Can RBMs be trained with zero step contrastive divergence?

Charles K. Fisher*
Unlearn.AI, Inc., 75 Hawthorne St. Ste 560, San Francisco, CA 94105
(Dated: November 7, 2022)

Restricted Boltzmann Machines (RBMs) are probabilistic generative models that can be trained by maximum likelihood in principle, but are usually trained by an approximate algorithm called Contrastive Divergence (CD) in practice. In general, a CD-k algorithm estimates an average with respect to the model distribution using a sample obtained from a k-step Markov Chain Monte Carlo Algorithm (e.g., block Gibbs sampling) starting from some initial configuration. Choices of k typically vary from 1 to 100. This technical report explores if it's possible to leverage a simple approximate sampling algorithm with a modified version of CD in order to train an RBM with k=0. As usual, the method is illustrated on MNIST.

Restricted Boltzmann Machines (RBMs) are probabilistic generative models composed of two layers of neurons (called the visible and hidden layers) with undirected connections between them [1]. Because RBMs are energy based models, it's relatively easy to derive the gradient of the log-likelihood function. Unfortunately, computing the gradient requires one to calculate averages with respect to the model distribution, which are typically intractable. As a result, RBMs are usually trained using approximate methods in which the required averages are estimated using Markov Chain Monte Carlo (MCMC) methods such as block Gibbs sampling [2, 3, 9].

Liao et al [6] recently reported that a few changes to the typical procedure for training RBMs significantly improved the performance of RBMs with Gaussian visible units on image generation tasks. First, Liao et al replace the exact Gibbs sampling step of the visible units with an approximate sampling method based on Langevin dynamics. Second, they initialize the samples used to compute the negative phase of the gradient with an isotropic Gaussian noise such that sampling the visible units can be viewed as type of denoising algorithm.

A gradient step of an RBM aims to decrease the energy of the observed samples (i.e., the positive phase) and increase the energy of the samples drawn from the model (i.e., the negative phase). Typically, one raises the energy of samples drawn from an approximation of the model distribution using something like k-step Contrastive Divergence (CD-k) with a large number of MCMC steps, but does this approximation actually need to be good? To explore this question, this technical report asks whether it's possible to train an RBM with a version of CD-0.

To be concrete, I will focus on RBMs with discrete visible and hidden units. I use the physics convention with Ising-like neurons that are ± 1 rather than Bernoulli units that are $(0, 1)$, but the two model types are theoretically the same as they are related by a simple linear change of variables. The joint energy function of this model is,

$$U(\mathbf{v}, \mathbf{h}) = -\mathbf{b}^T(\mathbf{v} - \boldsymbol{\mu}) - (\mathbf{v} - \boldsymbol{\mu})^T W \mathbf{h} \,. \tag{1}$$

Note that I have centered the visible units about their empirical mean $\boldsymbol{\mu} = \langle \mathbf{v} \rangle_d$, and I have not included a separate bias on the hidden units; though, that can easily be done. Neglecting the explicit bias on the hidden units effectively sets the bias to $W^T \boldsymbol{\mu}$, which works fine on binary MNIST.

The probability distribution of the visible units:

$$p(\mathbf{v}) = \frac{\sum_H e^{-U(\mathbf{v},\mathbf{h})}}{\sum_V \sum_H e^{-U(\mathbf{v},\mathbf{h})}} , \tag{2}$$

in which I use the notation $\sum_V$ and $\sum_H$ to denote sums over all allowed values of the visible and hidden units. Thus, the negative log-likelihood is,

$$\mathcal{L}(\mathbf{b}, W) = -\langle \log p(\mathbf{v}) \rangle_d \,. \tag{3}$$

As usual, one fits the model by attempting to minimize the negative log-likelihood.

