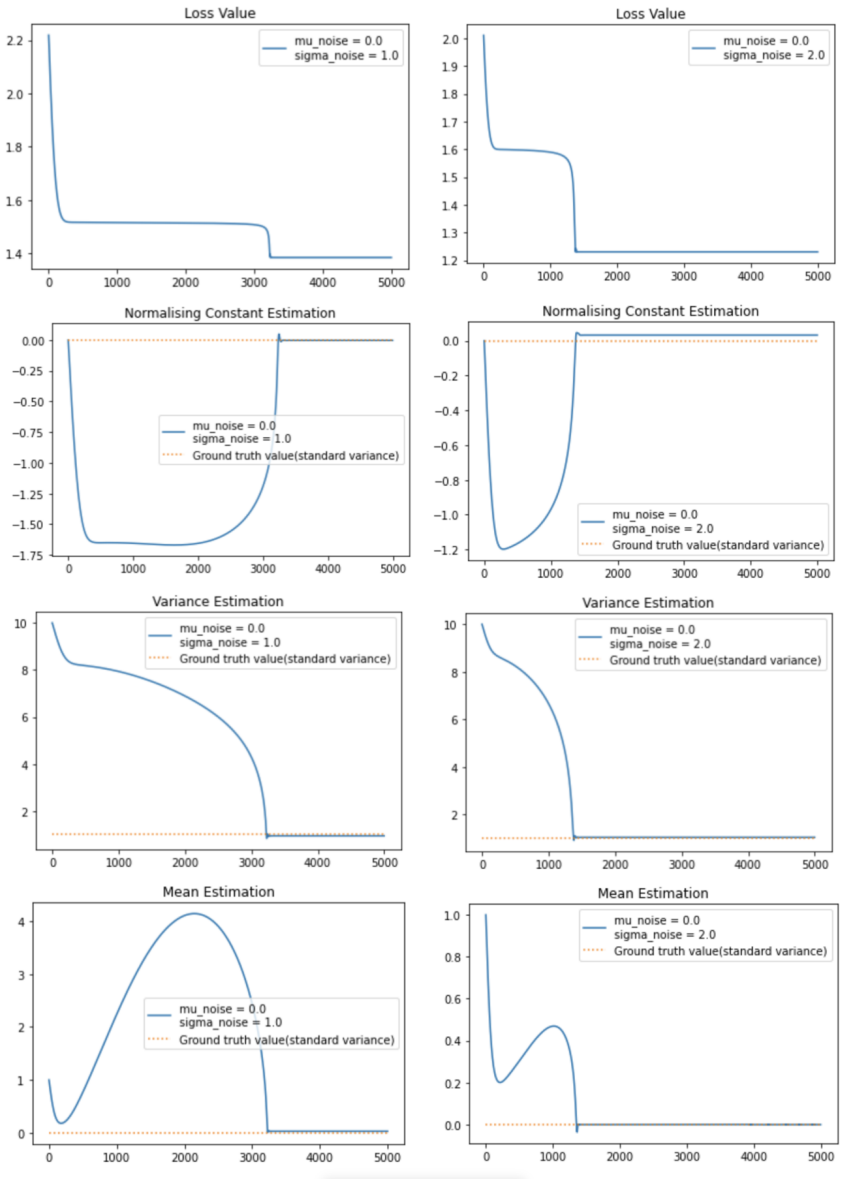

Figure 4.11: The figure shows the stuck stage of training an unnormalised NCE estimator when the noise is $p_{n1} \sim \mathcal{N}(0, 1.0^2)$ (left), $p_{n2} \sim \mathcal{N}(0, 2.0^2)$ (right). The flatness of loss function is highly related to the learning of normalizing constant. The increasing of noise variance will reduce the phenomenon of training stuck; however, the estimator will be less accurate.

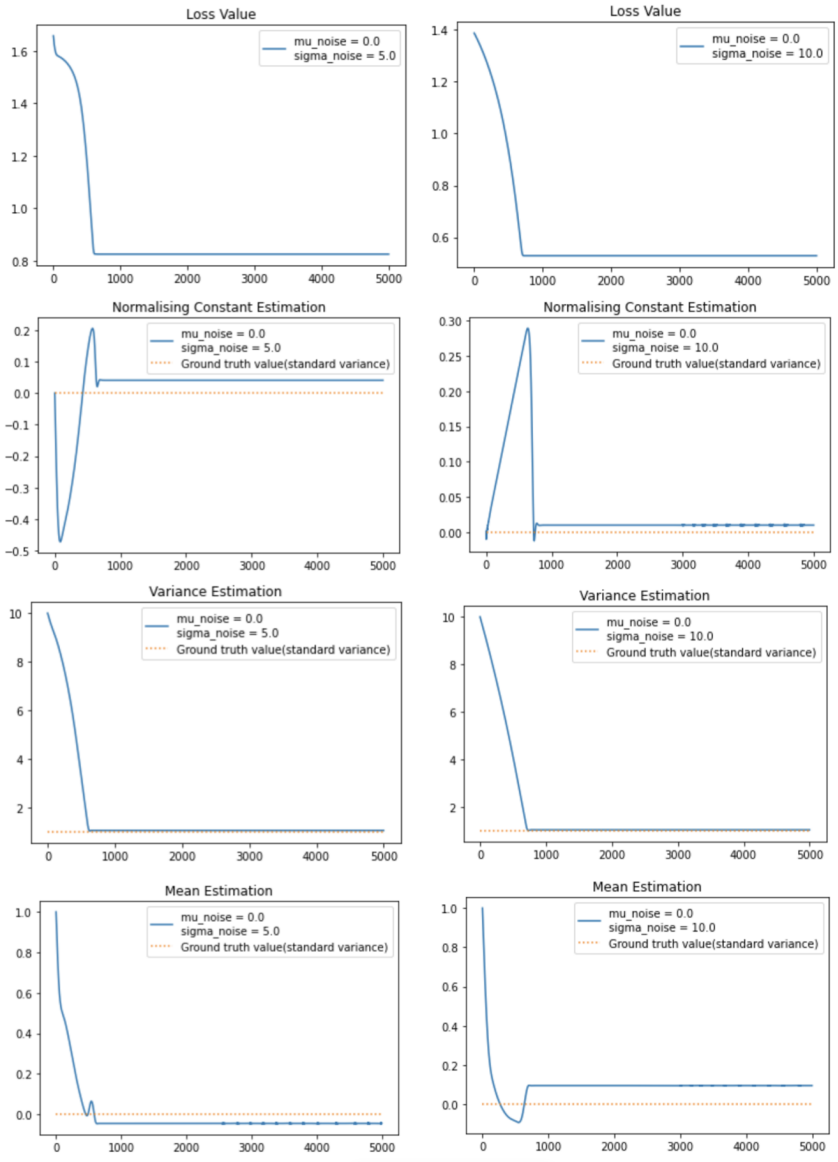

Figure 4.12: The figure shows the stuck stage of training an unnormalised NCE estimator when the noise is $p_{n3} \sim \mathcal{N}(0, 5.0^2)$ (left), $p_{n4} \sim \mathcal{N}(0, 10.0^2)$ (right). The flatness of loss function is highly related to the learning of normalizing constant. The increase in noise variance will reduce the phenomenon of training stuck; however, the estimator will be less accurate.

the training process and reduce the stuck time of loss function, although the statistical performance may not be perfect. In order to take care both accuracy and computing time, we recommend to use adaptive noise for NCE. Therefore we propose an adaptive NCE framework which can overcome the obstacles of NCE and guarantee statistical efficiency.

Chapter 5

Energy Based Adaptive Noise Contrastive Estimation

Comparing all the EBMs and considering all the obstacles of NCE, we design a framework for energy-based model. The idea comes from the practical performance of NCE. We know that there exists a computational and statistical trade-off in the choice of NCE. Therefore, it is natural to design an NCE with an adaptive noise to balance both computational and statistical performance. The adaptive noise contrastive estimation consists of two sets of learned parameters: adaptive noise and energy model. We choose adaptive noise as a Gaussian mixture model, because it is easy to sample from and the log-pdf of the Gaussian mixture model is easy to calculate. An energy model can be designed as a neural network whose inputs are training data and outputs are energy scalars.

5.1 Gaussian Mixture Model

A Gaussian Mixture Model (GMM) is a probabilistic model which is comprised of n Gaussian distributions, weighted by $w \in w_1, w_2, ..., w_n$, where $\sum_{i=1}^{n} = 1$. Each Gaussian distribution in the mixture model consists of mean μ and covariance Σ. As showed in figure 5.1.

Suppose the data $x \in \mathbb{R}^m$, the mean of each Gaussian distribution is defined as a tensor with a dimension of m, $\mu_i \in \mathbb{R}^m$. Usually, there are several different forms of GMM in practice, including diagonal GMM, low-rank GMM and full-rank GMM. The difference of these three GMMs is how to define the covariance matrix Σ_i of each Gaussian distribution:

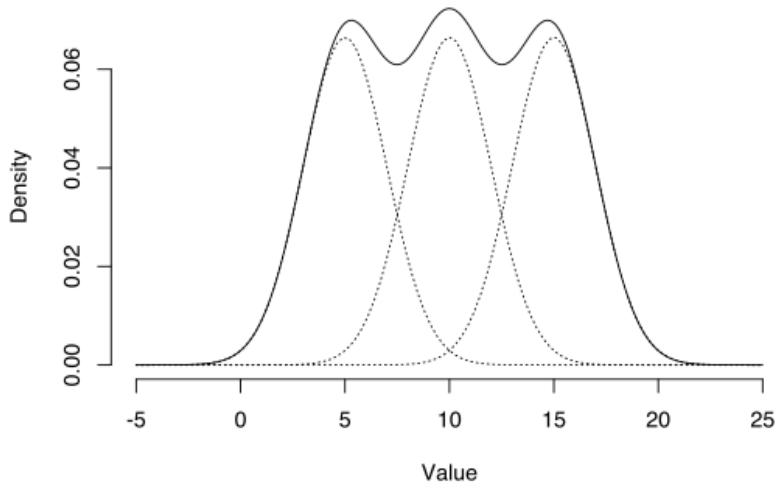

Figure 5.1: An example of Gaussian mixture model in one dimension.

