End-to-end Training of Deep Boltzmann Machines by Unbiased Contrastive Divergence with Local Mode Initialization

Shohei Taniguchi [1] **Masahiro Suzuki** [1] **Yusuke Iwasawa** [1] **Yutaka Matsuo** [1]

Abstract

We address the problem of biased gradient estimation in deep Boltzmann machines (DBMs). The existing method to obtain an unbiased estimator uses a maximal coupling based on a Gibbs sampler, but when the state is high-dimensional, it takes a long time to converge. In this study, we propose to use a coupling based on the Metropolis-Hastings (MH) and to initialize the state around a local mode of the target distribution. Because of the propensity of MH to reject proposals, the coupling tends to converge in only one step with a high probability, leading to high efficiency. We find that our method allows DBMs to be trained in an end-to-end fashion without greedy pretraining. We also propose some practical techniques to further improve the performance of DBMs. We empirically demonstrate that our training algorithm enables DBMs to show comparable generative performance to other deep generative models, achieving the FID score of 10.33 for MNIST.

1. Introduction

Boltzmann machines (Fahlman et al., 1983; Ackley et al., 1985; Hinton et al., 1984; 1986), which are energy-based models defined over binary vectors, have played an important role in the history of deep learning. For example, restricted Boltzmann machines (RBMs), Boltzmann machines with a single hidden layer, have been used for the pretraining of feedforward neural networks (FNNs) (Hinton et al., 2006; Hinton & Salakhutdinov, 2006; Hinton, 2007; Bengio et al., 2006; Ranzato et al., 2006). Although Boltzmann machines are rarely used for pretraining today, they have many applications as undirected generative models, e.g., missing value imputation of incomplete data (Wang & Wu, 2017; Ma, 2020), multimodal learning (Srivastava & Salakhutdinov, 2012), and collaborative filtering (Salakhutdinov et al., 2007). Boltzmann machines also have the potential as powerful generative models because it is known as a universal approximator of the probability mass function on discrete variables (Le Roux & Bengio, 2008). Among them, deep Boltzmann machines (DBMs) (Salakhutdinov & Larochelle, 2010), which are multi-layered undirected models, can capture complex structures by their deep structure while retaining the advantages of the Boltzmann machine.

However, the training of DBMs is known to be quite difficult. The difficulty in training is largely related to the gradient estimation in DBMs. Maximum likelihood learning of DBMs requires to estimate the gradient of their marginal log-likelihood. However, this gradient estimator tends to be biased because it involves several approximations. Specifically, this gradient estimator includes the expectation over the posterior and their joint distribution. Since these distributions are intractable in DBMs, some approximations are required, such as a variational approximation (for the posterior) and a persistent Gibbs sampler (for the joint distribution) (Salakhutdinov & Larochelle, 2010). Such approximations typically lead to biased gradient estimations. Both theoretical and empirical studies have shown that biased gradient estimators harm the convergence property of stochastic approximation algorithms (Carreira-Perpinan & Hinton, 2005; Schulz et al., 2010; Fischer & Igel, 2010). Therefore, it is necessary to seek an unbiased gradient estimator to mitigate the learning difficulties of DBM.

Recently, Jacob et al. (2020) have established a theory of an unbiased Markov chain Monte Carlo (MCMC) method that removes the bias of an MCMC-based estimator using a technique called *couplings*. The unbiased MCMC method enables us to construct an unbiased estimator of some expectation over an intractable random variable in a finite time. Based on this theory, Qiu et al. (2020) have proposed the unbiased contrastive divergence (UCD) algorithm, in which an unbiased gradient estimator of the marginal log-likelihood for energy-based latent variable models like RBMs and DBMs is constructed using a Gibbs-sampler-based coupling. They have shown that the UCD algorithm is empirically effective in avoiding non-convergent behavior

*Equal contribution [1]The University of Tokyo, 7-chōme-3-1 Hongō, Bunkyo City, Tokyo 113-8654, Japan. Correspondence to: Shohei Taniguchi <taniguchi@weblab.t.u-tokyo.ac.jp>.

Proceedings of the 40^{th} International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023.

in the training of RBMs. However, we empirically find that their Gibbs-sampler-based algorithm tends to take a long time to converge, especially in high-dimensional cases, due to the curse of dimensionality. In fact, their experiments are conducted only with relatively low-dimensional RBMs; therefore, the difficulty of applying UCD to DBMs, which tend to be more high-dimensional than RBMs, still remains. To use the UCD algorithm for unbiased estimation in DBMs, we need to establish a method where couplings converge in less time, even in high dimensions.

In this paper, we propose an improved UCD algorithm to efficiently perform unbiased gradient estimation in DBMs. The main insight of the proposed method is to search for an initial state of the Markov chain that reduces the number of coupling steps in UCD. To achieve this, we first change the Markov chain of couplings in UCD from a Gibbs sampler to the Metropolis-Hastings (MH) algorithm. The MH algorithm has a property that it rejects a proposal for the next state with a high probability if the energy of the current state is low. Therefore, we propose to set the initial value of the Markov chain to a state with low energy, i.e., near a *local mode*. This allows us to terminate the Markov chain to obtain an unbiased gradient estimator in just a few steps. More importantly, if the state is high-dimensional, the acceptance probability in the MH is even lower, resulting in termination with fewer coupling steps, such as 1. This means that the proposed method greatly alleviates the problem of the curse of dimensionality in the UCD algorithm.

To explore a local mode, we use the local search algorithm (Hromkovič, 2013), in which the state is iteratively updated from a random initialization by applying local changes to the state until convergence. Although a certain number of steps is additionally required to perform the local search, the proposed method is much more efficient in practice than the original UCD. We name this proposed learning algorithm of DBMs the *unbiased contrastive divergence with local mode initialization* (UCD-LMI).

Furthermore, we find that UCD-LMI allows us to learn DBMs without pretraining. DBMs are known to fail to train from random initialization, which is called the joint training problem (Goodfellow et al., 2016), and a widely known solution to this is greedy layer-wise pretraining. Although there have been attempts to train without pertaining (Montavon & Müller, 2012; Goodfellow et al., 2013), it is still difficult to show good generative performance. Our method shows high-generation performance by end-to-end training without pretraining and thus can be considered a potential solution to the joint training problem.

We also propose several practical techniques, including a simplified centering trick, a marginalization trick to reduce the variance, and orthogonal initialization, needed to successfully train a DBM. We empirically find that these additional techniques are also essential to achieve performance comparable to other DGMs.

In the experiment, we compare our method to existing learning methods of DBMs and RBMs on the image generation benchmarks using the MNIST, Fashion-MNIST, and CIFAR-10 datasets. Our method achieves the FID score of 10.33 for MNIST, which is comparable to the Wasserstein generative adversarial network (WGAN).

2. Preliminaries

2.1. Deep Boltzmann Machine

The deep Boltzmann Machine (DBM) (Salakhutdinov & Larochelle, 2010) is a kind of Boltzmann machine, which has several layers of latent variables. In the case of a deep Boltzmann machine with a visible layer, $\mathbf{v}$, and two hidden layers, $\mathbf{h}^{(1)}$ and $\mathbf{h}^{(2)}$, the joint probability is given by

$$P\left(\mathbf{v}, \mathbf{h}^{(1)}, \mathbf{h}^{(2)}\right) \propto \exp\left(-E(\mathbf{v}, \mathbf{h}^{(1)}, \mathbf{h}^{(2)}; \boldsymbol{\theta})\right). \quad (1)$$