SGD:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{b}} = \langle \mathbf{v} \rangle_d - \langle \mathbf{v} \rangle_m \,, \tag{4}$$

$$\frac{\partial \mathcal{L}}{\partial W} = \langle \mathbf{v} \tanh(W(\mathbf{v} - \boldsymbol{\mu}))^T \rangle_d - \langle \mathbf{v} \tanh(W(\mathbf{v} - \boldsymbol{\mu}))^T \rangle_m \,. \tag{5}$$

The expressions for the gradients contain two types of averages, one taken over the samples in the training dataset and one taken with respect to the distribution defined by the model itself. The former average is easy to calculate, but the latter is intractable.

In order to calculate averages with respect to the model distribution, i.e. $\langle f(\mathbf{v}) \rangle_m$, one needs to draw samples from the marginal distribution $p(\mathbf{v})$. Typically, this is

* drckf@unlearn.ai

done through block Gibbs sampling [2]. To perform block GS , we start with an initial hidden state $\mathbf{h}_0$ and then iteratively draw from $p(\mathbf{v}_i|\mathbf{h}_{i-1})$ and $p(\mathbf{h}_i|\mathbf{v}_i)$ for $i = 1, ..., k$. Each of these conditional distributions is easy to sample from due to the bipartite structure of the interaction graph in an RBM. The problem is that block GS tends to mix slowly , so that many iterations are required in order to generate reasonable samples from the distribution . However , if one could just sample $\mathbf{h}_0 \sim p(\mathbf{h})$ from the marginal distribution of the hidden states then one could get a sample of the visible states with a single draw from $p(\mathbf{v}|\mathbf{h}_0)$.

One can rewrite the conditional distribution $p(\mathbf{h}|\mathbf{v})$ as $p(\mathbf{h}|\boldsymbol{\phi}) = Z^{-1}e^{-\boldsymbol{\phi}^T\mathbf{h}}$ in which $\boldsymbol{\phi} = W^T(\mathbf{v} - \boldsymbol{\mu})$ is a field acting on the hidden units that is induced by the visible units. If the dimension of the visible space is large, then $\phi_i = \sum_j W_{ji}(v_j - \mu_j)$ is a weighted sum of random variables and (under some conditions) is approximately Gaussian. Thus, if $\boldsymbol{\mu}$ and Σ are the empirical mean and covariance matrix of the visible units, then $\boldsymbol{\phi} \sim \mathcal{N}(0, W^T\Sigma W)$ is approximately the distribution of the fields under the data distribution:

$$p(\mathbf{v}) \approx \sum_H \int d\boldsymbol{\phi} p(\mathbf{v}|\mathbf{h})p(\mathbf{h}|\boldsymbol{\phi})p(\boldsymbol{\phi}) , \qquad (6)$$

is way to approximate the marginal distribution of the visible units in an RBM using a stochastic directed neural network.

v,h ~p

algorithm for generating approximate samples from an RBM using a single backward pass , which we call "belief generation" due to the resemblance to a DBN.

- Step 1: Draw $\boldsymbol{\phi} \sim \mathcal{N}(0, W^T\Sigma W)$.
- Step 2: Draw $\mathbf{h} \sim Z^{-1}e^{-\boldsymbol{\phi}^T\mathbf{h}}$.
- Step 3: Draw $\mathbf{v} \sim p(\mathbf{v}|\mathbf{h})$ and use $\mathbf{v}$ as a sample from the model distribution.

Note that this can be done without having to compute $W^T\Sigma W$ every time. Instead, store a matrix Q that is a square root of Σ and then set $\boldsymbol{\phi} = W^T Qz$, $z \sim \mathrm{N}(0,1)$. In practice , a few Gibbs sampler updates need to be performed starting from the resulting sample in order to obtain good samples , but maybe good samples aren 't required for training?

To address this question, I trained an RBM on binary MNIST using CD-0. The RBM had 512 hidden units, and was trained for 300 epochs with a batch size of 1024 and a constant learning rate of $1e^{-3}$ using the ADAM optimizer [5]. The weights were randomly initialized from a Gaussian distribution with standard deviation 0.1, and no weight decay was used.