- Diagonal GMM: $\Sigma_i = \sigma_i \sigma_i^T$, where σ_i is a tensor with dimension of m, $\sigma_i \in \mathbb{R}^m$
- Low-rank GMM: $\Sigma_i = diag(d_i) + \sigma_i diag(s_i) \sigma_i^T$, where σ_i is a scale perturb factor with dimension of $m \times l$, $\sigma_i \in \mathbb{R}^{m \times l}$, usually $l \ll m$, d_i is a scale diagonal matrix with dimension of $m \times m$, $\sigma_i \in \mathbb{R}^{m \times m}$, s_i is a scale perturb diagonal matrix with dimension of $l \times l$, $\sigma_i \in \mathbb{R}^{l \times l}$.
- Full-rank GMM: $\Sigma_i = \sigma_i \sigma_i^T$, where σ_i is a tensor with dimension of $m \times m$, $\sigma_i \in \mathbb{R}^{m \times m}$

Diagonal GMM can be chosen for low-dimensional data, while low-rank GMM can be chosen as the noise model for high-dimensional data. The probability density function of GMM can be written as:

$$p_{GMM}(x) = \sum_{i=1}^{n} w_i \cdot [\det(2\pi\Sigma_i)^{-\frac{1}{2}} exp(-\frac{1}{2}(x - \mu_i)^T \Sigma_i^{-1}(x - \mu_i))]. \tag{5.1}$$

The GMM is easy to sample from and the log-pdf is easy to compute. In addition, for the noise of NCE, we do not require the noise to have a large capacity to express the data distribution perfectly. We only need a noise which can roughly simulate the data distribution. Therefore, GMM is a reasonable choice of noise. Our goal is to update the GMM towards data distribution during the training of NCE.

5.2 Algorithm of Energy-based Adaptive Noise Contrastive Estimation with Gaussian Mixture Model

We advise a framework which can apply adaptive GMM. Similar to GAN, we can define an objective function with a mini-max form:

$$\min_{\phi}\max_{\theta} J(\theta,\phi) = \min_{\phi}\max_{\theta} \mathbb{E}_{x\sim p_d}\ln\{\sigma[-E_\theta(x) - \ln p_\phi(x)]\} + \mathbb{E}_{x\sim p_\phi}\ln\{1-\sigma[-E_\theta(x) - \ln p_\phi(x)]\}, \tag{5.2}$$

where θ is the parameter of energy model, and ϕ is the parameter of GMM noise. The objective function consists of two parts: updating θ by maximizing the objective function, and updating noise ϕ by minimizing the objective function. The algorithm is shown as follows:

Algorithm 5.1 Energy-based Adaptive Noise Contrastive Estimation

Initialize energy θ
Initialize GMM noise ϕ
Set $j = 0$
while $j \leq epoch$ **do**
  while k steps **do**
    Sample $y_1, y_2, ..., y_n$ from $y \sim P_\phi$
    Sample $x_1, x_2, ..., x_n$ from data
    Update energy θ by maximizing objective function: $\frac{1}{n}\sum_{i=1}^{n}\ln\{\sigma[-E_\theta(x_i) - \ln p_\phi(x_i)]\} + \frac{1}{n}\sum_{i=1}^{n}\ln\{1-\sigma[-E_\theta(y_i) - \ln p_\phi(y_i)]\}$
  end while
  Sample $y_1, y_2, ..., y_n$ from$y \sim P_\phi$
  Sample $x_1, x_2, ..., x_n$ from data
  Update GMM noise ϕ by minimizing objective function: $\frac{1}{n}\sum_{i=1}^{n}\ln\{\sigma[-E_\theta(x_i) - \ln p_\phi(x_i)]\} + \frac{1}{n}\sum_{i=1}^{n}\ln\{1-\sigma[-E_\theta(y_i) - \ln p_\phi(y_i)]\}$
  $j = j + 1$
end while

Same as GAN, the log trick can also be applied to adaptive noise contrastive estimation

at the beginning of training noise GMM ϕ.

5.3 Optimality of Energy-based Adaptive Noise Contrastive Estimation with Gaussian Mixture Model

The optimality also consists of two parts, the optimality of energy and the optimality of noise. First, we need to investigate the optimality of energy given any noise.

5.3.1 Optimality of Energy

When the noise is fixed, the objective function is exactly the same as NCE. The energy will reach its optimum when and only when the energy model matches the negative log-likelihood of data distribution. The objective function of the energy learning phase is:

$$\max_{\theta} J(\theta, \phi) = \max_{\theta} \mathbb{E}_{x\sim p_d} \ln \sigma[-E_\theta(x) - \ln p_\phi(x)] + \mathbb{E}_{x\sim p_\phi} \ln 1 - \sigma[-E_\theta(x) - \ln p_\phi(x)], \tag{5.3}$$

according to the optimality of NCE, by maximizing the objective function, when and only when $E_\theta(x) = -\ln p_d$, the objective function reaches its maximum.

5.3.2 Optimality of Noise

Given the optimal energy $E_{\theta^*}(x)$, where $p^*_m(x) = \frac{e^{-E_{\theta^*}(x)}}{\int_{x\in\mathcal{X}} e^{-E_{\theta^*}(x)}} = p_d$, the objective function can be written as:

$$\begin{aligned} J(\theta^*, \phi) &= \mathbb{E}_{x\sim p_d} \ln\{\sigma[-E_{\theta^*}(x) - \ln p_\phi(x)]\} + \mathbb{E}_{x\sim p_\phi} \ln\{1 - \sigma[-E_{\theta^*}(x) - \ln p_\phi(x)]\} \\ &= \mathbb{E}_{x\sim p_d}[\ln \frac{p^*_m(x)}{p^*_m(x) + p_\phi(x)}] + \mathbb{E}_{x\sim p_\phi}[\ln \frac{p_\phi}{p^*_m(x) + p_\phi(x)}] \\ &= \mathbb{E}_{x\sim p_d}[\ln \frac{p_d(x)}{(p_d(x) + p_\phi(x))/2}] - \ln 2 + \mathbb{E}_{x\sim p_g}[\ln \frac{p_\phi(x)}{(p_d(x) + p_\phi(x))/2}] - \ln 2 \\ &= JSD(p_\phi, p_d) - \ln 4. \end{aligned} \tag{5.4}$$

Therefore, after the optimal energy is reached, minimizing the objective function by updating ϕ is equivalent to minimizing JSD between p_ϕ and p_d, which leads to the optimality of noise.

Different from GAN, it is not necessary to achieve the optimality of noise for ANCE, so in practice, we can train energy more times than training noise in each epoch to guarantee energy reaches its optimality.

5.4 Results of Adaptive Noise Contrastive Estimation

5.4.1 Feasibility of Adaptive Noise Contrastive Estimation

In this section, we use synthetic data to demonstrate that Adaptive Noise Contrastive Estimation (ANCE) is a feasible energy model, even when the noise data is considered a "bad" noise in the context of NCE because ANCE will be self-corrected towards data distribution.

Considering a two-dimensional 8 Gaussian mixture model as data distribution, whose data distribution is:

$$p_{data}(x) = \sum_{i=1}^{8} \frac{1}{8} \cdot [\det(2\pi\sigma)^{-\frac{1}{2}} exp(-\frac{1}{2}(x-\mu_i)^T \sigma^{-1}(x-\mu_i))], \tag{5.5}$$

where $\sigma = \begin{bmatrix} 0 & 0.05 \\ 0.05 & 0 \end{bmatrix}$, $\mu_i = [0,-1], [0,1], [1,0], [-1,0], [-\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}], [\frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}}], [-\frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}}]$, $[\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}]$ respectively. The data is shown in figure 5.2. We compare different performances of NCE and ANCE under the same noise initialization. The energy of NCE and ANCE are equally modelled by a three-layer neural network. In order to compare the statistical performance, we selected points on a circle with a radius of 1 and a line where the energy scalars of data distribution and energy model are recorded, as shown in figure 5.3. In addition, we sample data from energy by Metropolis-Adjusted Langevin Algorithm.

The first pair of comparison results are shown in figure 5.5 and figure 5.6. The noise of NCE and initialized noise of ANCE are chosen as:

$$p_{noise1}(x) = \sum_{i=1}^{8} \frac{1}{8} \cdot [\det(2\pi\sigma_1)^{-\frac{1}{2}} exp(-\frac{1}{2}(x-\mu_i)^T \sigma_1^{-1}(x-\mu_i))], \tag{5.6}$$

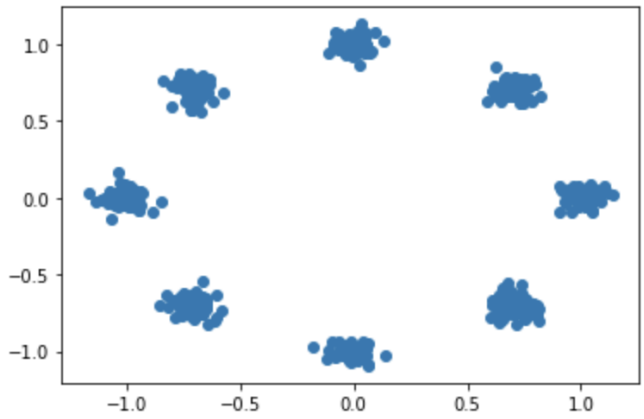

Figure 5.2: The figure shows the data to be estimated.

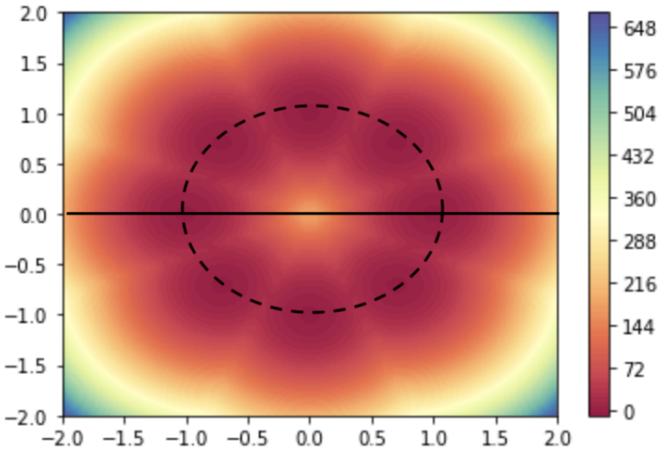

Figure 5.3: The energy of points on the circle $x^2 + y^2 = 1$ and line $x = 1$ are computed after the training of model.