Each unit of $\mathbf{v}$, $\mathbf{h}^{(1)}$, and $\mathbf{h}^{(2)}$ takes a binary value of 0 or 1. The energy function E is defined as follows:

$$\begin{aligned} &E(\mathbf{v}, \mathbf{h}^{(1)}, \mathbf{h}^{(2)}; \boldsymbol{\theta}) \\ &= -\mathbf{v}^\top \boldsymbol{W}^{(1)} \mathbf{h}^{(1)} - {\mathbf{h}^{(1)}}^\top \boldsymbol{W}^{(2)} \mathbf{h}^{(2)}, \end{aligned} \quad (2)$$

where $\boldsymbol{\theta}$ denotes all parameters, i.e., $\boldsymbol{\theta} = \left\{\boldsymbol{W}^{(1)}, \boldsymbol{W}^{(2)}\right\}$, and the bias parameters are omitted in this definition for clarity. Because the DBM has a bipartite structure, we can easily calculate the conditional distribution over the odd layers and the even layers as follows:

$$P(\mathrm{v}_i = 1 \mid \boldsymbol{h}^{(1)}) = \sigma\left(\boldsymbol{W}^{(1)}_{i,:} \boldsymbol{h}^{(1)}\right), \quad (3)$$

$$P(\mathrm{h}^{(1)}_j = 1 \mid \boldsymbol{v}, \boldsymbol{h}^{(2)}) = \sigma\left(\boldsymbol{v}^\top \boldsymbol{W}^{(1)}_{:,j} + \boldsymbol{W}^{(2)}_{j,:} \boldsymbol{h}^{(2)}\right), \quad (4)$$

$$P(\mathrm{h}^{(2)}_k = 1 \mid \boldsymbol{h}^{(1)}) = \sigma\left({\boldsymbol{h}^{(1)}}^\top \boldsymbol{W}^{(2)}_{k,:}\right). \quad (5)$$

We here consider a DBM with two hidden layers for simplicity, but it can be easily extended to deeper models.

Although the Boltzmann machine (including the DBM) is originally developed for binary variables, it can be extended to the case where the visible units are real-valued by slightly modifying the energy function (Welling et al., 2004; Hinton & Salakhutdinov, 2006). This extension is called the Gaussian-Bernoulli Boltzmann machine because its conditional distribution over the visible layer is a Gaussian.

2.2. Training DBM

To learn a DBM by maximum likelihood, we need to calculate the gradient of the log-likelihood with respect to the

parameters $\boldsymbol{\theta}$ as follows.

$$\begin{aligned}\nabla_{\boldsymbol{\theta}} \log p(\boldsymbol{v};\boldsymbol{\theta}) = &- \mathbb{E}\left[\nabla_{\boldsymbol{\theta}} E(\boldsymbol{v}, \mathbf{h}^{(1)}, \mathbf{h}^{(2)};\boldsymbol{\theta})\right] \\ &+ \mathbb{E}\left[\nabla_{\boldsymbol{\theta}} E(\tilde{\mathbf{v}}, \tilde{\mathbf{h}}^{(1)}, \tilde{\mathbf{h}}^{(2)};\boldsymbol{\theta})\right],\end{aligned} \tag{6}$$

$$\mathbf{h}^{(1)}, \mathbf{h}^{(2)} \sim P\left(\mathbf{h}^{(1)}, \mathbf{h}^{(2)} \mid \boldsymbol{v}\right), \tag{7}$$

$$\tilde{\mathbf{v}}, \tilde{\mathbf{h}}^{(1)}, \tilde{\mathbf{h}}^{(2)} \sim P\left(\mathbf{v}, \mathbf{h}^{(1)}, \mathbf{h}^{(2)}\right). \tag{8}$$

However, the expectations with respect to the posterior distribution over the hidden layers $P\left(\mathbf{h}^{(1)}, \mathbf{h}^{(2)} \mid \boldsymbol{v}\right)$ and the joint distribution $P\left(\mathbf{v}, \mathbf{h}^{(1)}, \mathbf{h}^{(2)}\right)$ are both intractable, so we need some approximation. A common way is to use a variational approximation for the posterior and a persistent Gibbs sampler for the joint distribution respectively. In the variational approximation, the posterior $P\left(\mathbf{h}^{(1)}, \mathbf{h}^{(2)} \mid \boldsymbol{v}\right)$ is approximated by a variational approximate distribution $Q\left(\boldsymbol{h}^{(1)}, \boldsymbol{h}^{(2)} \mid \boldsymbol{v}\right)$, and Q is also optimized by minimizing the Kullback–Leibler divergence to the true posterior.

The expectation over the joint distribution is approximated using samples obtained by running some steps of a Gibbs sampling. The Gibbs chain at each gradient step is initialized with their states from the previous gradient step. This approach is called *stochastic maximum likelihood* (SML) (Younes, 1998) or *persistent contrastive divergence* (PCD) (Tieleman, 2008).

2.3. Joint Training Problem

Unfortunately, training a DBM using PCD from a random initialization usually results in failure. This problem is often called the *joint training problem* (Goodfellow et al., 2016). The most popular method for overcoming the joint training problem is greedy layer-wise pretraining, in which each layer of the DBM is trained in isolation as an RBM. Although greedy pretraining is useful for training a DBM properly, it has some undesirable properties, e.g., the complexity of software implementations and the difficulty of tracking the performance during training.

To address them, there have been some attempts to train a DBM without pretraining. For example, the centered DBM (Montavon & Müller, 2012) uses a centering trick, in which the unit values are normalized using moving statistics, enabling accelerated training of DBMs. The multi-prediction DBM (Goodfellow et al., 2013) is another example of DBMs that does not require greedy pretraining, in which the model is trained using the back-propagation algorithm without relying on Markov chain Monte Carlo (MCMC) like Gibbs sampling. Although these methods can alleviate the joint training problem, it is still difficult to achieve good generative performance with DBMs.

What makes the training of DBMs more difficult than RBMs is that the posterior distribution over all the hidden units $P(\mathbf{h} \mid \mathbf{v})$ is intractable for DBMs, whereas it has an analytic form for RBMs. Therefore, training an RBM is much easier than a DBM because the first term of Eq. (6) can be easily approximated using exact samples from the posterior. In the case of a DBM, on the other hand, the posterior is no longer tractable; hence an approximation has to be performed. However, since existing approximation methods (e.g., variational approximation) have bias in their estimation, the resulting algorithm tends not to converge even to local minima of the log-likelihood, leading to poor performance. The problem of biased estimation applies not only to the posterior $P\left(\boldsymbol{h}^{(1)}, \boldsymbol{h}^{(2)} \mid \boldsymbol{v}\right)$ but also to the joint distribution $P\left(\boldsymbol{v}, \boldsymbol{h}^{(1)}, \boldsymbol{h}^{(2)}\right)$, because PCD also has bias as an estimator. We hypothesize that the lack of unbiasedness in the gradient estimator is a critical issue that causes difficulty in training DBMs.

3. Unbiased Gradient Estimation

In this section, we address the joint training problem of DBMs by proposing an algorithm that removes the bias of gradient estimation, enabling stable training of DBMs. First, we introduce a general concept of the unbiased MCMC method. Subsequently, we explain an existing UCD algorithm, which uses the unbiased MCMC for the training of DBMs, and point out its problem. Finally, our UCD-LCI algorithm is derived by addressing the problem.

3.1. Unbiased MCMC with Couplings

Jacob et al. (2020) have proposed an unbiased MCMC method to remove the bias of MCMC-based estimators. Suppose we want to estimate the expectation over some function f in terms of a random variable $\mathbf{x} \sim P(\mathbf{x})$, i.e., $\mathbb{E}[f(\mathbf{x})]$. If a Markov chain $\{\mathbf{x}_t\}$ satisfies $\mathbb{E}[f(\mathbf{x}_t)] \to \mathbb{E}[f(\mathbf{x})]$ as $t \to \infty$, then under some regularity conditions, the expectation can be expressed as a telescoping sum as follows:

$$\mathbb{E}[f(\mathbf{x})] = \mathbb{E}\left[f(\mathbf{x}_0) + \sum_{t=1}^{\infty} f(\mathbf{x}_t) - f(\mathbf{x}_{t-1})\right]. \tag{9}$$

In addition, if we assume that there is another Markov chain $\{\mathbf{y}_t\}$ such that (1) $\mathbf{x}_t$ and $\mathbf{y}_t$ have the same marginal distributions for all $t \geq 0$, and (2) $\mathbf{x}_t = \mathbf{y}_{t-1}$ for all $t \geq \tau \geq 1$, where τ is some random time, the following equation holds:

$$\mathbb{E}[f(\mathbf{x})] = \mathbb{E}\left[f(\mathbf{x}_0) + \sum_{t=1}^{\tau-1} f(\mathbf{x}_t) - f(\mathbf{y}_{t-1})\right]. \tag{10}$$

Therefore, the quantity $f(\mathbf{x}_0) + \sum_{t=1}^{\tau-1} f(\mathbf{x}_t) - f(\mathbf{y}_{t-1})$ is an unbiased estimator for the expectation $\mathbb{E}[f(\mathbf{x})]$. By carefully designing a coupling of the two Markov chains $\{(\mathbf{x}_t, \mathbf{y}_t)\}$, we can obtain an unbiased estimator in a finite time τ. We refer to the random time τ as the *coupling time*.

