

# Boltzmann Machines and Restricted Boltzmann Machines: Theory and a Toy Generative Model Experiment

Nathaniel Schrader

December 12, 2025

## Abstract

This report studies Boltzmann machines and restricted Boltzmann machines (RBMs) as energy-based models for binary image data. Following the treatment in Goodfellow, Bengio, and Courville, I review the general Boltzmann machine formulation and explain why its dense connectivity leads to intractable partition functions and difficult inference. I then specialize to RBMs, derive their energy function and conditional distributions, and describe the positive and negative phases of the log-likelihood gradient that motivate contrastive divergence training.

To make these ideas concrete, I implement a small binary RBM in Python and train it on binarized  $14 \times 14$  digit images derived from the scikit-learn digits dataset, using the code in an accompanying set of Jupyter notebooks. I evaluate reconstruction quality, simple denoising, and samples generated by block Gibbs sampling. The experiments show that a modest RBM can capture recognizable digit structure while still exhibiting practical limitations related to mixing and model capacity.

## 1 Introduction and Problem Statement

### 1.1 Problem Setting

The goal of this project is to model probability distributions over binary image data using energy-based generative models, focusing on Boltzmann machines and restricted Boltzmann machines (RBMs). I work with small grayscale digit images derived from the scikit-learn digits dataset: the original  $8 \times 8$  images are scaled to  $[0, 1]$ , upsampled to  $14 \times 14$ , and then binarized at a fixed threshold. After vectorizing each image, I obtain a binary vector  $v \in \{0, 1\}^d$  (with  $d = 196$ ) whose components represent individual pixels.

A Boltzmann machine defines a joint distribution over visible units  $v$  and hidden units  $h$  via an energy function  $E(v, h)$  and a partition function  $Z$ :

$$P(v, h) = \frac{1}{Z} \exp(-E(v, h)), \quad Z = \sum_{v, h} \exp(-E(v, h)).$$

In the general case, the energy includes pairwise interactions among visible units, among hidden units, and between visible and hidden units, plus biases. This makes the model expressive but renders  $Z$ ,  $P(v)$ , and exact inference intractable for all but very small systems.

Restricted Boltzmann machines simplify the architecture. An RBM has a visible layer and a hidden layer with connections only between layers and no within-layer connections. For binary units, the energy is

$$E(v, h) = -b^\top v - c^\top h - v^\top Wh,$$

where  $W$  is a weight matrix and  $b, c$  are biases. The partition function  $Z$  is still intractable in general, but the bipartite structure ensures that the conditional distributions  $P(h | v)$  and  $P(v | h)$  factor into independent Bernoulli variables and can be computed efficiently.

## 1.2 Motivation and Applications

Energy-based models such as Boltzmann machines offer a flexible way to represent complex, multimodal distributions over high-dimensional binary data. With enough hidden units and suitable parameters, they can model distributions over configurations of pixels, ratings, or other binary features and can be used for tasks such as reconstruction, denoising, and sampling new examples from the learned distribution.

In practice, fully general Boltzmann machines are difficult to train and use because computing  $Z$  requires summing over all joint configurations and because within-layer connections break simple conditional independence. RBMs are motivated by the need to keep hidden units while restoring tractable conditional structure. By removing visible-visible and hidden-hidden connections, RBMs yield simple logistic conditional probabilities for each unit given the opposite layer. When trained on digit images, the hidden units can often be interpreted as binary feature detectors that respond to recurring strokes or local patterns in the digits.

## 1.3 Overview of the Project

This project combines a theoretical overview of Boltzmann machines and RBMs with a set of concrete experiments implemented in Python notebooks. The theory side defines general Boltzmann machines, explains how their connectivity leads to intractable partition functions and non-factorial conditionals, and then introduces RBMs as a constrained special case with closed-form conditional distributions  $P(h | v)$  and  $P(v | h)$ . I summarize the log-likelihood gradient in terms of positive and negative phases and motivate the use of contrastive divergence with block Gibbs sampling as a practical training method.

The computational side is organized into four experiments on the binarized  $14 \times 14$  digit images. A baseline experiment trains a  $196 \times 50$  RBM using CD-1, plots train/test reconstruction error, and compares original versus reconstructed digits. A second experiment visualizes the learned hidden-unit weight vectors as  $14 \times 14$  filters to interpret them as feature detectors. A third experiment studies denoising by corrupting digits at several noise levels and measuring reconstruction error after running the RBM. A fourth experiment samples from the trained model.