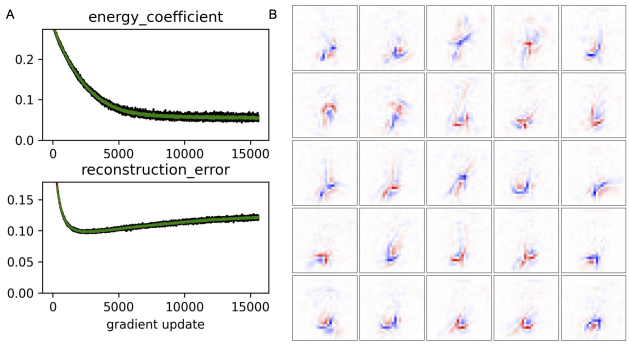

FIG. 1. **Training an RBM with CD-0 on MNIST.** A. Training dynamics. B. A random sample of learned weights at the end of training.

Figure 1A illustrates the training dynamics using two metrics. The first is a normalized statistical distance between observed and sampled images called the energy coefficient [8],

$$\frac{2\langle\langle||v - v'||\rangle_d\rangle_m - \langle\langle||v - v'||\rangle_d\rangle_d - \langle\langle||v - v'||\rangle_m\rangle_m}{2\langle\langle||v - v'||\rangle_d\rangle_m}$$

and the second is the 1-step reconstruction error. Both metrics decrease during training as the model learns reasonable localized features (Figure 1B).

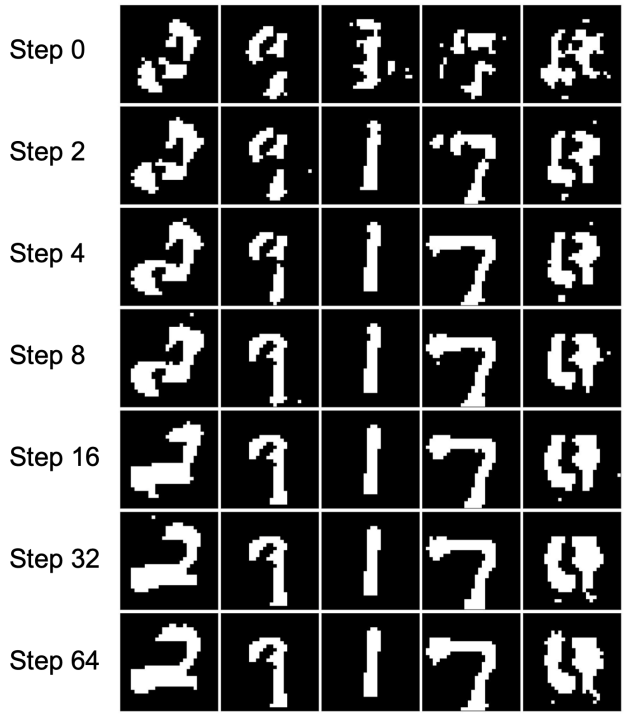

FIG. 2. **Samples from an RBM trained by CD-0.** The initial row (Step 0) presents samples directly obtained from the approximate ssampling algorithm, and each subsequent row illustrates how those samples evolve through block Gibbs sampling.

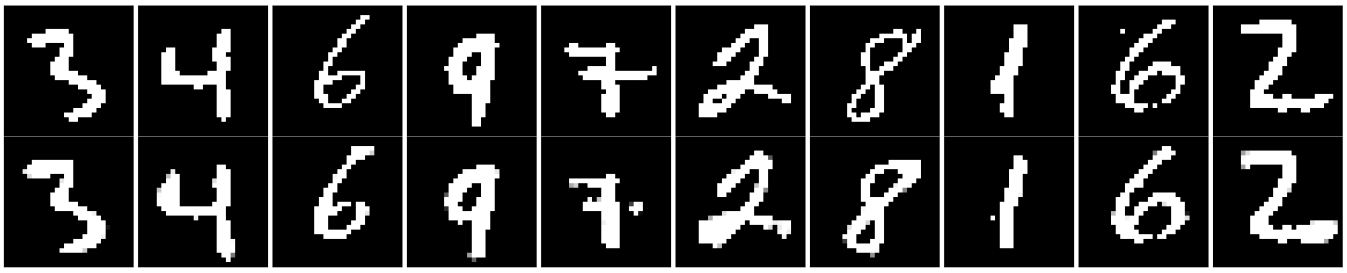

FIG. 3. **Reconstructions using an RBM trained by CD-0.** Original on top, reconstruction on bottom.