A key ingredient of the unbiased MCMC algorithm is the construction of the coupling $\{(\mathbf{x}_t, \mathbf{y}_t)\}$. A common way to construct such Markov chains is to use a maximal coupling as provided by Jacob et al. (2020), where the probability of $P(\mathbf{x}_t = \mathbf{y}_{t-1} \mid \mathbf{x}_{t-1}, \mathbf{y}_{t-2})$ is maximized at each step $t \geq 2$. By using the maximal coupling, the coupling time can be shortened; therefore, the unbiased estimator can be obtained efficiently.

3.2. Unbiased Contrastive Divergence

The unbiased MCMC method can be directly applied to the gradient estimation of a DBM in Eq. (6). For example, an unbiased estimator of the second term of Eq. (6) can be obtained as follows:

$$\begin{aligned} &\mathbb{E}\left[\nabla E(\mathbf{x}; \boldsymbol{\theta})\right] \qquad (11) \\ &= \mathbb{E}\left[\nabla E(\mathbf{x}_0; \boldsymbol{\theta}) + \sum_{t=1}^{\tau-1} \nabla E(\mathbf{x}_t; \boldsymbol{\theta}) - \nabla E(\mathbf{y}_{t-1}; \boldsymbol{\theta})\right], \end{aligned}$$

where $\mathbf{x} = (\tilde{\mathbf{v}}, \tilde{\mathbf{h}}^{(1)}, \tilde{\mathbf{h}}^{(2)})$, and $\{(\mathbf{x}_t, \mathbf{y}_t)\}$ is a coupling of Markov chains that satisfies (1) $\mathbb{E}\left[\nabla E\left(\mathbf{x}_t; \boldsymbol{\theta}\right)\right] \rightarrow \mathbb{E}\left[\nabla E\left(\mathbf{x}; \boldsymbol{\theta}\right)\right]$, (2) $\mathbf{x}_t$ and $\mathbf{y}_t$ have the same marginal distributions for all $t \geq 0$, and (3) $\mathbf{x}_t = \mathbf{y}_{t-1}$ after some random time τ. A similar derivation can be applied to the first term of Eq. (6) (see Appendix A).

Qiu et al. (2020) have proposed a method to construct a maximal coupling based on a Gibbs sampler for energy-based latent variable models such as DBMs, and named the resulting learning algorithm the *unbiased contrastive divergence* (UCD). However, we empirically observe that the coupling time of the Gibbs-based coupling tends to be very long, especially when $\mathbf{x}$ is high-dimensional due to the curse of dimensionality. Therefore, it is difficult to apply this method to high-dimensional DBMs.

3.3. UCD with Local Mode Initialization

To apply the UCD algorithm to DBMs, we develop an efficient coupling algorithm to construct an unbiased gradient estimator of the log-likelihood in Eq. (6). Here, we focus on the construction of an unbiased estimator for the second term of Eq. (6), but a similar derivation can be applied to the first term (see Appendix A for more details).

Our proposed coupling is based on the Metropolis–Hastings (MH) algorithm. A Markov chain of MH is constructed using a proposal distribution Q as follows:

$$P\left(\mathbf{x}_t \mid \mathbf{x}_{t-1}\right) = Q\left(\mathbf{x}_t \mid \mathbf{x}_{t-1}\right) A\left(\mathbf{x}_t, \mathbf{x}_{t-1}\right), \qquad (12)$$

$$A\left(\mathbf{x}_t, \mathbf{x}_{t-1}\right) = \min\left(1, \frac{P(\mathbf{x}_t)}{P(\mathbf{x}_{t-1})} \frac{Q(\mathbf{x}_{t-1} \mid \mathbf{x}_t)}{Q(\mathbf{x}_t \mid \mathbf{x}_{t-1})}\right). \qquad (13)$$

To obtain samples from this Markov chain, we first initialize the sample $\boldsymbol{x}_0$, and then repeat iterations of generating a proposal $\boldsymbol{x}'$ according to $Q(\mathbf{x}_t \mid \boldsymbol{x}_{t-1})$ and setting $\boldsymbol{x}_t = \boldsymbol{x}'$ with probability $A\left(\boldsymbol{x}', \boldsymbol{x}_{t-1}\right)$, otherwise $\boldsymbol{x}_t = \boldsymbol{x}_{t-1}$. In our method, we simply use a uniform distribution for the proposal Q. In this case, the acceptance probability is simplified as follows:

$$\begin{aligned} A\left(\mathbf{x}_t, \mathbf{x}_{t-1}\right) &= \min\left(1, P(\mathbf{x}_t)/P(\mathbf{x}_{t-1})\right) \qquad (14) \\ &= \min\left(1, \exp(E(\mathbf{x}_{t-1}; \boldsymbol{\theta}) - E(\mathbf{x}_t; \boldsymbol{\theta}))\right) \end{aligned}$$

In fact, a maximal coupling of the MH algorithm with the uniform proposal distribution can be easily constructed as shown in Algorithm 3 in the appendix (Wang et al., 2021). One might think that this coupling is less efficient than the Gibbs-based maximal coupling because the MH algorithm tends to mix slower than the Gibbs sampler, especially when the proposal is uniformly distributed. However, if we set $\mathbf{x}_0 = \mathbf{y}_0$, and its energy $E(\mathbf{x}_0; \boldsymbol{\theta})$ takes a very low value, the proposal for $\mathbf{x}_1$ is rejected with a very high probability, resulting in $\tau = 1$. This means that by starting from a low-energy state, this MH-based coupling can be very efficient, because the coupling time would be almost always 1. Importantly, this tendency is more pronounced when $\boldsymbol{x}$ is high-dimensional, because the acceptance probability of the MH step gets even lower in such cases. In other words, this method turns the curse of dimensionality in the MH algorithm into an advantage to shorten the coupling time.

A challenge to efficiently perform MH-based maximal coupling is how to find low-energy states for its initialization. To address it, we propose to use a local search algorithm (Battiti et al., 2008). Because the DBM has a bipartite structure as described in Section 2, the conditional minimum of the energy function for the even layers given the odd layers can be easily obtained as follows:

$$\arg\min_{\mathbf{v}} \left\{E \mid \mathbf{h}^{(1)}\right\} = \mathbf{1}_{\boldsymbol{W}^{(1)}\mathbf{h}^{(1)} \geq 0} \qquad (15)$$

$$\arg\min_{\mathbf{h}^{(2)}} \left\{E \mid \mathbf{h}^{(1)}\right\} = \mathbf{1}_{\mathbf{h}^{(1)\top}\boldsymbol{W}^{(2)} \geq 0} \qquad (16)$$

Similarly, the conditional minimum for the odd layers given the even layers is as follows:

$$\arg\min_{\mathbf{h}^{(1)}} \left\{E \mid \mathbf{v}, \mathbf{h}^{(1)}\right\} = \mathbf{1}_{\mathbf{v}^\top \boldsymbol{W}^{(1)} + \boldsymbol{W}^{(2)}\mathbf{h}^{(2)} \geq 0} \qquad (17)$$

As in Gibbs sampling, we iteratively update a sample of $\mathbf{x} = (\tilde{\mathbf{v}}, \tilde{\mathbf{h}}^{(1)}, \tilde{\mathbf{h}}^{(2)})$ by finding the conditional minimum of each block until the sample converges to a local minimum of the energy function, which is equivalent to a local mode of the target distribution $P(\mathbf{v}, \mathbf{h}; \boldsymbol{\theta})$. A concrete algorithm of the local search for DBMs is provided in Algorithm 1. Note that the uniform noise u is used to determine whether to update the even or odd layers first. This local search algorithm converges in finite steps, because the number of possible states is countably finite in binary cases. After finding a