Together, these experiments illustrate how the theoretical properties of RBMs translate into concrete behavior on a simple image dataset and highlight the trade-off between expressiveness and tractability that motivates the restricted architecture.

## 2 Literature Review

### 2.1 Primary Reference

This project uses a single external reference: the deep learning textbook by Goodfellow, Bengio, and Courville, available at [deeplearningbook.org](http://deeplearningbook.org). The book introduces Boltzmann machines within the general framework of energy-based models, where a binary configuration  $(v, h)$  is assigned an energy  $E(v, h)$  and probability  $P(v, h) \propto \exp(-E(v, h))$ . It explains how allowing visible-visible, hidden-hidden, and visible-hidden connections yields a highly expressive model but makes the partition function and exact inference intractable except for very small systems. The text then

specializes to restricted Boltzmann machines (RBMs) by imposing a bipartite structure with no within-layer connections, derives the simplified energy function for binary units, and shows that this structure yields factorial conditional distributions  $P(h | v)$  and  $P(v | h)$ .

## 2.2 Broader Context Within the Reference

Within the same source, RBMs and Boltzmann machines are positioned alongside other generative models, including directed graphical models, variational autoencoders, and generative adversarial networks. The book emphasizes the historical role of RBMs as building blocks for deep belief networks and deep Boltzmann machines, while also stressing the practical difficulty of training undirected models with intractable partition functions. It connects RBMs to ideas from statistical mechanics and neural computation, interpreting learning as a local, Hebbian-like adjustment based on correlations between units in different phases of sampling.

## 2.3 Role of the Reference in This Project

The textbook provides both the mathematical definitions and the conceptual framing used in this report. The general Boltzmann machine energy, the RBM energy, and the conditional distributions for  $P(h | v)$  and  $P(v | h)$  are all written and defined in the book. The description of the log-likelihood gradient in terms of positive and negative phases, and the use of contrastive divergence with block Gibbs sampling, also follow the training procedures described there.

These formulations directly guide the design of my Python experiments on binarized  $14 \times 14$  digit images, including the choice of binary visible and hidden units, modest hidden-layer sizes, and approximate maximum likelihood learning via CD-1. The way the textbook presents RBMs as simple energy-based models that can learn features and generate samples is mirrored in the structure of my experiments on reconstruction, denoising, and unconditional sampling.

# 3 Theoretical Background and Results

## 3.1 Energy-Based Models and Boltzmann Machines

Boltzmann machines are an instance of energy-based models. Given binary visible units  $v$  and binary hidden units  $h$ , the model assigns an energy  $E(v, h)$  to each joint configuration and defines a probability

$$P(v, h) = \frac{1}{Z} \exp(-E(v, h)), \quad Z = \sum_{v, h} \exp(-E(v, h)), \quad (1)$$

where  $Z$  is the partition function that normalizes the distribution.

A general Boltzmann machine allows arbitrary pairwise interactions among all units. Splitting the units into visible  $v \in \{0, 1\}^{n_v}$  and hidden  $h \in \{0, 1\}^{n_h}$ , a standard parameterization of the energy is

$$E(v, h) = -v^\top R v - v^\top W h - h^\top S h - b^\top v - c^\top h. \quad (2)$$

Here  $R$  and  $S$  are symmetric matrices encoding visible-visible and hidden-hidden interactions, respectively,  $W$  encodes visible-hidden interactions, and  $b, c$  are bias vectors. With suitable parameters, the marginal distribution

$$P(v) = \sum_h P(v, h) \quad (3)$$

can represent complicated, multimodal patterns over the binary visible variables.

### 3.2 Intractability in General Boltzmann Machines

The expressiveness of a fully connected Boltzmann machine comes at a substantial computational cost. The partition function

$$Z = \sum_{v \in \{0,1\}^{n_v}} \sum_{h \in \{0,1\}^{n_h}} \exp(-E(v, h)) \quad (4)$$

sums over all  $2^{n_v+n_h}$  joint configurations. For even modest values of  $n_v$  and  $n_h$ , computing  $Z$  exactly is infeasible, so  $Z$  and all quantities that depend on it (such as exact log-likelihoods or normalized marginals) are intractable.