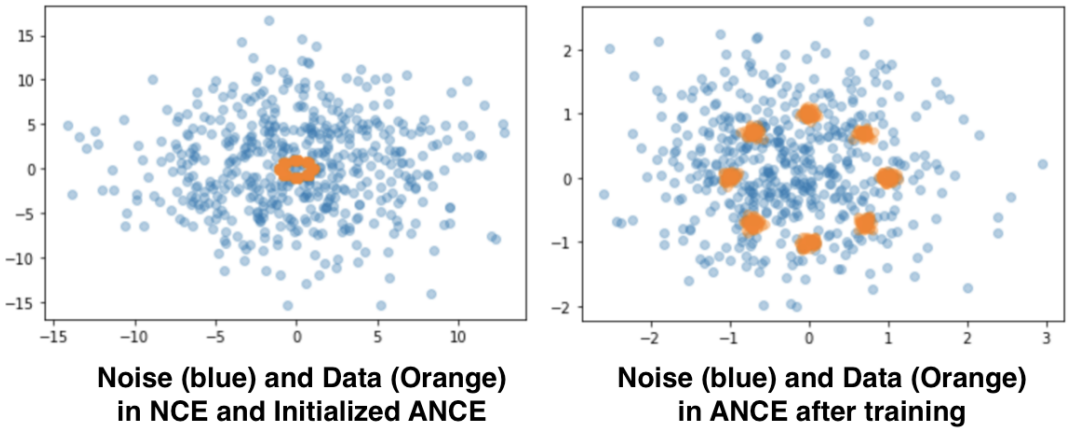

Figure 5.4: The left figure shows samples from the noise distribution p_{noise1} and data distribution p_{data}. The right figure shows samples from adaptive noise of ANCE after training. The noise distribution is updated toward data distribution.

where $\sigma_1 = \begin{bmatrix} 0 & 5 \\ 5 & 0 \end{bmatrix}$, $\mu_i = [0,-1],\ [0,1],\ [1,0],\ [-1,0],\ [-\frac{1}{\sqrt{2}},\frac{1}{\sqrt{2}}],\ [\frac{1}{\sqrt{2}},-\frac{1}{\sqrt{2}}],\ [-\frac{1}{\sqrt{2}},-\frac{1}{\sqrt{2}}],\ [\frac{1}{\sqrt{2}},\frac{1}{\sqrt{2}}]$, which has a larger variance than data distribution, as shown in left of figure 5.4. Figure 5.5 shows the results of NCE with p_{noise1}. Figure 5.6 shows the results of ANCE with initialized noise p_{noise1} training with the same epochs. The result shows that ANCE performs better because the noise distribution moves towards data distribution, as shown in the right of figure 5.4.

The second pair of comparison results are shown in figure 5.8 and figure 5.9. The noise of NCE and initialized noise of ANCE are chosen as:

$$p_{noise2}(x) = \sum_{i=1}^{8} \frac{1}{8} \cdot [\det(2\pi\sigma_2)^{-\frac{1}{2}} exp(-\frac{1}{2}(x-\mu_i)^T \sigma_2^{-1}(x-\mu_i))], \tag{5.7}$$

where $\sigma_2 = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$, $\mu_i = [0,-5],\ [0,5],\ [5,0],\ [-5,0],\ [-\frac{5}{\sqrt{2}},\frac{5}{\sqrt{2}}],\ [\frac{5}{\sqrt{2}},-\frac{5}{\sqrt{2}}],\ [-\frac{5}{\sqrt{2}},-\frac{5}{\sqrt{2}}],\ [\frac{5}{\sqrt{2}},\frac{5}{\sqrt{2}}]$, which has a larger radius than data, as shown in left of figure 5.7. In this situation, noise distribution also moves towards data distribution, leading to better estimation.

From the above experiments, it can be found that when the noise is not an ideal noise, it will affect the performance of NCE, significantly when the noise and data are not

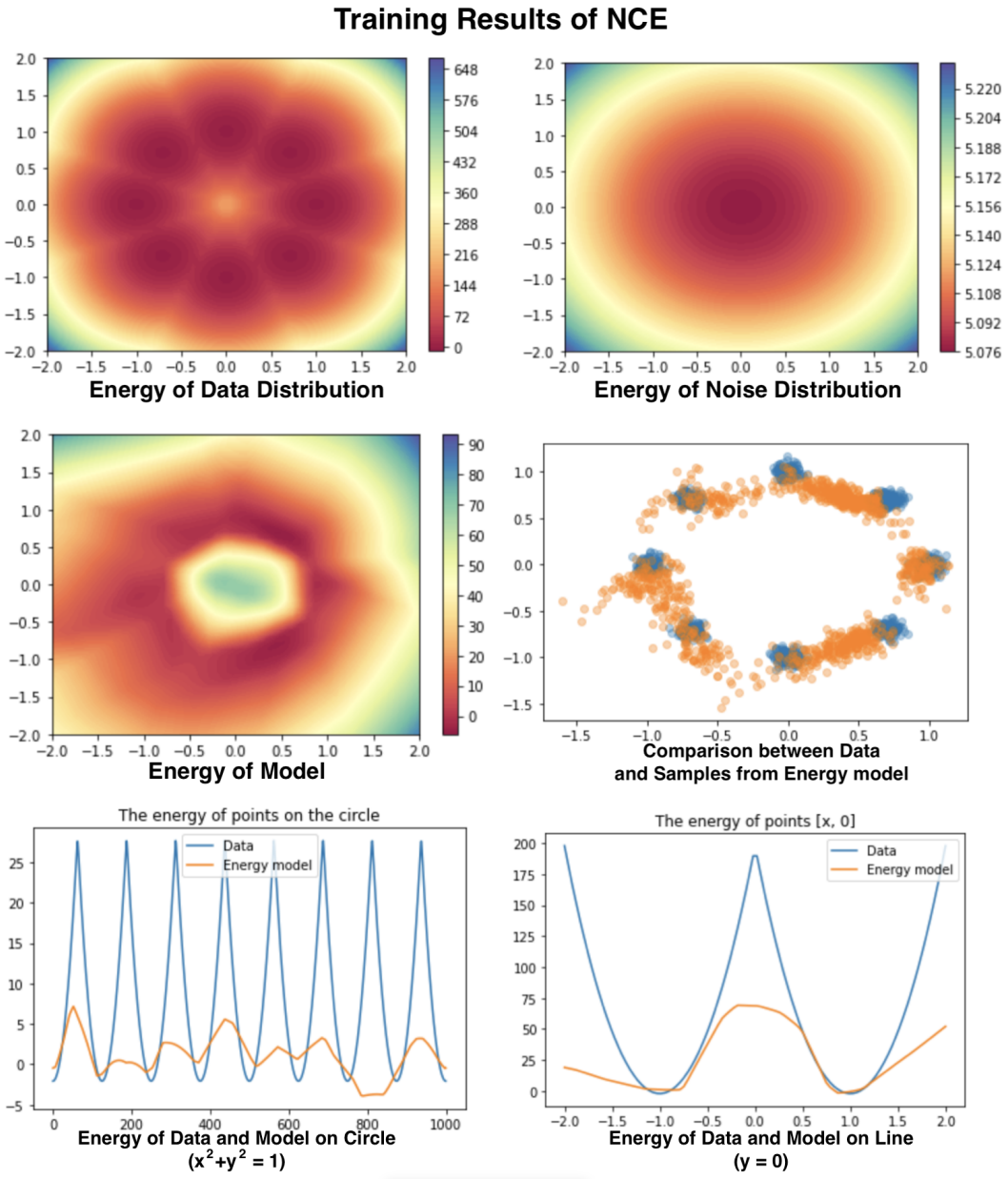

Figure 5.5: The figure shows the training results of NCE with noise p_{noise1}.

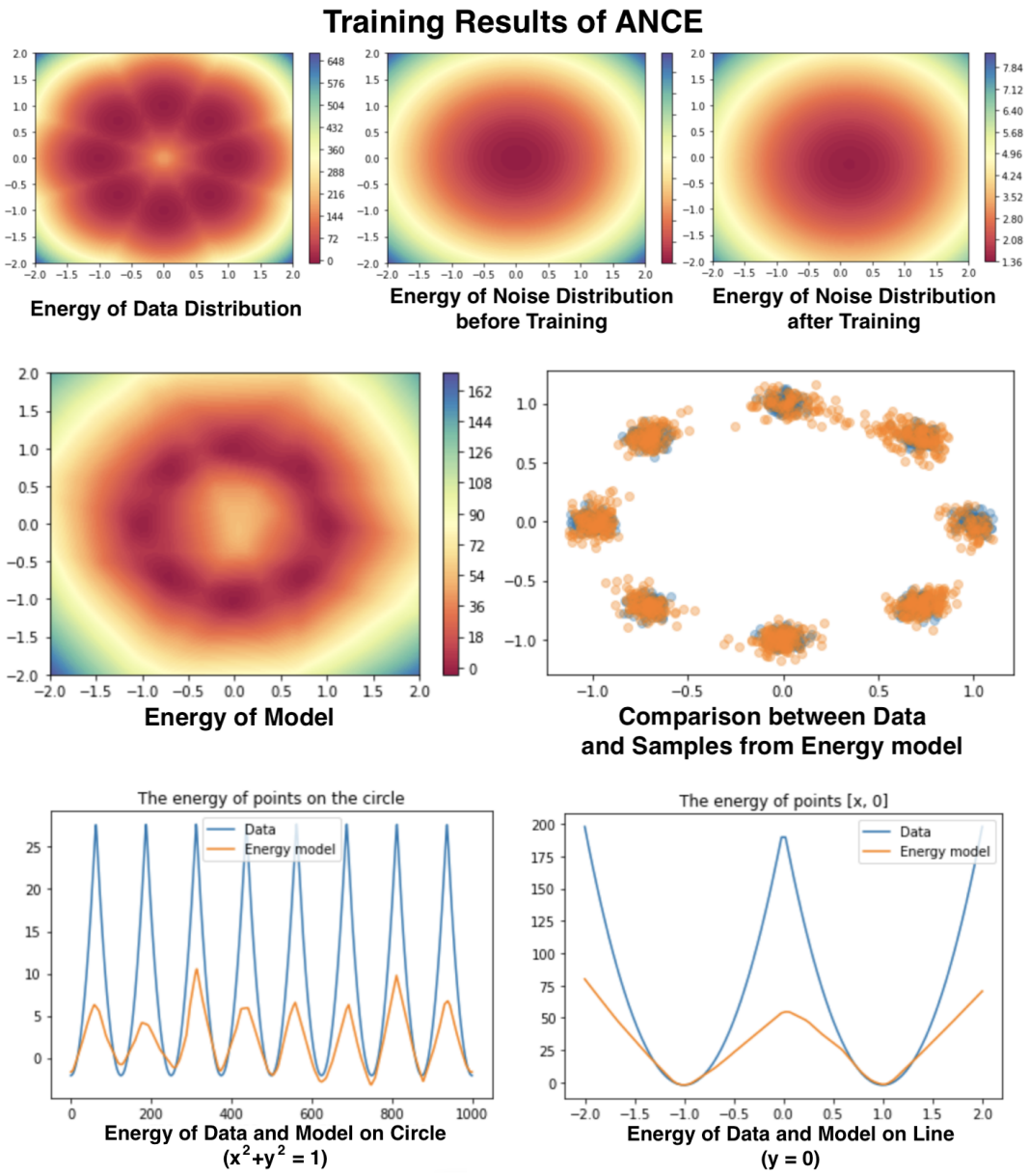

Figure 5.6: The figure shows the training results of ANCE with initialized noise p_{noise1}.