Algorithm 1 Local Search Algorithm for DBM

Draw an initial state $\left(\boldsymbol{v}_0, \boldsymbol{h}_0^{(1)}, \boldsymbol{h}_0^{(2)}\right)$ uniformly.
Draw u uniformly from $[0, 1)$.
$t \leftarrow 0$
repeat
  if $u < 0.5$ **then**
    $\boldsymbol{v}_{t+1} = \mathbf{1}_{\boldsymbol{W}^{(1)}\boldsymbol{h}_t^{(1)} \geq 0},\ \boldsymbol{h}_{t+1}^{(2)} = \mathbf{1}_{\boldsymbol{h}_t^{(1)\top}\boldsymbol{W}^{(2)} \geq 0}$
    $\boldsymbol{h}_{t+1}^{(1)} = \mathbf{1}_{\boldsymbol{v}_{t+1}^{\top}\boldsymbol{W}^{(1)} + \boldsymbol{W}^{(2)}\boldsymbol{h}_{t+1}^{(2)} \geq 0}$
  else
    $\boldsymbol{h}_{t+1}^{(1)} = \mathbf{1}_{\boldsymbol{v}_t^{\top}\boldsymbol{W}^{(1)} + \boldsymbol{W}^{(2)}\boldsymbol{h}_t^{(2)} \geq 0}$
    $\boldsymbol{v}_{t+1} \leftarrow \mathbf{1}_{\boldsymbol{W}^{(1)}\boldsymbol{h}_{t+1}^{(1)} \geq 0},\ \boldsymbol{h}_{t+1}^{(2)} \leftarrow \mathbf{1}_{\boldsymbol{h}_{t+1}^{(1)\top}\boldsymbol{W}^{(2)} \geq 0}$
  end if
  $t \leftarrow t + 1$
until $\left(\boldsymbol{v}_t, \boldsymbol{h}_t^{(1)}, \boldsymbol{h}_t^{(2)}\right) = \left(\boldsymbol{v}_{t-1}, \boldsymbol{h}_{t-1}^{(1)}, \boldsymbol{h}_{t-1}^{(2)}\right)$
$T \leftarrow t$
return $\left(\boldsymbol{v}_T, \boldsymbol{h}_T^{(1)}, \boldsymbol{h}_T^{(2)}\right), T$

local mode, we run a single step of Gibbs sampling from it, and use the moved state as an initial state of Algorithm 3. The reason for running a single Gibbs step is to prevent the support of the initial state distribution of Algorithm 3 from being limited only to the local modes. By this local mode initialization, it can be expected that the initial state of Algorithm 3 has low energy, making the coupling time τ almost always 1, especially when $\boldsymbol{x}$ is high-dimensional.

Using the local search to initialize the MH-based coupling yields an efficient learning algorithm of DBMs, which we name the *unbiased contrastive divergence with local mode initialization* (UCD-LMI). The UCD-LMI is summarized in Algorithm 2. Although the gradient is estimated using a single training example for simplicity, it is possible to use a minibatch of some training examples, and average the estimates to reduce the variance as usual. The parameter update rule of the simple stochastic gradient ascent can also be replaced by another fancy optimizer like Adam (Kingma & Ba, 2014) and AMSGrad (Reddi et al., 2019).

3.4. How to Apply to Non-binary Cases

Because our UCD-LMI heavily relies on the local search algorithm, it is only applicable to the case where all the variable $\mathbf{x}$ is binary. For example, when dealing with image data, it is common to use a DBM of a Gaussian-Bernoulli type (Welling et al., 2004; Hinton & Salakhutdinov, 2006; Liao et al., 2022). Because the visible units are defined as continuous variables in such cases, our method is difficult to be applied. However, this problem can be avoided by binarizing data via preprocessing. In the case of 8-bit RGB images, for instance, each pixel of an image takes an integer value from 0 to 255. The integer value can be transformed

Algorithm 2 UCD-LMI Algorithm

Draw an initial parameter $\boldsymbol{\theta}$.
repeat
  Sample an example $\boldsymbol{v}$ from the training set.
  Run Algorithm 5, and obtain the final value $\boldsymbol{h}_T^{(1)}, \boldsymbol{h}_T^{(2)}$.
  Run a single Gibbs step from $\boldsymbol{h}_T^{(1)}, \boldsymbol{h}_T^{(2)}$, and obtain $\boldsymbol{h}_{T+1}^{(1)}, \boldsymbol{h}_{T+1}^{(2)}$.
  Set $\boldsymbol{h}_0 = \boldsymbol{z}_0 = (\boldsymbol{h}_{T+1}^{(1)}, \boldsymbol{h}_{T+1}^{(2)})$.
  Run Algorithm 4 , and obtain $\{(\boldsymbol{h}_t, \boldsymbol{z}_{t-1})\},\ \tau$.
  Set $\{\boldsymbol{x}_t\} = \{(\boldsymbol{v}, \boldsymbol{h}_t, \boldsymbol{z}_t)\}$
  $\boldsymbol{g} \leftarrow -\left(\nabla E(\boldsymbol{x}_0) + \sum_{t=1}^{\tau-1} \nabla E(\boldsymbol{x}_t) - E(\boldsymbol{y}_{t-1})\right)$
  Run Algorithm 1, and obtain $\tilde{\boldsymbol{v}}_T, \tilde{\boldsymbol{h}}_T^{(1)}, \tilde{\boldsymbol{h}}_T^{(2)}$.
  Run a single Gibbs step from $\tilde{\boldsymbol{v}}_T, \tilde{\boldsymbol{h}}_T^{(1)}, \tilde{\boldsymbol{h}}_T^{(2)}$, and obtain $\tilde{\boldsymbol{v}}_{T+1}, \tilde{\boldsymbol{h}}_{T+1}^{(1)}, \tilde{\boldsymbol{h}}_{T+1}^{(2)}$.
  Set $\tilde{\boldsymbol{x}}_0 = \tilde{\boldsymbol{y}}_0 = (\tilde{\boldsymbol{v}}_{T+1}, \tilde{\boldsymbol{h}}_{T+1}^{(1)}, \tilde{\boldsymbol{h}}_{T+1}^{(2)})$.
  Run Algorithm 3 , and obtain $\{(\tilde{\boldsymbol{x}}_t, \tilde{\boldsymbol{y}}_{t-1})\},\ \tilde{\tau}$.
  $\boldsymbol{g} \leftarrow \boldsymbol{g} + \nabla E(\tilde{\boldsymbol{x}}_0) + \sum_{t=1}^{\tilde{\tau}-1} \nabla E(\tilde{\boldsymbol{x}}_t) - E(\tilde{\boldsymbol{y}}_{t-1})$
  $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \alpha \boldsymbol{g}$
until convergence
return $\boldsymbol{\theta}$

into an 8 dimensional binary vector by expressing it in the binary numerical system (e.g., $123 = 01111011_{(2)}$) without loss of information. By using this binarization as preprocessing, we can directly apply our method to real-world data like natural images.

4. Other Practical Techniques

In this section, we provide some additional techniques to successfully train DBMs using our UCD-LMI algorithm. Basically, these techniques are designed to reduce the variance of the gradient estimates by our UCD-LMI, and accelerate its training.

4.1. Simplified Centering Trick

In the original model definition, each of the visible and hidden units is assumed to take a value of 0 or 1. In this case, a gradient of the energy function in terms of each element of the weights is the product of connected unit values as follows:

$$\nabla_{W_{i,j}^{(1)}} E(\boldsymbol{v}, \boldsymbol{h}^{(1)}, \boldsymbol{h}^{(2)}; \boldsymbol{\theta}) = -v_i h_j^{(1)}, \tag{18}$$

$$\nabla_{W_{j,k}^{(2)}} E(\boldsymbol{v}, \boldsymbol{h}^{(1)}, \boldsymbol{h}^{(2)}; \boldsymbol{\theta}) = -h_j^{(1)} h_k^{(2)}. \tag{19}$$

This means that the gradient for $W_{i,j}^{(1)}$ is always 0 except when both v_i and $h_j^{(1)}$ are 1, and the same applies to $W_{j,k}^{(2)}$. For this reason, under the original model definition, the gradient tends to be sparse, leading to poor convergence.