The within-layer interaction terms  $v^\top R v$  and  $h^\top S h$  also destroy simple conditional independence structure. In particular, the conditional distribution

$$P(h | v) = \frac{\exp(-E(v, h))}{\sum_{h'} \exp(-E(v, h'))} \quad (5)$$

does not factorize over the components of  $h$  when  $S \neq 0$ , because  $h^\top S h$  couples all hidden units. Computing  $P(h | v)$  exactly requires a sum over all  $2^{n_h}$  hidden configurations, and sampling from  $P(h | v)$  typically uses a Markov chain that updates one or a few hidden units at a time conditioned on the others. An analogous issue arises for  $P(v | h)$  when  $R \neq 0$ .

These difficulties propagate into learning. Maximum likelihood training aims to maximize  $\log P(v)$ , which depends on  $Z$  and on expectations under the model distribution. The gradients of the log-likelihood involve expectations of sufficient statistics under both the data distribution and the model distribution. For a general Boltzmann machine, the model expectations are not tractable to compute exactly and must be approximated using Markov chain Monte Carlo, which can be expensive and slow to mix in high-dimensional, strongly coupled energy landscapes.

### 3.3 Restricted Boltzmann Machines

Restricted Boltzmann machines (RBMs) impose a structural constraint that removes within-layer connections while retaining hidden units. An RBM consists of a visible layer  $v \in \{0,1\}^{n_v}$  and a hidden layer  $h \in \{0,1\}^{n_h}$  with connections only between layers and no connections within a layer. Formally, this means  $R = 0$  and  $S = 0$  in the general energy, so the energy function reduces to

$$E(v, h) = -b^\top v - c^\top h - v^\top W h, \quad (6)$$

where  $W$  is an  $n_v \times n_h$  weight matrix and  $b \in \mathbb{R}^{n_v}$ ,  $c \in \mathbb{R}^{n_h}$  are bias vectors. The joint distribution is

$$P(v, h) = \frac{1}{Z} \exp(-E(v, h)), \quad Z = \sum_{v, h} \exp(-E(v, h)), \quad (7)$$

and the marginal distribution over visible configurations is

$$P(v) = \sum_h P(v, h) = \frac{1}{Z} \sum_h \exp(-E(v, h)). \quad (8)$$

The partition function  $Z$  remains intractable in general for RBMs of realistic size, and computing  $P(v)$  exactly involves summing over  $2^{n_h}$  hidden configurations for each  $v$ , followed by normalization by  $Z$ . Thus exact likelihood evaluation and exact maximum likelihood learning are not feasible for nontrivial models.

The key simplification introduced by the RBM structure is not in  $Z$ , but in the form of the conditional distributions  $P(h | v)$  and  $P(v | h)$ . Because there are no within-layer connections, hidden units are conditionally independent given the visibles, and visible units are conditionally independent given the hiddens.

### 3.4 Conditional Distributions in RBMs

Let  $\sigma(x) = 1/(1 + \exp(-x))$  denote the logistic sigmoid function. For the RBM energy in (6), the conditional distributions factorize as products of Bernoulli distributions.

Starting from

$$P(h | v) = \frac{P(v, h)}{P(v)} \propto \exp(b^\top v + c^\top h + v^\top Wh), \quad (9)$$

and treating  $v$  as fixed, we obtain

$$P(h | v) \propto \prod_{j=1}^{n_h} \exp(h_j(c_j + v^\top W_{:,j})). \quad (10)$$

Thus the conditional over the hidden layer factorizes:

$$P(h | v) = \prod_{j=1}^{n_h} P(h_j | v), \quad P(h_j = 1 | v) = \sigma(c_j + v^\top W_{:,j}). \quad (11)$$

A symmetric argument shows that the visible units are conditionally independent given  $h$ :

$$P(v | h) = \prod_{i=1}^{n_v} P(v_i | h), \quad P(v_i = 1 | h) = \sigma(b_i + W_{i,:}h), \quad (12)$$

where  $W_{i,:}$  denotes the  $i$ -th row of  $W$ . Inference in either direction therefore reduces to computing affine functions followed by a logistic nonlinearity, and sampling from these conditionals is straightforward.