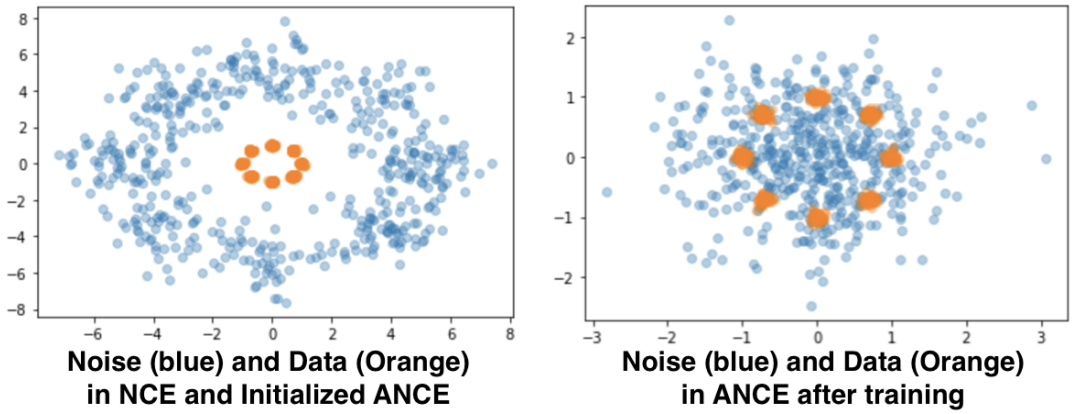

Figure 5.7: The left figure shows samples from the noise distribution p_{noise2} and data distribution p_{data}. The right figure shows samples from adaptive noise of ANCE after training. The noise distribution is updated towards data distribution.

overlapped in most regions, for example, in the second experiment. However, with the application of adaptive noise, the performance will be significantly improved.

As shown in figure 5.10, the figure shows the loss value during the training of NCE and ANCE.

5.4.2 Maximum Mean Discrepancy

In order to quantify the evaluation of the energy model, we apply MALA to sample data from the energy model and compute the maximum mean discrepancy between validation data and samples. Maximum Mean Discrepancy (MMD) is a probability measure which is often used to compare the difference between observations from two distributions [17]. The distance is based on the idea of embedding probabilities in a reproducing kernel Hilbert space (RKHS). [76] The general form of MMD is defined as [18]:

Definition 5.1. *$\mathcal{F}$ is a class of functions $f : X \to \mathbb{R}$, p and q are two distributions respectively, the MMD is defined as:*

$$MMD(\mathcal{F}, p, q) = \sup_{f \in \mathcal{F}} \mathbb{E}_{x \sim p}(f(x)) + \mathbb{E}_{y \sim q}(f(y)), \tag{5.8}$$

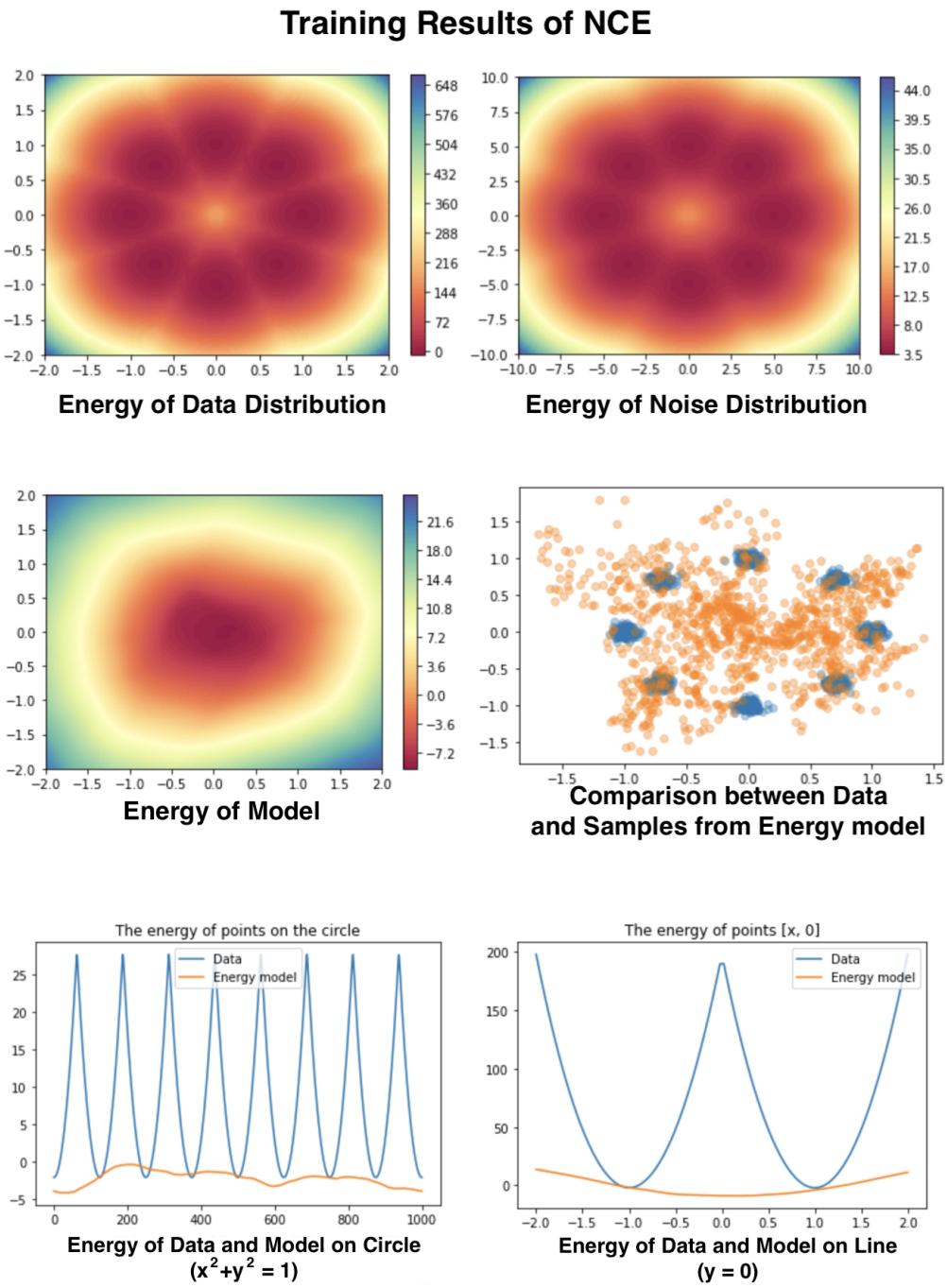

Figure 5.8: The figure shows the training results of NCE with noise p_{noise2}.

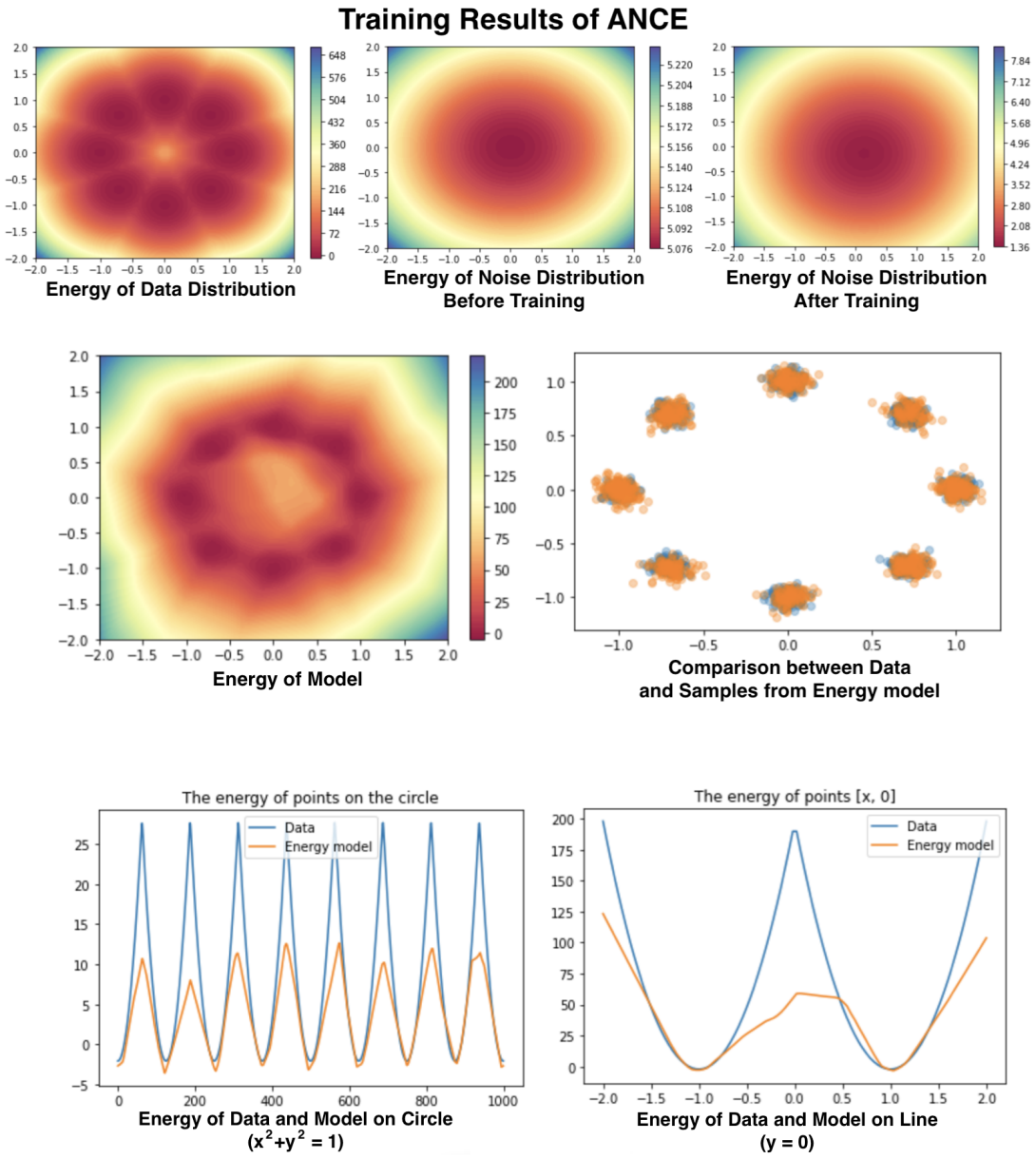

Figure 5.9: The figure shows the training results of ANCE with initialized noise p_{noise2}.