The centering trick (Montavon & Müller, 2012) is useful to alleviate the problem by normalizing the unit values using moving statistics. However, using moving statistics introduces non-stationarity into the optimization, limiting the advantage of using unbiased MCMC.

To address this issue, we use a simpler strategy, in which each unit value of $\{0, 1\}$ is cast into $\{-1, 1\}$ by mapping 0 into -1 via preprocessing. In this strategy, the gradients in Eqs. (18) and (19) always take nonzero values, alleviating the sparsity of the original definition without using moving statistics. Note that when we adopt this definition, we need to modify the indicator function $\mathbf{1}_{a\geq 0}$ in Algorithm 1 into a sign function, i.e., $\mathrm{sgn}\,(a) \coloneqq 2 \cdot \mathbf{1}_{a\geq 0} - 1$.

4.2. Variance Reduction by Marginalization

Thanks to the bipartite structure of a DBM, marginal distributions over the even layers and the odd layers are both tractable as follows:

$$P\left(\mathbf{v}, \mathbf{h}^{(2)}\right) \propto \exp\left(-E_{\mathrm{even}}(\mathbf{v}, \mathbf{h}^{(2)})\right), \tag{20}$$

$$P\left(\mathbf{h}^{(1)}\right) \propto \exp\left(-E_{\mathrm{odd}}(\mathbf{h}^{(1)})\right), \tag{21}$$

$$\begin{aligned} &E_{\mathrm{even}}(\mathbf{v}, \mathbf{h}^{(2)}) \\ &= -\sum_j \log\cosh(\mathbf{v}^\top \boldsymbol{W}^{(1)}_{:,j} + \boldsymbol{W}^{(2)}_{j,:}\mathbf{h}^{(2)}), \end{aligned} \tag{22}$$

$$\begin{aligned} E_{\mathrm{odd}}(\mathbf{h}^{(1)}) = &-\sum_i \log\cosh(\boldsymbol{W}^{(1)}_{i,:}\mathbf{h}^{(1)}) \\ &-\sum_k \log\cosh({\mathbf{h}^{(1)}}^\top \boldsymbol{W}^{(2)}_{k,:}), \end{aligned} \tag{23}$$

where $\cosh$ is a hyperbolic cosine function, i.e., $\cosh(a) = (\exp(a) + \exp(-a))/2$. Note that we here assume that each unit takes a value of 1 or -1 as described in Section 4.1. Similarly, the marginal posterior distributions over the even and odd layers given the visible layer are also derived as follows:

$$P\left(\mathbf{h}^{(2)} \mid \boldsymbol{v}\right) \propto \exp\left(-E_{\mathrm{even}}(\mathbf{v}, \mathbf{h}^{(2)})\right), \tag{24}$$

$$P\left(\mathbf{h}^{(1)} \mid \boldsymbol{v}\right) \propto \exp\left(-\tilde{E}_{\mathrm{odd}}(\boldsymbol{v}, \mathbf{h}^{(1)})\right), \tag{25}$$

$$\begin{aligned} \tilde{E}_{\mathrm{odd}}(\boldsymbol{v}, \mathbf{h}^{(1)}) = &-\boldsymbol{v}^\top \boldsymbol{W}^{(1)}\mathbf{h}^{(1)} \\ &-\sum_k \log\cosh({\mathbf{h}^{(1)}}^\top \boldsymbol{W}^{(2)}_{k,:}), \end{aligned} \tag{26}$$

Based on these facts, Eq. (6) can be rewritten as follows:

$$\begin{aligned} &\nabla_{\boldsymbol{\theta}} \log p(\boldsymbol{v}; \boldsymbol{\theta}) \\ &= -\frac{1}{2}\,\mathbb{E}\left[\nabla_{\boldsymbol{\theta}}(E_{\mathrm{even}}(\boldsymbol{v}, \mathbf{h}^{(2)}; \boldsymbol{\theta}) + \tilde{E}_{\mathrm{odd}}(\boldsymbol{v}, \mathbf{h}^{(1)}; \boldsymbol{\theta}))\right] \\ &\quad + \frac{1}{2}\,\mathbb{E}\left[\nabla_{\boldsymbol{\theta}}(E_{\mathrm{even}}(\tilde{\mathbf{v}}, \tilde{\mathbf{h}}^{(2)}; \boldsymbol{\theta}) + E_{\mathrm{odd}}(\tilde{\mathbf{h}}^{(1)}; \boldsymbol{\theta}))\right], \end{aligned} \tag{27}$$

Using this expression for constructing an unbiased estimator with MH-based coupling as described before, we can even reduce its variance, keeping the estimator unbiased.

4.3. Weight Initialization with Orthogonal Matrix

Aside from the learning algorithm, initialization of the weight matrices is also a key to the successful training of a DBM. A common way of initialization is to use random values chosen from a zero-mean Gaussian with a small variance (e.g., 0.01^2) (Hinton, 2012). However, It is more natural to change the scale of weight values depending on the dimensionality of the visible and hidden units. In the context of initialization for FNNs, adaptively scaling methods proposed by Glorot & Bengio (2010) and He et al. (2015) are widely used, but these are not designed for DBMs.

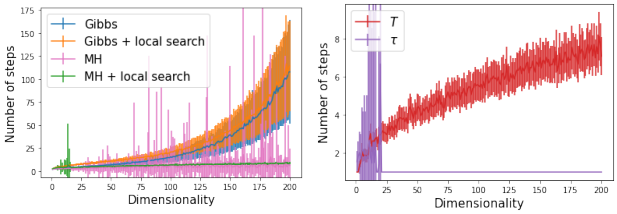
As an alternative, we propose to initialize the weights of a DBM with random (semi-)orthogonal matrices. Since the orthogonal matrix has a property that its inverse is equal to its transpose, i.e., $\boldsymbol{W}^\top \boldsymbol{W} = \boldsymbol{W}\boldsymbol{W}^\top = \boldsymbol{I}$, it is a natural choice for a model with a bidirectional nature like a DBM. Orthogonal initialization is often used for FNNs (Saxe et al., 2013) also, and a way to create uniform random samples of orthogonal matrices is known (Mezzadri, 2006); hence we directly apply it to the case of DBMs. How to initialize the bias parameters is provided in Appendix B.

4.4. Sampling in Test Time

When we perform sampling from a trained DBM, a common way is to run Gibbs sampling starting from random noise for finite steps and treat the final state of the visible unit as a sample from the model. However, we empirically find that it is difficult to obtain good samples in that way because Gibbs sampling is typically suffered from the curse of dimensionality. Instead, as in the training phase, we first use the local search algorithm to find a local minimum of the energy function, and then perform the MH algorithm for a few steps to obtain the final state as a sample. In practice, we can skip running the MH algorithm and just use the local minimum as a sample, because almost all proposals in the MH algorithm are rejected, and the sample hardly moves from the local minimum.

5. Experiment

In the experiments, we first verify that our MH-based coupling algorithm in the UCD-LMI is efficient compared to existing Gibbs-based coupling in the vanilla UCD using toy examples. Subsequently, we validate the generative performance of DBMs trained with our method using MNIST and CIFAR-10 datasets, and compare it with existing training methods of DBMs and other deep generative models.



(a) Coupling comparison (b) Respective # of steps

Figure 1. Efficiency comparison of coupling methods for unbiased gradient estimation. (a) We compare the total steps of local search and coupling (i.e., $\tau + T$ in Algorithms 3 and 1) between Gibbs-based coupling and our MH-based coupling when changing the dimensionality from 1 to 200. (b) We also provide the respective number of steps of T and τ for our MH-based coupling.

5.1. Toy Examples

In this experiment, we prepare randomly initialized RBMs, i.e., single-layer DBMs, and run some coupling algorithms to estimate the gradient of its partition function, i.e., the second term of Eq. (6), to compare their efficiency under various conditions. We change the dimensionality of the visible units and the hidden units in the range from 1 to 200. The weights of RBMs are initialized with random orthogonal matrices.

We compare our MH-based method with Gibbs-based maximal coupling proposed by Qiu et al. (2020). When performing Gibbs-based coupling, we simply initialize the state with uniform noise. To investigate the effect of the initialization by local search, we also try the cases where Gibbs-based coupling is initialized by local search and MH-based coupling is initialized by uniform noise. For a fair comparison, we compare the total time steps it takes to obtain the estimator, including the steps of local search.