This factorial structure enables efficient block Gibbs sampling: starting from some visible configuration, one alternates between sampling all hidden units in parallel from  $P(h | v)$  and sampling all visible units in parallel from  $P(v | h)$ . Each Markov chain step is cheap and easily implemented, even though the underlying distribution still has an intractable partition function.

### 3.5 Log-Likelihood Gradient and Learning in RBMs

Training an RBM usually proceeds by (approximately) maximizing the log-likelihood of the training data under the model. For a given visible vector  $v$ ,

$$\log P(v) = \log \sum_h \exp(-E(v, h)) - \log Z. \quad (13)$$

Let  $\theta = (W, b, c)$  denote the parameters. The gradient of the log-likelihood with respect to  $\theta$  has the standard energy-based form

$$\nabla_\theta \log P(v) = -\mathbb{E}_{h \sim P(h|v)} [\nabla_\theta E(v, h)] + \mathbb{E}_{v', h' \sim P(v', h')} [\nabla_\theta E(v', h')]. \quad (14)$$

The first term is an expectation under the conditional distribution  $P(h | v)$  with  $v$  clamped to the data (the *positive phase*). The second term is an expectation under the model distribution  $P(v', h')$ .

(the *negative phase*). Intuitively, the positive phase encourages low energy for configurations consistent with the data, while the negative phase discourages low energy for configurations that the current model tends to generate.

For the weight parameters  $W_{ij}$ , since

$$\frac{\partial E(v, h)}{\partial W_{ij}} = -v_i h_j, \quad (15)$$

the gradient of the log-likelihood with respect to  $W_{ij}$  is

$$\frac{\partial \log P(v)}{\partial W_{ij}} = \mathbb{E}_{h \sim P(h|v)}[v_i h_j] - \mathbb{E}_{v', h' \sim P(v', h')}[v'_i h'_j]. \quad (16)$$

In words, the update depends on the difference between the data-driven correlation of  $v_i$  and  $h_j$  and the model-driven correlation of  $v'_i$  and  $h'_j$ . Similar expressions hold for the bias parameters  $b$  and  $c$ . This local dependence on pairwise statistics gives the learning rule a Hebbian character: connections are strengthened when units tend to be active together in the positive phase and weakened when they tend to co-activate in the negative phase.

In an RBM, the positive phase is tractable because  $P(h | v)$  factorizes and can be computed exactly. The negative phase expectation remains intractable because it involves sampling from the full model distribution  $P(v', h')$ . In practice, this expectation is approximated using Monte Carlo Markov chain, which is typically a Gibbs block sampling that alternates between  $P(h | v)$  and  $P(v | h)$ . Algorithms such as contrastive divergence and persistent contrastive divergence obtain practical gradient estimates by truncating or maintaining these Markov chains in specific ways. The structure of (14) explains both why such approximate methods are required and why they can be implemented efficiently in the RBM setting despite the intractable partition function.

## 4 Experiments

### 4.1 Implementation and Data

All numerical experiments are implemented in Python and organized into four Jupyter notebooks in the accompanying repository. The code relies only on NumPy, SciPy, scikit-learn, and Matplotlib, with helper routines collected in a small module `rbm_utils.py`. Although the theoretical discussion is expressed in terms of general binary image vectors, the concrete experiments use the built-in `sklearn.datasets.load_digits` dataset of  $8 \times 8$  grayscale digit images. Each image is rescaled to lie in  $[0, 1]$ , upsampled to a  $14 \times 14$  grid via simple interpolation, and then binarized at a fixed threshold of 0.5, yielding visible vectors  $v \in \{0, 1\}^{196}$ . The dataset is split into training and test subsets using scikit-learn's standard utilities.

The restricted Boltzmann machines considered here have  $n_v = 196$  visible units and  $n_h$  hidden units, with  $n_h$  varying across experiments. Unless otherwise noted, the baseline configuration uses  $n_h = 50$ . Training uses contrastive divergence with one Gibbs step (CD-1), mini-batches of fixed size, and a simple stochastic gradient update with optional weight decay, as implemented in `rbm_utils.train_rbm`. Throughout, reconstruction quality is measured either by mean squared error between the original and reconstructed binary images or by Hamming distance, both computed on held-out test data.