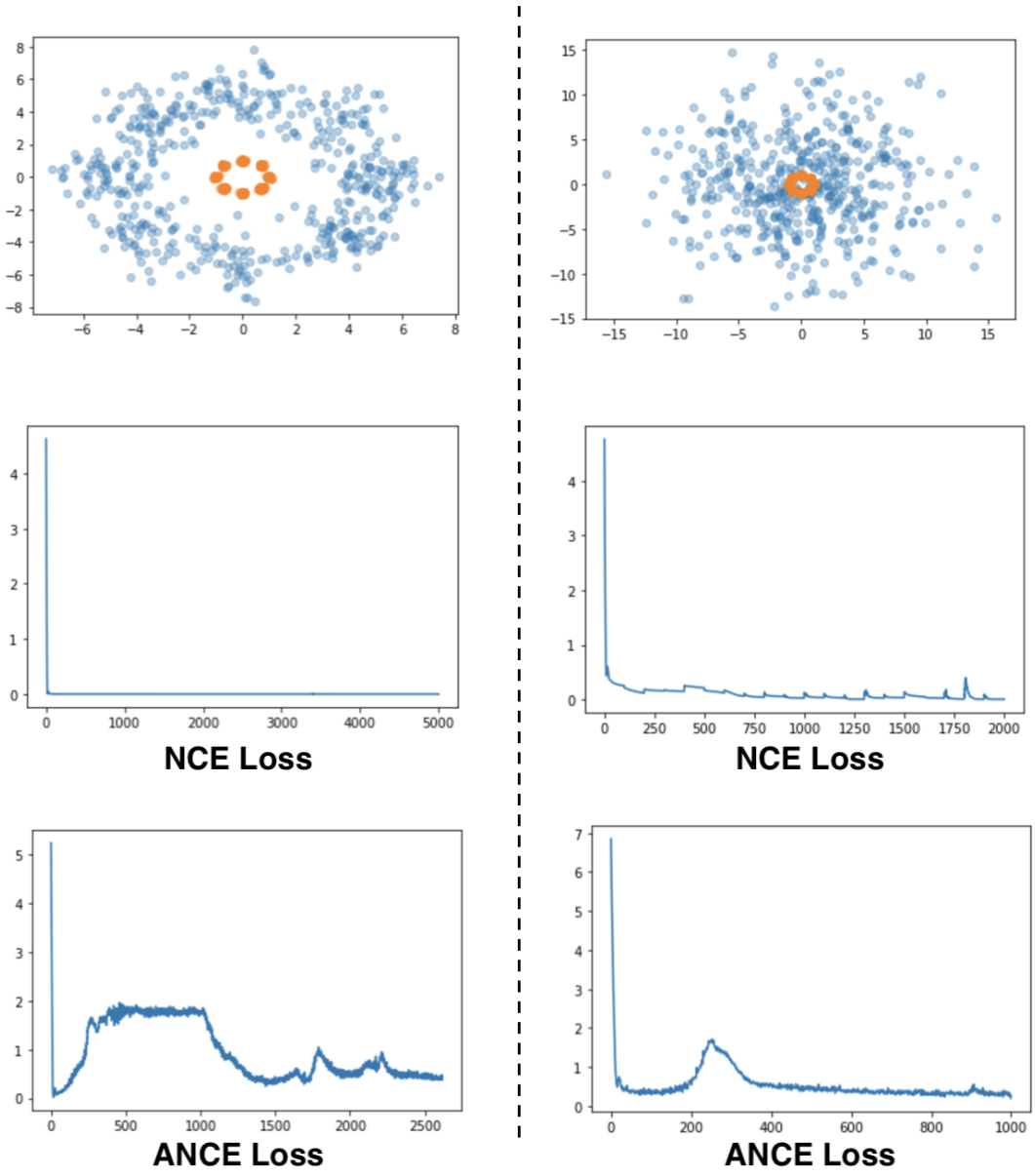

Figure 5.10: The figure shows the loss value during the training of NCE and ANCE.

an empirical estimate of the MMD between two datasets is defined as:

$$MMD(\mathcal{F}, p, q) = \sup_{f \in \mathcal{F}} \sum_{i}^{n} (f(x)) + \sum_{i}^{n} (f(y)), \tag{5.9}$$

where x_i, y_i are observations from distribution p and q.

In practice, the empirical estimation of MMD is accessed through the square of MMD by the following equation:

$$MMD^2(X, Y) = \frac{1}{n(n-1)} \sum_{i} \sum_{j \neq i} k(x_i, x_j) - 2\frac{1}{n^2} \sum_{i} \sum_{j} k(x_i, y_j) + \frac{1}{n(n-1)} \sum_{i} \sum_{j \neq i} k(y_i, y_j), \tag{5.10}$$

where $k(\cdot, \cdot)$ is a kernel function. One of the most common kernels is Radial Basis Function (RBF), where:

$$k(x_i, x_j) = \exp\left(\frac{-\|x_i - x_j\|^2}{2\sigma^2}\right). \tag{5.11}$$

Because it is difficult to directly evaluate an unnormalized energy model that is constructed through a neural network, we quantify the statistical performance indirectly through MMD by sampling from energy with the MALA algorithm. In the following experiments, we will use MMD to measure the accuracy of the energy model.

5.4.3 Experiments on Other Synthetic Dataset

Comparisons between NCE and ANCE are shown in different synthetics datasets, including two-circles, two-sines, swissroll, two-spirals, and target datasets. The experiment is conducted under the same settings of the same initialized noises. As shown in figure 5.11, ANCE achieves better results than NCE, in which the noise is regarded as an awful noise. Table 5.1 shows the MMD between samples and data for NCE and ANCE.

5.4.4 Trade-off Effect of ANCE

In this section, we choose 8 Gaussian whose variance is 0.05^2 as a dataset to demonstrate the trade-off effect of ANCE compared to NCE. As shown in figure 5.12, 4 different noises

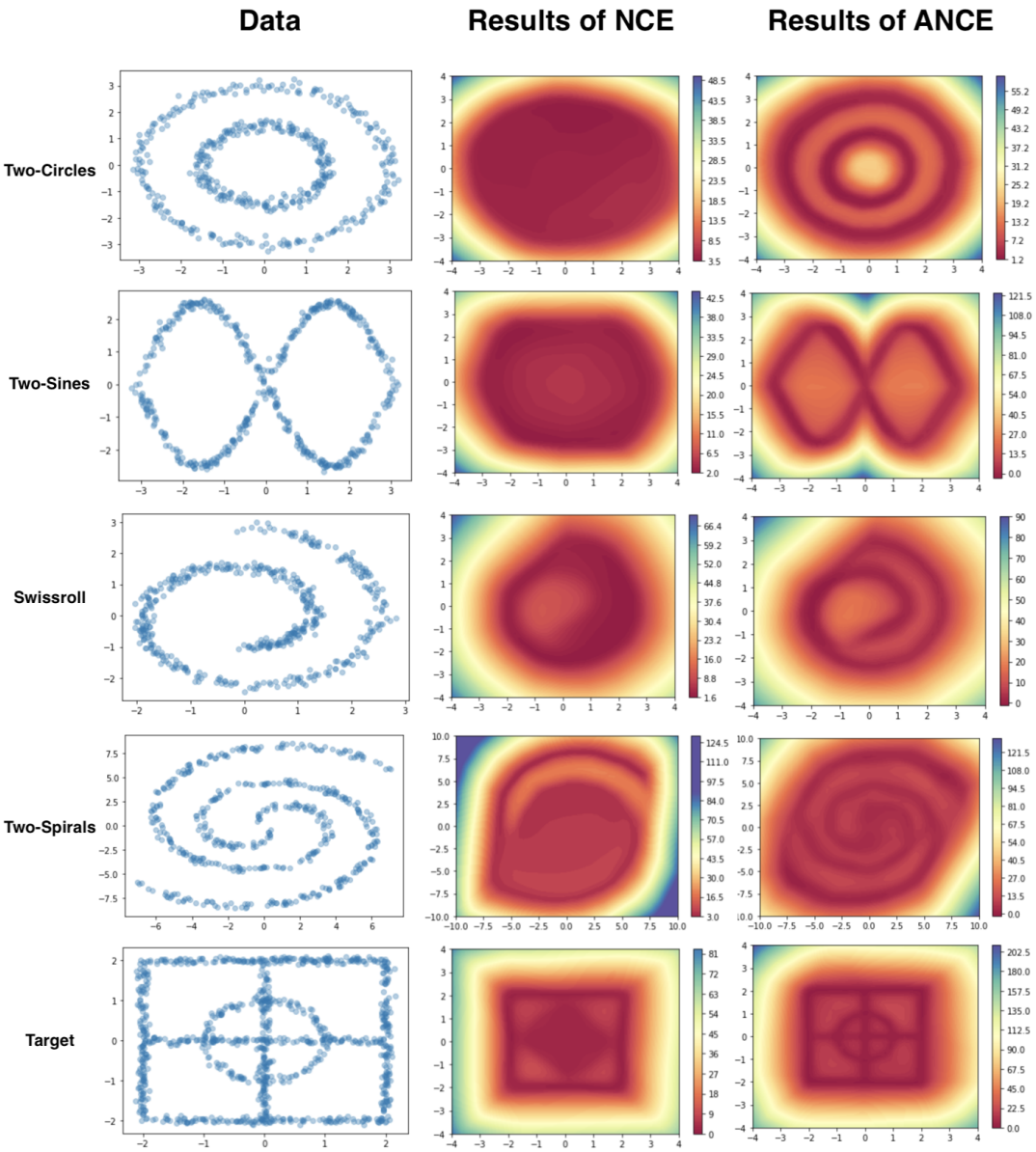

Figure 5.11: The figure shows the results of NCE and ANCE with the same initialized noises.

Dataset	MMD of NCE	MMD of ANCE
8 Gaussian	0.256	0.042
Two-circles	0.259	0.041
Two-sines	0.215	0.038
Swissroll	0.187	0.063
Two-spiral	0.162	0.072
Target	0.263	0.048

Table 5.1: MMD between samples from energy model and data for NCE and ANCE.

whose variance are 0.1^2, 1.0^2, 10.0^2, 100.0^2 respectively are chosen for NCE. It can be found that when the noise is close to the data distribution, such as NCE1 in the figure, the convergence of NCE is very slow; when the noise is far away from the data distribution, such as NCE3 and NCE4, the statistical performance of NCE is unsatisfying. Only when the noise is "ideal" for NCE, such as NCE2, the statistical and computational performance of NCE will be both satisfied. In this experiment, the initialled noise of ANCE is set as the same as NCE3. In order to compare the computational performance between NCE and ANCE, both the NCE updating process and GMM updating process are considered one iteration, which means each epoch which contains one update of NCE and GMM is considered as two iterations in ANCE. We compared the MMD between the validation dataset and sampled data from the trained energy in each iteration in different settings.

From the result, the trade-off effect can be demonstrated through the above experiment. Although ANCE converges less quickly than NCE when the noise is chosen ideally, ANCE achieves the same statistical performance. Moreover, ANCE improves both training accuracy and speed when the noise is considered a terrible noise for NCE.