The results are shown in Figure 1. We observe that the coupling time of the Gibbs-based method grows exponentially with dimensionality. In the training of DBMs for real-world data (e.g., images), we need to handle much more high-dimensional data; hence these results show that the Gibbs-based method is no longer practical for such cases. Unfortunately, initializing Gibbs-based coupling with a local mode does not help to reduce the coupling time.

On the other hand, it can be seen that our MH-based method works efficiently even for high-dimensional cases. When the state is initialized with uniform noise, the variance of $\tau + T$ is very high; hence the local mode initialization (LMI) contributes to variance reduction. We also observe that the coupling time τ is always 1 for MH-based coupling with the LMI when $d \geq 25$. Although the number of local search steps also increases with dimensionality, the increase is gradual compared to the Gibbs-based coupling, and the variance is also much smaller in high dimensions. These results demonstrate the efficiency of our coupling method to be practically used for the gradient estimator in the training of DBMs for high-dimensional data (e.g., natural images).

Table 1. Quantitative results of the image generation for MNIST and CIFAR-10. We report the Fréchet Inception Distance (FID) score. Lower is better.

Model	FID
MNIST	
VAE	16.14
2sVAE (Dai & Wipf, 2019)	12.60
PixelCNN++ (Salimans et al., 2017)	11.38
WGAN (Arjovsky et al., 2017)	10.28
NVAE (Vahdat & Kautz, 2020)	7.93
GB-RBM (Liao et al., 2022) (Gibbs)	47.53
GB-RBM (Liao et al., 2022) (Gibbs-Langevin)	17.49
Centered DBM (Montavon & Müller, 2012)	89.10
Unbiased DBM (ours)	10.33
CIFAR-10	
PixelCNN (Van Den Oord et al., 2016)	65.93
DCGAN (Radford et al., 2015)	37.11
WGAN-GP (Gulrajani et al., 2017)	36.40
Unbiased DBM (ours)	36.58

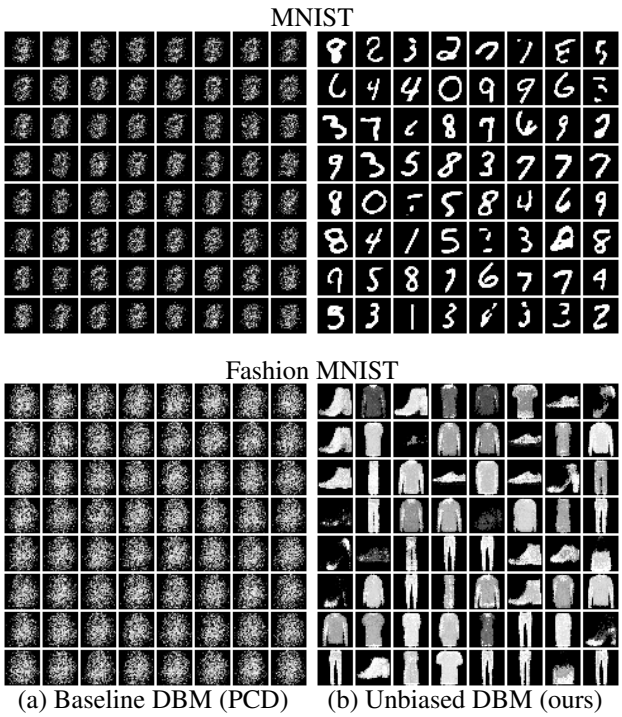
5.2. Image Generation

We train DBMs for image datasets including MNIST (Deng, 2012), Fashin-MNIST (Xiao et al., 2017) and CIFAR-10. We do not use greedy layer-wise pretraining, which we find that is not necessary for our UCD-LMI algorithm. The implementation details are provided in Appendix C. We compare their performance with other existing deep generative models (DGMs), including variational autoencoders (VAEs), generative adversarial networks (GANs), and autoregressive models. For MNIST, We also compare our method with the centered DBM (Montavon & Müller, 2012) as another way to train a DBM without pretraining. Unfortunately, the original UCD algorithm does not work for this setting because its coupling time is extremely long; hence we omit it from the comparison. On the other hand, we observe that the coupling time of our UCD-LMI falls within the range from 5 to 30 for all datasets, showing that our algorithm can withstand practical use.

The quantitative result using the Fréchet Inception Distance (FID) score for MNIST and CIFAR-10 is summarized in Table 1. Our method is labeled as "Unbiased DBM" in the table. It can be seen that our unbiased DBM achieves FID scores comparable to other DGMs, such as WGAN and WGAN-GP. For MNIST, we also provide the ablation study in Table 2 to demonstrate the effectiveness of each component of our method, including the unbiased gradient estima-

Table 2. Ablation study using the MNIST dataset.

Configuration				
Unbiased gradient	Simplified centering	Marginalization	Orthogonal initialization	FID
✓	✓	✓	✓	10.33
	✓	✓	✓	112.1
✓		✓	✓	30.27
✓	✓		✓	13.70
✓	✓	✓		18.20



(a) Baseline DBM (PCD) (b) Unbiased DBM (ours)

Figure 2. Generated samples of MNIST and Fashion-MNIST by a DBM trained with (a) a baseline method using PCD and (b) our UCD-LMI algorithm.

tion, the simplified centering trick, the variance reduction by marginalization, and the orthogonal initialization. When we train DBMs without the unbiased gradient estimator, the generative performance drastically drops, demonstrating that it is crucial for proper training of DBMs without greedy layer-wise pretraining. It can be observed that other techniques are also effective, while the improvement in FID scores is not so great compared to the importance of the unbiased gradient.

Figure 2 shows a comparison of generated samples of MNIST and Fashion-MNIST by trained DBMs using the vanilla PCD and our UCD-LMI without greedy pretraining. While the DBM trained with the vanilla PCD completely fails to learn data structure and generates meaningless samples, our UCD-LMI successfully generates clean samples.

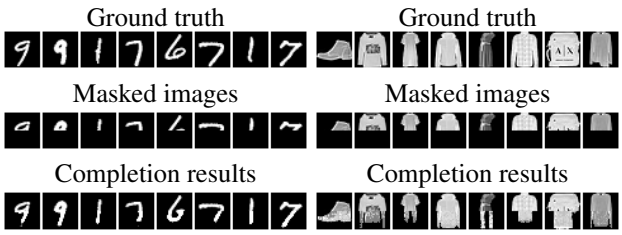


Figure 3. Completion

One of the advantages of DBMs over other DGMs is that DBMs can also be used for missing value imputation of incomplete data. Figure 3 shows examples of completed images of MNIST and Fashion-MNIST by a trained DBM with our UCD-LMI. We create incomplete images by masking the lower half pixels of the original images. When performing the completion, we run the local search algorithm to find a low-energy state given the incomplete data, and fill in the missing values using the obtained low-energy state. It can be seen that the completed images are close to but not exactly the same as the ground truth because of the uncertainty.

6. Conclusion

In this paper, we revisit the difficulty in training deep Boltzmann machines (DBMs), and propose a practical algorithm to obtain unbiased gradient estimates using the coupling technique based on the Metropolis–Hastings and the local search (Section 3). We also provide some practical techniques to successfully train a DBM (Section 4).

In the empirical study, we first demonstrate that our MH-based coupling is much more efficient than the existing Gibbs-based coupling (Section 5.1). We also perform experiments of image generation, and show that DBMs trained with our method achieves comparable performance to other DGMs in terms of the FID score (Section 5.2).

One of the remaining challenges of the DBM is that it is still difficult to be applied for extremely high-dimensional inputs like high-resolution images, because the weights are fully connected between layers, resulting in the growth of

the number of parameters. By applying our UCD-LMI algorithm to locally-connected DBMs (e.g., convolutional Boltzmann machines (Lee et al., 2009)), the problem could be alleviated, which is an important direction of future work.