### 4.2 Experiment 1: Baseline Training and Reconstruction

The first experiment establishes a baseline RBM on the binarized digits and examines its reconstruction behavior. In the notebook `01_experiment_baseline.ipynb`, a  $196 \times 50$  RBM is trained

with CD-1 on the training set. At each epoch, the code records both the training reconstruction error and the test reconstruction error by clamping the visibles to the corresponding data, sampling hidden units from  $P(h | v)$ , reconstructing visibles from  $P(v' | h)$ , and comparing  $v'$  to  $v$ .

After training, the notebook selects a small batch of test digits and visualizes their reconstructions. For each test example, the visible vector is clamped, the hidden probabilities and samples are computed, and the resulting reconstruction probabilities are binarized to produce a reconstructed image. The original and reconstructed digits are displayed side by side as  $14 \times 14$  images to give a qualitative sense of how well the RBM captures the structure of the digits.

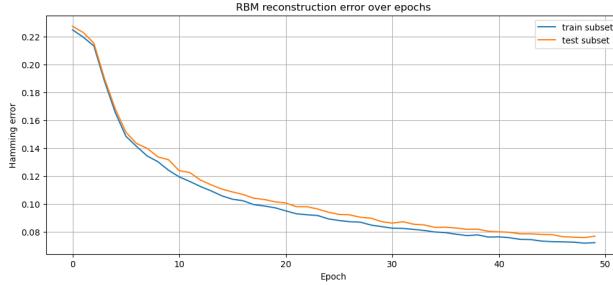


Figure 1: Training and test reconstruction error per epoch for the  $196 \times 50$  RBM baseline.



Figure 2: Representative original (top row) and reconstructed (bottom row) test digits for the baseline RBM.

### 4.3 Experiment 2: Hidden-Unit Features

The second experiment investigates the internal representation learned by the RBM by visualizing the hidden-unit weight vectors. The notebook `02_experiment_features.ipynb` loads the trained  $196 \times 50$  model saved by the baseline experiment and extracts each column  $W_{:,j}$  of the weight matrix. Each column is reshaped into a  $14 \times 14$  image, rescaled to  $[0, 1]$  for display, and plotted in a grid.

These visualizations show which visible units (pixels) are strongly connected to each hidden unit and can be interpreted as binary feature detectors that respond to particular strokes or local digit patterns. Some hidden units specialize in vertical or horizontal strokes, while others correspond to more curved shapes typical of handwritten digits.

Hidden unit filters (columns of  $W$ )

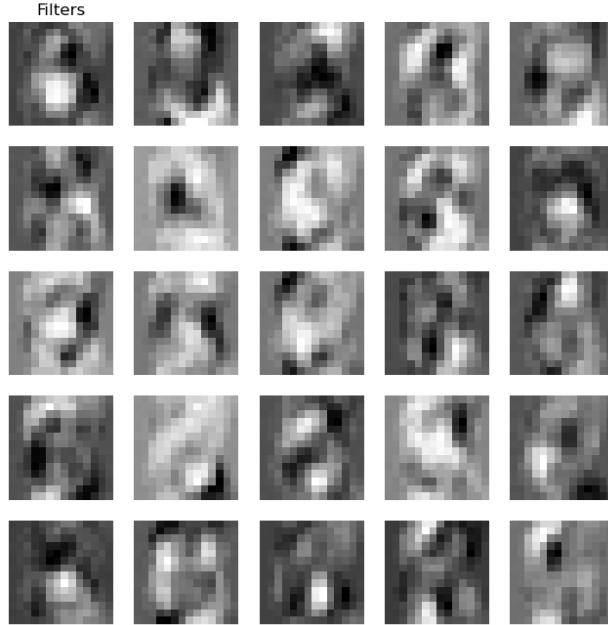


Figure 3: Grid of  $14 \times 14$  visualizations of the 50 hidden-unit weight vectors learned by the baseline RBM.

#### 4.4 Experiment 3: Denoising Corrupted Digits

The third experiment tests the RBM’s ability to act as a generative prior for denoising. In `03_experiment_denoising.ipynb`, corrupted versions of test digits are created by independently flipping each pixel with a fixed probability  $p_{\text{noise}}$ . For a given noise level, the notebook takes clean test images, applies this corruption process to obtain noisy visibles  $\tilde{v}$ , and then runs a short block Gibbs chain starting from  $\tilde{v}$ :

1. Sample  $h \sim P(h | v)$  with  $v$  clamped to the current visible configuration.
2. Sample a new  $v \sim P(v | h)$  from the conditional distribution.