Applying similar experiments to other datasets, we record the MMD between energy samples and data for NCE under different noises and ANCE, as shown in table 5.2. The GMM noises for NCE (NCE1, NCE2, NCE3, respectively) are set with the same means but different variances (3.0, 30.0, 300.0, respectively)for each NCE. From the table, it can be found that with an increase in the variance of the noise, the NCE estimator will be less accurate. By applying the ANCE, the accuracy of the estimator will not be affected by choice of noise.

In summary, several conclusions can be drawn from the graph and table:

- For NCE, a noise that is very close to data distribution is not a good choice because the convergence of training will be very slow.

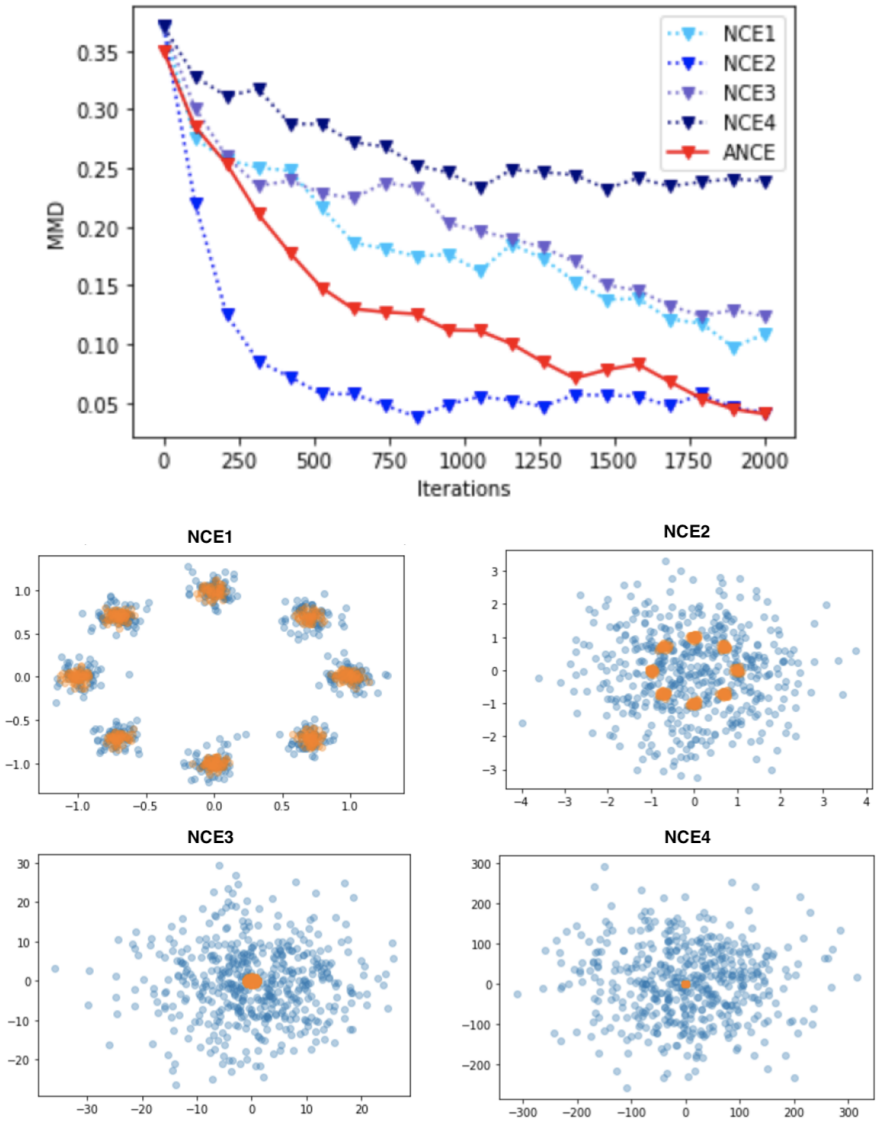

Figure 5.12: The figure shows the MMD of NCE and ANCE while training. The noises of NCE are set as $\sigma = 0.1, 1.0, 10.0, 100.0$ respectively. The initialed ANCE noise are set as $\sigma = 10.0$. The top figure shows the MMD of each set with respect to iterations. The other four show the training data and noise data in four different settings.

Dataset	MMD of ANCE	MMD of NCE1	MMD of NCE2	MMD of NCE3
Two-circles	0.041	0.039	0.152	0.259
Two-sines	0.038	0.040	0.163	0.215
Swissroll	0.063	0.052	0.149	0.187
Two-spiral	0.072	0.082	0.159	0.162
Target	0.048	0.046	0.237	0.263

Table 5.2: MMD between samples and data for ANCE and NCE under different noises.

- For NCE, when the noise is significantly different from the data, the accuracy of the estimator is limited.
- ANCE can achieve a trade-off effect which not only achieves good statistical performance but also overcomes the disadvantages of NCE brought by the bad choice of noise.

5.4.5 Comparison among NCE, NCE with NGD and ANCE

It is reported that Normalizing Gradient Descent (NGD) can partially solve the problem brought by density chasm [47]. Here we compare the different effects of ANCE and NCE with NGD. As is shown in figure 5.13, NCE with NGD will only slightly accelerate the training of NCE but it will not help improve the statistical performance. On the contrary, ANCE can improve statistical performance significantly. Normalizing gradient descent is equivalent to clipping by norm, which will intensify the gradient signal, but it will not change the statistical performance due to the noise gap.

5.4.6 Byproduct of ANCE

In addition to the energy model, ANCE also generates a byproduct - a GMM model that can estimate the data distribution. The following results show the results of the GMM model of ANCE. GMM achieves a reasonable estimation of data distribution, which proves the effectiveness of the GMM updating process of ANCE.

Similar to GAN, balancing the training of GMM and the energy model is essential in ANCE as well. But unlike GAN, which is often believed to achieve a better result by training discriminators fewer times than training generators in each epoch, it is not a

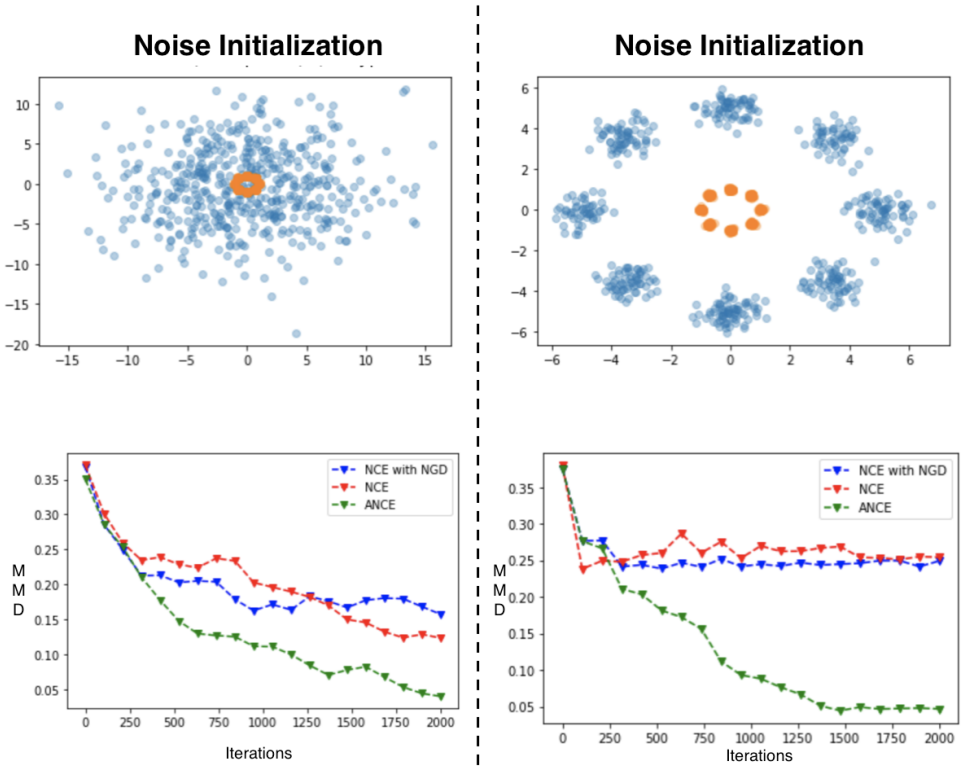

Figure 5.13: The figure shows the MMD of NCE, NCE with NGD and ANCE while training. The noise initialization are shown on the top of the figure. The bottom figure shows the MMD with respect to iterations.

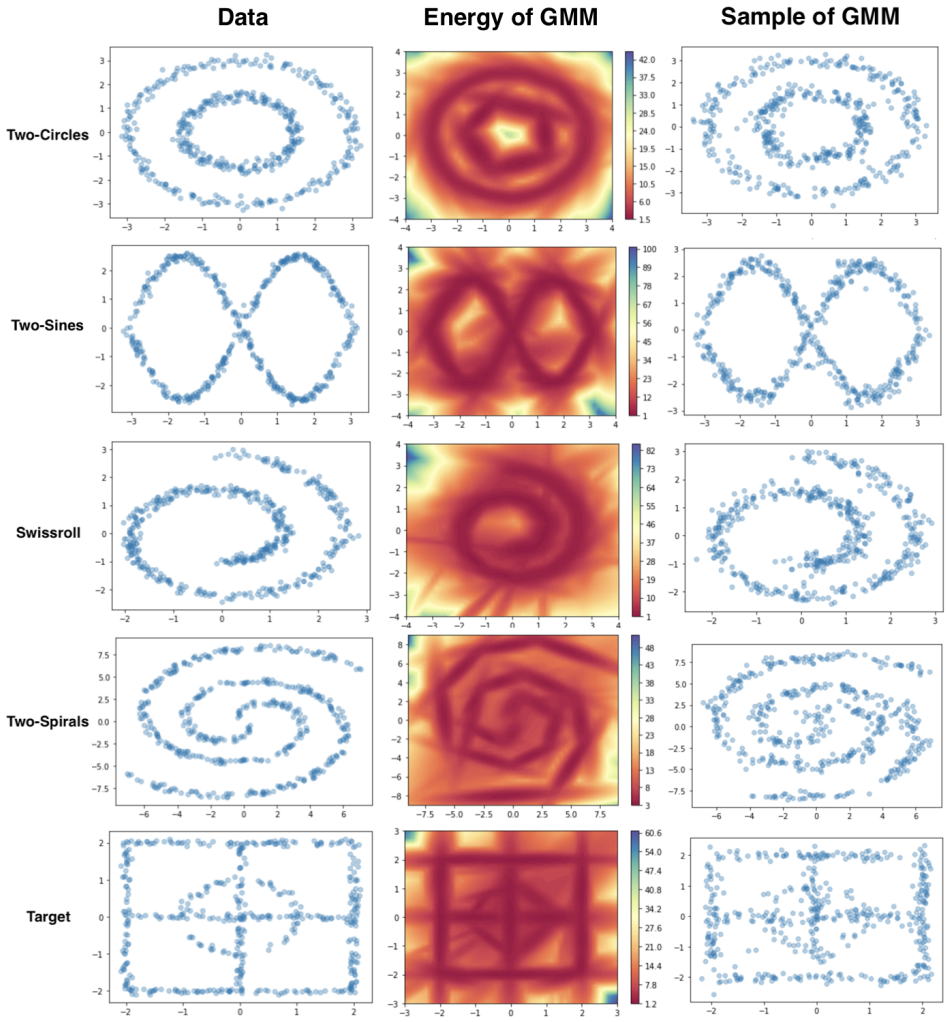

Figure 5.14: The figure shows the GMM results of ANCE.

requirement in ANCE. As a byproduct of an energy-based model, it is not necessary to request that GMM be able to reconstruct the data distribution as accurately as possible. In fact, we found that when the GMM model converges to data distribution too fast, the performance of energy may not achieve optimality. In other words, training GMM fewer times will not affect the performance of energy and even may lead to a better result. The process of training energy surpassing training GMM is acceptable in ANCE.