Acknowledgements

This work was supported by JSPS KAKENHI Grant Number JP21J22342 and the Mohammed bin Salman Center for Future Science and Technology for Saudi-Japan Vision 2030 at The University of Tokyo (MbSC2030). Computational resources of AI Bridging Cloud Infrastructure (ABCI) provided by National Institute of Advanced Industrial Science and Technology (AIST) were used for the experiments.

References

Ackley, D. H., Hinton, G. E., and Sejnowski, T. J. A learning algorithm for boltzmann machines. *Cognitive science*, 9 (1):147–169, 1985.

Arjovsky, M., Chintala, S., and Bottou, L. Wasserstein generative adversarial networks. In *International conference on machine learning*, pp. 214–223. PMLR, 2017.

Battiti, R., Brunato, M., and Mascia, F. *Reactive search and intelligent optimization*, volume 45. Springer Science & Business Media, 2008.

Bengio, Y., Lamblin, P., Popovici, D., and Larochelle, H. Greedy layer-wise training of deep networks. *Advances in neural information processing systems*, 19, 2006.

Carreira-Perpinan, M. A. and Hinton, G. On contrastive divergence learning. In *International workshop on artificial intelligence and statistics*, pp. 33–40. PMLR, 2005.

Dai, B. and Wipf, D. Diagnosing and enhancing vae models. *arXiv preprint arXiv:1903.05789*, 2019.

Deng, L. The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012.

Fahlman, S. E., Hinton, G. E., and Sejnowski, T. J. Massively parallel architectures for al: Netl, thistle, and boltzmann machines. In *National Conference on Artificial Intelligence, AAAI*, 1983.

Fischer, A. and Igel, C. Empirical analysis of the divergence of gibbs sampling based learning algorithms for restricted boltzmann machines. In *International conference on artificial neural networks*, pp. 208–217. Springer, 2010.

Glorot, X. and Bengio, Y. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pp. 249–256. JMLR Workshop and Conference Proceedings, 2010.

Goodfellow, I., Mirza, M., Courville, A., and Bengio, Y. Multi-prediction deep boltzmann machines. *Advances in Neural Information Processing Systems*, 26, 2013.

Goodfellow, I., Bengio, Y., and Courville, A. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., and Courville, A. C. Improved training of wasserstein gans. *Advances in neural information processing systems*, 30, 2017.

He, K., Zhang, X., Ren, S., and Sun, J. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pp. 1026–1034, 2015.

Hinton, G. E. To recognize shapes, first learn to generate images. *Progress in brain research*, 165:535–547, 2007.

Hinton, G. E. A practical guide to training restricted boltzmann machines. In *Neural networks: Tricks of the trade*, pp. 599–619. Springer, 2012.

Hinton, G. E. and Salakhutdinov, R. R. Reducing the dimensionality of data with neural networks. *science*, 313 (5786):504–507, 2006.

Hinton, G. E., Sejnowski, T. J., and Ackley, D. H. *Boltzmann machines: Constraint satisfaction networks that learn*. Carnegie-Mellon University, Department of Computer Science Pittsburgh, PA, 1984.

Hinton, G. E., Sejnowski, T. J., et al. Learning and relearning in boltzmann machines. *Parallel distributed processing: Explorations in the microstructure of cognition*, 1 (282-317):2, 1986.

Hinton, G. E., Osindero, S., and Teh, Y.-W. A fast learning algorithm for deep belief nets. *Neural computation*, 18 (7):1527–1554, 2006.

Hromkovič, J. *Algorithmics for hard problems: introduction to combinatorial optimization, randomization, approximation, and heuristics*. Springer Science & Business Media, 2013.

Jacob, P. E., O'Leary, J., and Atchadé, Y. F. Unbiased markov chain monte carlo methods with couplings. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 82(3):543–600, 2020.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Langley, P. Crafting papers on machine learning. In Langley, P. (ed.), *Proceedings of the 17th International Conference on Machine Learning (ICML 2000)*, pp. 1207–1216, Stanford, CA, 2000. Morgan Kaufmann.

Le Roux, N. and Bengio, Y. Representational power of restricted boltzmann machines and deep belief networks. *Neural computation*, 20(6):1631–1649, 2008.

Lee, H., Grosse, R., Ranganath, R., and Ng, A. Y. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In *Proceedings of the 26th annual international conference on machine learning*, pp. 609–616, 2009.

Liao, R., Kornblith, S., Ren, M., Fleet, D. J., and Hinton, G. Gaussian-bernoulli rbms without tears. *arXiv preprint arXiv:2210.10318*, 2022.

Ma, W. Restricted boltzmann machine for missing data imputation in biomedical datasets. *HKU Theses Online (HKUTO)*, 2020.

Mezzadri, F. How to generate random matrices from the classical compact groups. *arXiv preprint math-ph/0609050*, 2006.

Montavon, G. and Müller, K.-R. Deep boltzmann machines and the centering trick. In *Neural networks: tricks of the trade*, pp. 621–637. Springer, 2012.

Qiu, Y., Zhang, L., and Wang, X. Unbiased contrastive divergence algorithm for training energy-based latent variable models. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=r1eyceSYPr.

Radford, A., Metz, L., and Chintala, S. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

Ranzato, M., Poultney, C., Chopra, S., and Cun, Y. Efficient learning of sparse representations with an energy-based model. *Advances in neural information processing systems*, 19, 2006.

Reddi, S. J., Kale, S., and Kumar, S. On the convergence of adam and beyond. *arXiv preprint arXiv:1904.09237*, 2019.

Salakhutdinov, R. and Larochelle, H. Efficient learning of deep boltzmann machines. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pp. 693–700. JMLR Workshop and Conference Proceedings, 2010.

Salakhutdinov, R., Mnih, A., and Hinton, G. Restricted boltzmann machines for collaborative filtering. In *Proceedings of the 24th international conference on Machine learning*, pp. 791–798, 2007.

Salimans, T., Karpathy, A., Chen, X., and Kingma, D. P. Pixelcnn++: Improving the pixelcnn with discretized logistic mixture likelihood and other modifications. *arXiv preprint arXiv:1701.05517*, 2017.

Saxe, A. M., McClelland, J. L., and Ganguli, S. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. *arXiv preprint arXiv:1312.6120*, 2013.

Schulz, H., Müller, A., Behnke, S., et al. Investigating convergence of restricted boltzmann machine learning. In *NIPS 2010 Workshop on Deep Learning and Unsupervised Feature Learning*, volume 1, pp. 6–1, 2010.

Srivastava, N. and Salakhutdinov, R. R. Multimodal learning with deep boltzmann machines. *Advances in neural information processing systems*, 25, 2012.

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

Vahdat, A. and Kautz, J. Nvae: A deep hierarchical variational autoencoder. *Advances in Neural Information Processing Systems*, 33:19667–19679, 2020.

Van Den Oord, A., Kalchbrenner, N., and Kavukcuoglu, K. Pixel recurrent neural networks. In *International conference on machine learning*, pp. 1747–1756. PMLR, 2016.

Wang, G., O'Leary, J., and Jacob, P. Maximal couplings of the metropolis-hastings algorithm. In *International Conference on Artificial Intelligence and Statistics*, pp. 1225–1233. PMLR, 2021.

Wang, Z. and Wu, Q. Shape completion using deep boltzmann machine. *Computational Intelligence and Neuroscience*, 2017, 2017.

Welling, M., Rosen-Zvi, M., and Hinton, G. E. Exponential family harmoniums with an application to information retrieval. *Advances in neural information processing systems*, 17, 2004.

Xiao, H., Rasul, K., and Vollgraf, R. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

Younes, L. Stochastic gradient estimation strategies for markov random fields. In *Bayesian inference for inverse problems*, volume 3459, pp. 315–325. SPIE, 1998.