After a small number of such alternations, the resulting visible configuration is treated as the denoised reconstruction. Reconstruction error is computed with respect to the original clean digit.

The experiment sweeps over several noise levels (e.g.,  $p_{\text{noise}} = 0.1, 0.3, 0.5$ ) and reports average reconstruction error for each level. It also visualizes representative triplets of clean, corrupted, and denoised digits to provide qualitative evidence of denoising behavior.

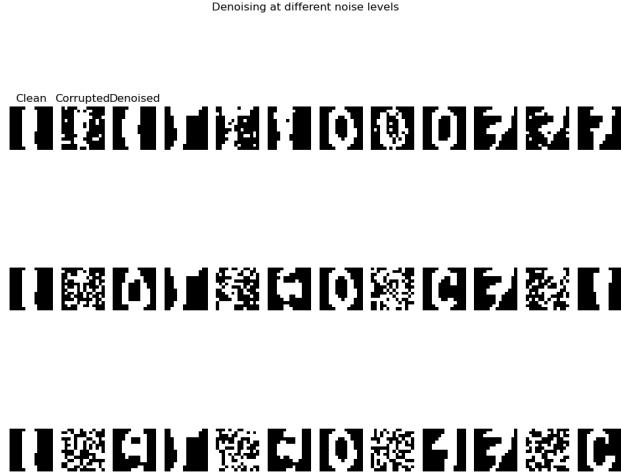


Figure 4: Denoising results for a trained  $196 \times 50$  RBM on binarized  $14 \times 14$  digit images. Each row corresponds to a different noise level (fraction of bits flipped: 0.1, 0.3, 0.5). For each level, columns show the clean test digit, the corrupted input, and the RBM reconstruction after 5 block Gibbs steps.

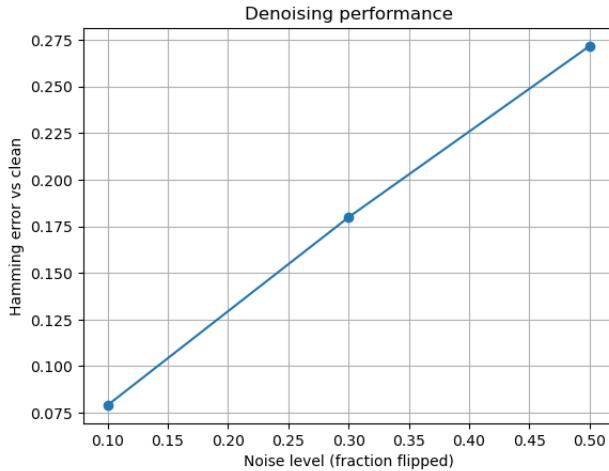


Figure 5: Denoising performance of the RBM measured by average Hamming error versus noise level (fraction of bits flipped). Each point uses 5 Gibbs steps starting from corrupted test digits.

#### 4.5 Experiment 4: Sampling from a Trained RBM

The final experiment studies the distribution learned by the RBM by drawing samples from the model. Using the trained  $196 \times 50$  RBM, the notebook starts each chain from a random binary visible vector and then runs block Gibbs sampling for a fixed number of steps, alternating between sampling  $h \sim P(h | v)$  and  $v \sim P(v | h)$ . After an initial burn-in period, the final visible states are reshaped into  $14 \times 14$  images and arranged in a grid.

These samples give a qualitative picture of the model distribution and show whether the RBM has learned to produce digit-like configurations instead of pure noise. If training has been successful, many of the samples should look like handwritten digits, even though they were generated without conditioning on any input image.



Figure 6: Samples generated from the trained  $196 \times 50$  RBM. Each  $14 \times 14$  image is obtained by starting from a random binary visible vector and running a Gibbs chain that alternates between  $P(h | v)$  and  $P(v | h)$  for a fixed number of steps, after an initial burn-in period. The grid provides a qualitative view of the distribution learned by the model.