5.4.7 Comparison between NCE and ANCE in High-dimensional Dataset

In this section, we train NCE and ANCE with MNIST dataset to demonstrate the feasibility of algorithms in high-dimensional data. First of all, we choose a Gaussian distribution as the noise to train NCE with MNIST. We found that it is very difficult to train NCE when a Gaussian distribution or uniform distribution is chosen. Even though we choose a mixture Gaussian as the noise distribution, NCE may face serious numerical problems. ANCE will overcome the problem with careful tuning and balancing between GMM and energy model. In this experiments, we simulate anomaly detection by rotating the image with different angles. As shown in figure 5.15, ANCE shows a significantly better result than NCE. The left figure shows the energy of number 7 and the right shows the energy of number 9. When number 9 is rotated 180 degrees, it changes to six, so the energy here is low. For number 7, when the number is rotated 180 degrees, although it is lower than the top, but still higher than - 2600. This results shows that ANCE is feasible and outperforms NCE in high dimensional data.

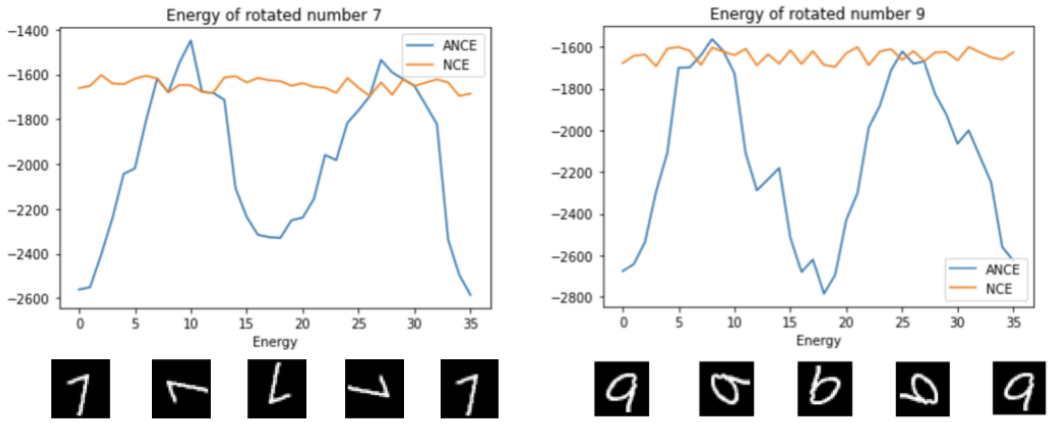

Figure 5.15: The figure shows the simulation of anomaly detection by rotating the image with different angles. From the result, it can be found that ANCE shows a significantly better result than NCE.

Chapter 6

Overview, Future Work, and Conclusion

6.1 Overview

In this thesis, we propose an effective energy-based model based on noise contrastive estimation. In summary, we made the following contributions.

Firstly, we decomposed and investigated several obstacles in the training of NCE. Our experiments with one-dimensional Gaussian data demonstrate three issues that may directly influence computational or statistical efficiency. The first obstacle tells us that there is a trade-off between training speed and accuracy inherently in NCE itself. With more noise data loaded, the accuracy will increase, but the training speed will be slower. The second obstacle is that when the computational resources are limited, the choice of noise will play a critical role in NCE. The density chasm brought by the difference in means of data and noise will fail the NCE. The third problem in NCE is a stuck stage of the loss function in NCE which is more apparent when the noise distribution is close to the data distribution. The three issues in NCE lead to a suggestion of noise choosing: it is better to choose a noise which has a much more significant variance compared to data distribution than the one that has a smaller variance, although it will sacrifice the accuracy of estimation.

Secondly, to solve the problems in NCE and avoid the negative influence of the choice of noise, we propose a framework of ANCE, in which the adaptive GMM noises are updated towards the data distribution. The ANCE can achieve better accuracy and training speed efficiency than the pure NCE when the noise of NCE is not "optimal." ANCE eliminates the negative impact on NCE brought by the bad choice of noise.

6.2 Future Work

There are several directions for future works and improvements.

Firstly, we studied the density chasm, which fails NCE training. We argue that it is inadequate to use KL divergence to measure the density chasm. However, the question about how to define a good or lousy noise remains. From the empirical experiments, we know that there should exist an optimal noise for NCE from the perspective of the accuracy of the estimator. Moreover, the optimal noise for NCE may not be the same as the data distribution, which is quite counter-intuitive. An interesting question is whether it is possible to reach the optimal noise and how it can assist in improving the accuracy of NCE. If it is possible to construct an objective function which drives the noise approaching the optimal noise, the statistical performance of NCE could be significantly improved.

Secondly, we provide a revised NCE framework with adaptive noise. Through our experiments, we found that an adaptive noise can significantly overcome the disadvantages of NCE and improve training efficiency. In this framework, we use GMM as noise. Actually, applying other popular probability models is also feasible as long as it is easy to sample and the pdf is easy to compute. For example, it is an attractive research topic to combine normalizing flow models or auto-regressive models with NCE.

Thirdly, more application scenarios of ANCE deserve more research. Especially when the data is highly dimensional, ANCE with GMM may not achieve the expected results. The main reason presumably lies in the limitations of the Gaussian mixture model.

Fourthly, the shortcomings of ANCE mainly include: it need to be well tuned to balance the GMM and energy model. Also we need to carefully avoid numerical error, especially when the data is high dimensional. It is still worthy to research on the training techniques of ANCE.

6.3 Conclusion

We studied the obstacles in the training of NCE. In order to overcome the challenges in choosing the noise of NCE, we proposed a revised version of NCE that involves an adaptive GMM noise. Adaptive noise based NCE can accelerate not only the convergence but also increase the statistical efficiency.

References

[1] Durgesh Agrawal, Yash Pote, and Kuldeep S Meel. Partition function estimation: A quantitative study, 2021.

[2] Jianmin Bao, Dong Chen, Fang Wen, Houqiang Li, and Gang Hua. Cvae-gan: fine-grained image generation through asymmetric training. In *Proceedings of the IEEE international conference on computer vision*, pages 2745–2754, 2017.

[3] Mohamed Ishmael Belghazi, Aristide Baratin, Sai Rajeswar, Sherjil Ozair, Yoshua Bengio, Aaron Courville, and R Devon Hjelm. Mine: Mutual information neural estimation. 2018.

[4] Steve Brooks, Andrew Gelman, Galin Jones, and Xiao-Li Meng. *Handbook of Markov Chain Monte Carlo*. CRC press, 2011.

[5] Chung Chan, Ali Al-Bashabsheh, Hing Pang Huang, Michael Lim, Da Sun Handason Tam, and Chao Zhao. Neural entropic estimation: A faster path to mutual information estimation, 2019.

[6] Omar Chehab, Alexandre Gramfort, and Aapo Hyvarinen. The optimal noise in noise-contrastive learning is not what you think, 2022.

[7] X. Chen, X. Liu, M. J. F. Gales, and P. C. Woodland. Recurrent neural network language model training with noise contrastive estimation for speech recognition. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5411–5415, 2015.

[8] Xiang Cheng, Niladri S. Chatterji, Peter L. Bartlett, and Michael I. Jordan. Underdamped langevin mcmc: A non-asymptotic analysis. In Sébastien Bubeck, Vianney Perchet, and Philippe Rigollet, editors, *Proceedings of the 31st Conference On Learning Theory*, volume 75 of *Proceedings of Machine Learning Research*, pages 300–323. PMLR, 06–09 Jul 2018.

[9] Bo Dai, Hanjun Dai, Arthur Gretton, Le Song, Dale Schuurmans, and Niao He. Kernel exponential family estimation via doubly dual embedding, 2018.

[10] Andrew D. Davis, Youssef Marzouk, Aaron Smith, and Natesh Pillai. Rate-optimal refinement strategies for local approximation mcmc, 2020.

[11] Laurent Dinh, David Krueger, and Yoshua Bengio. Nice: Non-linear independent components estimation, 2014.

[12] Yilun Du, Shuang Li, and Igor Mordatch. Compositional visual generation and inference with energy based models, 2020.

[13] Ruiqi Gao, Erik Nijkamp, Diederik P. Kingma, Zhen Xu, Andrew M. Dai, and Ying Nian Wu. Flow contrastive estimation of energy-based models, 2019.

[14] Charles J. Geyer. Markov chain monte carlo maximum likelihood. *Computing Science and Statistics: Proc*, 23rd Symp:156–163, 1991.

[15] Ian J. Goodfellow. On distinguishability criteria for estimating generative models, 2014.

[16] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks, 2014.

[17] Arthur Gretton, Karsten M. Borgwardt, Malte J. Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *J. Mach. Learn. Res.*, 13(null):723–773, mar 2012.

[18] Arthur Gretton, Karsten M. Borgwardt, Malte J. Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *Journal of Machine Learning Research*, 13(25):723–773, 2012.

[19] Yanming Guo, Yu Liu, Ard Oerlemans, Songyang Lao, Song Wu, and Michael S Lew. Deep learning for visual understanding: A review. *Neurocomputing*, 187:27–48, 2016.

[20] Michael Gutmann, Aapo Hyvärinen, et al. Estimation of unnormalized statistical models without numerical integration. In *Proceedings of the Sixth Workshop on Information Theoretic Methods in Science and Engineering (WITMSE)*. Citeseer, 2013.

[21] Michael Gutmann and Aapo Hyvärinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In Yee Whye Teh and Mike Titterington, editors, *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pages 297–304, Chia Laguna Resort, Sardinia, Italy, 13–15 May 2010. PMLR.