Algorithm 3 Maximal Coupling of Metropolis-Hastings Algorithm with Uniform Proposal

Draw $\boldsymbol{x}' \sim \mathrm{Unf}(\boldsymbol{x}')$.
$A_x \leftarrow \min\left(1, \exp(E(\boldsymbol{x}_0; \boldsymbol{\theta}) - E(\boldsymbol{x}'; \boldsymbol{\theta}))\right)$
Draw u uniformly from $[0, 1)$.
if $u < A_x$ **then** $\boldsymbol{x}_1 \leftarrow \boldsymbol{x}'$ **else** $\boldsymbol{x}_1 \leftarrow \boldsymbol{x}_0$
$t \leftarrow 1$
repeat
  Draw $\boldsymbol{x}' \sim \mathrm{Unf}(\boldsymbol{x}')$.
  $A_x \leftarrow \min\left(1, \exp(E(\boldsymbol{x}_t; \boldsymbol{\theta}) - E(\boldsymbol{x}'; \boldsymbol{\theta}))\right)$
  $A_y \leftarrow \min\left(1, \exp(E(\boldsymbol{y}_{t-1}; \boldsymbol{\theta}) - E(\boldsymbol{x}'; \boldsymbol{\theta}))\right)$
  Draw u uniformly from $[0, 1)$.
  if $u < A_x$ **then** $\boldsymbol{x}_{t+1} \leftarrow \boldsymbol{x}'$ **else** $\boldsymbol{x}_{t+1} \leftarrow \boldsymbol{x}_t$
  if $u < A_y$ **then** $\boldsymbol{y}_t \leftarrow \boldsymbol{x}'$ **else** $\boldsymbol{y}_t \leftarrow \boldsymbol{y}_{t-1}$
  $t \leftarrow t + 1$
until $\boldsymbol{x}_t = \boldsymbol{y}_{t-1}$
$\tau \leftarrow t$
return $\{(\boldsymbol{x}_t, \boldsymbol{y}_t)\}, \tau$

Algorithm 4 Maximal Coupling of Metropolis–Hastings Algorithm for Posterior $P(\boldsymbol{h} \mid \boldsymbol{v})$

Draw $\boldsymbol{x}' \sim \mathrm{Unf}(\boldsymbol{h}')$.
$A_h \leftarrow \min\left(1, \exp(E(\boldsymbol{v}, \boldsymbol{h}_0; \boldsymbol{\theta}) - E(\boldsymbol{v}, \boldsymbol{h}'; \boldsymbol{\theta}))\right)$
Draw u uniformly from $[0, 1)$.
if $u < A_h$ **then** $\boldsymbol{h}_1 \leftarrow \boldsymbol{h}'$ **else** $\boldsymbol{h}_1 \leftarrow \boldsymbol{h}_0$
$t \leftarrow 1$
repeat
  Draw $\boldsymbol{h}' \sim \mathrm{Unf}(\boldsymbol{h}')$.
  $A_h \leftarrow \min\left(1, \exp(E(\boldsymbol{v}, \boldsymbol{h}_t; \boldsymbol{\theta}) - E(\boldsymbol{v}, \boldsymbol{h}'; \boldsymbol{\theta}))\right)$
  $A_z \leftarrow \min\left(1, \exp(E(\boldsymbol{z}_{t-1}; \boldsymbol{\theta}) - E(\boldsymbol{h}'; \boldsymbol{\theta}))\right)$
  Draw u uniformly from $[0, 1)$.
  if $u < A_h$ **then** $\boldsymbol{h}_{t+1} \leftarrow \boldsymbol{h}'$ **else** $\boldsymbol{h}_{t+1} \leftarrow \boldsymbol{h}_t$
  if $u < A_z$ **then** $\boldsymbol{z}_t \leftarrow \boldsymbol{h}'$ **else** $\boldsymbol{z}_t \leftarrow \boldsymbol{z}_{t-1}$
  $t \leftarrow t + 1$
until $\boldsymbol{h}_t = \boldsymbol{z}_{t-1}$
$\tau \leftarrow t$
return $\{(\boldsymbol{h}_t, \boldsymbol{z}_t)\}, \tau$

A. Unbiased Estimator for the First Term of Eq. (6)

Similarly to Eq. (11), the unbiased estimator for the first term of Eq. (6) can be constructed as follows:

$$\begin{aligned} &\mathbb{E}\left[\nabla E(\boldsymbol{v}, \mathbf{h}; \boldsymbol{\theta})\right] \\ &= \mathbb{E}\left[\nabla E(\boldsymbol{v}, \mathbf{h}_0; \boldsymbol{\theta}) + \sum_{t=1}^{\tau-1} \nabla E(\boldsymbol{v}, \mathbf{h}_t; \boldsymbol{\theta}) - \nabla E(\boldsymbol{v}, \mathbf{z}_{t-1}; \boldsymbol{\theta})\right], \end{aligned} \tag{28}$$

, where $\mathbf{h} = (\mathbf{h}^{(1)}, \mathbf{h}^{(2)})$, and $\{(\mathbf{h}_t, \mathbf{z}_t)\}$ is a coupling of Markov chains that satisfies (1) $\mathbb{E}\left[\nabla E\left(\boldsymbol{v}, \mathbf{h}_t; \boldsymbol{\theta}\right)\right] \rightarrow \mathbb{E}\left[\nabla E\left(\boldsymbol{v}, \mathbf{h}; \boldsymbol{\theta}\right)\right]$, (2) $\mathbf{h}_t$ and $\mathbf{z}_t$ have the same marginal distributions for all $t \geq 0$, and (3) $\mathbf{h}_t = \mathbf{z}_{t-1}$ after some random time τ. A maximal coupling based on Metroplis-Hastings algorithm for this estimator is provided in Algorithm 4. To initialize this coupling with a state around a local mode of the posterior $P(\mathbf{h} \mid \boldsymbol{v})$, we use the local search as described in Algorithm 5.

Algorithm 5 Local Search Algorithm for Posterior $P(\boldsymbol{h} \mid \boldsymbol{v})$

Draw an initial state $\left(\boldsymbol{h}_0^{(1)}, \boldsymbol{h}_0^{(2)}\right)$ uniformly.
Draw u uniformly from $[0, 1)$.
$t \leftarrow 0$
repeat
  if $u < 0.5$ **then**
    $\boldsymbol{h}_{t+1}^{(2)} = \mathbf{1}_{\boldsymbol{h}_t^{(1)\top} \boldsymbol{W}^{(2)} \geq 0}$
    $\boldsymbol{h}_{t+1}^{(1)} = \mathbf{1}_{\boldsymbol{v}^\top \boldsymbol{W}^{(1)} + \boldsymbol{W}^{(2)} \boldsymbol{h}_{t+1}^{(2)} \geq 0}$
  else
    $\boldsymbol{h}_{t+1}^{(1)} = \mathbf{1}_{\boldsymbol{v}^\top \boldsymbol{W}^{(1)} + \boldsymbol{W}^{(2)} \boldsymbol{h}_t^{(2)} \geq 0}$
    $\boldsymbol{h}_{t+1}^{(2)} \leftarrow \mathbf{1}_{\boldsymbol{h}_{t+1}^{(1)\top} \boldsymbol{W}^{(2)} \geq 0}$
  end if
  $t \leftarrow t + 1$
until $\left(\boldsymbol{h}_t^{(1)}, \boldsymbol{h}_t^{(2)}\right) = \left(\boldsymbol{h}_{t-1}^{(1)}, \boldsymbol{h}_{t-1}^{(2)}\right)$
$T \leftarrow t$
return $\left(\boldsymbol{h}_T^{(1)}, \boldsymbol{h}_T^{(2)}\right), T$

B. Initialization of Bias Parameters

We empirically observe that the initialization of bias parameters is also important especially when we use the simplified centering trick in Section 4.1. A common way to initialize the bias parameters is simply setting 0. However, if we use the centering trick and set the bias zero, the energy function will be symmetric for the unit flip, i.e. $E(\boldsymbol{v}, \boldsymbol{h}; \boldsymbol{\theta}) = E(-\boldsymbol{v}, -\boldsymbol{h}; \boldsymbol{\theta})$. To avoid this, we initialize the bias parameters by sampling from a logistic distribution $\mathrm{Logistic}(0, 0.5)$, which we find works well in practice.

C. Implementation Details

In the experiment of image generation, we set the number of hidden units per layer equal to the visible units for MNIST and Fashion-MNIST, which means it is set to $6,272$ $(= 1 \times 28^2 \times 8)$. For CIFAR-10, we set it to $10,000$. We use the SGD as an optimizer and set the learning rate to 1×10^{-3}. We run $100,000$ gradient steps of training for both datasets. The implementation is available at `https://github.com/iShohei220/unbiased_dbm`.