Next, Figures 2 shows samples drawn from the model using different numbers of alternating block GS steps. The images in the first row, labeled Step 0, are obtained directly from the belief generation sampling algorithm. These initial samples are updated with different numbers of GS steps, as shown in the subsequent rows.

The samples obtained from the belief generation sampling algorithm are clearly very noisy, but they are in a sense close to realistic handwritten digits. One can view the Gibbs sampling steps as image denoising, much like accessing a memory in a Hopfield network [4, 7]. By initializing CD chains to noisy images, the gradient updates decrease the energy near the observed data points, and increase the energy near the noisy digits to create a basin of attraction so that GS updates move noisy samples closer to realistic samples.

The samples drawn from the CD-0 trained RBM aren't of particularly high quality – they are okay, but not great – but the reconstructions shown in Figure 3 are a different story. The CD-0 trained RBM learns to create faithful reconstructions because the observed samples are encoded in deep minima on the energy landscape.

One difficulties with RBMs is that the energy function pulls double duty; the topography of the energy function encodes patterns as basins of attraction, but the same topography also determines the rate of mixing during MCMC sampling. The model learns the patterns better as the basins of attraction get deeper, but this leads to slow hopping between basins. As a result, MCMC chains need to be extremely long to sample the whole distribution. Belief generation gets around the need to run long MCMC chains to obtain samples from the RBM. Instead, one draws a batch of initial samples that are close to the basins of attraction, then uses GS updates to move towards the energy minima.

As the model trains, the 1-step reconstruction error of the observed samples decreases substantially. However, the reconstruction error is not uniform across sample space. Rather, it decreases with each Gibbs sampling step (Table I) starting from an initial state created with belief generation. Much like a diffusion model, the RBM is iteratively denoising an initial image as it moves down the basin of attraction.

This technical report recasts the process of sampling from an RBM using long Markov Chains into a composition of a simple approximate sampling algorithm followed by a short Markov Chain. This approach may lead to a substantial speedup in training, enabling RBMs to scale to previously intractable problems.

	Step 0	Step 2	Step 4	Step 8	Step 16	Step 32
CD-0	0.35	0.27	0.23	0.21	0.20	0.18

TABLE I. One-step reconstruction error as a function of the number of GS steps away from the initial state.

[1] David H Ackley, Geoffrey E Hinton, and Terrence J Sejnowski. A learning algorithm for boltzmann machines. *Cognitive science*, 9(1):147–169, 1985.

[2] Geoffrey Hinton. A practical guide to training restricted Boltzmann machines. *Momentum*, 9(1):926, 2010.

[3] Geoffrey E Hinton. Training products of experts by minimizing contrastive divergence. *Training*, 14(8), 2006.

[4] John J Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the national academy of sciences*, 79(8):2554–2558, 1982.

[5] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[6] Renjie Liao, Simon Kornblith, Mengye Ren, David J Fleet, and Geoffrey Hinton. Gaussian-bernoulli rbms without tears. *arXiv preprint arXiv:2210.10318*, 2022.

[7] Pankaj Mehta, Marin Bukov, Ching-Hao Wang, Alexandre GR Day, Clint Richardson, Charles K Fisher, and David J Schwab. A high-bias, low-variance introduction to machine learning for physicists. *Physics reports*, 810: 1–124, 2019.

[8] Gábor J Székely. E-statistics: The energy of statistical samples. *Bowling Green State University, Department of Mathematics and Statistics Technical Report*, 3(05):1–18, 2003.

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