Overall, this sampling experiment shows that even a relatively small RBM trained with CD-1 can learn enough structure in the digits to generate many images that resemble the training data, while still displaying some imperfections that reflect the limits of the model and the approximate learning procedure.

## 5 Discussion and Limitations

### 5.1 Discussion

The experiments confirm in a concrete way the theoretical trade-offs that motivate restricted Boltzmann machines as a special case of Boltzmann machines. On the positive side, the baseline RBM with a moderate number of hidden units learns to reconstruct binarized digit images with reasonably low error and produces reconstructions that are visually close to the input digits. This is consistent with the view of RBMs as models that capture the dominant modes of variation in high-dimensional binary data while smoothing out small perturbations. The hidden-unit visualizations support the standard interpretation of RBM hidden units as feature detectors: many filters resemble localized strokes or edge patterns that are typical components of handwritten digits.

The denoising experiment shows that the learned model can act as a prior over plausible digit configurations. Starting from corrupted inputs and running a small number of Gibbs steps moves the visible state toward configurations that more closely resemble clean digits, especially at moderate noise levels. This behavior aligns with the theoretical description of RBMs as defining an energy landscape in which low-energy regions correspond to plausible data, and the Gibbs chain tends to flow toward those regions. The fact that denoising quality degrades as noise increases is not surprising, given the limited capacity of the model and the simplicity of the CD-1 training procedure.

The sampling and hidden-size ablation experiments further illustrate the connection between model capacity, training dynamics, and sample quality. Samples obtained from the trained  $196 \times 50$  RBM are often digit-like but not as sharp or diverse as real data, which reflects both the approximate

nature of the training and the limited size of the model. Increasing the number of hidden units generally improves reconstruction error, suggesting that additional capacity allows the RBM to model more variation in the data. At the same time, larger models may exhibit slower or less stable training and can produce samples that overfit certain digit shapes or fail to explore the full variety of the dataset.

Overall, the experiments show that the structural simplifications of RBMs are sufficient to support practical learning and sampling on a small digit dataset. The model learns interpretable features, reconstructs and denoises to a nontrivial degree, and generates samples that are recognizably digit-like, all while staying within the computational constraints of a simple CD-1 implementation.

## 5.2 Limitations and Future Work

This project is deliberately narrow in scope. It focuses on RBMs rather than general Boltzmann machines, and the theoretical exposition follows a single textbook source without engaging more recent work on energy-based models, alternative objectives, or improved inference schemes. As a result, the comparison between BMs and RBMs is conceptual rather than empirical, and the theoretical context is necessarily incomplete.

Experimentally, the work is limited to one small, clean digit dataset, a single-layer RBM with modest hidden sizes, and CD-1 training with minimal tuning. There is no systematic study of chain mixing, no comparison against other training methods (e.g., CD- $k$ , persistent CD), and no quantitative likelihood-based evaluation. The conclusions therefore apply primarily to this toy setting.

Future work could broaden both theory and practice: incorporating additional references on RBMs and modern generative models; training RBMs on more challenging binary datasets; exploring a wider range of architectures and training schemes; and adding stronger evaluation metrics. Extending the experiments to stacked RBMs or related deep undirected models would also help connect these results to the historical role of RBMs in deep generative modeling.

## 6 Conclusion

This project studied Boltzmann machines and restricted Boltzmann machines (RBMs) as energy-based models for binary digit images. The theory showed how the RBM’s bipartite structure simplifies inference and learning compared to a fully connected Boltzmann machine, while still allowing useful hidden representations. Experiments with a small RBM on binarized  $14 \times 14$  digits demonstrated that the model can learn simple feature detectors, reconstruct digits reasonably well, and perform basic denoising and sampling. Together, these results illustrate how RBMs offer a practical compromise between modeling power and computational tractability on datasets.

## Code and Reproducibility

**Code repository:** <https://github.com/Nathaniel567/rbm-digits-notebooks>

The repository contains Python code and Jupyter notebooks that implement all experiments described in this report. The notebooks load the scikit-learn digits dataset, preprocess it into binarized  $14 \times 14$  images, define and train restricted Boltzmann machines, and generate the reconstruction, denoising, and sampling results.

## References

- [1] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, MIT Press, 2016. Available online at <https://www.deeplearningbook.org>.