[22] Michael U Gutmann and Aapo Hyvärinen. Noise-contrastive estimation of unnormalized statistical models, with applications to natural image statistics. *Journal of machine learning research*, 13(2), 2012.

[23] Tuomas Haarnoja. Acquiring diverse robot skills via maximum entropy deep reinforcement learning. 2018.

[24] Tuomas Haarnoja, Haoran Tang, Pieter Abbeel, and Sergey Levine. Reinforcement learning with deep energy-based policies. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1352–1361. PMLR, 06–11 Aug 2017.

[25] Tianxing He, Yu Zhang, Jasha Droppo, and Kai Yu. On training bi-directional neural network language model with noise contrastive estimation. In *2016 10th International Symposium on Chinese Spoken Language Processing (ISCSLP)*, pages 1–5, 2016.

[26] Geoffrey E. Hinton. Products of experts. pages 1–6, 1999.

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

[28] Aapo Hyvärinen. Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research*, 6(24):695–709, 2005.

[29] James M. Joyce. *Kullback-Leibler Divergence*, pages 720–722. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011.

[30] Taesup Kim and Yoshua Bengio. Deep directed generative models with energy-based probability estimation, 2016.

[31] Diederik P Kingma and Max Welling. Auto-encoding variational bayes, 2013.

[32] Diederik P. Kingma and Max Welling. An introduction to variational autoencoders. *Foundations and Trends® in Machine Learning*, 12(4):307–392, 2019.

[33] Durk P Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. Improved variational inference with inverse autoregressive flow. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016.

[34] Polina Kirichenko, Pavel Izmailov, and Andrew Gordon Wilson. Why normalizing flows fail to detect out-of-distribution data, 2020.

[35] Ivan Kobyzev, Simon J.D. Prince, and Marcus A. Brubaker. Normalizing flows: An introduction and review of current methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(11):3964–3979, nov 2021.

[36] Frederic Koehler, Viraj Mehta, and Andrej Risteski. Representational aspects of depth and conditioning in normalizing flows. 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 5628–5636. PMLR, 18–24 Jul 2021.

[37] S. Kullback. Certain inequalities in information theory and the cramer-rao inequality. *The Annals of Mathematical Statistics*, 25(4):745–751, 1954.

[38] S. Kullback and R. A. Leibler. On Information and Sufficiency. *The Annals of Mathematical Statistics*, 22(1):79 – 86, 1951.

[39] Rithesh Kumar, Sherjil Ozair, Anirudh Goyal, Aaron Courville, and Yoshua Bengio. Maximum entropy generators for energy-based models, 2019.

[40] Matthieu Labeau and Alexandre Allauzen. An experimental analysis of Noise-Contrastive Estimation: the noise distribution matters. In *15th Conference of the European Chapter of the Association for Computational Linguistics:*, pages 15 – 20, Valencia, Spain, April 2017.

[41] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

[42] Yann Lecun, Sumit Chopra, and Raia Hadsell. *A tutorial on energy-based learning.* 01 2006.

[43] Yann LeCun and Fu Jie Huang. Loss functions for discriminative training of energy-based models. In Robert G. Cowell and Zoubin Ghahramani, editors, *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics*, volume R5 of *Proceedings of Machine Learning Research*, pages 206–213. PMLR, 06–08 Jan 2005. Reissued by PMLR on 30 March 2021.

[44] Erich L. Lehmann and George Casella. *Theory of Point Estimation*. Springer-Verlag, New York, NY, USA, second edition, 1998.

[45] Ruilin Li, Xin Wang, Hongyuan Zha, and Molei Tao. Improving sampling accuracy of stochastic gradient mcmc methods via non-uniform subsampling of gradients, 2020.

[46] Zengyi Li, Yubei Chen, and Friedrich T. Sommer. Learning energy-based models in high-dimensional spaces with multi-scale denoising score matching, 2019.

[47] Bingbin Liu, Elan Rosenfeld, Pradeep Ravikumar, and Andrej Risteski. Analyzing and improving the optimization landscape of noise-contrastive estimation, 2021.

[48] Alexander Ly, Maarten Marsman, Josine Verhagen, Raoul P.P.P. Grasman, and Eric-Jan Wagenmakers. A tutorial on fisher information. *Journal of Mathematical Psychology*, 80:40–55, 2017.

[49] Eric Mazumdar, Aldo Pacchiano, Yian Ma, Michael Jordan, and Peter Bartlett. On approximate thompson sampling with langevin algorithms. In *International Conference on Machine Learning*, pages 6797–6807. PMLR, 2020.

[50] Hongyuan Mei, Tom Wan, and Jason Eisner. Noise-contrastive estimation for multivariate point processes. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 5204–5214. Curran Associates, Inc., 2020.

[51] Andriy Mnih and Koray Kavukcuoglu. Learning word embeddings efficiently with noise-contrastive estimation. 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.

[52] Andriy Mnih and Yee Whye Teh. A fast and simple algorithm for training neural probabilistic language models. 2012.

[53] Frank Nielsen. On the jensen–shannon symmetrization of distances relying on abstract means. *Entropy*, 21(5), 2019.

[54] Liam Paninski. Estimation of entropy and mutual information. *Neural Comput.*, 15(6):1191–1253, jun 2003.

[55] George Papamakarios, Eric Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed, and Balaji Lakshminarayanan. Normalizing flows for probabilistic modeling and inference. 2019.

[56] Seonho Park and Panos M. Pardalos. Deep data density estimation through donsker-varadhan representation, 2021.

[57] Miika Pihlaja, Michael Gutmann, and Aapo Hyvarinen. A family of computationally efficient and simple estimators for unnormalized statistical models, 2012.

[58] Marc' aurelio Ranzato, Christopher Poultney, Sumit Chopra, and Yann Cun. Efficient learning of sparse representations with an energy-based model. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems*, volume 19. MIT Press, 2006.

[59] Marc'Aurelio Ranzato, Y-Lan Boureau, Sumit Chopra, and Yann LeCun. A unified energy-based framework for unsupervised learning. In Marina Meila and Xiaotong Shen, editors, *Proceedings of the Eleventh International Conference on Artificial Intelligence and Statistics*, volume 2 of *Proceedings of Machine Learning Research*, pages 371–379, San Juan, Puerto Rico, 21–24 Mar 2007. PMLR.

[60] Ali Razavi, Aaron Van den Oord, and Oriol Vinyals. Generating diverse high-fidelity images with vq-vae-2. *Advances in neural information processing systems*, 32, 2019.

[61] Danilo Jimenez Rezende and Shakir Mohamed. Variational inference with normalizing flows, 2015.

[62] Benjamin Rhodes, Kai Xu, and Michael U. Gutmann. Telescoping density-ratio estimation, 2020.

[63] Lionel Riou-Durand and Nicolas Chopin. Noise contrastive estimation: asymptotics, comparison with mc-mle, 2018.

[64] Oren Rippel and Ryan Prescott Adams. High-dimensional probability estimation with deep density models, 2013.

[65] Christian P Robert, Víctor Elvira, Nick Tawn, and Changye Wu. Accelerating mcmc algorithms. *Wiley Interdisciplinary Reviews: Computational Statistics*, 10(5):e1435, 2018.

[66] Stanislau Semeniuta, Aliaksei Severyn, and Erhardt Barth. A hybrid convolutional variational autoencoder for text generation. *arXiv preprint arXiv:1702.02390*, 2017.

[67] Francois Septier and Gareth W Peters. Langevin and hamiltonian based sequential mcmc for efficient bayesian filtering in high-dimensional spaces. *IEEE Journal of selected topics in signal processing*, 10(2):312–327, 2015.

[68] K. Sharma, Bhopendra Singh, Edwin Herman, R. Regine, S. Suman Rajest, and Ved P Mishra. Maximum information measure policies in reinforcement learning with deep energy-based model. In *2021 International Conference on Computational Intelligence and Knowledge Economy (ICCIKE)*, pages 19–24, 2021.

[69] Noah A Smith and Jason Eisner. Guiding unsupervised grammar induction using contrastive estimation. In *Proc. of IJCAI Workshop on Grammatical Inference Applications*, pages 73–82, 2005.

[70] Yang Song, Sahaj Garg, Jiaxin Shi, and Stefano Ermon. Sliced score matching: A scalable approach to density and score estimation, 2019.

[71] Yang Song and Diederik P. Kingma. How to train your energy-based models, 2021.

[72] Kevin Swersky, Marc'Aurelio Ranzato, David Buchman, Benjamin M. Marlin, and Nandode Freitas. On autoencoders and score matching for energy based models. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, ICML'11, page 1201–1208, Madison, WI, USA, 2011. Omnipress.

[73] Kou Tanaka, Hirokazu Kameoka, and Kazuho Morikawa. Vae-space: Deep generative model of voice fundamental frequency contours. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5779–5783. IEEE, 2018.

[74] Chenyang Tao, Liqun Chen, Shuyang Dai, Junya Chen, Ke Bai, Dong Wang, Jianfeng Feng, Wenlian Lu, Georgiy Bobashev, and Lawrence Carin. On fenchel mini-max learning. 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. Curran Associates, Inc., 2019.

[75] Joy A. Thomas Thomas M. Cover. *Elements of Information Theory*. Wiley, 1991.

[76] Ilya O Tolstikhin, Bharath K. Sriperumbudur, and Bernhard Schölkopf. Minimax estimation of maximum mean discrepancy with radial kernels. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016.

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

[78] Bin Wang and Zhijian Ou. Improved training of neural trans-dimensional random field language models with dynamic noise-contrastive estimation. In *2018 IEEE Spoken Language Technology Workshop (SLT)*, pages 70–76, 2018.

[79] Bin Wang and Zhijian Ou. Learning neural trans-dimensional random field language models with noise-contrastive estimation. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6134–6138, 2018.

[80] T. Xifara, C. Sherlock, S. Livingstone, S. Byrne, and M. Girolami. Langevin diffusions and the metropolis-adjusted langevin algorithm. *Statistics & Probability Letters*, 91:14–19, aug 2014.

[81] Li-Chia Yang, Szu-Yu Chou, and Yi-Hsuan Yang. Midinet: A convolutional generative adversarial network for symbolic-domain music generation. *arXiv preprint arXiv:1703.10847*, 2017.

[82] Shuangfei Zhai, Yu Cheng, Rogerio Feris, and Zhongfei Zhang. Generative adversarial networks as variational training of energy based models, 2016.

[83] Lily H. Zhang, Mark Goldstein, and Rajesh Ranganath. Understanding failures in out-of-distribution detection with deep generative models, 2021.

[84] Nan Zhang, Shifei Ding, Jian Zhang, and Yu Xue. An overview on restricted boltzmann machines. *Neurocomputing*, 275:1186–1199, 2018.